ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC BÁCH KHOA

BÁO CÁO BÀI TẬP LỚN
MÔN XÁC SUẤT THỐNG KÊ
NHÓM 11 – LỚP P01

Giảng viên hướng dẫn: Nguyễn Đình Huy

Người thực hiện:

Tên	MSSV	Tỷ lệ đóng góp
Nguyễn Hoàng Hải	2310873	25%
Lưu Minh Đạt	2310658	25%
Đỗ Trọng Khoa	2351043	25%
Phạm Đắc Minh Khoa	2311634	25%

Thành phố Hồ Chí Minh, ngày 27 tháng 11 năm 2024

Mục lục

I. Tổng quan dữ liệu

1. Tổng quan về 3 tệp tin:

- 3 tệp tin chứa dữ liệu về một cửa hàng bán đồ điện tử trực tuyến. Cửa hàng có trữ đồ tại 3 kho hàng nơi mà sẽ giao đồ được đặt tới khách hàng.
- Tệp tin "dirty_data.csv" chứa các thông về người đặt hàng, mã đơn hàng, xử lý đơn hàng, thời gian, ngày tháng, địa điểm,...
- Tệp tin "missing_data.csv" có một vài dữ liệu bị khuyết cần phải xử lý.
- Tệp tin "warehouses.csv" chứa dữ liệu về tọa độ của 3 kho hàng.

2. Các biến chính trong bộ dữ liệu:

- Order_price: Giá gốc của đơn hàng (USD)
- Delivery_charges: Phí giao hàng (USD)
- Coupon_discount: Giảm giá trên đơn hàng (%)
- Order_total: Thành tiền (USD)
- Season: Mùa
- Is_expedited_delivery: Thể hiện xem khách hàng có yêu cầu rằng đơn hàng này thuộc nhóm giao hàng nhanh hay không? (True/False)
- Distance_to_nearest_warehouse: Khoảng cách từ khách hàng tới kho gần nhất (km)
- Is_happy_customer: Thể hiện xem khách hàng có hài lòng với dịch vụ hay không? (True/False)

3. Các bước thực hiện:

- Đọc dữ liệu (Import data)
- Làm sạch dữ liệu (Cleaning data); Xóa bỏ NA (dữ liệu khuyết)
- Làm rõ dữ liệu (Data visualization)
- Chuyển đổi biến (nếu có)
- Thống kê mô tả
- Thống kê suy diễn
- Xây dựng mô hình hồi quy logistic để đánh giá các nhân tố ảnh hưởng đến việc khách hàng hài lòng hay không
- Dự báo khách hàng có hài lòng với đơn hàng hay không?
- Mở rộng

II. Kiến thức nền tảng:

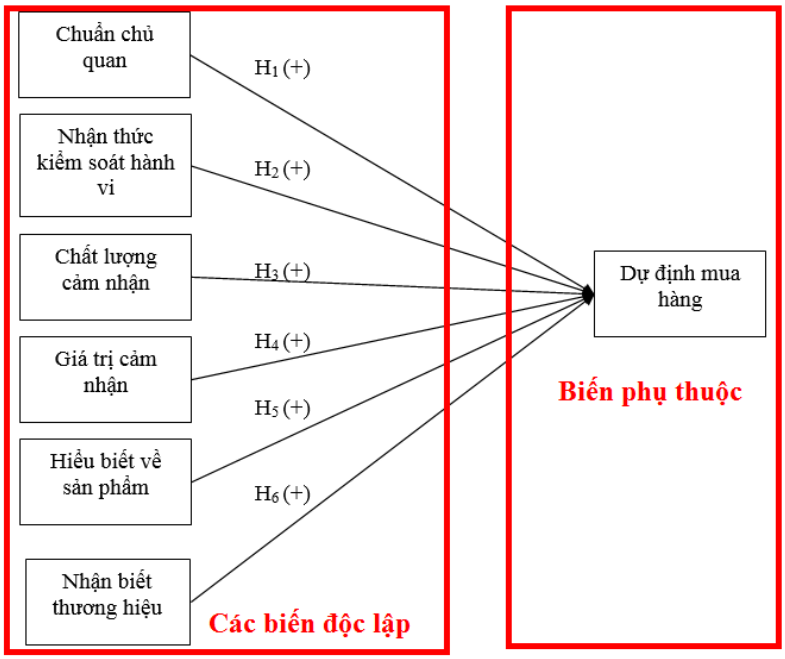
1. Phân tích hồi quy:

1.1. Định nghĩa:

Phân tích hồi quy là một kỹ thuật thống kê nhằm phân tích mối liên hệ phụ thuộc giữa một biến gọi là biến phụ thuộc (biến được giải thích, biến nội sinh) với một hoặc nhiều biến khác gọi là biến độc lập (biến giải thích, biến ngoại sinh, biến hồi quy). Mục tiêu của hồi quy là tìm ra một hàm số thể hiện mối quan hệ này, từ đó có thể dự đoán giá trị của biến phụ thuộc dựa trên các biến độc lập.

Phân tích hồi quy được sử dụng trong một số bối cảnh trong kinh doanh, tài chính và kinh tế. Về kinh tế, nó được sử dụng để giúp các nhà quản lý đầu tư định giá tài sản và hiểu mối quan hệ giữa các yếu tố như giá cả hàng hóa và cổ phiếu của các doanh nghiệp kinh doanh những mặt hàng đó.

Ví dụ: "dự định mua điện thoại hãng A". Việc "dự định mua điện thoại" là biến phụ thuộc. Các yếu tố độc lập gồm: chuẩn chủ quan, nhận thức kiểm soát hành vi, chất lượng cảm nhận, giá trị cảm nhận, hiểu biết về sản phẩm, nhận biết thương hiệu.



1.2. Một số khái niệm cơ bản:

1.2.1. Các thành phần của hồi quy:

- **Biến phụ thuộc (dependent variable):**

Biến phụ thuộc là biến chịu ảnh hưởng của biến khác trong một mô hình (cần dự đoán hoặc mô hình hóa). Thường là kết quả của việc biến độc lập thay đổi.

Ví dụ, nhu cầu về một hang hoá bị ảnh hưởng bởi giá cả của nó.

- **Biến độc lập (independent variable / regressor(s)):**

Biến độc lập là biến tác động tới biến khác trong một mô hình (được sử dụng để dự đoán giá trị của biến phụ thuộc). Biến này được coi là nguyên nhân hoặc yếu tố tác động đến biến phụ thuộc.

Chẳng hạn, giá hàng hoá là biến số độc lập ảnh hưởng tới lượng cầu về nó.

- **Tham số hồi quy:**

Là các tham số trong mô hình hồi quy, biểu thị mức độ và hướng của mối quan hệ giữa biến phụ thuộc và các biến độc lập.

Chẳng hạn, nếu hệ số hồi quy dương, thì khi biến độc lập tăng, biến phụ thuộc cũng có xu hướng tăng.

- **Sai số:**

Là phần chênh lệch giữa giá trị thực tế của biến phụ thuộc so với giá trị dự đoán từ mô hình hồi quy.
Sai số cho thấy mức độ chính xác của mô hình.

1.2.2. Các loại hồi quy cơ bản:

- **Hồi quy đơn:** $y = \alpha + \beta x + \varepsilon$

Trong đó: y : Biến phụ thuộc; α : Hệ số chặn; ε : Sai số; β : Hệ số hồi quy, biểu thị độ dốc của đường hồi quy; x : Biến độc lập

- **Hồi quy bội:** $y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_n x_n + \varepsilon$

Trong đó: y: Biến phụ thuộc; ε: Sai số; $\beta_0, \beta_1, \beta_2, \ldots, \beta_n$ là các hệ số hồi quy và $x_1, x_2, \ldots, x_n$ là các biến độc lập

2. Hồi quy logistic và logistic nhị phân:

2.1. Hồi quy logistic:

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

Ví dụ: giả sử bạn muốn đoán xem khách truy cập trang web của bạn sẽ nhấp vào nút thanh toán trong giỏ hàng của họ hay không. Phân tích hồi quy logistic xem xét hành vi của khách truy cập trước đây, chẳng hạn như thời gian dành cho trang web và số lượng các mặt hàng trong giỏ hàng. Quá trình phân tích này xác định rằng, trước đây, nếu khách truy cập dành hơn năm phút trên trang web và thêm hơn ba mặt hàng vào giỏ hàng, họ sẽ nhấp vào nút thanh toán. Nhờ vào thông tin này, sau đó, hàm hồi quy logistic có thể dự đoán hành vi của một khách mới truy cập trang web.

2.2. Hồi quy Binary logistic:

2.2.1. Lý thuyết:

Hồi quy Binary logistic là mô hình phổ biến trong nghiên cứu dùng để ước lượng xác suất một sự kiện sẽ xảy ra. Đặc trưng của hồi quy nhị phân là biến phụ thuộc chỉ có hai giá trị: 0 và 1. Trên thực tế, có rất nhiều hiện tượng tự nhiên, hiện tượng kinh tế, xã hội,… mà chúng ta cần dự đoán khả năng xảy ra của nó như chiến dịch quảng cáo có được chấp nhận hay không, người vay có trả được nợ hay không, công ty có phá sản hay không, khách hàng có mua hay không,… Những biến nghiên cứu có hai biểu hiện như vậy được mã hóa thành hai giá trị 0 và 1, được gọi là biến nhị phân. Khi biến phụ thuộc ở dạng nhị phân, chúng ta không thể phân tích với dạng hồi quy tuyến tính thông thường vì mô hình sẽ vi phạm các giả định hồi quy. Các giả định quan trọng này bị vi phạm sẽ làm mất hiệu lực thống kê của các kiểm định trong hồi quy, dẫn đến kết quả ước lượng không còn chính xác. Trong khi đó, hồi quy Binary logistic lại không cần thiết phải thỏa mãn các giả định này.

2.2.2. Phương trình:

Thay vì chúng ta ước lượng giá trị của biến phụ thuộc Y theo biến độc lập X như ở hồi quy đa biến, thì trong hồi quy Binary logistic, chúng ta sẽ ước lượng xác suất xảy ra sự kiện Y(probability) khi biết giá trị X. Biến phụ thuộc Y có hai giá trị 0 và 1, với 0 là không xảy ra sự kiện và 1 là xảy ra sự kiện. Từ đặc điểm này, chúng ta có thể đánh giá được khả năng xảy ra sự kiện (Y=1) nếu xác suất dự đoán lớn hơn 0.5, ngược lại, khả năng không xảy ra sự kiện (Y = 0) nếu xác suất dự đoán nhỏ hơn 0.5. Ta có hàm xác suất như sau:

$$P_i = P(Y = 1) = E(Y = 1|X) = \frac{e^{-(\beta_0+\beta_1 X_{1i}+\beta_2 X_{2i}+\beta_3 X_{3i}+\ldots+\beta_k X_{ki})}}{1+e^{-(\beta_0+\beta_1 X_{1i}+\beta_2 X_{2i}+\beta_3 X_{3i}+\ldots+\beta_k X_{ki}+)}}$$

Trong đó P_i là xác suất xảy ra sự kiện. Thực hiện các phép chuyển đổi toán học, ta thu được hàm hồi quy mới như sau:

$$\log\left(\frac{P_i}{1+P_i}\right) = \beta_0 + \beta_1 X_{1i} + \beta_2 X_{2i} + \beta_3 X_{3i} + \ldots + \beta_k X_{ki}$$

Trong đó: P_i : xác suất xảy ra sự kiện (Y=1); $1 - P_i$ xác suất xảy ra sự kiện (Y=0)

Lí do hàm hồi quy logistic có dạng như trên là vì giá trị xác suất bị ràng buộc trong phạm vi từ 0 đến 1. Trong khi đó, hàm hồi quy tuyến tính có giá trị từ âm vô cùng đến dương vô cùng. Và đối với các giá trị của hàm nằm ngoài khoảng (0,1) thì sự diễn giải theo mô hình hồi quy logistic không mang ý nghĩa thống kê.

Để giải quyết vấn đề này, có hai bước tiếp cận thông qua chúng ta thực hiện hai biến đổi:

- Thứ nhất, chúng ta sử dụng một chỉ số thống kê quan trọng đó là Odds:

$$Odd = \frac{P}{1-P}$$

Trong đó, Odd được định nghĩa là tỉ số của hai xác suất. Nếu p là xác suất mắc bệnh, thì (1 - p) là xác suất sự kiện không mắc bệnh. Như vậy, giá trị của Odd có thể lớn hơn 1 và khi xác suất P tiến dần tới 1 thì đồng thời Odd cũng tiến ra vô cùng. Giá trị của Odd bây giờ nằm trong khoảng từ 0 đến vô cùng, vì vậy mô hình hồi quy vẫn còn hạn chế vì Odd bị giới hạn bởi 0.

Để khắc phục điều này, ta thực hiện phép biến đổi tiếp theo, trình bày công thức của Odd ở trên theo logarit của Odd:

$$\text{Log(Odd)} = \beta_0 + \beta_1 X_{1i} + \beta_2 X_{2i} + \beta_3 X_{3i} + \ldots + \beta_k X_{ki}$$

Từ đây việc hoàn chuyển từ P sang Odd và từ Odd sang log(Odd) làm cho vế phải của phương trình có dạng của hàm tuyến tính đa biến và giá trị của log(Odd) không bị giới hạn bởi cận trên, cận dưới nên việc phân tích các tham số dựa vào tích chất của hàm hồi quy đa biến sẽ dễ dàng hơn.

3. Phân tích phương sai (Anova):

"**ANalysis Of VAriance (ANOVA)**" (Phân tích phương sai), là một phương pháp thống kê dùng để so sánh sự khác biệt giữa các nhóm hoặc tập hợp dữ liệu. Dựa trên các trị trung bình của các mẫu quan sát từ các nhóm và thông qua kiểm định giả thuyết của kết luận về sự bằng nhau của các trung bình tổng thể này

3.1. Phân tích phương sai 1 mẫu:

3.1.1. Tổng quan:

Là kiểm định sự khác biệt của trung bình giữa các nhóm dựa trên một yếu tố duy nhất.

Các giả định mô hình:

- Tổng thể có phân phối chuẩn.
- Tổng thể có phương sai bằng nhau.
- Mẫu được chọn là ngẫu nhiên và độc lập.
- Giả sử trung bình của tổng thể là $\mu_i, (i = 1,2,3, \ldots)$ thì
  - ✓ Giả thuyết : "Không có sự khác biệt về trung bình giữa các nhóm" :

    $H_0: \mu_1 = \mu_2 = \cdots = \mu_k$
  - ✓ Đối thuyết : "Có tồn tại sự khác biệt giữa các nhóm "

    $H_1: \mu_1 \neq \mu_k$

3.1.2. Các bước thực hiện:

- Xét sự biến thiên giữa các nhóm để kiểm tra sự bằng nhau của trung bình giữa các nhóm:

  $SST = SSW + SSB$

Trong đó:

$SST = \sum_{i=1}^{k}\sum_{j=1}^{n}(y_{ij}^2 - \frac{y_{..}^2}{N})$: Tổng bình phương toàn phần

SSW: Tổng bình phương bên trong các nhóm

$SSB = \sum_{i=1}^{k}(\frac{y_i^2}{n} - \frac{y_{..}^2}{N})$: Tổng bình phương giữa các nhóm

- Trung bình bình phương toàn phần : $MST = \frac{SST}{kn-1}$
- Trung bình bình phương trong từng nhóm : $MSW = \frac{SSW}{k(n-1)}$
- Trung bình bình phương giữa các nhóm : $MSB = \frac{SSB}{k-1}$
- Thống kê kiểm định : $F = \frac{MSB}{MSW}$
- Bảng Anova 1 nhân tố:

Nguồn của sự biến thiên	SS	df	MS	F
Giữa các nhóm	SSB	$k-1$	MSB	$F = \frac{MSB}{MSW}$
Trong từng nhóm	SSW	$k(n-1)$	MSW	
Tổng	SST	$kn-1$		

- Bác bỏ H_0khi: $f > f_{\alpha;k-1,k(n-1)}$

3.2. Phân tích phương sai 2 mẫu (Anova 2 nhân tố):

3.2.1. Tổng quan:

ANOVA 2 nhân tố dùng để nghiên cứu tác động :

- 2 nhân tố được quan tam trên 1 biến phụ thuộc
- Tương tác giữa các mức khác nhau của 2 nhân tố
- Các giả định mô hình :
  - Tổng thể có phân phối chuẩn
  - Tổng thể có phương sai bằng nhau
  - Mẫu ngẫu nhiên được chọn độc lập

3.2.2. Các bước thực hiện:

Tính tổng các bình phương: $SST = SSG + SSB + SSE$

Trong đó :

- Tổng bình phương:
  - Toàn phần : $SST = \sum_{i=1}^{a}\sum_{j=1}^{b}(y_{ij} - \overline{y})^2$
  - Giữa các nhóm : $SSG = b\sum_{i=1}^{a}(\overline{y_i} - \overline{y})^2$
  - Giữa các khối: $SSB = a\sum_{j=1}^{b}(\overline{y_j} - \overline{y})^2$
  - Sai số : $SSE = \sum_{i=1}^{a}\sum_{j=1}^{b}(y_{ij} - \overline{y_j} - \overline{y_i} + \overline{y})^2$
- Các trung bình bình phương:
  - $MST = \frac{SST}{ab-1}$
  - $MSG = \frac{SSG}{a-1}$
  - $MSB = \frac{SSB}{b-1}$
  - $MSE = \frac{SSE}{(a-1)(b-1)}$
- Đối với các nhóm nhân tố A:

$$\begin{cases} H_{0a}: \alpha_1 = \alpha_2 = \cdots = \alpha_k \\ H_{1a}: \alpha_1 \neq \alpha_k, (k\ bất\ kì) \end{cases}$$

Thống kê kiểm định : $F_a = \frac{MSG}{MSE}$

Bác bỏ H_{0a}khi: $f_{0a} > f_{a;a-1,(a-1)(b-1)}$

- Đối với các nhóm nhân tố B:

$$\begin{cases} H_{0b}: \beta_1 = \beta_2 = \cdots = \beta_k \\ H_{1b}: \beta_1 \neq \beta_k, (k\ bất\ kì) \end{cases}$$

Thống kê kiểm định : $F_b = \frac{MSB}{MSE}$

Bác bỏ H_{0a}khi: $f_{0b} > f_{a;b-1,(a-1)(b-1)}$

III. Tiền xử lý dữ liệu:

1. Cài thư viện cần thiết:

```
> install.packages("dplyr")
> install.packages("geosphere")
> install.packages("readxl")
> install.packages("ggplot2")
> install.packages("ggpubr")
> install.packages("ROCR")
```

2. Gọi thư viện cần xài:

```
> library(carData)
> library(car)
> library(tools)
> library(dplyr)
> library(geosphere)
> library(readxl)
> library(ggplot2)
> library(ggpubr)
> library(ROCR)
```

3. Nhiệm vụ 1: Xử lý dữ liệu file dirty_data:

3.1. Đọc dữ liệu:

```
> dirty_data = read.csv("./dirty_data.csv")
> View(dirty_data)
```

order_id	customer_id	date	nearest_warehouse	shopping_cart
ORD182494	ID6197211592	2019-06-22	Thompson	[('Lucent 330S', 1), ('Thunder line', 2), ('iStrea
ORD395518	ID0282825849	2019-12-29	Thompson	[('Thunder line', 1), ('Universe Note', 2)]
ORD494479	ID0579391891	2019-03-02	Nickolson	[('Thunder line', 1), ('pearTV', 2)]
ORD019224	ID4544561904	2019-01-12	Nickolson	[('Universe Note', 1), ('Alcon 10', 2), ('Olivia x
ORD104032	ID6231506320	2019-11-28	Nickolson	[('Universe Note', 1), ('Olivia x460', 1), ('iStre
ORD146760	ID0311654900	2019-09-16	Bakers	[('Thunder line', 2), ('Universe Note', 1)]
ORD337984	ID3394768956	2019-09-14	Thompson	[('Candle Inferno', 1), ('Alcon 10', 1), ('Toshik
ORD072312	ID0774517121	2019-05-23	Thompson	[('Universe Note', 1), ('Thunder line', 2), ('iSt
ORD377837	ID4769265355	2019-10-09	Bakers	[('Alcon 10', 2), ('Thunder line', 1), ('Candle l
ORD462194	ID5301568579	2019-03-21	Thompson	[('Universe Note', 1), ('Lucent 330S', 1), ('Tos
ORD034800	ID4283908179	2019-08-03	Bakers	[('Alcon 10', 2), ('pearTV', 2), ('iStream', 1), ('(
ORD361636	ID0589500304	2019-12-05	Nickolson	[('Lucent 330S', 1), ('pearTV', 2)]
ORD124395	ID0702352304	2019-02-11	Thompson	[('Alcon 10', 1), ('Universe Note', 1), ('pearTv
ORD255642	ID3085953531	2019-12-24	Nickolson	[('iAssist Line', 2), ('Alcon 10', 1), ('pearTV', 1
ORD496722	ID0589449820	2019-04-09	Nickolson	[('pearTV', 2), ('iStream', 1), ('Lucent 330S', 1
ORD449130	ID0356449717	2019-05-17	Bakers	[('Toshika 750', 2), ('Alcon 10', 1), ('Thunder l
ORD036056	ID0767733196	2019-08-10	Nickolson	[('Alcon 10', 1), ('Olivia x460', 2), ('Lucent 33(
ORD428910	ID2180614753	2019-07-15	Nickolson	[('Olivia x460', 1), ('Candle Inferno', 1)]

Hình 1. Bảng dữ liệu "dirty_data" sau khi được đọc.

3.2. Làm sạch dữ liệu:

3.2.1. Tạo một tệp con bao gồm một số biến cần phân tích:

> cleaning_data = dirty_data[,c("nearest_warehouse","order_price","delivery_charges","customer_lat","customer_long","coupon_discount","order_total","season","is_expedited_delivery","distance_to_nearest_warehouse","is_happy_customer")]

3.2.2. Kiểm tra dữ liệu khuyết:

> apply(is.na(cleaning_data),2,which)

→ integer(0)

➔ Tệp không có dữ liệu khuyết

3.2.3. Làm sạch các dữ liệu có trong một số cột có kiểu dữ liệu character (chr):

> str(cleaning_data)

```
'data.frame':   500 obs. of  11 variables:
 $ nearest_warehouse           : chr  "Thompson" "Thompson" "Nickolson" "N
ickolson" ...
 $ order_price                 : int  12200 9080 10670 24800 9145 7810 137
00 7960 25390 13320 ...
 $ delivery_charges            : num  79.9 62.7 65.9 57.6 75.5 ...
 $ customer_lat                : num  -37.8 -37.8 -37.8 -37.8 37.8 ...
 $ customer_long               : num  145 145 145 145 145 ...
 $ coupon_discount             : int  10 0 10 15 25 10 5 5 10 15 ...
 $ order_total                 : num  11060 9143 9669 21138 6934 ...
 $ season                      : chr  "Winter" "Summer" "Autumn" "Summer"
...
 $ is_expedited_delivery       : chr  "True" "False" "False" "False" ...
 $ distance_to_nearest_warehouse: num  1.28 1.162 1.095 0.857 0.587 ...
 $ is_happy_customer           : chr  "True" "False" "True" "False" ...
```

Hình 2. Dữ liệu mà hàm str() xuất ra

nearest_warehouse, season (không làm sạch cột is_expedited_delivery và is_happy_customer bởi vì có logical mặc dù kiểu dữ liệu chr.)

- nearest_warehouse:

> table(cleaning_data$nearest_warehouse, useNA = "always")

➔ Không có dữ liệu khuyết nhưng có một vài lỗi lộn xộn

```
  Bakers nickolson Nickolson  thompson  Thompson      <NA>
     119         3       181         4       193         0
```

Hình 3.1. Output của hàm table() trước khi làm sạch của cột nearest_warehouse

⇔ Dùng hàm toTitleCase() để tái định nghĩa và đồng nhất cột

> cleaning_data$nearest_warehouse = toTitleCase(cleaning_data$nearest_warehouse)

> table(cleaning_data$nearest_warehouse, useNA = "always")

```
  Bakers Nickolson  Thompson      <NA>
     119       184       197         0
```

Hình 3.2. Output của hàm table() sau khi làm sạch của cột nearest_warehouse

- season:

Tương tự như của cột nearest_warehouse

> cleaning_data$season = toTitleCase(cleaning_data$season)

```
> table(cleaning_data$season, useNA = "always")

Autumn Spring Summer Winter   <NA>
   127    134    124    115      0
```

Hình 4. Output của hàm table() sau khi làm sạch của cột season

- Tạo tệp cleaned_data sau khi xử lý xong dữ liệu và xóa cleaning_data.

```
> cleaned_data = cleaning_data
> rm(cleaning_data)
> write.csv(cleaned_data, file = "./cleaned_data_from_dirty_data.csv", row.names = FALSE)
```

4. Nhiệm vụ 2: Xử lý dữ liệu trong file missing_data:

```
> missing_data = read.csv("./missing_data.csv")
```

- date & season

Xây dựng hàm thời gian để suy ra các ô mùa (season) còn trống: nhận thấy thời gian của cột date với season chưa tương ứng nhau, vì vậy cần xử lí đồng thời ô dữ liệu trống và ô mùa chưa phù hợp với date.

```
> as.Date(missing_data$date)
> missing_data = missing_data %>%
+ mutate(date_onset = as.Date(date, format = "%Y/%m/%d"))
> fix_season = function(date) {
+   month = as.POSIXlt(date)$mon +1
+   if (month %in% c(1, 2, 3)) {
+     return("Spring")
+   } else if (month %in% c(4, 5, 6)) {
+     return("Summer")
+   } else if (month %in% c(7, 8, 9)) {
+     return("Autumn")
+   } else {
+     return("Winter")
+   }
+ }
> missing_data$season = sapply(missing_data$date, fix_season)
```

➔ Sau khi xử lý, các ô mùa bị thiếu đã được điền đầy đủ và những ô mùa chưa đúng cũng được sửa.

- nearest_warehouse

```
> warehouses = read.csv("./warehouses.csv")
```

Xây dựng hàm tìm địa chỉ kho gần nhất: sử dụng hàm distVincentySphere trong package geosphere để tính khoảng cách giữa khách hàng và các kho và trả về tên kho gần nhất.

```
> find_nearest_warehouse = function(customer, warehouses) {
+     distances = geosphere::distVincentySphere(
+         cbind(customer["customer_long"], customer["customer_lat"]), cbind(warehouses$lon, warehouses$lat)
+     )
+     nearest_warehouse = warehouses[which.min(distances), ]
+     return(nearest_warehouse$names)
+ }
> empty_rows = missing_data$nearest_warehouse == ""
> missing_data$nearest_warehouse[empty_rows] = apply(missing_data[empty_rows, c("customer_long", "customer_lat")], 1, function(row) find_nearest_warehouse(row, warehouses))
```

➔ Những ô trống trong cột nearest_warehouse đã được điền đầy đủ.

- is_happy_customer

```
> get_sentiment = function(review) {
+     positive_keywords <- c("good", "excellent", "love", "like", "amazing", "happy", "recommend","great", "nice")
+     return(any(grepl(paste(positive_keywords, collapse = "|"), tolower(review))))
+ }
> missing_data$is_happy_customer = ifelse(missing_data$is_happy_customer == "", sapply(missing_data$latest_customer_review, get_sentiment), missing_data$is_happy_customer)
```

Lúc này, những ô dữ liệu trống trong cột biến is_happy_customer đã được gán đầy đủ nhưng chưa đồng nhất về cách viết do trong file dữ liệu là True/False còn hàm get_setiment mặc định trả về TRUE/FALSE, vì vậy ta cần thêm một bước để đồng nhất cột dữ liệu này.

```
> missing_data$is_happy_customer[missing_data$is_happy_customer == 'TRUE'] = "True"
> missing_data$is_happy_customer[missing_data$is_happy_customer == 'FALSE'] = "False"
```

- order_price

Công thức: order_price = (order_total - delivery_charges) / (1 - coupon_discount / 100).

```
> get_price = function(total, discount, deli){
+     price= (total-deli)/(1-discount/100)
+     return(price)
+ }
> missing_data$order_price[is.na(missing_data$order_price)] = get_price(missing_data$order_total[is.na(missing_data$order_price)],missing_data$coupon_discount[is.na(missing_data$order_price)],missing_data$delivery_charges[is.na(missing_data$order_price)])
```

- order_total

Công thức: order_total = order_price*(1-coupon_discount/100)+delivery_charges

```
> get_total = function(price, discount, deli){
+     total= price*(1-discount/100)+deli
+     return(total)
+ }
> missing_data$order_total[is.na(missing_data$order_total)] = get_total(missing_data$order_price[is.na(missing_data$order_total)],missing_data$coupon_discount[is.na(missing_data$order_total)],missing_data$delivery_charges[is.na(missing_data$order_total)])
```

- distance_to_nearest_warehouse

Dựa trên hàm tìm nearest_warehouse để xây dựng nên hàm tìm distance_to_nearest_warehouse.

```
> get_distance = function(customer, warehouses){
+     distances = geosphere::distVincentySphere(
+         cbind(customer["customer_long"], customer["customer_lat"]), cbind(warehouses$lon, warehouses$lat)
+     )
+     distance = min(distances)
+     return(distance/1000)
+ }
> missing_data$distance_to_nearest_warehouse[is.na(missing_data$distance_to_nearest_warehouse)] = apply(missing_data[is.na(missing_data$distance_to_nearest_warehouse), c("customer_long", "customer_lat")], 1, function(row2) get_distance(row2, warehouses))
```

- NA Values

Những ô NA còn lại là những ô dữ liệu không có giá trị thống kê nên ta có thể xóa những hàng chứa dữ liệu NA mà không ảnh hưởng kết quả. Cột date_onset cũng có giá trị NA nên xóa cả cột.

```
> na_count = colSums(is.na(missing_data))
> na_count
```

```
                order_id                  customer_id
                       0                            0
                    date              nearest_warehouse
                       0                            0
           shopping_cart                  order_price
                       0                            0
         delivery_charges                 customer_lat
                       0                           10
            customer_long               coupon_discount
                      10                            0
             order_total                       season
                       0                            0
 is_expedited_delivery  distance_to_nearest_warehouse
                       0                            0
 latest_customer_review             is_happy_customer
                       0                            0
             date_onset
                     500
```

Hình 5. Bảng dữ liệu tóm tắt của missing_data

```
> missing_data = missing_data %>%
+     select(-date_onset)
> missing_data = na.omit(missing_data)
```

> na_count = colSums(is.na(missing_data))

> na_count

```
                order_id                  customer_id 
                       0                            0 
                    date             nearest_warehouse 
                       0                            0 
           shopping_cart                  order_price 
                       0                            0 
         delivery_charges                 customer_lat 
                       0                            0 
            customer_long              coupon_discount 
                       0                            0 
             order_total                       season 
                       0                            0 
is_expedited_delivery  distance_to_nearest_warehouse 
                       0                            0 
  latest_customer_review            is_happy_customer 
                       0                            0 
```

Hình 6. Số lượng giá trị NA sau khi đã làm sạch

➔ Nhận thấy rằng file missing_data đã được xử lý thành công.

- Ghi ra file dữ liệu để sao lưu

> write.csv(missing_data, file = "cleaned_data_from_missing_data.csv", row.names = FALSE)

IV. Thống kê mô tả:

1. Chuyển đổi kiểu dữ liệu:

- Tạo một tệp new_data mới bao gồm các biến cần phân tích

```
> new_data = missing_data[,c("nearest_warehouse", "order_price", "delivery_charges", "customer_lat", "customer_long", "coupon_discount", "order_total", "season", "is_expedited_delivery", "distance_to_nearest_warehouse", "is_happy_customer")]
```

- Trong RStudio, kiểu kí tự không có tác dụng trong thống kê, vì vậy cần chuyển đổi những cột có dữ liệu kiểu ký tự (nearest_warehouse, season) sang kiểu Factor. Đối với biến chỉ có giá trị False/True như is_expedited_delivery và is_happy_customer, thay False = 0, True = 1.

```
> new_data$is_expedited_delivery[new_data$is_expedited_delivery == 'True'] = 1
> new_data$is_expedited_delivery[new_data$is_expedited_delivery == 'False'] = 0
> new_data$is_expedited_delivery = as.integer(new_data$is_expedited_delivery)
> new_data$is_happy_customer[new_data$is_happy_customer == 'True'] = 1
> new_data$is_happy_customer[new_data$is_happy_customer == 'False'] = 0
> new_data$is_happy_customer = as.integer(new_data$is_happy_customer)
> new_data$is_happy_customer = as.integer(new_data$is_happy_customer)
```

- Ngoài ra, cần phải sắp xếp trật tự các bậc dữ liệu trong cột season và nearest_warehouse

```
> new_data$season = factor((new_data$season), levels = c("Spring","Summer","Autumn","Winter"))
> levels(new_data$season)
[1] "Spring" "Summer" "Autumn" "Winter"
> new_data$nearest_warehouse = factor((new_data$nearest_warehouse), levels = c("Bakers", "Nickolson", "Thompson"))
> levels(new_data$nearest_warehouse)
[1] "Bakers"    "Nickolson" "Thompson"
```

2. Phân tích thống kê mô tả:

2.1. Tổng quát:

```
> summary(new_data)
```

```
nearest_warehouse  order_price    delivery_charges  customer_lat    customer_long   coupon_discount  order_total
Bakers   :105     Min.   :  580   Min.   : 46.20   Min.   :-37.83   Min.   :144.9   Min.   : 0.00   Min.   :  568.6
Nickolson:173     1st Qu.: 7012   1st Qu.: 67.14   1st Qu.:-37.82   1st Qu.:145.0   1st Qu.: 5.00   1st Qu.: 6271.4
Thompson :202     Median :12220   Median : 77.39   Median :-37.81   Median :145.0   Median :10.00   Median :10687.6
                  Mean   :13242   Mean   : 77.84   Mean   :-37.81   Mean   :145.0   Mean   :11.21   Mean   :11863.7
                  3rd Qu.:18323   3rd Qu.: 85.41   3rd Qu.:-37.81   3rd Qu.:145.0   3rd Qu.:15.00   3rd Qu.:16231.7
                  Max.   :37300   Max.   :110.99   Max.   :-37.79   Max.   :145.0   Max.   :25.00   Max.   :37362.5
   season    is_expedited_delivery distance_to_nearest_warehouse is_happy_customer
Spring:110   Min.   :0.0000        Min.   :0.0549                Min.   :0.0000
Summer:105   1st Qu.:0.0000        1st Qu.:0.7218                1st Qu.:1.0000
Autumn:124   Median :1.0000        Median :1.0433                Median :1.0000
Winter:141   Mean   :0.5021        Mean   :1.0767                Mean   :0.7729
             3rd Qu.:1.0000        3rd Qu.:1.3924                3rd Qu.:1.0000
             Max.   :1.0000        Max.   :3.1388                Max.   :1.0000
```

Hình 7. Bảng tóm tắt của new_data sau khi chuyển đổi kiểu dữ liệu

Đối với biến định lượng, hàm summary() cho biết các giá trị đặc trưng của mẫu như giá trị max, min, mean, tứ phân vị thứ nhất và phần tư, trung vị, còn đối với biến phân loại hàm này chủ yếu thống kê số lượng của từng biến phân loại.

2.2. Phân tích chi tiết:

- Tổng doanh thu từng cửa hàng:

```
> tongstore = new_data %>%
    + group_by(nearest_warehouse) %>%
    + summarise(total_sales = sum(order_total, na.rm = TRUE))
> print(tongstore)
```

```
# A tibble: 3 x 2
  nearest_warehouse total_sales
  <fct>                   <dbl>
1 Bakers                1219908.
2 Nickolson             2226331.
3 Thompson              2248334.
```

Hình 8. Output của function tongstore

➔ Thompson có doanh thu nhiều nhất.

- Tổng doanh thu theo mùa:

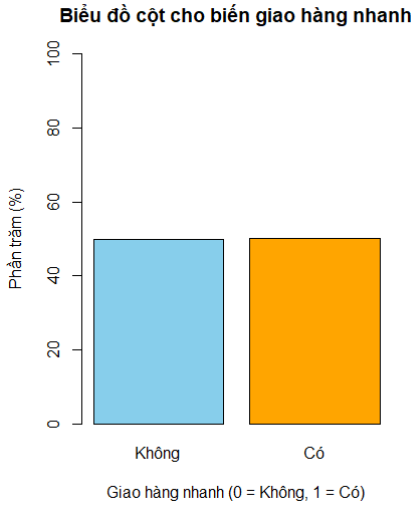
```
> tongmua = new_data %>%
+   group_by(season) %>%
+   summarise(total_sales = sum(order_total, na.rm = TRUE))
> print(tongmua)
```

```
# A tibble: 4 x 2
  season total_sales
  <fct>        <dbl>
1 Spring     1285827.
2 Summer     1328948.
3 Autumn     1433526.
4 Winter     1646272.
```

Hình 9. Output của function tongmua

➔ Doanh thu nhiều nhất vào mùa đông.

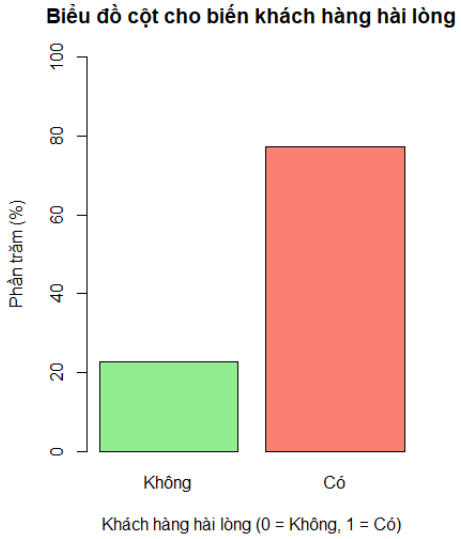
Đối với biến phân loại như is_expedited_delivery, bar plot là đồ thị thích hợp để biểu diễn. *Vẽ đồ thị bar plot đối với biến is_expedited_delivery (đơn vị là %):*



Hình 10. Biểu độ cột cho biến giao hàng nhanh

➔ Từ biểu đồ, có 50% khách có nhu cầu giao hàng nhanh

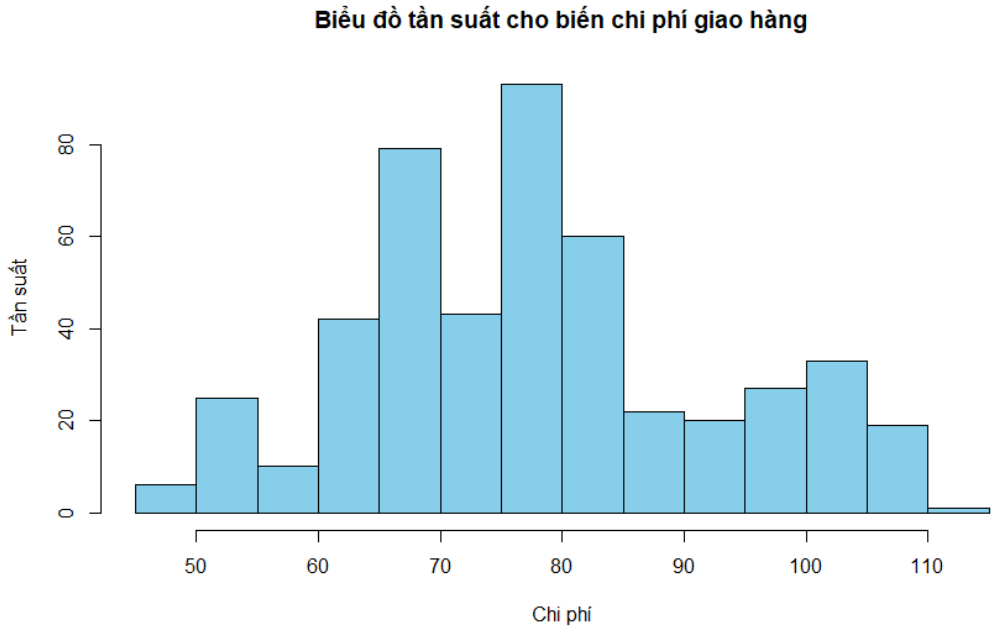
Đồ thị bar plot cho biến is_customer_happy (đơn vị %):



Hình 11. Biểu đồ cột cho biến khách hàng hài lòng

➔ Tỷ lệ khách hàng hài lòng chiếm gần 80%, có thể thấy cửa hàng rất được lòng khách hàng.

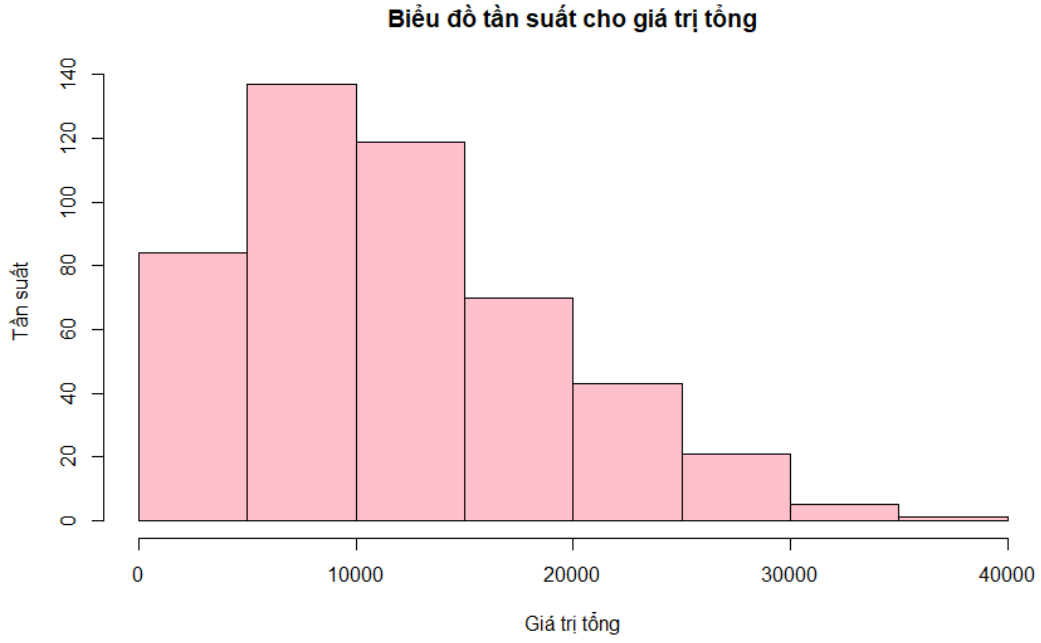
Đồ thị histogram cho biến delivery_charges:



Hình 12. Biểu đồ tần suất cho biến chi phí giao hàng

➔ Chi phí giao hàng dao động từ 60 → 80.

Đồ thị histogram cho biến order_total:



Hình 13. Biểu đồ tần suất cho giá trị tổng

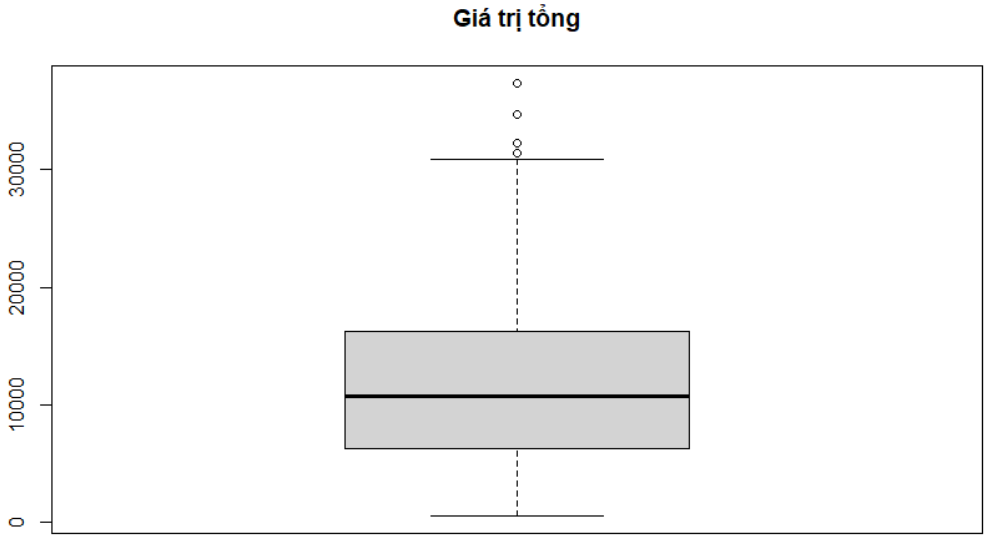
➔ Chi phí đơn hàng sau khi được áp mã giảm giá chủ yếu phân bố quanh giá trị 10000.

2.3. Xử lý ngoại lai:

Dữ liệu ngoại lai là những giá trị dữ liệu (records) được ghi nhận có sự khác biệt bất thường so với những giá trị dữ liệu khác, không theo một quy tắc chung nào và có thể gây ra sự sai lệch trong kết quả phân tích và việc xây dựng thuật toán dự đoán. Ngoài đồ thị histogram, boxplot cũng là một trong những dạng đồ thị phổ biến để biểu thị phân bố của biến, boxplot có một ứng dụng quan trọng có ý nghĩa trong thống kê là biểu thị những

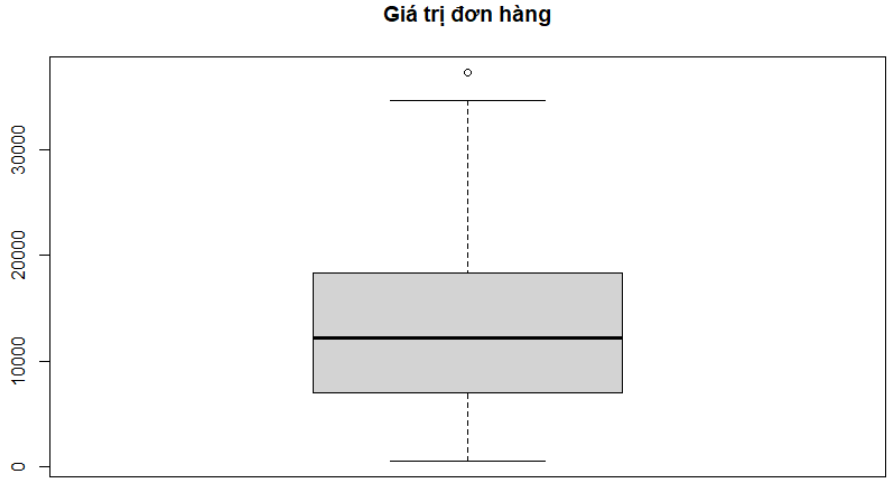
điểm ngoại lai. Từ đây ta có thể nhận biết được những biến nào có giá trị ngoại lai cần loại bỏ thông qua việc sử dụng đồ thị boxplot. Đối với boxplot, dữ liệu ngoại lai được xác định là những dấu chấm ở 2 đầu của biểu đồ. Có nhiều phương pháp thống kê để xử lí ngoại lai, trong bài báo cáo này, nhóm sử dụng phương pháp IQR (Interquartile Range) để xác định giới hạn của các giá trị ngoại lai.

- *Đồ thị boxplot cho biến order_total:*



Hình 14.1 Biểu đồ hộp cho biến giá trị tổng

- Đồ thị boxplot cho biến order_price:



Hình 14.2. Biểu đồ hộp cho biến giá trị đơn hàng

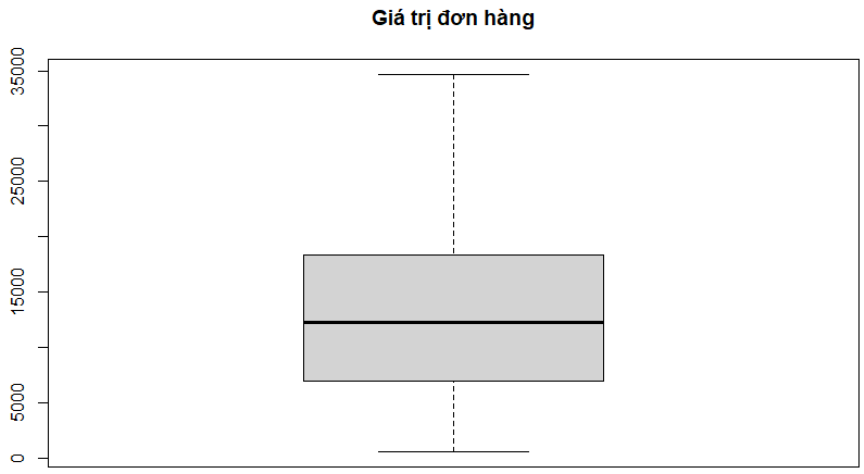
➔ Từ đồ thị trên cho thấy có nhiều điểm ngoại lai khiến ta không thể nhìn rõ được phân phối, cần xử lý những điểm ngoại lai.

Xây dựng hàm xử lý ngoại lai:

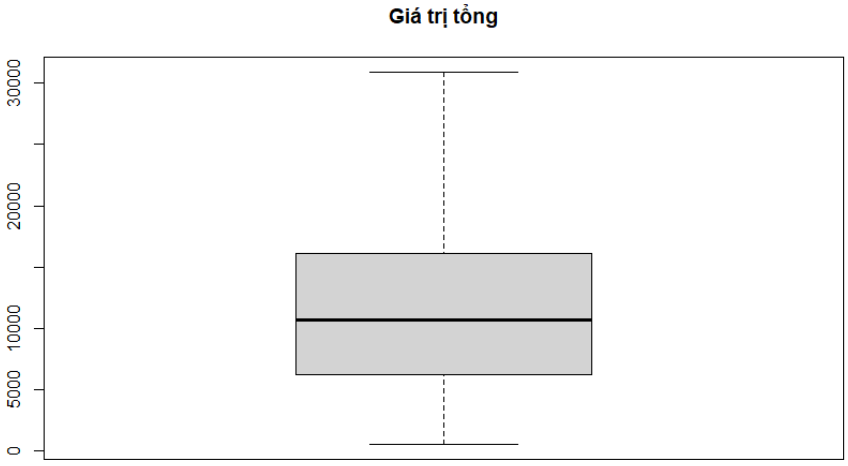
```
> xulingoailai = function(data) {
+ Q1 = quantile(data, 0.25, na.rm = TRUE)
+ Q3 = quantile(data, 0.75, na.rm = TRUE)
+ IQR = Q3 - Q1
+ lower_bound = Q1 - 1.5 * IQR
+ upper_bound = Q3 + 1.5 * IQR
+ data[data < lower_bound | data > upper_bound] = NA
+ return(data)
+ }
> new_data$order_total = xulingoailai(new_data$order_total)
> new_data$order_price = xulingoailai(new_data$order_price)
> boxplot(new_data$order_price,main ="Giá trị đơn hàng")
> boxplot(new_data$order_total,main ="Giá trị tổng")
```

Những biến nằm ngoài khoảng IQR sẽ được gán cho giá trị NA, từ đó chỉ cần loại bỏ những biến NA ta sẽ thu được file dữ liệu sạch đã xoá các biến ngoại lai.

Kiểm tra lại:

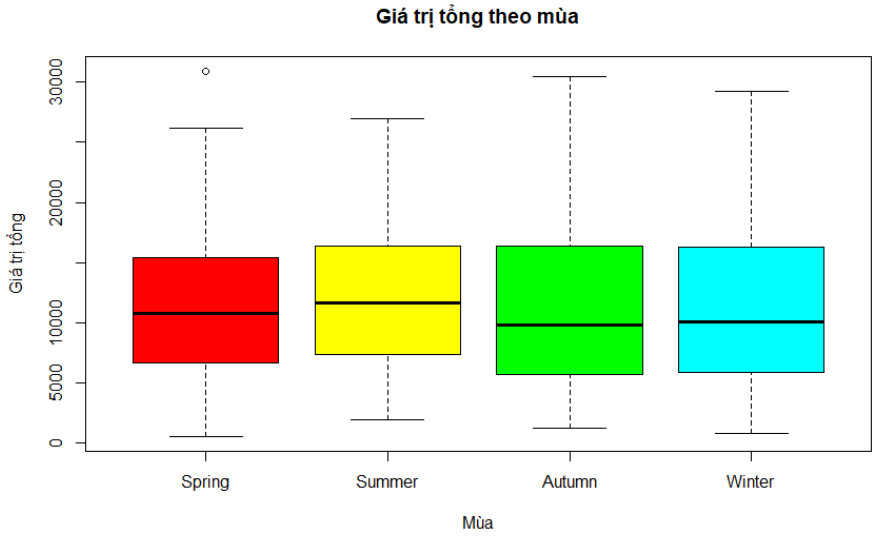


Hình 15.1. Biểu đồ hộp sau khi đã làm sạch của giá trị đơn hàng



Hình 15.2. Biểu đồ hộp sau khi đã làm sạch của giá trị tổng

- Bây giờ tiến hành kiểm tra ngoại lai của biến order_total theo từng mùa:



Hình 16.1 Biểu đồ hợp thể hiện phân bố và ngoại lai của biến order_total theo từng mùa

➔ Khi xét theo từng mùa riêng lẻ, biến order_total xuất hiện 1 điểm ngoại lai ứng với mùa xuân, vì vậy ta cần tiếp tục xử lí điểm ngoại lai này.

```
Spring_data = subset(new_data,new_data$season =="Spring")
Spring_data$order_total = xulingoailai(Spring_data$order_total)
Summer_data = subset(new_data,new_data$season =="Summer")
Summer_data$order_total = xulingoailai(Summer_data$order_total)
Autumn_data = subset(new_data,new_data$season =="Autumn")
Autumn_data$order_total = xulingoailai(Autumn_data$order_total)
Winter_data = subset(new_data,new_data$season =="Winter")
```

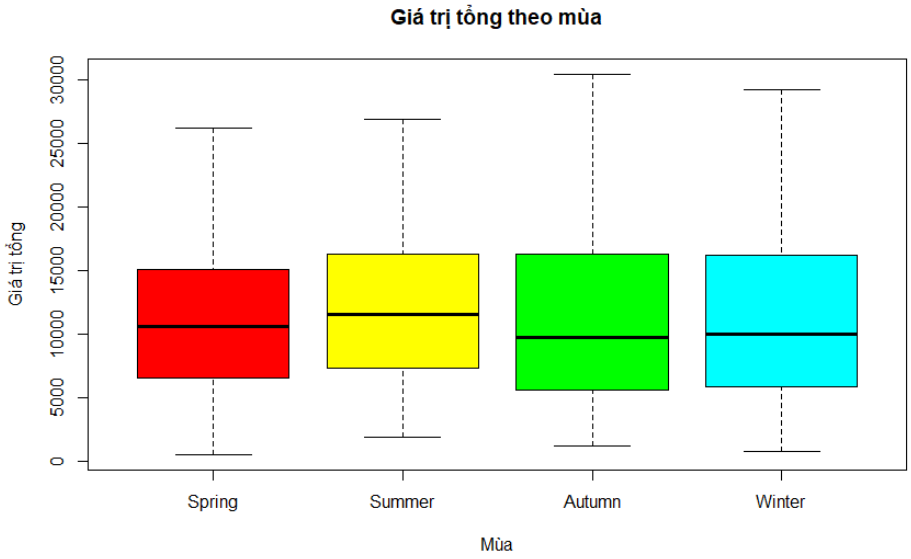
Winter_data$order_total = xulingoailai(Winter_data$order_total)

new_data_2 = rbind(Spring_data,Summer_data,Autumn_data,Winter_data)

apply(is.na(new_data_2),2,sum)

apply(is.na(new_data_2),2,mean)

new_data_2 = na.omit(new_data_2)

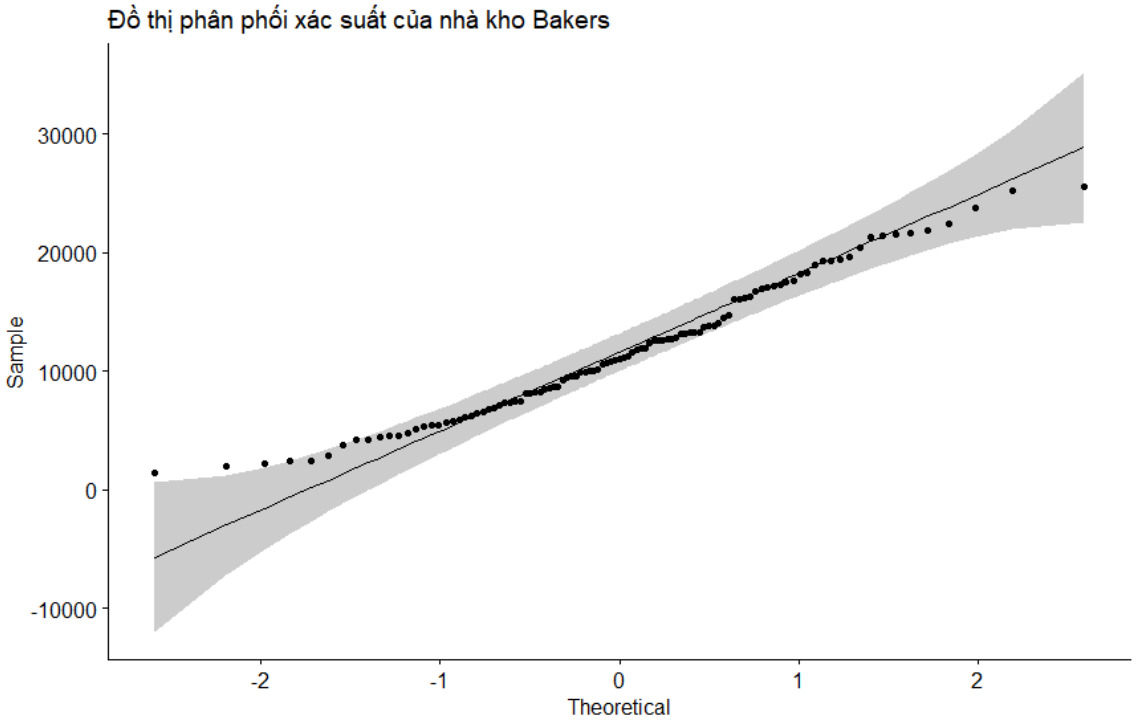


Hình 16.2. Tất cả các điểm ngoại lai của order_total đã được xử lí

2.4. Kiểm tra giả thiết phân phối chuẩn của biến order_total:

Giả định 1: Chi phí đặt hàng ở các kho hàng đều tuân theo phân phối chuẩn

- Bakers

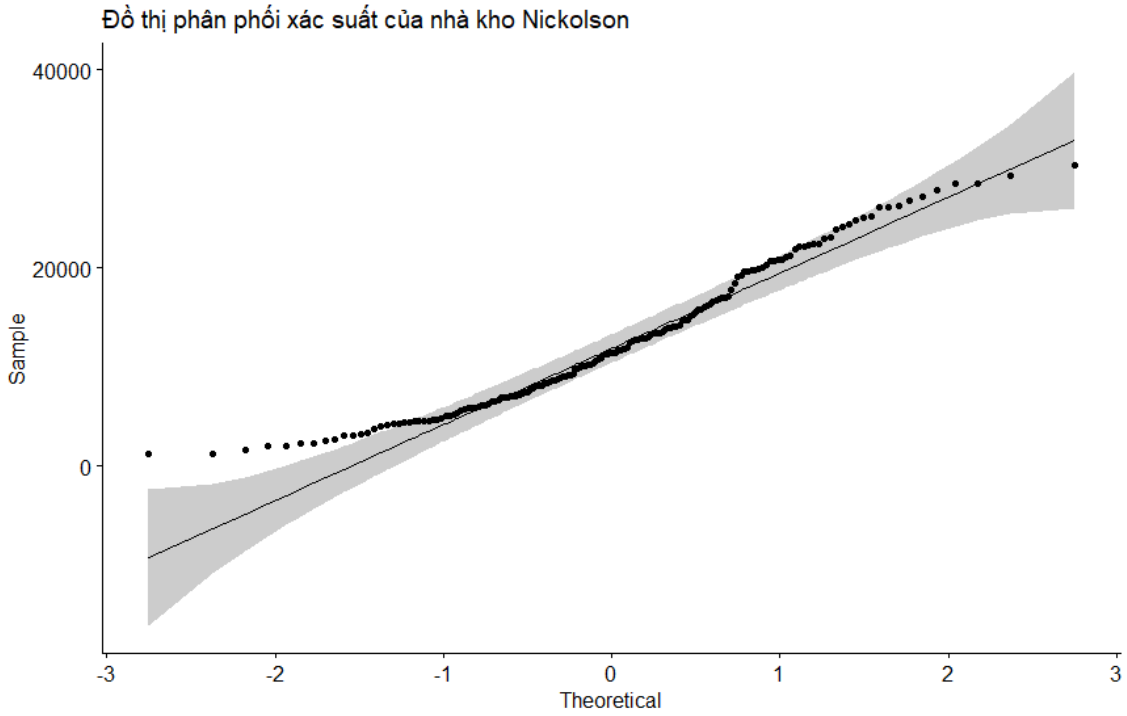


> shapiro.test(Bakers_data$order_total)

```
        Shapiro-Wilk normality test

data:  Bakers_data$order_total
W = 0.97346, p-value = 0.03318
```

- Nickolson

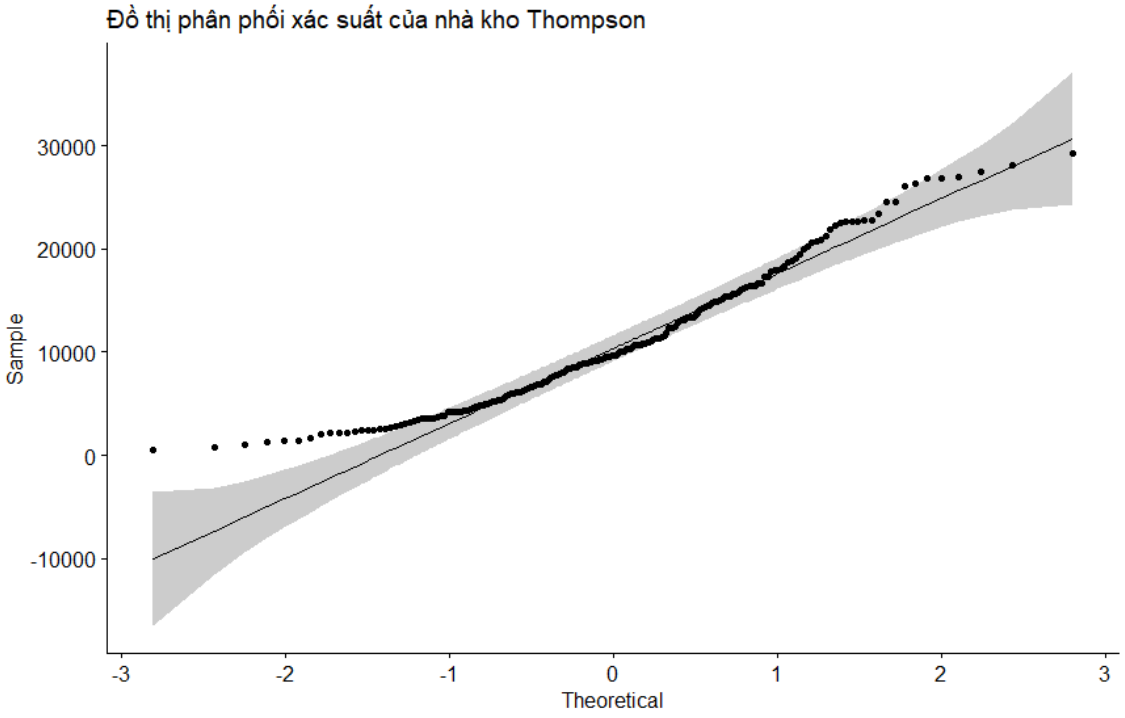


> shapiro.test(Nickolson_data$order_total)

```
        Shapiro-Wilk normality test

data:  Nickolson_data$order_total
W = 0.95293, p-value = 1.846e-05
```

- Thompson



> shapiro.test(Thompson_data$order_total)

```
        Shapiro-Wilk normality test

data:  Thompson_data$order_total
W = 0.94858, p-value = 1.381e-06
```

➔ **Nhận xét:**

- Dựa trên biểu đồ Q-Q và kết quả của kiểm định Shapiro-Wilk, sự phân phối chuẩn của dữ liệu ở 3 kho hàng được biểu hiện như sau:
  - Bakers:
    - Biểu đồ Q-Q: Các điểm dữ liệu có xu hướng nằm gần đường chéo hơn so với Thompson, tuy nhiên vẫn có sự lệch nhẹ ở hai đuôi.
    - Kiểm định Shapiro – Wilk: p-value = 0.03318 (<0.05) nên dữ liệu không tuân theo phân phối chuẩn).
  - Nickolson:
    - Tương tự Thompson, các điểm lệch khỏi đường chéo rõ ràng hơn, đặc biệt ở phần đuôi. Dữ liệu này cũng không tuân theo phân phối chuẩn.
    - Kiểm định Shapiro – Wilk: p-value = 1.846e-05 (< 0.05) nên dữ liệu không tuân theo phân phối chuẩn.
  - Thompson:
    - Biểu đồ Q-Q: Các điểm dữ liệu không hoàn toàn nằm trên đường chéo (đường chuẩn). Đặc biệt ở hai phần đuôi (trái và phải), các điểm lệch ra khỏi đường chéo khá rõ. Điều này cho thấy dữ liệu không tuân theo phân phối chuẩn.
    - Kiểm định Shapiro – Wilk: p-value = 1.381e-06 (< 0.05) nên dữ liệu không tuân theo phân phối chuẩn).
- Vậy, dữ liệu từ cả 3 kho hàng không tuân theo phân phối chuẩn. Điều này cho thấy chi phí đặt hàng ở các kho không tuân theo phân phối chuẩn, có thể ảnh hưởng đến quá trình quản lý và dự đoán chi phí đặt hàng nên cần được xem xét và điều chỉnh để đảm bảo tính chính xác và đáng tin cậy của mô hình.

V. Thống kê suy diễn:

1. So sánh trung bình về chi phí đặt hàng của khách hàng ở 3 kho hàng để xem có kho hàng nào mà chi phí đặt hàng nhiều hơn không?

Giả thiết H0: Phương sai chi phí đặt hàng ở 3 kho hàng bằng nhau.

Giả thiết H1: Phương sai chi phí đặt hàng ở 3 kho hàng khác nhau.

- Thực hiện kiểm định Levene

```
> leveneTest(order_total~nearest_warehouse,  data = new_data_2)
Levene's Test for Homogeneity of Variance (center = median)
       Df F value  Pr(>F)
group   2  2.7341 0.06598 .
      472
---
Signif. codes:
0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

Với p = 0.06598, lớn hơn mức ý nghĩa 5%, chúng ta không có đủ bằng chứng để bác bỏ giả thuyết H0. Do đó, có thể kết luận rằng không có sự khác biệt đáng kể về phương sai của chi phí đặt hàng giữa ba kho hàng.Thực hiện phân tích phương sai 1 nhân tố:

Giả thiết H0: Chi phí đặt hàng trung bình ở 3 kho hàng bằng nhau.

Giả thiết H1: Chi phí đặt hàng trung bình ở 3 kho hàng khác nhau.

```
> one.way = aov(order_total~nearest_warehouse,  data = new_data_2)
```

```
> summary(one.way)
                    Df    Sum Sq   Mean Sq F value Pr(>F)
nearest_warehouse    2 2.258e+08 112906960   2.504 0.0829
Residuals          472 2.129e+10  45099029

nearest_warehouse .
Residuals
---
Signif. codes:
0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

df: Bậc tự do; Sum Sq: Tổng bình phương; Mean Sq: Trung bình bình phương; F value: Giá trị F; Pr(>F): trị số P liên quan đến kiểm định F

Với giá trị p-value là 0.0829, lớn hơn mức ý nghĩa 5%, chúng ta không có đủ bằng chứng để bác bỏ giả thuyết H0. Do đó, có thể kết luận rằng không có sự khác biệt đáng kể về chi phí đặt hàng trung bình của khách hàng giữa ba kho hàng.

2. So sánh trung bình về chi phí đặt hàng của khách hàng ở 4 mùa để xem có mùa nào khách hàng đặt hàng nhiều nhất không?

Giả thiết H0: Phương sai chi phí đặt hàng ở 4 mùa bằng nhau.

Giả thiết H1: Có ít nhất một cặp mùa có phương sai chi phí đặt hàng khác nhau.

```
> leveneTest(order_total~season,  data = new_data_2)
```

```
Levene's Test for Homogeneity of Variance (center = median)
       Df F value Pr(>F)
group   3   0.793 0.4982
      471
```

Với giá trị p-value là 0.4982 lớn hơn mức ý nghĩa 5%, chúng ta không có đủ bằng chứng để bác bỏ giả thuyết H0. Do đó, có thể kết luận rằng không có sự khác biệt đáng kể về phương sai của chi phí đặt hàng giữa 4 mùa. Thực hiện phân tích phương sai 1 nhân tố:

Giả thiết H0: Chi phí đặt hàng trung bình ở 4 mùa bằng nhau.

Giả thiết H1: Có ít nhất 2 mùa có chi phí đặt hàng trung bình khác nhau.

```
> one.way_2 = aov(order_total~season,  data = new_data_2)
```

```
> summary(one.way_2)
```

```
                Df    Sum Sq  Mean Sq F value Pr(>F)
season           3 8.349e+07 27831337   0.612  0.608
Residuals      471 2.143e+10 45496946
```

Với giá trị p-value là 0.608, lớn hơn mức ý nghĩa 5%, chúng ta không có đủ bằng chứng để bác bỏ giả thuyết H0. Do đó, có thể kết luận rằng không có sự khác biệt về chi phí đặt hàng trung bình của khách hàng giữa bốn mùa trong năm.

3. Xây dựng mô hình hồi quy logistic:

- Phân tích các yếu tố ảnh hưởng đến việc hài lòng của khách hàng
    - Đổi kiểu dữ liệu của các biến phân loại thành integer trước khi đưa vào mô hình hồi quy

```
> new_data_2$nearest_warehouse = factor((new_data_2$nearest_warehouse),  levels = c("Bakers", "Nickolson", "Thompson"))
```

```
> new_data_2$season = factor((new_data_2$season),  levels = c("Spring", "Summer", "Autumn","Winter"))
```

```
> new_data_3 = new_data_2[,c("nearest_warehouse", "delivery_charges", "customer_lat", "customer_long", "coupon_discount", "season", "order_total", "distance_to_nearest_warehouse", "is_expedited_delivery", "is_happy_customer")]
```

```
> new_data_3$season = as.integer(new_data_3$season)
```

```
> new_data_3$nearest_warehouse = as.integer(new_data_3$nearest_warehouse)
```

    - Xét mô hình hồi quy logistic gồm biến is_happy_customer là biến phụ thuộc và các biến còn lại là biến độc lập

```
> mohinh1 = glm(is_happy_customer~.,  family = "binomial", data = new_data_3)
```

```
> summary(mohinh1)
```

```
Call:
glm(formula = is_happy_customer ~ ., family = "binomial",

Coefficients:
                              Estimate Std. Error
(Intercept)                  4.484e+03  2.544e+03
nearest_warehouse           -5.981e-01  4.632e-01
delivery_charges             2.680e-01  2.848e-02
customer_lat                 5.465e+01  2.263e+01
customer_long               -1.675e+01  1.626e+01
coupon_discount             -3.389e-02  1.803e-02
season                      -5.718e-01  1.747e-01
order_total                 -1.701e-05  2.432e-05
distance_to_nearest_warehouse -1.844e+00  3.695e-01
is_expedited_delivery       -4.469e+00  5.462e-01
                            z value Pr(>|z|)
(Intercept)                   1.763  0.07795 .
nearest_warehouse            -1.291  0.19665
delivery_charges              9.410  < 2e-16 ***
customer_lat                  2.416  0.01571 *
customer_long                -1.030  0.30307
coupon_discount              -1.880  0.06013 .
season                       -3.273  0.00106 **
order_total                  -0.699  0.48427
distance_to_nearest_warehouse  -4.992 5.99e-07 ***
is_expedited_delivery        -8.182 2.80e-16 ***
---
Signif. codes:
0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 509.27  on 474  degrees of freedom
Residual deviance: 266.27  on 465  degrees of freedom
AIC: 286.27

Number of Fisher Scoring iterations: 6
```

➔ Các biến như nearest_warehouse, customer_long, coupon_discount, order_total có p-value > mức ý nghĩa 5% nên không có giá trị thống kê, còn biến customer_lat cũng không có giá trị thống kê vì chỉ mỗi 1 trị số trong tọa độ vị trí của khách hàng.

- Xét mô hình hồi quy mới sau khi loại bỏ các biến không có giá trị thống kê trên

> new_data_4 = new_data_3[,c("delivery_charges", "season", "distance_to_nearest_warehouse", "is_expedited_delivery", "is_happy_customer")]

> mohinh2 = glm(is_happy_customer~., family = "binomial", data = new_data_4)

> summary(mohinh2)

```
Call:
glm(formula = is_happy_customer ~ ., family = "binomial", data = new_data_4)

Coefficients:
                             Estimate Std. Error
(Intercept)                -12.36417    1.55693
delivery_charges             0.25483    0.02707
season                      -0.54349    0.17238
distance_to_nearest_warehouse  -1.48004    0.32991
is_expedited_delivery          -4.16165    0.51648
                           z value Pr(>|z|)
(Intercept)                 -7.941 2.00e-15 ***
delivery_charges             9.413  < 2e-16 ***
season                      -3.153  0.00162 **
distance_to_nearest_warehouse  -4.486 7.25e-06 ***
is_expedited_delivery          -8.058 7.77e-16 ***
---
Signif. codes:
0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 509.27  on 474  degrees of freedom
Residual deviance: 276.55  on 470  degrees of freedom
AIC: 286.55

Number of Fisher Scoring iterations: 6
```

➔ loại bỏ những biến không có giá trị thống kê trong mô hình 1 cho mô hình 2 chỉ gồm các biến có giá trị thống kê. Giá trị AIC của 2 mô hình gần như tương đương nhau.

- Kiểm định mô hình bằng ANOVA

Giả thuyết H0: 2 mô hình có hiệu quả như nhau.

Giả thuyết H1: 2 mô hình có hiệu quả khác nhau.

> anova(mohinh1, mohinh2, test = "LRT") (LRT = Likelihood Ratio Test)

```
Analysis of Deviance Table

Model 1: is_happy_customer ~ nearest_warehouse + delivery_c
harges + customer_lat +
    customer_long + coupon_discount + season + order_total
+
    distance_to_nearest_warehouse + is_expedited_delivery
Model 2: is_happy_customer ~ delivery_charges + season + di
stance_to_nearest_warehouse +
    is_expedited_delivery
  Resid. Df Resid. Dev Df Deviance Pr(>Chi)
1       465     266.27
2       470     276.55 -5  -10.277  0.06775 .
---
Signif. codes:
0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

➔ Từ kết quả phân tích trên ta thấy p-value = 0.06775 > 0.05 nên ta không có cơ sở để bác bỏ H0. Ở mức ý nghĩa 5%, chúng ta có thể kết luận rằng cả 2 mô hình là tương tự nhau nhưng ta chọn mô hình 2 vì mô hình này có chứa các biến có ý nghĩa thống kê.

- Ta tính được logit(P) (với P là xác suất khách hàng hài lòng):

Logit(P) = -12.36417 + 0.25483 . **delivery_charges** – 0.54349 . **season** – 1.48004 . **distance_to_nearest_warehouse** - 4.16165 . **is_expedited_delivery**

  - Ý nghĩa các tham số trong mô hình hồi quy:
    - Hệ số chặn (intercept)= -12.36417 là giá trị logit(P) khi các biến độc lập còn lại bằng 0. Trong trường hợp này logit(P)= -12.36417.
    - Null deviance: Độ lệch khi không có biến độc lập trong mô hình.
    - Residual deviance: Độ lệch khi có biến độc lập trong mô hình.

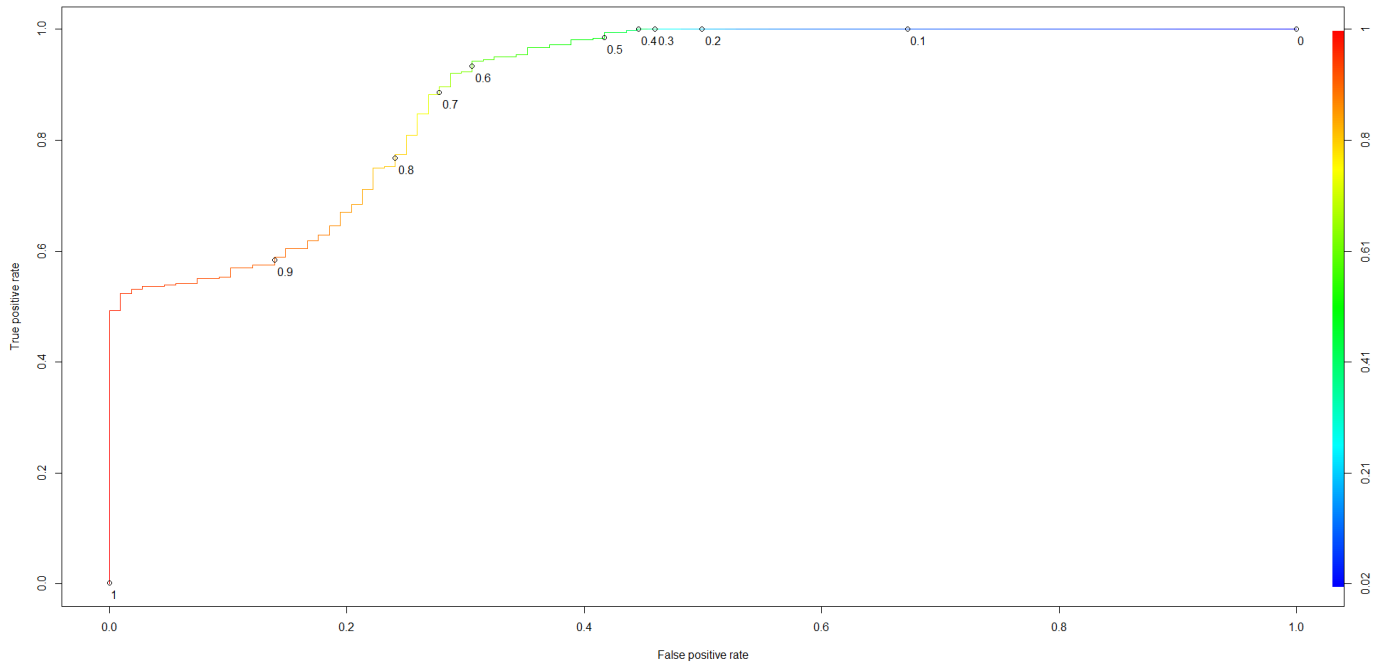
- Hệ số hồi quy của biến độc lập vừa phản ánh mức độ tác động đồng thời cũng thể hiện chiều tác động của biến độc lập lên biến phụ thuộc. Ví dụ: đối với chi phí giao hàng (delivery_charges), nếu tăng chi phí lên 1 đơn vị thì giá trị của logit(P) tăng lên 0.25483, điều này ảnh hưởng đến mức độ hài lòng của khách hàng.

VI. Mở rộng:

1. Xác định tính phù hợp của mô hình bằng đường cong ROC (Receiver Operating Characteristic):

Đường cong ROC là một biểu đồ cho thấy hiệu suất của mô hình phân loại ở tất cả các ngưỡng phân loại. Mỗi điểm trên đường cong ROC là tọa độ tương ứng với tần suất dương tính thật (độ nhạy) trên trục tung và tần suất dương tính giả (1 - độ đặc hiệu) trên trục hoành. Đường biểu diễn càng lệch về phía bên trên và bên trái thì sự phân biệt giữa 2 trạng thái (ví dụ có bệnh hoặc không bệnh) càng rõ.

```
> dubaoROC = predict(mohinh2, type = "response", newdata = new_data_4)
> ROCRpred = prediction(dubaoROC, new_data_4$is_happy_customer)
> ROCRperf = performance(ROCRpred, "tpr", "fpr")
> plot(ROCRperf, colorize=TRUE, print.cutoffs.at= seq(0,1,by=0.1), text.adj=c(-0.2,1.7))
```



➔ đường ROC cho thấy mô hình 2 là mô hình cho dự đoán có độ chính xác rất tốt.

VII. Tài liệu tham khảo:

1. Nguồn code: https://drive.google.com/drive/folders/19ljw9STSOQ2zHbGxqfliAgqViN6VfRkp

2. Coolican, H. (2018). Research methods and statistics in psychology. Routledge

3. Hanneman, R. A., Kposowa, A. J., & Riddle, M. D. (2012). Basic statistics for social research (Vol. 38). John Wiley & Sons.

4. Hoàng Trọng và Chu Nguyễn Mộng Ngọc, Phân tích dữ liệu nghiên cứu với SPSS. Nhà xuất bản thống kê năm 2005.

5. Nguyễn Văn Tuấn. 2007. Phân tích hồi qui logistic trong: Phân tích số liệu và tạo biểu đồ bằng R. Nhà Xuất bản Khoa học và Kỹ thuật. Trang 215 - 218.

6. Michael (2023), (thảo luận về Latitude Longitude Coordinates to State Code in R), truy cập từ https://stackoverflow.com/questions/8751497/latitude-longitude-coordinates-to-state-code-in-r