# 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)

A screenshot of a computer

Description automatically generated

*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)

A screenshot of a computer

Description automatically generated

*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



*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")



*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")



*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

A screenshot of a computer program

Description automatically generated

*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

A screen shot of a computer code

Description automatically generated

*Hình 6. 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)

A screenshot of a computer

Description automatically generated

*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:

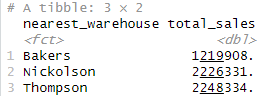
* Tổng doanh thu từng cửa hàng:

> tongstore = new\_data %>%

+ group\_by(nearest\_warehouse) %>%

+ summarise(total\_sales = sum(order\_total, na.rm = TRUE))

> print(tongstore)



*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 screenshot of a computer code

Description automatically generated

*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 %):*

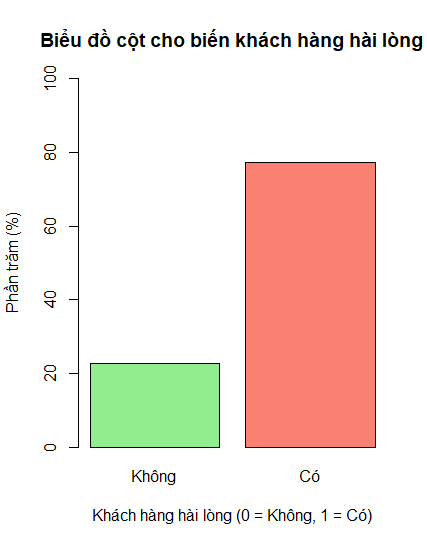
A blue and orange squares

Description automatically generated

*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:*

A graph of blue bars

Description automatically generated with medium confidence

*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:*

A graph of a number of pink bars

Description automatically generated

*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:*

A graph with lines and dots

Description automatically generated

*Hình 14.1 Biểu đồ hộp cho biến giá trị tổng*

* Đồ thị boxplot cho biến order\_price:

A diagram of a graph

Description automatically generated

*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:*

A diagram of a graph

Description automatically generated

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

A diagram of a graph

Description automatically generated

*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:

A chart with different colored squares

Description automatically generated

*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)

A chart with different colored squares

Description automatically generated

*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

A graph of a curve

Description automatically generated

> shapiro.test(Bakers\_data$order\_total)

A black text on a white background

Description automatically generated

* Nickolson

A graph of a curve

Description automatically generated

> shapiro.test(Nickolson\_data$order\_total)

A black text on a white background

Description automatically generated

* Thompson

A graph with lines and numbers

Description automatically generated

> shapiro.test(Thompson\_data$order\_total)

A black text on a white background

Description automatically generated

* **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)

A white background with black text

Description automatically generated

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)

A white background with black text

Description automatically generated

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)

A white background with black text

Description automatically generated

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)



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)

A screenshot of a computer

Description automatically generated

🡺 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)

A screenshot of a computer

Description automatically generated

🡺 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)

A computer screen shot of a program

Description automatically generated

🡺 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))A white background with colorful dots

Description automatically generated

🡺 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: [Link](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>