Hầu hết các lệnh t đã comment, dưới đây là giải thích thêm:

#------------ Mean Squared Error -------------------

sse <- sum((train\_data$Recommended\_Customer\_Price - regression\_model$fitted.values)^2)

n <- length(regression\_model$fitted.values)

df = length(coef(regression\_model))

train\_mse\_unbiased <- sse / (n - df)

Dùng công thức MSE = SSE / (n - <Số biến trong mô hình hồi quy>)

#------------ R Squared ----------------------

test\_r2 <- 1 - sum((true\_test\_price - predicted\_test\_price) ^ 2) / sum((true\_test\_price - mean(true\_test\_price)) ^ 2)

Thằng này dùng công thức

SSE là sum((true\_test\_price - predicted\_test\_price) ^ 2), tương đương

SST là sum((true\_test\_price - mean(true\_test\_price)) ^ 2), tương đương

#------------ Adjusted R Squared ----------------------

n <- nrow(test\_data)

k <- ncol(test\_data) - 1

test\_adjusted\_r2 = 1 - (1 - test\_r2 ^ 2) \* (n - 1) / (n - k - 1)

Dùng thằng Adjusted R2 thay cho R2 vì nó khắc phục nhược điểm của R2 là khi có thêm biến mới không liên quan gì tới Y, thì R2 không thay đổi trong khi đó Adjusted R2 sẽ giảm để nói rằng mô hình không phù hợp

Công thức là

#------ Tạo data frame với giá trị dự đoán và phần dư chuẩn hóa ----------

df <- data.frame(

Fitted\_Values = regression\_model$fitted.values,

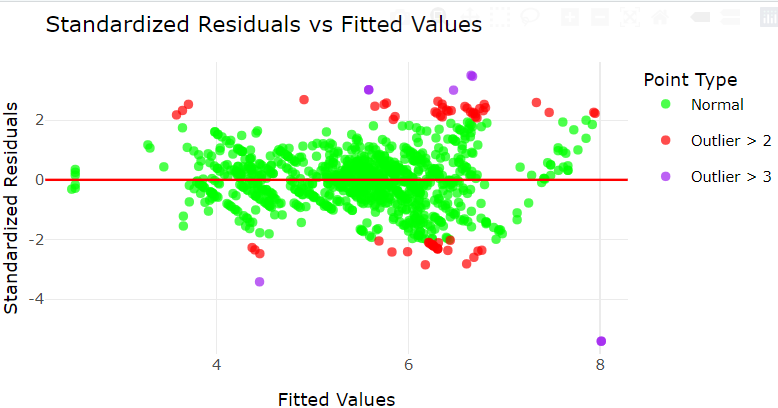
Residuals = rstandard(regression\_model)

)

Cái lệnh rstandard(regression\_model) sẽ trả về phần dư chuẩn hóa của regression\_model: nó được tính như sau, giả sử phần dư gốc của điểm thứ i là ei

* Gọi Q là ma trận kích thước n x (m + 1), ở đây n là số phần tử trong mẫu (= số hàng của dataset dùng để train mô hình hồi quy) còn m là số cột (Product Collection, Vertical\_Segment, …) trong mô hình hồi quy, bản chất là Q = cái bảng dataset để train mô hình hồi quy được chèn thêm 1 cột vào bên trái, cột này Full giá trị 1 (ứng với intercept)
* Tính ma trận , ở đây H có kích thước n x n
* Gọi Hi là phần tử thứ i trên đường chéo của H, thì phần dư chuẩn hóa ri ứng với điểm thứ i như sau

là sai số chuẩn của phần dư, =



Điểm ngoại lai (Outlier) là điểm có phần dư chuẩn hóa lớn, thường thì phần dư chuẩn hóa > 2 hoặc > 3 thì là điểm ngoại lai (điểm mà mô hình dự đoán sai bét)

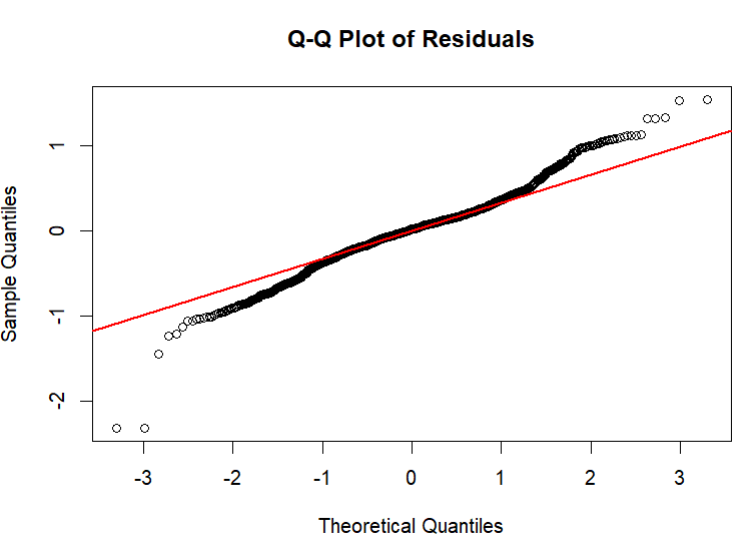
#--- Biểu đồ Q-Q, trục hoành là phân vị của phân phối chuẩn theo lí thuyết -----

#--- trục tung là phân vị thực tế của phần dư ----

#--- Các điểm càng phân bổ dọc theo đường chéo thì phân phối của phần dư càng gần phân phối chuẩn -------------

qqnorm(regression\_model $ residuals, main = "Q-Q Plot of Residuals")

qqline(regression\_model $ residuals, col = "red", lwd = 2)



Cái biểu đồ Q-Q này thì như đã comment:V

Giả sử các phần dư chuẩn hóa sắp xếp theo thứ tự tăng dần là ,

tính , là hàm phân vị ngược của phân phối chuẩn tắc, ví dụ (0.6) trả về giá trị x sao cho tích phân từ âm vô cực đến x của cái đồ thị phân phối chuẩn tắc là = 0.6

lập các cặp tọa độ rồi plot (trục hoành q, trục tung r)

Cái đường màu đỏ là đường hồi quy của các cái điểm này, các điểm này càng gần đường màu đỏ thì phân phối của phần dư càng giống phân phối chuẩn

log\_predicted\_price\_1 <- predict(regression\_model, newdata = first\_cpu, se.fit = TRUE)

predicted\_price\_1 <- exp(log\_predicted\_price\_1 $ fit) - 1

# Do cái mô hình hồi quy lúc trước là ta đã lấy log (giá CPU + 1) nên giờ sau khi dự đoán thì phải e mũ lên rồi trừ 1

lower\_bound\_1 = exp((log\_predicted\_price\_1 $ fit) - 1.645 \* (log\_predicted\_price\_1 $ se.fit)) - 1

upper\_bound\_1 = exp((log\_predicted\_price\_1 $ fit) + 1.645 \* (log\_predicted\_price\_1 $ se.fit)) - 1

# Ở đây do số phần tử trong mẫu lớn > 1000, nên dùng phân phối chuẩn thay cho phân phối Student, cái giá trị 1.645 là Z Value cho khoảng tin cậy đối xứng 90%

# log\_predicted\_price\_1 $ se.fit là sai số chuẩn của giá CPU đã lấy log, do đó khoảng tin cậy 90% của giá CPU đã lấy log = giá CPU đã lấy log +- Z Value \* sai số chuẩn

# Đây mới chỉ là giá CPU đã lấy log, còn để tính khoảng tin cậy cho giá gốc CPU thì e mũ rồi -1 hai cận (do trước đó ta log (giá CPU + 1))

Định dạng in ra (dự đoán giá của 2 CPU):

First CPU recommended price: 1778.506

90% Confident Interval: [ 1150.395 , 2749.265 ]

* Tức kì vọng giá CPU này = 1778.506 , và giá của CPU này 90% là rơi vào đoạn [ 1150.395 , 2749.265 ]

Second CPU recommended price: 3167.399

90% Confident Interval: [ 2162.34 , 4639.393 ]

* Giải thích tương tự CPU đầu tiên