

ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH

TRƯỜNG ĐẠI HỌC BÁCH KHOA

Khoa Khoa học ứng dụng

Bộ môn Toán ứng dụng



BÀI TẬP LỚN MÔN XÁC SUẤT VÀ THỐNG KÊ

**PHÂN TÍCH DỮ LIỆU CỦA BỘ XỬ LÝ ĐỒ HỌA VÀ XÂY DỰNG MÔ HÌNH
DỰ ĐOÁN CHO GIÁ BÁN CHƯA BIẾT CỦA GPU_s DỰA TRÊN CÁC THÔNG
SỐ KỸ THUẬT BẰNG MÔ HÌNH HỒI QUY TUYẾN TÍNH BỘI**

HỌC KỲ: 223 (2022-2023)

GVHD: TS. PHAN THỊ HƯỜNG

NHÓM: 05 - LỚP: DT04

KHOA: KHOA HỌC VÀ KỸ THUẬT MÁY TÍNH

NGÀY: 30/08/2023

Thành phố Hồ Chí Minh – 2023

STT	HỌ VÀ TÊN	MSSV	ĐÓNG GÓP	ĐIỂM
1	Hồ Vũ Hoàng Hà	2210845	100%	
2	Nguyễn Nhật Nam	2212147	100%	
3	Nguyễn Trung Nguyên	2011710	0%	
4	Nguyễn Lê Vân Tú	2252881	100%	

MỤC LỤC

I. TỔNG QUAN DỮ LIỆU	4
1.1 Giới thiệu chung	4
1.2 Các biến trong bảng dữ liệu	4
II. KIẾN THỨC NỀN	6
III. TIỀN XỬ LÝ DỮ LIỆU	12
IV. THỐNG KÊ TẢ	14
1. Scatter plots	14
2. Histogram	16
3. Boxplot	17
V. THỐNG KÊ SUY DIỄN	18
5.1 Yếu tố nào sẽ tác động đến “Release_Price” nhiều nhất?	18
5.1.1 Chia dữ liệu	18
5.1.2 Mô hình phù hợp	18
5.1.3 Mô hình suy diễn	22
5.1.4 Giả định về mô hình cần kiểm tra	23
5.1.5 Dự đoán cho giá bán của GPU	24
5.1.6 Kết luận	27
VI. THẢO LUẬN VÀ MỞ RỘNG	28
VII. NGUỒN DỮ LIỆU VÀ NGUỒN CODE	28
VIII. TÀI LIỆU THAM KHẢO	28

Danh sách các hình

Hình 1: 6 dòng đầu của dữ liệu.....	12
Hình 2: Bảng dữ liệu gồm 6 biến sau khi lọc lại.....	13
Hình 3: Dữ liệu sau khi chuyển thành dạng số.....	13
Hình 4: Thống kê dữ liệu khuyết.....	13
Hình 5: Dữ liệu gồm các hàng chứa biến.....	14
Hình 6: Biểu đồ phân tán của 6 biến đối với biến Release_Price	15
Hình 7: Biểu đồ cột của 7 biến.....	16
Hình 8: Biểu đồ hộp của 7 biến.....	17
Hình 9: Nhận diện điểm ngoại lai của biến Release_Price	18
Hình 10: Mô hình gốc.....	19
Hình 11: Mô hình cải thiện thứ nhất.....	20
Hình 12: Mô hình cải thiện thứ hai.....	20
Hình 13: Mô hình cải thiện thứ ba.....	21
Hình 14: Mô hình cải thiện thứ tư.....	21
Hình 15: Giả định về mô hình.....	23
Hình 16: Mô hình hồi quy tuyến tính.....	24
Hình 17: Dự đoán cho 30% giá trị còn lại của Release_Price	25
Hình 18: So sánh với 30% dữ liệu ban đầu.....	25
Hình 19: Bộ dữ liệu thiếu các giá trị của biến Release_Price	26
Hình 20: Kết quả dự đoán.....	27

I. TỔNG QUAN DỮ LIỆU

1.1 Giới thiệu chung

GPU (Graphic Processing Unit): Là bộ vi xử lý chuyên phân tích những khối dữ liệu hình ảnh. Những tác vụ liên quan tới đồ họa và video. Khác biệt với CPU là GPU chuyên dụng xử lý những tác vụ hình ảnh. Hai hãng sản xuất GPU nổi tiếng trên thị trường hiện nay đó chính là Nvidia và AMD/ATI mỗi hãng đều có những đặc điểm và lợi thế khác nhau.

GPU còn xử lý thông tin đa luồng, song song và bộ nhớ ở tốc độ cao. Kỹ thuật GPU đang dần trở nên dễ lập trình, cung cấp nhiều tiềm năng cho việc tăng tốc xử lý cho nhiều chương trình với nhiều mục đích khác nhau, hơn cả bộ xử lý trung tâm (CPUs).

Trong đề tài này, ta sẽ phân tích tệp dữ liệu chứa thông số kỹ thuật chi tiết, ngày phát hành và giá phát hành của hơn 3000 bộ xử lý đồ họa (GPU) trong máy tính.

Dữ liệu đầu vào có chứa tệp lưu trữ dưới dạng CSV: gpus.csv cho bộ xử lý đồ họa (GPU). Bảng dữ liệu có danh sách các cột biểu thị cho thông tin cần thiết của dữ liệu và một số đặc trưng sẽ bao gồm: tốc độ xung nhịp, nhiệt độ tối đa, độ phân giải màn hình, mức tiêu thụ năng lượng, số luồng, ngày phát hành, giá phát hành, kích thước khuôn, hỗ trợ ảo hóa và nhiều đề mục tương tự khác.

Dữ liệu được trích từ:

<https://www.kaggle.com/datasets/iliassekkaf/computerparts?resource>

1.2 Các biến trong bảng dữ liệu

STT	TÊN BIẾN	LOẠI BIẾN	ĐẶC TẢ
1	Best_resolution	Ngẫu nhiên	Kích thước tối ưu nhất cho màn hình để giảm thiểu tình trạng màn hình bị ngưng/dừng đột ngột.
2	Boost_Clock (MHz)	Liên tục	Tốc độ xử lý của card đồ họa sau khi boost clock
3	Core_Speed (MHz)	Liên tục	Tốc độ xử lý của card đồ

			họa
4	DVI_Connection	Rời rạc	Số lượng cổng hỗ trợ kết nối với các thiết bị điện tử như máy tính, tivi, laptop,... giúp truyền hình ảnh, phát video trực tiếp đến màn hình thông qua một kết nối duy nhất mà không cần phải chuyển về analog.
5	DisplayPort_Connecti on	Rời rạc	Số lượng cổng có chức năng trích xuất hình ảnh và âm thanh chất lượng cao từ thiết bị nguồn sang màn hình TV, laptop, máy chiếu, màn hình máy tính,...
6	HDMI_Connection	Rời rạc	Số lượng cổng dùng để truyền tải âm thanh và hình ảnh với những thiết bị khác như tivi, laptop... cùng độ phân giải cao.
7	Memory	Liên tục	Bộ nhớ của card đồ họa
8	Memory_Bandwidth	Liên tục	Tốc độ mà bộ xử lý có thể đọc hoặc lưu trữ dữ liệu vào bộ nhớ bán dẫn. Băng thông bộ nhớ thường được biểu thị bằng đơn vị byte/giây, mặc dù điều này có thể thay đổi đối với các hệ thống có kích thước dữ liệu tự nhiên không phải là bội số của byte 8 bit thường được sử dụng.
9	Memory_Bus	Liên tục	Một loại bus máy tính, thường ở dạng một bộ dây hoặc dây dẫn kết nối các thành phần điện và cho phép

			truyền dữ liệu và địa chỉ từ bộ nhớ chính đến bộ xử lý trung tâm (CPU) hoặc bộ điều khiển bộ nhớ.
10	Memory_Speed	Liên tục	Là tốc độ bộ nhớ RAM của card được tính bằng MHz, hiểu đơn giản là tốc độ mà card có thể truy cập dữ liệu được lưu trữ trên RAM.
11	PSU	Liên tục	Nguồn điện (Watt và Ampe) để card đồ họa hoạt động.
12	Pixels_Rate	Liên tục	Số pixel mỗi giây mà GPU có thể tạo ra.
13	Process	Liên tục	Loại công nghệ nano mà GPU sử dụng.
14	ROPs	Liên tục	Số lượng đường ống vận hành Raster trong GPU
15	Resolution_WxH	Liên tục	Độ phân giải tối đa được hỗ trợ.
16	TMUs	Liên tục	Số đơn vị ánh xạ kết cấu trong GPU.
17	Texture_Rate	Liên tục	Số pixel kết cấu mỗi giây GPU có thể tạo ra.
18	VGA Connection	Rời rạc	Số lượng đầu nối tiêu chuẩn được sử dụng cho đầu ra hình ảnh.
19	Release_Price	Liên tục	Giá bán ra của GPU

II. KIẾN THỨC NỀN

I. Phân tích hồi quy:

1. Định nghĩa:

Hồi quy (regression) là phương pháp thống kê toán học để ước lượng và kiểm định các quan hệ giữa các biến ngẫu nhiên, và có thể từ đó đưa ra các dự báo. Các quan hệ ở đây được viết dưới dạng các hàm số hay phương trình. Ý tưởng chung như sau: giả sử ta có một biến ngẫu nhiên Y , mà ta muốn ước lượng xấp xỉ dưới dạng một hàm số của các biến ngẫu nhiên khác (control variables), hay còn gọi là biến tự do, trong khi

Y được gọi là biến phụ thuộc, tức là khi ta có các giá trị của, thì ta muốn từ đó ước lượng được giá trị của Y. Hàm số F này có thể phụ thuộc vào một số tham số nào đó. Ta có thể viết Y như sau: trong đó là phần sai số (cũng là một biến ngẫu nhiên). Ta muốn chọn hàm F một cách thích hợp nhất có thể, và các tham số, sao cho sai số là nhỏ nhất có thể. Đại lượng được gọi là sai số chuẩn (standard error) của mô hình hồi quy. Mô hình nào mà có sai số chuẩn càng thấp thì được coi là càng chính xác.

2. Bản chất:

*** Bản chất của biến phụ thuộc Y**

Y nói chung được giả định là một biến ngẫu nhiên, và có thể được đo lường bằng một trong bốn thước đo sau đây: thang đo tỷ lệ, thang đo khoảng, thang đo thứ bậc, và thang đo danh nghĩa. Thang đo tỷ lệ (ratio scale): Một thang đo tỷ lệ có 3 tính chất: (1) tỷ số của hai biến, (2) khoảng cách giữa hai biến, và (3) xếp hạng các biến. Với thang đo tỷ lệ, ví dụ Y có hai giá trị, và thì tỷ số / và khoảng cách (-) là các đại lượng có ý nghĩa; và có thể so sánh hoặc xếp thứ tự. Thang đo khoảng (interval scale): Thang đo khoảng không thỏa mãn tính chất đầu tiên của các biến có thang đo tỷ lệ. Thang đo thứ bậc (ordinal scale): Các biến chỉ thỏa mãn tính chất xếp hạng của thang đo tỷ lệ, chứ việc lập tỉ số hay tính khoảng cách giữa hai giá trị không có ý nghĩa. Thang đo danh nghĩa (nominal scale): Các biến thuộc nhóm này không thỏa mãn bất kì tính chất nào của các biến theo thang đo tỷ lệ. (như giới tính, tôn giáo,...).

*** Bản chất của biến ngẫu nhiên X:**

Các biến ngẫu nhiên có thể được đo theo bất kỳ một trong bốn thang đo vừa nêu trên, mặc dù trong nhiều ứng dụng thực tế thì các biến giải thích được đo theo thang đo tỷ số và thang đo khoảng.

*** Bản chất của sai số ngẫu nhiên (nhiều):**

Sai số ngẫu nhiên đại diện cho tất cả các biến không được đưa vào mô hình vì những lý do như không có sẵn dữ liệu, các lỗi đo lường trong dữ liệu. Và cho dù nguồn tạo nhiễu là gì đi nữa, thì người ta giả định rằng ảnh hưởng trung bình của sai số ngẫu nhiên lên Y là không đáng kể. Ta cũng giả định là hạng nhiễu có phân phối chuẩn với trung bình bằng 0 và phương sai không đổi là:

$$\sigma^2$$

*** Bản chất của tham số hồi quy**

Tham số hồi quy (tổng thể), , là những con số cố định (fixed numbers) và không ngẫu nhiên (not random), mặc dù mình không thể biết giá trị thực của là bao nhiêu.

3. Ý nghĩa của hồi quy tuyến tính

Thuật ngữ tuyến tính (linear) trong mô hình hồi quy tuyến tính nghĩa là tuyến tính các hệ số hồi quy (linearity in the regression coefficients), và không phải tuyến tính các biến Y và X.

4. Mô hình hồi quy bội

Mô hình hồi quy bội là mô hình hồi quy trong đó: biến phụ thuộc Y phụ thuộc vào (k-1) biến độc lập X_2, X_3, \dots, X_k có dạng như sau:

Hàm hồi quy tổng thể: $E(Y|X_2, X_3, \dots, X_k) = \beta_1 + \beta_2 X_2 + \beta_3 X_3 + \dots + \beta_k X_k$

Mô hình hồi quy tổng thể: $Y = \beta_1 + \beta_2 X_{2i} + \beta_3 X_{3i} + \dots + \beta_k X_{ki} + \epsilon$

Trong đó:

- u_k là sai số ngẫu nhiên.
- β_1 là hệ số tự do (hệ số chặn), bằng giá trị trung bình của Y khi $X_j=0$.
- β_j là hệ số hồi quy riêng (hay hệ số góc), thể hiện ảnh hưởng của riêng từng biến độc lập X_j lên trung bình của Y khi các biến khác được giữ không đổi. Cụ thể, khi X_j tăng hoặc giảm 1 đơn vị, trong điều kiện các biến độc lập khác không đổi, thì Y trung bình sẽ thay đổi j đơn vị. Có thể nhận thấy khả năng có thể xảy ra đối với các hệ số góc:
 - Hệ số $\beta_j > 0$: khi đó mối quan hệ giữa Y và X_j là thuận chiều, nghĩa là khi X_j tăng (hoặc giảm) trong điều kiện các biến độc lập khác không đổi thì Y cũng sẽ tăng (hoặc giảm).
 - Hệ số $\beta_j < 0$: khi đó mối quan hệ giữa Y và X_j là ngược chiều, nghĩa là khi X_j tăng (hoặc giảm) trong điều kiện các biến độc lập khác không đổi thì Y sẽ giảm (hoặc tăng).
 - Hệ số $\beta_j = 0$: có thể cho rằng giữa Y và X_j không có tương quan với nhau, cụ thể là Y có thể không phụ thuộc vào X_j hay là X_j không thực sự ảnh hưởng tới Y.
 - ϵ : sai số.

Dựa vào kết quả ước lượng với một mẫu cụ thể, ta có thể đánh giá được mối quan hệ giữa biến phụ thuộc và các biến độc lập trong mô hình một cách tương đối.

Dù mô hình có nhiều biến độc lập nhưng vẫn tồn tại những yếu tố tác động đến biến phụ thuộc nhưng không đưa vào mô hình vì nhiều lý do (không có số liệu hoặc không muốn đưa vào). Do đó trong mô hình vẫn tồn tại sai số ngẫu nhiên đại diện cho các yếu tố khác ngoài biến X_j ($j=2,3,\dots,k$) có tác động đến Y nhưng không đưa vào mô hình như là biến số.

• **Các giả thiết của mô hình hồi quy bội:**

- ❖ **Giả thiết 1:** Việc ước lượng được dựa trên cơ sở mẫu ngẫu nhiên.
- ❖ **Giả thiết 2:** Kỳ vọng của sai số ngẫu nhiên tại mỗi giá trị ($X_{2i}, X_{3i}, \dots, X_{ki}$) bằng 0: $E(u_i | X_{2i}, X_{3i}, \dots, X_{ki}) = 0$
- ❖ **Giả thiết 3:** Phương sai của sai số ngẫu nhiên tại các giá trị ($X_{2i}, X_{3i}, \dots, X_{ki}$) đều bằng nhau.
 - Từ giả thiết 2 và 3 ta có thể nói sai số ngẫu nhiên (u) tuân theo phân phối chuẩn.
- ❖ **Giả thiết 4:** Giữa các biến độc lập X_j không có quan hệ cộng tuyến hoàn hảo, nghĩa là không tồn tại hằng số $\lambda_2, \lambda_3, \dots, \lambda_k$ không đồng thời bằng 0 sao cho:

$$\lambda_2 X_2 + \lambda_3 X_3 + \dots + \lambda_k X_k = 0.$$

- Có thể nhận thấy nếu giữa các biến $X_j (j = 2, 3, \dots, k)$ có quan hệ cộng tuyến hoàn hảo thì sẽ có ít nhất một trong các biến này sẽ suy ra được từ các biến còn lại. Do đó, giả thiết 4 được đưa ra để loại trừ tình huống này.

- **Phương pháp ước lượng mô hình hồi quy bội – Phương pháp bình phương nhỏ nhất (OLS):**

Sau khi xây dựng và tìm hiểu ý nghĩa của các hệ số hồi quy trong mô hình, vấn đề tiếp theo ta quan tâm là làm sao để có được các ước lượng đáng tin cậy cho các hệ số β_j này. Cũng như với mô hình hồi quy hai biến, ta sẽ sử dụng phương pháp bình phương nhỏ nhất (OLS) để ước lượng các hệ số trong mô hình hồi quy k biến.

- Xét mô hình k biến: $Y = \beta_1 + \beta_2 X_2 + \beta_3 X_3 + \dots + \beta_k X_k + u_k$

Giả sử có một mẫu quan sát với giá trị thực tế là $(X_i, X_{2i}, \dots, X_{ki})$ với $(i = 1, 2, \dots, n)$. Ta sẽ sử dụng thông tin từ mẫu để xây dựng các ước lượng cho các hệ số $\beta_j (j = 1, 2, \dots, k)$, ký hiệu là $\hat{\beta}_j (j = 1, 2, \dots, k)$. Từ các giá trị ước lượng này có thể viết thành hàm hồi quy mẫu như sau:

$$Y_i = \beta_1 + \beta_2 X_{2i} + \beta_3 X_{3i} + \dots + \beta_k X_{ki}$$

Tại mỗi quan sát i, hàm hồi quy mẫu được viết thành:

$$\hat{Y}_i = \hat{\beta}_1 + \hat{\beta}_2 X_{2i} + \hat{\beta}_3 X_{3i} + \dots + \hat{\beta}_k X_{ki}$$

Trong đó $\hat{\beta}_j$ là giá trị ước lượng cho β_j và sai lệch giữa hai giá trị này được gọi là phần dư cách tính:

$$e_i = Y_i - \hat{Y}_i$$

Tương tự như mô hình hồi quy hai biến, phương pháp OLS nhằm xác định các giá trị $\hat{\beta}_j (j = 1, 2, \dots, k)$ sao cho tổng bình phương các phần dư là bé nhất:

$$\sum_{i=1}^n e_i^2 = \sum_{i=1}^n (Y_i - \hat{Y}_i)^2 = \sum_{i=1}^n (Y_i - \hat{\beta}_1 - \hat{\beta}_2 X_{2i} - \dots - \hat{\beta}_k X_{ki})^2 = f(\hat{\beta}_1, \hat{\beta}_2, \dots, \hat{\beta}_k) \rightarrow \text{Min}$$

Khi đó, các giá trị:

$$X_i = X_1 + X_2 + X_3 + \dots + X_k$$

$$\begin{cases} \frac{\partial f(\hat{\beta}_1, \hat{\beta}_2, \dots, \hat{\beta}_k)}{\partial \hat{\beta}_1} = -2 \sum_{i=1}^n (Y_i - \hat{\beta}_1 - \hat{\beta}_2 X_{2i} - \dots - \hat{\beta}_k X_{ki}) = 0 \\ \frac{\partial f(\hat{\beta}_1, \hat{\beta}_2, \dots, \hat{\beta}_k)}{\partial \hat{\beta}_2} = -2 \sum_{i=1}^n X_{2i} (Y_i - \hat{\beta}_1 - \hat{\beta}_2 X_{2i} - \dots - \hat{\beta}_k X_{ki}) = 0 \\ \dots \dots \dots \\ \frac{\partial f(\hat{\beta}_1, \hat{\beta}_2, \dots, \hat{\beta}_k)}{\partial \hat{\beta}_k} = -2 \sum_{i=1}^n X_{ki} (Y_i - \hat{\beta}_1 - \hat{\beta}_2 X_{2i} - \dots - \hat{\beta}_k X_{ki}) = 0 \end{cases}$$

- Với điều kiện số quan sát trong mẫu lớn hơn số hệ số hồi quy cần ước lượng và giả thiết 4 được thỏa mãn thì hệ phương trình trên sẽ có nghiệm duy nhất. Việc

giải hệ phương trình khá dễ dàng qua các phần mềm thống kê nếu số biến không quá lớn. Các giá trị ước lượng bằng phương pháp OLS dựa trên số liệu mẫu cụ thể được xem như là các ước lượng điểm của các hệ số trong tổng thể. Với mô hình hồi quy bội (hồi quy k biến với $k > 2$), việc giải hệ phương trình để tìm các ước lượng hệ số j ($j = 1, 2, 3 \dots k$) sẽ trở nên khó khăn hơn so với mô hình hồi quy 2 biến do đó ta sẽ có được các kết quả này với sự giúp của các phần mềm thống kê.

- Từ kết quả ước lượng từ phương pháp OLS, ta có thể khai thác các thông tin để đánh giá tác động của biến độc lập đối với sự thay đổi của biến phụ thuộc thông qua ý nghĩa các hệ số hồi quy.
- Khi các giả thiết từ 1 đến 4 thỏa mãn thì các ước lượng thu được từ phương pháp OLS là ước lượng tuyến tính, không chệch và có phương sai nhỏ nhất trong lớp các ước lượng tuyến tính không chệch. Hay nói một cách khác, nếu giả thiết từ 1 đến 4 được thỏa mãn thì ước lượng OLS là ước lượng tốt nhất trong lớp các ước lượng tuyến tính không chệch.

- **Đánh giá mức độ phù hợp của mô hình hồi quy bội:**

Khi đánh giá một mô hình dựa trên số liệu mẫu, nếu chỉ quan tâm đến các ước lượng hệ số và độ lệch chuẩn của nó thì chưa đầy đủ. Có một con số cũng góp phần không nhỏ khi đánh giá chất lượng mô hình đó là **hệ số xác định**.

Sau khi ước lượng được mô hình hồi quy trong một khoảng tin cậy, ta muốn biết hàm hồi quy mẫu phù hợp với số liệu mẫu đến mức nào. Có thể đánh giá điều đó qua hệ số xác định bội. Ký hiệu .

Cách xác định hệ số xác định bội:

Ta có:

$$TSS = \sum_{i=1}^n y_i^2 = \sum_{i=1}^n (Y_i - \bar{Y})^2$$

$$ESS = \sum_{i=1}^n \hat{y}_i^2 = \sum_{i=1}^n (\hat{Y}_i - \bar{Y})^2$$

$$RSS = \sum_{i=1}^n e_i^2 = \sum_{i=1}^n (Y_i - \hat{Y}_i)^2$$

Trong đó:

TSS (total sum of square): Tổng bình phương độ lệch toàn phần.

ESS (Explained sum of square): Tổng bình phương độ lệch phần hồi quy.

RSS (Residual sum of square): Tổng bình phương phần dư.

$$TSS = ESS + RSS$$

Khi đó hệ số xác định bội của mô hình được xác định bởi công thức sau:

$$R^2 = 1 - (ESS/TSS)$$

Do các thành phần TSS, ESS, RSS đều không âm, nên từ biểu thức trên có thể thấy:

$$0 \leq R^2 \leq 1.$$

Giá trị R^2 gắn liền với mẫu do đó nó đo sự phù hợp của mô hình (hàm hồi quy) với số liệu mẫu. Ta kỳ vọng rằng nếu mô hình có độ phù hợp cao với số liệu mẫu thì nó cũng phù hợp trong tổng thể.

Ý nghĩa của hệ số xác định bội

Với mô hình hồi quy k biến, R^2 có ý nghĩa như sau:

R^2 là tỷ lệ (hay tỷ lệ phần trăm) sự thay đổi của biến phụ thuộc được giải thích bởi các biến độc lập trong mô hình.

Với điều kiện $0 \leq R^2 \leq 1$, ta có hai trường hợp đặc biệt đó là:

➤ $R^2 = 1$ nghĩa là 100% sự thay đổi của biến phụ thuộc được giải thích bởi các biến độc lập trong mô hình

➤ $R^2 = 0$ nghĩa là các biến độc lập không giải thích được một chút nào đối với sự thay đổi của biến phụ thuộc.

Rõ ràng, trong thực tế, khi xem xét các mối quan hệ giữa các biến thông qua các mô hình hồi quy thì R^2 thường nằm trong khoảng (0,1) nhiều hơn.

Một tính chất quan trọng của R^2 là nó sẽ tăng khi ta đưa thêm biến độc lập vào mô hình. Dễ dàng thấy rằng TSS không phụ thuộc vào số biến giải thích trong mô hình nhưng RSS lại giảm. Do đó, nếu tăng số biến độc lập trong mô hình thì R^2 cũng tăng. Như vậy, việc đưa thêm một biến bất kỳ vào mô hình nói chung sẽ làm gia tăng R^2 , không kể nó có giúp giải thích thêm cho biến phụ thuộc hay không. Điều này ngụ ý rằng R^2 chưa phải là thước đo tốt khi muốn so sánh các mô hình với số biến khác nhau.

Để giải quyết vấn đề thiếu sót này, ta xem xét khái niệm R^2 hiệu chỉnh, ký hiệu là $R^2_{\text{hiệu chỉnh}}$ và được định nghĩa như sau:

$$\begin{aligned} R^2_{\text{hiệu chỉnh}} &= 1 - \frac{ESS/(n-k)}{TSS/(n-1)} = 1 - \frac{ESS(n-1)}{TSS(n-k)} \\ &= 1 - \frac{n-1}{n-k}(1 - R^2) \end{aligned}$$

Ta thấy rằng khi số biến độc lập (k - 1) tăng lên thì R^2 cũng tăng lên nhưng tăng chậm hơn so với R^2 .

Giá trị R^2 thường được sử dụng thay R^2 khi so sánh hai mô hình có cùng biến phụ thuộc nhưng số lượng biến độc lập khác nhau.

Trong thực tế, khi muốn đánh giá sự phù hợp của mô hình thì R^2 hơn vì R^2 rất dễ đưa ra một kết quả lạc quan quá mức cho sự phù hợp của mô hình hồi quy khi số lượng biến giải thích lớn hơn nhiều số lượng biến ta quan sát. Tuy nhiên, ta không thể nói

trong mọi bài toán R^2 đều đưa ra mức độ phù hợp của mô hình hồi quy một cách chính xác nhất mà phải dựa vào đặc trưng của từng bài toán cụ thể mà thực hiện tính toán sao cho phù hợp.

III. TIỀN XỬ LÝ DỮ LIỆU

1. Đọc dữ liệu

Ta dùng lệnh sau để đọc dữ liệu từ file All_GPU.csv

```
allgpus <- read.csv("C:/Users/NAM
NGUYEN/Desktop/archive/All_GPUs.csv")
```

Các lệnh để xem 6 dòng đầu tiên của dữ liệu:

```
head(allgpus)
```

```
1 Tesla G92b 738 MHz 2 Yes
2 R600 XT 1366 x 768 \n- 2 Yes
3 R600 PRO 1366 x 768 \n- 2 Yes
4 RV630 1024 x 768 \n- 2 Yes
5 RV630 1024 x 768 \n- 2 Yes
6 RV630 1024 x 768 \n- 2 Yes
Direct_X DisplayPort_Connection HDMI_Connection Integrated L2_Cache
1 DX 10.0 NA 0 No 0KB
2 DX 10 NA 0 No 0KB
3 DX 10 NA 0 No 0KB
4 DX 10 NA 0 No 0KB
5 DX 10 NA 0 No 0KB
6 DX 10 NA 0 No 0KB
Manufacturer Max_Power Memory Memory_Bandwidth Memory_Bus Memory_Speed
1 Nvidia 141 Watts 1024 MB 64GB/sec 256 Bit 1000 MHz
2 AMD 215 Watts 512 MB 106GB/sec 512 Bit 828 MHz
3 AMD 200 Watts 512 MB 51.2GB/sec 256 Bit 800 MHz
4 AMD 256 MB 36.8GB/sec 128 Bit 1150 MHz
5 AMD 45 Watts 256 MB 22.4GB/sec 128 Bit 700 MHz
6 AMD 50 Watts 256 MB 35.2GB/sec 128 Bit 1100 MHz
Memory_Type Name Notebook_GPU Open_GL
1 GDDR3 GeForce GTS 150 No 3.3
2 GDDR3 Radeon HD 2900 XT 512MB No 3.1
3 GDDR3 Radeon HD 2900 Pro No 3.1
4 GDDR4 Radeon HD 2600 XT Diamond Edition No 3.3
5 GDDR3 Radeon HD 2600 XT No 3.1
6 GDDR4 Radeon HD 2600 XT 256MB GDDR4 No 3.3
PSU Pixel_Rate Power_Connector Process ROPs Release_Date
1 450 Watt & 38 Amps 12 GPixel/s None 55nm 16 \n01-Mar-2009
2 550 Watt & 35 Amps 12 GPixel/s None 80nm 16 \n14-May-2007
3 550 Watt & 35 Amps 10 GPixel/s None 80nm 16 \n07-Dec-2007
4 3 GPixel/s None 65nm 4 \n01-Jul-2007
5 400 Watt & 25 Amps 3 GPixel/s None 65nm 4 \n28-Jun-2007
6 400 Watt & 26 Amps 3 GPixel/s None 65nm 4 \n26-Jun-2007
Release_Price Resolution_WxH SLI_Crossfire Shader TMUs Texture_Rate
1 2560x1600 Yes 4 64 47 GTexel/s
2 2560x1600 Yes 4 16 12 GTexel/s
3 2560x1600 Yes 4 16 10 GTexel/s
4 2560x1600 Yes 4 8 7 GTexel/s
5 2560x1600 Yes 4 8 6 GTexel/s
```

Hình 1: 6 dòng đầu của dữ liệu

2. Làm sạch dữ liệu

Dựa trên nguồn sau có tính tham khảo tốt và phù hợp với nhu cầu sử dụng của đề tài:

[Các loại card màn hình \(VGA\) và thông số quan trọng thường gặp – GEARVN.COM](http://www.gearvn.com)

Nhóm đã thống nhất bằng cách tạo một bảng dữ liệu mới gồm những dữ liệu cần sử

dụng là: Boost_Clock, Core_Speed, Memory, Memory_Bandwidth, Memory_Bus,

Memory_Speed, Release_Price, Texture_Rate bằng cách dùng lệnh sau: new_df<-

ALL_GPUs[,c("Boost_Clock", "Core_Speed", "Memory", "Memory_Bandwidth",

"Memory_Bus", "Memory_Speed", "Release_Price", "Texture_Rate")] >head(new_df)


```

Boost_Clock Core_Speed Memory Memory_Bandwidth Memory_Bus Memory_Speed
1          738 MHz 1024 MB          64GB/sec      256 Bit      1000 MHz
2          \n-   512 MB          106GB/sec      512 Bit      828 MHz
3          \n-   512 MB          51.2GB/sec      256 Bit      800 MHz
4          \n-   256 MB          36.8GB/sec      128 Bit      1150 MHz
5          \n-   256 MB          22.4GB/sec      128 Bit      700 MHz
6          \n-   256 MB          35.2GB/sec      128 Bit      1100 MHz

Release_Price Texture_Rate
1          47 GTexel/s
2          12 GTexel/s
3          10 GTexel/s
4           7 GTexel/s
5           6 GTexel/s
6           6 GTexel/s

```

Hình 2: Bảng dữ liệu gồm 6 biến sau khi lọc lại

Lúc này, trong bảng dữ liệu vẫn còn đơn vị của các biến nên các dữ liệu có định dạng chuỗi (string), ta cần xóa các đơn vị đó và biến dữ liệu thành dạng số.

	Boost_Clock	Core_Speed	Memory	Memory_Bandwidth	Memory_Bus	Memory_Speed	Release_Price	Texture_Rate
1	NA	738.0000	1024.000	64.0000	256.0000	1000.000	NA	47.00000
2	NA	946.8939	512.000	106.0000	512.0000	828.000	NA	12.00000
3	NA	946.8939	512.000	51.2000	256.0000	800.000	NA	10.00000
4	NA	946.8939	256.000	36.8000	128.0000	1150.000	NA	7.00000
5	NA	946.8939	256.000	22.4000	128.0000	700.000	NA	6.00000
6	NA	946.8939	256.000	35.2000	128.0000	1100.000	NA	6.00000
7	NA	870.0000	2048.000	134.4000	256.0000	1050.000	NA	35.00000
8	NA	946.8939	256.000	51.2000	256.0000	800.000	NA	7.00000
9	NA	946.8939	2048.000	160.0000	256.0000	1250.000	NA	62.00000
10	NA	946.8939	64.000	2.9000	64.0000	366.000	NA	90.27463
11	NA	946.8939	128.000	5.2000	128.0000	325.000	NA	2.00000
12	NA	650.0000	6144.000	177.6000	384.0000	925.000	NA	36.00000

Hình 3: Dữ liệu sau khi chuyển thành dạng số

Sau khi hoàn tất chuyển đổi dữ liệu, ta bắt đầu thống kê dữ liệu khuyết bằng lệnh: `apply(is.na(df),2,sum)`

Kết quả:

```

Boost_Clock      Core_Speed      Memory Memory_Bandwidth
1960              936          420          125
Memory_Bus      Memory_Speed Texture_Rate Release_Price
62              105          544          2871

```

Hình 4: Thống kê dữ liệu khuyết

Quan sát ta thấy có nhận xét sau:

- 1960 dữ liệu thuộc biến Boost_Clock bị khuyết.
- 936 dữ liệu thuộc biến Core_Speed bị khuyết.
- 420 dữ liệu thuộc biến Memory bị khuyết.
- 125 dữ liệu thuộc biến Memory_Bandwidth bị khuyết.
- 62 dữ liệu thuộc biến Memory_Bus bị khuyết.
- 105 dữ liệu thuộc biến Memory_Speed bị khuyết.
- 544 dữ liệu thuộc biến Texture_Rate bị khuyết.

- 2871 dữ liệu thuộc biến Release_Price bị khuyết.

Phương pháp xử lý dữ liệu khuyết:

- Vì số lượng dữ liệu khuyết của biến Boost_Clock chiếm hơn 50% tổng số dữ liệu của biến nên nhóm xin đề xuất loại bỏ biến này để không gây ảnh hưởng đến dữ liệu chung.
- Ngoài biến Release_Price là nhân tố chính để dự đoán số liệu trong đề tài này, tất cả các dữ liệu khuyết của các biến còn lại sẽ được thay thế bằng giá trị trung bình của các quan sát còn lại của biến.

Ta bắt đầu tạo một bộ dữ liệu mới chỉ bao gồm những giá trị Release_Price

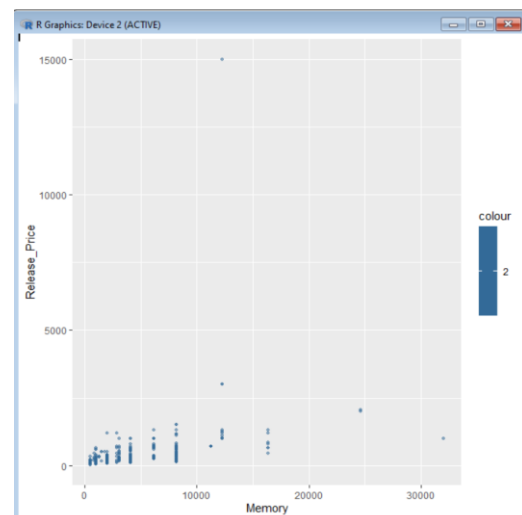
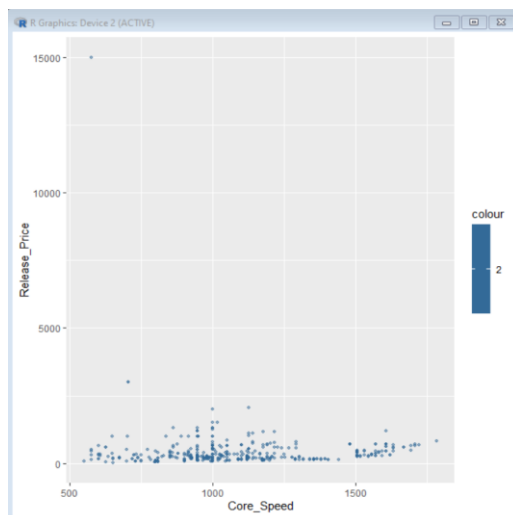
	Core_Speed	Memory	Memory_Bandwidth	Memory_Bus	Memory_Speed	Release_Price	Texture_Rate
1	946.8939	2872.769	480.0000	384.0000	1250	1199.00	343
2	946.8939	12288.000	547.2000	96.0000	1425	1199.00	354
3	705.0000	12288.000	672.0000	384.0000	1750	2999.00	420
4	705.0000	12288.000	672.0000	384.0000	1750	2999.00	420
5	1140.0000	12288.000	336.6000	384.0000	1753	1099.00	240
6	1127.0000	24576.000	673.2000	384.0000	1753	2059.00	467
7	1000.0000	24576.000	673.2000	384.0000	1753	1999.00	418
8	1000.0000	12288.000	336.6000	384.0000	1753	999.00	209
9	1127.0000	12288.000	336.6000	384.0000	1753	1029.99	233
10	1000.0000	12288.000	384.0000	3072.0000	500	999.00	320

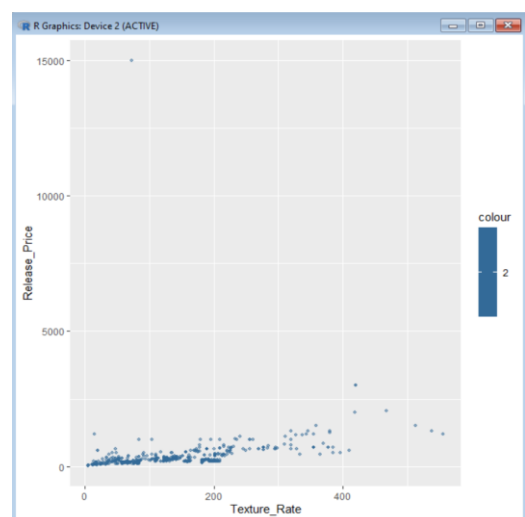
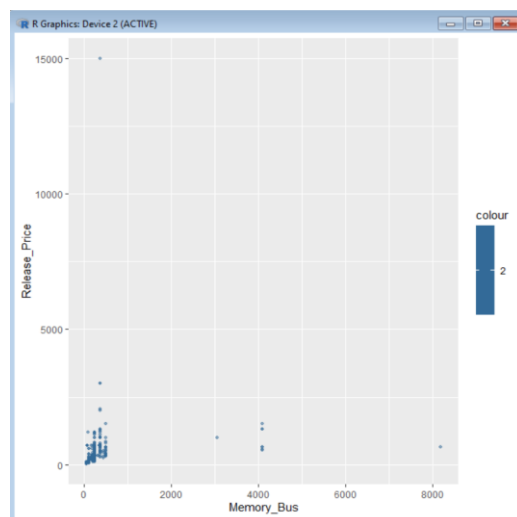
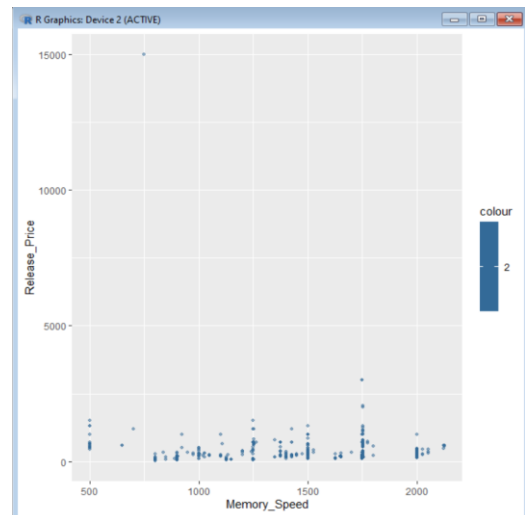
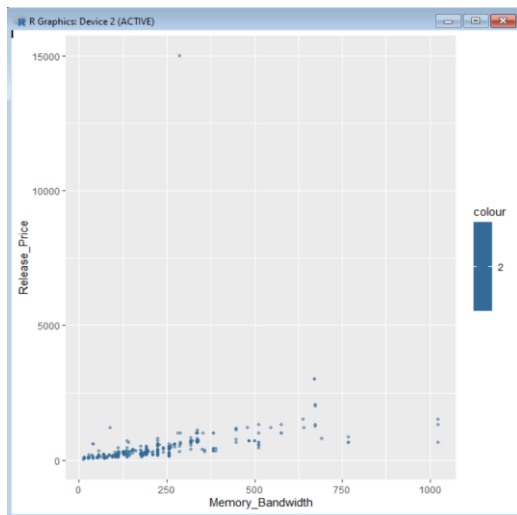
Hình 5: Dữ liệu gồm các hàng chứa biến

IV. THỐNG KÊ TẢ

1. Scatter plots

Dưới đây là biểu đồ phân tán dữ liệu được vẽ theo bộ dữ liệu đã làm mới



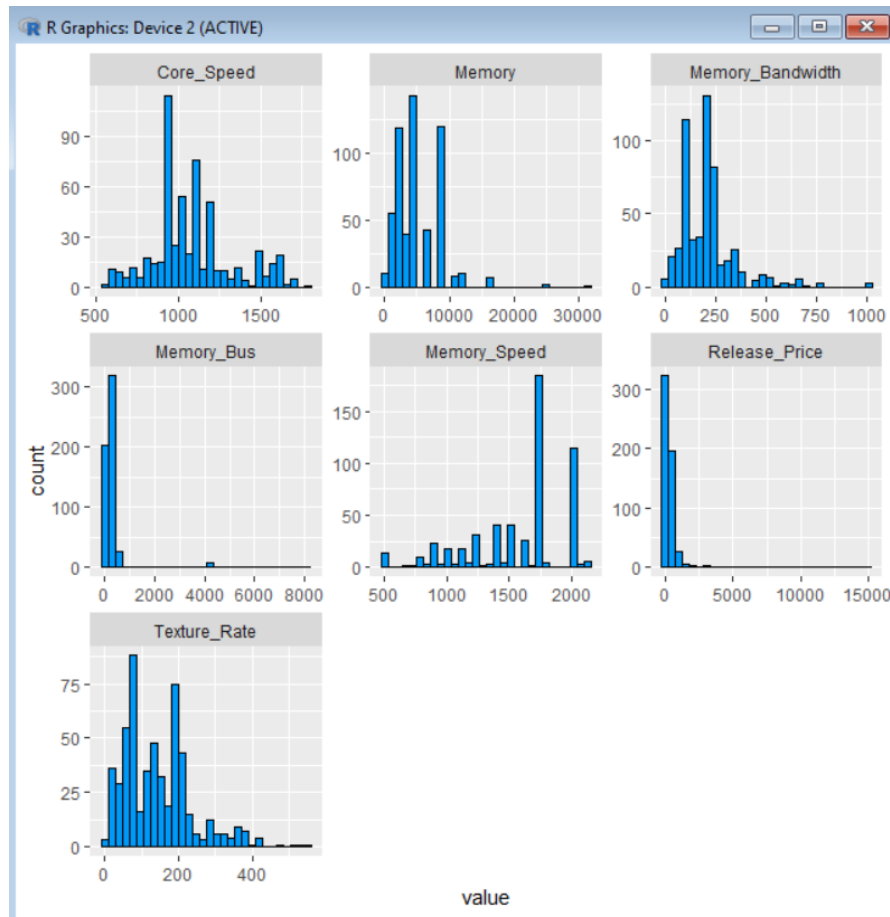


Hình 6: Biểu đồ phân tán của 6 biến đối với biến Release_Price

Nhận xét: dựa vào các biểu đồ, giá trị “Release_Price” dao động trong khoảng 49\$ tới 14.9999\$. Tuy nhiên, ta chỉ có thể vẽ đường hồi quy cho các biến “Texture_Rate”, “Memory_Bandwidth”. Nghĩa là, “Release_Price” có mối quan hệ tuyến tính mạnh với các biến này. Vì thế, ta có thể kết luận rằng mối quan hệ giữa các biến này là mối quan hệ tuyến tính giữa các biến độc lập là thuận và tương đối mạnh khi mà ta có thể vẽ được đường thẳng hồi quy với xu hướng đi lên.

2. Histogram

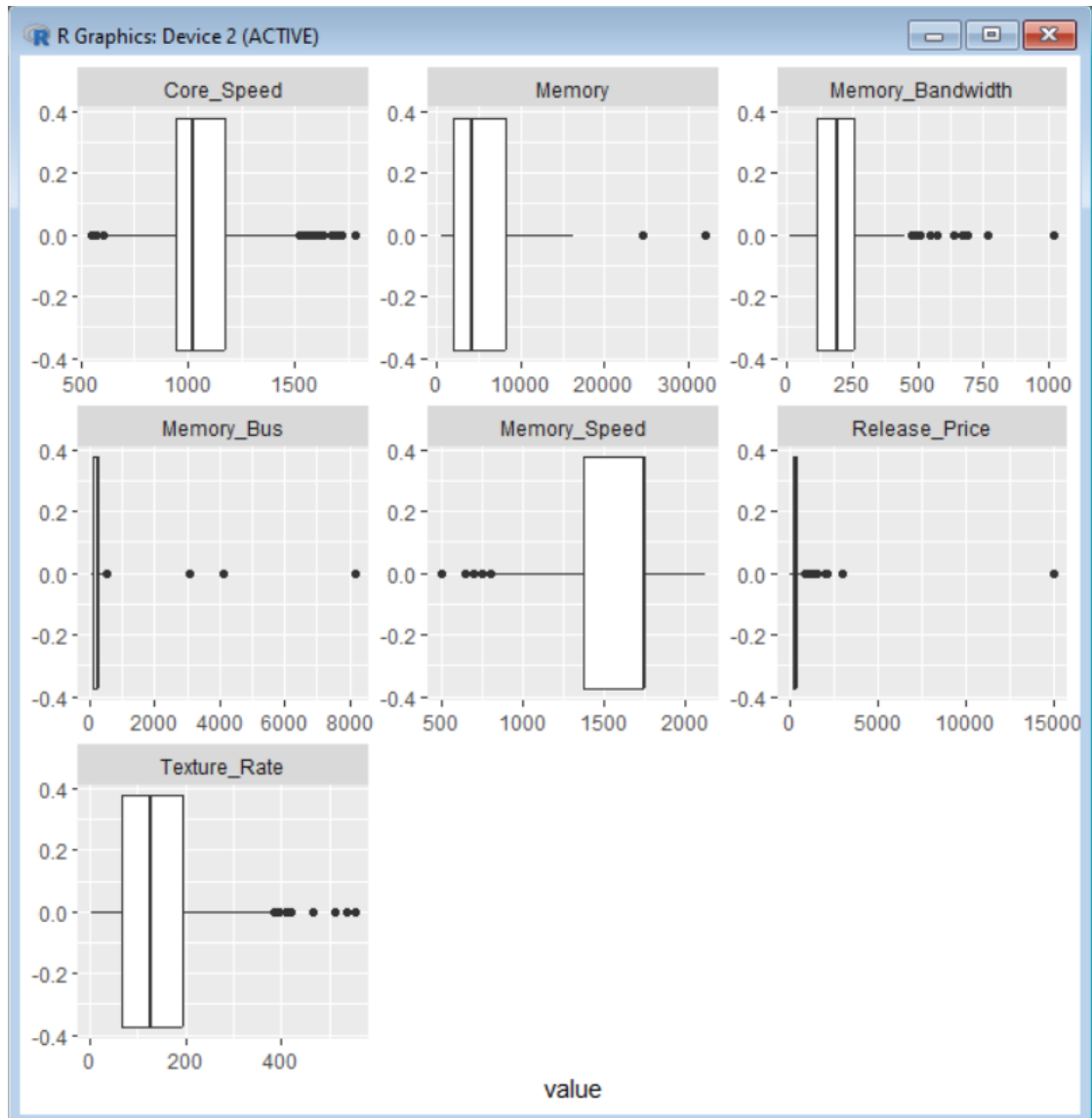
Nhóm đã tiến hành vẽ biểu đồ tần suất (Histogram) nhằm biểu thị mức độ phân phối của bộ dữ liệu được cung cấp.



Hình 7: Biểu đồ cột của 7 biến

Nhận xét: Dựa trên các Histogram, ta có thể thấy rằng giá trị của biến “Release_Price” chủ yếu giao động ở mức 0 đến 800 (đơn vị \$) và tập trung chủ yếu trong khoảng 40 đến 250 (đơn vị \$). Giá trị có số lượng ít nhất là trong khoảng 800 đến 900 (đơn vị \$).

3. Boxplot



Hình 8: Biểu đồ hộp của 7 biến

Nhận xét: Nhìn chung, ta có thể kết luận rằng “Memory_Bus” thể hiện mức độ lệch cao nhất tiếp đến là “Memory”. Đây có thể là do sự xuất hiện của một số yếu tố khác trong các biến này. Bên cạnh đó, hầu như các biến còn lại đều có phân phối chuẩn sau khi chuẩn hóa.

V. THỐNG KÊ SUY DIỄN

5.1 Yếu tố nào sẽ tác động đến “Release_Price” nhiều nhất?

5.1.1 Chia dữ liệu

Trước khi xây dựng mô hình, ta bắt đầu chia bộ dữ liệu chứa tổng cộng 556 giá trị quan trắc ra thành ba bộ dữ liệu mới như sau:

- Bộ thứ nhất: chứa 70% dữ liệu, tương ứng với 389 giá trị quan trắc
- Bộ thứ hai: chứa 30% dữ liệu, tương ứng với 167 giá trị quan trắc.

5.1.2 Mô hình phù hợp

Nhiệm vụ của chúng ta là xây dựng một mô hình cho các yếu tố ảnh hưởng lên biến Release_Price. Với biến chúng ta quan tâm (Release_Price) là biến phụ thuộc và các biến còn lại được xem là biến độc lập.

$$\text{Release_Price} = \beta_0 + \beta_1 \cdot \text{Core_Speed} + \beta_2 \cdot \text{Memory} + \beta_3 \cdot \text{Memory_Bandwidth} + \beta_4 \cdot \text{Memory_Bus} + \beta_5 \cdot \text{Memory_Speed} + \beta_6 \cdot \text{Texture_Rate} + \varepsilon$$

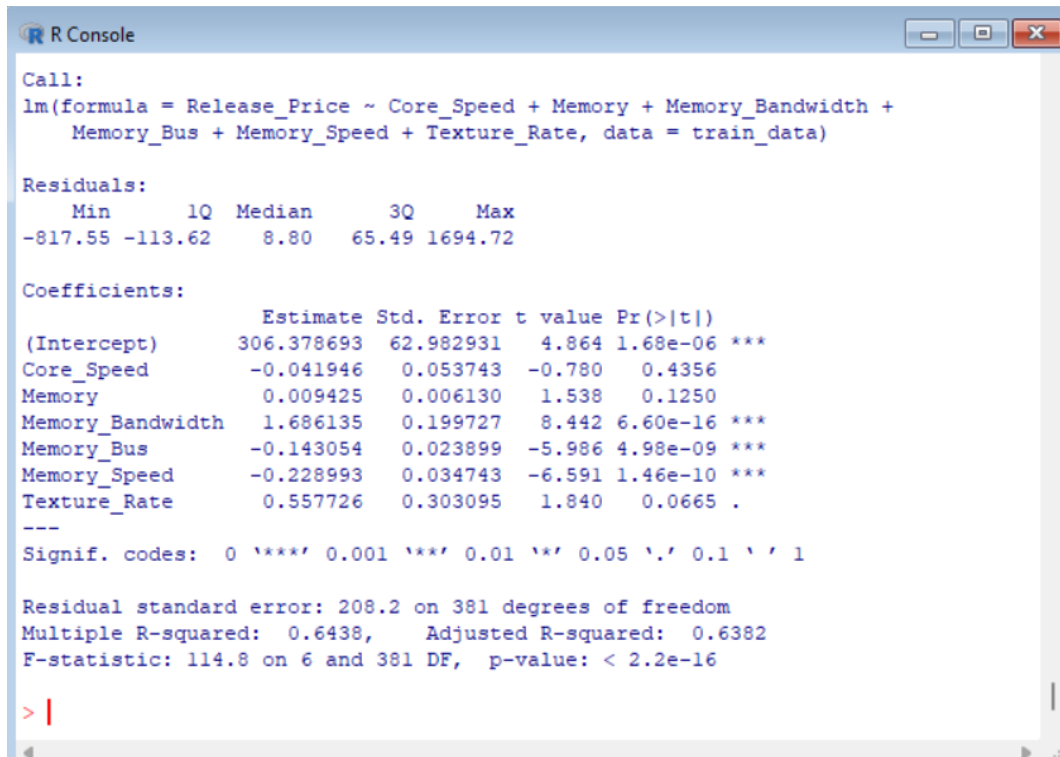
Trước hết, ta đặc biệt chú ý các giá trị ngoại lai 14.999 của biến “Release_Price”

210	214	215	218	219	219.99	220	221	224	229
3	1	1	1	2	2	8	2	1	9
229.99	230	235	235.02	239	239.99	240	243.44	245	249
3	2	1	1	4	1	11	1	3	10
249.99	250	255	259.99	260	265	269	269.99	270	278
6	15	1	2	2	1	2	1	3	1
279	279.99	289	289.99	292	298	299	299.99	300	300.64
5	1	1	4	1	3	9	3	1	1
318	319.49	320	329	329.99	330	339	339.99	349	349.99
2	1	2	7	4	1	1	1	6	3
358	359	359.99	369	369.99	370	379	380	398	399
1	1	1	2	1	1	4	1	3	6
400	419	429	430	438	439	449	449.99	450	458
1	3	2	1	2	1	7	1	4	1
480	485	499	500	549	549.99	559	560	569.99	579.99
1	2	6	1	3	4	1	1	3	3
585	590	599	609	619	649	649.99	658	659.98	669
2	2	4	1	1	14	1	2	1	3
679	690	699	699.99	719	749	779	799	799.99	829
3	1	17	3	1	2	1	1	2	1
858	988	999	999.99	1029.99	1099	1099.98	1158	1159.98	1199
1	1	8	1	1	1	1	1	1	4
1249	1298	1299	1499	1998	2059	2999	14999		
1	2	2	2	1	1	2	1		

Hình 9: Nhận diện điểm ngoại lai của biến Release_Price

Nhằm xây dựng một mô hình có thể đánh giá nhiều yếu tố tác động đến “Release_Price”, ta thấy rằng biến quan tâm “Release_Price” có thể được xem như một biến phụ thuộc trong khi các biến khác chính là các biến độc lập tác động tới biến này. Đồng thời, tiến hành xóa bỏ điểm ngoại lai để đảm bảo độ chính xác của mô hình.

Bắt đầu với mô hình gốc (model_1) như sau:



```

R Console

Call:
lm(formula = Release_Price ~ Core_Speed + Memory + Memory_Bandwidth +
    Memory_Bus + Memory_Speed + Texture_Rate, data = train_data)

Residuals:
    Min       1Q   Median       3Q      Max
-817.55 -113.62    8.80   65.49 1694.72

Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept)   306.378693   62.982931   4.864 1.68e-06 ***
Core_Speed    -0.041946    0.053743  -0.780  0.4356
Memory         0.009425    0.006130   1.538  0.1250
Memory_Bandwidth 1.686135    0.199727   8.442 6.60e-16 ***
Memory_Bus    -0.143054    0.023899  -5.986 4.98e-09 ***
Memory_Speed  -0.228993    0.034743  -6.591 1.46e-10 ***
Texture_Rate   0.557726    0.303095   1.840  0.0665 .
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 208.2 on 381 degrees of freedom
Multiple R-squared:  0.6438,    Adjusted R-squared:  0.6382
F-statistic: 114.8 on 6 and 381 DF,  p-value: < 2.2e-16

> |

```

Hình 10: Mô hình gốc

Nhận xét: có tổng cộng 6 biến gồm: Core_Speed, Memory, Memory_Bandwidth, Memory_Bus, Memory_Speed, Texture_Rate. Kết quả phân tích cho thấy sự thay đổi của của 4 biến: Memory, Memory_Bandwidth, Memory_Speed, Texture_Rate có tác động sâu sắc đến “Release_Price”. Trong khi đó, biến Core_Speed và Memory_Bus không ảnh hưởng nhiều đến “Release_Price”.

Mô hình cải thiện thứ nhất: là mô hình đã loại bỏ biến Texture_Rate từ bộ dữ liệu gốc

```

R Console

Call:
lm(formula = Release_Price ~ Core_Speed + Memory + Memory_Bandwidth +
    Memory_Bus + Memory_Speed, data = train_data)

Residuals:
    Min       1Q   Median       3Q      Max
-914.83 -109.01    9.46   62.59 1709.25

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  290.724384   62.600324   4.644 4.70e-06 ***
Core_Speed   -0.009124    0.050855  -0.179  0.8577
Memory        0.014297    0.005546   2.578  0.0103 *
Memory_Bandwidth 1.947440    0.140885  13.823 < 2e-16 ***
Memory_Bus   -0.154003    0.023219  -6.633 1.13e-10 ***
Memory_Speed -0.239865    0.034344  -6.984 1.28e-11 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 208.8 on 382 degrees of freedom
Multiple R-squared:  0.6406,    Adjusted R-squared:  0.6359
F-statistic: 136.2 on 5 and 382 DF,  p-value: < 2.2e-16

> |

```

Hình 11: Mô hình cải thiện thứ nhất

Nhận xét: ta thấy rằng giá trị “Adjusted R-squared” thấp hơn so với mô hình gốc nên ta không nhận giá trị này để cho là một giá trị hiệu quả hơn.

Mô hình cải thiện thứ hai: là mô hình đã loại bỏ biến Core_Speed từ bộ dữ liệu gốc

```

Call:
lm(formula = Release_Price ~ Memory + Memory_Bandwidth + Memory_Bus +
    Memory_Speed + Texture_Rate, data = train_data)

Residuals:
    Min       1Q   Median       3Q      Max
-823.11 -115.05    3.67   58.22 1713.91

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  278.438740   51.793802   5.376 1.33e-07 ***
Memory        0.008836    0.006080   1.453  0.1470
Memory_Bandwidth 1.740236    0.187217   9.295 < 2e-16 ***
Memory_Bus   -0.146699    0.023427  -6.262 1.02e-09 ***
Memory_Speed -0.237889    0.032803  -7.252 2.30e-12 ***
Texture_Rate  0.479213    0.285768   1.677  0.0944 .
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 208 on 382 degrees of freedom
Multiple R-squared:  0.6432,    Adjusted R-squared:  0.6385
F-statistic: 137.7 on 5 and 382 DF,  p-value: < 2.2e-16

```

Hình 12: Mô hình cải thiện thứ hai

Nhận xét: ta thấy rằng giá trị “Adjusted R-squared” gần như tương đương so với mô hình gốc nên ta nhận thấy rằng giá trị này chính là một giá trị tối ưu và chọn làm mô hình phù hợp nhất.

Mô hình cải thiện thứ ba: là mô hình đã loại bỏ biến Memory_Bandwidth từ bộ dữ liệu gốc

```
Call:
lm(formula = Release_Price ~ Core_Speed + Memory + Memory_Bus +
    Memory_Speed + Texture_Rate, data = train_data)

Residuals:
    Min       1Q   Median       3Q      Max
-741.91 -108.13   -0.97   78.76 1811.91

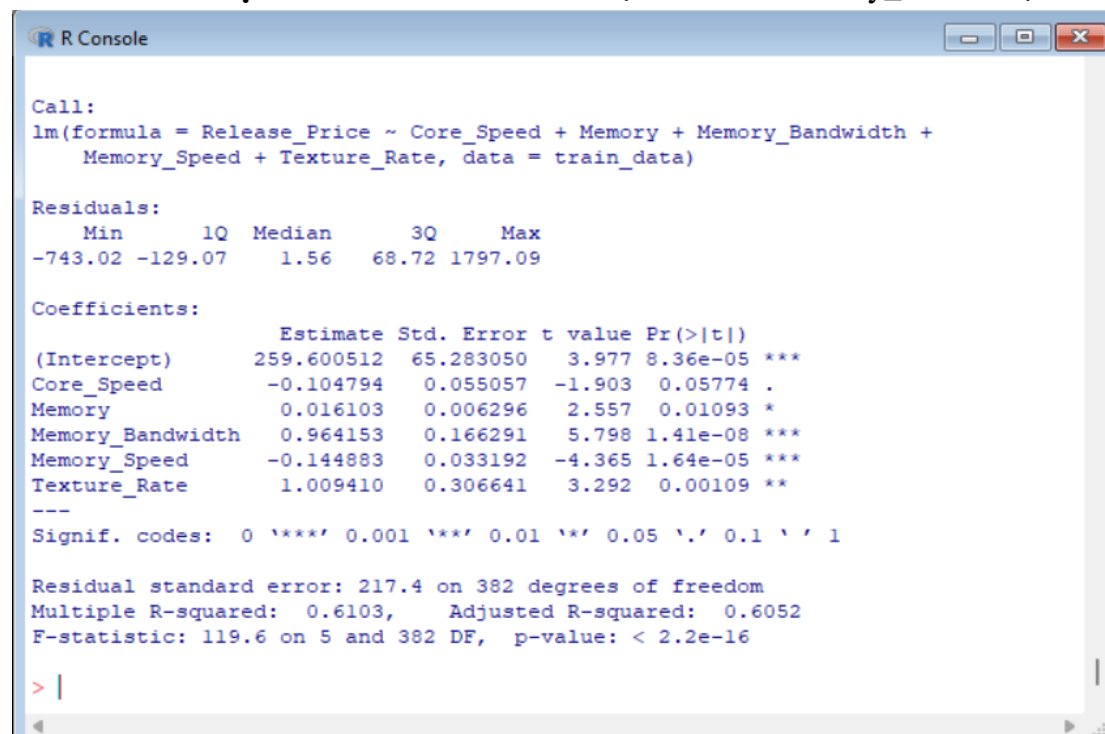
Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  444.473373   66.179855   6.716 6.78e-11 ***
Core_Speed   -0.199407    0.054843  -3.636 0.000315 ***
Memory        0.018240    0.006572   2.775 0.005787 **
Memory_Bus   -0.021206    0.020727  -1.023 0.306916
Memory_Speed -0.189224    0.037455  -5.052 6.79e-07 ***
Texture_Rate  2.377014    0.231911  10.250 < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 226.5 on 382 degrees of freedom
Multiple R-squared:  0.5771,    Adjusted R-squared:  0.5716
F-statistic: 104.3 on 5 and 382 DF,  p-value: < 2.2e-16
```

Hình 13: Mô hình cải thiện thứ ba

Nhận xét: ta thấy rằng giá trị “Adjusted R-squared” thấp hơn so với mô hình cải thiện thứ hai (model_fix2) nên ta không nhận giá trị này.

Mô hình cải thiện thứ tư: là mô hình đã loại bỏ biến Memory_Bus từ bộ dữ liệu gốc



```
R Console

Call:
lm(formula = Release_Price ~ Core_Speed + Memory + Memory_Bandwidth +
    Memory_Speed + Texture_Rate, data = train_data)

Residuals:
    Min       1Q   Median       3Q      Max
-743.02 -129.07    1.56   68.72 1797.09

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  259.600512   65.283050   3.977 8.36e-05 ***
Core_Speed   -0.104794    0.055057  -1.903  0.05774 .
Memory        0.016103    0.006296   2.557  0.01093 *
Memory_Bandwidth  0.964153    0.166291   5.798 1.41e-08 ***
Memory_Speed -0.144883    0.033192  -4.365 1.64e-05 ***
Texture_Rate  1.009410    0.306641   3.292  0.00109 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 217.4 on 382 degrees of freedom
Multiple R-squared:  0.6103,    Adjusted R-squared:  0.6052
F-statistic: 119.6 on 5 and 382 DF,  p-value: < 2.2e-16

> |
```

Hình 14: Mô hình cải thiện thứ tư

Nhận xét: tương tự với mô hình cải thiện số ba, mô hình cải thiện thứ tư (model_fix4) cũng không cho một giá trị tối ưu nhất so với mô hình cải thiện số hai.

KẾT LUẬN: thông qua việc so sánh giữa các mô hình cải thiện với nhau, nhóm đã đi đến thống nhất rằng mô hình cải thiện số hai (model_fix2) chính là mô hình đạt hiệu quả cao nhất vì có “Adjusted R-squared” lớn nhất, có nghĩa là sự biến đổi của “Release_Price” phụ thuộc rất nhiều vào các biến độc lập.

5.1.3 Mô hình suy diễn

Dựa vào kết luận ở trên về mô hình cải thiện số hai (model_fix2), nhóm đề xuất một mô hình hồi quy tuyến tính mà ở đó các biến độc lập sẽ tác động lên biến phụ thuộc “Release_Price” bằng phương trình sau:

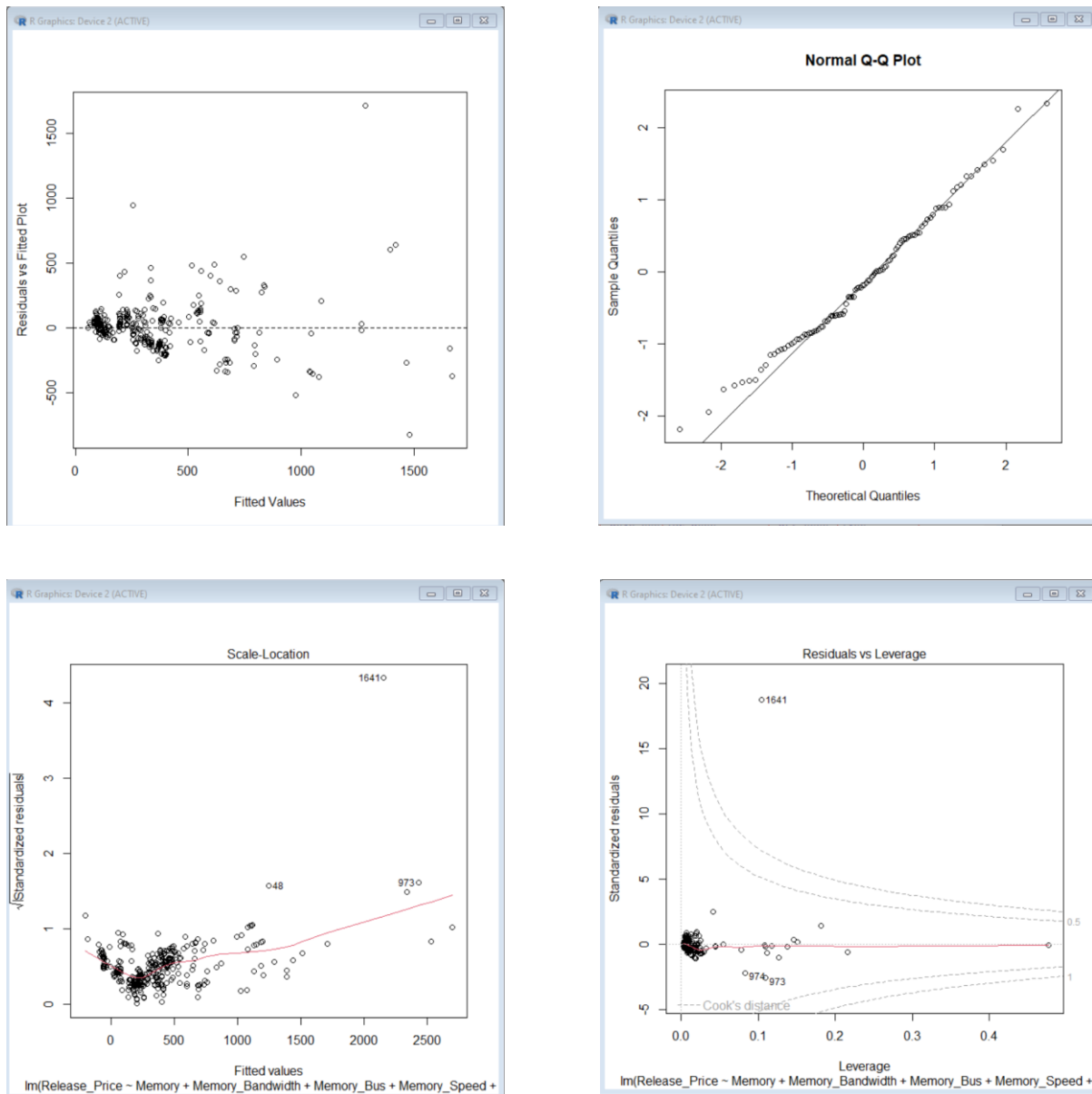
$$\text{Release_Price} = 278.438740 + 0.008836 \cdot \text{Memory} + 1.740236 \cdot \text{Memory_Bandwidth} - 0.146699 \cdot \text{Memory_Bus} - 0.237889 \cdot \text{Memory_Speed} + 0.479213 \cdot \text{Texture_Rate}$$

Nhận xét: quan sát thấy rằng giá trị p -value tương ứng với F-statistic thấp hơn 2.2×10^{-16} . Điều này có nghĩa rằng dữ liệu mang ý nghĩa thống kê tốt và chặt chẽ. Không những vậy, sẽ có ít nhất một dự đoán trong mô hình có ý nghĩa giải thích rất tốt cho biến “Release_Price”.

Một cách tổng quan, hệ số hồi quy (β_i) và giá trị p -value chính là 2 tác động cụ thể của các biến độc lập. Nhận thấy rằng giá trị p -value của “Memory_Speed” và “Memory_Bus” rất nhỏ (lần lượt là 2.30×10^{-12} và 1.02×10^{-9}), điều này có nghĩa tác động của 2 biến này đến “Release_Price” rất đáng kể. Bên cạnh đó, các biến “Memory_Bandwidth”, “Memory” và “Texture_Rate” không ảnh hưởng quá nhiều đến “Release_Price”.

Xét theo một chiều hướng khác, hệ số hồi quy β_i của biến dự báo có thể được xem là có tác động ở mức vừa phải tới biến phụ thuộc “Release_Price”. Cụ thể hơn, với $\beta_1 = 0.008836$, khi ta tăng 1 đơn vị của β_1 , giá trị “Release_Price” sẽ tăng 0.008836\$ (giả sử các biến khác đang là hằng số). Một ví dụ khác là với $\beta_2 = 1.740236$, khi ta tăng “Memory” lên 1 đơn vị thì giá trị “Release_Price” sẽ tăng 1.740236\$ (giả sử các biến khác đang là hằng số).

5.1.4 Giả định về mô hình cần kiểm tra



Hình 15: Giả định về mô hình

Nhận xét:

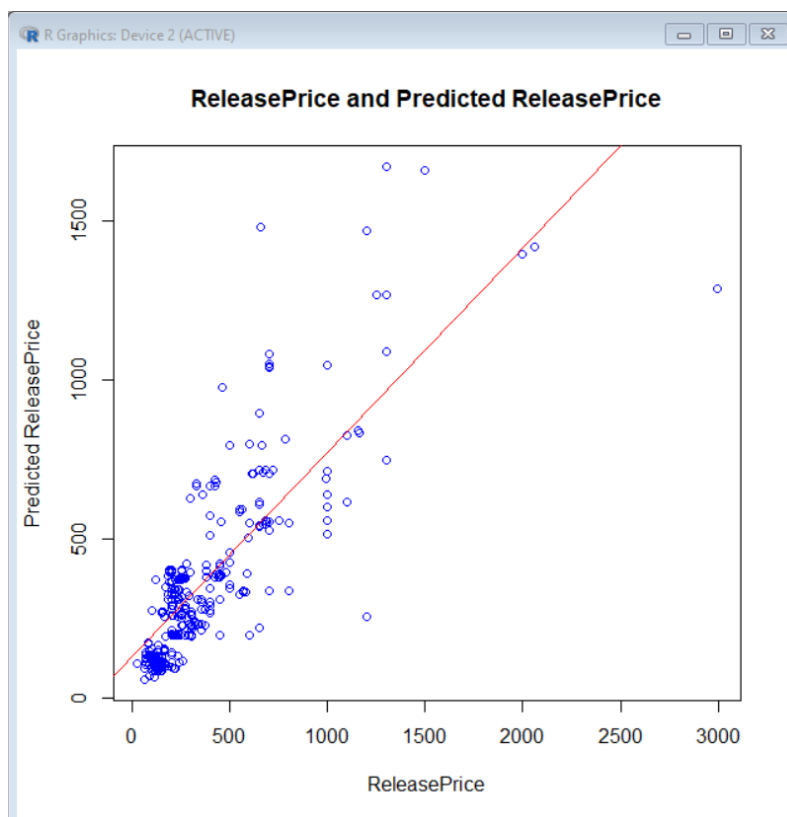
- **Residuals vs Fitted:** các giá trị dự đoán với các giá trị dư (sai số) tương ứng, được sử dụng để kiểm tra giả định rằng các sai số có kỳ vọng bằng 0 và tính đồng nhất của các phương pháp sai số. Dựa vào đồ thị, chúng ta thấy đường màu đen không gần với đường $y = 0$ nên giả định sai số với kỳ vọng bằng 0 không được thỏa mãn. Các sai số không được phân bố ngẫu nhiên dọc theo đường màu đen, vì vậy giả định rằng phương sai là một hằng số không được thỏa mãn.
- **Normal Q-Q:** các giá trị sai số được chuẩn hóa, cho phép kiểm tra giả định về

phân bố chuẩn của sai số. Dựa trên biểu đồ, chúng ta thấy rằng các sai số chủ yếu tập trung vào đường dự kiến phân phối chuẩn, do đó phân phối chuẩn của các sai số được coi là thỏa mãn.

- **Scale-Location:** căn bậc hai của các giá trị dư được chuẩn hóa bằng giá trị dự đoán, được sử dụng để kiểm tra giả định rằng phương sai của sai số là không đổi. Dựa vào đồ thị ta thấy, các sai số không nằm rải rác ngẫu nhiên dọc theo đường màu đen nên giả sử phương sai là một hằng số thì không thỏa mãn.
- **Residuals vs Leverage:** cho phép quan sát và xác định những điểm có tác động lớn (giá trị quan trọng đáng kể) nếu xuất hiện trong bộ dữ liệu. Những điểm này đã xuất hiện trong đồ thị và ta có thể dễ dàng nhận ra các điểm 973, 974 và 1641 là những giá trị quan trọng có tác động đáng kể tới bộ dữ liệu.

5.1.5 Dự đoán cho giá bán của GPU

Đồ thị dưới đây biểu thị cho mô hình hồi quy tuyến tính được áp dụng cho train_data (70% dữ liệu).



Hình 16: Mô hình hồi quy tuyến tính

Nhận xét: Dựa vào đồ thị, chúng ta có thể thấy các giá trị quan trọng phân tán rộng ra xa đường hồi quy, chứng tỏ giá trị dự đoán và giá trị quan sát ban đầu có mối quan hệ tuyến tính trung bình. Chúng ta có thể kết luận rằng mô hình hồi quy mà nhóm đang sử dụng thực sự tốt cho việc dự đoán.

Ta bắt đầu sử dụng mô hình cải thiện số hai (model_fix2) để dự đoán cho 30% giá trị “Release_Price” còn lại.

	Memory_Bus	Memory_Speed	Release_Price	Texture_Rate	predictions_2
148	256.0000	1753	369.99	142	318.61435
149	256.0000	1753	299.00	113	286.62122
150	256.0000	1753	299.00	113	304.71718
151	128.0000	1753	229.99	83	95.76779
152	128.0000	1753	209.99	86	97.20543
153	128.0000	1753	229.99	87	97.68464
154	128.0000	1753	219.99	84	96.24701
155	128.0000	1753	199.00	75	91.93409
156	128.0000	1755	174.00	67	87.79863
157	384.0000	1502	649.00	173	476.72967
158	128.0000	1753	214.00	80	94.33015
159	128.0000	1750	179.00	67	88.46600
160	128.0000	1653	169.00	67	100.75173
161	128.0000	1653	318.00	114	325.48764
162	128.0000	1653	159.00	57	95.95960
163	128.0000	1653	179.00	67	100.75173
164	256.0000	1502	269.00	105	286.63597
165	256.0000	1502	249.00	99	283.76070
166	128.0000	1250	119.00	35	127.33994

Hình 17: Dự đoán cho 30% giá trị còn lại của Release_Price

Sau khi dự đoán xong giá trị cho 30% số lượng “Release_Price” theo mô hình cải thiện số hai, ta bắt đầu so sánh với giá trị thực tế (đặt tên là actual) của 30% giá trị “Release_Price” như sau:

```
> predictions <- predict(model_fix2, newdata = test_data)
> actual <- test_data$Release_Price
>
> rmse <- sqrt(mean((predictions - actual)^2))
> rmse
[1] 161.8106
> data_range<-range(test_data)
> data_range
[1] 5 32000
> view(test_data)
> data_range<-range(test_data$Release_Price)
> data_range
[1] 49 1499
```

Hình 18: So sánh với 30% dữ liệu ban đầu

Nhận xét: ta thấy rằng sai số trung bình bình phương (RMSE) là 161.8106, có thể xem rằng đây là một giá trị tương đối nhỏ so với vùng dữ liệu được thống kê [49;1499] nên ta có khẳng định rằng mô hình hồi quy tuyến tính được sử dụng tương đối hợp lý và chính xác.

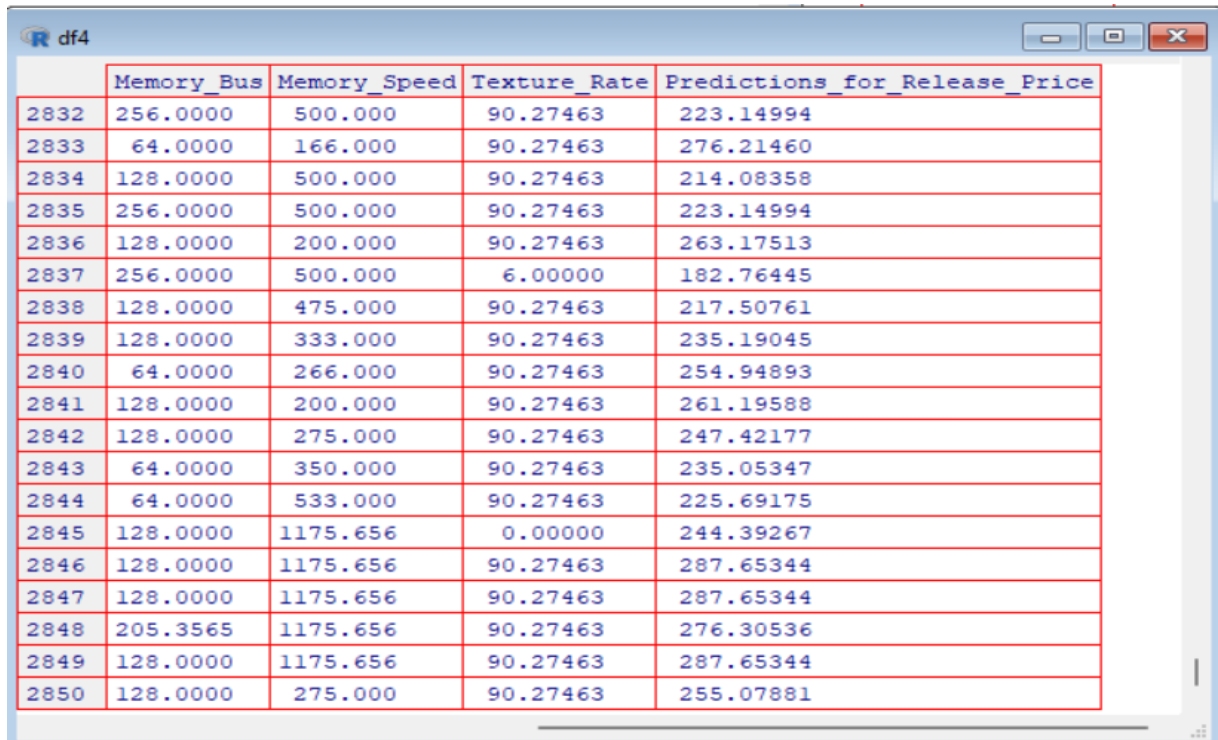
Một lần nữa, mô hình cải thiện số hai (model_fix2) được khẳng định có thể dự báo tương đối chính xác cho giá bán ra của GPU. Do vậy, nhóm đã tiến hành áp dụng mô hình này với bộ dữ liệu mới mà trong đó có hơn 2800 biến “Release_Price” đang bị khuyết và cần dự đoán số liệu.

Đầu tiên, ta trích xuất bộ dữ liệu mà trong đó, biến “Release_Price” đang không có giá trị.

	Memory	Memory_Bandwidth	Memory_Bus	Memory_Speed	Release_Price
2833	128.000	1.3000	64.0000	166.000	NA
2834	256.000	16.0000	128.0000	500.000	NA
2835	256.000	32.0000	256.0000	500.000	NA
2836	256.000	3.2000	128.0000	200.000	NA
2837	256.000	32.0000	256.0000	500.000	NA
2838	128.000	15.2000	128.0000	475.000	NA
2839	256.000	5.3000	128.0000	333.000	NA
2840	256.000	2.1000	64.0000	266.000	NA
2841	32.000	3.2000	128.0000	200.000	NA
2842	256.000	4.4000	128.0000	275.000	NA
2843	128.000	2.8000	64.0000	350.000	NA
2844	2872.769	8.5000	64.0000	533.000	NA
2845	2872.769	137.3508	128.0000	1175.656	NA
2846	2872.769	137.3508	128.0000	1175.656	NA
2847	2872.769	137.3508	128.0000	1175.656	NA
2848	2872.769	137.3508	205.3565	1175.656	NA
2849	2872.769	137.3508	128.0000	1175.656	NA
2850	256.000	8.8000	128.0000	275.000	NA

Hình 19: Bộ dữ liệu thiếu các giá trị của biến Release_Price

Tiếp theo, bắt đầu áp dụng mô hình cải thiện số hai (model_fix2) cho bộ dữ liệu và có được kết quả như sau:



	Memory_Bus	Memory_Speed	Texture_Rate	Predictions_for_Release_Price
2832	256.0000	500.000	90.27463	223.14994
2833	64.0000	166.000	90.27463	276.21460
2834	128.0000	500.000	90.27463	214.08358
2835	256.0000	500.000	90.27463	223.14994
2836	128.0000	200.000	90.27463	263.17513
2837	256.0000	500.000	6.00000	182.76445
2838	128.0000	475.000	90.27463	217.50761
2839	128.0000	333.000	90.27463	235.19045
2840	64.0000	266.000	90.27463	254.94893
2841	128.0000	200.000	90.27463	261.19588
2842	128.0000	275.000	90.27463	247.42177
2843	64.0000	350.000	90.27463	235.05347
2844	64.0000	533.000	90.27463	225.69175
2845	128.0000	1175.656	0.00000	244.39267
2846	128.0000	1175.656	90.27463	287.65344
2847	128.0000	1175.656	90.27463	287.65344
2848	205.3565	1175.656	90.27463	276.30536
2849	128.0000	1175.656	90.27463	287.65344
2850	128.0000	275.000	90.27463	255.07881

Hình 20: Kết quả dự đoán

5.1.6 Kết luận

Sau khi sử dụng phương pháp hồi quy tuyến tính bội, nhóm nhận ra các yếu tố ảnh hưởng đáng kể đến giá bán ra của sản phẩm và chúng tôi đã có được phương trình mô hình để dự đoán giá của các GPU chỉ bằng cách đo 6 chỉ số (Core_Speed, Memory, Memory_Speed, Memory_Bandwidth, Memory_Bus, Texture_Rate). Mô hình này sẽ giúp cho những khách hàng có nhu cầu mua bán cũng như xem xét sự phù hợp của chi phí bán ra cho bộ sản phẩm này. Vì thế, mô hình đã phần nào giúp cho những khách hàng tiềm năng đến với sản phẩm này và đồng thời hỗ trợ nhà sản xuất trong việc định giá sản phẩm.

Nhìn chung, giá bán ra của GPU sau khi sử dụng mô hình có giá trị tương đối phù hợp so với giá gốc ban đầu.

Thêm vào đó, có thể dễ dàng nhận ra nếu giá trị của các biến “Memory”, “Memory_Bandwidth” và “Texture_Rate” càng lớn thì giá trị của “Release_Price” sẽ càng cao. Điều này có ý nghĩa rằng giá bán ra của các GPU sẽ tỉ lệ thuận với các biến được đề cập và các biến còn lại sẽ là tỉ lệ nghịch với giá bán của GPU.

VI. THẢO LUẬN VÀ MỞ RỘNG

1. ƯU ĐIỂM

- Hồi quy tuyến tính bội cho phép chúng ta phân tích đồng thời tác động của nhiều biến độc lập lên một biến phụ thuộc. Điều này giúp chúng ta xác định được tầm quan trọng tương đối của từng biến độc lập trong việc dự đoán biến phụ thuộc.
- Bằng cách sử dụng 6 yếu tố chính tác động đến biến phụ thuộc “Release_Price”, chúng ta đã có thể dự đoán giá bán ra của các GPU.

2. HẠN CHẾ

- Chỉ có thể sử dụng cho kiểu dữ liệu tuyến tính.
- Cần phải có đầy đủ 6 biến theo yêu cầu để mô hình hoạt động tối ưu nhất
- Để mô hình hoạt động hiệu quả, cần phải loại bỏ các điểm ngoại lai là những giá trị quan trắc có sự khác biệt đáng kể so với các giá trị quan trắc khác trong bộ dữ liệu vì chúng có thể làm sai lệch mối quan hệ giữa biến dự đoán và biến phụ thuộc và dẫn đến dự đoán không chính xác.

VII. NGUỒN DỮ LIỆU VÀ NGUỒN CODE

1. NGUỒN DỮ LIỆU:

<https://www.kaggle.com/datasets/iliassekkaf/computerparts>

2. NGUỒN CODE:

<https://drive.google.com/drive/u/1/folders/18kb1rgfnzWzJXafUWvHJPek3jW2VxOaz>

VIII. TÀI LIỆU THAM KHẢO

1. Nguyễn Tiến Dũng (chủ biên), Nguyễn Đình Huy, *Xác suất – Thống kê & Phân tích số liệu*, 2019.
2. Phan Thị Hương, *Bài giảng điện tử môn học Xác suất và Thống kê*, truy cập từ e-learning.hcmut.edu.vn.
3. Douglas C.Montgomery & George C.Runger, *APPLIED STATISTICS AND PROBABILITY FOR ENGINEERS 6TH EDITION*.
4. Ilissek, *COMPUTERS PARTS(CPU's and GPU's)*, truy cập từ <https://www.kaggle.com/datasets/iliassekkaf/computerparts>.
5. [Các loại card màn hình \(VGA\) và thông số quan trọng thường gặp – GEARVN.COM](http://GEARVN.COM).