

ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC BÁCH KHOA
KHOA KHOA HỌC VÀ KỸ THUẬT MÁY TÍNH



Xác suất thống kê MT2013

Báo cáo bài tập lớn

Phân tích dữ liệu GPU

Giảng viên hướng dẫn: Nguyễn Kiều Dung

STT	Tên	MSSV	Nhóm Lớp
1	Lê Ngô Khôi Nguyên	2412338	L07
2	Phan Nguyễn Hoàng Minh	2412114	L07
3	Trần Đình Ý	2414100	L07
4	Phạm Nghĩa Trung	2413713	L08
6	Nguyễn Gia Hưng	2411356	L09
7	Nguyễn Quốc Trung	2413707	L14
8	Phạm Minh Đức	2414106	L15

THÀNH PHỐ HỒ CHÍ MINH, THÁNG 12 NĂM 2025



Danh sách thành viên

STT	Họ và tên	MSSV	Lớp	Vai trò	Điểm
1	Lê Ngô Khôi Nguyên	2412338	L07	Thống kê mô tả	
2	Phan Nguyễn Hoàng Minh	2412114	L07	Anova một yếu tố	
3	Trần Đình Ý	2414100	L07	Tổng hợp bài làm	
4	Phạm Nghĩa Trung	2413713	L08	Tổng quan dữ liệu và kiến thức nền	
6	Nguyễn Gia Hưng	2411356	L09	Hồi quy tuyến tính bội	
7	Nguyễn Quốc Trung	2413707	L14	Kiểm định 2 mẫu, Thảo luận và mở rộng	
8	Phạm Minh Đức	2414106	L15	Kiểm định 1 mẫu	

Mục lục

1	Tổng quan dữ liệu	1
2	Kiến thức nền	3
2.1	Thống kê mô tả và thống kê suy diễn	3
2.2	Các đặc trưng của tổng thể và mẫu	3
2.2.1	Khái niệm	3
2.2.2	Tỷ lệ	3
2.2.3	Trung bình	3
2.2.4	Phương sai, độ lệch chuẩn	4
2.2.5	Các đặc trưng khác	5
2.3	Ước lượng	5
2.4	Kiểm định giả thuyết thống kê	6
2.4.1	Khái niệm chung về kiểm định	6
2.4.2	Quy tắc kiểm định:	7
2.4.3	Các sai lầm trong bài toán kiểm định	7
2.4.4	Các bước thực hiện kiểm định	8
2.5	Các mô hình kiểm định được sử dụng trong báo cáo	8
2.5.1	Bài toán kiểm định trung bình 1 mẫu	8
2.5.2	Bài toán kiểm định 2 mẫu	10
2.5.3	Phân tích phương sai (anova)	11
2.5.4	Phân tích phương sai 1 yếu tố	11
2.5.5	Hồi quy tuyến tính bội	12
3	Tiền xử lý dữ liệu	14
3.1	Đọc dữ liệu vào R	14
3.2	Làm sạch dữ liệu	15
4	Thống kê mô tả	18
4.1	Tính các giá trị đặc trưng của các biến.	18
4.1.1	Đối với các biến có giá trị số.	18
4.1.2	Đối với các biến phân loại	20
4.1.3	Kiểm tra lại các đặc điểm của các biến.	22
4.2	Sự phân phối tần số của biến Memory_Bandwidth.	23
4.3	Đồ thị scatter plot cho biến Memory_Bandwidth theo Memory_Bus, L2_Cache, Memory_Speed, Process.	24

5	Thống kê suy diễn	26
5.1	Bài toán 1 mẫu.	26
5.1.1	Mục đích kiểm định.	26
5.1.2	Giả thiết nghiên cứu.	26
5.1.3	Thực hiện kiểm định.	26
5.2	Bài toán 2 mẫu.	28
5.2.1	Mục đích kiểm định.	28
5.2.2	Giả thiết nghiên cứu.	28
5.2.3	Thực hiện kiểm định.	28
5.3	Phân tích phương sai (ANOVA) một yếu tố về ảnh hưởng của nhà sản xuất đến băng thông bộ nhớ	31
5.3.1	Mục đích kiểm định.	31
5.3.2	Giả thiết nghiên cứu.	31
5.3.3	Kiểm tra giả định thống kê của ANOVA một yếu tố.	31
5.3.4	Xây dựng mô hình ANOVA một yếu tố trong R.	31
5.3.5	Nhận xét về mô hình ANOVA một yếu tố.	35
5.4	Áp dụng mô hình hồi quy tuyến tính bội vào phân tích hiệu suất GPU.	36
5.4.1	Xây dựng mô hình.	36
5.4.2	Kiểm tra giả định của mô hình hồi quy.	38
5.4.3	Nhận xét về mô hình hồi quy tuyến tính.	41
6	Thảo luận và mở rộng	42
6.1	Thảo luận	42
6.2	Mở rộng	42
7	Nguồn dữ liệu và nguồn code	46

Danh sách hình

3.1	Hiển thị 5 dòng đầu tiên của dữ liệu sau khi đọc vào R	14
3.2	Tỷ lệ dữ liệu khuyết thiêu trong các đặc trưng	16
3.3	Các cột còn lại sau khi loại bỏ các cột có tỷ lệ NA > 15%	17
3.4	Dữ liệu sau khi làm sạch	17
4.1	Biểu đồ hộp cho các biến số	19
4.2	Biểu đồ hộp cho các biến số sau khi loại bỏ ngoại lai và quan sát không phù hợp	22
4.3	Biểu đồ tần số của biến Memory_Bandwidth	23
4.4	Đồ thị scatter plot cho biến Memory_Bandwidth theo Memory_Bus, L2_Cache, Memory_Speed, Process	25
5.1	Bảng kết quả các đặc điểm thống kê mẫu	26
5.2	Đồ thị Q-Q plot cho mẫu	27
5.3	Kết quả tính giá trị Z_qs trong R	27
5.4	Bảng kết quả các đặc điểm thống kê mẫu cho từng nhóm	29
5.5	Đồ thị Q-Q plot cho hai mẫu	29
5.6	Kết quả tính giá trị Z_qs trong R	30
5.7	Kết quả kiểm định Shapiro-Wilk cho phần dư của mô hình ANOVA	32
5.8	Biểu đồ Q-Q plot cho phần dư của mô hình ANOVA	33
5.9	Kết quả kiểm định Levene cho phương sai đồng nhất	33
5.10	Bảng ANOVA một yếu tố trong R	34
5.11	Kết quả so sánh hậu nghiệm Tukey HSD trong R	34
5.12	Kết quả của mô hình hồi quy tuyến tính	37
5.13	Đồ thị giữa giá trị thực tế và dự đoán của Memory_Bandwidth (mô hình gốc) .	37
5.14	Biểu đồ Residuals vs Fitted	38
5.15	Biểu đồ Q-Q	39
5.16	Biểu đồ Scale-Location	40
6.1	Kết quả mô hình hồi quy tuyến tính bội sau khi biến đổi logarit	44
6.2	Kết quả dự đoán giá trị Memory_Bandwidth sử dụng mô hình mới	45
6.3	Đồ thị giữa giá trị thực tế và dự đoán của Memory_Bandwidth (mô hình cải tiến)	45

Danh sách bảng

1.1	Bảng mô tả một vài thông số quan trọng của tập dữ liệu GPU	2
2.1	Công thức của bài toán kiểm định tỷ lệ & trung bình 1 mẫu	8
2.2	Các dạng toán kiểm định 2 mẫu	10
2.3	Bảng tóm tắt ANOVA 1 yếu tố	12

Danh sách đoạn mã

1	Đọc dữ liệu từ file trong R	14
2	Thay thế giá trị rác thành NA	15
3	Trực quan hóa tỷ lệ dữ liệu khuyết thiểu	15
4	Loại bỏ các cột có tỷ lệ NA > 15%	16
5	Chuẩn hoá các đơn vị đo trong dữ liệu	17
6	Tính các giá trị đặc trưng của các biến	18
7	Vẽ biểu đồ hộp cho các biến số	19
8	Loại bỏ các ngoại lai khỏi các biến số	20
9	Đếm số lượng các nhóm con trong các biến phân loại	20
10	Loại bỏ các quan sát không phù hợp khỏi tập dữ liệu	21
11	Vẽ đồ thị scatter plot.	24
12	Tính các đặc điểm thống kê mẫu trong R	26
13	Tính giá trị Z_qs trong R	27
14	Tính các đặc điểm thống kê mẫu cho từng nhóm trong R	28
15	Tính giá trị Z_qs trong R	30
16	Xây dựng mô hình ANOVA một yếu tố trong R	31
17	Kiểm định Shapiro-Wilk cho phần dư của mô hình ANOVA.	32
18	Biểu đồ Q-Q plot cho phần dư của mô hình ANOVA	32
19	Kiểm định Levene cho phương sai đồng nhất	33
20	Bảng ANOVA một yếu tố trong R	34
21	So sánh hậu nghiệm Tukey HSD trong R	34
22	Xây dựng mô hình hồi quy tuyến tính bội trong R	36
23	Vẽ đồ thị Residuals-Fitted trong R	38
24	Biểu đồ Q-Q plot cho phần dư của mô hình hồi quy tuyến tính	39
25	Vẽ đồ thị Scale-Location trong R	39
26	Kiểm định Durbin-Watson trong R	40
27	Phép biến đổi logarit cho các biến.	43
28	Dự đoán giá trị Memory_Bandwidth sử dụng mô hình mới.	44



1 Tổng quan dữ liệu

Trong thời đại công nghệ phát triển nhanh chóng, bộ xử lý đồ họa (GPU – Graphics Processing Unit) đã trở thành một trong những thành phần quan trọng nhất của máy tính hiện đại. Ban đầu, GPU được thiết kế với mục đích chính là xử lý hình ảnh và đồ họa trong các trò chơi điện tử, phần mềm thiết kế và dựng phim. Tuy nhiên, cùng với sự tiến bộ của công nghệ, vai trò của GPU đã vượt xa khỏi phạm vi đồ họa thuần túy. Ngày nay, GPU đóng vai trò cốt lõi trong các lĩnh vực như trí tuệ nhân tạo (AI), học sâu (Deep Learning), mô phỏng khoa học, và xử lý dữ liệu lớn. Nhờ vào cấu trúc song song mạnh mẽ với hàng nghìn lõi xử lý, GPU có thể thực hiện hàng loạt phép tính phức tạp cùng lúc, giúp rút ngắn đáng kể thời gian xử lý so với CPU truyền thống. Dưới đây là tập dữ liệu khảo sát các yếu tố về GPU cung cấp những thông tin chi tiết như tốc độ xung nhịp, nhiệt độ tối đa, mức tiêu thụ điện năng, kích thước khuôn chip, ngày phát hành, giá bán, và nhiều đặc trưng kỹ thuật khác. Việc phân tích các dữ liệu này giúp ta hiểu rõ hơn về sự phát triển của công nghệ GPU qua các giai đoạn, từ đó đánh giá xu hướng tiến hóa về hiệu năng, giá thành, và mức độ tối ưu năng lượng. Bên cạnh đó, người nghiên cứu có thể khám phá mối quan hệ giữa giá thành và hiệu suất hoạt động, tìm hiểu xem liệu có nhà sản xuất nào nổi bật trong một phân khúc nhất định hay không. Thông qua việc khai thác và phân tích dữ liệu GPU, chúng ta có thể dự đoán xu hướng của các thế hệ GPU tương lai, phục vụ cho các ứng dụng như học máy, đồ họa máy tính, và tính toán hiệu năng cao.

Tập dữ liệu All_GPUs, bao gồm 34 thông số của 3406 bộ xử lý đồ họa (GPU) khác nhau đến từ 3 nhà sản xuất chính là NVIDIA, AMD và Intel. Dữ liệu được quan sát và thu thập từ trang web [Kaggle](#). Tập dữ liệu này có tương đối nhiều thông số, trong đó có thể kể đến một số thông số được nêu ra trong bảng 1.1 dưới đây:



STT	Tên biến	Đơn vị	Mô tả
1	Manufacturer		Hãng của sản xuất GPU
2	Release_Date		Năm sản xuất của GPU
3	Memory_Bandwidth	Gigabyte/giây (GB/s)	Lượng dữ liệu tối đa mà bộ nhớ GPU có thể truyền tải trong mỗi giây
4	Memory_Speed	MHz	Độ rộng của bus bộ nhớ, ảnh hưởng đến tốc độ truy cập và hiệu suất của bộ nhớ GPU
5	L2_Cache	KB	Bộ nhớ đệm cấp 2 giúp GPU truy cập nhanh hơn vào dữ liệu được sử dụng thường xuyên, tối ưu hóa hiệu suất
6	Memory_Bus	Bit	Độ rộng của kênh truyền dữ liệu trong RAM (Memory).
7	Memory		Dung lượng bộ nhớ đồ họa (VRAM) của GPU, quyết định khả năng xử lý và lưu trữ dữ liệu hình

Bảng 1.1: Bảng mô tả một vài thông số quan trọng của tập dữ liệu GPU

Như đã đề cập ở trên, tập dữ liệu có tổng cộng 34 thông số khác nhau, tuy nhiên nếu liệt kê hết ở bảng 1.1 thì sẽ rất dài. Ngoài ra, trong các thông số đó, có nhiều thông số mang tính kỹ thuật cao và không phổ biến, những dữ liệu này sẽ được xử lý sau để dễ dàng hơn trong việc phân tích và trực quan hóa dữ liệu.



2 Kiến thức nền

2.1 Thống kê mô tả và thống kê suy diễn

Thống kê mô tả (descriptive statistics): là quá trình thu thập, biểu diễn, tổng hợp và xử lý dữ liệu để biến đổi dữ liệu thành thông tin.

Thống kê suy diễn (Inferential statistics): xử lý các thông tin có được từ thống kê mô tả, từ đó đưa ra các cơ sở cho những dự đoán (predictions), dự báo (forecasts) và các ước lượng (estimations).

2.2 Các đặc trưng của tổng thể và mẫu

2.2.1 Khái niệm

Tổng thể thống kê (population): là tập hợp các phần tử thuộc đối tượng nghiên cứu, cần được quan sát, thu thập và phân tích theo một hoặc một số đặc trưng nào đó. Các phần tử tạo thành tổng thể thống kê được gọi là đơn vị tổng thể.

Mẫu (sample): là một số đơn vị được chọn ra từ tổng thể theo một phương pháp lấy mẫu nào đó. Các đặc trưng mẫu được sử dụng để suy rộng ra các đặc trưng của tổng thể nói chung.

Đặc điểm thống kê (dấu hiệu nghiên cứu): là các tính chất quan trọng liên quan trực tiếp đến nội dung nghiên cứu và khảo sát cần thu thập dữ liệu trên các đơn vị tổng thể. Người ta chia làm 2 loại: đặc điểm thuộc tính và đặc điểm số lượng.

2.2.2 Tỷ lệ

Với một tổng thể có N phần tử và M phần tử mang tính chất A nào đó. Tỷ lệ tổng thể (kí hiệu: p) được tính bởi công thức:

$$p = \frac{M}{N}$$

Với một mẫu có n phần tử và có m phần tử mang tính chất A nào đó. Tỷ lệ mẫu (kí hiệu: f hay \bar{p}) được tính bởi công thức:

$$p = f = \frac{m}{n}$$

2.2.3 Trung bình

Trung bình (mean): là đại lượng thường được sử dụng nhất để đo giá trị trung tâm của dữ liệu (Trung bình bị ảnh hưởng bởi các giá trị ngoại lai). Với một tổng thể có N phần tử,



trung bình tổng thể (kí hiệu: μ hay \bar{X}) tính bởi công thức:

$$\mu = \frac{1}{N} \sum_{i=1}^N X_i = \frac{X_1 + X_2 + \cdots + X_N}{N}$$

Với một mẫu có n phần tử, trung bình mẫu (kí hiệu: \bar{x}) tính bởi công thức:

$$\bar{x} = \frac{1}{n} \sum_{i=1}^n x_i = \frac{x_1 + x_2 + \cdots + x_n}{n}$$

Trong trường hợp X có bảng phân phối tần số như sau:

X	x_1	x_2	x_3	\dots	x_k
Tần số	n_1	n_2	n_3	\dots	n_k

Ta lại có trung bình mẫu tính bởi công thức:

$$\bar{x} = \frac{1}{n} \sum_{i=1}^k x_i n_i = \frac{x_1 n_1 + x_2 n_2 + \cdots + x_k n_k}{n}$$

2.2.4 Phương sai, độ lệch chuẩn

Phương sai (Variance): là trung bình của bình phương độ lệch các giá trị so với trung bình. Phương sai phản ánh độ phân tán hay sự biến thiên của dữ liệu.

Độ lệch chuẩn (Standard deviation): Độ lệch chuẩn (Standard deviation) là căn bậc hai của phương sai. Độ lệch chuẩn dùng để đo sự biến thiên, biểu diễn sự biến thiên xung quanh trung bình và cũng có cùng đơn vị đo với dữ liệu gốc.

Với một tổng thể có N phần tử, phương sai tổng thể (kí hiệu: σ^2) tính bởi công thức:

$$\sigma^2 = \frac{1}{N} \sum_{i=1}^N (x_i - \mu)^2$$

Khi đó: σ được gọi là độ lệch chuẩn của tổng thể.

Với một mẫu có n phần tử, phương sai mẫu (kí hiệu: s^2) tính bởi công thức:

$$s^2 = \frac{1}{n-1} \sum_{i=1}^n (x_i - \bar{x})^2$$

Trong trường hợp X có bảng phân phối tần số như sau:

X	x_1	x_2	x_3	\dots	x_k
Tần số	n_1	n_2	n_3	\dots	n_k

Ta lại có phương sai mẫu tính bởi công thức:

$$s^2 = \frac{1}{n-1} \sum_{i=1}^k n_i (x_i - \bar{x})^2$$

Khi đó: s được gọi là độ lệch chuẩn mẫu.



2.2.5 Các đặc trưng khác

Yếu vị (Mode): là giá trị của phần tử có số lần xuất hiện lớn nhất trong mẫu. Yếu vị không bị ảnh hưởng bởi các điểm ngoại lai.

Hệ số biến thiên (Coefficient of variation): đo lường mức độ biến động tương đối của mẫu dữ liệu, được dùng khi người ta muốn so sánh mức độ biến động của các mẫu không cùng đơn vị đo. Đơn vị tính bằng %.

$$CV(\text{tongthe}) = \frac{\sigma}{\mu} \times 100\%$$

$$CV(\text{mau}) = \frac{s}{\bar{x}} \times 100\%$$

Sai số chuẩn (Standard Error): là giá trị đại diện cho độ lệch chuẩn của giá trị trung bình trong tập dữ liệu. Nó phục vụ như một thước đo biến động cho các biến ngẫu nhiên hay độ lượn độ phân tán. Độ phân tán càng nhỏ, dữ liệu càng chính xác.

Trung vị (Median): Giả sử X có N quan sát, sắp các quan sát này theo thứ tự tăng dần. Trung vị là giá trị nằm chính giữa dãy số này và chia nó thành 2 phần bằng nhau. Cụ thể:

Giả sử mẫu có kích thước n được sắp xếp tăng dần theo giá trị được khảo sát:

$$x_1 \leq x_2 \leq \dots \leq x_{n-1} \leq x_n$$

Nếu $n = 2k + 1$ (n lẻ) thì trung vị mẫu là giá trị x_{k+1}

Nếu $n = 2k$ (n chẵn) thì trung vị mẫu là giá trị $\frac{x_k + x_{k+1}}{2}$

Trung vị không bị ảnh hưởng bởi các điểm ngoại lai (outliers).

Tứ phân vị (Quartiles): Giá trị trung vị chia mẫu dữ liệu đã sắp thứ tự thành 2 tập có số phần tử bằng nhau. Trung vị của tập dữ liệu nhỏ hơn là Q_1 (gọi là tứ phân vị dưới) và trung vị của tập dữ liệu lớn hơn là Q_3 (gọi là tứ phân vị trên). Q_2 được lấy bằng giá trị trung vị. Độ trải giữa, hay là khoảng tứ phân vị $IQR = Q_3 - Q_1$.

Điểm Outlier: còn gọi là điểm dị biệt, điểm ngoại lệ, điểm ngoại lai.... Đó là các phần tử của mẫu có giá trị nằm ngoài khoảng

$$(Q_1 - 1.5 \times IQR; Q_3 + 1.5 \times IQR)$$

2.3 Ước lượng

Lý thuyết ước lượng là một nội dung trọng tâm trong thống kê và nghiên cứu khoa học, tập trung vào việc xác định giá trị của các tham số (parameters) của quần thể dựa trên những mẫu (samples) được chọn ra từ quần thể đó. Mục tiêu chính của ước lượng là tìm ra các giá trị gần đúng cho những đại lượng đặc trưng của quần thể như trung bình tổng thể (μ), phương sai tổng thể (σ^2), và tỷ lệ phần tử có đặc điểm nhất định trong quần thể (p).



- **Khoảng Tin Cậy (Confidence Interval - CI):** Là một loại ước lượng khoảng được sử dụng để chỉ ra phạm vi mà ta tin rằng tham số của tổng thể nằm trong đó. Khoảng tin cậy thường được xác định bởi hai giới hạn: giới hạn dưới và giới hạn trên. Ví dụ, một khoảng tin cậy 95% cho trung bình tổng thể có thể là (20, 30), nghĩa là ta tin rằng với độ tin cậy 95%, trung bình thực sự của tổng thể nằm trong khoảng từ 20 đến 30.
- **Mức Ý Nghĩa (Significance Level - α):** Là ngưỡng mà ta chọn để quyết định ý nghĩa. Ví dụ mức ý nghĩa thường được chọn ở mức 0.05 - nghĩa là khả năng kết quả quan sát sự khác biệt được nhìn thấy trên số liệu là ngẫu nhiên chỉ là 5%.
- **Độ Tin Cậy (Confidence Level):** Được biểu thị dưới dạng một tỷ lệ phần trăm chỉ mức độ tin tưởng hoặc sự chắc chắn mà khoảng tin cậy ước lượng của chúng ta bao gồm tham số tổng thể thực sự. Ví dụ: nếu ta xây dựng khoảng tin cậy với mức tin cậy 95%, ta tin chắc rằng 95 trên 100 lần ước tính sẽ nằm giữa giá trị trên và giá trị dưới được chỉ định bởi khoảng tin cậy.

$$\gamma = 1 - \alpha$$

Có hai phương pháp ước lượng thường được sử dụng là ước lượng điểm (point estimation) và ước lượng khoảng (interval estimation), tuy nhiên trong phạm vi bài này, nhóm chỉ nhắm đến ước lượng bằng khoảng tin cậy.

2.4 Kiểm định giả thuyết thống kê

2.4.1 Khái niệm chung về kiểm định

Trong thống kê, **kiểm định (hypothesis testing)** là quá trình đánh giá một giả thuyết về dữ liệu để xác định xem liệu có đủ bằng chứng để chấp nhận hay bác bỏ giả thuyết đó. Mục tiêu của kiểm định là đưa ra quyết định dựa trên dữ liệu mẫu có sẵn để rút ra những kết luận về tổng thể.

Quá trình kiểm định thường bắt đầu bằng việc xây dựng hai giả thuyết:

- **Giả thuyết không (null hypothesis, ký hiệu H_0):** là giả thiết về yếu tố cần kiểm định của tổng thể ở trạng thái bình thường, không chịu tác động của các hiện tượng liên quan. Yếu tố trong H_0 phải được xác định cụ thể.
- **Giả thuyết thay thế - giả thuyết đối (alternative hypothesis, ký hiệu H_1):** là một mệnh đề mâu thuẫn với H_0 , H_1 thể hiện xu hướng cần kiểm định.



Miền Bác Bỏ (Rejection Region): là miền số thực thỏa $P(G \in RR | H_0 \text{ đúng}) = \alpha$. α là một số khá bé, thường không quá 10% và được gọi là mức ý nghĩa của kiểm định. Một ký hiệu khác của miền bác bỏ được dùng trong bài: W_α

Miền Chấp Nhận (Acceptance Region): phần bù của miền bác bỏ trong R

Tiêu chuẩn kiểm định: là hàm thống kê $G = G(X_1, X_2, \dots, X_n, \theta_0)$, xây dựng trên mẫu ngẫu nhiên $W = (X_1, X_2, \dots, X_n)$ và tham số θ_0 liên quan đến H_0 ; Điều kiện đặt ra với thống kê G là nếu H_0 đúng thì quy luật phân phối xác suất của G phải hoàn toàn xác định.

2.4.2 Quy tắc kiểm định:

Từ mẫu thực nghiệm, ta tính được một giá trị cụ thể của tiêu chuẩn kiểm định, gọi là **giá trị kiểm định thống kê**:

$$g_{qs} = G(x_1, x_2, \dots, x_n, \theta_0)$$

Theo nguyên lý xác suất bé, biến cő $G \in RR$ có xác suất nhỏ nên với 1 mẫu thực nghiệm ngẫu nhiên, nó không thể xảy ra.

Do đó:

- + Nếu $g_{qs} \in RR$ thì bác bỏ H_0 , thừa nhận giả thiết H_1 .
- + Nếu $g_{qs} \notin RR$: ta chưa đủ dữ liệu khẳng định H_0 sai. Vì vậy ta chưa thể chứng minh được H_1 đúng.

2.4.3 Các sai lầm trong bài toán kiểm định

Kết luận của một bài toán kiểm định có thể mắc các sai lầm sau:

- **Sai lầm loại I:** Bác bỏ giả thiết H_0 trong khi H_0 đúng. Xác suất mắc phải sai lầm này nếu H_0 đúng chính bằng mức ý nghĩa α . Nguyên nhân mắc phải sai lầm loại I thường có thể do kích thước mẫu quá nhỏ, có thể do phương pháp lấy mẫu ...
- **Sai lầm loại II:** Thừa nhận H_0 trong khi H_0 sai, tức là mặc dù thực tế H_1 đúng nhưng giá trị thực nghiệm g_{qs} không thuộc RR.

Quyết định	Tình huống	
	H_0 đúng	H_0 sai
Bác bỏ H_0	Sai lầm loại I. Xác suất = α	Quyết định đúng
Không bác bỏ H_0	Quyết định đúng	Sai lầm loại II. Xác suất = β



2.4.4 Các bước thực hiện kiểm định

1. Phát biểu giả thuyết và đối thuyết của bài toán.
2. Tính giá trị thống kê kiểm định (tiêu chuẩn kiểm định) cho bài toán.
3. Xác định miền bác bỏ tốt nhất cho bài toán.
4. Đưa ra kết luận.

2.5 Các mô hình kiểm định được sử dụng trong báo cáo

2.5.1 Bài toán kiểm định trung bình 1 mẫu

Dạng bài	Phân bố của tổng thể	Giả thiết H_0	Giả thiết đối H_1	Miền bác bỏ RR (miền bác bỏ H_0 với mức ý nghĩa α)	Hàm thống kê kiểm định (Tiêu chuẩn kiểm định)
Kiểm định tỷ lệ 1 mẫu	* X có phân phối Không - mít. * $n \geq 30$. (1)	$p = p_0$	$p \neq p_0$	$(-\infty; -z_{\alpha/2}) \cup (z_{\alpha/2}; +\infty)$	$Z_{qs} = \frac{f - p_0}{\sqrt{\frac{p_0(1-p_0)}{n}}} \approx N(0, 1)$
			$p < p_0$	$(-\infty; -z_\alpha)$	
			$p > p_0$	$(z_\alpha; +\infty)$	
Kiểm định trung bình 1 mẫu	* X có phân phối chuẩn. * Đã biết σ^2 . (2a)	$\mu = \mu_0$	$\mu \neq \mu_0$	$(-\infty; -z_{\alpha/2}) \cup (z_{\alpha/2}; +\infty)$	$Z_{qs} = \frac{\bar{X} - \mu_0}{\frac{\sigma}{\sqrt{n}}} \sim N(0, 1)$
			$\mu < \mu_0$	$(-\infty; -z_\alpha)$	
			$\mu > \mu_0$	$(z_\alpha; +\infty)$	
Kiểm định trung bình 1 mẫu	* X có phân phối chuẩn. * Chưa biết σ^2 . (2b)	$\mu = \mu_0$	$\mu \neq \mu_0$	$(-\infty; -t_{\alpha/2;(n-1)}) \cup (t_{\alpha/2;(n-1)}; +\infty)$	$T_{qs} = \frac{\bar{X} - \mu_0}{\frac{s}{\sqrt{n}}} \sim T_{(n-1)}$
			$\mu < \mu_0$	$(-\infty; -t_{\alpha;(n-1)})$	
			$\mu > \mu_0$	$(t_{\alpha;(n-1)}; +\infty)$	
Kiểm định trung bình 1 mẫu	* X có phân phối tùy ý. * $n \geq 30$. (2c) X không có giả thiết PPC	$\mu = \mu_0$	$\mu \neq \mu_0$	$(-\infty; -z_{\alpha/2}) \cup (z_{\alpha/2}; +\infty)$	$Z_{qs} = \frac{\bar{X} - \mu_0}{\frac{\sigma}{\sqrt{n}}} \approx N(0, 1)$ Nếu chưa biết σ thì thay bởi s
			$\mu < \mu_0$	$(-\infty; -z_\alpha)$	
			$\mu > \mu_0$	$(z_\alpha; +\infty)$	

Bảng 2.1: Công thức của bài toán kiểm định tỷ lệ & trung bình 1 mẫu





2.5.2 Bài toán kiểm định 2 mẫu

Phân bố của tổng thể	GT H_0	GT H_1	Miền bắc bỏ RR	T/chuẩn kiểm định
* 2 mẫu độc lập * X_1, X_2 có pp chuẩn * Đã biết σ_1^2 và σ_2^2 (4a)	$\mu_1 = \mu_2$	$\mu_1 \neq \mu_2$	$(-\infty; -z_{\alpha/2}) \cup (z_{\alpha/2}; +\infty)$	$Z_{qs} = \frac{\bar{X}_1 - \bar{X}_2}{\sqrt{\frac{\sigma_1^2}{n_1} + \frac{\sigma_2^2}{n_2}}}$
		$\mu_1 < \mu_2$	$(-\infty; -z_\alpha)$	
		$\mu_1 > \mu_2$	$(z_\alpha; +\infty)$	
* 2 mẫu độc lập * X_1, X_2 có pp chuẩn * Chưa biết $\sigma_1^2; \sigma_2^2; \sigma_1^2 = \sigma_2^2$ (4b)	$\mu_1 = \mu_2$	$\mu_1 \neq \mu_2$	$(-\infty; -t_{\alpha/2;(n_1+n_2-2)}) \cup (t_{\alpha/2;(n_1+n_2-2)}; +\infty)$	$T_{qs} = \frac{\bar{X}_1 - \bar{X}_2}{\sqrt{s_p^2 \left(\frac{1}{n_1} + \frac{1}{n_2} \right)}}$
		$\mu_1 < \mu_2$	$(-\infty; -t_{\alpha;(n_1+n_2-2)})$	
		$\mu_1 > \mu_2$	$(t_{\alpha;(n_1+n_2-2)}; +\infty)$	
* 2 mẫu độc lập * X_1, X_2 có pp chuẩn * Chưa biết $\sigma_1^2, \sigma_2^2; \sigma_1^2 \neq \sigma_2^2$ (4c)	$\mu_1 = \mu_2$	$\mu_1 \neq \mu_2$	$(-\infty; -t_{\alpha/2;(\nu)}) \cup (t_{\alpha/2;(\nu)}; +\infty)$	$T_{qs} = \frac{\bar{X}_1 - \bar{X}_2}{\sqrt{\frac{s_1^2}{n_1} + \frac{s_2^2}{n_2}}}$
		$\mu_1 < \mu_2$	$(-\infty; -t_{\alpha;(\nu)})$	
		$\mu_1 > \mu_2$	$(t_{\alpha;(\nu)}; +\infty)$	
* 2 mẫu độc lập * X_1, X_2 có pp tùy ý * Mẫu lớn: $n_1, n_2 \geq 30$ * Đã biết hoặc chưa biết σ_1^2, σ_2^2 (4d)	$\mu_1 = \mu_2$	$\mu_1 \neq \mu_2$	$(-\infty; -z_{\alpha/2}) \cup (z_{\alpha/2}; +\infty)$	$Z_{qs} = \frac{\bar{X}_1 - \bar{X}_2}{\sqrt{\frac{\sigma_1^2}{n_1} + \frac{\sigma_2^2}{n_2}}}$
		$\mu_1 < \mu_2$	$(-\infty; -z_\alpha)$	
		$\mu_1 > \mu_2$	$(z_\alpha; +\infty)$	
* 2 mẫu phụ thuộc tương ứng theo cặp * X_1, X_2 có pp chuẩn * Chưa biết σ_D^2 (4e)	$\mu_1 = \mu_2$	$\mu_1 \neq \mu_2$	$(-\infty; -t_{\alpha/2;(n-1)}) \cup (t_{\alpha/2;(n-1)}; +\infty)$	$T_{qs} = \frac{\bar{X}_D - \mu_0}{s_D} \sqrt{n}$
		$\mu_1 < \mu_2$	$(-\infty; -t_{\alpha;(n-1)})$	
		$\mu_1 > \mu_2$	$(t_{\alpha;(n-1)}; +\infty)$	
* 2 mẫu phụ thuộc tương ứng theo cặp * 2 mẫu lớn: $n \geq 30$ * Đã biết hoặc chưa biết σ_D^2 (4f)	$\mu_1 = \mu_2$	$\mu_1 \neq \mu_2$	$(-\infty; -z_{\alpha/2}) \cup (z_{\alpha/2}; +\infty)$	$Z_{qs} = \frac{\bar{X}_D - \mu_0}{\sigma_D} \sqrt{n}$
		$\mu_1 < \mu_2$	$(-\infty; -z_\alpha)$	
		$\mu_1 > \mu_2$	$(z_\alpha; +\infty)$	

Bảng 2.2: Các dạng toán kiểm định 2 mẫu



2.5.3 Phân tích phương sai (anova)

Phân tích phương sai là một mô hình dùng để xem xét sự biến động của một biến ngẫu nhiên định lượng X chịu tác động trực tiếp của một hay nhiều yếu tố nguyên nhân (định tính). Được làm hai loại là phân tích phương sai 1 yếu tố và phân tích phương sai 2 yếu tố.

2.5.4 Phân tích phương sai 1 yếu tố

Giả thiết

- Các tổng thể có phân phối chuẩn $N(\mu_i; \sigma_i^2)$, $i = 1, 2, \dots, k$ với k là tổng thể (thông thường $k \geq 3$).
- Phương sai các tổng thể chưa biết nhưng được giả định là bằng nhau ($\sigma_1^2 = \sigma_2^2 = \dots = \sigma_k^2$).
- Các mẫu quan sát (từ k tổng thể) được lấy độc lập.

Các bước thực hiện

Bước 1: Đặt giả thiết kiểm định

$$H_0 : \mu_1 = \mu_2 = \dots = \mu_k,$$

$$H_1 : \exists \mu_i \neq \mu_j \quad (i \neq j)$$

Bước 2: Tính trung bình mẫu của các nhóm $\bar{x}_1, \bar{x}_2, \dots, \bar{x}_k$ theo công thức:

$$\bar{x}_i = \frac{\sum_{j=1}^{n_i} x_{ij}}{n_i}, \quad (i = 1, 2, \dots, k)$$

Bước 3: Tính tổng các bình phương lệch (tổng bình phương):

$$\begin{aligned} SSW &= \sum_{i=1}^k \sum_{j=1}^{n_i} (x_{ij} - \bar{x}_i)^2 \\ SSB &= \sum_{i=1}^k n_i (\bar{x}_i - \bar{x})^2 \\ SST &= SSW + SSB = \sum_{i=1}^k \sum_{j=1}^{n_i} (x_{ij} - \bar{x})^2 \end{aligned}$$

Bước 4: Tính các phương sai:

$$MSW = \frac{SSW}{N - k}, \quad MSB = \frac{SSB}{k - 1}$$

Bước 5: Tính thống kê kiểm định (tiêu chuẩn kiểm định, giá trị quan sát):

$$F = \frac{MSB}{MSW}$$

Bước 6: Xác định miền bác bỏ của bài toán:

$$RR = (F_{\alpha;k-1;N-k}; +\infty)$$



Tìm giá trị $F_{\alpha;k-1;N-k}$ tra bảng Fisher mức ý nghĩa α và cột $k-1$ và dòng $N-k$.

Bước 7: Đưa ra kết luận:

- Nếu $F > F_{\alpha;k-1;N-k} \iff F \in RR \Rightarrow$ Bác bỏ H_0 , chấp nhận H_1
- Nếu $F < F_{\alpha;k-1;N-k} \iff F \notin RR \Rightarrow$ không bác bỏ H_0 (chưa bác bỏ được H_0 , chấp nhận H_0)

Bảng 2.3: Bảng tóm tắt ANOVA 1 yếu 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	N-k	MSW	
Toàn bộ	SST	N-1		

2.5.5 Hồi quy tuyến tính bội

Khái niệm: Hồi quy tuyến tính là một kỹ thuật phân tích dữ liệu dự đoán giá trị của dữ liệu không xác định bằng cách sử dụng một giá trị dữ liệu liên quan và đã biết khác.

Bài toán phân tích hồi quy là bài toán nghiên cứu mối liên hệ phụ thuộc của một biến (gọi là biến phụ thuộc) vào một hay nhiều biến khác (gọi là các biến độc lập), với ý tưởng ước lượng được giá trị trung bình (tổng thể) của biến phụ thuộc theo giá trị của các biến độc lập, dựa trên mẫu được biết trước.

Trong hồi quy tuyến tính đơn, chỉ có một biến độc lập được sử dụng để dự đoán biến phụ thuộc. Tuy nhiên, trong hồi quy tuyến tính bội, có nhiều hơn một biến độc lập được sử dụng.

Hàm hồi quy tuyến tính đơn có dạng:

$$Y = f_Y(X) = E(Y|X) = \beta_0 + \beta_1 X.$$

Hàm hồi quy tuyến tính bội có dạng:

$$Y = f_Y(X_1; X_2; \dots; X_k) = E(Y|(X_1; X_2; \dots; X_k)) = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \dots + \beta_k X_k.$$

Trong đó:

- Y là biến phụ thuộc (biến cần dự đoán).
- X_1, X_2, \dots, X_k là các biến độc lập (biến giải thích).
- β_0 (cũng được gọi là hệ số điều chỉnh) là hệ số mức độ tự do của mô hình, tức là giá trị dự đoán của biến phụ thuộc khi tất cả các biến độc lập đều bằng 0.



- $\beta_1, \beta_2, \dots, \beta_k$ là các hệ số hồi quy tương ứng với từng biến độc lập.
- ϵ là sai số ngẫu nhiên (error term) biểu thị sự khác biệt giữa giá trị thực tế và giá trị dự đoán bởi mô hình.



3 Tiên xử lý dữ liệu

3.1 Đọc dữ liệu vào R

Ta đọc dữ liệu từ file dữ liệu đã cho dưới định dạng CSV vào R bằng hàm `read.csv()` như sau:

```
1 df <- read.csv("./data_sets/All_GPUs.csv")
2 head(df, 5)
```

Listing 1: Đọc dữ liệu từ file trong R

Sau khi đọc dữ liệu xong, ta có thể sử dụng hàm `head()` để hiển thị 5 dòng đầu tiên của dữ liệu nhằm kiểm tra xem dữ liệu đã được đọc đúng chưa (Hình 3.1).

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

Hình 3.1: Hiển thị 5 dòng đầu tiên của dữ liệu sau khi đọc vào R

Như vậy, ta đã hoàn thành việc đọc dữ liệu từ file CSV vào R và có thể tiến hành các bước tiền xử lý dữ liệu tiếp theo.

3.2 Làm sạch dữ liệu

Vì trong file dữ liệu ban đầu, có thể tồn tại các giá trị bị thiếu (NA) hoặc các giá trị không hợp lệ, ta cần thực hiện các bước làm sạch dữ liệu để đảm bảo tính chính xác và độ tin cậy của phân tích sau này.

Trước tiên, ta sẽ phải thay thế tất cả các giá trị rác, không hợp lệ thành giá trị *NA* trong R. Ví dụ, nếu một ô bất kì có giá trị là chuỗi rỗng "" hoặc ký tự đặc biệt như "N/A", ta sẽ thay thế chúng bằng *NA* như sau:

```

1 df <- df %>%
2   mutate(across(where(is.character), trimws))
3 df[df == " "] <- NA
4 df[df == "N/A"] <- NA
5 df[df == "NA "] <- NA
6 df[df == "-"] <- NA
7 df[df == "Unknown Release Date"] <- NA
8 # Chỉ lấy năm sản xuất, không lấy ngày công chiếu
9 df$Release_Date <- as.Date(df$Release_Date, format = "%d-%b-%Y")
10 df$Release_Date <- format(df$Release_Date, "%Y")

```

Listing 2: Thay thế giá trị rác thành NA

Sau khi thay thế các giá trị rác, nhóm nhận thấy rằng có rất nhiều yếu tố có số lượng *NA* lớn, điều này buộc nhóm phải lựa chọn giữa loại bỏ và chuẩn hóa. Trong nội dung bài báo cáo này, nhóm sẽ loại bỏ những đặc điểm (cột) có số lượng giá trị *NA* vượt quá 15% tổng số dòng dữ liệu. Để thực hiện việc này, ta có thể sử dụng đoạn mã sau:

```

1 # Dem so luong gia tri NA trong moi cot
2 missing_counts = freq.na(df)
3 # Ve do thi ty le du lieu khuyet
4 ggplot(missing_counts, aes(x = rownames(missing_counts), y =
5   missing_counts[,2], )) +
6   geom_bar(stat = "identity", fill = "cyan") +
7   geom_text(aes(label = paste0(missing_counts[,2], "%")), vjust =
8     -0.5, size = 2) +
9   labs(title = "Missing rate", x = "Feature", y = "Rate (%)") +
10  theme_minimal() +
11  theme(axis.text.x = element_text(
12    size = 10,

```

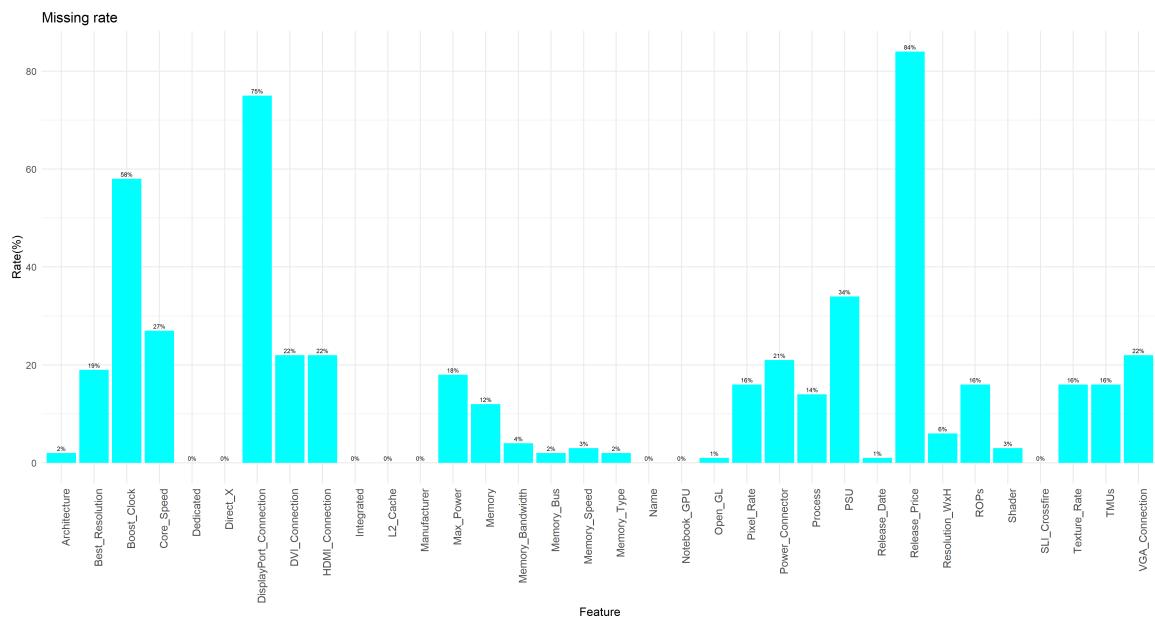
```

11    angle = 90,
12    hjust = 1
13  ))

```

Listing 3: Trực quan hóa tỷ lệ dữ liệu khuyết thiếu

Kết quả trực quan hóa tỷ lệ dữ liệu khuyết thiếu được thể hiện trong Hình 3.2.



Hình 3.2: Tỷ lệ dữ liệu khuyết thiếu trong các đặc trưng

Tiếp theo sau đó, ta sẽ loại bỏ các cột có tỷ lệ giá trị NA vượt quá 15% tổng số dòng dữ liệu như sau:

```

1 missing_counts_df <- data.frame(
2   feature = rownames(missing_counts),
3   percent = missing_counts[,2]
4 )
5 cols_to_keep <- missing_counts_df$feature[missing_counts_df$percent <= 15 & missing_counts_df$feature != "Architecture" &
6   missing_counts_df$feature != "Name"]
6 df_filtered <- df[, cols_to_keep, drop = FALSE]
7 head(df_filtered, 5)
8 df_filtered <- na.omit(df_filtered)

```

Listing 4: Loại bỏ các cột có tỷ lệ NA > 15%



Bằng câu lệnh `print(names(df_filtered))`, ta có thể kiểm tra lại các cột còn lại sau khi đã loại bỏ các cột có tỷ lệ giá trị *NA* vượt quá 15% (Hình 3.3).

```
> names(df_filtered)
[1] "Process"      "Memory"        "Resolution_WxH"  "Memory_Bandwidth" "Shader"
[6] "Memory_Speed" "Memory_Bus"     "Memory_Type"    "Open_GL"          "Release_Date"
[11] "Dedicated"   "Integrated"   "Direct_X"       "L2_Cache"        "Manufacturer"
[16] "Notebook_GPU" "SLI_Crossfire"
> |
```

Hình 3.3: Các cột còn lại sau khi loại bỏ các cột có tỷ lệ NA > 15%

Mặc dù đã làm sạch dữ liệu bằng cách loại bỏ các cột có tỷ lệ khuyết hoặc số lượng giá trị *NA* cao, vẫn còn một yếu tố khiến việc phân tích dữ liệu trở nên khó khăn, đó là các đơn vị đo, do đó ta cần chuẩn hoá các đơn vị đo trong dữ liệu bằng cách loại bỏ chúng.

```
1 # Chuan hoa don vi do trong cac cot
2 remove_unit_cols <- c("Memory_Bandwidth", "Memory_Speed", "Memory
   _Bus", "Direct_X")
3 main_df <- df_filtered
4 main_df[remove_unit_cols] <- lapply(df_filtered[remove_unit_cols
  ], function(x) {
5   as.numeric(gsub("[^0-9.]", "", x))
6 })
7 clean_cache <- function(x) {
8   main <- as.numeric(sub("KB.*", "", x))           # 2304
9   mult      <- as.numeric(sub(".*\\"\\(x([0-9]+)\\)\"", "\\\\"1", x))  #
  2
10  if (is.na(mult)) mult <- 1
11  return(main * mult)
12 }
13 main_df$L2_Cache <- sapply(main_df$L2_Cache, clean_cache)
```

Listing 5: Chuẩn hoá các đơn vị đo trong dữ liệu

Dữ liệu đã được làm sạch được lưu vào biến `main_df` và được hiện trong Hình 3.4.

	Manufacturer	Release_Date	Memory_Bandwidth	Memory_Speed	L2_Cache	Open_GL	Memory_Type	Resolution_WxH	
	Direct_X	Process	Memory	Shader	Memory_Bus	Dedicated	Integrated	Notebook_GPU	SLI_Crossfire
1	Nvidia	2009	64.0	1000	0	3.3	GDDR3	2560x1600	
2	AMD	2007	106.0	828	0	3.1	GDDR3	2560x1600	
3	AMD	2007	51.2	800	0	3.1	GDDR3	2560x1600	
4	AMD	2007	36.8	1150	0	3.3	GDDR4	2560x1600	
5	AMD	2007	22.4	700	0	3.1	GDDR3	2560x1600	
.									

Hình 3.4: Dữ liệu sau khi làm sạch



4 Thông kê mô tả

Sau khi đã loại bỏ các biến có tỉ lệ dữ liệu khuyết cao, nhóm tiến hành chọn lọc các biến quan trọng. Sau quá trình tham khảo, nhóm nhận thấy thông số quyết định hiệu suất xử lí của GPU là băng thông bộ nhớ (Memory_Bandwidth) vì nó quyết định tốc độ truyền tải dữ liệu giữa bộ nhớ và GPU, đặc biệt với các tác vụ yêu cầu GPU xử lý lượng dữ liệu lớn. Băng thông càng cao thì tốc độ xử lý và khả năng đa nhiệm của hệ thống càng nhanh và ổn định. Ngoài ra, nhóm lựa chọn các biến có tầm ảnh hưởng khác nhau đến Memory_Bandwidth gồm: Manufacturer, Process, Memory_Speed, Memory_Bus, Memory_Type, L2_Cache, Dedicated.

4.1 Tính các giá trị đặc trưng của các biến.

4.1.1 Đối với các biến có giá trị số.

Ta tiến hành lấy các biến trong dữ liệu mẫu ra để tính các giá trị đặc trưng của các biến như sau:

```
1 numeric_data <- main_df[, c("Memory_Bandwidth", "Process", "Memory_Speed", "Memory_Bus", "L2_Cache")]
2 sds <- sapply(numeric_data, sd, na.rm = TRUE)
3 summary(numeric_data)
4 sds
```

Kết quả:

Listing 6: Tính các giá trị đặc trưng của các biến

```
> summary(numeric_data)
Memory_Bandwidth    Process      Memory_Speed      Memory_Bus      L2_Cache
Min. : 4.0   Min. : 14.00   Min. : 275   Min. : 32.0   Min. : 0.0
1st Qu.: 64.0  1st Qu.: 28.00  1st Qu.: 950   1st Qu.: 128.0  1st Qu.: 256.0
Median : 134.4 Median : 28.00  Median :1250   Median :192.0   Median : 512.0
Mean   : 162.7 Mean   : 32.21  Mean   :1275   Mean   : 222.4  Mean   : 847.4
3rd Qu.: 224.4 3rd Qu.: 40.00  3rd Qu.:1527   3rd Qu.: 256.0  3rd Qu.:1024.0
Max.   :1280.0 Max.   :150.00  Max.   :2127   Max.   :4096.0  Max.   :6144.0
> sds
Memory_Bandwidth    Process      Memory_Speed      Memory_Bus      L2_Cache
137.13660   13.76039   395.62258   236.03473   941.24799
```

Nhận xét: từ kết quả ở trên, ta có thể xem giá trị lớn nhất, nhỏ nhất, độ lệch chuẩn và tứ phân vị của các biến cũng như giá trị trung bình của chúng.

Nhóm cũng thử vẽ biểu đồ hộp để trực quan hóa trực quan hóa các số liệu :



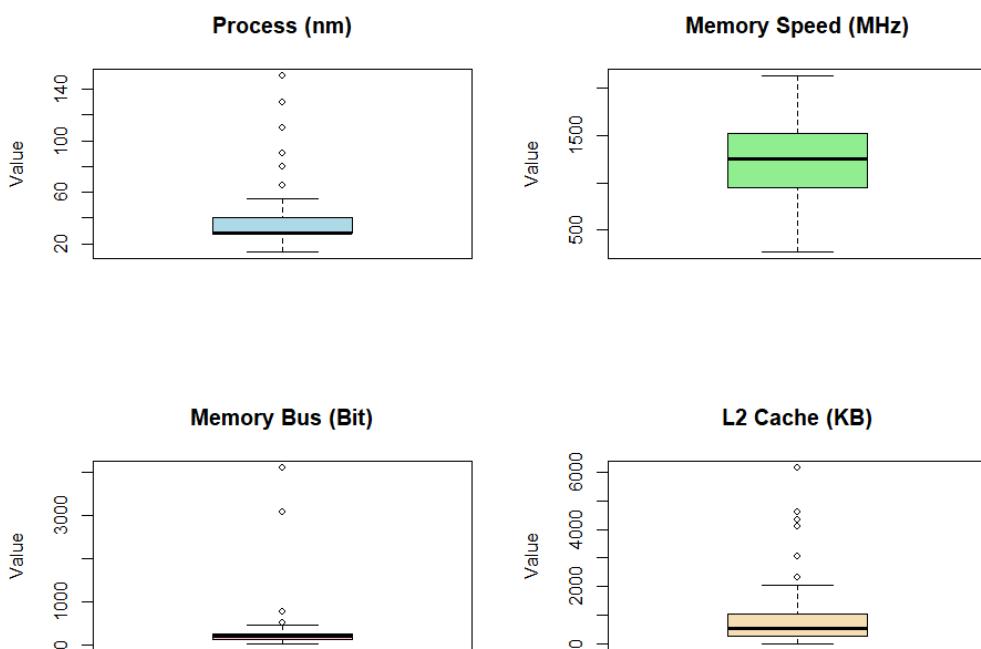
```

1 par(mfrow = c(2, 2)) # Chia khung thành 2 hàng và 2 cột
2
3 boxplot(main_df$Process, main = "Process (nm)", col = "lightblue"
   , ylab = "Value")
4 boxplot(main_df$Memory_Speed, main = "Memory Speed (MHz)", col =
   "lightgreen", ylab = "Value")
5 boxplot(main_df$Memory_Bus, main = "Memory Bus (Bit)", col = "
   pink", ylab = "Value")
6 boxplot(main_df$L2_Cache, main = "L2 Cache (KB)", col = "wheat",
   ylab = "Value")
7
8 par(mfrow = c(1, 1)) # Chính lại nhu ban dau

```

Kết quả:

Listing 7: Vẽ biểu đồ hộp cho các biến số



Hình 4.1: Biểu đồ hộp cho các biến số

Nhận xét: từ biểu đồ hộp ở hình 4.1, ta thấy rằng các biến có một vài ngoại lai nằm ngoài phạm vi của biểu đồ hộp. Bên cạnh đó biểu đồ hộp cũng cho thấy sự khớp như kết quả ở phần



trên. Mặc dù vậy các ngoại lai này có thể ảnh hưởng đến các phân tích thống kê tiếp theo, do đó nhóm quyết định loại bỏ các ngoại lai này.

```
1 # Định nghĩa hàm ngoại lai
2 is_outlier <- function(x) {
3   if (all(is.na(x))) return(rep(FALSE, length(x)))
4
5   Q1 <- quantile(x, 0.25, na.rm = TRUE)
6   Q3 <- quantile(x, 0.75, na.rm = TRUE)
7   IQR <- Q3 - Q1
8
9   lower_bound <- Q1 - 1.5 * IQR
10  upper_bound <- Q3 + 1.5 * IQR
11
12  return(x < lower_bound | x > upper_bound)
13}
14 # Loại bỏ ngoại lai khỏi tập dữ liệu
15 outlier_matrix <- sapply(numeric_data, is_outlier)
16 rows_with_outliers <- rowSums(outlier_matrix) > 0
17 main_df <- main_df[!rows_with_outliers, ]
```

Listing 8: Loại bỏ các ngoại lai khỏi các biến số

4.1.2 Đối với các biến phân loại

Đối với các biến phân loại như Manufacturer, Memory_Type, Dedicated, ta có thể đếm số lượng các nhóm con trong các biến này và tần số xuất hiện của chúng như sau:

```
1 table(main_df$Manufacturer)
2 table(main_df$Memory_Type)
3 table(main_df$Dedicated)
```

Kết quả: Listing 9: Đếm số lượng các nhóm con trong các biến phân loại



```
> table(main_df$Manufacturer)
AMD      ATI  Intel Nvidia
885       76      3   1277
> table(main_df$Memory_Type)
DDR2     DDR3  eDRAM  GDDR2  GDDR3  GDDR4  GDDR5 GDDR5X
16      356      2      3     93      2    1740      29
> table(main_df$Dedicated)
No      Yes
3    2238
```

Nhận xét: từ kết quả trên, ta có thể thấy được rằng có 4 hãng sản xuất GPU chính trong dữ liệu là Nvidia, AMD, Intel và ATI. Trong đó Nvidia và AMD chiếm đa số, còn Intel và ATI chỉ chiếm một phần rất nhỏ, đặc biệt chỉ có 3 quan sát cho Intel, do đó nhóm quyết định loại bỏ các quan sát của Intel khỏi tập dữ liệu. Ngoài ra, các loại bộ nhớ cũng có sự phân phối không đều, với DDR3, GDDR3, GDDR5 chiếm đa số, các loại bộ nhớ còn lại chỉ chiếm một phần rất nhỏ. Do đó nhóm cũng sẽ loại bỏ các quan sát có loại bộ nhớ là DDR2, eDRAM, GDDR2, GDDR4 khỏi tập dữ liệu. Cuối cùng, biến Dedicated có sự phân phối rất không đều, với chỉ 3 quan sát có giá trị No, do đó nhóm quyết định loại bỏ các quan sát này khỏi tập dữ liệu. Và cũng sẽ không dùng biến Dedicated trong các phân tích tiếp theo.

```
1 # 1. Tao danh sach ca gia tri can xoa
2 gia_tri_can_xoa <- c("Intel", "DDR2", "eDRAM", "GDDR2", "GDDR4",
3                           "GDDR5X")
4
5 # 2. Ham thay the
6 thay_the_na <- function(df, danh_sach_xoa) {
7   df[] <- lapply(df, function(x) {
8     replace(x, x %in% danh_sach_xoa, NA)
9   })
10  return(df)
11}
12 main_df <- thay_the_na(main_df, gia_tri_can_xoa)
13 main_df <- na.omit(main_df)
```

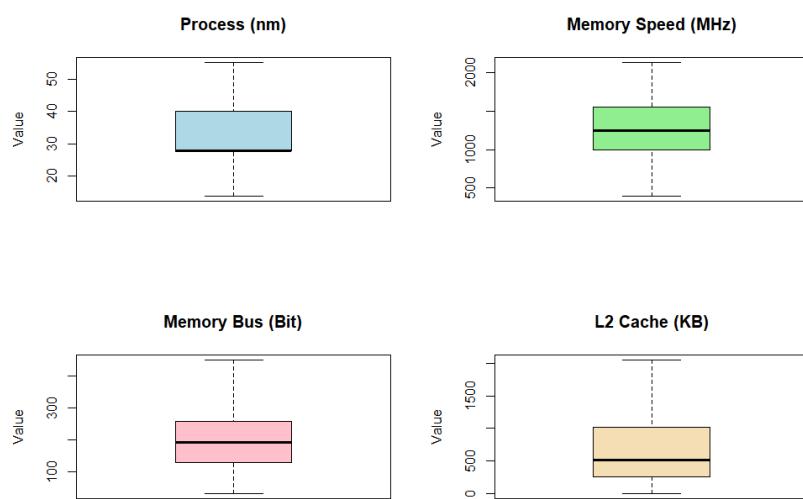
Listing 10: Loại bỏ các quan sát không phù hợp khỏi tập dữ liệu

4.1.3 Kiểm tra lại các đặc điểm của các biến.

Sau khi đã một lần nữa loại bỏ các ngoại lai cũng như quan sát không phù hợp, ta tiến hành kiểm tra lại dữ liệu như sau:

Đối với các biến có giá trị số:

```
> summary(numeric_data)
Memory_Bandwidth    Process      Memory_Speed      Memory_Bus      L2_Cache
Min.   : 6.4       Min.   :14.00     Min.   : 400      Min.   : 32.0      Min.   :  0.0
1st Qu.: 64.0      1st Qu.:28.00     1st Qu.:1000     1st Qu.:128.0     1st Qu.: 256.0
Median :120.0      Median :28.00     Median :1250      Median :192.0      Median : 512.0
Mean   :135.0      Mean   :30.58     Mean   :1293      Mean   :193.4      Mean   : 690.7
3rd Qu.:192.3      3rd Qu.:40.00     3rd Qu.:1600     3rd Qu.:256.0      3rd Qu.:1024.0
Max.   :448.8      Max.   :55.00     Max.   :2127      Max.   :448.0      Max.   :2048.0
> sds
Memory_Bandwidth    Process      Memory_Speed      Memory_Bus      L2_Cache
89.382734        8.879749     382.916229     89.449949     637.044795
```



Hình 4.2: Biểu đồ hộp cho các biến số sau khi loại bỏ ngoại lai và quan sát không phù hợp

Nhận xét: sau khi loại bỏ các ngoại lai và giá trị không phù hợp, ta thấy rằng phương sai của các biến đã giảm đáng kể, ngoài ra trong biểu đồ hộp, các ngoại lai cũng đã không còn, khiến cho biểu đồ hộp trực quan hơn. Bên cạnh đó việc loại bỏ các ngoại lai và quan sát không phù hợp cũng giúp cho các phân tích thống kê tiếp theo trở nên chính xác hơn.

Đối với các biến phân loại:

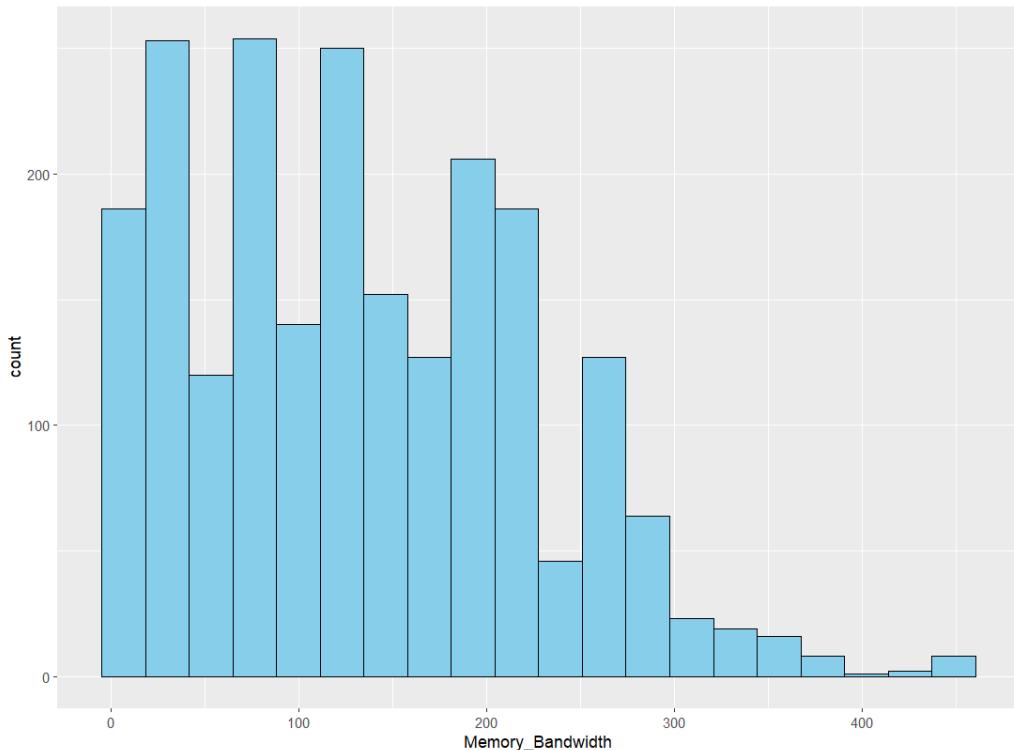
Như đã đề cập ở trên vì biến Dedicated đã bị loại bỏ khỏi tập dữ liệu, ta chỉ kiểm tra lại hai biến còn lại là Manufacturer và Memory_Type như sau:

```
> table(main_df$Manufacturer)
AMD      ATI Nvidia
876      76    1236
> table(main_df$Memory_Type)
DDR3  GDDR3  GDDR5
355     93   1740
```

Nhận xét: như vậy ta chỉ còn lại 3 hãng sản xuất GPU chính là Nvidia, AMD và ATI. Bên cạnh đó ta cũng chỉ còn lại 3 loại bộ nhớ chính là DDR3, GDDR3 và GDDR5.

4.2 Sự phân phối tần số của biến Memory_Bandwidth.

Sự phân phối tần số của biến Memory_Bandwidth được thể hiện qua biểu đồ tần số sau:



Hình 4.3: Biểu đồ tần số của biến Memory_Bandwidth



Nhận xét: từ biểu đồ tần số ở hình 4.3, ta thấy rằng sự phân phối của biến Memory_Bandwidth của các GPU trong tập dữ liệu. Biến không tuân theo phân phối chuẩn, phân phối lệch phải, biến có phần lớn giá trị tập trung thấp, chủ yếu ở ngưỡng dưới 250 GB/s và có xu hướng giảm dần khi giá trị của “Memory_Bandwidth” tăng lên. Điều này phản ánh nhu cầu thị trường đối với các loại GPU. Các trường hợp có băng thông bộ nhớ cao là rất hiếm, có sự chênh lệch lớn giữa các đối tượng quan sát được, cho thấy một số loại GPU hiệu năng cao có khả năng được sử dụng cho các công việc đặc thù. Do đó khi phân tích thống kê chỉ phù hợp khi xem xét các GPU có hiệu năng trung bình, phổ thông.

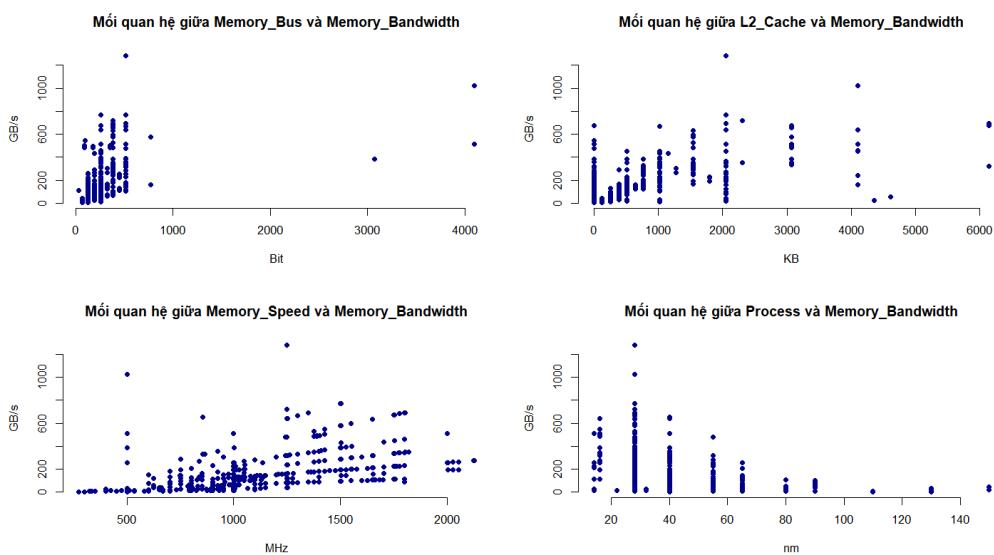
4.3 Đồ thị scatter plot cho biến Memory_Bandwidth theo Memory_Bus, L2_Cache, Memory_Speed, Process.

Ta tiến hành vẽ các đồ thị scatter plot để quan sát mối quan hệ giữa biến Memory_Bandwidth với các biến Memory_Bus, L2_Cache, Memory_Speed, Process như sau:

```
1 ve_bieu_do_tuy_chinh <- function(du_lieu_x, du_lieu_y, nhan_x,
2   nhan_y) {
3   plot(x = du_lieu_x,
4     y = du_lieu_y,
5     main = paste("Mối quan hệ giữa", nhan_x, "và", nhan_y),
6     xlab = nhan_x,
7     ylab = nhan_y,
8     pch = 19,           # Style
9     col = "darkblue",  # Color
10    frame = FALSE)    # Remove extra frame
11 }
```

Kết quả:

Listing 11: Vẽ đồ thị scatter plot.



Hình 4.4: Đồ thị scatter plot cho biến Memory_Bandwidth theo Memory_Bus, L2_Cache, Memory_Speed, Process

Nhận xét:

- Dựa vào đồ thị scatter plot cho Memory_Bandwidth theo Memory_Bus, nhóm thấy được rằng phần lớn điểm dữ liệu tập trung ở vùng Memory_Bus nhỏ dưới 1000 bit, với băng thông không quá 500 GB/s. Điều này cho thấy đây là các Bus bộ nhớ phổ biến cho các hệ thống bộ nhớ. Mối quan hệ phụ thuộc giữa Memory Bandwidth và Memory Bus chỉ ở mức trung bình. Có sự phân tán rất lớn trong băng thông bộ nhớ, cho thấy rằng ngoài Memory Bus còn nhiều yếu tố khác ảnh hưởng đến Memory_Bandwidth.
- Dựa vào đồ thị scatter plot cho Memory_Bandwidth theo L2_Cache, nhóm thấy được rằng phần lớn các điểm dữ liệu tập trung ở vùng Cache nhỏ, dưới 2000 KB, với băng thông bộ nhớ từ thấp đến trung bình. Từ đây cho thấy đa số hệ thống trong tập dữ liệu có dung lượng L2_Cache khiêm tốn. Ở vùng L2 Cache lớn hơn, trên mức 4000 KB, băng thông bộ nhớ có độ phân tán rất rộng, từ mức trung bình đến cao.
- Dựa vào đồ thị scatter plot cho Memory_Bandwidth theo Memory_Speed, nhóm thấy được rằng Phần lớn dữ liệu tập trung ở vùng bộ nhớ thấp, dưới 1000 MHz, băng thông trải dài từ thấp đến trung bình. Giữa Memory_Bandwidth và Memory_Speed có mối quan hệ phụ thuộc rõ ràng, dù ở băng thông cao hơn vẫn có sự phân tán dữ liệu khá lớn. Khi tốc độ bộ nhớ tăng lên thì băng thông bộ nhớ cũng có xu hướng tăng dần, điều này cho thấy Memory_Speed có ảnh hưởng mạnh mẽ đến Memory_Bandwidth.



5 Thống kê suy diễn

5.1 Bài toán 1 mẫu.

Ngày nay VRam hay Memory trong các bộ xử lý đồ họa GPU là một trong những yếu tố quan trọng quyết định hiệu năng của GPU. Có một người dùng cho rằng băng thông bộ nhớ băng thông trung bình của các GPU lớn hơn 2GB. Hãy kiểm định xem ý kiến của người dùng đó có đúng không với mức ý nghĩa 5%.

5.1.1 Mục đích kiểm định.

Dánh giá xem kích thước bộ nhớ trung bình (Memory) của các GPU có lớn hơn 2GB hay không.

5.1.2 Giả thiết nghiên cứu.

- Giả thiết không (H_0): Trung bình kích thước bộ nhớ của các GPU **không lớn hơn** 2GB.
- Giả thiết đối (H_1): Trung bình kích thước bộ nhớ của các GPU **lớn hơn** 2GB.

5.1.3 Thực hiện kiểm định.

Bước 1: Tính các đặc điểm thống kê mẫu.

```
1 Mem <- main_df$Memory
2 n   <- length(Mem)
3 tb  <- mean(Mem)
4 s   <- sd(Mem)
```

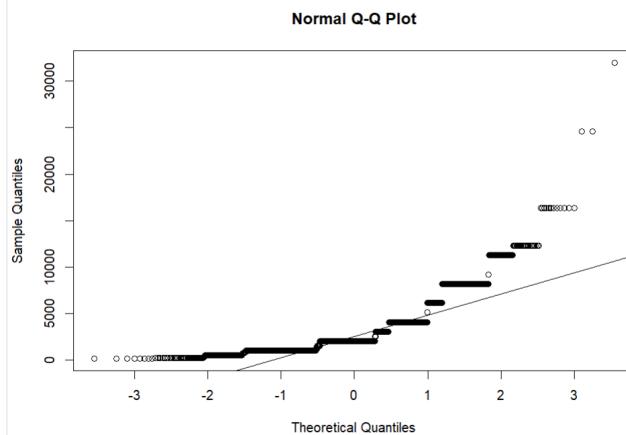
Listing 12: Tính các đặc điểm thống kê mẫu trong R

Kết quả:

```
> print(mem_df)
      n        tb        s
1 2610 3136.564 2838.941
```

Hình 5.1: Bảng kết quả các đặc điểm thống kê mẫu

Bước 2: Kiểm tra phân phối chuẩn của mẫu bằng kiểm định Q-Q plot.



Hình 5.2: Đồ thị Q-Q plot cho mẫu

Nhận xét: có thể thấy rằng mẫu không tuân theo phân phối chuẩn và phân bố theo một vài giá trị cụ thể. Như vậy bài toán trên là kiểm định 1 mẫu không phải phân phối chuẩn với $n > 30$.

Bước 3: Thực hiện kiểm định giả thiết.

Xác định mức ý nghĩa $\alpha = 0.05$. Suy ra miền bác bỏ $RR = (Z_{\alpha/2}, +\infty) = (1.644854, +\infty)$.
Tìm tiêu chuẩn kiểm định:

$$Z_{qs} = \frac{(\bar{X} - \mu_0)}{\frac{S}{\sqrt{n}}} = \frac{3136.564 - 2048}{\frac{2838.941}{\sqrt{2610}}} \approx 19.5892.$$

Tìm các giá trị trong R:

```
1 z0 <- (tb - 2048) / (s/sqrt(n))
2 alpha = 0.05
3 RR <- qnorm(p = 1 - alpha)
```

Listing 13: Tính giá trị Z_{qs} trong R

```
> z0
[1] 19.58925
> RR
[1] 1.644854
```

Hình 5.3: Kết quả tính giá trị Z_{qs} trong R

Kết luận: Vì $Z_{qs} = 19.5892 \in RR$, nên ta bác bỏ giả thiết không H_0 và chấp nhận giả thiết đối H_1 . Kết luận rằng trung bình kích thước bộ nhớ của các GPU lớn hơn 2GB với mức ý nghĩa 5%.



5.2 Bài toán 2 mẫu.

Băng thông bộ nhớ (Memory Bandwidth) là thông số quan trọng ảnh hưởng đến hiệu năng của bộ xử lý đồ họa GPU. Trên thị trường Việt Nam hiện nay có hai hãng là kỳ phùng địch thủ của nhau là Nvidia và AMD và cũng có nhiều cuộc tranh cãi, so sánh hiệu năng GPU của hai hãng này. Có một người dùng cho rằng băng thông bộ nhớ trung bình của Nvidia lớn hơn so với băng thông trung bình của AMD. Hãy kiểm định xem ý kiến của người dùng đó có đúng không với mức ý nghĩa 5%.

5.2.1 Mục đích kiểm định.

Dánh giá xem băng thông bộ nhớ trung bình (Memory Bandwidth) của Nvidia có lớn hơn băng thông bộ nhớ trung bình của AMD hay không.

5.2.2 Giả thiết nghiên cứu.

- Giả thiết không (H_0): Trung bình băng thông bộ nhớ của Nvidia **không lớn hơn** trung bình băng thông bộ nhớ của AMD.
- Giả thiết đối (H_1): Trung bình băng thông bộ nhớ của Nvidia **lớn hơn** trung bình băng thông bộ nhớ của AMD.

5.2.3 Thực hiện kiểm định.

Bước 1: Chia dữ liệu thành 2 nhóm và tính các đặc điểm thống kê mẫu cho từng nhóm.

```
1 nvidia <- subset(main_df, Manufacturer == "Nvidia")$Memory_Bandwidth
2 amd   <- subset(main_df, Manufacturer == "AMD")$Memory_Bandwidth
3
4 # Lay n s trung binh cua du lieu
5 n1 <- length(nvidia)
6 n2 <- length(amd)
7
8 sd1<- sd(nvidia)
9 sd2<- sd(amd)
10
11 tb1<- mean(nvidia)
12 tb2<- mean(amd)
```



```

13
14 bang_ket_qua <- data.frame(
15   "Nhóm" = c(1, 2),
16   "n"      = c(n1, n2),
17   "tb"     = c(tb1, tb2),
18   "sd"     = c(sd1, sd2)
19 )
20 print(bang_ket_qua)

```

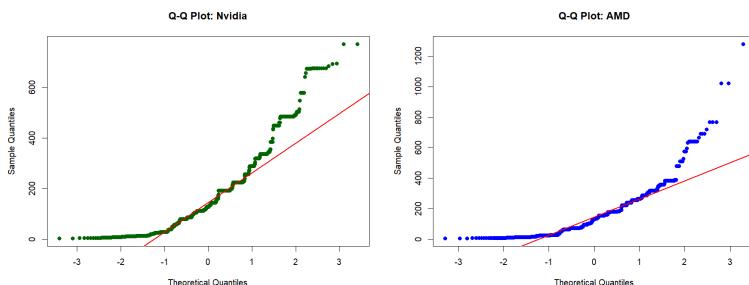
Listing 14: Tính các đặc điểm thống kê mẫu cho từng nhóm trong R

Kết quả:

	Nhóm	n	tb	sd
1	1	1524	165.2467	136.3701
2	2	1005	155.3672	141.1330

Hình 5.4: Bảng kết quả các đặc điểm thống kê mẫu cho từng nhóm

Bước 2: Kiểm tra phân phối chuẩn của hai mẫu bằng Q-Q plot.



Hình 5.5: Đồ thị Q-Q plot cho hai mẫu

Nhận xét: có thể thấy rằng cả hai mẫu đều không tuân theo phân phối chuẩn vì và có xu hướng lệch ở hai đầu. Như vậy bài toán ở đây là kiểm định 2 mẫu không phải phân phối chuẩn với $n > 30$.

Bước 3: Thực hiện kiểm định giả thiết.

Xác định mức ý nghĩa $\alpha = 0.05$. Suy ra miền bác bỏ $RR = (Z_{\alpha/2}, +\infty) = (1.644854, +\infty)$.
Tìm tiêu chuẩn kiểm định:

$$Z_{qs} = \frac{(\bar{X}_1 - \bar{X}_2)}{\sqrt{\frac{S_1^2}{n_1} + \frac{S_2^2}{n_2}}} = \frac{165.2467 - 155.3672}{\sqrt{\frac{136.3701}{1524} + \frac{141.1330}{1005}}} \approx 1.745874$$

Tìm các giá trị trong R:



```
1 z0 <- (tb1-tb2)/sqrt(s1^2/n1 + s2^2/n2)
2 alpha = 0.05
3 RR <- qnorm(p = 1 - alpha)
```

Listing 15: Tính giá trị Z_qs trong R

```
> RR <- qnorm(p = 1 - alpha)
> z0
[1] 1.745874
> RR
[1] 1.644854
>
```

Hình 5.6: Kết quả tính giá trị Z_qs trong R

Kết luận: Vì $Z_{qs} = 1.745874 \in RR$, nên ta bác bỏ giả thiết không H_0 và chấp nhận giả thiết đối H_1 . Kết luận rằng trung bình băng thông bộ nhớ của Nvidia lớn hơn trung bình băng thông bộ nhớ của AMD với mức ý nghĩa 5%.



5.3 Phân tích phương sai (ANOVA) một yếu tố về ảnh hưởng của nhà sản xuất đến băng thông bộ nhớ

5.3.1 Mục đích kiểm định.

Phân tích One-way ANOVA nhằm đánh giá: Băng thông bộ nhớ (Memory_Bandwidth) có sự khác biệt về trung bình giữa các hãng sản xuất GPU hay không?

Các hãng trong dữ liệu: AMD, ATI, Intel và NVIDIA.

5.3.2 Giả thiết nghiên cứu.

- Giả thiết không (H_0): Trung bình băng thông bộ nhớ giữa các hãng sản xuất là **bằng nhau**.
- Giả thiết đối (H_1): Có **ít nhất hai hãng sản xuất** có trung bình băng thông bộ nhớ **khác nhau**.

5.3.3 Kiểm tra giả định thống kê của ANOVA một yếu tố.

Đặt cặp giả thiết thống kê:

- Các quan sát độc lập.
- Phân dữ liệu tuân theo phân phối chuẩn (tương đương với việc biến phụ thuộc tuân theo phân phối chuẩn trong mỗi nhóm).
- Phương sai giữa các nhóm đồng nhất.

5.3.4 Xây dựng mô hình ANOVA một yếu tố trong R.

Bước 1: Xây dựng mô hình ANOVA.

Ta có thể dùng lệnh:

```
1 anova_model <- aov(Memory_Bandwidth ~ Manufacturer, data = main_df)
```

Listing 16: Xây dựng mô hình ANOVA một yếu tố trong R

Bước 2: Kiểm tra các giả định thống kê.

- Tính độc lập:** vì dữ liệu được thu thập từ các mẫu GPU khác nhau nên ta có thể coi các quan sát là độc lập.



- **Phân phối chuẩn:** Ta sử dụng Shapiro-Test và biểu đồ Q-Q plot để kiểm tra giả định phân phối chuẩn của phần dư.

Kiểm định chuẩn hóa phần dư.

H_0 : Phần dư tuân theo phân phối chuẩn.

H_1 : Phần dư không tuân theo phân phối chuẩn.

```
1 res <- residuals(anova_model)
2 shapiro.test(res)
```

Listing 17: Kiểm định Shapiro-Wilk cho phần dư của mô hình ANOVA.

Kết quả của `shapiro.test()` cho phần dư trả về được thể hiện trong hình 5.7, dễ dàng, ta có thể nhận thấy rằng $p_value = 2.2e - 16 << 0.05$, do đó ta bác bỏ giả thiết không H_0 và thừa nhận giả thiết đối H_1 . Kết luận rằng phần dư không tuân theo phân phối chuẩn. Và từ đó khẳng định biến phụ thuộc (Memory_Bandwidth) không tuân theo phân phối chuẩn.

```
> res <- residuals(anova_model)      # lấy phần dư Y-Y
> shapiro.test(res)                 # kiểm định Shapiro-Wilk
                                         Shapiro-Wilk normality test
                                         data: res
                                         W = 0.86529, p-value < 2.2e-16
```

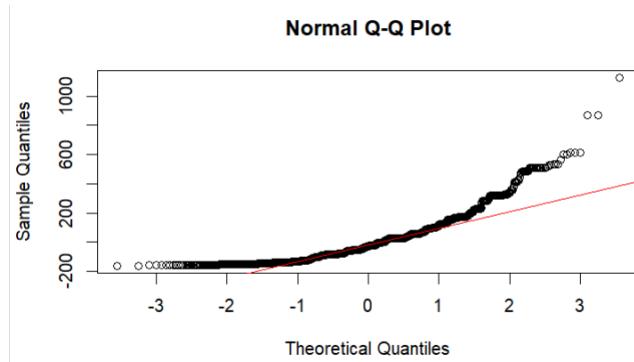
Hình 5.7: Kết quả kiểm định Shapiro-Wilk cho phần dư của mô hình ANOVA

Để trực quan hơn, ta có thể sử dụng biểu đồ Q-Q plot để kiểm tra giả định phân phối chuẩn của phần dư như sau:

```
1 qqnorm(res)
2 qqline(res)
```

Listing 18: Biểu đồ Q-Q plot cho phần dư của mô hình ANOVA

Kết quả được thể hiện ở hình 5.8. Ta có thể thấy rằng các điểm dữ liệu không nằm dọc theo đường thẳng, điều này cho thấy phần dư không tuân theo phân phối chuẩn. Và có xu hướng lệch ở 2 đầu.



Hình 5.8: Biểu đồ Q-Q plot cho phần dư của mô hình ANOVA

Từ 2 phép kiểm định trên, ta chắc chắn kết luận được rằng Memory_Bandwidth không tuân theo phân phối chuẩn.

- **Phương sai đồng nhất:** Ta sử dụng kiểm định Levene để kiểm tra giả định phương sai đồng nhất.

H_0 : Phương sai giữa các nhóm là đồng nhất

H_1 : Phương sai giữa các nhóm không đồng nhất

Ta có thể sử dụng lệnh sau trong R để thực hiện kiểm định Levene:

```
1 leveneTest(Memory_Bandwidth ~ Manufacturer, data = main_df)
```

Listing 19: Kiểm định Levene cho phương sai đồng nhất

Ta thu được kết quả kiểm định Levene được thể hiện trong hình 5.9. Ta thấy rằng $p_value = 2.742e-5 << 0.05$, do đó ta bác bỏ giả thiết không H_0 và chấp nhận giả thiết đối H_1 . Kết luận rằng phương sai giữa các nhóm không đồng nhất.

```
> leveneTest(Memory_Bandwidth~Manufacturer, data = main_df)
Levene's Test for Homogeneity of Variance (center = median)
      Df F value    Pr(>F)
group     3  7.9702 2.742e-05 ***
2606
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' '
```

Hình 5.9: Kết quả kiểm định Levene cho phương sai đồng nhất

Nhận xét: Từ kết quả kiểm tra các giả định thống kê, ta thấy rằng cả hai giả định về phân phối chuẩn và phương sai đồng nhất đều không được thỏa mãn, do đó mô hình ANOVA một yếu tố không phù hợp để phân tích dữ liệu này.



Bước 3: Kiểm tra giả thiết thống kê.

Ta sử dụng lệnh `summary()` để in ra bảng ANOVA.

```
1 summary(anova_model)
```

Listing 20: Bảng ANOVA một yếu tố trong R

Kết quả bảng ANOVA được thể hiện trong hình 5.10. Ta thấy rằng $p_value = 0.00085 << 0.05$, do đó ta bác bỏ giả thiết không H_0 và chấp nhận giả thiết đối H_1 . Kết luận rằng có ít nhất hai hãng sản xuất có trung bình bằng thông bộ nhớ khác nhau.

```
> summary(anova_model)
   Df  Sum Sq Mean Sq F value    Pr(>F)
Manufacturer  3  311663  103888  5.553 0.00085 ***
Residuals   2606 48754357   18709
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Hình 5.10: Bảng ANOVA một yếu tố trong R

Bước 4: So sánh hậu nghiệm (Tukey HSD).

Do mô hình ANOVA chỉ dùng để kiểm tra xem có sự khác biệt về trung bình giữa các nhóm hay không, mà không chỉ ra được cụ thể nhóm nào khác biệt, nên ta cần thực hiện so sánh hậu nghiệm để biết những hàng nào khác nhau có ý nghĩa thống kê.

Ta sử dụng lệnh `TukeyHSD()` trong R để thực hiện so sánh hậu nghiệm như sau:

```
1 tukey_result <- TukeyHSD(anova_model)
```

Listing 21: So sánh hậu nghiệm Tukey HSD trong R

```
> tukey_result <- TukeyHSD(anova_model)
> tukey_result
  Tukey multiple comparisons of means
    95% family-wise confidence level

Fit: aov(formula = Memory_Bandwidth ~ Manufacturer, data = main_df)

$Manufacturer
      diff      lwr      upr      p adj
ATI-AMD    56.161041  14.832212  97.489870 0.0027300
Intel-AMD   -136.167164 -339.475794  67.141466 0.3124803
Nvidia-AMD    9.879555  -4.408335  24.167445 0.2843862
Intel-ATI   -192.328205 -399.201193  14.544783 0.0792325
Nvidia-ATI   -46.281486  -87.100328  -5.462644 0.0188205
Nvidia-Intel  146.046719  -57.158854 349.252292 0.2513201
```

Hình 5.11: Kết quả so sánh hậu nghiệm Tukey HSD trong R

Kết quả so sánh hậu nghiệm Tukey HSD được thể hiện trong hình 5.11. Ta có thể thấy rằng có hai cặp hãng sản xuất có sự khác biệt về trung bình bằng thông bộ nhớ với $p_value < 0.05$,



đó là ATI-AMD và Intel-ATI. Các cặp còn lại không có sự khác biệt về trung bình băng thông bộ nhớ.

Nhận xét: Ta thấy rằng ATI có băng thông trung bình cao hơn so với AMD và NVIDIA.

5.3.5 Nhận xét về mô hình ANOVA một yếu tố.

Từ kết quả kiểm tra các giả định thống kê, ta thấy rằng cả hai giả định về phân phối chuẩn và phương sai đồng nhất đều không được thỏa mãn, do đó mô hình ANOVA một yếu tố không phù hợp để phân tích dữ liệu này. Điều đó khiến nhóm phải thực hiện thêm các kiểm định ANOVA khác trong phần mở rộng. Tuy nhiên, kết quả kiểm định ANOVA và so sánh hậu nghiệm Tukey HSD vẫn cho thấy có sự khác biệt về trung bình băng thông bộ nhớ giữa các hãng sản xuất GPU.



5.4 Áp dụng mô hình hồi quy tuyến tính bội vào phân tích hiệu suất GPU.

5.4.1 Xây dựng mô hình.

- Việc chia dữ liệu thành hai phần (train và test) là cần thiết để tránh overfitting (hiện tượng mô hình học quá kỹ dữ liệu được cung cấp, đến mức nó ghi nhớ dữ liệu thay vì suy ra các quy luật tổng quát). Để đảm bảo mô hình có được khả năng tổng quát hóa tốt, ta chia bộ dữ liệu thành: training_set chiếm 80% và testing_set chiếm 20%. Với 2610 mẫu dữ liệu, tỷ lệ này đảm bảo training_set có đủ dữ liệu để mô hình học hỏi các đặc trưng phức tạp, đồng thời testing_set vẫn có kích thước đủ lớn (522 mẫu) để đánh giá một cách đáng tin cậy hiệu suất thực tế của mô hình.
- Để đánh giá và phân tích quan hệ giữa các thông số của GPU, ta chọn:
 - Biến phụ thuộc: Memory_Bandwidth
 - Biến độc lập: Process, Memory, Memory_Speed, Memory_Bus, L2_Cache.

Lúc này, **mô hình có dạng:**

$$\text{Memory_Bandwidth} = \beta_0 + \beta_1 \text{Process} + \beta_2 \text{Memory} + \beta_3 \text{Memory_Speed} \\ + \beta_4 \text{Memory_Bus} + \beta_5 \text{L2_Cache} + \epsilon$$

```
1 library(caret)
2 # co dinh cach chon cac dong ngau nhie
3 set.seed(111)
4 # chon du lieu de train voi bien muc tieu 'Memory_Bandwidth'
5 train_index <- createDataPartition(main_df$Memory_Bandwidth,
6   p = 0.8,
7   list = FALSE)
8 training_set <- main_df[train_index, ]
9 # tao testing_set bang cac dong con lai
10 testing_set <- main_df[-train_index, ]
11 # tao mo hinh hoi quy tuyen tinh
12 model<-lm(Memory_Bandwidth ~ Process + Memory + Memory_Speed
13   + Memory_Bus + L2_Cache, training_set)
14 summary((model))
```

Listing 22: Xây dựng mô hình hồi quy tuyến tính bội trong R



- Kết quả sau khi chạy trên R:

```

Call:
lm(formula = Memory_Bandwidth ~ Process + Memory + Memory_Speed +
    Memory_Bus + L2_Cache, data = training_set)

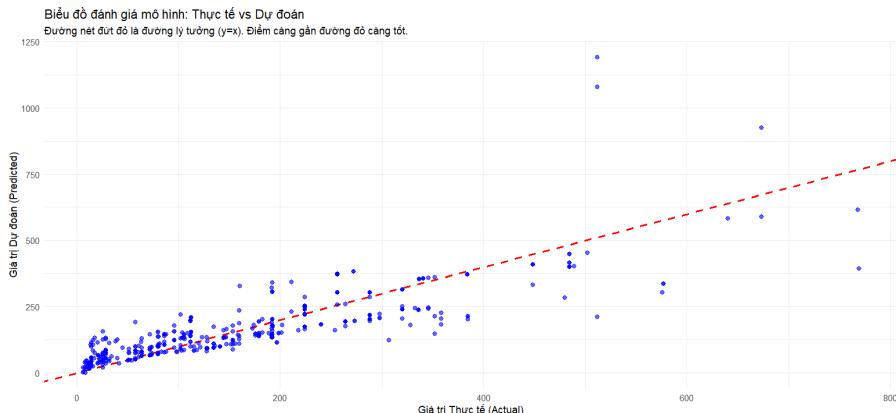
Residuals:
    Min      1Q  Median      3Q     Max 
-511.27 -28.83   -8.90   35.05  683.36 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept) -1.106e+02  1.044e+01 -10.588 < 2e-16 ***
Process      1.004e+00  1.505e-01   6.670 3.27e-11 ***
Memory       2.735e-02  8.032e-04  34.050 < 2e-16 ***
Memory_Speed 7.650e-02  5.347e-03  14.305 < 2e-16 ***
Memory_Bus   1.790e-01  6.484e-03  27.602 < 2e-16 ***
L2_Cache     2.137e-02  2.307e-03   9.266 < 2e-16 ***  
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 68.9 on 2084 degrees of freedom
Multiple R-squared:  0.7536,    Adjusted R-squared:  0.753 
F-statistic: 1275 on 5 and 2084 DF,  p-value: < 2.2e-16

```

Hình 5.12: Kết quả của mô hình hồi quy tuyến tính



Hình 5.13: Đồ thị giữa giá trị thực tế và dự đoán của Memory_Bandwidth (mô hình gốc)

- Nhận xét:

- Hệ số Multiple R-squared có giá trị là 0.7536. Điều này có nghĩa là 75.36% sự biến thiên của biến phụ thuộc Memory_Bandwidth có thể được giải thích bởi sự thay đổi của 5 biến độc lập trong mô hình. Ngoài ra, Hệ số Adjusted R-squared có giá trị là 0.753 (rất gần với Multiple R-squared). Như vậy, mô hình hoạt động khá hiệu quả và không có biến nào bị đưa vào một cách không cần thiết.

- Tất cả các biến dự đoán đều có p-values rất nhỏ ($\ll 0.05$). Điều này khẳng định rằng tất cả 5 yếu tố này đều là yếu tố dự đoán có ý nghĩa thống kê cao.
- Chỉ số Residual Standard Error (RSE) là 68.9. Đây là độ lệch chuẩn của Memory_Bandwidth thực tế so với dự đoán.
- Các hệ số ước lượng β_i có giá trị như sau: $\beta_0 = -110.6$, $\beta_1 = 1.004$, $\beta_2 = 0.02735$, $\beta_3 = 0.0765$, $\beta_4 = 0.179$, $\beta_5 = 0.02137$.

- **Kết luận:** Đường thẳng hồi quy ước lượng của mô hình:

$$\begin{aligned}\text{Memory_Bandwidth} = & -110.6 + 1.004 \cdot \text{Process} + 0.02735 \cdot \text{Memory} \\ & + 0.0765 \cdot \text{Memory_Speed} + 0.179 \cdot \text{Memory_Bus} \\ & + 0.02137 \cdot \text{L2_Cache}\end{aligned}$$

5.4.2 Kiểm tra giả định của mô hình hồi quy.

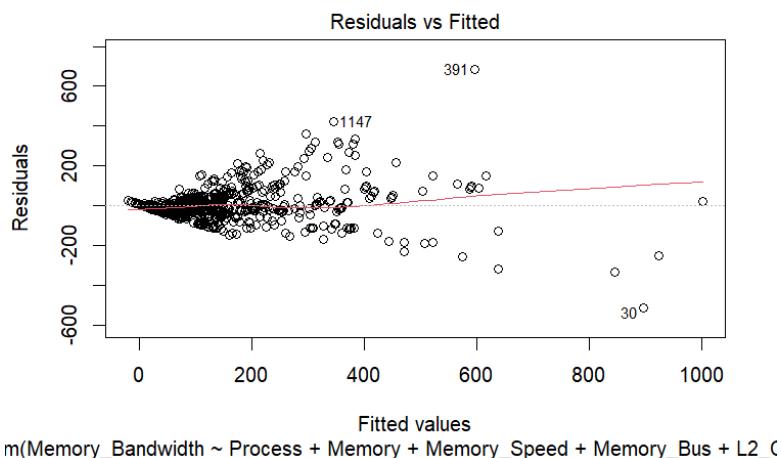
- **Tính tuyến tính.**

Ta dùng mô hình chẩn đoán Residuals-Fitted để kiểm tra bằng lệnh:

```
1 plot(model, which=1) # Residuals vs Fitted
```

Listing 23: Vẽ đồ thị Residuals-Fitted trong R

Kết quả:



Hình 5.14: Biểu đồ Residuals vs Fitted



Nhận xét: Đường xu hướng (màu đỏ) nhìn chung tương đối phẳng nhưng vẫn có xu hướng tăng lên theo chiều tăng của fitted values. Như vậy, mô hình chỉ có thể thỏa mãn tương đối tính tuyến tính.

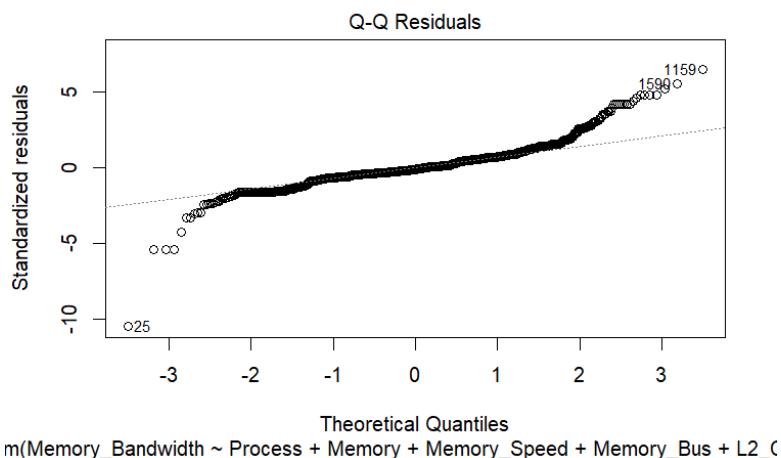
- **Sai số có phân phối chuẩn.**

Ta có thể sử dụng biểu đồ Q-Q plot để kiểm tra giả định phân phối chuẩn của phần dư như sau:

```
1 plot(model, which=2)
```

Listing 24: Biểu đồ Q-Q plot cho phần dư của mô hình hồi quy tuyến tính

Kết quả:



Hình 5.15: Biểu đồ Q-Q

Nhận xét: Biểu đồ Q-Q cho thấy phần lớn các điểm nằm sát đường chéo nên sai số của mô hình gần như là có phân phối chuẩn, đặc biệt là ở vùng trung tâm. Tuy nhiên, phần đuôi của đồ thị lại có xu hướng lệch lên so với đường tham chiếu.

- **Phương sai đồng nhất.**

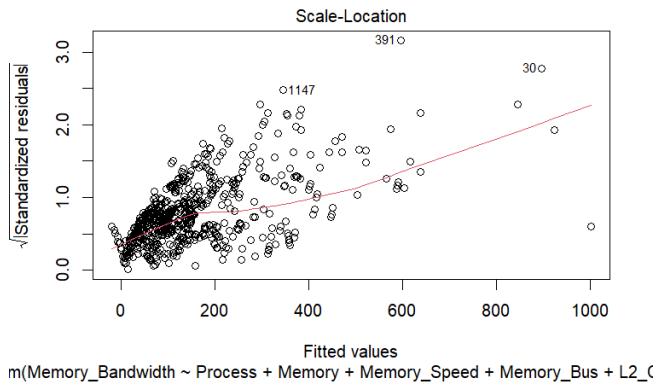
Ta dùng mô hình chẩn đoán Scale-Location để kiểm tra bằng lệnh:

```
1 plot(model, which=3) # Scale-Location
```

Listing 25: Vẽ đồ thị Scale-Location trong R



Kết quả:



Hình 5.16: Biểu đồ Scale-Location

Nhận xét: Đường xu hướng (màu đỏ) có độ dốc lớn, các điểm có độ phân tán ngày càng cao. Có thể nhận định: mô hình không thỏa mãn tốt tính chất đồng nhất của phương sai.

- **Tính độc lập của sai số.**

Ta dùng kiểm định Durbin-Watson để kiểm tra tính chất này:

```
1 durbinWatsonTest(model) # Kiểm tra tính độc lập của sai số
```

Listing 26: Kiểm định Durbin-Watson trong R

Kết quả:

```
durbinWatsonTest(model) # Kiểm tra tính độc lập của sai số
  lag      Autocorrelation      D-W Statistic      p-value
    1          0.47708            1.045552            0
  Alternative hypothesis: rho != 0
```

Nhận xét: Tính độc lập của sai số không được thỏa mãn vì hệ số Tự tương quan là tương đối lớn (0.47708).

- **Kiểm tra đa cộng tuyến.**

Để kiểm tra tính đa cộng tuyến giữa các biến độc lập trong mô hình, ta sử dụng hệ số phỏng đại phương sai (Variance Inflation Factor - VIF). Hệ số VIF đo lường mức độ mà



phương sai của các ước lượng hồi quy bị phóng đại do sự đa cộng tuyến giữa các biến độc lập. Trong RStudio, ta có thể sử dụng hàm `vif()` từ gói `car` để tính hệ số VIF cho từng biến độc lập trong mô hình hồi quy tuyến tính.

```
library(car)
vif(model) #Kiểm tra đa cộng tuyến
```

Kết quả:

```
vif(model) #Kiểm tra đa cộng tuyến
      Process      Memory Memory_Speed   Memory_Bus      L2_Cache
1.858279    2.202248     1.995969     1.098141    2.065179
```

Nhận xét: Có thể thấy, hầu hết các biến có chỉ số VIF tuyến ở mức thấp. Như vậy, mô hình có tính ổn định cao về mặt đa cộng tuyến.

5.4.3 Nhận xét về mô hình hồi quy tuyến tính.

Mô hình hồi quy tuyến tính được xây dựng nhìn chung có thể thỏa mãn tương đối một số giả định quan trọng nhưng cũng tồn tại nhiều hạn chế.



6 Thảo luận và mở rộng

6.1 Thảo luận

Bài báo cáo đã cung cấp cơ sở dữ liệu và phân tích các thuộc tính về GPU cho ta thấy cái nhìn bao quát và hiểu rõ hơn các yếu tố ảnh hưởng đến hiệu suất của GPU. Các mô hình được xây dựng để dự đoán có thể đánh giá hiệu năng GPU thông qua các thông số kỹ thuật. Sau khi phân tích, ta có một số nhận xét sau:

- Thống kê mô tả:
 - Cho thấy sự phân bố không đồng đều của các thông số, đặc biệt là Memory_Bandwidth – thông số quyết định hiệu suất xử lý của GPU phần lớn giá trị tập trung ở mức thấp.
 - Các biến số như Memory_Speed, Memory_Bus, L2_Cache có ảnh hưởng nhất định đến Memory_Bandwidth, tuy nhiên mức độ ảnh hưởng khác nhau và có sự phân tán lớn trong dữ liệu.
 - Trong dữ liệu chỉ gồm có một số ít nhà sản xuất chiếm đa số, điều này có thể ảnh hưởng đến tính tổng quát của mô hình dự đoán.
 - Dữ liệu có một số ngoại lai, điều này có thể ảnh hưởng đến trực tiếp đến kết quả phân tích và các mô hình dự đoán.
- Thống kê suy diễn:
 - Mô hình hồi quy tuyến tính đa biến cho thấy các biến Process, Memory, Memory_Speed, Memory_Bus, L2_Cache có ảnh hưởng đáng kể đến Memory_Bandwidth, tuy nhiên mô hình có R-squared không cao, chỉ khoảng 0.7536, do đó đây chưa phải là mô hình hồi quy tốt nhất.
 - Như đã đề cập ở trên, phần lớn các biến xét đến trong các mô hình và phân tích đều không tuân theo phân phối chuẩn, điều này ảnh hưởng đến tính chính xác của các mô hình dự đoán, buộc nhóm phải tìm các giải pháp để cải thiện các mô hình đó.

6.2 Mở rộng

Cải thiện mô hình hồi quy tuyến tính bội.

Như đã đề cập ở phần thống kê suy diễn, mô hình hồi quy tuyến tính bội hiện tại có R-squared không cao, chỉ khoảng 0.7536 vì biến Memory_Bandwidth không tuân theo phân phối chuẩn. Do đó Ta có thể thử dùng phép biến đổi logarit (log) cho các biến để ổn định phương



sai của các sai số, làm cho độ phân tán của sai số giảm xuống, phương sai trở nên đồng nhất hơn. Vì trong dữ liệu có giá trị 0 ở một số biến (VD: L2_Cache) nên ta sẽ lấy $\log(X + 1)$ để tránh giá trị lỗi. Ta thực hiện như sau:

```

1 training_set_log <- training_set
2 testing_set_log <- testing_set
3 training_set_log$Memory <- log(training_set_log$Memory + 1)
4 training_set_log$Memory_Bus <- log(training_set_log$Memory_Bus +
   1)
5 training_set_log$Memory_Bandwidth <- log(training_set_log$Memory_Bandwidth +
   1)
6
7 training_set_log$Memory_Speed <- log(training_set_log$Memory_Speed +
   1)
8 training_set_log$L2_Cache <- log(training_set_log$L2_Cache + 1)
9 training_set_log$Process <- log(training_set_log$Process + 1)
10 testing_set_log$Memory <- log(testing_set_log$Memory + 1)
11 testing_set_log$Memory_Bus <- log(testing_set_log$Memory_Bus + 1)
12 testing_set_log$Memory_Bandwidth <- log(testing_set_log$Memory_Bandwidth +
   1)
13
14 testing_set_log$Memory_Speed <- log(testing_set_log$Memory_Speed +
   1)
15 testing_set_log$L2_Cache <- log(testing_set_log$L2_Cache + 1)
16 testing_set_log$Process <- log(testing_set_log$Process + 1)
17 new_model <- lm(
18   Memory_Bandwidth ~ Process + Memory + Memory_Speed +
19   Memory_Bus + L2_Cache ,
20   training_set_log
21 )
22 summary(new_model)

```

Listing 27: Phép biến đổi logarit cho các biến.

Kết quả:



```
Call:  
lm(formula = Memory_Bandwidth ~ Process + Memory + Memory_Speed +  
    Memory_Bus + L2_Cache, data = training_set_log)  
  
Residuals:  
    Min      1Q  Median      3Q     Max  
-0.8576 -0.2314 -0.0200  0.2240  1.5997  
  
Coefficients:  
            Estimate Std. Error t value Pr(>|t|)  
(Intercept) -11.058366  0.325446 -33.979 <2e-16 ***  
Process       -0.077328  0.037783  -2.047  0.0408 *  
Memory        0.258541  0.015337  16.857 <2e-16 ***  
Memory_Speed   1.253716  0.033029  37.958 <2e-16 ***  
Memory_Bus     0.979913  0.016856  58.134 <2e-16 ***  
L2_Cache      -0.001152  0.003559  -0.324  0.7462  
---  
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1  
  
Residual standard error: 0.349 on 2084 degrees of freedom  
Multiple R-squared:  0.8874,   Adjusted R-squared:  0.8872  
F-statistic: 3286 on 5 and 2084 DF,  p-value: < 2.2e-16
```

Hình 6.1: Kết quả mô hình hồi quy tuyến tính bội sau khi biến đổi logarit

Kết luận:

Đường thẳng hồi quy ước lượng của mô hình:

$$\begin{aligned}\ln(\text{Memory_Bandwidth} + 1) = & -11.06 - 0.0773 \ln(\text{Process} + 1) \\ & + 0.2585 \ln(\text{Memory} + 1) \\ & - 0.0012 \cdot \ln(\text{L2_Cache} + 1) \\ & + 0.9799 \ln(\text{Memory_Bus} + 1) \\ & + 1.2537 \ln(\text{Memory_Speed} + 1)\end{aligned}$$

Có thể dễ dàng nhận thấy, các chỉ số đã có sự cải thiện rõ rệt so với mô hình trước. Do đó, mô hình hoạt động hiệu quả, chính xác hơn.

Áp dụng mô hình để dự đoán giá trị Memory_Bandwidth:

```
1 pred_log_values <- predict(new_model, newdata = testing_set_log)  
2  
3 # Chuyen tu log ve gia tri binh thuong  
4 pred_original_scale <- exp(pred_log_values) - 1  
5  
6 # Lay gia tri thuc te  
7 actual_original_scale <- testing_set$Memory_Bandwidth
```



```

8
9 # Tao df cho de nhin
10 results_summary_display_original <- data.frame(
11   Memory_Bandwidth_Actual = actual_original_scale,
12   Memory_Bandwidth_Predict = pred_original_scale,
13   Error_Original = actual_original_scale - pred_original_scale # 
14   Tinh sai so
14 )
15
16 head(results_summary_display_original, 10)

```

Listing 28: Dự đoán giá trị Memory_Bandwidth sử dụng mô hình mới.

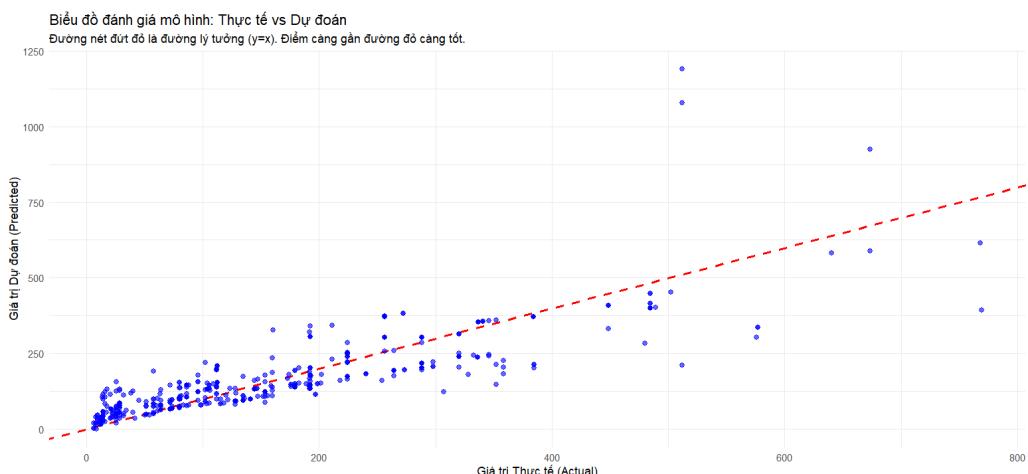
Kết quả:

```

> # Hiển thị 10 hàng đầu tiên của kết quả dự đoán (ở thang độ gốc)
> head(results_summary_display_original, 10)
  Memory_Bandwidth_Actual Memory_Bandwidth_Predict Error_Original
1                  64.0          91.01085      -27.010847
2                  57.6          55.80195       1.798053
3                 288.4          376.28867     -87.888669
4                 673.2          653.20084      19.999158
5                 288.4          376.28867     -87.888669
6                 512.0          1294.93726    -782.937261
7                 288.4          376.28867     -87.888669
8                 288.4          376.28867     -87.888669
9                 264.0          339.62518     -75.625175
10                147.2          164.02141     -16.821412

```

Hình 6.2: Kết quả dự đoán giá trị Memory_Bandwidth sử dụng mô hình mới



Hình 6.3: Đồ thị giữa giá trị thực tế và dự đoán của Memory_Bandwidth (mô hình cải tiến)



Nhận xét:

Có thể thấy rằng mô hình sau khi biến đổi logarit đã cải thiện đáng kể độ chính xác trong việc dự đoán giá trị Memory_Bandwidth. Đồ thị ở hình 6.3 cho thấy các điểm dữ liệu tập trung gần đường chéo $y = x$ hơn so với mô hình gốc ở hình 5.13, điều này chứng tỏ rằng sai số giữa giá trị thực tế và giá trị dự đoán đã giảm đáng kể. Do đó, việc áp dụng phép biến đổi logarit đã giúp ổn định phương sai của sai số và nâng cao hiệu suất của mô hình hồi quy tuyến tính bội.

7 Nguồn dữ liệu và nguồn code

- **Nguồn dữ liệu:** Dữ liệu được lấy từ trang Kaggle tại địa chỉ: [Kaggle](#).
- **Nguồn code:** Toàn bộ mã nguồn R được sử dụng trong báo cáo này đều được viết bởi nhóm sinh viên và tham khảo từ các tài liệu, bài báo, và nguồn học thuật khác để thực hiện báo cáo.



Tài liệu tham khảo

- [1] Nguyễn Đình Huy, *Giáo trình Xác suất và Thống kê*, NXB Đại học Quốc gia TP.HCM, 2019.
- [2] Nguyễn Kiều Dung, *Bài giảng Xác suất Thống kê*, Tài liệu lưu hành nội bộ, Trường Đại học Bách Khoa - ĐHQG TP.HCM, 2024.