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

XÁC SUẤT THỐNG KÊ

Bài tập lớn

Phân tích và Dự đoán Giá tiền CPU trên Thị trường

Hướng dẫn viên: Hoàng Văn Hà
Sinh viên: Nguyễn Đoàn Minh Tâm - 2213026
Nguyễn Minh Tân - 2213057
Hồ Gia Thắng - 2213187
Võ Tá Bảo Long - 2211911
Trần Phước Nhật - 2212412

Tp. Hồ Chí Minh, Tháng 05/2024

Mục lục

I Tổng quan dữ liệu

1 Đề tài

Trong thời đại khoa học kỹ thuật ngày càng phát triển như hiện nay, máy tính đã trở thành một phần không thể thiếu đối với cuộc sống của chúng ta. Hằng ngày, ta sử dụng nhiều loại máy tính khác nhau từ chiếc điện thoại thông minh, máy tính bảng, laptop đến máy tính để bàn. Những thiết bị này không chỉ phục vụ nhu cầu làm việc, học tập, mà còn giúp chúng ta giải trí và kết nối với thế giới. Trong đó, bộ vi xử lý (CPU) đóng vai trò cốt lõi, quyết định hiệu năng và khả năng xử lý của máy tính. Chính vì vậy mà giá trị của 1 chiếc máy tính thường phụ thuộc rất nhiều vào CPU của chiếc máy đó. Do vậy, việc phân tích và dự đoán giá tiền CPU trên thị trường trở nên cấp thiết nhằm giúp người tiêu dùng và doanh nghiệp đưa ra những quyết định mua sắm và đầu tư hợp lý, đồng thời thúc đẩy sự cạnh tranh và phát triển công nghệ trong ngành công nghiệp này.

Đề tài "Phân tích và Dự đoán Giá tiền CPU trên Thị trường" tập trung vào việc nghiên cứu và phân tích các yếu tố ảnh hưởng đến giá tiền của CPU. Sử dụng các phương pháp xác suất và thống kê, đề tài sẽ tiến hành phân tích dữ liệu từ thị trường CPU, xác định các yếu tố chính ảnh hưởng đến giá cả và xây dựng mô hình dự đoán giá tiền của CPU.

II Kiến thức nền

1 Analysis of Variance - Phân tích phương sai (ANOVA)

Phân tích phương sai (Analysis of Variance) hay còn gọi là kiểm định ANOVA là một kỹ thuật thống kê tham số được phát triển vào năm 1918 bởi nhà thống kê nổi tiếng người Anh Ronald Aylmer Fisher. Mục tiêu của kỹ thuật này là nhằm so sánh trung bình của nhiều nhóm (tổng thể) dựa trên các trị trung bình của các mẫu quan sát từ các nhóm này và thông qua kiểm định giả thuyết của kết luận về sự bằng nhau của các trung bình tổng thể này. Trong nghiên cứu, phân tích phương sai được dùng như một công cụ để xem xét ảnh hưởng của một yếu tố nguyên nhân (định tính) đến một yếu tố kết quả (định lượng).

Phân tích phương sai một yếu tố

Phân tích phương sai một yếu tố (One way ANOVA) là phân tích ảnh hưởng của một yếu tố nguyên nhân (dạng biến định tính) ảnh hưởng đến một yếu tố kết quả (dạng biến định lượng) đang nghiên cứu. Trường hợp k tổng thể có phân phối bình thường và phương sai bằng nhau: Giả sử rằng chúng ta muốn so sánh trung bình của k tổng thể (ví dụ k = 3) dựa trên những mẫu ngẫu nhiên độc lập gồm $n_1, n_2, n_3, ..., n_k$ quan sát từ k tổng thể. Cần ghi nhớ ba giả định sau đây về các nhóm tổng thể được tiến hành phân tích ANOVA:

- Các tổng thể này có phân phối bình thường.
- Các phương sai tổng thể bằng nhau.
- Các quan sát được lấy mẫu là độc lập nhau.

Nếu trung bình của các tổng thể được ký hiệu là $\mu_1, \mu_2, \mu_3, ..., \mu_k$ thì khi các giả định trên được đáp ứng, mô hình phân tích phương sai một yếu tố ảnh hưởng được mô tả dưới dạng kiểm định giả thuyết như sau: $H_0 : \mu_1, \mu_2, \mu_3, ..., \mu_k$. Giả thuyết cho rằng trung bình của k tổng thể đều bằng nhau (về mặt nghiên cứu liên hệ thì giả thuyết này cho rằng yếu tố nguyên nhân không có tác động gì đến vấn đề ta đang nghiên cứu). Và giả thuyết đối là: H1: Tồn tại ít nhất 1 cặp có $\mu_i \neq \mu_j, i \neq j$ Để kiểm định ta đưa ra 3 giả thiết sau:

- Mỗi mẫu tuân theo phân phối chuẩn $N(\mu, \phi^2)$.
- Các phương sai tổng thể bằng nhau.
- Ta lấy k mẫu độc lập từ k tổng thể. Mỗi mẫu được quan sát n_j lần.

Các bước phân tích ANOVA:

1. Tính các giá trị trung bình: Trung bình từng cột. trung bình chung.
2. Tính tổng độ lệch bình phương:
   Tổng độ lệch bình phương được sinh ra bởi yếu tố cột $SSG = \sum_{i=1}^{k} n_i(\overline{x_i} - \overline{x})^2$
   Tổng độ lệch bình phương chung: $SST = \sum_{i=1}^{k}\sum_{j=1}^{n_i}(x_{ij} - \overline{x})^2$
   Tổng độ lệch bình phương của k cột: $SSW = SST - SSG$
3. Tính phương sai:
   $MSG = \frac{SSG}{k-1}$
   $MSW = \frac{SSW}{n-k}$
4. Giá trị kiểm định: $F = \frac{MSG}{MSW}$
5. Bác bỏ giả thuyết H_0 khi $F > F_{k-1,n-k,\alpha}$

Phân tích phương sai hai yếu tố

Phân tích phương sai 2 yếu tố nhằm xem xét cùng lúc hai yếu tố nguyên nhân (dưới dạng dữ liệu định tính) ảnh hưởng đến yếu tố kết quả (dưới dạng dữ liệu định lượng) đang nghiên cứu. Ví dụ: Nghiên cứu ảnh hưởng của loại chất đốt và loại lò sấy đến tỷ lệ vải loại 1 sấy khô. Phân tích phương sai 2 yếu tố giúp chúng ta đưa thêm yếu tố nguyên nhân vào phân tích làm cho kết quả nghiên cứu càng có giá trị. Giả sử ta nghiên cứu ảnh hưởng của 2 yếu tố nguyên nhân định tính đến một yếu tố kết quả định lượng nào đó. Ta lấy mẫu không lặp lại, sau đó các đơn vị mẫu của yếu tố nguyên nhân thứ nhất sắp xếp thành K nhóm (cột), các đơn vị mẫu của yếu tố nguyên nhân thứ hai sắp xếp thành H khối (hàng). Như vậy, ta có bảng kết hợp 2 yếu tố nguyên nhân gồm K cột và H hàng và $(K \times H)$ ô dữ liệu. Tổng số mẫu quan sát là $n = (K \times H)$.

Để kiểm định ta đưa ra 2 giả thiết sau:

- Mỗi mẫu tuân theo phân phối chuẩn $N(\mu, \phi^2)$.
- Ta lấy K mẫu độc lập từ K tổng thể, H mẫu độc lập từ H tổng thể. Mỗi mẫu được quan sát 1 lần không lặp.

Các bước tiến hành:

1. Tính các số trung bình
2. Tính tổng các độ lệch bình phương
3. Tính các phương sai
4. Kiểm định giả thuyết

2 Multivariate Linear Regression - Hồi quy Tuyến tính (MLR)

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. Nó mô hình toán học biến không xác định hoặc phụ thuộc và biến đã biết hoặc độc lập như một phương trình tuyến tính. Ví dụ, giả sử rằng bạn có dữ liệu về chi phí và thu nhập của bạn trong năm ngoái. Kỹ thuật hồi quy tuyến tính phân tích dữ liệu này và xác định rằng chi phí của bạn là một nửa thu nhập của bạn. Sau đó, họ tính toán một chi phí trong tương lai không rõ bằng cách giảm một nửa thu nhập được biết đến trong tương lai.

Phân tích hồi quy tuyến tính phải sửa đổi hoặc biến đổi các giá trị dữ liệu về mặt toán học để đáp ứng bốn giả định sau đây:

1. Mối quan hệ tuyến tính
2. Phần dư độc lập
3. Tính chuẩn
4. Phương sai không đổi

Công thức tổng quát của Hồi quy tuyến tính như sau:

$$y_i = \beta_0 + \beta_1 x_{i,1} + \beta_2 x_{i,2} + ... + \beta_k x_{i,k} + \epsilon_i$$

Trong đó:

- y: biến phụ thuộc, là biến chịu tác động của biến khác.
- x, x_1, x_2, x_n: biến độc lập, là biến tác động lên biến khác.
- β_0: hằng số hồi quy, hay còn được gọi là hệ số chặn. Đây là chỉ số nói lên giá trị của y sẽ là bao nhiêu nếu tất cả x cùng bằng 0. Nói cách khác, chỉ số này cho chúng ta biết giá trị của y là bao nhiêu nếu không có các x. Khi biểu diễn trên đồ thị Oxy, β_0 là điểm trên trục Oy mà đường hồi quy cắt qua.
- $\beta_1, \beta_2, \beta_n$: hệ số hồi quy, hay còn được gọi là hệ số góc. Chỉ số này cho chúng ta biết về mức thay đổi của y gây ra bởi x tương ứng. Nói cách khác, chỉ số này nói lên có bao nhiêu đơn vị y sẽ thay đổi nếu x tăng hoặc giảm một đơn vị.
- Sai số ϵ: chỉ số này càng lớn càng khiến cho khả năng dự đoán của hồi quy trở nên kém chính xác hơn hoặc sai lệch nhiều hơn so với thực tế. Sai số trong hồi quy tổng thể hay phần dư trong hồi quy mẫu đại diện cho hai giá trị, một là các biến độc lập ngoài mô hình, hai là các sai số ngẫu nhiên.

III Tiền xử lý số liệu

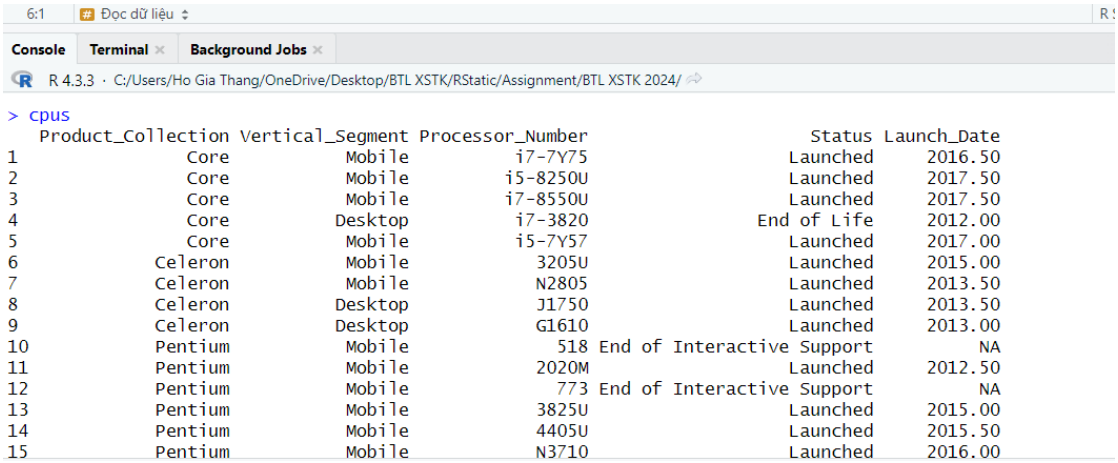
1 Các biến dữ liệu đầu vào được sử dụng trong bài tập này

Biến	Kiểu dữ liệu	Đơn vị	Mô tả
Product_Collection	data text	none	Danh mục sản phẩm CPU
Processor_Number	data text và số	none	Mã định danh của các CPU
Launch Date	Qx'yy	none	Ngày phát hành CPU Với x là quý yy là 2 số cuối của năm phát hành
Lithography	number	nm	Kích thước các bóng bán dẫn dùng trong mạch CPU
Recommended_Customer_Price	number	$	Giá tiền CPU
Max_Memory_Size	number	GB	Dung lượng bộ nhớ chính (RAM)
Cache	number	Byte	Dung lượng bộ nhớ Cache
Bus_Speed	number	Hz	Tốc độ Bus của CPU
nb_of_Threads	number	none	Số luồng thực thi mà CPU có thể thực thi

2 Đọc dữ liệu

Để đọc dữ liệu trong R, ta sử dụng lệnh `read.csv()` để đọc dữ liệu từ file CSV đầu vào.

Sau khi đọc file gõ `cpus` trên console để kiểm tra xem đã có dữ liệu hay chưa.



Hình 1: Kiểm tra dữ liệu trên console sau khi thực hiện lệnh đọc dữ liệu

3 Xử lý dữ liệu

3.1 Quan sát dữ liệu

Từ dữ liệu này, ta nhận thấy rằng tập dữ liệu đọc vào có các vấn đề sau:

- Tập dữ liệu này có 40 cột và 2284 hàng.
- Tập dữ liệu không đồng nhất về đơn vị. Có quá nhiều biến có đơn vị khác nhau. Như biến Cache, Bus_Speed, Max_Memory_Size,... Các biến này sẽ được khảo sát ở phần sau.
- Tập dữ liệu có nhiều dữ liệu bị khuyết. Do nhiều nguyên nhân mà một tập dữ liệu thường bị mất một số dữ liệu bên trong nó. Vì vậy, người ta thường sẽ có những phương pháp để xử lý dữ liệu sẽ được đề cập ở phần sau.
- Có nhiều biến dữ liệu gây nhiễu. Khiến ta không thể khảo sát chính xác đối tượng mong muốn. Chính vì lý do này mà ta sẽ không khảo sát toàn bộ các biến trong tập dữ liệu mà chỉ khảo sát những biến có mức độ liên quan cao đến dữ liệu.

3.2 Định dạng dữ liệu

Như đã đề cập bên trên, tập dữ liệu gốc có quá nhiều biến không đúng định dạng. Vì vậy, ta cần phải định dạng lại nội dụng của những biến này để thuận tiện cho việc xử lý dữ liệu sau này.

Đầu tiên, ta sẽ định dạng lại Product_Collection. Để có thể xử dụng được hàm này, ta cần phải sử dụng thư viện dplyr. Trong đoạn mã này, hàm 'gsub' được sử dụng để thay thế các chuỗi khớp với mẫu đã chỉ định trong biến Product_Collection bằng các chuỗi tương ứng. Kết quả thu được sau khi gọi hàm này là các collection. Bao gồm: *Core*, *X-series*, *Celeron*, *Pentium*, *Quark*, *Core*, *Atom*, *Itanium*, *Xeon*.

Tiếp theo, ta sẽ định dạng lại Recommended_Customer_Price. Khi quan sát biến Recommended_Customer_Price ta nhận ra rằng: Giá tiền đôi lúc sẽ là một giá trị cố định, nhưng đôi lúc cũng có thể là 1 khoảng tiền. Để tiện cho việc xử lý. Đối với những giá trị là khoảng tiền ta thay thế nó bởi giá trị trung bình của khoảng đó.

Đối với Lithography ta giữ nguyên đơn vị(nm) và chuyển về dạng số để tiện cho việc thao tác.

Đối với cache, Bus_Speed, Max_Memory_Size ta sẽ đưa về các kiểu dữ liệu lần lượt là byte, GB/s, Hz, GB.

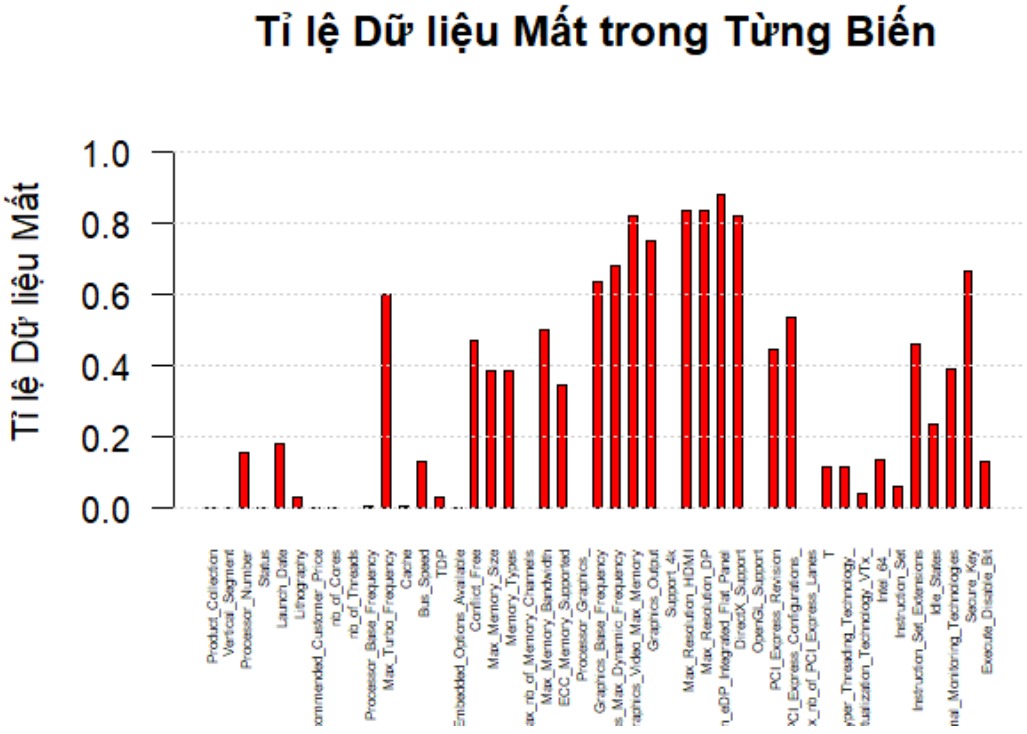
Do Launch_Date sử dụng định dạng quý là Qx'yy nên để dễ dàng chuyển về định dạng theo năm. Ta sẽ quy ước theo tỷ lệ: Q1 là 0, Q2 là 0.25, Q3 là 0.5 và Q4 là 0.75. Còn đối với yy. Ta sẽ quy ước những số có giá trị lớn hơn 24 sẽ là năm 19yy. Còn những năm có giá trị bé hơn bằng 24 ta sẽ quy ước là 20yy.

3.3 Xử lý dữ liệu khuyết

Như đã đề cập bên trên, có nhiều phương pháp để xử lý dữ liệu khuyết và có thể kể đến một số phương pháp như:

- Loại bỏ dữ liệu khuyết. Phương pháp này giúp làm sạch dữ liệu và đơn giản hóa quá trình phân tích. Tuy nhiên, nhược điểm phương pháp này là loại bỏ dữ liệu có thể dẫn đến mất mát thông tin quan trọng và làm biến đổi phân phối của dữ liệu, do đó, cần phải áp dụng cẩn thận và có sự kiểm soát.
- Điền giá trị trung bình. Phương pháp điền giá trị trung bình là một cách tiếp cận phổ biến để xử lý dữ liệu khuyết. Bằng cách này, chúng ta thay thế các giá trị thiếu bằng giá trị trung bình của biến tương ứng. Việc này giúp giảm thiểu sự biến động của dữ liệu và có thể tăng độ tin cậy của kết quả phân tích. Tuy nhiên, phương pháp này có thể dẫn đến sự giảm sút của phương sai, làm mất đi một phần của sự biến động trong dữ liệu gốc. Điều này có thể làm mất đi thông tin quan trọng về sự đa dạng của dữ liệu và làm suy giảm khả năng phát hiện các mẫu đặc biệt.
- Sử dụng mô hình dự đoán. Phương pháp này có ưu điểm hơn so với điền giá trị trung bình là bảo toàn độ biến thiên của dữ liệu. Thay vì thay thế giá trị thiếu bằng một giá trị cố định như giá trị trung bình, các mô hình này sử dụng thông tin từ các biến khác để ước lượng giá trị thích hợp cho các giá trị thiếu. Điều này giúp giữ lại sự đa dạng và biến thiên trong dữ liệu gốc. Ngoài ra, việc sử dụng mô hình dự đoán có thể tăng độ chính xác của dự đoán. Thay vì chỉ dựa vào giá trị trung bình của biến, các mô hình dự đoán có khả năng học từ dữ liệu có sẵn để tạo ra các ước lượng phù hợp hơn. Điều này có thể dẫn đến kết quả dự đoán chính xác hơn và mang lại giá trị cao hơn cho quá trình phân tích. Tuy nhiên, nhược điểm phương pháp này đó là cần lượng lớn dữ liệu để huấn luyện mô hình, độ phức tạp cao hơn điền giá trị trung bình và cần phải chọn mô hình phù hợp để khảo sát.

Trong phần này để xử lý dữ liệu khuyết ta sẽ đề cập đến việc áp dụng 2 phương pháp chính đó là loại bỏ dữ liệu khuyết và điền giá trị trung bình.



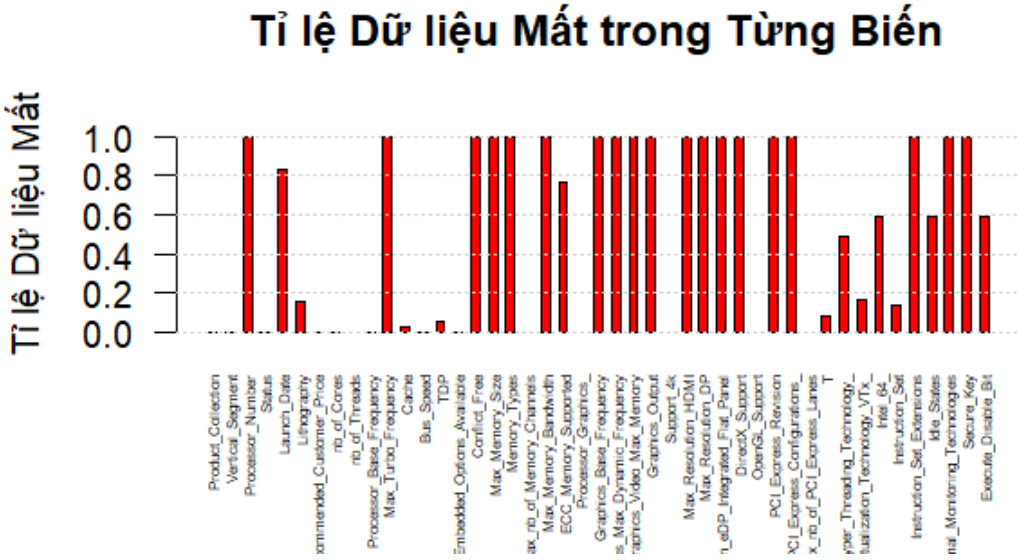
Hình 2: Tỷ lệ dữ liệu khuyết trong dữ liệu

3.3.a Loại bỏ dữ liệu khuyết

Đối với phương pháp này, mục tiêu mà ta cần quan tâm đó là loại bỏ những biến ít cống hiến đối với tập dữ liệu và những biến ít quan trọng để tập trung vào những biến chính trong mô hình.

Về việc loại bỏ những cột không quan trọng, ta sẽ áp dụng những phương pháp như phân tích phương sai hoặc hồi quy tuyến tính nhằm xác định những cột có đóng góp lớn cho cột mà ta đang xét. Trong bài tập này ta sẽ loại bỏ những cột mà ta không quan tâm bằng cách chọn những cột chính mà ta quan tâm và lưu vào df_cpus để xử lý.

Đầu tiên, khi quan sát tập dữ liệu ta nhận thấy rằng, tập dữ liệu mà Processor_Number bỏ trống có tỷ lệ dữ liệu khuyết rất cao và không có ý nghĩa nhiều trong mô hình phân tích vì lấy ra sẽ loại bỏ những hàng bị mất biến này ra khỏi tập dữ liệu.



Hình 3: Tỷ lệ dữ liệu khuyết trong dữ liệu với các biến không có Processor_Number

Tiếp đến, ta sẽ loại bỏ những hàng không có dữ liệu của Recommended_Customer_Price ra khỏi mô hình phân tích. Đối với các biến này, ta sẽ áp dụng những mô hình phân tích để dự đoán dữ liệu của Recommended_Customer_Price.

3.3.b Điền giá trị trung bình

Đối với Lithography, nb_of_Threads, Cache, Launch_Date, Bus_Speed, Max_Memory_Size ta sẽ áp dụng phương pháp xử lý dữ liệu bằng cách điền giá trị trung bình cho dữ liệu. Điều này sẽ giúp làm giảm độ biến động của tập dữ liệu để ta có thể dễ dàng áp dụng những mô hình phân tích dữ liệu để phân tích Recommended_Customer_Price.

IV Thống kê mô tả

1 Thông số tổng quát

Sau khi làm sạch dữ liệu, tiến hành thống kê mô tả cho các biến ngẫu nhiên. Kết quả thu được trả về giá trị nhỏ nhất (Min.), điểm phân vị thứ nhất (1st Qu.), trung vị (Median), giá trị trung bình (Mean), điểm phân vị thứ ba (3rd Qu.) và giá trị lớn nhất (Max.) của từng biến.

```
Product_Collection  Launch_Date   Recommended_Customer_Price  Lithography        Cache
Length:1291         Min.   :2005   Min.   :    2.54           Min.   :14.00   Min.   :    8000
Class :character    1st Qu.:2012   1st Qu.:  161.00           1st Qu.:14.00   1st Qu.: 3000000
Mode  :character    Median :2014   Median :  299.50           Median :22.00   Median : 6000000
                    Mean   :2014   Mean   :  841.62           Mean   :25.32   Mean   :10147924
                    3rd Qu.:2016   3rd Qu.:  685.50           3rd Qu.:32.00   3rd Qu.:12000000
                    Max.   :2018   Max.   :13011.00           Max.   :90.00   Max.   :60000000
nb_of_Threads       Bus_Speed          Max_Memory_Size
Min.   : 1.000   Min.   :0.000e+00   Min.   :   1.0
1st Qu.: 4.000   1st Qu.:5.000e+09   1st Qu.:  32.0
Median : 8.000   Median :5.000e+09   Median :  64.0
Mean   : 9.488   Mean   :5.228e+09   Mean   : 323.1
3rd Qu.: 9.488   3rd Qu.:6.400e+09   3rd Qu.: 323.1
Max.   :56.000   Max.   :9.600e+09   Max.   :4198.4
```

Hình 4: Bảng tóm tắt thông số các biến của tập dữ liệu

Nhận xét: Tổng quan, các biến trong tập dữ liệu được thu thập từ năm 2005 đến năm 2018 hầu hết có phân phối không đồng đều. Điểm tứ phân vị dưới Q1 và điểm tứ phân vị trên Q3 có sự chênh lệch ít nhất gấp 2 lần và nhiều nhất gấp 8 lần .Điều này cho thấy rằng dữ liệu có sự phân tán khá lớn dẫn đến giá trị trung bình có thể không đại diện được tập dữ liệu.

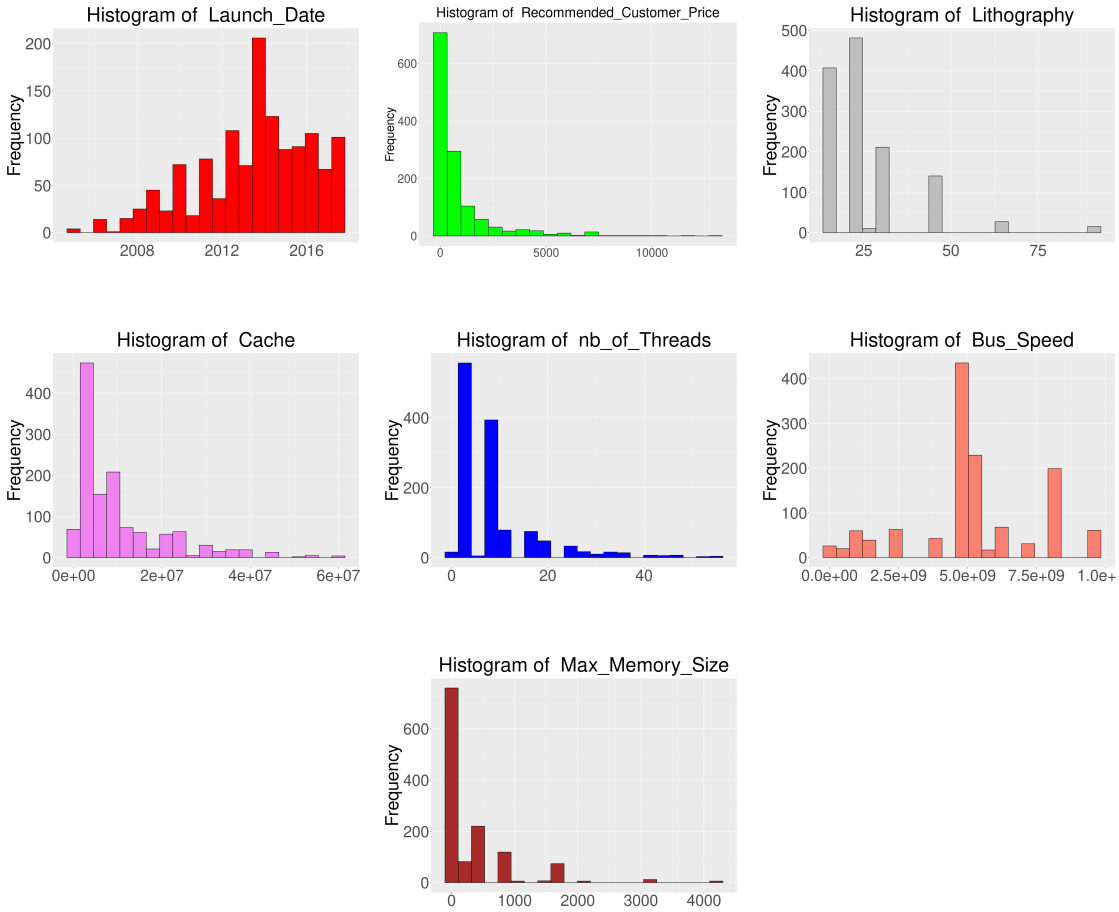
Tuy nhiên, cũng có một số điểm đáng chú ý:

- Biến "Recommended_Customer_Price", "nb_of_Threads" có giá trị trung bình lớn hơn hoặc bằng giá trị điểm phần vị thứ ba nên ta có thể hiểu rằng tập dữ liệu lệch trái mạnh và giá trị trung bình còn bị ảnh hưởng bởi các điểm ngoại lai.
- Hầu hết các biến trong tập dữ liệu có giá trị trung bình lớn hơn trung vị nên ta có thể kết luận đa số các biến tập dữ liệu bị lệch trái. Ngoại trừ "Launch_Date" và "Bus_Speed" có giá trị trung bình bằng trung vị hoặc không chênh lệch nhiêu nên phân phối đối xứng.

Tiếp theo, nhóm tác giả phân tích dữ liệu của biến bằng cách trực quan hóa dữ liệu qua ba dạng biểu đồ là biểu đồ tần suất (histogram), biểu đô phân tán (scatterplot) và biểu đồ hộp (boxplot) cho các biến liên tục. Biểu đồ tần suất sẽ cho thấy cái nhìn tổng quan về phân phối của dữ liệu. Biểu đồ phân tán sẽ cho thấy mối quan hệ giữa các biến độc lập với biến phụ thuộc ("Recommended_Customer_Price"). Biểu đồ hộp giúp biểu diễn rõ ràng các đại lượng quan trọng như giá trị lớn nhất, nhỏ nhất, điểm phân vị,...

2 Biểu đồ tần suất (Histogram)

Biểu đồ tần suất sử dụng để mô tả phân bố của một biến liên tục (continuous).

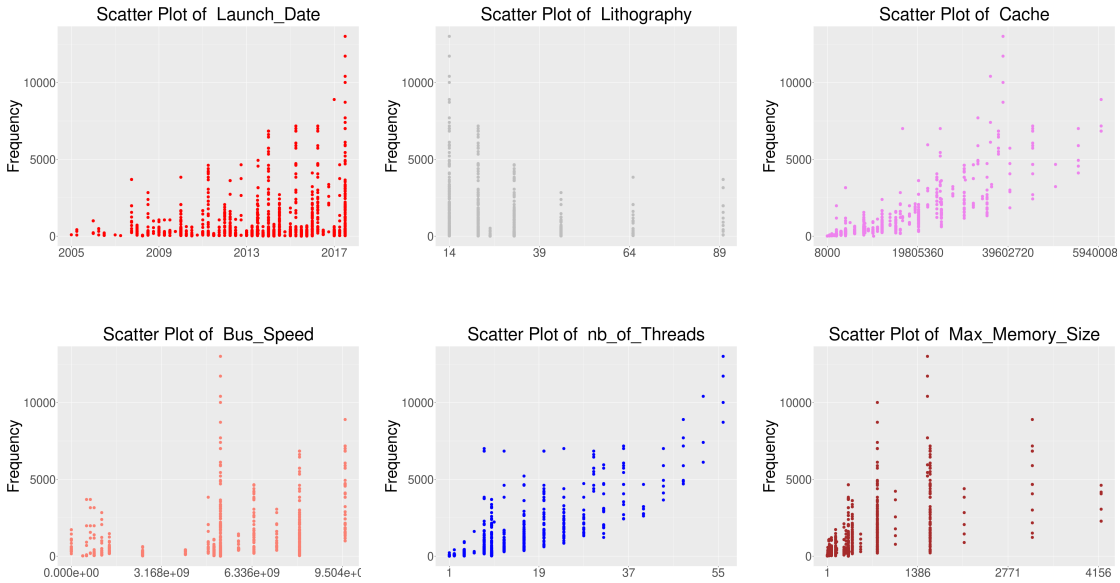


Nhận xét: Dựa trên các biểu đồ tần suất, ta có thể thấy rằng các biểu đồ có phương hướng lệch trái, ngoại trừ "Launch_Date". Vì CPU có các thông số càng cao thì số lượng càng ít và càng về sau, số lượng CPU được sản xuất ra càng nhiều nên lượng dữ liệu để phân tích lớn dần. Một số biến có phân bố dữ liệu bị rời rạc như "Lithography", "nb_of_Threads", "Bus_Speed", "Max_Memory_Size" là do các thông số đó luôn được sản xuất ở các giá trị nhất định.

3 Biểu đồ phân tán (Scatter plot).

Biểu đồ phân tán sử dụng các dấu chấm để thể hiện giá trị (điểm giao nhau) của các biến độc lập và biến phụ thuộc được chọn là "Recommended_Customer_Price". Biểu đồ được sử dụng để quan sát mối tương quan giữa 2 biến định lượng và cung cấp thông tin thống kê chi tiết từng dữ liệu nên không bị ảnh hưởng bởi điểm ngoại lai.

Trục hoành (trục X) mô tả biến độc lập. Trục tung (Y) mô tả biến phụ thuộc.

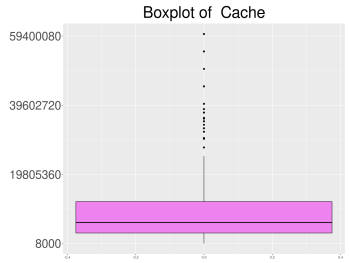


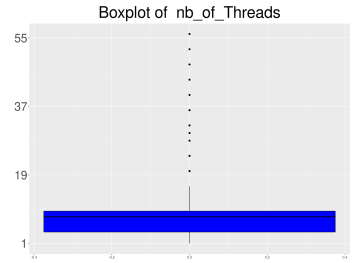
Nhận xét: Dựa vào các biểu đồ phân tán đã phân tích ở trên ta chỉ có thể vẽ đường hồi quy cho các biến "Cache", "nb_of_Threads". Từ đó, ta nhận thấy rằng, biến "Recommended_Customer_Price" có mối quan hệ tuyến tính mạnh với các biến này. Ta có thể kết luận rằng mối quan hệ giữa các biến này là mối quan 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.

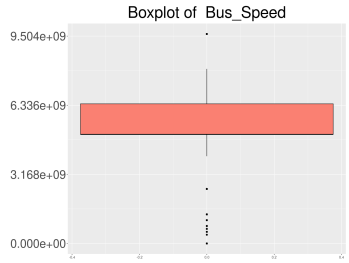
4 Biểu đồ hộp (boxplot)

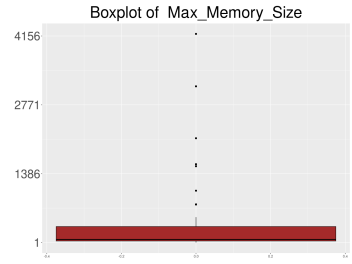
Biểu đồ hộp là biểu đồ dùng để mô tả một biến định lượng theo một cách trực quan và đơn giản. Qua biểu đồ, chúng ta sẽ có thông tin về: Trung vị(Median), điểm tứ phân vị (Quartile), khoảng tứ phân vị (IQR), min, max, điểm ngoại lai (Outliner). Biểu đồ càng đối xứng thì phân bố của biến càng giống phân bố chuẩn.

Nhận xét: Dựa vào biểu đồ trên ta thấy được trung vị của hầu hết các biến đều nằm sát giá trị nhỏ nhất, khoảng tứ phân vị hẹp, giá trị ngoại lai chênh lệch nhiều với trung vị. Ta có









thể kết luận rằng phần lớn biểu đồ lệch trái, giá trị dữ liệu ít phân tán, nhiều giá trị ngoại lai gây ảnh hưởng tới giá trị trung bình của tập dữ liệu. Các biểu đồ này được sử dụng cho thống kê mô tả đề tài trên vì chúng có thể giúp chúng ta hiểu rõ hơn về mối quan hệ giữa các biến số, phân phối tần suất của dữ liệu, và các đặc điểm thống kê cơ bản của dữ liệu.

5 Kết luận

- Launch_Date: Thời điểm ra mắt CPU rơi vào khoảng từ năm 2005 đến năm 2018, và trong những năm từ 2012 đến 2016 có lượng CPU được sản xuất ra lớn nhất với 765 CPU. Giá trị trung bình bằng giá trị trung vị là 2014 chứng tỏ biến "Launch_Date" có hình dáng phân phối lệch đối xứng.

- Recommended_Customer_Price: Giá bán khuyến nghị của CPU dao động trong khoảng từ 2.54$ đến 13011$, tuy nhiên phần lớn CPU có giá bán dao động từ 161$ đến 685.5$, với 652 CPU được đề nghị ở mức giá này. Giá trị trung bình bằng 841,62$ và trung vị là 299.5$. Nhận thấy rằng lẽ ra biến "Recommended_Customer_Price" có hình dáng lệch phải sâu sắc nhưng vì các giá trị ngoại lai từ 2000$ đến 14000$ chiếm tỉ lệ tuy không nhiều nhưng giá trị lớn hơn rất nhiều so với giá tiền trung vị đồng thời trải dài làm ảnh hưởng hình dạng đồ thị thành lệch trái sâu sắc.

- Lithography: Kỹ thuật in thạch bản có kích thước dao động từ 14 (nm) đến 90 (nm), phân bổ nhiều nhất ở phạm vi từ 14 (nm) đến 32 (nm) với hơn 1109 CPU có kích thước của các tính năng trên bóng bán dẫn được báo cáo ở mức này. Giá trị trung bình bằng 25,32 (nm), lớn hơn trung vị là 22 (nm) cho thấy rằng hình dáng phân phối của "Lithography" lẽ ra hơi lệch phải nhưng hình dạng thật lại lệch trái. Nguyên nhân là vì các giá trị từ 50 (nm) đến 90 (nm) chiếm tỉ lệ không nhiều nhưng giá trị gấp bội lần trung vị làm ảnh hưởng đến hình dạng đồ thị.

- Cache: Dung lượng của bộ nhớ cache dao động từ 8000 (byte) đến 6×10^7 (byte), tập trung nhiều nhất ở khoảng 0.3×10^7 (byte) đến 1.2 x 10^7 với 764 CPU có dung lượng ở mức này. Giá trị trung bình là $\approx$ 1 x 10^7 (byte) lớn hơn giá trị trung vị là 0.6×10^7 (byte). Điều này chứng tỏ lẽ ra hình dáng phân phối của biến "Cache" lệch phải nhưng cũng vì lý do phân phối của của giá trị từ 2 x 10^7 đến 6 x 10^7 trải dài và gây ảnh hưởng không ít tới giá trị trung bình làm ảnh hưởng đến hình dạng đồ thị lệch sang trái.

- nb_of_Threads: Số lượng thread nằm trong phạm vi từ 1 (luồng) đến 56 (luồng), tuy nhiên, nhiều nhất là 798 CPU sử dụng từ 4 đến giá trị trung bình là 9.488 (luồng). Giá trị trung bình là 9.488 (luồng) nhỏ hơn giá trị trung vị là 8 (luồng). Điều này chứng tỏ hình dáng phân phối của biến "nb_of_Threads" lệch phải nhưng vì độ trải dài của dữ liệu từ 12 (luồng) đến 56 (luồng) gây ảnh hưởng đến hình dạng của đồ thị, làm đồ thị lệch sang bên trái.

- Bus_Speed: Tốc độ xử lý dữ liệu trong một giây của CPU nằm trong khoảng 0 đến 9.6×10^9 (Hz/s), phân bố nhiều nhất trong khoảng 5×10^9 (Hz/s) đến 6.4 x 10^9 với 721 CPU. Giá trị trung bình là 5.228×10^9 (Hz/s) lớn hơn giá trị trung vị là 5×10^9 (Hz/s) chứng tỏ hình dáng phân phối của biến "Bus_Speed" hơi lệch phải.

- Max_Memory_Size: Dung lượng tối đa của bộ nhớ thay đổi trong phạm vi từ 1 (GB) đến 4198.4 (GB), và tập trung chủ yếu trong phạm vi từ 32 (GB) với 731 CPU. Giá trị trung bình là 323.1 (GB) lớn hơn nhiều với trung vị là 64 (GB). Vì sự chênh lệch giữa các giá trị ngoại lai (từ 500 (GB) tới 4198 (GB)) với giá trị trung vị cực lớn dẫn đến hình dạng đồ thị của biến "Max_Memory_Size" lệch trái sâu sắc thay vì lệch phải sâu sắc.

V Thống kê suy diễn

1 Mục tiêu:

- Phát hiện mối quan hệ giữa các biến.
- Xác định mức độ ảnh hưởng của các biến độc lập đến biến phụ thuộc.
- Xây dựng mô hình dự đoán về giá trị của biến phụ thuộc dựa vào các biến độc lập.

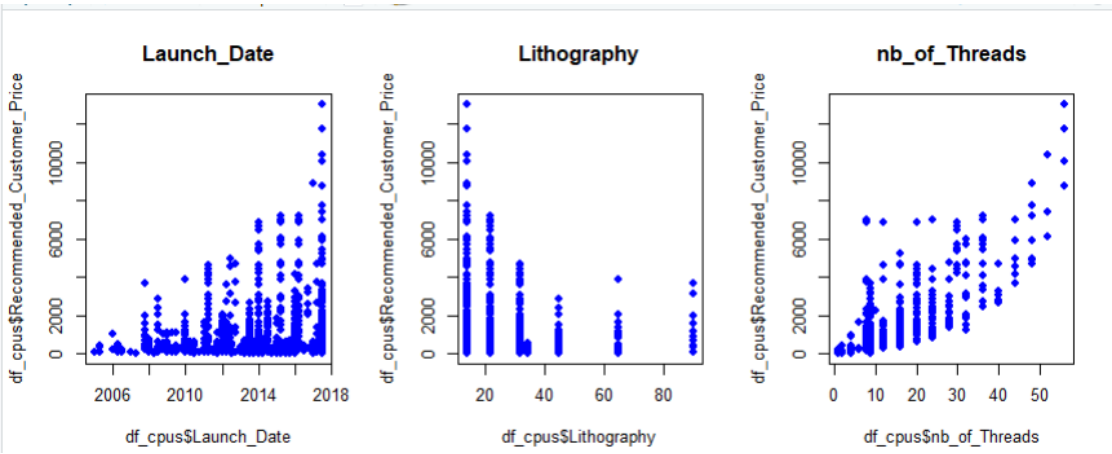
2 Xây dựng mô hình

Mô hình hồi quy bao gồm:

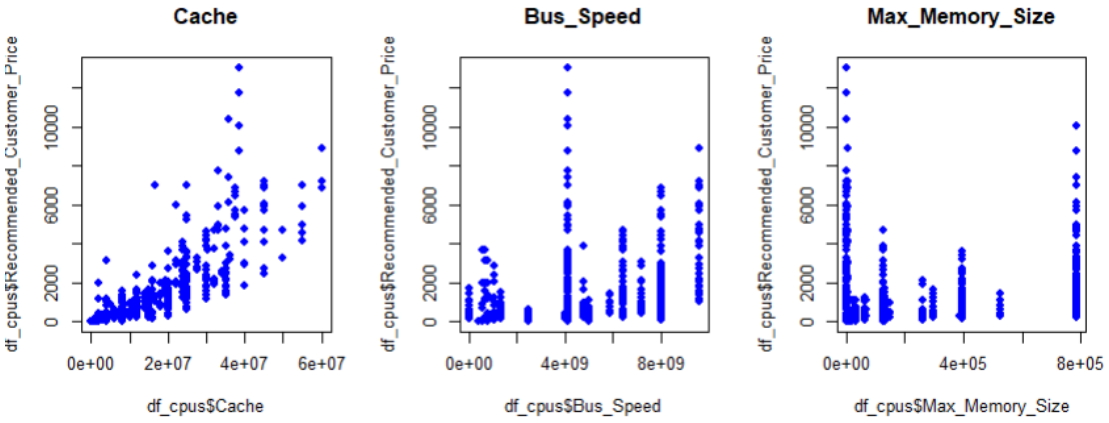
- Biến phụ thuộc: ***Recommended_ Customer_ Price***.
- Biến độc lập: ***Launch_ Date, Lithography, nb_ of_ Threads, Cache, Bus_ Speed, Max_ Memory_ Size***: lần lượt là ngày ra mắt, thạch bản, số luồng, bộ nhớ đệm, tốc độ bus, bộ nhớ tối đa.

Mô hình được biểu diễn như sau:
$Recommended\_Customer\_Price = \beta_0 + \beta_1 \times Launch\_Date + \beta_2 \times Lithography + \beta_3 \times nb\_of\_Threads + \beta_4 \times Cache + \beta_5 \times Bus\_Speed + \beta_6 \times Max\_Memory\_Size + \epsilon$

- Ta lần lượt vẽ đồ thị phân tán của biến phụ thuộc với các biến độc lập để xác định mối quan hệ tuyến tính giữa chúng.



Từ các đồ thị phân tán, ta có thể nhận xét rằng các biến ***Cache*** và ***nb_ of_ Threads*** có mối quan hệ tuyến tính với biến ***Recommeded_ Customer_ Price*** trong khi các biến còn lại thì không.



Ta ước lượng các hệ số $\beta_i, i = 0, ..., 6$ sử dụng lệnh lm():

```
Call:
lm(formula = Recommended_Customer_Price ~ Launch_Date + Lithography +
    nb_of_Threads + Cache + Bus_Speed + Max_Memory_Size, data = df_cpus)

Residuals:
    Min      1Q  Median      3Q     Max
-2746.7  -282.0    27.3   217.9  6940.2

Coefficients:
                 Estimate Std. Error t value Pr(>|t|)
(Intercept)     -1.710e+05  2.906e+04  -5.885 5.06e-09 ***
Launch_Date      8.459e+01  1.441e+01   5.872 5.48e-09 ***
Lithography      1.798e+01  2.994e+00   6.006 2.47e-09 ***
nb_of_Threads    5.735e+01  4.705e+00  12.188  < 2e-16 ***
Cache            8.049e-05  4.262e-06  18.886  < 2e-16 ***
Bus_Speed       -2.696e-08  1.213e-08  -2.222   0.0264 *
Max_Memory_Size -8.460e-04  9.991e-05  -8.468  < 2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 722.1 on 1284 degrees of freedom
Multiple R-squared:  0.7509,    Adjusted R-squared:  0.7498
F-statistic: 645.2 on 6 and 1284 DF,  p-value: < 2.2e-16
```

- Từ kết quả phân tích, ta thu được $\hat{\beta}_0 = -171000, \hat{\beta}_1 = 84.59, \hat{\beta}_2 = 17.98, \hat{\beta}_3 = 57.35, \hat{\beta}_4 = 0.00008049, \hat{\beta}_5 = -0.00000002696, \hat{\beta}_6 = -0.0008460$. Như vậy, đường thẳng hồi quy ước lượng cho bởi phương trình sau:
- $Recommended\_Customer\_Price = -171000 + 84.59 \times Launch\_Date + 17.98 \times Lithography + 57.35 \times nb\_of\_Threads + 0.00008049 \times Cache - 0.00000002696 \times Bus\_Speed - 0.0008460 \times Max\_Memory\_Size$.

- Giá trị ***Adjusted R-squared*** của mô hình là 0.7498. Tức là trong 100% sự biến thiên của Y(***Recommended_ Customer_ Price***) so với trung bình của nó thì có khoảng 74.98% sự biến đổi có thể được giải thích bởi các biến trong mô hình.
- Phân tích kết quả:
  - Trước hết, ta thấy rằng $p - value$ tương ứng với thống kê F bé hơn 2.2×10^{-16}, có ý nghĩa rất cao. Điều này chỉ ra rằng, ít nhất một biến dự báo trong mô hình có ý nghĩa giải thích rất cao cho biến giá cả ***Recommended_ Customer_ Price***.
  - Để xét ảnh hưởng cụ thể của từng biến độc lập, ta xét trọng số (β_i) và p-value tương ứng. Ta nhận thấy rằng, ngoại trừ biến ***Bus_ Speed*** có $p - value$ lớn hơn 0.01 các biến còn lại đều có $p - value$ bé hơn 0.01. Với mức độ tin cậy 99%, ta có thể nói biến ***Bus_ Speed*** không có ý nghĩa đối với mô hình hồi quy bội.
  - Mặt khác, hệ số hồi quy β_i của một biến dự báo cũng có thể được xem như ảnh hưởng trung bình lên biến phụ thuộc doanh thu khi tăng một đơn vị của biến dự báo đó, giả sử rằng các biến dự báo khác không đổi. Cụ thể, $\hat{\beta_3} = 57.35$ thì với mỗi 1 luồng gia tăng trên cpu ta có thể kỳ vọng giá cả của sản phẩm sẽ tăng 57.35 đơn vị về mặt trung bình (giả sử các biến còn lại không đổi). Tương tự với các hệ số hồi quy còn lại.
  - Ngoài ra ta được sai số chuẩn dư (RSE) của mô hình là khoảng 722.1 là có thể chấp nhận được.
  - Vì biến ***Bus_ Speed*** không có ý nghĩa trong mô hình hồi quy, ta tiến hành loại bỏ biến này ra khỏi mô hình và xem xét độ hiệu quả của nó.

3 Hồi quy stepwise

- Từ kết quả trên ta đưa ra hai mô hình thống kê như sau:
- Mô hình thứ nhất gồm các biến ***Launch_ Date, Lithography, nb_ of_ Threads, Cache, Bus_ Speed, và Max_ Memory_ Size.***

```
Start:  AIC=17002.15
Recommended_Customer_Price ~ Launch_Date + Lithography + nb_of_Threads +
    Cache + Bus_Speed + Max_Memory_Size

                 Df Sum of Sq       RSS   AIC
<none>                         669527111 17002
- Bus_Speed       1   2575349 672102460 17005
- Launch_Date     1  17979227 687506337 17034
- Lithography     1  18810022 688337132 17036
- Max_Memory_Size 1  37391480 706918590 17070
- nb_of_Threads   1  77458189 746985299 17142
- Cache           1 185983054 855510165 17317
> |
```

- Mô hình thứ hai loại bỏ đi biến *Bus_Speed* vì biến đó không có ý nghĩa trong thống kê:

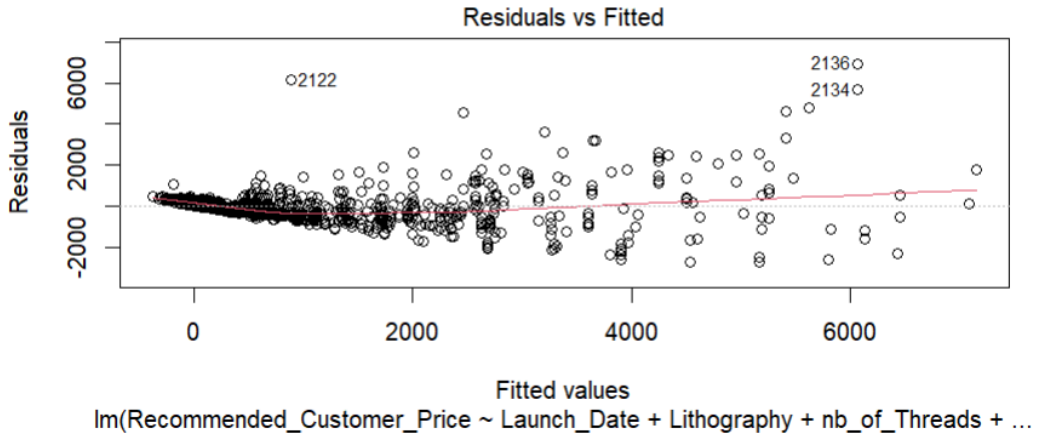
Như chúng ta có thể thấy từ kết quả trên: Khi so sánh hai mô hình, mô hình với giá trị AIC thấp hơn thì ta ưu tiên. Nếu mô hình mới (sau khi loại bỏ một biến) có giá trị AIC thấp hơn mô hình cũ, điều này có thể được coi là một dấu hiệu tích cực về việc loại bỏ biến đó. Như vậy, sau khi loại bỏ biến ***Bus_ Speed***, ta nhận được mô hình có giá trị AIC cao hơn. Vì vậy, ta sẽ thêm biến đó vào lại mô hình mặc dù biến đó không mang ý nghĩa thống kê.

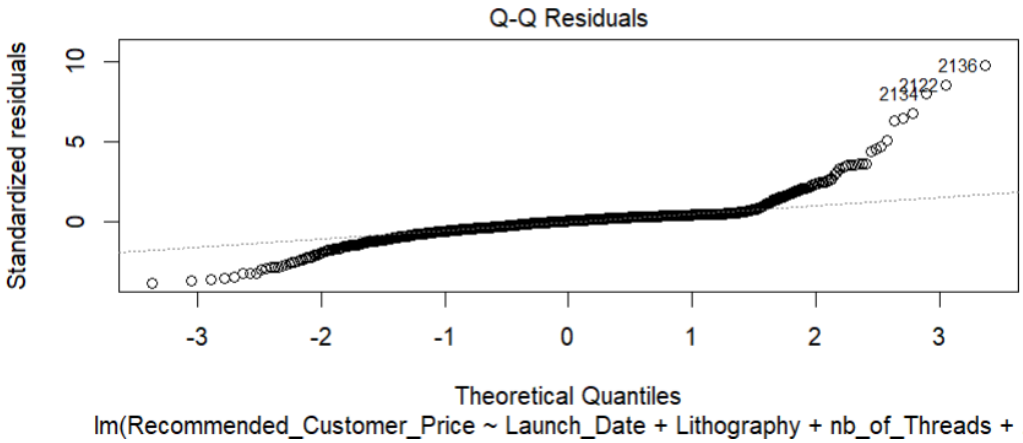
```
Start:  AIC=17005.11
Recommended_Customer_Price ~ Launch_Date + Lithography + nb_of_Threads +
    Cache + Max_Memory_Size

                  Df Sum of Sq       RSS   AIC
<none>                          672102460 17005
- Launch_Date      1  17919388 690021848 17037
- Lithography      1  25780129 697882589 17052
- Max_Memory_Size  1  36969869 709072328 17072
- nb_of_Threads    1  92541562 764644021 17170
- Cache            1 211966612 884069072 17357
```

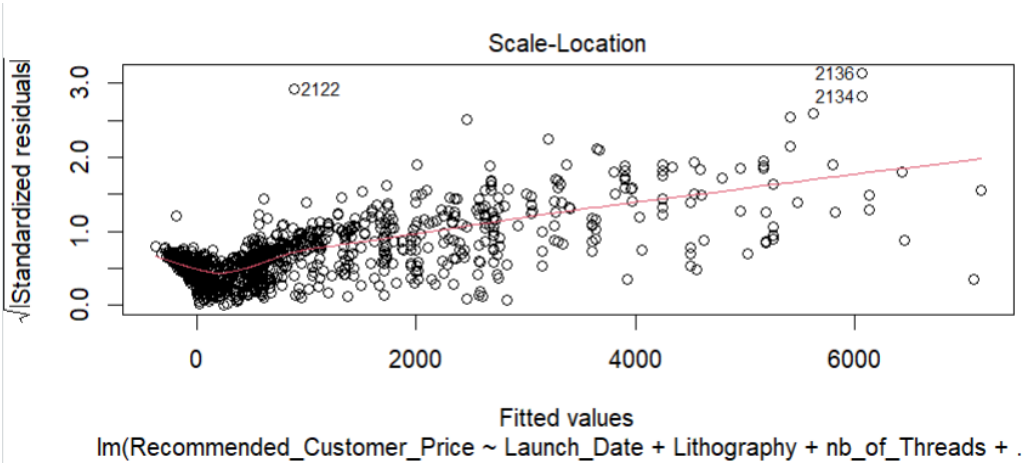
4 Kiểm tra các giả định của mô hình

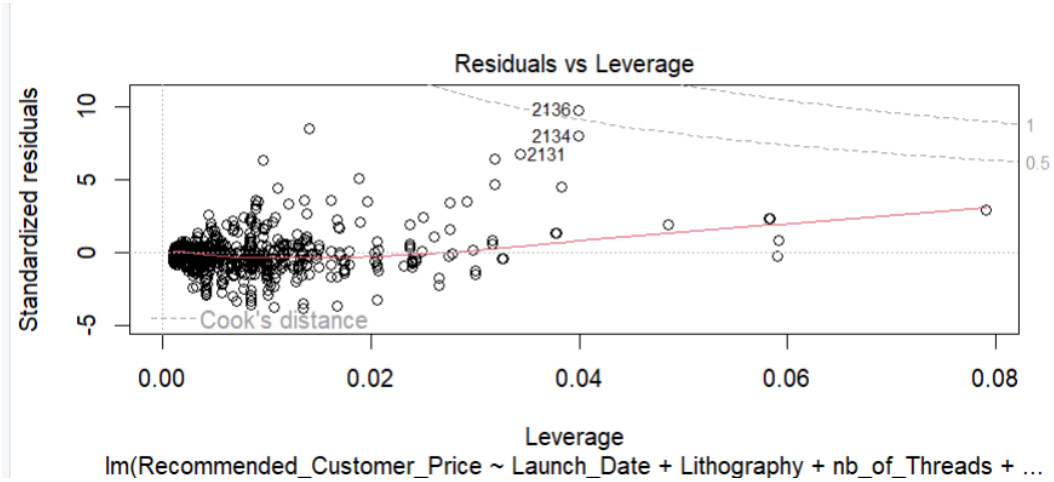
Nhắc lại các giả định của mô hình hồi quy: $Y_i = \beta_0 + \beta_1 X_1 + ... + \beta_n X_n, i = 1,,,n$

1. Tính tuyến tính của dữ liệu.
2. Sai số có phân phối chuẩn.
3. Phương sai của các sai số là hằng số: $\epsilon_i \sim N(0, \sigma^2)$
4. Các sai số ϵ_i độc lập với nhau.




Q-Q Residuals
Standardized residuals
2136
2122
2134
Theoretical Quantiles
lm(Recommended_Customer_Price ~ Launch_Date + Lithography + nb_of_Threads + ...


Scale-Location
√|Standardized residuals|
2122
2136
2134
Fitted values
lm(Recommended_Customer_Price ~ Launch_Date + Lithography + nb_of_Threads + ...

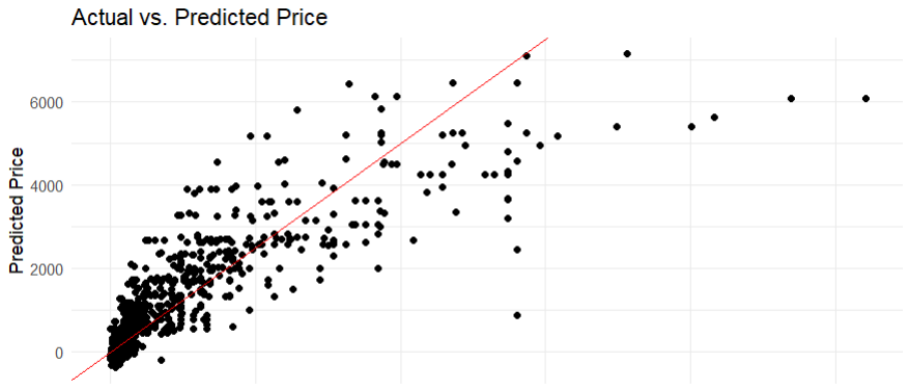

Residuals vs Leverage
Standardized residuals
2136
2134
2131
Cook's distance
Leverage
lm(Recommended_Customer_Price ~ Launch_Date + Lithography + nb_of_Threads + ...

Nhận xét:

- **Residuals vs Fitted:** Từ đồ thị trên ta có thể thấy giả định tuyến tính của mô hình hơi vi phạm. Điều này có thể nhận ra được do mối quan hệ của biến phụ thuộc và các biến độc lập đa số là mối quan hệ phi tuyến.
- **Normal-QQ:** Đồ thị cho thấy giả định sai số có phân phối chuẩn được thỏa mãn.
- **Scale-Location**: Đồ thị cho ta thấy rằng giả định về tính đồng nhất của phương sai cũng hơi bị vi phạm. Tuy nhiên, ta cũng thấy này sự vi phạm này tương đối nhỏ và có thể chấp nhận được.
- **Residuals vs Laverage:** Đồ thị cho ta thấy giá trị quan trắc thứ 2131, 2134 và 2136 có thể là các điểm có có ảnh hưởng cao trong dữ liệu.

5 Dự đoán giá bán của CPU

- Sau khi đã xây dựng mô hình, ta tiến hành thử nghiệm bằng cách tính giá của CPU dựa vào các biến tương ứng trong mô hình. Sau đó so sánh giá cả thực tế so với giá cả dự đoán với tập dữ liệu đã được làm sạch.



Nhận xét:

- Dựa vào đồ thị ta có thể thấy đa số các giá trị quan trắc phân tán xung quanh đường thẳng hồi quy, chứng tỏ rằng giá trị dự đoán và giá trị thực tế có mối quan hệ tuyến tính khá tốt. Có thể kết luận rằng mô hình này có thể dự đoán tốt giá thành của sản phẩm.

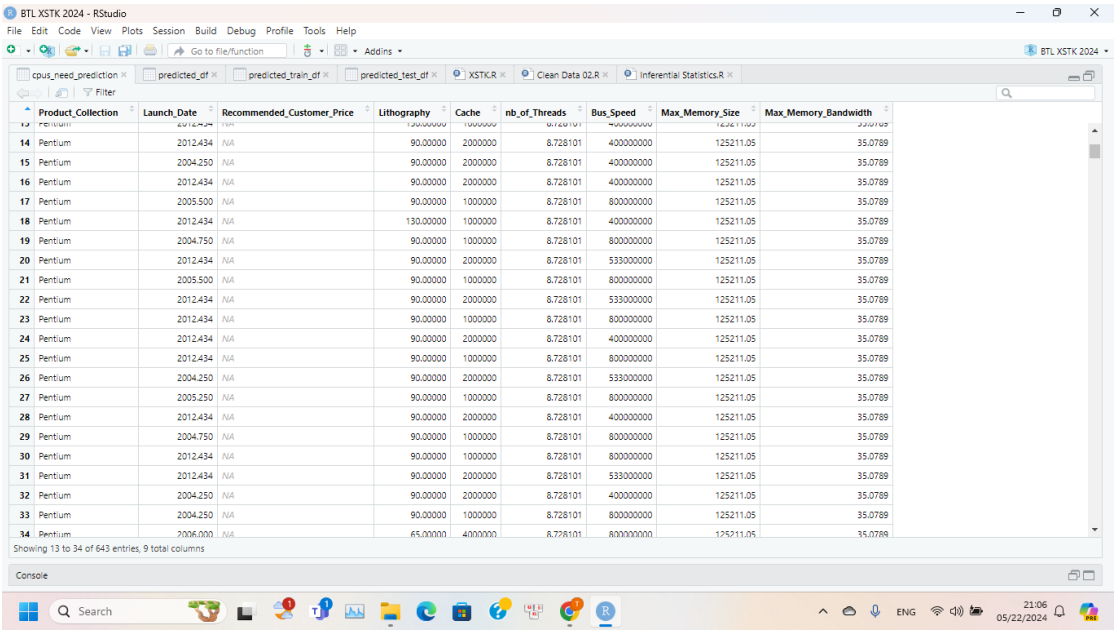
Để xác định mức độ tin cậy của mô hình khi dự đoán ta tính toán giá trị Mean Squared Error (MSE) của mô hình. Giá trị MSE càng nhỏ thì mô hình dự đoán càng chính xác.

```
> range_of_Price <- range(df_cpus$Recommended_Customer_Price)
> print(range_of_Price)
[1]     2.54 13011.00
> MSE = sqrt(mean((predicted_Price - df_cpus$Recommended_Customer_Price)^2))
> print(MSE)
[1] 720.1467
>
```

Nhận xét:

- Ta thấy rằng sai số trung bình bình phương (MSE) là 720.1467, có thể xem rằng đây là một giá trị tương đối nhỏ so với vùng dữ liệu được thống kê [2.54;13011] nên ta có khẳng định rằng mô hình hồi quy tuyến tính được sử dụng tương đối hợp lý và chính xác.

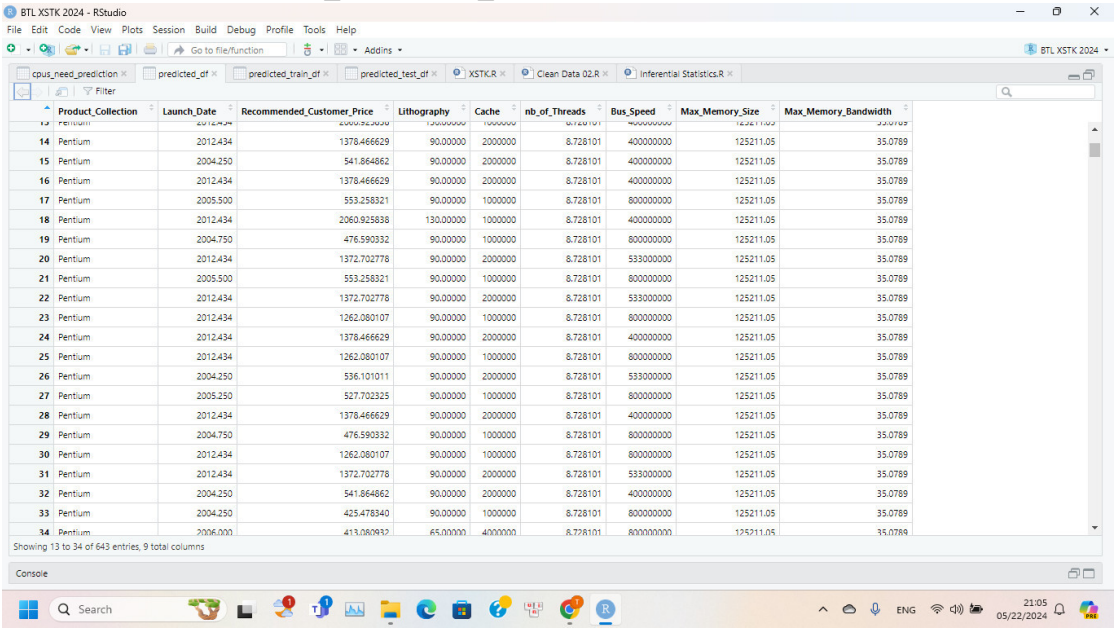
Ta sử dụng mô hình đã được xây dựng để dự đoán giá trị ***Recommended_ Customer_ Price*** bị khuyết trong tập dữ liệu ban đầu
Tập dữ liệu cần dự đoán giá trị ***Recommended_ Customer_ Price***:



	Product_Collection	Launch_Date	Recommended_Customer_Price	Lithography	Cache	nb_of_Threads	Bus_Speed	Max_Memory_Size	Max_Memory_Bandwidth
14	Pentium	2012.434	NA	90.00000	2000000	8.728101	400000000	125211.05	35.0789
15	Pentium	2004.250	NA	90.00000	2000000	8.728101	400000000	125211.05	35.0789
16	Pentium	2012.434	NA	90.00000	2000000	8.728101	400000000	125211.05	35.0789
17	Pentium	2005.500	NA	90.00000	1000000	8.728101	800000000	125211.05	35.0789
18	Pentium	2012.434	NA	130.00000	1000000	8.728101	400000000	125211.05	35.0789
19	Pentium	2004.750	NA	90.00000	1000000	8.728101	800000000	125211.05	35.0789
20	Pentium	2012.434	NA	90.00000	2000000	8.728101	533000000	125211.05	35.0789
21	Pentium	2005.500	NA	90.00000	1000000	8.728101	800000000	125211.05	35.0789
22	Pentium	2012.434	NA	90.00000	2000000	8.728101	533000000	125211.05	35.0789
23	Pentium	2012.434	NA	90.00000	1000000	8.728101	800000000	125211.05	35.0789
24	Pentium	2012.434	NA	90.00000	2000000	8.728101	400000000	125211.05	35.0789
25	Pentium	2012.434	NA	90.00000	1000000	8.728101	800000000	125211.05	35.0789
26	Pentium	2004.250	NA	90.00000	2000000	8.728101	533000000	125211.05	35.0789
27	Pentium	2005.250	NA	90.00000	1000000	8.728101	800000000	125211.05	35.0789
28	Pentium	2012.434	NA	90.00000	2000000	8.728101	400000000	125211.05	35.0789
29	Pentium	2004.750	NA	90.00000	1000000	8.728101	800000000	125211.05	35.0789
30	Pentium	2012.434	NA	90.00000	1000000	8.728101	800000000	125211.05	35.0789
31	Pentium	2012.434	NA	90.00000	2000000	8.728101	533000000	125211.05	35.0789
32	Pentium	2004.250	NA	90.00000	2000000	8.728101	400000000	125211.05	35.0789
33	Pentium	2004.250	NA	90.00000	1000000	8.728101	800000000	125211.05	35.0789
34	Pentium	2006.000	NA	65.00000	4000000	8.728101	800000000	125211.05	35.0789

Hình 13: Tỷ lệ dữ liệu khuyết trong dữ liệu

Các giá trị ***Recommended_ Customer_ Price*** sau khi được dự đoán bởi mô hình:



	Product_Collection	Launch_Date	Recommended_Customer_Price	Lithography	Cache	nb_of_Threads	Bus_Speed	Max_Memory_Size	Max_Memory_Bandwidth
14	Pentium	2012.434	1378.466629	90.00000	2000000	8.728101	400000000	125211.05	35.0789
15	Pentium	2004.250	541.864862	90.00000	2000000	8.728101	400000000	125211.05	35.0789
16	Pentium	2012.434	1378.466629	90.00000	2000000	8.728101	400000000	125211.05	35.0789
17	Pentium	2005.500	553.258321	90.00000	1000000	8.728101	800000000	125211.05	35.0789
18	Pentium	2012.434	2060.925838	130.00000	1000000	8.728101	400000000	125211.05	35.0789
19	Pentium	2004.750	476.590332	90.00000	1000000	8.728101	800000000	125211.05	35.0789
20	Pentium	2012.434	1372.702778	90.00000	2000000	8.728101	533000000	125211.05	35.0789
21	Pentium	2005.500	553.258321	90.00000	1000000	8.728101	800000000	125211.05	35.0789
22	Pentium	2012.434	1372.702778	90.00000	2000000	8.728101	533000000	125211.05	35.0789
23	Pentium	2012.434	1262.080107	90.00000	1000000	8.728101	800000000	125211.05	35.0789
24	Pentium	2012.434	1378.466629	90.00000	2000000	8.728101	400000000	125211.05	35.0789
25	Pentium	2012.434	1262.080107	90.00000	1000000	8.728101	800000000	125211.05	35.0789
26	Pentium	2004.250	536.101011	90.00000	2000000	8.728101	533000000	125211.05	35.0789
27	Pentium	2005.250	527.702325	90.00000	1000000	8.728101	800000000	125211.05	35.0789
28	Pentium	2012.434	1378.466629	90.00000	2000000	8.728101	400000000	125211.05	35.0789
29	Pentium	2004.750	476.590332	90.00000	1000000	8.728101	800000000	125211.05	35.0789
30	Pentium	2012.434	1262.080107	90.00000	1000000	8.728101	800000000	125211.05	35.0789
31	Pentium	2012.434	1372.702778	90.00000	2000000	8.728101	533000000	125211.05	35.0789
32	Pentium	2004.250	541.864862	90.00000	2000000	8.728101	400000000	125211.05	35.0789
33	Pentium	2004.250	425.478340	90.00000	1000000	8.728101	800000000	125211.05	35.0789
34	Pentium	2006.000	413.060932	65.00000	4000000	8.728101	800000000	125211.05	35.0789

Hình 14: Tỷ lệ dữ liệu khuyết trong dữ liệu

VI Thảo luận và mở rộng

CPU là viết tắt của Central Processing Unit, là bộ xử lý trung tâm của máy tính, còn được gọi là bộ xử lý, bộ xử lý trung tâm, hoặc bộ vi xử lý. CPU xử lý tất cả các lệnh mà nó nhận được từ phần cứng và phần mềm chạy trên máy tính. Hiệu suất và giá thành của CPU phụ thuộc vào một số yếu tố như:

- Số lõi (nhân): Số lõi trong CPU quyết định khả năng xử lý đa nhiệm. Các CPU đa lõi có thể thực hiện nhiều tác vụ cùng một lúc, tăng hiệu suất.
- Tốc độ xung nhịp: Tần số hoạt động của CPU ảnh hưởng đến tốc độ xử lý. Một chu kỳ xung nhịp tương đương với 1Hz.
- Độ rộng địa chỉ bus và data bus: Độ rộng của bus dữ liệu và bus địa chỉ quyết định khả năng truyền tải dữ liệu giữa CPU và các thành phần khác.
- Siêu phân luồng: Các CPU hỗ trợ siêu phân luồng có thể xử lý nhiều luồng công việc cùng một lúc.
- Nhiệt độ: Nhiệt độ ảnh hưởng đến hiệu suất và tuổi thọ của CPU.
- Băng thông: Băng thông của bus dữ liệu quyết định khả năng truyền tải dữ liệu giữa CPU và RAM.

1 Thảo luận

1.1 Ưu điểm:

- Dễ hiểu và triển khai: Mô hình hồi quy tuyến tính là một mô hình đơn giản và dễ hiểu. Nó thường được sử dụng như là một bước đầu tiên trong quá trình mô hình hóa do tính đơn giản và khả năng giải thích cao.
- Dễ dàng điều chỉnh và mở rộng : Có thể dễ dàng điều chỉnh mô hình bằng cách thêm hoặc loại bỏ các biến độc lập. Ngoài ra, mô hình hồi quy tuyến tính có thể được mở rộng để bao gồm các biến tương tác và các biến động.

1.2 Hạn chế:

Hồi quy tuyến tính nhạy cảm với các điểm dữ liệu ngoại lệ

2 Mở rộng: Mô hình hồi quy Ridge và Lasso

2.1 Hiện tượng quá khớp (Overfitting)

Quá khớp (Overfitting) xảy ra khi mô hình thống kê khớp chuẩn xác với bộ dữ liệu huấn luyện. Điều này khiến cho giải thuật không thể biểu diễn chính xác trên dữ liệu mới. Sự tổng quát hóa của mô hình đối với dữ liệu mới giúp chúng ta sử dụng được giải thuật học máy (machine learning algorithms) để dự đoán và phân loại dữ liệu.

Khi một giải thuật học máy được tạo nên, một bộ dữ liệu mẫu (training data) sẽ được sử dụng để huấn luyện mô hình. Tuy nhiên khi một mô hình được huấn luyện quá lâu với bộ dữ liệu mẫu hoặc khi mô hình quá phức tạp, mô hình bắt đầu thích nghi với dữ liệu nhiễu, những biến không ảnh hưởng đến kết quả của mô hình hay phân tích dự đoán hoặc phân loại biến. Điều này dẫn tới việc mô hình không đủ tổng

quát đối với những dữ liệu mới thì mô hình sẽ không thể thực hiện được các tác vụ phân loại hay dự đoán chính xác.

2.2 Cách để giải quyết hiện tượng quá khớp (Overfitting)

Trong thống kê và máy học, "regularization" là một kỹ thuật được sử dụng để kiểm soát và giảm thiểu quá mức phức tạp của mô hình, ngăn chặn hiện tượng quá khớp overfitting. Mục tiêu của regularization là tối ưu hóa hiệu suất dự đoán của mô hình trên dữ liệu mới bằng cách kiểm soát các tham số của mô hình. Có hai kỹ thuật regularization chính: L1 regularization và L2 regularization

2.3 Hồi quy Ridge - L2 regularization

Hồi quy Ridge là một sửa đổi của hồi quy bình phương tối thiểu để làm cho nó phù hợp hơn cho việc lựa chọn biến. Trong hồi quy Ridge, chúng ta không chỉ cố gắng giảm thiểu tổng bình phương của phần dư mà còn một thành phần khác bằng tổng bình phương của các tham số hồi quy nhân với một tham số điều chỉnh. Nói cách khác, trong hồi quy Ridge, chúng ta cố gắng giảm thiểu lượng dưới đây:

$$\sum_{i=1}^{n}(y_i - \beta_0 - \beta_i X_i)^2 + \lambda \sum_{i=1}^{n}(\beta_i)^2$$

Trong phương trình trên, giá trị $\lambda \geq 0$. $\sum_{i=1}^{n}(y_i - \beta_0 - \beta_i X_i)^2$ chính là tổng bình phương phần dư và $\lambda \sum_{i=1}^{n}(\beta_i)^2$ là thành phần điều chuẩn

- Trường hợp $\lambda = 0$, thành phần điều chuẩn bị tiêu giảm và chúng ta quay trở về bài toán hồi qui tuyến tính.
- Trường hợp λ nhỏ thì vai trò của thành phần điều chuẩn trở nên ít quan trọng. Mức độ kiểm soát quá khớp của mô hình sẽ trở nên kém hơn.
- Trường hợp λ lớn chúng ta muốn gia tăng mức độ kiểm soát lên độ lớn của các hệ số ước lượng và qua đó giảm bớt hiện tượng quá khớp

Khi tăng dần hệ số λ thì hồi qui Ridge sẽ có xu hướng thu hẹp hệ số ước lượng từ mô hình.

2.4 Hồi quy Lasso - L1 regularization

Trong hồi qui Lasso, thay vì sử dụng thành phần điều chuẩn là chuẩn bậc hai thì chúng ta sử dụng chuẩn bậc 1.

$$\sum_{i=1}^{n}(y_i - \beta_0 - \beta_i X_i)^2 + \lambda \sum_{i=1}^{n}|\beta_i|$$

Khi tiến hành hồi qui mô hình Lasso trên một bộ dữ liệu mà có các biến đầu vào đa cộng tuyến (multicollinear) thì mô hình hồi qui Lasso sẽ có xu hướng lựa chọn ra một biến trong nhóm các biến đa cộng tuyến và bỏ qua những biến còn lại. Trong khi ở mô hình hồi qui tuyến tính thông thường và hồi qui Ridge thì có xu hướng sử dụng tất cả các biến đầu vào.

VII Nguồn dữ liệu và code

- **Nguồn dữ liệu:** Link nguồn dữ liệu
- **Source code:** Link source code

VIII Tài liệu tham khảo

References

[1] Steven P. Sanderson II, MPH. A Complete Guide to Stepwise Regression in R, 2023.

[2] Datacamp article. Linear Regression in R Tutorial, 2022

[3] Peter Dalgaard Introductory Statistics with R, second edition, 2008

[4] Applied Statistics with R Applied Statistics with R, 2020

[5] Nguyễn Đình Huy (Chủ biên), Giáo trình xác suất và thống kê, 2019

[6] Nguyen T Handling missing data, ranalytics.vn, Truy cập ngày 04/05/2024

[7] Hoàng Văn Hà Xử lý dữ liệu khuyết với phân tích thành phần chính, Truy cập ngày 30/04/2024

[8] dhruv5819 Data visualization with R and ggplot2, Truy cập ngày 20/12/2023

[9] Machine Learning: Hồi quy Ridge và hồi quy Lasso, Truy cập ngày 10/05/2024