TRƯỜNG ĐẠI HỌC BÁCH KHOA
KHOA CÔNG NGHỆ THÔNG TIN

# TIỂU LUẬN CUỐI KỲ

**HỌC PHẦN:** KHOA HỌC DỮ LIỆU

**TÊN ĐỀ TÀI**: DỰ ĐOÁN GIÁ LAPTOP

| **NHÓM** | 7 |
|---|---|
| **HỌ VÀ TÊN SINH VIÊN** | **LỚP HỌC PHẦN** |
| Cao Kiều Văn Mạnh | 20.15 |
| Lê Nguyễn Ngọc Lâm | |
| Trần Đình Minh Khoa | |

ĐÀ NẴNG, 06/2023

# TÓM TẮT

Hiện nay, laptop (máy tính xách tay) dần trở thành một thiết bị hết sức phổ biến và cần thiết với mọi người nhờ sự tiện dụng, nhỏ gọn của nó. Nhu cầu mua sắm và sử dụng laptop vì thế mà ngày càng gia tăng. Do đó, với tiểu luận này, nhóm chúng em quyết định thực hiện đề tài **"Dự đoán giá laptop"**. Nhóm đã tiến hành thu thập dữ liệu gồm các thuộc tính và giá của các mẫu laptop từ trang **LaptopMedia** (một website chứa thông tin của hơn 300 000 mẫu laptop). Sau đó, nhóm tiếp tục thực hiện các bước làm sạch, chuẩn hóa dữ liệu, lựa chọn các thuộc tính quan trọng ảnh hưởng nhiều đến giá laptop. Từ đó, nhóm chúng em đã xây dựng được chương trình để đưa ra dự đoán về giá cả của laptop. Chương trình sử dụng hai mô hình **Linear Regression** và **Support Vector Regression** để dự đoán giá của laptop và kết quả của mô hình được so sánh bằng các metrics **MAE**, **RMSE** và **MAPE**.

# BẢNG PHÂN CÔNG NHIỆM VỤ

| Sinh viên thực hiện | Các nhiệm vụ | Tự đánh giá |
|---|---|---|
| Cao Kiều Văn Mạnh | Mô tả, trực quan hóa dữ liệu | Đã hoàn thành |
| | Làm sạch và chuẩn hóa dữ liệu | Đã hoàn thành |
| | Lựa chọn đặc trưng | Đã hoàn thành |
| | Tổng hợp, viết báo cáo | Đã hoàn thành |
| Lê Nguyễn Ngọc Lâm | Làm sạch và chuẩn hóa dữ liệu | Đã hoàn thành |
| | Xây dựng và huấn luyện mô hình | Đã hoàn thành |
| | Tổng hợp, viết báo cáo | Đã hoàn thành |
| Trần Đình Minh Khoa | Thu thập dữ liệu | Đã hoàn thành |
| | Đánh giá, so sánh hiệu quả hai mô hình | Đã hoàn thành |
| | Trực quan hóa kết quả của quá trình | Đã hoàn thành |
| | Tổng hợp, viết báo cáo | Đã hoàn thành |

# MỤC LỤC

# DANH SÁCH BẢNG

# DANH SÁCH HÌNH ẢNH

# 1. Giới thiệu

Ngày nay, cùng với sự phát triển của khoa học và công nghệ, laptop trở thành một công cụ thiết yếu cho nhu cầu học tập, làm việc cho đến giải trí. Điều này khiến cho thị trường laptop ngày càng phát triển và việc lựa chọn một chiếc laptop sao cho phù hợp với nhu cầu và túi tiền không phải là điều dễ dàng đối với nhiều người.

Giải pháp nhóm đưa ra là sử dụng các công cụ như **Selenium, BeautifulSoup** để hỗ trợ cào dữ liệu về giá bán laptop, sau đó xây dựng các mô hình hồi quy tuyến tính nhằm dự đoán giá laptop kết hợp với các kĩ thuật xử lý dữ liệu trống, dữ liệu ngoại lệ và chuẩn hóa.

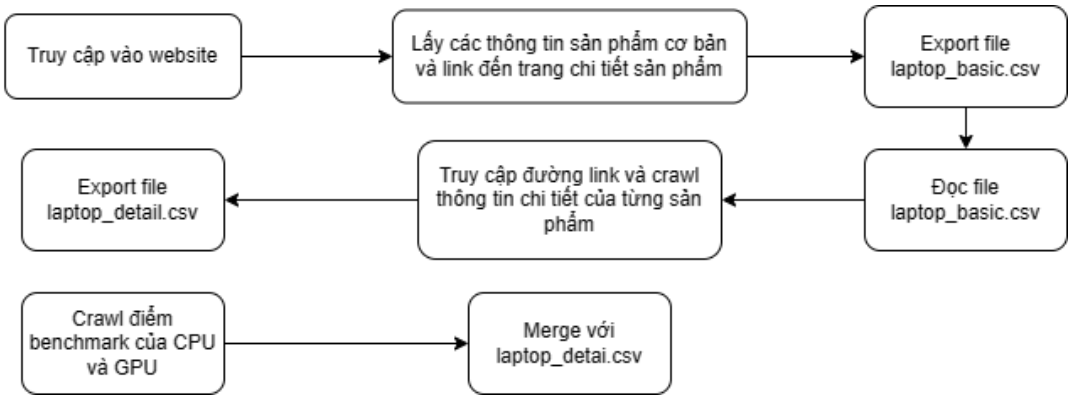
# 2. Thu thập và mô tả dữ liệu

## 2.1. Thu thập dữ liệu

Nhóm đã tiến hành thu thập dữ liệu từ website **LaptopMedia**, một website cung cấp thông tin của hơn 300.000 mẫu laptop. Đường link dẫn tới website là: https://laptopmedia.com

Quá trình thu thập dữ liệu có thể tổng quát như sau:

- **Đầu vào:** URL của website muốn cào dữ liệu.
- **Kết quả:** Một file dữ liệu thô chứa thông tin chi tiết của từng laptop.

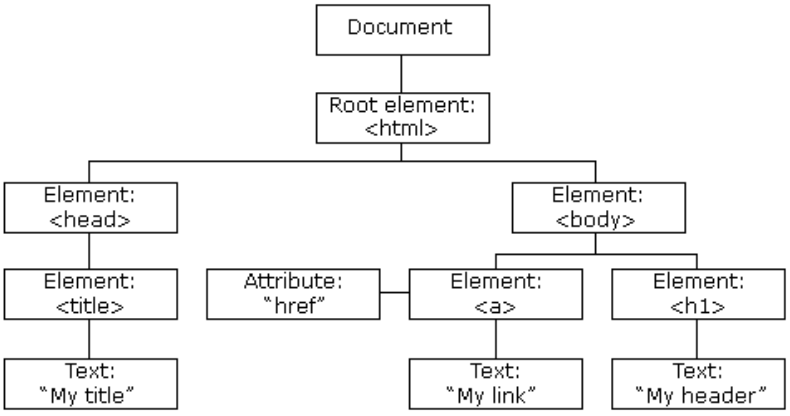



*Hình 1. Các bước cào dữ liệu*

Dựa trên quá trình phân tích về cấu trúc và đặc trưng của website LaptopMedia, nhóm lựa chọn các công cụ sau để tiến hành cào dữ liệu: ngôn ngữ Python và các thư viện hỗ trợ bao gồm Selenium, Requests và BeatifulSoup.

- **Selenium:** Selenium là một công cụ tự động hóa giúp giả lập các thao tác của người dùng trên trình duyệt. Nhóm sử dụng Selenium để tiến hành mở website và đợi cho toàn bộ thông tin được tải lên đầy đủ trước khi tiến hành quá trình cào dữ liệu.
- **Requests:** Thư viện của Python, giúp lập trình gửi nhận HTTP request. Thư viện này giúp lấy source text HTML của các trang website.

- **Beautiful Soup:** Thư viện của Python, giúp phân tích source text HTML lấy được từ hai công cụ ở trên và phân tích thành cấu trúc cây HTML như bên dưới.

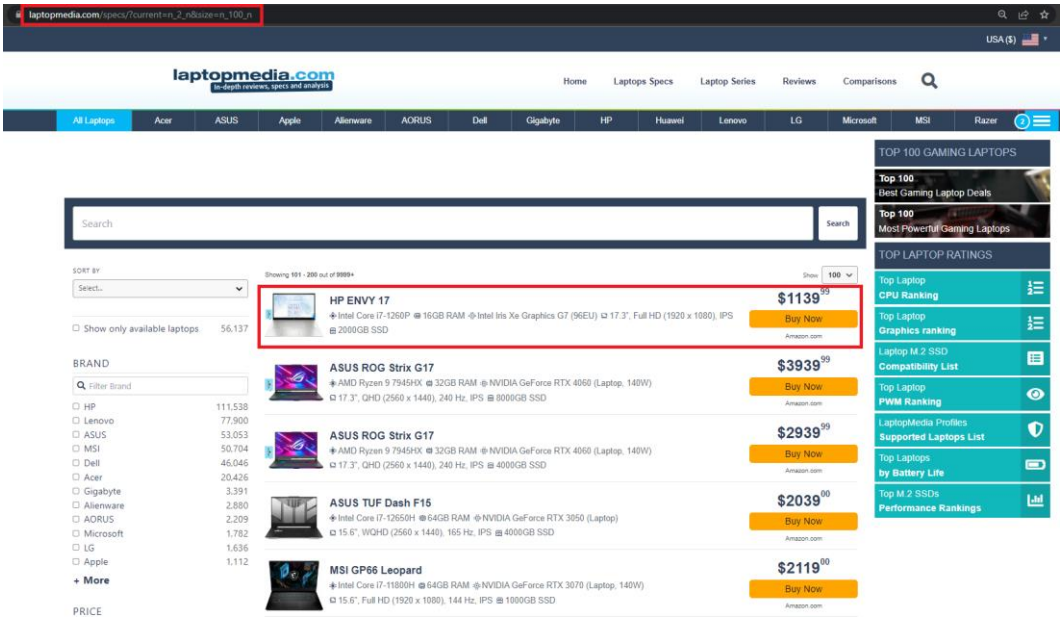



*Hình 2. Cấu trúc của một cây HTML*

Tổng quan về quá trình cào dữ liệu từ trang LaptopMedia:

**Bước 1:** Tiến hành thu thập toàn bộ đường dẫn đến trang thông tin chi tiết của từng sản phẩm:

- Sử dụng Selenium, truy cập các trang https://laptopmedia.com/specs/?current=n_{page_index}_n&size=n_100_n, với page_index chạy từ 1-100, là số chỉ của trang sẽ cào.

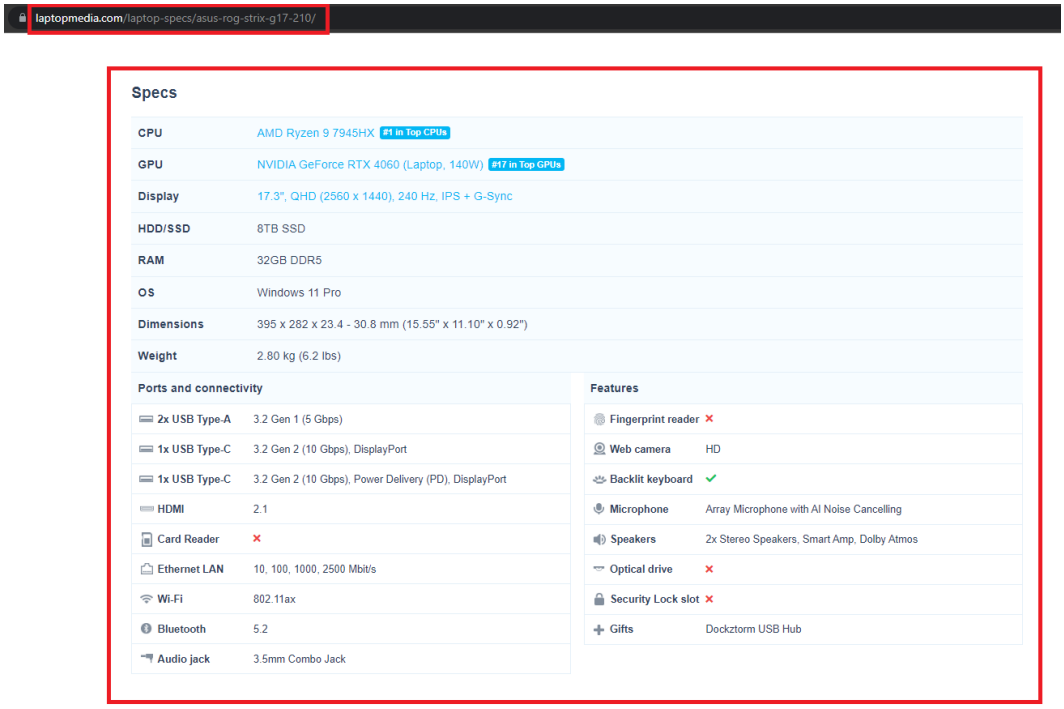



*Hình 3. Trang thông tin website LaptopMedia*

- Trong các trang này ta có thể thu thập các thông tin cơ bản của sản phẩm như là CPU, RAM, GPU, độ phân giải màn hình, bộ nhớ trong, giá bán, ... cùng với link đến trang thông tin chi tiết của sản phẩm.

**Bước 2:** Lấy thông tin chi tiết của sản phẩm:

- Vì số lượng sản phẩm rất lớn và cần phải truy cập vào link chi tiết của từng sản phẩm, phần lớn thời gian tiêu tốn dùng để chờ trang load xong, do đó nhóm đặt thời gian này bằng khoảng gấp đôi thời gian cần thiết thực tế để tránh xảy ra lỗi, thiếu dữ liệu.
- Áp dụng multithreading vào crawl, với mỗi thread sẽ điều khiển một cửa sổ Chrome, phương pháp này giúp giảm thời gian crawl đi 5-10 lần, tùy thuộc vào số thread chạy. Điểm yếu của phương pháp này là phải chọn số thread phù hợp với cấu hình máy để tránh tràn RAM, đồng thời tối đa tốc độ crawl.
- Bên trong trang thông tin chi tiết của sản phẩm, ta có thể thu thập được thêm nhiều thông tin và tính năng các sản phẩm có ảnh hưởng đến giá, như là khối lượng, màn hình, ram, tính năng,...




*Hình 4. Trang thông tin chi tiết của một laptop*

**Bước 3:** Lấy thông tin về CPU và GPU tại đường dẫn sau:

- CPU: https://laptopmedia.com/top-laptop-cpu-ranking
- GPU: https://laptopmedia.com/top-laptop-graphics-ranking
- Từ các trang này có thể thu thập điểm hiệu năng của CPU và GPU, giúp dễ dàng đánh giá hiệu năng của chúng.

| # | CPU | Cinebench 23 |
|---|---|---|
| 1. | AMD Ryzen 9 7945HX | 34357 |
| 2. | Intel Core i9-13950HX | 32485 |
| 3. | Intel Core i9-13980HX | 30647 |
| 4. | Intel Core i9-13900HX | 27070 |
| 5. | Intel Core i9-12950HX | 21007 |
| 6. | Intel Core i7-13700HX | 20213 |
| 7. | Intel Core i7-12800HX | 19686 |
| 8. | Intel Core i9-12900HX | 19659 |
| 9. | Intel Core i9-13900H | 19272 |

| # | GPU | 3DMark Time Spy (G) |
|---|---|---|
| 1. | NVIDIA GeForce RTX 4090 (Laptop, 175W) | 21036 |
| 2. | NVIDIA GeForce RTX 4080 (Laptop, 175W) | 19414 |
| 3. | NVIDIA GeForce RTX 4090 (Laptop, 150W) | 19147 |
| 4. | NVIDIA GeForce RTX 3080 Ti (Laptop, 175W) | 13304 |
| 5. | NVIDIA GeForce RTX 3080 Ti (Laptop, 150W) | 12769 |
| 6. | NVIDIA GeForce RTX 3080 (Laptop, 165W) | 12427 |
| 7. | NVIDIA GeForce RTX 3080 (Laptop, 150W) | 12163 |
| 8. | NVIDIA GeForce RTX 4070 (Laptop, 105W) | 12022 |
| 9. | NVIDIA GeForce RTX 4070 (Laptop, 140W) | 11933 |

*Hình 5. Trang thông tin hiệu năng của CPU và GPU*

### 2.2. Làm sạch dữ liệu thô

Dữ liệu sau khi thu thập từ nhiều nguồn khác nhau có thể lẫn những mẫu dữ liệu không đảm bảo chất lượng, vì vậy trước khi thực hiện các bước xử lý dữ liệu cần thiết phải loại bỏ những mẫu không cần thiết.

Sau khi tiến hành cào, nhóm thu được hai tập dữ liệu thô với kích thước như sau:

- Small dataset: 1000 dòng và 42 cột
- Big dataset: 10000 dòng và 97 cột.

Nhóm tiến hành làm sạch với quy trình sau:

- Loại bỏ những cột không chứa thông tin cần thiết.
- Xóa các cột thông tin chứa không quá 20% dữ liệu là not null, xóa các dòng không chứa thông tin trong nhiều hơn 3 cột.
- Chỉnh sửa thông tin trong các cột dữ liệu (định dạng, kiểu dữ liệu, thông tin). Tách các cột dữ liệu chứa các thông tin tổng hợp thành các cột mới.
- Xóa dữ liệu sai thực tế, dữ liệu trùng lặp, xóa dữ liệu trống trong cột price.

### 2.3. Mô tả dữ liệu

Sau khi tiến hành quy trình làm sạch dữ liệu thô, nhóm thu được hai tập dữ liệu sạch với thông tin như sau:

- Small dataset: 947 dòng và 20 cột
- Big dataset: 8636 dòng và 20 cột

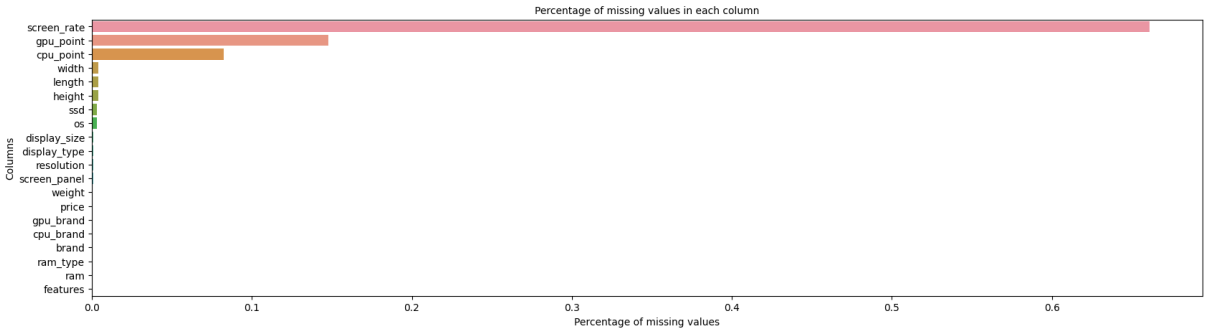
Cả hai tập dữ liệu có 20 cột thông tin như nhau, cụ thể:

| STT | Tên cột | Ý nghĩa | Kiểu dữ liệu | Số mẫu dữ liệu trống | |
|---|---|---|---|---|---|
| | | | | Small dataset | Bigdataset |
| 1 | price | Giá bán của laptop (USD) | float64 | 0 | 0 |
| 2 | brand | Hãng sản xuất | object | 0 | 0 |
| 3 | ram | Dung lượng RAM (GB) | float64 | 0 | 0 |
| 4 | ram_type | Loại RAM | object | 0 | 1 |
| 5 | display_size | Kích thước màn hình (inch) | float64 | 1 | 14 |
| 6 | display_type | Loại màn hình (HD/FHD,..) | object | 1 | 15 |
| 7 | resolution | Độ phân giải | object | 1 | 14 |
| 8 | screen_rate | Tần số quét (Hz) | float64 | 626 | 5969 |
| 9 | screen_panel | Tấm nền màn hình | object | 1 | 14 |
| 10 | lenght | Chiều dài (mm) | float64 | 4 | 44 |
| 11 | witdth | Chiều rộng (mm) | float64 | 4 | 48 |
| 12 | height | Độ dày (mm) | float64 | 4 | 48 |
| 13 | cpu_brand | Hãng sản xuất CPU | object | 0 | 0 |

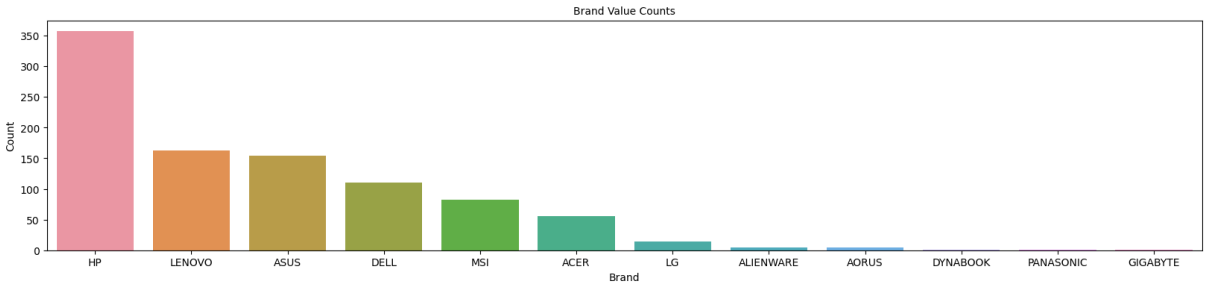
| | | | | | |
|---|---|---|---|---|---|
| 14 | gpu_brand | Hãng sản xuất GPU | object | 0 | 0 |
| 15 | ssd | Dung lượng SSD (GB) | float64 | 3 | 57 |
| 16 | os | Hệ điều hành | object | 3 | 19 |
| 17 | weight | Cân nặng (Kg) | float64 | 0 | 0 |
| 18 | cpu_point | Điểm hiệu năng CPU | float64 | 78 | 736 |
| 19 | gpu_point | Điểm hiệu năng GPU | float64 | 140 | 1174 |
| 20 | features | Tính năng | object | 0 | 0 |

*Bảng 1. Thông tin về dữ liệu*

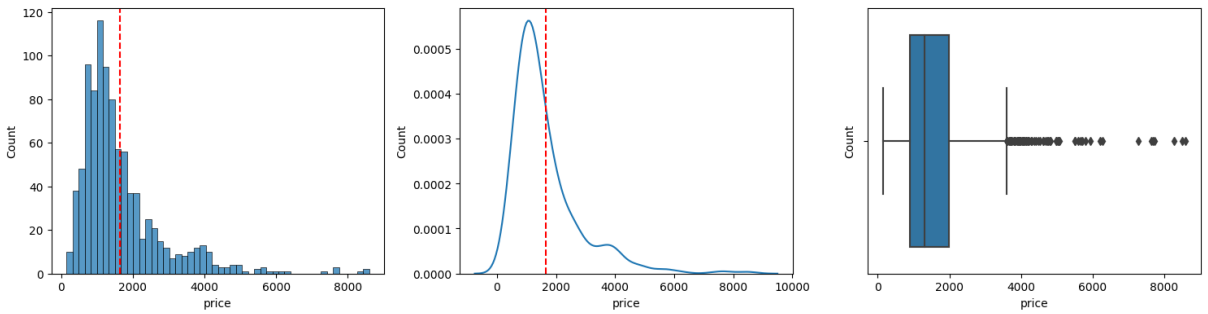
Một số thống kê mô tả về dữ liệu trong tập Small dataset



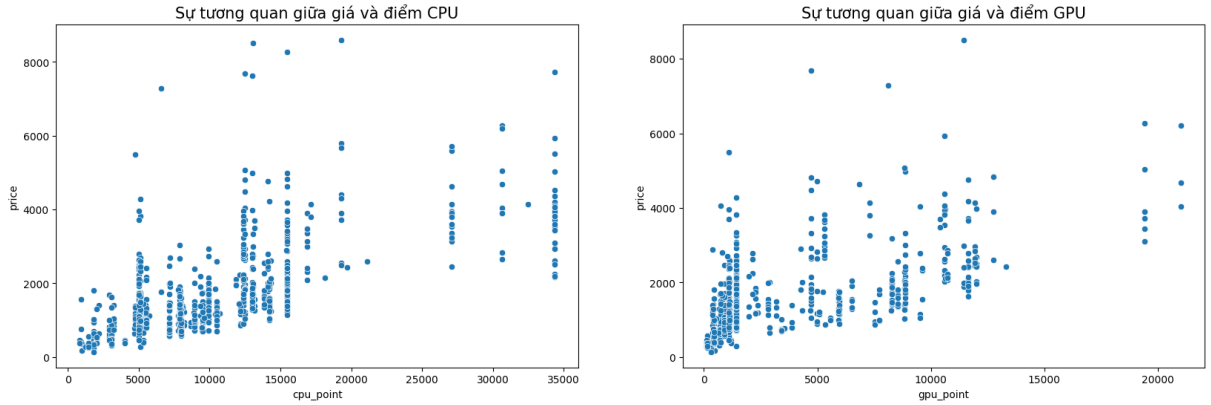

*Hình 6. Phần trăm dữ liệu null trong các cột trên Small dataset.*




*Hình 7. Biểu đồ cột số lượng hãng sản xuất laptop trên Small dataset*

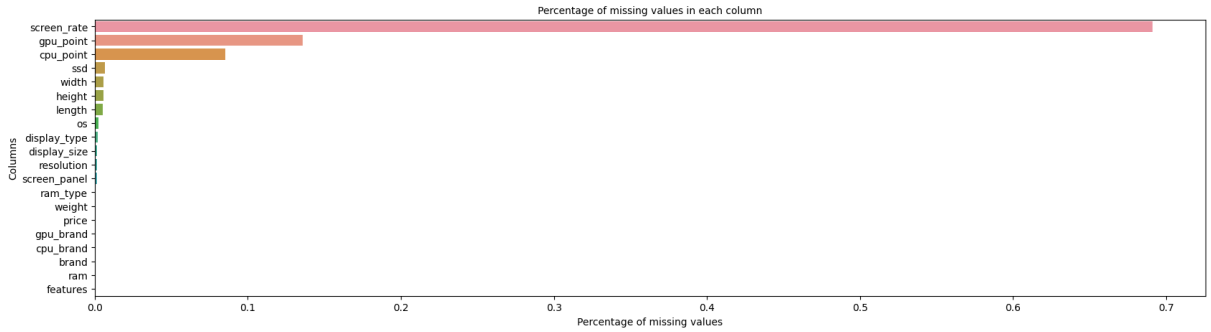



*Hình 8. Đồ thị histogram và boxplot của cột price trên Small dataset*

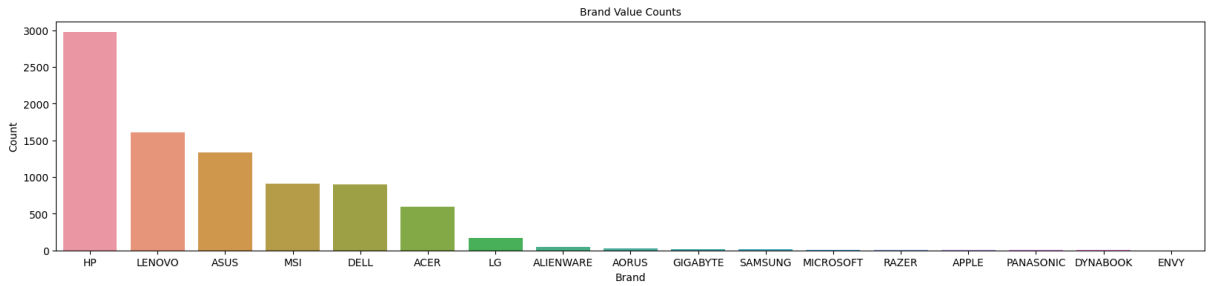



*Hình 9. Sự tương quan của điểm CPU, GPU với giá tiền trên Small dataset*

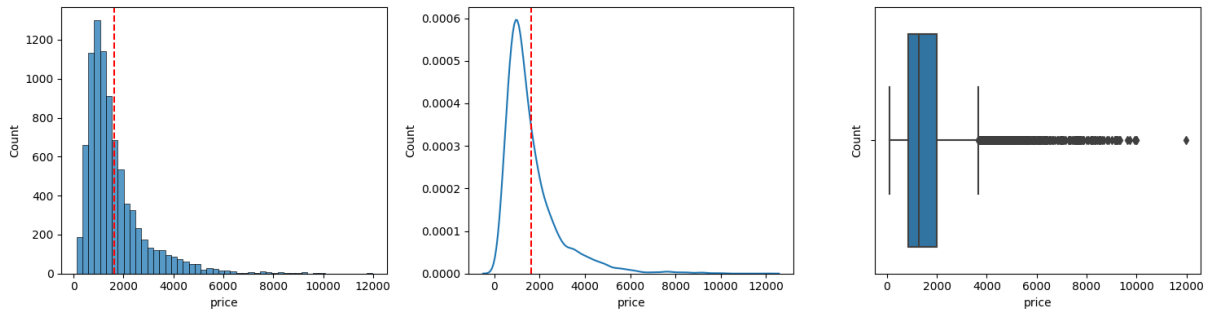
Một số thống kê mô tả trên tập Big dataset



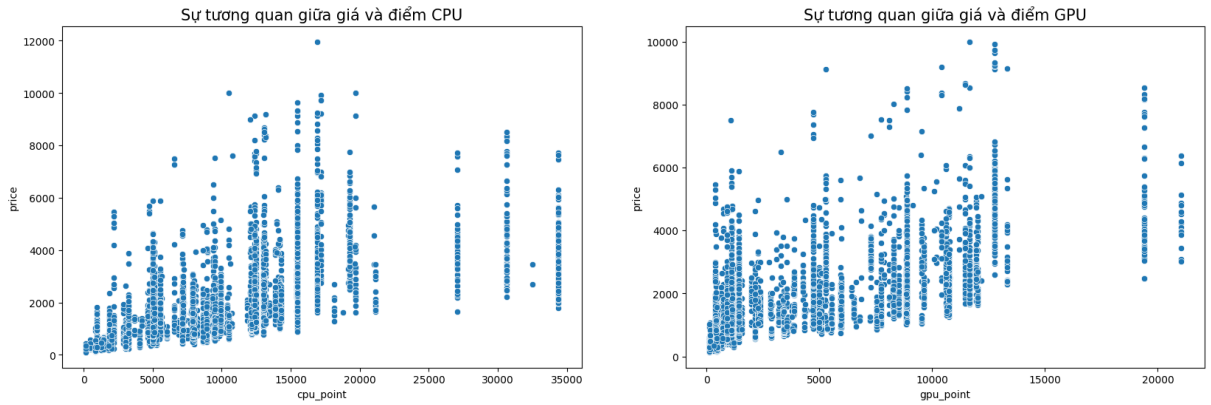

*Hình 10. Phần trăm dữ liệu null trong các cột trên Big dataset*




*Hình 11. Biểu đồ cột số lượng hãng sản xuất laptop trên Big dataset*




*Hình 12. Đồ thị histogram và boxplot của cột price trên Big dataset*




*Hình 13. Sự tương quan của điểm CPU, GPU với giá tiền trên Big dataset*

# 3. Trích xuất đặc trưng

## 3.1. Xử lý dữ liệu trống

Đối với dữ liệu thuộc kiểu category, để đảm bảo quá trình xử lý về sau, nhóm tiến hành xóa toàn bộ dữ liệu trống, đồng thời xóa các giá trị chiếm không quá 1% tổng số bảng ghi trong các cột.

Đối với dữ liệu số, sử dụng phương pháp Iterative imputer để lấp đầy dữ liệu trống. Phương pháp này dùng dữ liệu các trường còn lại làm X, dữ liệu trống làm y. Tiến hành huấn luyện mô hình hồi quy trên dữ liệu có sẵn để dự đoán giá trị tại các ô trống với mục tiêu làm cho những dữ liệu trống đó tự nhiên nhất, ít làm ảnh hưởng đến các thuộc tính khác khi huấn luyện.

## 3.2. Mã hóa dữ liệu phân loại

Dữ liệu thuộc loại category cần được mã hóa sang dạng số để có thể huấn luyện và dự đoán vì các mô hình sẽ không làm việc với các dữ liệu kiểu chuỗi ký tự. Dữ liệu category có các giá trị trong thuộc tính có vai trò như nhau, vì vậy trong tập dữ liệu của đề tài này, các cột có kiểu dữ liệu là Object sẽ được mã hóa theo One hot vector.

**Original Data**

| Team | Points |
|---|---|
| A | 25 |
| A | 12 |
| B | 15 |
| B | 14 |
| B | 19 |
| B | 23 |
| C | 25 |
| C | 29 |

**One-Hot Encoded Data**

| Team_A | Team_B | Team_C | Points |
|---|---|---|---|
| 1 | 0 | 0 | 25 |
| 1 | 0 | 0 | 12 |
| 0 | 1 | 0 | 15 |
| 0 | 1 | 0 | 14 |
| 0 | 1 | 0 | 19 |
| 0 | 1 | 0 | 23 |
| 0 | 0 | 1 | 25 |
| 0 | 0 | 1 | 29 |

*Hình 14. Mô tả cách mã hóa One hot vector*

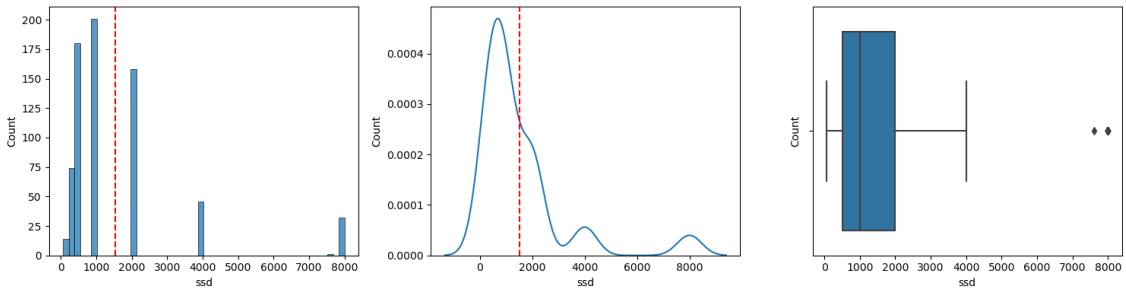
### 3.3. Chia dữ liệu ban đầu thành các tập Huấn luyện/Kiểm thử

Lấy 80% dữ liệu trong tập dữ liệu đã làm sạch để huấn luyện và 20% dữ liệu còn lại không liên quan đến 80% trước để đánh giá. Ta sẽ kết hợp chia đều theo hãng sản xuất để đảm bảo sự phân bố của dữ liệu trong tập huấn luyện và kiểm thử. Vậy, ta sử dụng hàm train_test_split() chia tập dữ liệu thành tập huấn luyện/Kiểm thử theo tỷ lệ 80/20. Kết quả như sau:

- Small dataset:
    - Training set: 80% ~ 709 mẫu.
    - Testing set: 20% ~ 178 mẫu.
- Big dataset:
    - Training set: 80% ~ 6394 mẫu.
    - Testing set: 20% ~ 1599 mẫu.

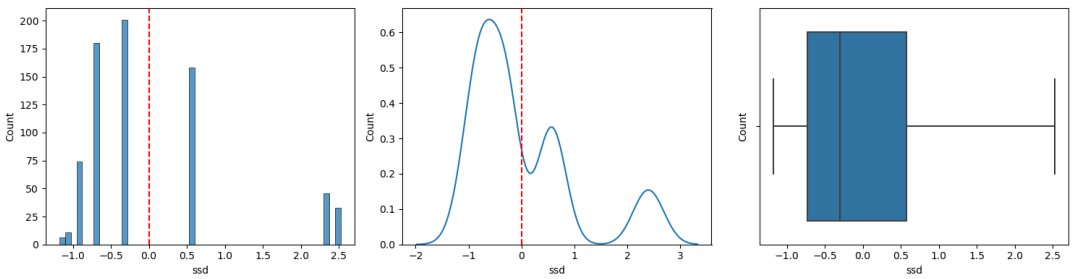
### 3.4. Xử lý ngoại lệ cho tập huấn luyện

Một số mô hình rất nhạy với giá trị ngoại lệ trong đó có mô hình hồi quy tuyến tính, vì vậy cần kéo các giá trị ngoại lệ về khoảng giá trị chấp nhận được. Có hai dạng xử lý ngoại lệ là xử lý dữ liệu phân bố chuẩn và xử lý ngoại lệ bị mất cân bằng. Lưu ý các cận được tính trên tập train và việc gán giá trị được thực hiện trên cả tập train và tập test.




*Hình 15. Dữ liệu về dung lượng SSD trước khi xử lý ngoại lệ*

Qua biểu đồ boxplot ở trên cho thấy tất cả các thuộc tính đều có giá trị ngoại lệ vì vậy việc xử lý ngoại lệ theo cách trên được áp dụng cho tất cả các trường.




*Hình 16. Dữ liệu về dung lượng SSD sau khi xử lý ngoại lệ*

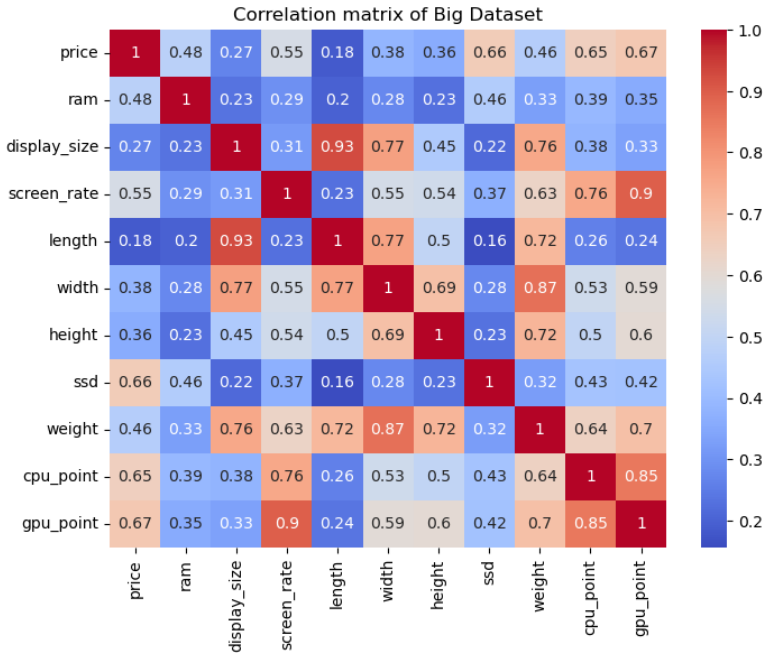
### 3.5. Chuẩn hóa dữ liệu

Nhóm tiến hành chuẩn hóa dữ liệu có dạng số sử dụng hàm Standard Scaler trong thư viện sklearn.preprocessing. Lưu ý: mean và std được tính trên tập huấn luyện, sau đó chuẩn hóa cho cả tập huấn luyện và tập kiểm thử.

### 3.6. Lựa chọn đặc trưng

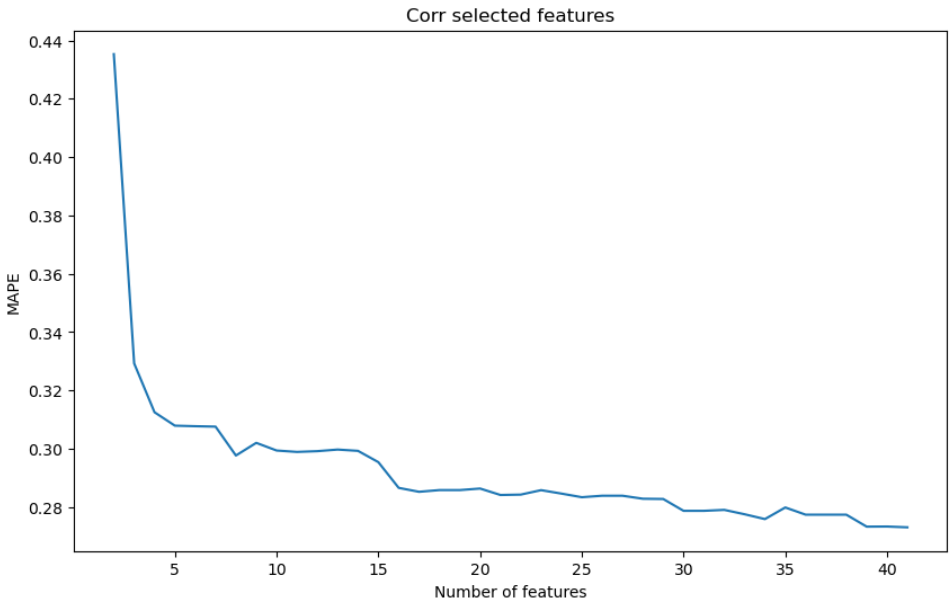
Tập dữ liệu có nhiều thuộc tính không đồng nghĩa với việc hiệu suất sẽ tốt. Đôi khi các thuộc tính lỗi lại gây giảm hiệu suất cho mô hình dự đoán. Vì vậy, nhóm tiến hành lựa chọn và giữ lại những đặc trưng đắt giá của mô hình. Mô hình được lựa chọn để kiểm thử các phương pháp lựa chọn đặc trưng là mô hình Linear Regression với dữ liệu huấn luyện.

#### 3.6.1. Lựa chọn đặc trưng dựa vào correlation matrix.




*Hình 17. Correlation matrix của tập huấn luyện*

Ở phương pháp này, ý tưởng chính là loại trừ các feature có mức độ tương quan cao đến nhau dẫn đến việc dư thừa thông tin khi suy ra giá trị biến phụ thuộc và tiến hành việc lựa chọn lần lượt *i* đặc trưng có mức độ tương quan cao nhất với biến target sao cho MAPE là thấp nhất.

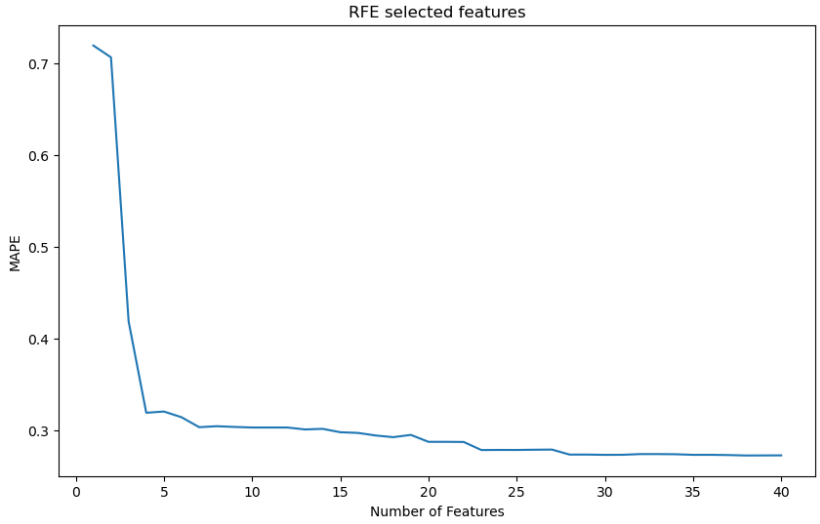



*Hình 18. Features Selection với kĩ thuật dựa trên ma trận tương quan*

Như ta thấy ở hình trên thì với việc sử dụng hết tất cả các đặc trưng mới cho ra MAPE thấp nhất vậy nên ta không thể loại bỏ được feature nào.

### 3.6.2. Recursive Feature Elimination (RFE)

Ở phương pháp này thì thuật toán RFE sẽ train model được định nghĩa trước (model được sử dụng là Linear Regression Model) trên toàn bộ features và xếp hạng feature theo mức độ quan trọng. Các đặc trưng (feature) có mức độ quan trọng thấp nhất sẽ bị loại bỏ. Quá trình này sẽ lặp đi lặp lại đến khi đã đạt được số lượng đặc trưng *i* mong muốn.



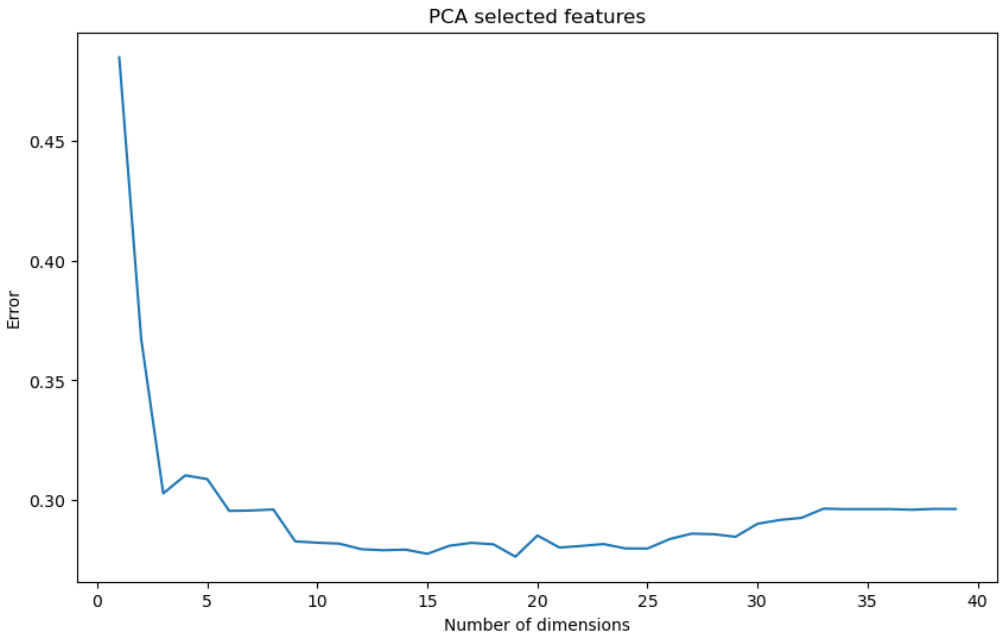

*Hình 19. Features Selection với kĩ thuật RFE*

Ta có thể thấy MAPE cho ra kết quả thấp nhất với nhiều features nhất có thể nên ta không thể loại bỏ được feature nào.

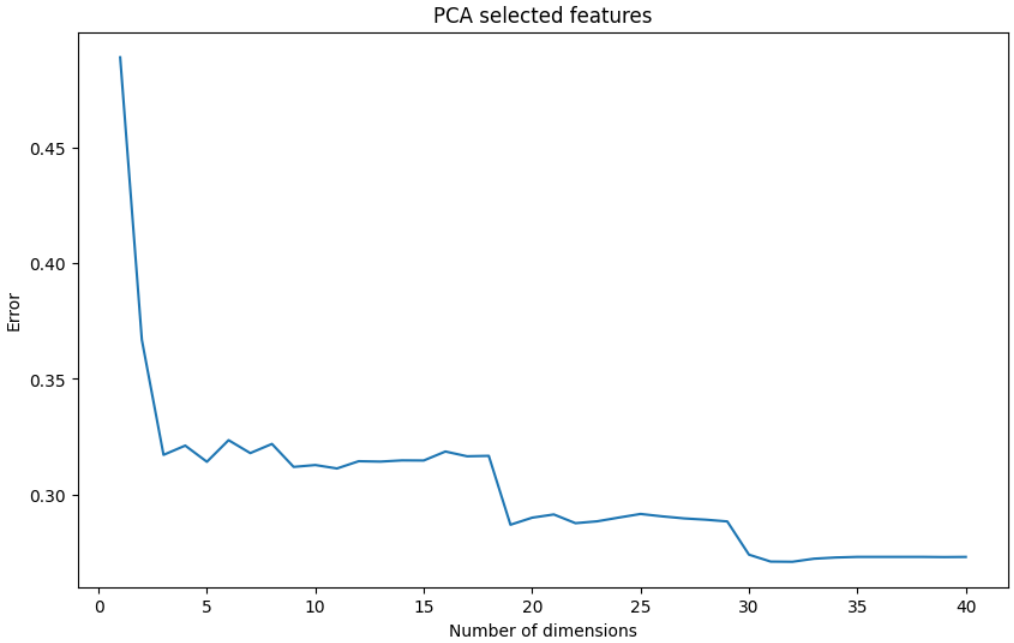
### 3.7. Giảm chiều dữ liệu

Phương pháp giảm chiều dữ liệu thường được dùng để giảm kích thước của dữ liệu, đồng thời tìm ra quy luật ẩn, một không gian mới mà thể hiện tốt hơn thuộc tính đó. Phân tích thành phần chính (PCA) là phương pháp giảm chiều dữ liệu khá phổ biến với mục tiêu tìm ra không gian mới mà mức độ phân tán dữ liệu càng cao càng tốt.

Trong đề tài này việc áp dụng phương pháp PCA giúp cải thiện MAPE đến 2% trên Small Dataset với số chiều 18 và 0.21% trên Big Dataset với số chiều 31.




*Hình 20. PCA trên Small Dataset.*




*Hình 21. PCA trên Big Dataset*

# 4. Mô hình hóa dữ liệu

## 4.1. Mô hình sử dụng

### 4.1.1. Linear Regression – Hồi quy tuyến tính

Hồi quy tuyến tính được xây dựng dựa trên giả định rằng quan hệ giữa biến độc lập và biến phụ thuộc có thể được miêu tả bằng một đường thẳng. Vì vậy, nhiệm vụ của chúng ta là tìm được một đường thẳng tốt nhất để biểu diễn mô hình này sao cho khoảng cách giữa các điểm dữ liệu thực tế và điểm dữ liệu dự đoán trên đường thẳng là nhỏ nhất.

- Phương trình tuyến tính của mô hình (vectorized form): $\hat{\boldsymbol{y}} = \boldsymbol{w}.\boldsymbol{X} + \boldsymbol{b}$

  **Trong đó**:

  - $\hat{y}$ là vector chứa giá trị dự đoán.
  - X là ma trận đặc trưng, với số hàng là số lượng mẫu, số cột là số lượng đặc trưng,
  - **w**: vector trọng số ứng với từng phần tử trong X.
  - b: vector hệ số tự do, có số phần tử bằng số lượng mẫu.

- **Loss function :**

$$\boldsymbol{MSE}(\boldsymbol{w}) = \frac{\mathbf{1}}{\mathbf{2}}.\frac{\mathbf{1}}{\boldsymbol{m}}\sum_{\boldsymbol{i}=\mathbf{1}}^{\boldsymbol{m}}(w^{\boldsymbol{T}}\boldsymbol{x}^{(\boldsymbol{i})} - \boldsymbol{y}^{\boldsymbol{i}})^{\mathbf{2}}$$

  **Trong đó:**

  - $\boldsymbol{x}^{(\boldsymbol{i})}$ là mẫu thứ i.
  - $\boldsymbol{y}^{(\boldsymbol{i})}$ là giá trị cần dự đoán của mẫu thứ i.

- **Thuật toán:**
  - **1. Chuẩn bị dữ liệu:** Thu nhập dữ liệu bao gồm biến phụ thuộc (y) và biến độc lập (x). Đảm bảo dữ liệu đã được clean (đã xử lí ngoại lệ, missing values, …).
  - **2. Khởi tạo tham số:** Khởi tạo ma trận trọng số **W** và vector bias **b** với giá trị ngẫu nhiên.
  - **3. Gradient descent:** Định nghĩa thuật toán gradient descent để cập nhật tham số của mô hình. Mục tiêu là để cực tiểu hoá hàm cost bằng cách điều chỉnh tham số theo chiều ngược lại với gradient.
  - **4. Train model:** Với tất cả thông tin về hypothesis function (phương trình tuyến tính), hàm cost function, thuật toán tối ưu tham số ta tiến hành việc lặp qua dataset với số lần xác định (epochs). Mỗi lần lặp qua dataset ta tính giá trị dự

đoán $\hat{\boldsymbol{y}}$ sử dụng hypothesis function, sau đó tính **gradient** và cập nhật tham số theo công thức sau:

$$w = w - \alpha \frac{\mathbf{1}}{\boldsymbol{m}} \boldsymbol{X^T}(\boldsymbol{X}.\boldsymbol{w} + \boldsymbol{b} - \boldsymbol{y})$$

$$b = b - \alpha \frac{\mathbf{1}}{\boldsymbol{m}} \sum_{\boldsymbol{i=1}}^{\boldsymbol{m}} (\boldsymbol{X}.\boldsymbol{w} + \boldsymbol{b} - \boldsymbol{y})$$

Trong đó :

- X là ma trận đặc trưng của đầu vào
- y là vector chứa giá trị cần dự đoán
- m là số lượng mẫu dùng để train

- **5. Đánh giá mô hình:** Tiến hành dự đoán trên tập test sử dụng bộ trọng số w, bias vừa train xong và đánh giá kết quả dựa vào metrics được sử dụng trong đề tài RMSE, MAE, MAPE.

Với tiểu luận này, nhóm chúng em sử dụng Linear Regression do thư viện sklearn cung cấp nên tham số α đã được lựa chọn tối ưu nhất.

### 4.1.2. Support Vector Regression

Support Vector Regression (SVR) là thuật toán được sử dụng để dự đoán giá trị liên tục dựa vào các biến độc lập. SVR được phát triển từ thuật toán Support Vector Machine (SVM) và chú trọng vào việc xây dựng một đường hồi quy tuyến tính tốt nhất trong không gian đặc trưng. Để đạt được mục tiêu thì thuật toán SVR sử dụng hàm loss tương ứng và các ràng buộc để tối thiểu hoá sai số dự báo và đồng thời đảm bảo rằng sai số không vượt quá một ngưỡng ($\varepsilon$) cho trước. Trong đề tài này nhóm chúng em sử dụng SVR với kernel là hàm tuyến tính (linear).

- **Hypotheis function :**

$$\hat{y} = w^T X + b$$

**Trong đó:**

- $\hat{\boldsymbol{y}}$ : Giá trị dự đoán
- $\boldsymbol{X}$ : Ma trận đặc trưng của đầu vào
- $\boldsymbol{w}$ : Vector trọng số
- $\boldsymbol{b}$ : Vector bias

- **Loss function:** Có nhiều hàm loss có thể áp dụng cho SVR bao gồm : linear, quadratic, huber [5]. Để giảm độ phức tạp cho quá trình training trong đề tài này nhóm em sẽ sử dụng linear loss.

$$L(y, \hat{y}) = max(0, |y - \hat{y}| - \varepsilon)$$

- **Thuật toán:**
    - **1. Chuẩn bị dữ liệu:** Thu nhập dữ liệu bao gồm biến phụ thuộc (y) và biến độc lập (x). Đảm bảo dữ liệu đã được clean (đã xử lí ngoại lệ, missing values, …).
    - **2. Train model:** Thuật toán tối ưu sẽ gồm 2 phần bao gồm tìm tham số tối ưu cho mô hình sao cho cực tiểu hoá hàm $\varepsilon$ -insensitive loss function và regularization để kiểm soát độ phức tạp của mô hình

$$minimize\ \frac{1}{2}||w||^2\ +\ C\ *\ \sum_{i=0}^{M} max\left(0, \left|y^{(i)} - \hat{y}^{(i)}\right| - \varepsilon\right)$$

Trong đó:

- w: vector trọng số của mô hình
- C: Siêu tham số của mô hình để kiểm soát việc cực tiểu hoá độ phức tạp tính toán của mô hình hay cực tiểu hoá hàm loss.
- $\varepsilon$: Siêu tham số của mô hình dùng để điều chỉnh chiều rộng của cạnh xung quanh giá trị target.
- $y^{(i)}$: Giá trị target của mẫu thứ i.
- $\hat{y}^{(i)}$: Giá trị dự đoán của mẫu thứ i

Sử dụng thuật toán gradient descent để tìm giá trị tối ưu của w và b.

- **3. Fine tune model:** Xác định các bộ siêu tham số tối ưu cho mô hình.
- **4. Đánh giá model:** Tiến hành dự đoán trên tập test sử dụng bộ trọng số w, bias vừa train xong và đánh giá kết quả dựa vào metrics được sử dụng trong đề tài RMSE, MAE, MAPE.

### 4.2. Điều chỉnh tham số huấn luyện

**4.2.1. Linear Regression:** không có siêu tham số để điều chỉnh

**4.2.2. Support Vector Regression**

Hai tham số cần điều chỉnh của mô hình SVR với kernel linear là:

- C: tham số điều chỉnh cho kĩ thuật regularization. Phải là số dương (bắt buộc), biểu thức regularization tỉ lệ nghịch với C và công thức là l2 penalty.
- epsilon: tham số quyết định khoảng cách giữa cạnh xung quanh giá trị target.

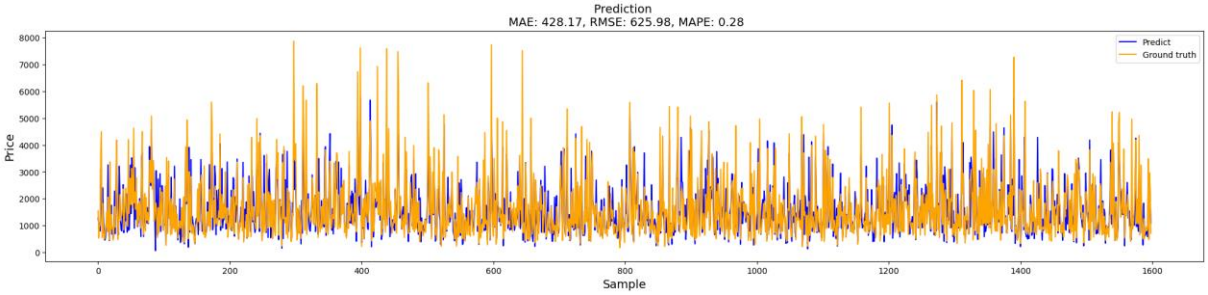
Sử dụng RandomizedSearchCV với n_iters=100, ta thu được bộ tham số tốt nhất như sau

- Big dataset: C = 9.9565 và epsilon: 0.252
- Small dataset: C = 9.9065 và epsilon = 0.252

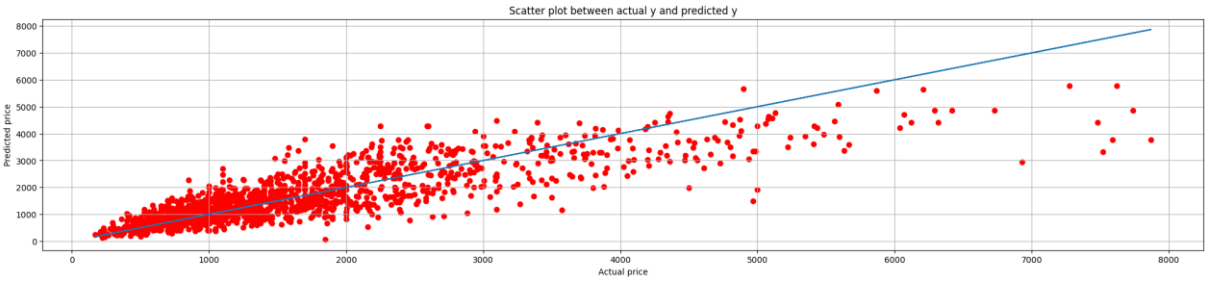
## 4.3. Kết quả của các mô hình

### 4.3.1. Linear Regression

**Kết quả trên Big Dataset:**

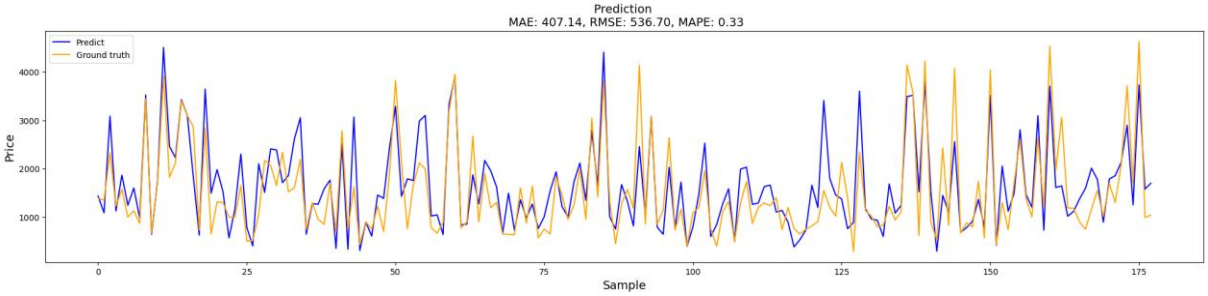



*Hình 22. Biểu đồ đường giá dự đoán và giá thực tế mẫu sử dụng Linear Regression*

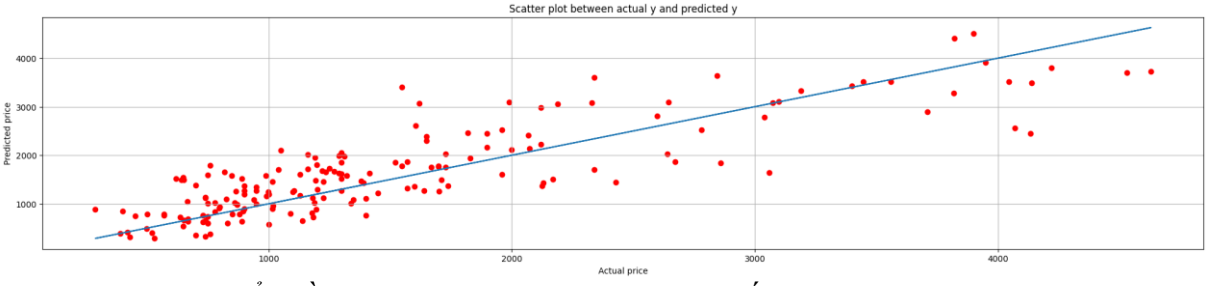



*Hình 23. Biểu đồ scatter giá dự đoán và giá thực Linear Regression*

**Kết quả trên Small Dataset:**



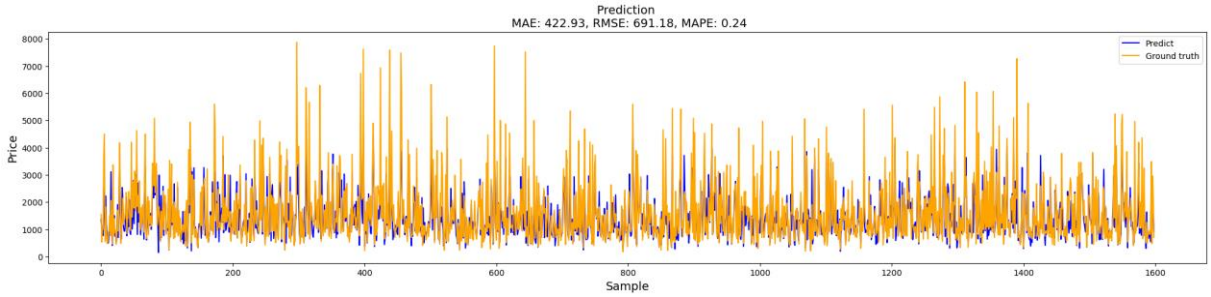

*Hình 24. Biểu đồ đường giá dự đoán và giá thực tế sử dụng Linear Regression*



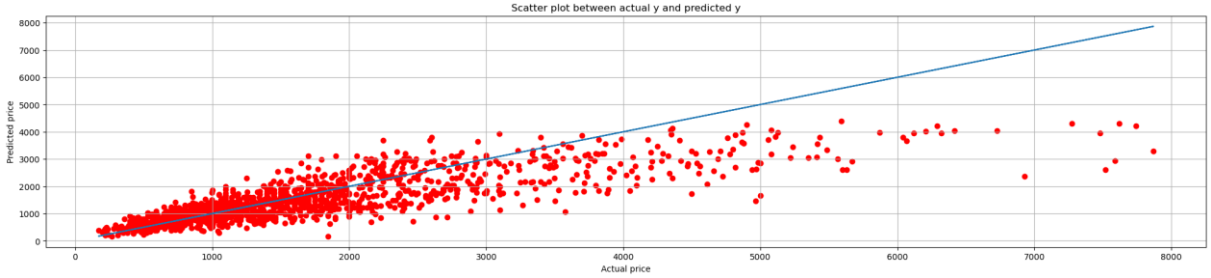

*Hình 25. Biểu đồ scatter giá dự đoán và giá thực tế sử dụng Linear Regression*

### 4.3.2. Linear Support Vector Regression

**Kết quả trên Big Dataset:**

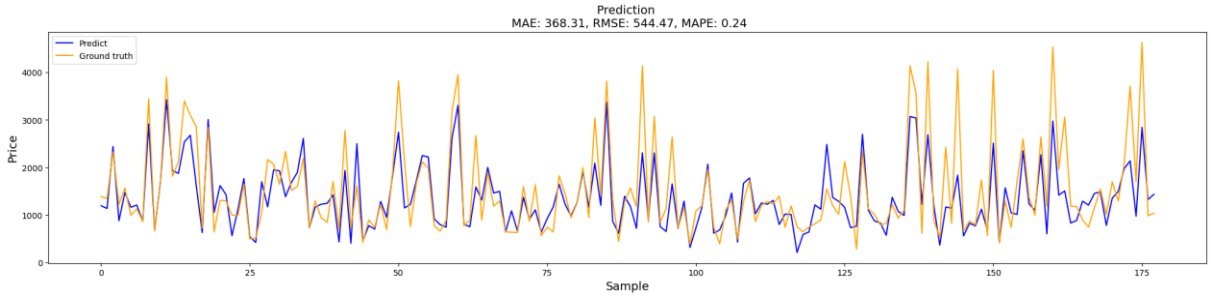



*Hình 26. Biểu đồ đường giá dự đoán và giá thực tế trên sử dụng SVR*




*Hình 27. Biểu đồ scatter giá dự đoán và giá thực sử dụng SVR*

**Kết quả trên Small Dataset:**




*Hình 28. Biểu đồ đường giá dự đoán và giá thực tế trên sử dụng SVR*


*Hình 29. Biểu đồ scatter giá dự đoán và giá thực sử dụng SVR*

### 4.4. Metrics đánh giá mô hình

#### 4.4.1. Khái niệm và mô tả

- ***MAE (Mean absolute error):*** Sai số tuyệt đối trung bình.

$$\frac{\sum_{i=1}^{n} |y_i - y'_i|}{n}$$

- ***RMSE (Root mean square error):*** Sai số trung bình bình phương. Tương tự với MAE nhưng thay vì tính trung bình trị tuyệt đối thì RMSE tính căn bậc hai của trung bình bình phương độ lệch giữa giá trị dự đoán và giá trị thực tế.

$$\sqrt{\frac{\sum_{i=1}^{n} (y_i - y'_i)^2}{n}}$$

- ***MAPE (Mean absolute percentage error):*** Sai số tuyệt đối trung bình phần trăm. MAPE cho biết các giá trị dự đoán trung bình sai lệch bao nhiêu phần trăm so với giá trị thực tế

$$\frac{1}{n} \sum_{i=1}^{n} \left| \frac{y_i - y'_i}{y_i} \right|$$

#### 4.4.2. Kết quả đánh giá

| Mô hình | Metrics | Small DS | Big DS |
|---|---|---|---|
| **Linear Regression** | MAE (USD) | 407.1 | 428.2 |
| | RMSE (USD) | 536.7 | 625.9 |
| | MAPE (%) | 32.6 | 27.7 |
| **Support Vector Regression** | MAE (USD) | 368.3 | 422.9 |
| | RMSE (USD) | 544.5 | 691.2 |
| | MAPE (%) | 24.1 | 24.0 |

*Bảng 2. Kết quả đánh giá mô hình*

# 5. Kết luận:

## 5.1. Hiệu suất của mô hình:

Ta có một số kết luận về hiệu suất của mô hình như sau:

- Hiệu suất của mô hình SVR tốt hơn so với mô hình Linear Regression.
- Trong đó chênh lệch trên MAE và RMSE không nhiều tuy nhiên đối với MAPE thì ta có thể thấy được sự chênh lệch lên đến 8% trên small DS và 3.8% trên big DS.
- Nhóm đánh giá hầu hết sai số đến từ việc dự liệu phân bố không đồng đều trên mọi điểm dữ liệu. Trong đó, những mẫu laptop có giá tầm trung từ 4200$ trở xuống kết quả dự đoán khá khớp trong khi các mẫu laptop có giá tầm cao trở lên hầu như đều sai lệch so với giá trị dự đoán dẫn đến sai số lớn.

## 5.2. Giải thích, dự đoán nguyên nhân:

Một số giải thích và dự đoán cho kết quả đạt được như sau:

- Mô hình Linear Regression được xây dựng dựa trên việc tìm ra đường thẳng sao cho fit nhiều điểm nhất có thể trên đường thẳng này còn mô hình SVR được xây dựng dựa trên việc tìm ra best margin sao cho fit được nhiều dữ liệu nhất có thể trong support region của nó nên từ đó ta có được một đường thẳng tuyến tính mà các giá trị xung quanh xấp xỉ giá trị thực tế nhiều nhất nên cho ra MAPE tốt hơn so với mô hình Linear Regression.
- Đề tài chỉ sử dụng các mô hình tuyến tính đơn giản nên chưa thể khám phá ra được các đặc trưng tiềm ẩn của dữ liệu và chưa xây dựng được mối quan hệ phi tuyến giữa các đặc trưng dẫn đến việc metrics của mô hình chưa cao.
- Ngoài ra việc MAPE của mô hình SVR và mô hình Linear Regression xấp xỉ nhau là vì dữ liệu phân bố không được đều, dữ liệu còn thiếu nhiều mẫu laptop có giá trị cao trong cả 2 dataset.

## 5.3. Hướng phát triển

Mặc dù dữ liệu thu thập có nhiều thuộc tính nhưng chưa khai thác hết các thuộc tính đó, sau bước lựa chọn thuộc tính thì loại bỏ rất nhiều thuộc tính, đặc biệt là thuộc tính thuộc loại category. Vì vậy cần lựa chọn phương thức mã hóa thích hợp để khai thác các thuộc tính này.

- Thu thập thêm dữ liệu, làm đa dạng dữ liệu

- Thử nghiệm down sample (đối với những mẫu chiếm tỉ lệ lớn trong tập dữ liệu) hoặc up sample (đối với những mẫu dữ liệu chiếm tỉ lệ thấp trong tập dữ liệu) để giúp cân bằng dữ liệu.
- Sử dụng một số mô hình khác như: Random Forest Regressor, XGBoost,...

**6. Tài liệu tham khảo**

[1] Nasima Tamboli, All You Need To Know About Different Types Of Missing Data Values And How To Handle It.

[2] Tiep Vu Huu, Bài 27: Principal Component Analysis (phần 1/2), https://machinelearningcoban.com/2017/06/15/pca/

[3] Tiep Vu, "Xử lý các giá trị ngoại lệ", Machine Learning cho dữ liệu dạng bảng, https://machinelearningcoban.com/tabml_book/ch_data_processing/process_outliers.html.

[4] FirebirD, "Lý thuyết Hồi quy tuyến tính Linear Regression", https://allaravel.com/blog/ly-thuyet-hoi-quy-tuyen-tinh-linear-regression.

[5] Mariette Awad & Rahul Kanna, "Support Vector Regression". https://link.springer.com/chapter/10.1007/978-1-4302-5990-9_4#:~:text=SVR%20adopts%20an%20%CE%B5%2Dinsensitive,%CE%B5%20from%20the%20desired%20output.