HỌC VIỆN NGÂN HÀNG

KHOA HỆ THỐNG THÔNG TIN QUẢN LÝ

BÁO CÁO

ĐỀ TÀI:

XÂY DỰNG MÔ HÌNH DỰ ĐOÁN GIÁ BÁN LAPTOP

Giảng viên hướng dẫn: Vũ Trọng Sinh

Nhóm thực hiện: Nhóm 24 lớp 212IS42A18

Hà nội,ngày 17 tháng 6 năm 2022

MỤC LỤC

Bảng 1: Bảng phân công công việc.

Họ và tên	Mã SV	Công việc	Phần trăm đóng góp
Nguyễn Thị Diệu Linh	23A4010352	Giới thiệu mô hình và tập toán, hỗ trợ làm word, powpoint, mã lệnh	33,5%
Zonthiza SOUKKASEUN	23A4011162	Tổng quan về thị trường, làm word, hỗ trợ làm powpoint, mã lệnh	33,5%
Ngô Hoàng Ngọc	23A4010463	Mã lệnh chương trình, phân tích tập dữ liệu, powpoint, quay video	33%

Nội dung

I. Tổng quan về thị trường.

Laptop ngày nay đã trở thành thiết bị không thể thiếu trong nhu cầu giải trí, học tập và làm việc của mỗi người trong xã hội. Nhưng nhiều người nếu muốn mua laptop thì phải tự tìm hiểu rất nhiều chi tiết của chiếc laptop để có thể biết giá bán của nó, nhiều người không biết đã bị bán đắt lên nhiều lần khiến việc mua laptop thành nỗi lo sợ với nhiều người.

Tất cả các loại hình dịch vụ đều phát sinh từ nhu cầu thực tế của người tiêu dùng. Cũng giống như smartphone nói riêng hay điện thoại nói chung, laptop cũng có riêng một thị trường đầy năng động, phong phú . Chỉ tính riêng trên thị trường Hà Nội, các cửa hàng mua bán laptop cũng nhiều như "nấm". Ngoài các con phố có truyền thống như Lê Thanh Nghị hay Thái Hà, còn mở rộng thêm các địa chỉ mới như Khuất Duy Tiến, Hoàng Văn Thái…

Vấn đề đặt ra

Người có nhu cầu mua laptop mới muốn tìm mua laptop nhưng lại không có thời gian tìm hiểu sự ảnh hưởng của các linh kiện máy tính tới giá của chiếc laptop. Mua tại các showroom laptop thì sợ đắt, mua tại các cửa hàng xách tay thì sợ bị mua với giá cao. Chúng ta cần có sự hỗ trợ của trí tuệ nhân tạo để dễ dàng định giá laptop cho người mua.

Còn với người bán, cửa hàng bán laptop việc tận dụng trí tuệ nhân tạo trong việc định giá laptop có thể tiết kiệm chi phí nhân công định giá khi nhập hàng, kế toán…

Các tác động đến giá bán Laptop trên thị trường

Thị trường sản phẩm laptop ngày càng đa dạng về cấu hình, kiểu dáng, màu sắc và cả thương hiệu.Những laptop mang cấu hình, phần cứng xịn xò sẽ đắt hơn những laptop có cấu hình yếu hơn và những laptop mang thương hiệu nổi tiếng như Apple, Vaio, Dell sẽ có giá thành cao hơn so với các dòng của LG, Acer…

Lý do chọn đề tài

Vài năm covid trở lại đây, đã có sự bùng nổ về nhu cầu mua Laptop cho các bạn trẻ học online, work from home,….vì Laptop rất thuận tiện khi sử dụng, nhỏ gọn dễ dàng mang đi mà có những chiếc máy cấu hình không thua kém máy cây.

Nắm bắt được nhu cầu laptop ngày càng tăng cao: sinh viên, nhân viên,… ai cũng cần cho bản thân một chiếc laptop để học tập và làm việc. Nhóm em tin rằng việc định giá laptop bằng trí tuệ nhân tạo có thể tạo ra sự cạnh tranh lớn trong thị trường laptop này.Chính vì vậy nhóm em đã chọn nghiên cứu đề tài: “Xây dựng mô hình dự đoán giá bán laptop”.

II. Phân tích tập dữ liệu.

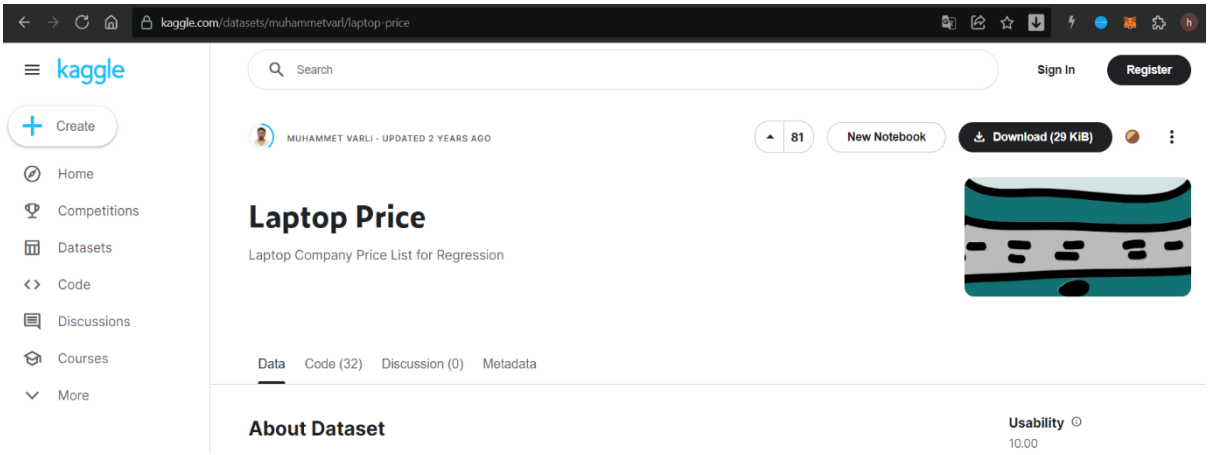
1. Giới thiệu cách thức thu thập dữ liệu.

Sau khi tham gia môn học “Trí tuệ nhân tạo trong kinh doanh”, đồng thời được

giảng viên phổ biến về bài tập lớn và thảo luận với nhóm thì chúng em đã quyết định chọn (chủ đề " Dự đoán giá bán của laptop trên thị trường".)

Bọn em thu thập dữ liệu trên trang mạng cộng đồng Kaggle: https://www.kaggle.com/datasets/muhammetvarl/laptop-price

(Đây là một trang web được cộng đồng phân tích dữ liệu tin tưởng, dữ liệu không bị thừa và rõ ràng trong phân tích)



Hình 1. Trang web thu thập dữ liệu.

2. Phân tích, xác định các thuộc tính dữ liệu cần thu thập.

Khi bắt tay vào phân tích nhóm em đã phải phân tích những thuộc tính nào của chiếc laptop có thể ảnh hưởng tới giá cả, bọn em dựa vào kiến thức của bản thân cộng thêm đọc qua hàng nhiều bài báo lớn nhỏ để có thể hình dung ra những cái gì quan trọng trong việc phân tích giá cả của laptop.

Chính vì vậy bọn em khi chọn xong nguồn dữ liệu phù hợp trên mạng, bọn em đã phải tiến hành thêm bước là làm sạch dữ liệu để tăng độ chính xác mô hình lên cao hơn.

Dữ liệu trước khi xử lí:

Tên thuộc tính	Mô tả	Kiểu dữ liệu	Minh họa
Company	Tên hãng sản xuất	Object	Apple, Asus,…
Product	Tên sản phẩm	Object	Mac book, inspiron,…
TypeName	Loại sản phẩm	Object	Notebook,ultrabook,…
Inches	Kích thước màn hình	float64	13.3,15.6,…
ScreenResolution	Độ phân giải màn hình	Object	1440x990,…
Cpu	Bộ xử lí trung tâm	Object	Intel core i5, Intel core i7,…
Ram	Bộ nhớ truy xuất ngẫu nhiên	Object	4GB,8GB,…

Memory	Bộ nhớ	Object	256SSD, 1TB HDD
GPU	Bộ xử lí đồ họa	Object	Intel HD graphics,…
OpSys	Hệ điều hành	Object	Window 7, Mac OS
Weight	Khối lượng máy	Object	1.3kg,2.04kg,…
Price_euros	Giá theo euro	float64	575,1000,2300,…

Sau khi đọc qua dữ liệu thì em thấy dữ liệu còn chưa phù hợp lắm:

-Một số thuộc tính có thể ở dạng số học như: Ram, weight,….

-Tên sản phẩm thì bị thừa nhiều chỗ.

-Thuộc tính Cpu và Gpu có thể được chia ra rõ hơn nữa thành: Hãng cung cấp Cpu, Thế hệ Cpu, Tốc độ Cpu, nơi sản xuất Gpu, loại Gpu.

-Màn hình có các tính năng IPS hay touch screen không ?

-Hệ điều hành có thể quy về Windows, Mac, phần còn lại.

Mục tiêu của bọn em là sẽ xử lí dữ liệu thô thành các kiểu dữ liệu như sau để phân tích:

Tên thuộc tính	**Mô tả**	**Kiểu dữ liệu**	**Minh họa**
Company	Tên hãng sản xuất	Object	Apple, Asus,…
TypeName	Tên sản phẩm	Object	Mac book, inspiron,…
Inches	Kích thước màn hình	float64	Notebook,ultrabook,…
Touchscreen	Có touch screen hay không?	Int 64	1: Có touchscreen 0: không có touchscreen
ScreenResolution	Độ phân giải màn hình	Object	1440x990,…
Ips	Có bảng điều khiển Ips hay không	Int 64	1: Có IPS 0: không có IPS
Cpu_Vender	Nhà sản xuất Cpu	Object	Intel, Intel,…
Cpu_Type	Thế hệ Cpu	Object	core i5, core i7,…
Cpu_Speed	Tốc độ xử lí Cpu	float64	5.6GHz,..
Ram	Mức bộ nhớ Ram	float64	4GB,8GB,…
Memory	Bộ nhớ máy tính	Object	256 SSD, 1TB HDD
Gpu_vender	Nhà sản xuất Gpu	Object	AMD, Intel,…
Gpu_Type	Loại Gpu	Object	HD graphics,…
Weight	Khối lượng máy	float 64	1.3kg,2.04kg,…
OpSys	Hệ điều hành	Object	Window 7, Mac OS
Price_euros	Giá theo euro	Float64	575,1000,2300,…

3. Gán nhãn dữ liệu xây dựng mô hình.

-Dữ liệu của bọn em gồm 1301 giá chiếc laptop.

Số lượng thuộc tính: gồm 15 thuộc tính (1 biến phụ thuộc và 14 biến độc lập). Cụ thể như sau:

Biến độc lập:

Company
TypeName
Inches
Touchscreen
Ips
Cpu_Vender
Cpu_Type
Cpu_Speed
Ram
Memory
Gpu_Type
Weight
OpSys

Biến phụ thuộc:

Price_euros

Vì kiến thức, nhân lực và thời gian làm việc của nhóm còn hạn hẹp nên bọn em mới chỉ nghiên cứu và áp dụng được mô hình hồi quy tuyển tính, nên độ chính xác mô hình không cao có thể sẽ xảy ra.

III. Giới thiệu mô hình học máy và thuật toán sử dụng.

Từ những phân tích dữ liệu được nêu trên đều là cơ sở để hình thành nên giá bán của một chiếc laptop trên thị trường .Sau khi nghiên cứu và tìm hiểu từng thuộc tính đó, nhóm em đã sử dụng thuật toán Linear Regression - Hồi quy tuyến tính để có thể dự đoán được giá bán của một chiếc laptop trên thị trường

1. Giới thiệu về học máy có giám sát.

Học có giám sát là một kĩ thuật của ngành học máy để xây dựng một hàm (function) từ dữ liệu huấn luyện. Dữ liệu huấn luyện bao gồm các cặp gồm đối tượng đầu vào (thường dạng vec-tơ), và đầu ra mong muốn. Đầu ra của một hàm có thể là một giá trị liên tục (gọi là hồi quy), hay có thể là dự đoán một nhãn phân loại cho một đối tượng đầu vào (gọi là phân loại). Nhiệm vụ của chương trình học có giám sát là dự đoán giá trị của hàm cho một đối tượng bất kì là đầu vào hợp lệ, sau khi đã xem xét một số ví dụ huấn luyện (nghĩa là, các cặp đầu vào và đầu ra tương ứng). Để đạt được điều này, chương trình học phải tổng quát hóa từ các dữ liệu sẵn có để dự đoán được những tình huống chưa gặp phải theo một cách "hợp lý"

Học có giám sát có thể tạo ra hai loại mô hình. Phổ biến nhất, học có giám sát tạo ra một mô hình toàn cục (*global model*) để ánh xạ đối tượng đầu vào đến đầu ra mong

muốn.

Để có thể giải quyết một bài toán nào đó của học có giám sát ta phải xem xét nhiều bước khác nhau:

Xác định loại của các ví dụ huấn luyện. Trước khi làm bất cứ điều gì, chúng ta nên quyết định loại dữ liệu nào sẽ được sử dụng làm ví dụ. Như là một ký tự viết tay đơn lẻ, toàn bộ một từ viết tay, hay toàn bộ một dòng chữ viết tay.

Thu thập tập huấn luyện. Tập huấn luyện cần đặc trưng cho thực tế sử dụng của hàm chức năng. Vì thế, một tập các đối tượng đầu vào được thu thập và đầu ra tương ứng được thu thập, hoặc từ các chuyên gia hoặc từ việc đo đạc tính toán.

Xác định việc biểu diễn các đặc trưng đầu vào cho hàm chức năng cần tìm. Sự chính xác của hàm chức năng phụ thuộc lớn vào cách các đối tượng đầu vào được biểu diễn. Thông thường, đối tượng đầu vào được chuyển đổi thành một vec-tơ đặc trưng, chứa một số các đặc trưng nhằm mô tả cho đối tượng đó. Số lượng các đặc trưng không nên quá lớn, do sự bùng nổ tổ hợp (*curse of dimensionality*); nhưng phải đủ lớn để dự đoán chính xác đầu ra.

Xác định cấu trúc của hàm chức năng cần tìm và giải thuật học tương ứng. Ví dụ, người kĩ sư có thể lựa chọn việc sử dụng mạng nơ-ron thần kinh hay cây quyết định.

Hoàn thiện thiết kế. Chúng ta sẽ chạy giải thuật học từ tập huấn luyện thu thập được. Các tham số của giải thuật học có thể được điều chỉnh bằng cách tối ưu hóa hiệu năng trên một tập con (gọi là tập *kiểm chứng -validation* set) của tập huấn luyện, hay thông qua kiểm chứng chéo (*cross-validation*). Sau khi học và điều chỉnh tham số, hiệu năng của giải thuật có thể được đo đạc trên một tập kiểm tra độc lập với tập huấn luyện.

Thuật toán hồi quy tuyến tính (Linear Regression)

2. Giới thiệu chung về thuật toán hồi quy tuyến tính.

Hồi quy tuyến tính là thuật toán tìm ra phương trình tuyến tính dựa trên tập dữ liệu quan hệ giữa X (dữ liệu đầu vào) và Y (dữ liệu đầu ra). X là biến giải thích và Y là biến phụ thuộc. Ví dụ, tạo ra mô hình quan hệ giữa chiều cao và cân nặng bằng mô hình hồi quy tuyến tính. Trước khi thử tạo ra mô hình quan hệ, chúng ta nên xác định liệu giữa các mối quan hệ này có liên quan với nhau hay không. Điều này chỉ ra rằng, không nhất thiết phải có sự tương tác giữa các biến, nhưng cần phải có sự liên quan. Để chỉ ra mối quan hệ giữa các biến, chúng ta có thể dùng biểu đồ phân tán như một công cụ trong việc xác định mức độ liên quan. Nếu không có mối quan hệ nào giữa các biến được đưa vào mô hình, thì mô hình hồi quy tuyến tính sẽ không giúp ích được trong trường hợp này.

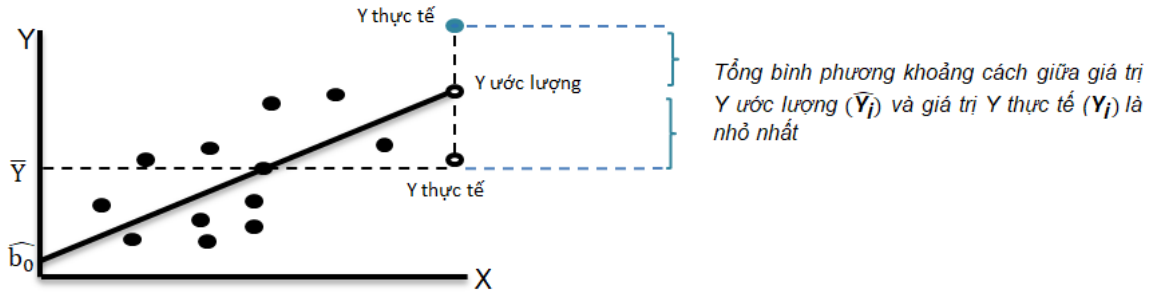
Mô hình hồi quy tuyến tính :

Công thức tổng quát của hồi quy tuyến tính:

$$Y_i = b_0 + b_1 X_i + \varepsilon_i,\ i = 1,\ldots,n$$

Trong đó:
Y = dependent variable (biến phụ thuộc)
X = independent variable (biến độc lập)
$\mathbf{b_0}$ = the intercept (hệ số chặn)
$\mathbf{b_1}$= the slope coefficient (hệ số góc)
$\boldsymbol{\varepsilon_i}$ = the error term (sai số)

Biến phụ thuộc (Y): Là biến mà sự thay đổi giá trị được giải thích thông qua biến độc lập.
Biến độc lập (X): Là biến được đưa vào mô hình để giải thích cho sự thay đổi giá trị của biến phụ thuộc.
Hệ số chặn (b_0): Thể hiện giá trị của biến phụ thuộc khi giá trị của biến độc lập bằng 0; tại giao điểm của đường hồi quy với trục biểu diễn giá trị của biến phụ thuộc (thường là trục tung).
Hệ số góc (b_1): Là số đơn vị thay đổi trong giá trị của biến phụ thuộc nếu như biến độc lập tăng hoặc giảm một đơn vị.
Sai số (ε_i): Là khoảng chênh lệch giữa giá trị thực tế của biến phụ thuộc và giá trị ước lượng thông qua mô hình hồi quy.



Hình 2 : Mô tả thuật toán hồi quy tuyến tính

Hàm mất mát của thuật toán hồi quy tuyến tính

Mục tiêu của tất cả các mô hình học có giám sát (*supervised learning*) trong machine learning là tìm ra một hàm số dự báo mà giá trị của chúng sai khác so với ground truth là nhỏ nhất. Ground truth ở đây chính là giá trị của biến mục tiêu $\boldsymbol{y}$. Sai khác này được đo lường thông qua các hàm mất mát (*loss function*). Huấn luyện mô hình machine learning thực chất là qui về tìm cực trị của hàm mất mát. Tuỳ thuộc vào bài toán mà chúng ta có những dạng hàm mất mát khác nhau.

Trong bài toán dự báo chúng ta sẽ sử dụng hàm MSE (*Mean Square Error*) làm hàm mất mát. Hàm số này có giá trị bằng trung bình của tổng bình phương sai số giữa giá trị dự báo và ground truth. Giả sử chúng ta xét phương trình hồi qui đơn biến gồm n quan sát có biến phụ thuộc là $\mathbf{y} = \{y_1, y_2, \ldots, y_n\}$ và biến đầu vào $\mathbf{x} = \{x_1, x_2, \ldots, x_n\}$.

Véc tơ $\mathbf{w} = (w_0, w_1)$có giá trị w_0, w_1lần lượt là hệ số góc và hệ số ước lượng. Phương trình hồi qui tuyến tính đơn biến có dạng:

$$\hat{y_i} = f(x_i) = w_0 + w_1 * x_i$$

Trong đó (x_i, y_i)là điểm dữ liệu thứ i.

Mục tiêu của chúng ta là đi tìm véc tơ w sao cho sai số giữa giá trị dự báo và thực tế là nhỏ nhất. Tức là tối thiểu hoá hàm mất mát chính là hàm MSE:

$$\mathcal{L}(\mathbf{w}) = \frac{1}{2n}(y_i - \hat{y_i})^2 = \frac{1}{2n}\sum_{i=1}^{n}.(y_i - w_0 - w_1 * x_i)^2$$

Ký hiệu $\mathcal{L}(\mathbf{w})$ thể hiện rằng hàm mất mát là một hàm theo w trong điều kiện ta đã biết đầu vào là véc tơ x và véc tơ biến phụ thuộc $\mathbf{y}$. Ta có thể tìm cực trị của phương trình trên dựa vào đạo hàm theo w_0 và w_1 như sau:

Đạo hàm theo w_0

$$\begin{aligned}
\frac{\delta\mathcal{L}(\mathbf{w})}{\delta w_0} &= \frac{-1}{n}\sum_{i=1}^{n}.(y_i - w_0 - w_1 * x_i) \\
&= \frac{-1}{n}\sum_{i=1}^{n}.y_i + w_0 + w_1\frac{1}{n}\sum_{i=1}^{n}.x_i \qquad (1) \\
&= -\overline{\mathbf{y}} + w_0 + w_1\overline{\mathbf{x}} \\
&= 0
\end{aligned}$$

Đạo hàm theo w_1:

$$\begin{aligned}
\frac{\delta\mathcal{L}(\mathbf{w})}{\delta w_1} &= \frac{-1}{n}\sum_{i=1}^{n}.x_i(y_i - w_0 - w_1 * x_i) \\
&= \frac{-1}{n}\sum_{i=1}^{n}.x_i y_i + w_0\frac{1}{n}\sum_{i=1}^{n}.x_i + w_1\frac{1}{n}\sum_{i=1}^{n}.x_i^2 \qquad (2) \\
&= -\overline{\mathbf{xy}} + w_0\overline{\mathbf{x}} + w_1\overline{\mathbf{x}^2} \\
&= 0
\end{aligned}$$

Từ phương trình (1) ta suy ra: $w_0 = \overline{\mathbf{y}} - w_1\overline{\mathbf{x}}$ .Thế vào phương trình (2) ta tính được:

$$\begin{aligned}
-\overline{\mathbf{xy}} + w_0\overline{\mathbf{x}} + w_1\overline{\mathbf{x}^2} &= -\overline{\mathbf{xy}} + (\overline{\mathbf{y}} - w_1\overline{\mathbf{x}})\overline{\mathbf{x}} + w_1\overline{\mathbf{x}^2} \\
&= -\overline{\mathbf{xy}} + \overline{\mathbf{yx}} - w_1\overline{\mathbf{x}}^2 + w_1\overline{\mathbf{x}^2} \\
&= 0
\end{aligned}$$

Từ đó suy ra:

$$w_1 = \frac{\overline{xy} - \overline{x}\,\overline{y}}{\overline{x}^2 - \overline{x^2}}$$

Sau khi tính được w1 thế vào ta tính được:

$$w_0 = \overline{y} - w_1\overline{x}$$

Đạo hàm bậc nhất bằng 0 mới chỉ là điều kiện cần để w là cực trị của hàm mất mát. Để khẳng định cực trị đó là cực tiểu thì chúng ta cần chứng minh thêm đạo hàm bậc hai lớn hơn hoặc bằng 0 hay hàm số đó là hàm lồi. Điều này khá dễ dàng và mình xin dành cho bạn đọc. Bài tập bên dưới dây sẽ giúp bạn hiểu dễ hơn cách tìm nghiệm của phương trình hồi qui tuyến tính đơn biến.

3. Phương sai trong mô hình hồi quy tuyến tính.

Phân tích phương sai là một kỹ thuật thống kê cơ bản để xác định lần lượt mức độ ảnh hưởng của mô hình và sai số lên biến phụ thuộc. Ở cấp độ nền tảng, bạn cần nắm được những khái niệm cơ bản trong kỹ thuật phân tích phương sai.

Total sum of squares (SST): đo lường tổng mức biến thiên của biến phụ thuộc.

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

Regression sum of squares (RSS): đo lường phần biến thiên của biến phụ thuộc được giải thích bởi biến độc lập trong mô hình ước lượng.

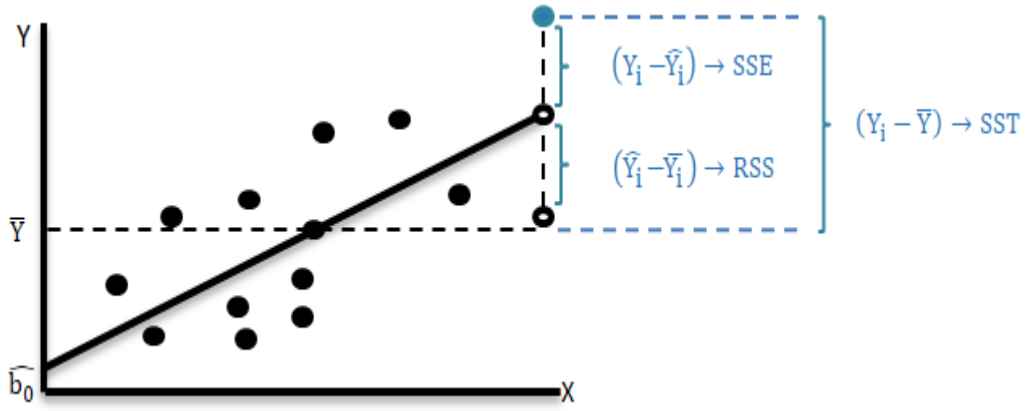
$$RSS = \sum_{i=1}^{n} (\hat{Y}_i - \bar{Y})^2$$

Sum of squared errors (SSE): đo lường phần biến thiên của biến phụ thuộc không được giải thích bởi biến độc lập của mô hình.

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

Total variation = explained variation + unexplained variation; SST = RSS + SSE

Vì vậy, mối quan hệ này có thể được mô tả rõ hơn qua đồ thị sau :



Hình 3 : Mối quan hệ giữa các cấu phần của SST

Kết quả của quá trình phân tích phương sai được biểu diễn trên bảng (ANOVA table) sau đây:

Source of Variation	**Degrees of Freedom**	**Sum of Squares**	**Mean Sum of Squares**
Regression (explained)	1	**RSS**	$\mathbf{MRS = \frac{RRS}{k} = \frac{RSS}{1} = RSS}$
Error (unexplained)	n - 2	**SSE**	$\mathbf{MSE = \frac{SSE}{n-2}}$
Total	n - 1	**SST**	

Hệ số xác định trong mô hình hồi quy tuyến tính

Hệ số xác định (R^2) thể hiện tỷ lệ phần trăm mức biến thiên của biến phụ thuộc có thể được giải thích thông qua biến độc lập. Vì vậy, cũng thể hiện mức độ mạnh yếu của mối quan hệ tuyến tính giữa hai biến.

$$R^2 = \frac{\text{Total variation (SST)} - \text{unexplained variation (SSE)}}{\text{total variation (SST)}}$$

4. Giới thiệu chung và hướng dẫn Google colab

Giới thiệu chung về Google colab :

Colaboratory hay còn gọi là Google Colab, là một sản phẩm từ Google Research, nó cho phép chạy các dòng code python thông qua trình duyệt, đặc biệt phù hợp với

Data analysis, machine learning và giáo dục. Colab không cần yêu cầu cài đặt hay cấu hình máy tính, mọi thứ có thể chạy thông qua trình duyệt, bạn có thể sử dụng tài nguyên máy tính từ CPU tốc độ cao và cả GPUs và cả TPUs đều được cung cấp cho bạn.

IV. Mã lệnh chương trình.

1. Kết nối googledrive

```
from google.colab import drive
drive.mount('/content/drive')
```

2. Khai báo các thư viện cần thiết

```
# Thư viện dùng để đọc dữ liệu từ tệp csv
import pandas as pd
#Thư viện vẽ biểu đồ
import matplotlib as plt
import seaborn as sb
from matplotlib import pyplot as plt
#Thư viện dùng để chia tập dữ liệu
from sklearn.model_selection import train_test_split
# Thư viện dùng để tiền xử lý dữ liệu
from sklearn.preprocessing import LabelEncoder
#Thư viện dùng kiểm thử mô hình
from sklearn.metrics import r2_score
#Thư viện xây dựng mô hình hồi quy tuyến tính
from sklearn.linear_model import LinearRegression
```

3. Đọc dữ liệu từ googledrive

```
data=pd.read_csv('/content/drive/MyDrive/LaptopPricePrediction/dataframe.csv',encoding='latin-1')
#Kiểm tra thử dữ liệu
data.head()
```

	Unnamed: 0	Company	Product	TypeName	Inches	ScreenResolution	Cpu	Ram	Memory	Gpu	OpSys	Weight	Price_euros
0	1	Apple	MacBook Pro	Ultrabook	13.3	IPS Panel Retina Display 2560x1600	Intel Core i5 2.3GHz	8GB	128GB SSD	Intel Iris Plus Graphics 640	macOS	1.37kg	1339.69
1	2	Apple	Macbook Air	Ultrabook	13.3	1440x900	Intel Core i5 1.8GHz	8GB	128GB Flash Storage	Intel HD Graphics 6000	macOS	1.34kg	898.94
2	3	HP	250 G6	Notebook	15.6	Full HD 1920x1080	Intel Core i5 7200U 2.5GHz	8GB	256GB SSD	Intel HD Graphics 620	No OS	1.86kg	575.00
3	4	Apple	MacBook Pro	Ultrabook	15.4	IPS Panel Retina Display 2880x1800	Intel Core i7 2.7GHz	16GB	512GB SSD	AMD Radeon Pro 455	macOS	1.83kg	2537.45
4	5	Apple	MacBook Pro	Ultrabook	13.3	IPS Panel Retina Display 2560x1600	Intel Core i5 3.1GHz	8GB	256GB SSD	Intel Iris Plus Graphics 650	macOS	1.37kg	1803.60

Ta nhận thấy nhiều dữ liệu chưa đồng nhất, cần phải được làm gọn để có thể cho được kết quả tốt nhất.

4. Xử lí dữ liệu thô

4.1. Cột Product

Có một số sản phẩm bị thừa thông tin, kèm theo thông số máy tính (VD: 15-BS101nv (i7-8550U/8GB/256GB/FHD/W10)- những thông số trong ngoặc ở phía sau đã có). Ta cần phải lọc bớt chỉ để lại tên chính của sản phẩm.

Mã xử lí:

```
#Xử lí cột Product:
data['Product'] = data['Product'].str.split('(').apply(lambda x: x[0])
```

Chúng ta đã loại bỏ toàn bộ phần từ dấu ngoặc trở đi vì sau dấu ngoặc cũng chỉ là mã sản phẩm, không cần thiết trong việc phân tích.

4.2. Xử lí cột CPU

Cpu của máy tính thì chúng ta cần quan tâm thế hệ nào ? Nhà sản xuất nào ? Tốc độ bao nhiêu ?

Mã xử lí:

```
#Xử lí cột CPU
#Chia phần Cpu thành 3 thuộc tính nữa là : Speed, Vender, Type
#Speed CPU sẽ ở phía cuối: xGhz
data['Cpu_Speed'] = data['Cpu'].str.split(' ').apply(lambda x: x[-1]).str.replace('GHz', '')
data['Cpu_Speed'] = data['Cpu_Speed'].astype('float')
#Vender là nhà sản xuất: sẽ ở phần đầu:
data['Cpu_Vender'] = data['Cpu'].str.split(' ').apply(lambda x: x[0])
#Type sẽ lấy ở phần giữa, sau khi xử lí sẽ ở dạng list.
data['Cpu_Type'] = data['Cpu'].str.split(' ').apply(lambda x: x[1:-1])
#Chuyển list thành chuỗi
data['Cpu_Type'] = data['Cpu_Type'].apply(lambda x: ' '.join(x))
```

Ta nhận thấy có thể để Cpu_speed ở dạng float để dễ dàng phân tích.

Cột cpu sau khi xử lí đã chia thành Cpu_vender (nhà sản xuất), Cpu_Type (Đời cpu), Cpu_Speed (Tốc độ xử lí).

4.3. Xử lí Gpu

Về thông số Gpu, ta cần quan tâm về nhà sản xuất và loại nào.

Mã xử lí:

```
#Xử lí Gpu
#Chia Gpu thành Gpu vender vs Gpu type
data['Gpu_vender']=data['Gpu'].str.split(' ').apply(lambda x: x[0])
data['Gpu_Type'] = data['Gpu'].str.split(' ').apply(lambda x: x[1:])
data['Gpu_Type']=data['Gpu_Type'].apply(lambda x: ' '.join(x))
```

Sau khi xử lí thì Gpu đã được chia thành Gpu_vender và Gpu_Type.

4.4. Xử lí Ram

Chúng ta cần đưa ram về dạng số để kết quả phân tích chính xác hơn.

Mã xử lí:

```
#Xử lí cột Ram
data['Ram']=data['Ram'].str.replace('GB','')
data['Ram'] = data['Ram'].astype('int')
```

Chuyển ram về dạng interger để thuận tiện phân tích.

4.5. Xử lí cân nặng máy tính

Mã xử lí:

```
#Xử lí cột Weight
data['Weight'] = data['Weight'].str.replace('kg', '')
data['Weight'] = data['Weight'].astype('float')
```

Đưa khối lượng về dạng float.

4.6. Xử lí tính năng màn hình

Mã xử lí:

```
# Xem màn hình có công nghệ ips hay touchscreen không, có=1, không =0
data['Touchscreen'] = data['ScreenResolution'].apply(lambda x:1 if 'Touchscreen' in x else 0)
data['Ips'] = data['ScreenResolution'].apply(lambda x:1 if 'IPS' in x else 0)
```

Công nghệ càng nhiều thì giá laptop càng cao nên chúng ta cần phân loại máy có những công nghệ nào.

4.7. Xử lí hệ điều hành

Mã xử lí:

```
#Xử lí hệ điều hành đưa về mac, window và phần còn lại
def hedieuhanh(x):
  if x == 'macOS' or x=='Mac OS X':
    return 'Mac'
  elif x== 'Windows 10' or x=='Windows 10 S' or x=='Windows 7':
    return 'Windows'
  else:
    return 'Others/No OS/Linux'
data['OpSys'] = data['OpSys'].apply(hedieuhanh)
```

Phân loại về 3 loại chính: Hệ điều hành Windows, Hệ điều hành Mac, Phần còn lại.

4.8. Sắp xếp lại các cột cần phân tích

```
#Sắp xếp và loại đi những cột thừa
data = data.reindex(columns=['Company','Product', 'TypeName', 'Inches', 'Touchscreen','ScreenResolution', 'Ips',
'Cpu_Vender', 'Cpu_Type','Cpu_Speed','Ram', 'Memory', 'Gpu_vender', 'Gpu_Type', 'Weight', 'OpSys', 'Price_euros' ])
```

5. Mô tả qua dữ liệu sau khi xử lí

Sau khi xử lí dữ liệu chúng ta sẽ tiến hành xem qua các dữ liệu đã có đúng dạng mình mong muốn chưa? có bị thiếu xót gì không ?

```
#Kiểm tra kiểu dữ liệu của các cột sau ghi đã xử lý:
data.dtypes
```

```
Company              object
Product              object
TypeName             object
Inches              float64
Touchscreen           int64
ScreenResolution     object
Ips                   int64
Cpu_Vender           object
Cpu_Type             object
Cpu_Speed           float64
Ram                   int64
Memory               object
Gpu_vender           object
Gpu_Type             object
Weight              float64
OpSys                object
Price_euros         float64
dtype: object
```

```
#Thuộc tính của tập dữ liệu
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1303 entries, 0 to 1302
Data columns (total 17 columns):
 #   Column            Non-Null Count  Dtype  
---  ------            --------------  -----  
 0   Company           1303 non-null   object 
 1   Product           1303 non-null   object 
 2   TypeName          1303 non-null   object 
 3   Inches            1303 non-null   float64
 4   Touchscreen       1303 non-null   int64  
 5   ScreenResolution  1303 non-null   object 
 6   Ips               1303 non-null   int64  
 7   Cpu_Vender        1303 non-null   object 
 8   Cpu_Type          1303 non-null   object 
 9   Cpu_Speed         1303 non-null   float64
 10  Ram               1303 non-null   int64  
 11  Memory            1303 non-null   object 
 12  Gpu_vender        1303 non-null   object 
 13  Gpu_Type          1303 non-null   object 
 14  Weight            1303 non-null   float64
 15  OpSys             1303 non-null   object 
 16  Price_euros       1303 non-null   float64
dtypes: float64(4), int64(3), object(10)
memory usage: 173.2+ KB
```

```
# Hiển thị số lượng thuộc tính bị trống
data.isnull().sum()
```

```
Company             0
Product             0
TypeName            0
Inches              0
Touchscreen         0
ScreenResolution    0
Ips                 0
Cpu_Vender          0
Cpu_Type            0
Cpu_Speed           0
Ram                 0
Memory              0
Gpu_vender          0
Gpu_Type            0
Weight              0
OpSys               0
Price_euros         0
dtype: int64
```

Và chúng ta có tệp dữ liệu để phân tích:

```
#Dữ liệu hoàn chỉnh để phân tích
data.head()
```

	Company	Product	TypeName	Inches	Touchscreen	ScreenResolution	Ips	Cpu_Vender	Cpu_Type	Cpu_Speed	Ram	Memory	Gpu_vender	Gpu_Type	Weight	OpSys	Price_euros
0	Apple	MacBook Pro	Ultrabook	13.3	0	IPS Panel Retina Display 2560x1600	1	Intel	Core i5	2.3	8	128GB SSD	Intel	Iris Plus Graphics 640	1.37	Mac	1339.69
1	Apple	Macbook Air	Ultrabook	13.3	0	1440x900	0	Intel	Core i5	1.8	8	128GB Flash Storage	Intel	HD Graphics 6000	1.34	Mac	898.94
2	HP	250 G6	Notebook	15.6	0	Full HD 1920x1080	0	Intel	Core i5 7200U	2.5	8	256GB SSD	Intel	HD Graphics 620	1.86	Others/No OS/Linux	575.00
3	Apple	MacBook Pro	Ultrabook	15.4	0	IPS Panel Retina Display 2880x1800	1	Intel	Core i7	2.7	16	512GB SSD	AMD	Radeon Pro 455	1.83	Mac	2537.45
4	Apple	MacBook Pro	Ultrabook	13.3	0	IPS Panel Retina Display 2560x1600	1	Intel	Core i5	3.1	8	256GB SSD	Intel	Iris Plus Graphics 650	1.37	Mac	1803.60

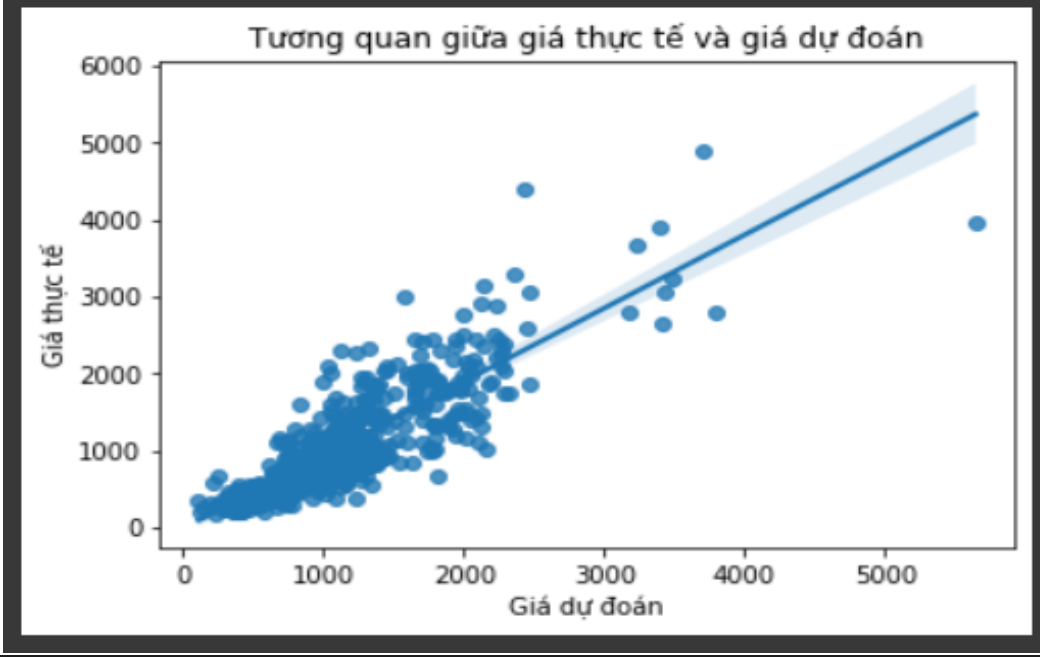
6. Xây dựng mô hình dự đoán giá Laptop

```
# Mã hóa các thuộc tính từ dạng định danh (object) về dạng số, kể cả dạng classification(0 và 1)
lb_make = LabelEncoder()
data['Company'] = lb_make.fit_transform(data['Company'])
data['TypeName'] = lb_make.fit_transform(data['TypeName'])
data['Touchscreen'] = lb_make.fit_transform(data['Touchscreen'])
data['Ips'] = lb_make.fit_transform(data['Ips'])
data['Cpu_Vender'] = lb_make.fit_transform(data['Cpu_Vender'])
data['Cpu_Type'] = lb_make.fit_transform(data['Cpu_Type'])
data['Memory'] = lb_make.fit_transform(data['Memory'])
data['Gpu_vender'] = lb_make.fit_transform(data['Gpu_vender'])
data['Gpu_Type'] = lb_make.fit_transform(data['Gpu_Type'])
data['OpSys'] = lb_make.fit_transform(data['OpSys'])
data['Product'] = lb_make.fit_transform(data['Product'])
data['ScreenResolution'] = lb_make.fit_transform(data['ScreenResolution'])
features=['Company','Product','TypeName','Inches','ScreenResolution','Touchscreen','Ips','Cpu_Vender','Cpu_Type','Cpu_Speed','Ram'
,'Memory','Gpu_vender','Gpu_Type','Weight','OpSys']
target=['Price_euros']
X = data[features]
Y = data[target]
# Chia bộ dữ liệu thành hai tập: train & test (theo tỉ lệ 60% & 40%)
X_train, X_test, Y_train,Y_test = train_test_split(X, Y, test_size=0.4, random_state=110)
```

```
#Xây dựng mô hình LinearRegression
model = LinearRegression()
# Đưa dữ liệu vào huấn luyện mô hình
model.fit(X_train,Y_train)
```

7. Kiểm tra mô hình huấn luyện

```
#Vẽ đồ thị tương quan giữa giá dự đoán và giá thực tế:
def vedothi():
  sb.regplot(x=Y_pred,y=Y_test)
  plt.xlabel ('Giá dự đoán')
  plt.ylabel ('Giá thực tế')
  plt.title('Tương quan giữa giá thực tế và giá dự đoán')
vedothi()
```




```
#Xây dựng mô hình LinearRegression
model = LinearRegression()
# Đưa dữ liệu vào huấn luyện mô hình
model.fit(X_train,Y_train)
# Kiểm thử mô hình trên tập test
Y_pred = model.predict(X_test)
print ('Độ chính xác của mô hình: ', r2_score(Y_pred,Y_test)*100,'%')
```

```
Độ chính xác của mô hình:  63.40943245437538 %
```

8. Tiến hành dự đoán giá

```
'''Dự đoán giá laptop gồm thông tin:
Company:Asus => 2
Product: X553SA-XX031T => 543
TypeName:Notebook => 3
Inches:15.6=> 15.6
Touchscreen:0=> 0
ScreenResolution  1366x768 => 0
Ips:0=> 0
Cpu_Vender:Intel =>1
Cpu_Type:Celeron Dual Core N3050 =>20
Cpu_Speed 1.6=> 1.6
Ram:4 =>4
Memory:500GB SSD => 26
Gpu_vender: Intel => 2
Gpu_Type:HD Graphics=> 42
Weight:2.2 =>2.2
OpSys:Windows =>2
Price_euros:369
'''
mylaptop=[[2,543,3,15.6,0,0,0,1,20,1.6,4,26,2,42,2.2,2]]
price_predict=model.predict(mylaptop)
print ("Giá bán laptop dự đoán là: ",price_predict)
```

```
Giá bán laptop dự đoán là:  [[373.87746133]]
```

V. Kết luận.

Sau khi thực hiện nghiên cứu đề tài: " Dự đoán giá bán của laptop" nhóm em thu được các kết luận như sau:

Nhóm em mới chỉ định giá được tương đối giá laptop qua các thông số cụ thể mà nguồn dữ liệu cung cấp, trên thực tế laptop có rất nhiều các tính năng khác như : vân tay, pin , có hỗ trợ hdd sdd không ? Sản xuất tại nhà máy nào,...

Em tin rằng nếu nhóm em có khối lượng data đủ lớn, nhân lực đủ mạnh, thời gian đủ lâu thì bọn em có thể phân tích chính xác hơn nữa!

Nhưng nếu mà bạn là người muốn mua laptop thì mô hình trên cũng giúp bạn định giá sơ qua laptop bạn muốn sở hữu, không bị mua với giá 'trên trời' khi đi mua laptop.

Ngoài ra, mô hình này có thể giúp cho các cửa hàng kinh doanh laptop có thêm căn cứ định giá những chiếc laptop, từ đó có thể góp thêm phần phát triển chiến thuật kinh doanh, kéo thêm nguồn thu nhập.

1. Ưu điểm

Thuật toán hồi quy tuyến tính cho thất sự nhanh chóng, tiện lợi , chỉ cần có thông số chi tiết là có thể dự đoán gần đúng giá laptop, từ đó có thể tính toán được giá bán sau khi nhập về.

Giúp giảm chi phí nhân công, áp dụng mô hình giúp tiết kiệm một khoản hàng tháng từ đó tăng lợi thế cạnh tranh trên thị trường.

Có thể giúp người dùng tham khảo giá một số laptop phổ biến trên thị trường.

2. Nhược điểm

Vì chỉ áp dụng mô hình hồi quy tuyến tính nên độ chính xác chỉ xấp xỉ 63,4%, nếu có đủ thời gian áp dụng thêm các mô hình dự đoán khác thì có thể cải thiện độ chính xác.

Nguồn dữ liệu đầu vào rất quan trọng, như bài trên bọn em đã phải xử lí nguồn dữ liệu khá nhiều để có thể cải thiện tính chính xác cho mô hình.

Khối lượng dữ liệu chưa đủ lớn, có một số loại chi tiết chỉ xuất hiện 1-2 lần khiến máy tính không chắc chắn khi dự đoán giá máy khi xuất hiện các chi tiết đó.

Tài liệu tham khảo

[1] “ Kiến thức cơ bản về hồi quy tuyến tính “

[2] “Bài giảng Trí tuệ nhân tạo trong kinh doanh, Học viện Ngân hàng”

[3] Slide bài giảng môn học “Trí tuệ nhân tạo trong kinh doanh”

[4] Tham khảo thông tin từ trang web SAPP Knowledge Base

[5] “Nguồn dữ liệu - trang web Kaggle”.