ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH

TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN

KHOA TOÁN – TIN HỌC

BÁO CÁO BÀI TẬP LỚN

Đề tài: Dự đoán giá xe ô tô

Môn: Python cho khoa học dữ liệu

Giảng viên: T.S Nguyễn Tấn Trung

Nhóm: TNT

Tp. Hồ Chí Minh, 23/01/2022

Mục lục

I. Giới thiệu

1. Thông tin nhóm:

Tên nhóm: TNT

STT	Họ tên	MSSV	Email	Ghi chú
1	Đoàn Quang Nhật Tài	19110431	19110431@student.hcmus.edu.vn	Nhóm trưởng
2	Trần Quang Nghĩa	19110392	19110392@student.hcmus.edu.vn	
3	Huỳnh Thị Bảo Trân	19110482	19110482@student.hcmus.edu.vn	

2. Giới thiệu đề tài:

Dự đoán giá xe ô tô.

3. Hướng giải quyết đề tài:

Từ câu hỏi được đặt ra: "Dự đoán về giá xe dựa trên các đặc tính nào của nó?". Với dữ liệu đầu vào là các đặc trưng, tính chất của xe. Nhóm em tiến hành chuẩn bị, phân tích, xử lý dữ liệu để đưa ra kết quả đầu ra là giá xe dự đoán.

Đầu tiên, về việc chuẩn bị dữ liệu, nhóm em tiến hành thu thập dữ liệu. Sau khi crawl dữ liệu, nhóm đã đọc tìm hiểu các thuộc tính. Về sơ bộ, dữ liệu có khoảng 15 dòng xe với hơn 30 thuộc tính liên quan đến từng đặc trưng của xe. Từ đó, tiến hành làm sạch, chuyển đổi tập dữ liệu thô, tiền xử lý dữ liệu (chia các thuộc tính, loại bỏ dữ liệu nhiễu…) sao cho phù hợp bài toán.

Thứ hai, nhóm đã tiến hành xây dựng các mô hình hồi quy tuyến tính để tính toán và xử lý dữ liệu, đưa ra các dự đoán về giá xe. Chi tiết cụ thể, nhóm trình bày ở trong phần III. Xây dựng mô hình.

Cuối cùng, để làm dữ liệu trở nên dễ hiểu và sinh động, nhóm em đã tiến hành trực quan hoá các dữ liệu thông qua từng loại biểu đồ. Đồng thời đưa ra nhận xét về dữ liệu và kết quả dự đoán giá xe ô tô.

4. Bảng phân công công việc:

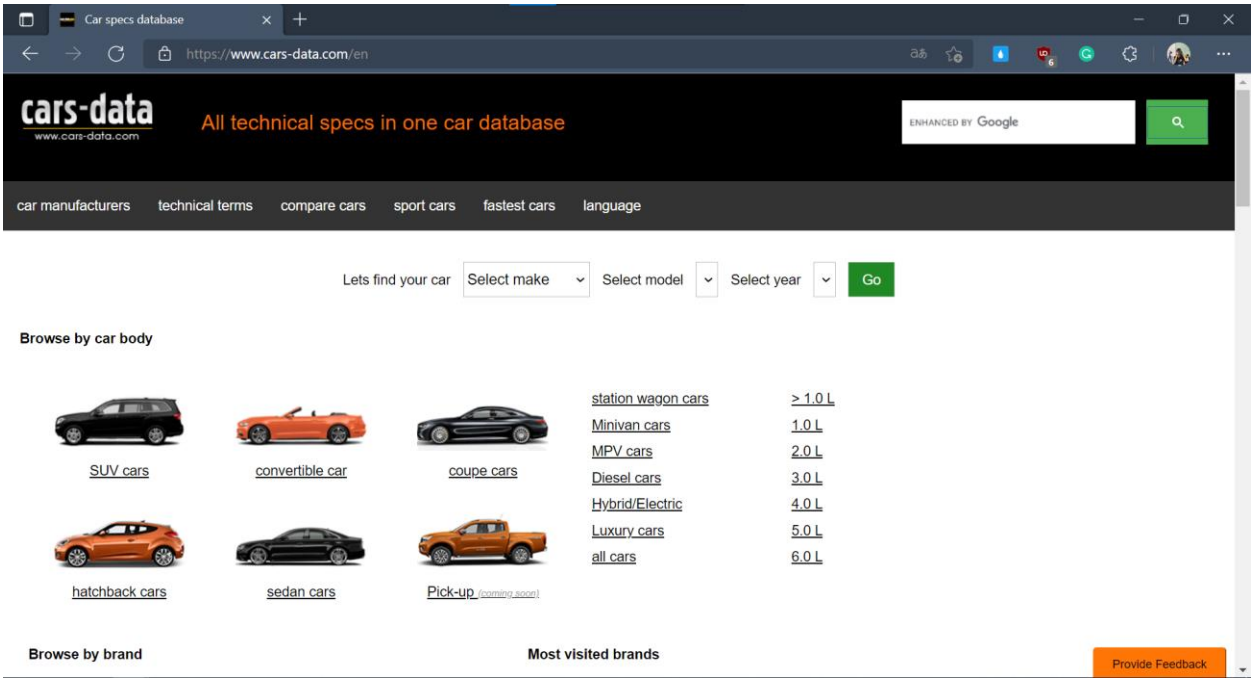
Thời gian	Người thực hiện	Công việc	Kết quả
20/12/2021 (Chọn đề tài)	Nhóm	Thảo luận và chọn đề tài.	Chọn đề tài: "Dự đoán giá xe ô tô".
27/12/2021 (Làm rõ đề tài)	Nhóm	Phân tích các yếu tố xác định, thực hiện bài toán.	Thực hiện crawl dữ liệu bằng Requests, BeautifulSoup. Xây dựng mô hình hồi quy tuyến tính để dự báo. Trực quan dữ liệu bằng biểu đồ.
06/01/2022- 15/01/2022 (Thực hiện đồ án)	Tài	Chuẩn bị dữ liệu: thu thập dữ liệu, chuyển đổi dữ liệu thô.	Crawl được dữ liệu. Đưa ra được file dữ liệu đã xử lý dữ liệu thô.
	Nghĩa	Xây dựng mô hình hồi quy tuyến tính để dự đoán.	Xây dựng được mô hình hồi quy cơ bản.
	Trân	Trực quan hoá dữ liệu trên bằng các loại biểu đồ.	Trực quan được dữ liệu trên một vài biểu đồ.
16/01/2022 (Đánh giá lần 1)	Tài	Viết báo cáo, kiểm tra tiến độ, chuẩn bị demo.	Mức độ hoàn tất là 65% so với lúc đầu đề ra.
	Nghĩa + Trân	Hiệu chỉnh các sai sót ở từng phần thực hiện.	
17/01/2022 (Đánh giá lần 2)	Nhóm	Kiểm tra và cải thiện các lỗi ở đánh giá lần 1. Kết nối các phần lại với nhau.	Cải thiện được các lỗi trong tiền xử lý dữ liệu, chọn được biểu đồ trực quan, mô hình ổn định.
22/01/2022 (Chuẩn bị báo cáo)	Nhóm	Hoàn thành báo cáo quá trình thực hiện. Kiểm tra lần cuối các kết quả model và mức độ hoàn thành.	Hoàn thành xong.

II. Chuẩn bị dữ liệu

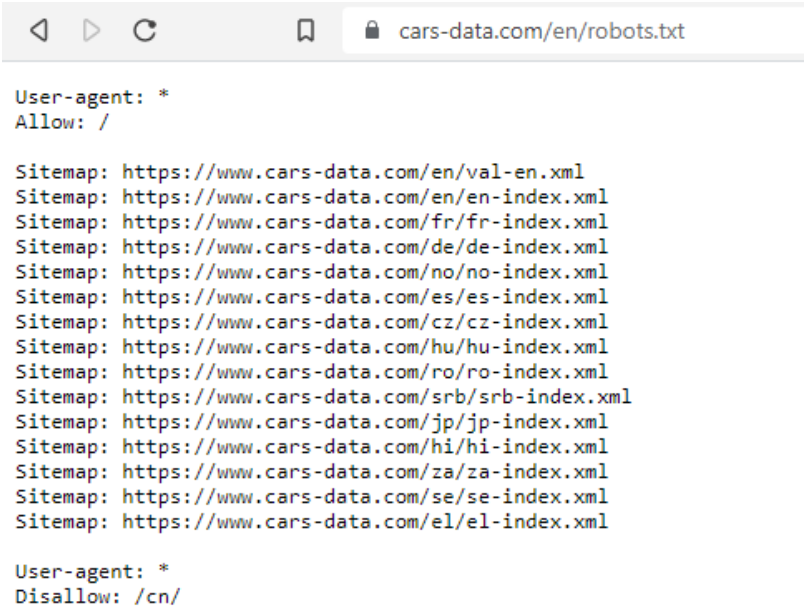
1. Thu thập dữ liệu:

Dữ liệu thu thập trên trang: http://www.cars-data.com/en/.

Trang web có 94 hãng, nhưng để tiết kiệm thời gian, nhóm tiến hành thu thập dữ liệu của 15 hãng đầu tiên để thuận tiện trong việc trực quan và xây dựng các model.



Dữ liệu được thu thập hợp pháp và chính thống trên trang web.



Dữ liệu có khoảng hơn 16.000 dòng (Dữ liệu bao gồm khoảng 34 thuộc tính liên quan đến các đặc điểm của ô tô như phụ tùng, trọng lượng, hãng…).

Xây dựng các hàm lấy thuộc tính như: **get_attributes** và **get_car**. Hàm **get_attributes** dùng để truy cập vào từng lớp của trang web để trích xuất các thông tin/dữ liệu cần thiết, hàm **get_car** thu thập các thông tin/dữ liệu theo từng thuộc tính của từng xe trong hãng.

2. Chuyển đổi dữ liệu thô:

Trước khi xử lý, dữ liệu có dạng như sau:

url	name	model	brand	price	eLabel	bodyType	length	height	width	weight	weightTotal	emissionsCO2	modelDate	fuelType	numberOfAxles	number
https://www.cars-ta.com/en/aiways-u5-standaa...	Aiways U5 Standaard	Aiways	Aiways	39.083	a	5-doors, suv/crossover	4680 mm	1700 mm	1865 mm	NaN	-	0 g/km	2020	NaN	2	
https://www.cars-ta.com/en/aiways-u5-premium...	Aiways U5 Premium	Aiways	Aiways	42.934	a	5-doors, suv/crossover	4680 mm	1700 mm	1865 mm	NaN	-	0 g/km	2020	NaN	2	
https://www.cars-ta.com/en/aiways-u5-showroo...	Aiways U5 Showroom	Aiways	Aiways	40.934	a	5-doors, suv/crossover	4680 mm	1700 mm	1865 mm	NaN	-	0 g/km	2020	NaN	2	
https://www.cars-ta.com/en/subaru-svx-specs/...	Subaru SVX	Subaru	Subaru	53.090	NaN	3-doors, coupé	4625 mm	1300 mm	1770 mm	NaN	2045	#NAME?	1992	gasoline	2	
https://www.cars-ta.com/en/subaru-e-wagon-aw...	Subaru E-wagon AWD	Subaru	Subaru	17.241	NaN	5-doors, mpv	3525 mm	1925 mm	1415 mm	NaN	1600	#NAME?	1993	gasoline	2	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
https://www.cars-data.com/en/tvr-sagaris-specs...	TVR Sagaris	TVR	TVR	134.500	NaN	2-doors, convertible	4060 mm	1175 mm	1770 mm	NaN	-	#NAME?	2006	gasoline	2	

Sau khi đọc dữ liệu thấy rằng có nhiều thuộc tính không cần thiết, bị lỗi crawl không lấy được, nên nhóm đã loại bỏ các thuộc tính không sử dụng ở trên. Đồng thời, nhóm tiến hành tách và bỏ bớt các đơn vị đo của các thuộc tính như kg của weight… để dễ thực hiện phân tích dữ liệu. Cuối cùng, xuất ra file csv để thực hiện các bước tiếp theo nhằm giải quyết bài toán.

Sau khi xử lý dữ liệu thô, dữ liệu có dạng như sau:

	name	brand	price	length	height	width	weightTotal	modelDate	fuelType	numberOfAxles	numberOfDoors	numberOfForwardGears	seatingCapacity	vehicleTran
3	Subaru SVX	Subaru	53.090	4625	1300	1770	2045	1992	gasoline	2	3	4	5	
5	Subaru Impreza 2.5 WRX STi AWD	Subaru	52.285	4465	1440	1740	1900	2005	gasoline	2	4	6	5	Manual tra
6	Subaru Impreza 2.0R AWD	Subaru	26.565	4465	1440	1740	1780	2005	gasoline	2	4	5	5	Manual tra
7	Subaru Impreza 2.0R AWD	Subaru	28.565	4465	1440	1740	1780	2005	gasoline	2	4	4	5	
8	Subaru Impreza 2.5 WRX AWD	Subaru	33.785	4465	1440	1740	1860	2005	gasoline	2	4	5	5	Manual tra
...	...	...	...	...	...	...	...	...	...	...	...	...	...	
	Seat Leon SC 2.0 TDI													

III. Xây dựng mô hình

1. Mô hình hoá:

Để xây dựng được một mô hình tuyến tính trong Python ta cần phân biệt được loại biến, tức là ta cần xác định được đâu là biến định tính và đâu là biến định lượng. Trong **statsmodels**, mô hình bình phương cực tiểu được định nghĩa thông qua hàm **ols** (ordinary least-squares). Các đối số chính của hàm này có dạng:

$$y \sim x_1 + x_2 + \cdots + x_d$$

Trong đó y là biến giải thích và các $x_1, x_2, \ldots, x_d$ là các biến phụ thuộc. Nếu tất cả các biến là định lượng, công thức này mô tả mô hình tuyến tính:

$$Y_i = \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \cdots + \beta_d x_{id} + \varepsilon_i, i = 1, \ldots, n$$

Trong đó x_{ij} là biến phụ thuộc thứ j cho lần quan sát thứ i và sai số ε_i là các biến ngẫu nhiên độc lập sao cho $\mathbb{E}\varepsilon_i = 0$ và $Var\varepsilon_i = \sigma^2$.

Biến phân loại có thể được định nghĩa bằng cách để tên biến được bao bọc trong C().

2. Phân tích:

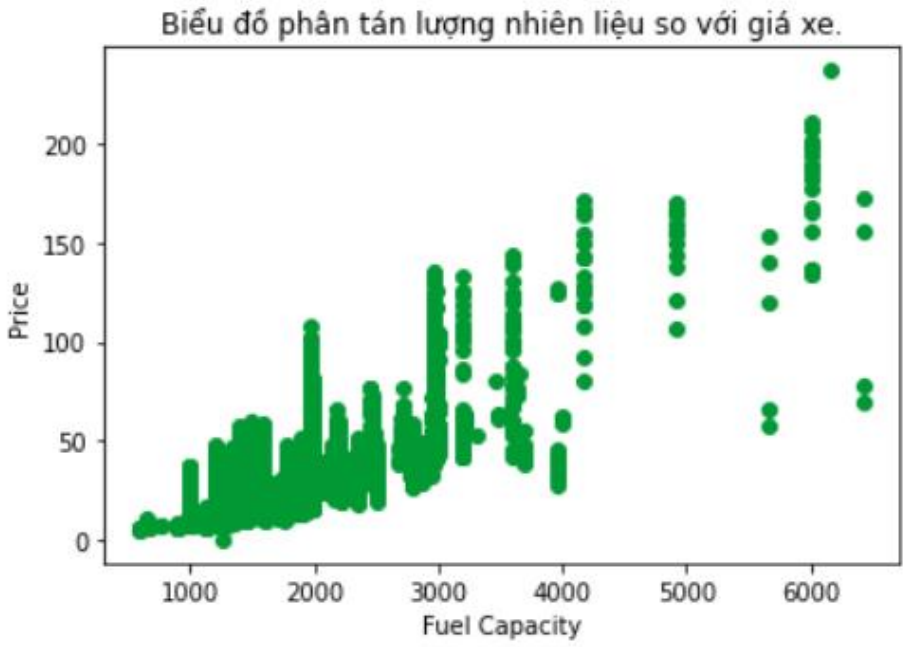
Ta cùng xem xét mô hình hồi quy đơn biến bằng cách dùng tập đã thu thập được cars_data_processed.csv từ phần crawl dữ liệu, trong đó chứa các độ đo như giá xe, lượng nhiên liệu, tốc độ, chiều dài xe, … của khoảng 13000 quan trắc.

Yêu cầu bài toán là ta sẽ điều tra mối quan hệ giữa biến giải thích price (giá xe) và các biến phụ thuộc để kiểm tra xem, giá xe có bị tác động bởi các biến đó hay không. Trong trường hợp này, ta sẽ xét biến phụ thuộc là fuelCapacity (lượng nhiên liệu) để xây dựng một mô hình hồi quy đơn biến và kiểm tra xem lượng nhiên liệu có tác động đến giá xe không.

Trước tiên, ta cần vẽ biểu đồ phân tán để trực quan hóa các dữ liệu bằng lệnh dưới đây:

```
plt.scatter(cars.fuelCapacity, cars.price, c = '#009933') # Vẽ biểu đồ phân tán
plt.ylabel('Price')
plt.xlabel('Fuel Capacity')
plt.title('Biểu đồ phân tán lượng nhiên liệu so với giá xe.')
plt.show()
```

Và kết quả thu được biểu đồ sau:



Dựa vào hình trên, ta có thể thấy rằng hầu hết giá xe sẽ tăng khi lượng nhiên liệu tăng lên, điều này khá dễ để nhận ra vì các điểm dữ liệu gần như là hình dấu sắc.

Ta sẽ tiến hành phân tích dữ liệu thông qua mô hình hồi quy tuyến tính đơn biến:

$$Y_i = \beta_0 + \beta_1 x_i + \varepsilon_i, \qquad i = 1, \dots, n$$

Trong thư viện **statsmodels**, chúng ta sẽ thực hiện phân tích qua hàm **ols** như sau:

```
model = ols('price~fuelCapacity', data = cars) # Xây dựng mô hình
fit = model.fit()
b0, b1 = fit.params
fit.params
```

```
Intercept      -9.823174
fuelCapacity    0.022818
dtype: float64
```

Kết quả trên đưa ra ước lượng bình phương cực tiểu của β_0 và β_1. Trong bài toán này thì ước lượng bình phương cực tiểu là $\widehat{\beta_0} = -9.823$ và $\widehat{\beta_1} = 0.023$.

Ta có thể trình bày một bản tóm tắt các kết quả nhận được với hàm **summary**.

```
print(fit.summary())
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                  price   R-squared:                       0.451
Model:                            OLS   Adj. R-squared:                  0.451
Method:                 Least Squares   F-statistic:                 1.062e+04
Date:                Sun, 23 Jan 2022   Prob (F-statistic):               0.00
Time:                        12:51:36   Log-Likelihood:                -50405.
No. Observations:               12913   AIC:                         1.008e+05
Df Residuals:                   12911   BIC:                         1.008e+05
Df Model:                           1
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
Intercept      -9.8232      0.399    -24.601      0.000     -10.606      -9.041
fuelCapacity    0.0228      0.000    103.063      0.000       0.022       0.023
==============================================================================
Omnibus:                     4249.829   Durbin-Watson:                   0.351
Prob(Omnibus):                  0.000   Jarque-Bera (JB):            24400.273
Skew:                           1.465   Prob(JB):                         0.00
Kurtosis:                       9.063   Cond. No.                     6.82e+03
==============================================================================
```

Các dữ liệu đầu ra quan trọng bao gồm:

- coef: ước lượng tham số của đường hồi quy.
- std err: độ lệch chuẩn của ước lượng đường hồi quy.
- t: là tham số của thống kê theo phân phối Student được tính toán thông qua các giả thuyết H_0: $\beta_i = 0$ và H_1: $\beta_i \neq 0$ với $i = 0,1$.
- P>|t|: là p-value.
- [0.025 0.975]: Khoảng tin cậy 95% cho các tham số.
- R-Squared: Hệ số xác định R^2.
- Adj. R-Squared: Hệ số xác định R^2 hiệu chỉnh.
- F-statistic: thống kê kiểm định F-test. Các bậc tự do *Df Model = 1* và *Df Residuals* $= n - 2$ cũng được hiển thị, p-value cũng được đưa ra tại *Prob (F-statistic)*.

Kết quả cho thấy mối quan hệ tuyến tính giữa biến giải thích price và biến phụ thuộc fuelCapacity (bằng chứng là độ dốc của đường hồi quy bằng 0), vì p-value cho thử nghiệm là rất nhỏ (xấp xỉ 0). Ước lượng của độ dốc chỉ ra rằng sự khác biệt giữa giá xe có lượng nhiên liệu khác nhau xẽ chênh lệch 0.023 (nghìn euro).

Tuy nhiên, dựa vào giá trị của hệ số xác định R^2 ta thấy rằng chỉ có 45.1% sự thay đổi của giá xe được giải thích bởi lượng nhiên liệu. Do đó, ta sẽ thêm các biến phụ thuộc khác vào mô hình để tăng khả năng dự đoán của mô hình.

3. Phân tích phương sai (ANOVA):

Khi thêm nhiều biến phụ thuộc vào ta sẽ gọi đó là mô hình hồi quy đa biến. Tiếp tục với dữ liệu của bài toán, ta sẽ vào đó các biến phụ và đồng thời sẽ phân tích phương sai của mô hình. Thay vì chỉ giải thích thông qua biến fuelCapacity ta sẽ thêm vào đó nhiều biền khác, bao gồm:

C(brand)', 'modelDate', 'C(fuelType)','numberOfForwardGears','C(vehicleTransmission)','cargoVolume', 'accelerationTime', 'speed', 'trailerWeight', 'vEengineType', 'vEengineDisplacement', 'torque'

Ta cũng sẽ vẽ các biểu đồ phân tán điểm giữa giá tiền với các biến này. Để thuận tiện hơn, ta cần xây dựng hàm vẽ biểu đồ như sau:

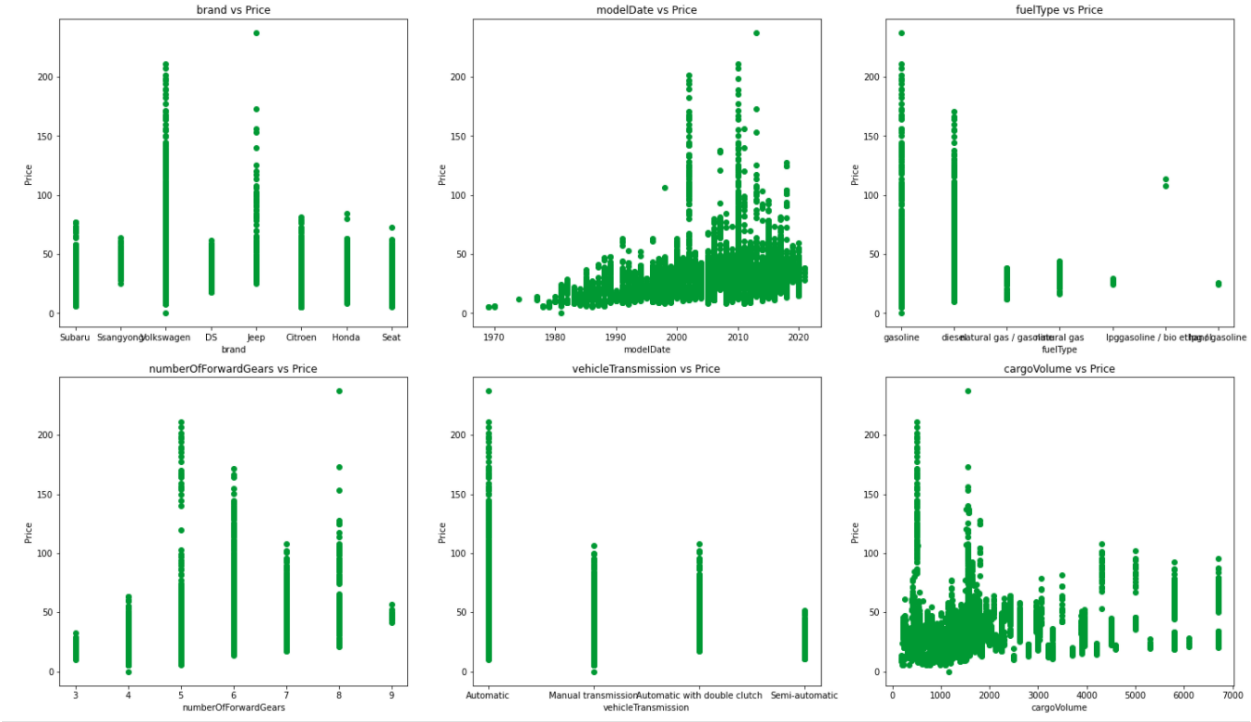
```
def scatter(x,fig): # Xây dựng hàm scatter vẽ biểu đồ
    plt.subplot(7,3,fig)
    plt.scatter(cars[x],cars['price'], c = '#009933')
    plt.title(x+' vs Price')
    plt.ylabel('Price')
    plt.xlabel(x)
```

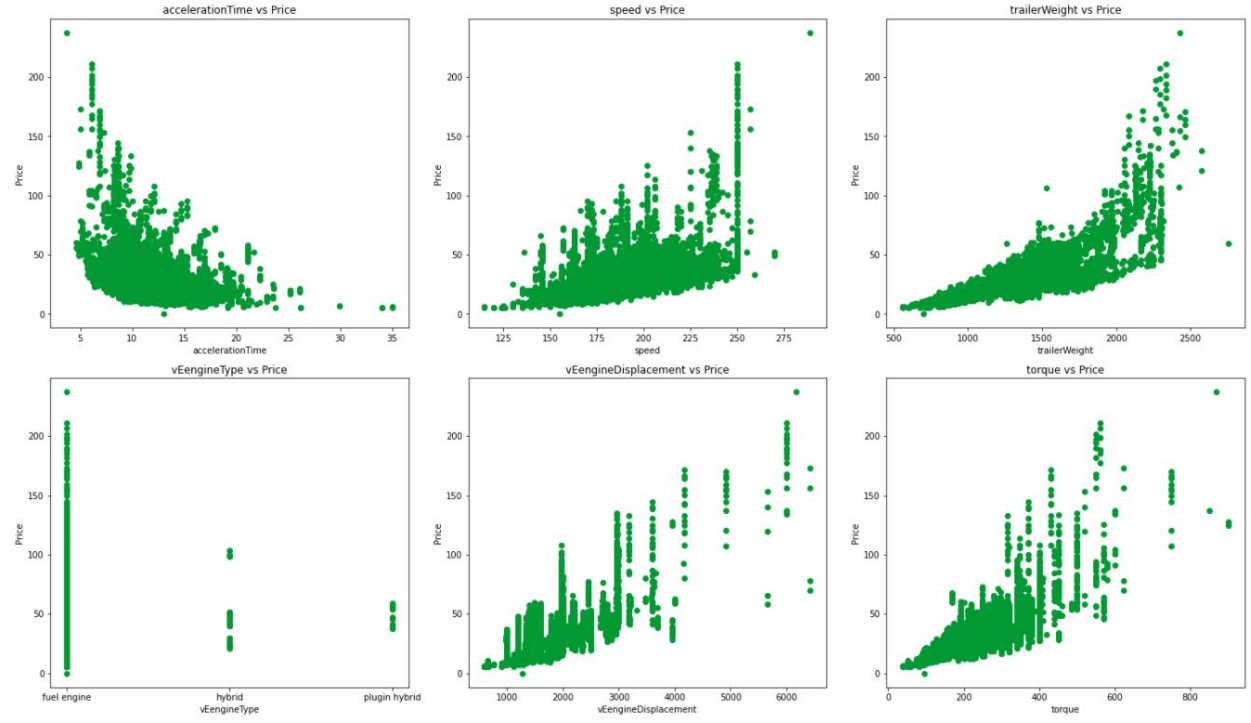
Sau đó, ta gọi hàm và vẽ hình:

```
plt.figure(figsize=(20,40))
scatter('brand', 1)
scatter('modelDate', 2)
scatter('fuelType', 3)
scatter('numberOfForwardGears', 4)
scatter('vehicleTransmission', 5)
scatter('cargoVolume', 6)
plt.tight_layout()
```

```
plt.figure(figsize=(20,40))
scatter('accelerationTime', 1)
scatter('speed', 2)
scatter('trailerWeight', 3)
scatter('vEengineType', 4)
scatter('vEengineDisplacement', 5)
scatter('torque', 6)
plt.tight_layout()
```

Ta thu được các kết quả sau (Trang kế tiếp):





Các kết quả này sẽ hiển thị rõ hơn khi xem trong tệp Code Python.

Dựa vào biểu đồ, ta thu được kết quả là các biến phụ thuộc sau có tác động đến giá xe:

C(brand)', 'modelDate', 'C(fuelType)','numberOfForwardGears','C(vehicleTransmission)','cargoVolume', 'accelerationTime', 'speed', 'trailerWeight', 'vEengineDisplacement', 'torque'

Và ta dùng phương pháp lựa chọn biến bằng cách sau:

```
remaining_features = {'C(brand)', 'modelDate', 'C(fuelType)',
          'numberOfForwardGears','C(vehicleTransmission)',
          'cargoVolume', 'accelerationTime', 'fuelCapacity', 'speed',
          'trailerWeight', 'vEengineDisplacement', 'torque'}
selected_features = []
while remaining_features :
    PF = [] # Danh sách (P value , biến)
    for f in remaining_features :
        temp = selected_features + [f] # các biến tạm thời
        formula = 'price~' + '+'.join(temp)
        fit = ols(formula, data = cars).fit()
        pval = fit.pvalues[-1]
        if pval < 0.05:
            PF.append((pval, f))
    if PF: # Nếu rỗng
        PF.sort(reverse = True)
        (best_pval, best_f) = PF.pop()
        remaining_features.remove(best_f)
        print('feature {} with P-value = {:.2E}'.format(best_f, best_pval))
        selected_features.append(best_f)
    else:
        break
```

Các kết quả thu được như sau:

```
feature accelerationTime with P-value = 0.00E+00
feature C(brand) with P-value = 0.00E+00
feature C(fuelType) with P-value = 0.00E+00
feature C(vehicleTransmission) with P-value = 0.00E+00
feature fuelCapacity with P-value = 0.00E+00
feature modelDate with P-value = 0.00E+00
feature torque with P-value = 0.00E+00
feature vEengineDisplacement with P-value = 0.00E+00
feature trailerWeight with P-value = 3.43E-192
feature cargoVolume with P-value = 2.19E-156
feature numberOfForwardGears with P-value = 2.48E-33
feature speed with P-value = 2.08E-08
```

Gọi $\mathcal{F}$ là tập các biến ta thu được như trên. Như vậy, ta sẽ thêm các biến ấy vào mô hình hồi quy đa biến bằng cách dùng hàm **ols**.

Với mỗi biến giá xe ta có: $price = \beta_0 + \sum_{i=1}^{n} \beta_i x_i + \varepsilon$

Trong đó, ε là sai số và tuân theo phân phối chuẩn với kỳ vọng là 0 và phương sai là σ^2; các $x_i \in \mathcal{F}$.

Ta xây dựng mô hình hồi quy đa biến bằng hàm **ols** và in ra các thông số cần thiết bằng hàm **summary** như sau:

```
# Xây dựng mô hình
anova = ols('price~C(brand)+modelDate+C(fuelType)'
          +'+numberOfForwardGears+C(vehicleTransmission)'
          +'+cargoVolume+accelerationTime+fuelCapacity+speed'
          +'+trailerWeight+vEengineDisplacement+torque', data = cars)
fit = anova.fit()
print(fit.summary())
```

Ta thu được kết quả:

```
==============================================================================
Dep. Variable:                  price   R-squared:                       0.796
Model:                            OLS   Adj. R-squared:                  0.795
Method:                 Least Squares   F-statistic:                     2093.
Date:                Sun, 23 Jan 2022   Prob (F-statistic):               0.00
Time:                        13:32:33   Log-Likelihood:                -44022.
No. Observations:               12913   AIC:                         8.809e+04
Df Residuals:                   12888   BIC:                         8.828e+04
Df Model:                          24
Covariance Type:            nonrobust
==============================================================================================================================
                                                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------------------------------------------------------
Intercept                                   -638.7446     22.289    -28.657      0.000    -682.435    -595.054
C(brand)[T.DS]                                 2.8736      0.589      4.880      0.000       1.719       4.028
C(brand)[T.Honda]                              1.4247      0.314      4.540      0.000       0.810       2.040
C(brand)[T.Jeep]                             -10.9623      0.666    -16.472      0.000     -12.267      -9.658
C(brand)[T.Seat]                              -1.1442      0.219     -5.224      0.000      -1.573      -0.715
C(brand)[T.Ssangyong]                        -12.0071      0.967    -12.415      0.000     -13.903     -10.111
C(brand)[T.Subaru]                            -0.8195      0.419     -1.958      0.050      -1.640       0.001
C(brand)[T.Volkswagen]                         2.2280      0.187     11.913      0.000       1.861       2.595
C(fuelType)[T.gasoline]                        9.1958      0.235     39.180      0.000       8.736       9.656
C(fuelType)[T.gasoline / bio ethanol]         45.0608      5.231      8.615      0.000      34.808      55.314
C(fuelType)[T.lpg]                             9.4687      2.458      3.852      0.000       4.650      14.287
C(fuelType)[T.lpg / gasoline]                  8.7423      4.242      2.061      0.039       0.428      17.057
C(fuelType)[T.natural gas]                     6.8731      0.875      7.859      0.000       5.159       8.587
C(fuelType)[T.natural gas / gasoline]          7.0929      0.885      8.012      0.000       5.358       8.828
C(vehicleTransmission)[T.Automatic with double clutch]     1.1367      0.299      3.797      0.000       0.550       1.724
C(vehicleTransmission)[T.Manual transmission]             -0.6252      0.189     -3.316      0.001      -0.995      -0.256
C(vehicleTransmission)[T.Semi-automatic]                   0.3596      0.471      0.763      0.445      -0.564       1.283
modelDate                                      0.2927      0.011     26.162      0.000       0.271       0.315
numberOfForwardGears                           1.5061      0.125     12.035      0.000       1.261       1.751
cargoVolume                                   -0.0026   9.71e-05    -26.880      0.000      -0.003      -0.002
accelerationTime                               1.0488      0.060     17.575      0.000       0.932       1.166
fuelCapacity                                   0.0055      0.000     41.442      0.000       0.005       0.006
speed                                         -0.0460      0.008     -5.609      0.000      -0.062      -0.030
trailerWeight                                  0.0218      0.001     37.762      0.000       0.021       0.023
vEengineDisplacement                           0.0055      0.000     41.442      0.000       0.005       0.006
torque                                         0.0919      0.002     37.841      0.000       0.087       0.097
==============================================================================
```

Từ bảng trên, ta suy ra ngắn gọn như sau:

- Các biến phụ thuộc có tác động đến giá xe (bằng chứng là các p-value xấp xỉ 0.0 hoặc hầu như không quá 0.05)

- Hệ số xác định $R^2 \approx 80$. Có nghĩa là các biến phụ thuộc trong tập $\mathcal{F}$ đã giải thích gần 80% sự thay đổi của giá xe (p-value xấp xỉ 0.0).

4. Dự đoán:

Trong thư viện **statsmodels**, một phương thức dùng để tính toán khoảng tin cậy và dự đoán ta dùng hàm **get_prediction**. Hay đơn giản hơn là hàm **predict**, chỉ trả về các giá trị dự đoán (độ tin cậy 95%). Biến giải thích được nhạp dưới dạng dictionary.
Giả sử ta muốn dự đoán giá của một chiếc xe có các giá trị của các thuộc tính như sau:

```
x = {'brand': ['Subaru'], 'modelDate': [2005],
          'fuelType': ['gasoline'], 'numberOfForwardGears': [5], 'vehicleTransmission': ['Automatic'],
          'cargoVolume': [460.0], 'accelerationTime': [10.5], 'fuelCapacity': [3000], 'speed': [210],
          'trailerWeight': [1300], 'vEengineDisplacement': [2500], 'torque': [350]} # input dữ liệu bằng dictionary
pred = fit.get_prediction(x)
pred.summary_frame(alpha = 0.05).unstack()
```

Và trả về các giá trị dự đoán bằng hàm **get_prediction** thì ta thu được kết quả sau:

```
mean           0    55.041550
mean_se        0     0.510611
mean_ci_lower  0    54.040677
mean_ci_upper  0    56.042422
obs_ci_lower   0    40.650338
obs_ci_upper   0    69.432762
dtype: float64
```

Ở hàng mean ta sẽ thu được giá trị dự đoán của giá xe: 55.042 (nghìn euro, tức là xấp xỉ 55 042 euro). Và có khoảng tin cậy là: [54.041;56.042]

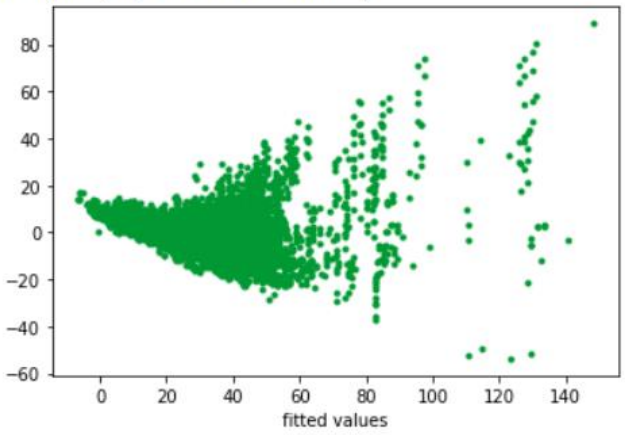
5. Xác thực mô hình:

Ta có thể thực hiện phân tích các sai số để kiểm tra xem liệu các giả định cơ bản của mô hình hồi quy tuyến tính có được xác định hay không. Có thể sử dụng biểu đồ sai số khác nhau để kiểm tra xem các giả định về sai số $\{\varepsilon_i\}$ có thỏa mãn hay không.

Ta dùng biểu đồ phân tán các sai số $\{\varepsilon_i\}$ so với các giá trị được điều chỉnh $\hat{y}_i$. Khi các giả định của mô hình là hợp lệ, các sai số sẽ được xem như là các biến ngẫu nhiên thông thường đối với mỗi giá trị phù hợp và phương sai không đổi. Trong trường hợp này, các sai số gần như đối xứng qua đường thẳng y = 0 (không được hiển thị).

```
plt.plot(fit.fittedvalues, fit.resid, '.', c = '#009933')
plt.xlabel('fitted values')
```

```
Text(0.5, 0, 'fitted values')
```



IV. Trực quan hoá dữ liệu

1. Quá trình:

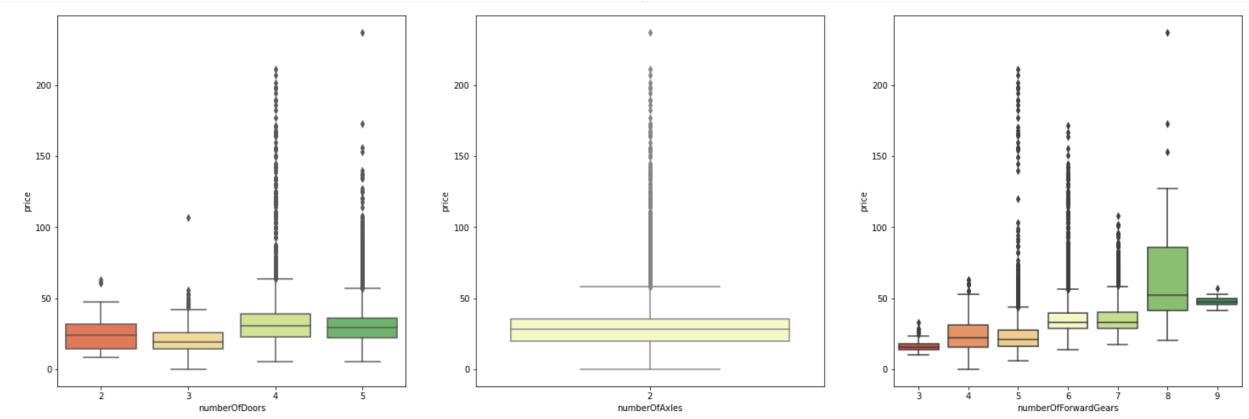
Đọc hiểu các thuộc tính và số liệu sau khi đã phân tích và tiến hành trực quan, trình bày các dữ liệu thành các biểu đồ. Đồng thời đưa ra các nhận xét, kết luận về dữ liệu đã có để có cái nhìn trực quan hơn.

2. Các biểu đồ:

Nhóm đã sử dụng thư viện **seaborn** để vẽ các biểu đồ thông qua hàm **boxplot**, **barplot** và **distplot** như sau:

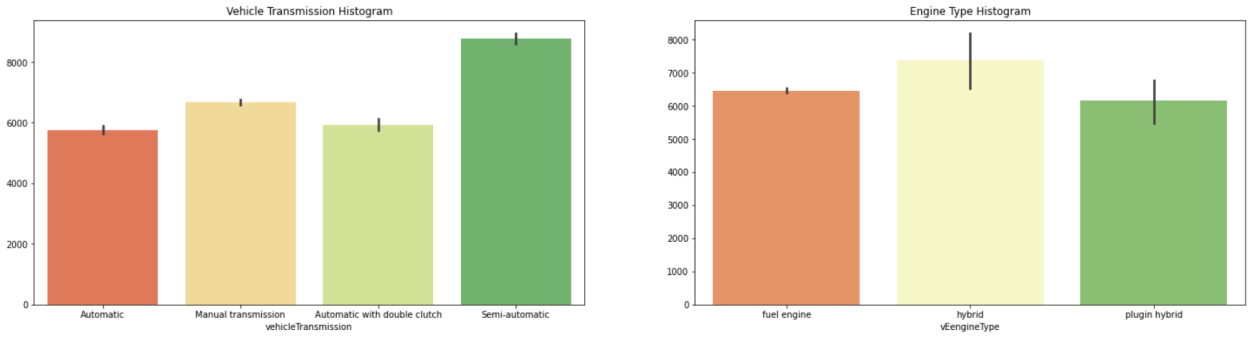
- Biểu đồ boxplot: biểu hiện các dữ liệu ngoại lai theo từng thuộc tính.

```
sns.boxplot(x = 'numberOfDoors', y = 'price', data = dataset,
            palette = 'RdYlGn')
```



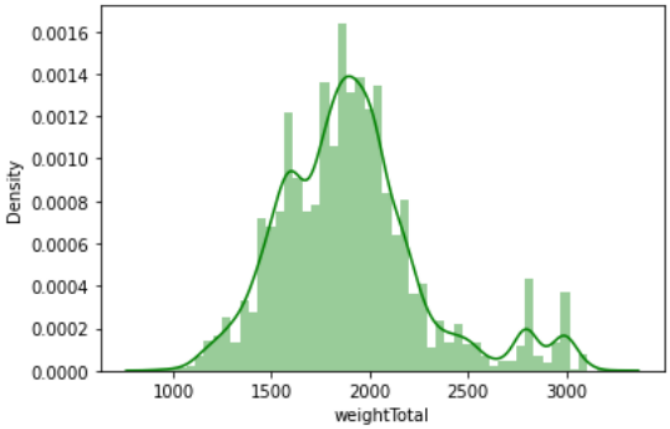
- Biểu đồ cột: biểu hiện sự tương quan giữa các thuộc tính với nhau.

```
sns.barplot(x = 'vEengineType', y = data_vEengine.index,
            data = data_vEengine, palette = 'RdYlGn')
```



- Biểu đồ distplot: biểu hiện mức phân bố của các thuộc tính trong dữ liệu.

```
sns.distplot(dataset['weightTotal'], color = 'g')
```



Chi tiết cụ thể, nhóm đã trình bày trong file code Python.

V. Tài liệu tham khảo

Tài liệu: Data Science and Machine Learning, Mathematical and Statistical Methods (Dirk P. Kroese, Zdravko I. Botev, Thomas Taimre, Radislav Vaisman)

Website: seaborn: statistical data visualization — seaborn 0.11.2 documentation (pydata.org)

Website: Car Price Prediction (Linear Regression - RFE) | Kaggle