TRƯỜNG ĐẠI HỌC BÁCH KHOA

KHOA CÔNG NGHỆ THÔNG TIN

TIỂU LUẬN CUỐI KỲ

HỌC PHẦN: KHOA HỌC DỮ LIỆU

DỰ ĐOÁN GIÁ XE

HỌ VÀ TÊN SINH VIÊN	LỚP HỌC PHẦN	ĐIỂM BẢO VỆ
Đỗ Nguyên Ánh	19N13	
Nguyễn Trung Hiếu	19N13	
Trương Công Thái	19N13	

ĐÀ NẴNG, 06/2022

TÓM TẮT

BẢNG PHÂN CÔNG NHIỆM VỤ

Sinh viên thực hiện	Các nhiệm vụ	Tự đánh giá theo 3 mức (Đã hoàn thành/Chưa hoàn thành/Không triển khai)
Đỗ Nguyên Ánh	- Thu thập dữ liệu (crawl) - Thống kê tổng quan về tập dữ liệu và xuất ra các thống kê mô tả trực quan về các đặc trưng	-Đã hoàn thành
Nguyễn Trung Hiếu	- Lựa chọn đặc trưng, làm sạch và chuẩn hóa dữ liệu, giảm chiều dữ liệu, … - Trực quan hoá	-Đã hoàn thành
Trương Công Thái	- Mô hình hóa dữ liệu - Đánh giá và kết luận mô hình	-Đã hoàn thành

MỤC LỤC

1. Giới thiệu

Trong phần này, chúng em xin giới thiệu về đề tài "Dự đoán giá ô tô" dựa trên kết quả thu được từ trang web http://bonbanh/. Dữ liệu thu về là một bảng data gồm 3640 hàng và gồm 11 cột. Dữ liệu mới thu thập có 11 đặc trưng, gồm có các cột id, year, new or old, Address, Production Origin, Color, Car Gearbox, Used, Company, Fuel Type, Price.

Các đặc trưng lần lượt thể hiện năm bán, loại xe cũ hay là mới, Địa chỉ bán, nơi xe sản xuất, số tay tự động hay là không, số kilomet đã đi, Tên công ty, Loại nhiên liệu xe chạy và cuối cùng là giá xe.

Nhìn vào bức tranh tổng thể.

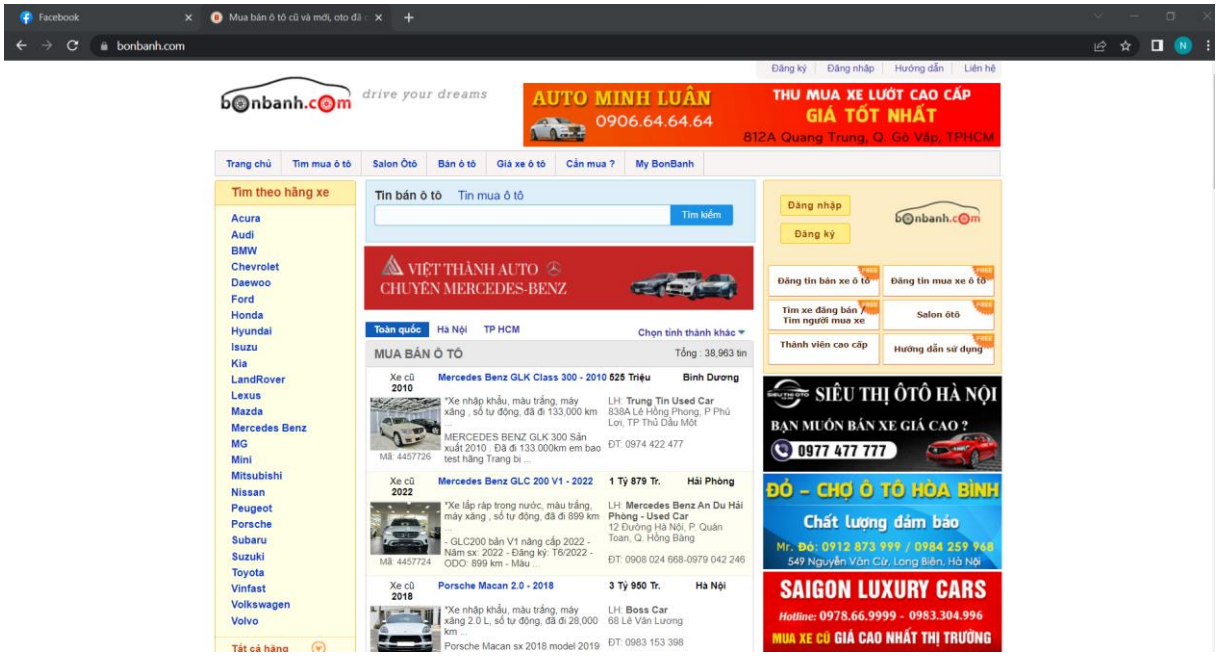
Nhiệm vụ đầu tiên là sử dụng tập dữ liệu được crawl từ trang web để xây dựng một mô hình dự đoán giá ô tô. Mô hình của ta sẽ cần học từ tập dữ liệu này và dự đoán giá xe dựa trên tất cả số liệu đã cho.

2. Thu thập và mô tả dữ liệu

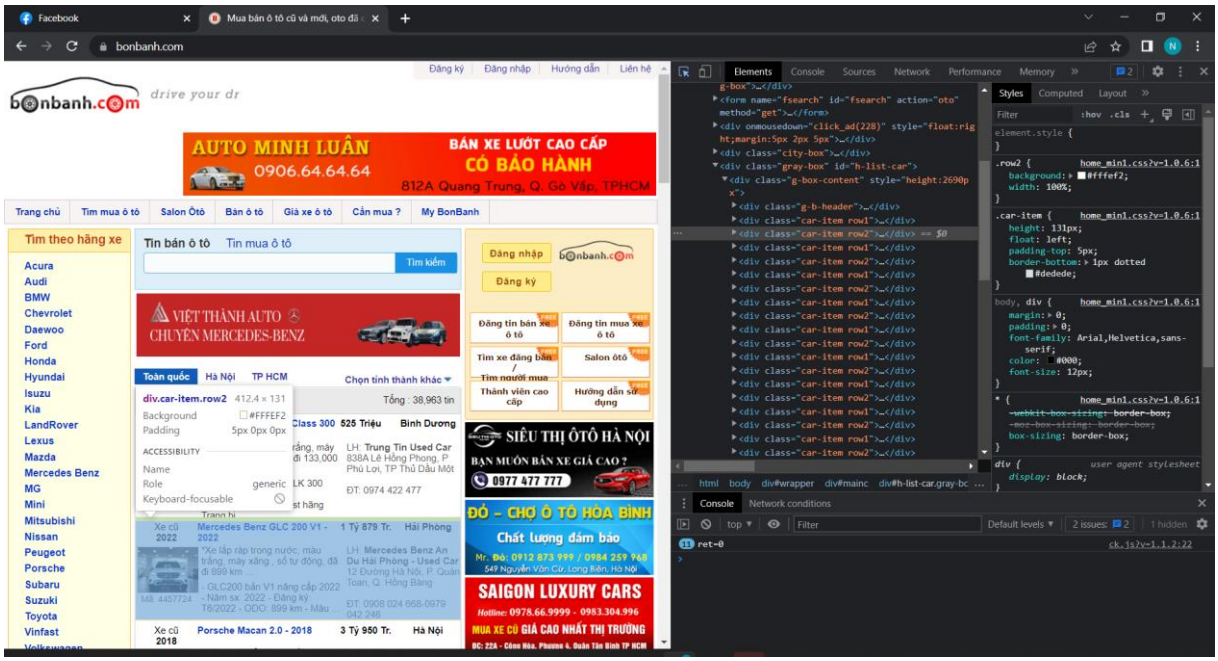
2.1. Thu thập dữ liệu

Link: https://bonbanh.com/

Website:



Bước đầu tiên em sẽ Inspect vào Website này và tìm đến thẻ có chứa dữ liệu cần thu thập trong file html để tìm ra phương pháp crawl dữ liệu



Sau đó em sẽ sử dụng python để code crawl dữ liệu về:

Import thư viện để có thể lấy file html của trang web và xử lý để lấy các dữ liệu cần thiết

```
import requests
from bs4 import BeautifulSoup
```

Code xử lý: lấy file html của trang web và xử dụng phương thức find() và find_all() để tìm đến thẻ cần lấy dữ liệu, cắt chuỗi cần thiết từ thuộc tính text trong thẻ đó và ghi vào file crawl_data.csv.

```
link_base = "https://bonbanh.com/oto/page,"
list_link = []

with open("craw_data.csv", "w", encoding='utf_8') as file:
    # Ghi tiêu đề cho file csv
    file.write("Id,Year,New Or Old,Name,Address,Production Origin,Color,Fuel,Car Gearbox,Used (kilomet),Price\n")
    # Tạo link truy cập
    for index in range(1,500):
        item = link_base + str(index)
        list_link.append(item)

    # Kiểm tra xem bạn có phải là robot không
    headers = {'User-Agent': 'Mozilla/5.0 (Macintosh; Intel Mac OS X 10_10_1) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/39.0.2171.95 Safari/537.36'}
    data = []
    for i in range(len(list_link)):

        link = list_link[i]
        # Sau khi đã có link thì mình sẽ requests về cái trang web đó
        response = requests.get(link, headers=headers)
        # Xử lý file html khi đã lấy từ website về
        soup = BeautifulSoup(response.text, 'html.parser')
        data = soup.find('div', attrs={'class':'g-box-content'}).find_all('a', attrs={'itemprop':'url'})
        #lấy thông tin
        for j in range(1,20):
            try:
                # Lấy dữ liệu từ các thẻ div chứa dữ liệu cần lấy bằng cách cắt chuỗi từ các thuộc tính text trong thẻ
                new_or_old = data[j].find_all('div')[0].text[:-5]
```

```
            # Lấy dữ liệu từ các thẻ div chứa dữ liệu cần lấy bằng cách cắt chuỗi từ các thuộc tính text trong thẻ
            new_or_old = data[j].find_all('div')[0].text[:-5]

            year = data[j].find_all('div')[0].text[-5:]

            name = data[j].find_all('div')[1].text[:-6]

            price = data[j].find_all('div')[2].text

            address = data[j].find_all('div')[3].text

            id = data[j].find_all('div')[4].text[6:]

            production_origin = data[j].find_all('div')[5].text.split('...')[0][2:].split(', ')[0]

            color = data[j].find_all('div')[5].text.split('...')[0][2:].split(', ')[1]

            fuel = data[j].find_all('div')[5].text.split('...')[0][2:].split(', ')[2]

            car_gearbox = data[j].find_all('div')[5].text.split('...')[0][2:].split(', ')[3]

            if(data[j].find_all('div')[0].text[:-5].strip() == 'Xe cũ'):

                used = data[j].find_all('div')[5].text.split('...')[0][2:].split(', ')[4].replace(',','').split(' ')[2]
            else:
                used = "0"
        except:
            break
        # Mỗi lần lấy được dữ liệu sẽ được lưu vào biến srt1
        srt1 = f"{str(id.strip())},{str(year.strip())},{str(new_or_old.strip())},\"{str(name.strip())}\",{str(address.strip())},{str(production_origin.stri
        # Ghi str1 vào file craw_data.csv
        file.write(srt1)
file.close()
```

2.2. Mô tả dữ liệu

2.2.1 Sau khi crawl dữ liệu rồi thì việc đầu tiên là đọc file data

```
df = pd.read_csv('craw_data.csv',sep=',', engine='python',encoding='utf_8')
df.head()
```

Python

	Id	Year	New Or Old	Name	Address	Production Origin	Color	Fuel	Car Gearbox	Used (kilomet)	Price
0	4132061	2022	Xe mới	Mercedes Benz GLE Class GLE 450 4Matic	Hà Nội	Xe nhập khẩu	màu đỏ	máy xăng 3.0 L	số tự động	0	4 Tỷ 359 Triệu
1	3864921	2022	Xe mới	Mercedes Benz GLC 300 4Matic	Hà Nội	Xe lắp ráp trong nước	màu đỏ	máy xăng 2.0 L	số tự động	0	2 Tỷ 398 Triệu
2	4147049	2022	Xe mới	Mercedes Benz GLC 300 4Matic	Hà Nội	Xe lắp ráp trong nước	màu đen	máy xăng 2.0 L	số tự động	0	2 Tỷ 398 Triệu
3	4248075	2019	Xe cũ	Lexus RX 300	Hà Nội	Xe nhập khẩu	màu đen	máy xăng 2.0 L	số tự động	10000	3 Tỷ 80 Triệu
4	4294679	2022	Xe mới	Toyota Veloz Cross Top 1.5 CVT	Hà Nội	Xe nhập khẩu	màu trắng	máy xăng 1.5 L	số tự động	0	688 Triệu

Hình 1: Data

2.2.2 Đưa ra thông tin của các cột

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3640 entries, 0 to 3639
Data columns (total 11 columns):
 #   Column             Non-Null Count  Dtype 
---  ------             --------------  ----- 
 0   Id                 3640 non-null   int64 
 1   Year               3640 non-null   int64 
 2   New Or Old         3640 non-null   object
 3   Name               3640 non-null   object
 4   Address            3640 non-null   object
 5   Production Origin  3640 non-null   object
 6   Color              3640 non-null   object
 7   Fuel               3640 non-null   object
 8   Car Gearbox        3640 non-null   object
 9   Used (kilomet)     3640 non-null   int64 
 10  Price              3640 non-null   object
dtypes: int64(3), object(8)
memory usage: 312.9+ KB
```

Hình 2: Thông tin các cột của data

Chú thích:

-Theo thông tin từ ảnh trên thì cái bảng này gồm 11 cột và có 3640 hàng

- Thông tin các cột
- Id: Là id của xe
- Year: năm sản xuất xe
- New or old: là tình trạng xe mới hay là cũ
- Name: Tên xe
- Address :Địa chỉ
- Production Origin: Nguồn gốc sản xuất
- Color : màu sắc
- Fuel:Nhiên liệu sử dụng
- Car Gaerbox: loại hộp số ô tô
- Used(kilomet): số km xe đã đi đc

2.2.3 Thống kê thông số cơ bản

```
df.describe()
```

	Id	Year	Used (kilomet)
count	3.640000e+03	3640.000000	3.640000e+03
mean	4.153745e+06	2018.516209	3.074837e+04
std	4.565672e+05	4.462141	5.824055e+04
min	5.598400e+04	1990.000000	0.000000e+00
25%	4.151636e+06	2016.000000	0.000000e+00
50%	4.311742e+06	2020.000000	1.000000e+04
75%	4.378078e+06	2022.000000	5.000000e+04
max	4.411163e+06	2022.000000	1.502479e+06

Hình 3: Một số thông số cơ bản

	Id	Year	New Or Old	Name	Address	Production Origin	Color	Fuel	Car Gearbox	Used (kilomet)	Price
0	4132061	2022	Xe mới	Mercedes Benz GLE Class GLE 450 4Matic	Hà Nội	Xe nhập khẩu	màu đỏ	máy xăng 3.0 L	số tự động	0	4 Tỷ 359 Triệu
1	3864921	2022	Xe mới	Mercedes Benz GLC 300 4Matic	Hà Nội	Xe lắp ráp trong nước	màu đỏ	máy xăng 2.0 L	số tự động	0	2 Tỷ 398 Triệu
2	4147049	2022	Xe mới	Mercedes Benz GLC 300 4Matic	Hà Nội	Xe lắp ráp trong nước	màu đen	máy xăng 2.0 L	số tự động	0	2 Tỷ 398 Triệu
3	4248075	2019	Xe cũ	Lexus RX 300	Hà Nội	Xe nhập khẩu	màu đen	máy xăng 2.0 L	số tự động	10000	3 Tỷ 80 Triệu
4	4294679	2022	Xe mới	Toyota Veloz Cross Top 1.5 CVT	Hà Nội	Xe nhập khẩu	màu trắng	máy xăng 1.5 L	số tự động	0	688 Triệu

Hình 4: Data

```
print(set(df['Fuel']))
```
✓ 0.1s Python

```
{'máy xăng 3.4 L', 'máy dầu 2.8 L', 'máy xăng 5.4 L', 'máy xăng 1.0 L', 'máy dầu 7.8 L', 'máy xăng 2.0 L', 'máy xăng 5.7 L', 'máy xăng 1.4 L', 'xe hybrid 1.8 L', 'máy xăng 2.6 L', 'xe điện', 'máy xăng 2.2 L', 'máy dầu 1.7 L', 'máy xăng 3.5 L', 'máy dầu 3.2 L', 'máy xăng 4.2 L', 'máy xăng 2.7 L', 'máy dầu 2.4 L', 'xe hybrid 3.0 L', 'máy dầu 2.2 L', 'máy xăng 2.9 L', 'xe hybrid 2.0 L', 'máy xăng 6.7 L', 'xe điện 0.2 L', 'máy dầu 11.1 L', 'xe hybrid 2.5 L', 'máy dầu 2.1 L', 'máy dầu', 'máy xăng 5.5 L', 'máy xăng 1.6 L', 'máy xăng 4.6 L', 'máy xăng 1.3 L', 'máy xăng 3.7 L', 'máy xăng 2.4 L', 'máy xăng 5.6 L', 'máy dầu 1.9 L', 'máy xăng 4.5 L', 'máy xăng 0.9 L', 'máy xăng 6.0 L', 'máy xăng 1.5 L', 'máy dầu 2.5 L', 'máy xăng 3.8 L', 'máy xăng 1.8 L', 'xe hybrid 3.5 L', 'máy xăng 3.0 L', 'máy xăng 6.6 L', 'máy xăng 3.2 L', 'máy xăng 6.8 L', 'máy xăng 4.8 L', 'máy dầu 2.0 L', 'máy xăng 4.7 L', 'máy xăng 0.8 L', 'xe điện 0.1 L', 'máy xăng 2.5 L', 'máy xăng 3.6 L', 'máy xăng 1.1 L', 'máy xăng', 'máy xăng 1.25 L', 'máy xăng 5.0 L', 'máy xăng 3.3 L', 'máy xăng 1.2 L', 'máy dầu 3.0 L', 'xe hybrid', 'máy xăng 6.2 L', 'máy xăng 2.3 L', 'máy xăng 4.0 L'}
```

Hình 5: Trích xuất dữ liệu cột Fuel

Nhận xét: Nhận thấy có dữ liệu có 2 phần:

- Loại máy
- Dung tích

2.2.4 Từ hai cột " Name " và " Fuel ", nhận thấy có khả năng chia thêm cột

```
for i in range(df.shape[0]):
    #tách company từ cột Name
    df.at[i, 'Company'] = df['Name'][i].split(' ')[0]

    #tách Fuel Type và Machine Capacity (L)
    df.at[i, 'Fuel Type'] = df['Fuel'][i].split(' ')[0] + ' ' + df['Fuel'][i].split(' ')[1]
    if(len(df['Fuel'][i].split(' ')[:]) == 4):
        df.at[i, 'Machine Capacity (L)'] = df['Fuel'][i].split(' ')[2]
    elif(len(df['Fuel'][i].split(' ')[:]) == 2):
        df.at[i, 'Machine Capacity (L)'] = None
```

*Hình **6** : Code chia thêm cột từ cột "Name" và "Fuel"*

Cột “ Name ” chia thêm cột “ company ” và cột “Fuel” thêm “ Fuel Type ” và “ Machine Capacity(L) “

Kết quả:

	Id	Year	New Or Old	Name	Address	Production Origin	Color	Fuel	Car Gearbox	Used (kilomet)	Price	Company	Fuel Type	Machine Capacity (L)
3635	3298640	2022	Xe mới	Suzuki Super Carry Truck 1.0 MT	Hà Nội	Xe lắp ráp trong nước	màu xanh	máy xăng 1.0 L	số tay	0	238 Triệu	Suzuki	máy xăng	1.0
3636	4184814	2016	Xe cũ	Porsche Cayenne S	Hà Nội	Xe nhập khẩu	màu nâu	máy xăng 3.6 L	số tự động	52000	4 Tỷ 150 Triệu	Porsche	máy xăng	3.6
3637	3394496	2022	Xe mới	Suzuki Ertiga Sport 1.5 AT	Hưng Yên	Xe nhập khẩu	màu trắng	máy xăng 1.5 L	số tự động	0	558 Triệu	Suzuki	máy xăng	1.5
3638	4376164	2019	Xe cũ	Toyota Yaris 1.5G	Hà Nội	Xe nhập khẩu	màu đỏ	máy xăng 1.5 L	số tự động	46000	610 Triệu	Toyota	máy xăng	1.5
3639	3281968	2022	Xe mới	VinFast Lux A 2.0 Cao cấp	Hà Nội	Xe lắp ráp trong nước	màu nâu	máy xăng 2.0 L	số tự động	0	944 Triệu	VinFast	máy xăng	2.0

***Hình 7**: Data sau khi được thêm cột*

2.2.5 Missing data

```
df.isnull().sum()
```

```
Id                      0
Year                    0
New Or Old              0
Name                    0
Address                 0
Production Origin       0
Color                   0
Fuel                    0
Car Gearbox             0
Used (kilomet)          0
Price                   0
Company                 0
Fuel Type               0
Machine Capacity (L)    81
dtype: int64
```

Hình 6:Số dữ liệu trống của data

```
print(set(df['Machine Capacity (L)']))
```

0.1s Pytho

```
{nan, 1.5, 2.0, 3.0, 1.25, 2.4, 1.2, 1.6, 1.8, 1.3, 1.4, 2.9, 3.5, 5.7, 2.5, 1.0, 7.8, 11.1, 4.0, 4.5, 5.0, 5.5, nan, nan, nan, nan, nan, nan, nan, nan,
nan, nan, nan, nan, nan, nan, nan, nan, nan, nan, nan, nan, 1.9, 3.4, 0.2, 5.4, nan, nan, nan, nan, nan, nan, nan, nan, nan, 6.0, nan, nan, nan, nan,
nan, nan, nan, nan, nan, 2.3, 2.8, 3.8, 3.3, 0.8, 4.8, 6.8, 0.9, nan, nan, nan, nan, nan, nan, nan, nan, nan, nan, 1.7, 0.1, nan, nan, nan, nan, nan, nan,
nan, nan, nan, 2.2, 2.7, 3.2, 3.7, 4.7, 4.2, nan, nan, nan, nan, nan, nan, 6.7, 6.2, nan, nan, nan, nan, nan, nan, nan, nan, nan, nan, nan, nan, 4.6, 5.6,
6.6, 1.1, 2.1, 2.6, 3.6, nan, nan, nan, nan}
```

Hình 7: Dữ liệu cột Machine Capacity(L)

-Nhận xét : Cột " Machine Capacity(L) " còn trống 81 hàng

2.2.6 Xử lý dữ liệu trống bằng phương pháp thay thế bằng giá trị ngẫu nhiên

- Kỹ thuật này cũng giả định rằng dữ liệu trống hoàn toàn ngẫu nhiên
- Thay thế dữ liệu trống bằng các giá trị ngẫu nhiên của cột tương ứng
- Dùng hàm dropna() của đối tượng DataFrame để bỏ qua các NaN values

```
# xử lý các NaN trong cột Machine Capacity (L)
df['Machine Capacity (L)'].isnull().sum()
# lấy ngẫu nhiên từ cột Machine Capacity (L) một giá trị khác NaN,
# kết quả sẽ ko lặp lại sau mỗi lần thực hiện lệnh
df['Machine Capacity (L)'].dropna().sample()
```

```
934    1.5
Name: Machine Capacity (L), dtype: float64
```

*Hình **8**: Xử lý dữ liệu trống*

```
random_samples = df['Machine Capacity (L)'].dropna().sample(n=df['Machine Capacity (L)'].isnull()
.sum(),random_state=0)
random_samples
```

```
2643    1.6
1490    2.0
583     2.0
117     2.2
2222    2.0
       ...
1031    2.0
832     2.2
409     2.3
3495    1.5
1019    2.0
Name: Machine Capacity (L), Length: 81, dtype: float64
```

***Hình 9** :Chỉ số dữ liệu của từng hàng*

Sau đây là các chỉ số của các hang đã trống

```
df[df['Machine Capacity (L)'].isnull()].index
```

```
Int64Index([ 220,  221,  244,  262,  281,  313,  339,  346,  368,  373,  375,
             647,  701,  745,  767,  891,  944, 1083, 1129, 1160, 1231, 1283,
            1340, 1414, 1417, 1443, 1520, 1599, 1601, 1609, 1617, 1790, 1822,
            1846, 1860, 1981, 1987, 2034, 2128, 2130, 2214, 2251, 2252, 2305,
            2331, 2333, 2360, 2383, 2400, 2444, 2492, 2502, 2524, 2547, 2589,
            2685, 2767, 2771, 2834, 2850, 2910, 2973, 2977, 3023, 3066, 3147,
            3168, 3170, 3256, 3258, 3260, 3284, 3291, 3292, 3295, 3303, 3442,
            3502, 3560, 3597, 3604],
           dtype='int64')
```

***Hình 10**:Danh sách các hàng bị trống dữ liệu*

2.2.7 Random dữ liệu

Thay thế dữ liệu trống bằng các giá trị ngẫu nhiên của cột

```
df['Machine Capacity (L) Random']=df['Machine Capacity (L)']
df.loc[df['Machine Capacity (L)'].isnull(), 'Machine Capacity (L) Random']=random_samples
df.tail()
```

Python

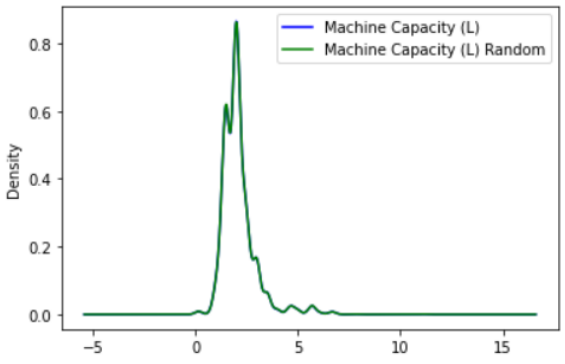
	Id	Year	New Or Old	Name	Address	Production Origin	Color	Fuel	Car Gearbox	Used (kilomet)	Price	Company	Fuel Type	Machine Capacity (L)	Machine Capacity (L) Random
3635	3298640	2022	Xe mới	Suzuki Super Carry Truck 1.0 MT	Hà Nội	Xe lắp ráp trong nước	màu xanh	máy xăng 1.0 L	số tay	0	238 Triệu	Suzuki	máy xăng	1.0	1.0
3636	4184814	2016	Xe cũ	Porsche Cayenne S	Hà Nội	Xe nhập khẩu	màu nâu	máy xăng 3.6 L	số tự động	52000	4 Tỷ 150 Triệu	Porsche	máy xăng	3.6	3.6
3637	3394496	2022	Xe mới	Suzuki Ertiga Sport 1.5 AT	Hưng Yên	Xe nhập khẩu	màu trắng	máy xăng 1.5 L	số tự động	0	558 Triệu	Suzuki	máy xăng	1.5	1.5
3638	4376164	2019	Xe cũ	Toyota Yaris 1.5G	Hà Nội	Xe nhập khẩu	màu đỏ	máy xăng 1.5 L	số tự động	46000	610 Triệu	Toyota	máy xăng	1.5	1.5
3639	3281968	2022	Xe mới	VinFast Lux A 2.0 Cao cấp	Hà Nội	Xe lắp ráp trong nước	màu nâu	máy xăng 2.0 L	số tự động	0	944 Triệu	VinFast	máy xăng	2.0	2.0

***Hình 11**: Thêm dữ liệu vào chỗ còn trống*

2.2.8 Hàm mật độ xác suất của cột "Machine Capacity (L)" trước và sau khi điền

```
fig = plt.figure()
ax = fig.add_subplot(111)
df['Machine Capacity (L)'].plot(kind='kde', ax=ax, color='blue')
df['Machine Capacity (L) Random'].plot(kind='kde', ax=ax, color='green')
lines, labels = ax.get_legend_handles_labels()
ax.legend(lines, labels, loc='best')
```

```
<matplotlib.legend.Legend at 0x1b1ed19d2a0>
```



***Hình 12**: Xác suất sau khi thêm dữ liệu vào*

2.2.9 Xóa các cột ban đầu mà không ảnh hưởng và chuyển cột "Price" về cuối dataframe

```
df.drop(["Name"],axis=1,inplace=True)
df.drop(["Id"],axis=1,inplace=True)
df.drop(["Fuel"],axis=1,inplace=True)
df.drop(["Machine Capacity (L)"],axis=1,inplace=True)
```

***Hình 13**; Xóa cột không quan trọng*

Xóa các cột: Name, Id, Fuel, Machine Capacity(L).

```
moving_column = df.pop('Price')
df.insert(10,'Price',moving_column)
```
✓ 0.1s

```
df.head()
```
✓ 0.1s

	Year	New Or Old	Address	Production Origin	Color	Car Gearbox	Used (kilomet)	Company	Fuel Type	Machine Capacity (L) Random	Price
0	2022	Xe mới	Hà Nội	Xe nhập khẩu	màu đỏ	số tự động	0	Mercedes	máy xăng	3.0	4 Tỷ 359 Triệu
1	2022	Xe mới	Hà Nội	Xe lắp ráp trong nước	màu đỏ	số tự động	0	Mercedes	máy xăng	2.0	2 Tỷ 398 Triệu
2	2022	Xe mới	Hà Nội	Xe lắp ráp trong nước	màu đen	số tự động	0	Mercedes	máy xăng	2.0	2 Tỷ 398 Triệu
3	2019	Xe cũ	Hà Nội	Xe nhập khẩu	màu đen	số tự động	10000	Lexus	máy xăng	2.0	3 Tỷ 80 Triệu
4	2022	Xe mới	Hà Nội	Xe nhập khẩu	màu trắng	số tự động	0	Toyota	máy xăng	1.5	688 Triệu

Hình 14: Chuyển cột "Price" về cuối dataframe

2.2.10 Chuyển dữ liệu về file clean_data

Sau khi làm xong thì chuyển dữ liệu về file clean data:

```
Year,New Or Old,Address,Production Origin,Color,Car Gearbox,Used (kilomet),Company,Fuel Type,Machine Capacity (L) Random,Price
2022, Xe mới, Hà Nội, Xe nhập khẩu, màu đỏ, số tự động, 0, Mercedes, máy xăng, 3.0, 4 Tỷ 359 Triệu
2022, Xe mới, Hà Nội, Xe lắp ráp trong nước, màu đỏ, số tự động, 0, Mercedes, máy xăng, 2.0, 2 Tỷ 398 Triệu
2022, Xe mới, Hà Nội, Xe lắp ráp trong nước, màu đen, số tự động, 0, Mercedes, máy xăng, 2.0, 2 Tỷ 398 Triệu
2019, Xe cũ, Hà Nội, Xe nhập khẩu, màu đen, số tự động, 10000, Lexus, máy xăng, 2.0, 3 Tỷ 80 Triệu
2022, Xe mới, Hà Nội, Xe nhập khẩu, màu trắng, số tự động, 0, Toyota, máy xăng, 1.5, 688 Triệu
2017, Xe cũ, TP HCM, Xe lắp ráp trong nước, màu đen, số tự động, 38000, Mercedes, máy xăng, 2.0, 1 Tỷ 539 Triệu
2017, Xe cũ, Hà Nội, Xe nhập khẩu, màu trắng, số tự động, 75000, Honda, máy xăng, 2.4, 795 Triệu
2019, Xe cũ, Hà Nội, Xe nhập khẩu, màu trắng, số tự động, 35100, Mazda, máy xăng, 1.5, 503 Triệu
2010, Xe cũ, Hà Nội, Xe nhập khẩu, màu đen, số tự động, 7000, Hyundai, máy dầu, 2.0, 535 Triệu
2018, Xe cũ, Hà Nội, Xe lắp ráp trong nước, màu xanh, số tự động, 59000, Mercedes, máy xăng, 3.0, 3 Tỷ 450 Triệu
2022, Xe cũ, TP HCM, Xe lắp ráp trong nước, màu trắng, số tự động, 30, Mercedes, máy xăng, 2.0, 3 Tỷ 189 Triệu
2020, Xe cũ, Long An, Xe lắp ráp trong nước, màu bạc, số tự động, 9000, Hyundai, máy xăng, 1.25, 399 Triệu
2020, Xe mới, Hà Nội, Xe nhập khẩu, màu đen, số tự động, 0, BMW, máy xăng, 3.0, 5 Tỷ 869 Triệu
2021, Xe mới, Hà Nội, Xe nhập khẩu, màu trắng, số tự động, 0, Mazda, máy xăng, 2.0, 909 Triệu
2020, Xe cũ, Hà Nội, Xe lắp ráp trong nước, màu vàng, số tự động, 20000, Hyundai, máy xăng, 2.4, 1 Tỷ 120 Triệu
2021, Xe mới, Hà Nội, Xe nhập khẩu, màu trắng, số tay, 0, Mitsubishi, máy xăng, 1.2, 380 Triệu
2015, Xe cũ, Hà Nội, Xe nhập khẩu, màu đỏ, số tự động, 250000, Nissan, máy xăng, 1.6, 660 Triệu
2011, Xe cũ, Hưng Yên, Xe lắp ráp trong nước, màu đen, số tự động, 120000, Toyota, máy xăng, 1.8, 412 Triệu
2021, Xe cũ, TP HCM, Xe lắp ráp trong nước, màu đen, số tự động, 8000, Mercedes, máy xăng, 2.0, 1 Tỷ 779 Triệu
1993, Xe cũ, Hà Nội, Xe nhập khẩu, màu bạc, số tay, 300000, Toyota, máy xăng, 2.4, 69 Triệu
2015, Xe cũ, Hà Nội, Xe lắp ráp trong nước, màu cát, số tay, 62000, Toyota, máy xăng, 1.5, 299 Triệu
2016, Xe cũ, Hà Nội, Xe lắp ráp trong nước, màu cát, số tay, 62000, Toyota, máy xăng, 2.0, 368 Triệu
2016, Xe cũ, Hà Nội, Xe lắp ráp trong nước, màu cát, số tay, 60000, Toyota, máy xăng, 1.5, 305 Triệu
2016, Xe cũ, Hà Nội, Xe lắp ráp trong nước, màu cát, số tay, 60000, Toyota, máy xăng, 1.5, 299 Triệu
2008, Xe cũ, Hà Nội, Xe nhập khẩu, màu xám, số tự động, 91000, Toyota, máy xăng, 1.3, 285 Triệu
2019, Xe cũ, Bình Dương, Xe nhập khẩu, màu đen, số tự động, 56000, Ford, máy dầu, 2.0, 1 Tỷ 88 Triệu
2022, Xe mới, Hà Nội, Xe lắp ráp trong nước, màu đỏ, số tự động, 0, Mercedes, máy xăng, 2.0, 2 Tỷ 129 Triệu
2022, Xe mới, Thanh Hóa, Xe lắp ráp trong nước, màu trắng, số tự động, 0, Mazda, máy xăng, 1.5, 789 Triệu
2022, Xe mới, Hà Nội, Xe lắp ráp trong nước, màu xanh, số tự động, 0, VinFast, máy xăng, 1.4, 352 Triệu
2017, Xe cũ, Hà Nội, Xe lắp ráp trong nước, màu đỏ, số tự động, 35000, Ford, máy xăng, 1.5, 495 Triệu
2022, Xe mới, Hà Nội, Xe lắp ráp trong nước, màu đỏ, số tự động, 0, Hyundai, máy xăng, 2.0, 925 Triệu
```

Hình 15: File clean_data

2.2.11 Thống kê mô tả trực quan về dữ liệu.

+ Encoding for 'Price' column

```
for i in range (df.shape[0]):
    result = 0.0
    list_temp = df['Price'][i].split(' ')[:]

    for j in range(len(list_temp)):

        if(list_temp[j] == 'Tỷ'): # 4 Tỷ 359 Triệu , 688 Triệu
            base1 = 1000000000.0
            temp_result = pd.to_numeric(list_temp[j-1], downcast='integer') * base1
            result = result + temp_result
        if (list_temp[j] == 'Triệu'):
            base2 = 1000000.0
            temp_result = pd.to_numeric(list_temp[j-1], downcast='integer') * base2
            result = result + temp_result

    df.at[i, 'Price'] = result


df['Price'] = pd.to_numeric(df['Price'], downcast='float')
```

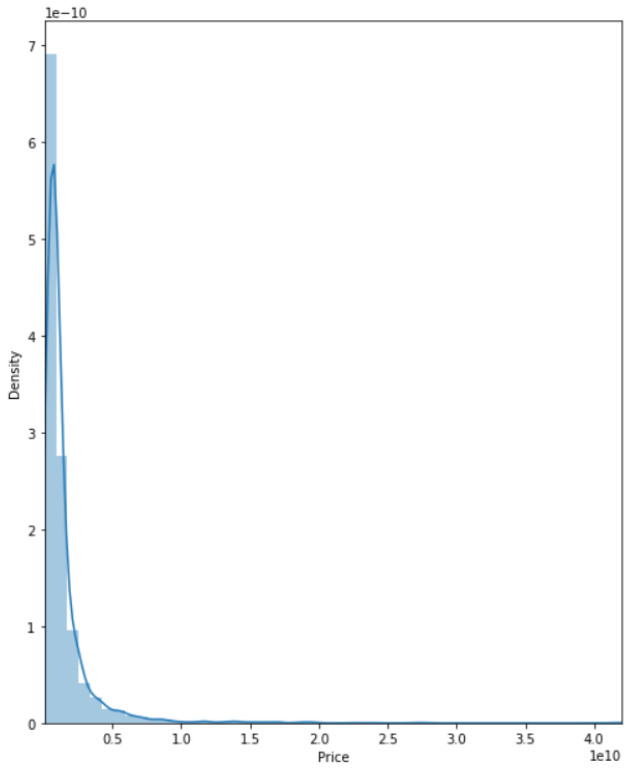
Hình 16: Encoding for " Price " column

	Year	New Or Old	Address	Production Origin	Color	Car Gearbox	Used (kilomet)	Company	Fuel Type	Machine Capacity (L) Random	Price
0	2022	Xe mới	Hà Nội	Xe nhập khẩu	màu đỏ	số tự động	0	Mercedes	máy xăng	3.0	4.359000e+09
1	2022	Xe mới	Hà Nội	Xe lắp ráp trong nước	màu đỏ	số tự động	0	Mercedes	máy xăng	2.0	2.398000e+09
2	2022	Xe mới	Hà Nội	Xe lắp ráp trong nước	màu đen	số tự động	0	Mercedes	máy xăng	2.0	2.398000e+09
3	2019	Xe cũ	Hà Nội	Xe nhập khẩu	màu đen	số tự động	10000	Lexus	máy xăng	2.0	3.080000e+09
4	2022	Xe mới	Hà Nội	Xe nhập khẩu	màu trắng	số tự động	0	Toyota	máy xăng	1.5	6.880000e+08

***Hình 17**:Kết quả sau khi encoding for "Price"*

2.2.12 Trực quan sự phân bố giá trị của cột 'Price'

```
f, ax = plt.subplots(figsize=(8,10))
sns.distplot(df['Price'])
plt.xlim([6.300000e+07,4.199000e+10])
```



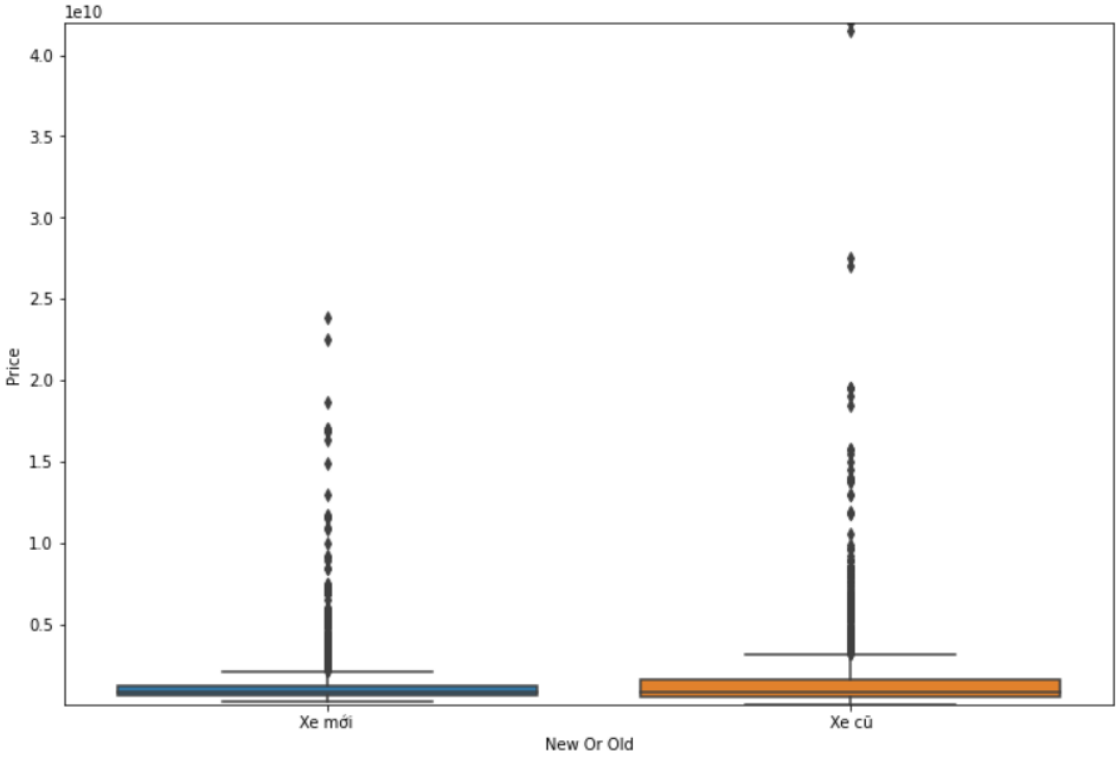
***Hình 18**: Biểu đồ thể hiện sự phân bố các xe*

Nhận xét:

2.2.13 Phân bố giá trị của cột ” New Or Old ”

```
var = 'New Or Old'
data = pd.concat([df['Price'], df[var]], axis=1)
f, ax = plt.subplots(figsize=(12, 8))
fig = sns.boxplot(x=var, y="Price", data=data)
fig.axis(ymin=6.300000e+07, ymax=4.199000e+10);
```

✓ 0.3s



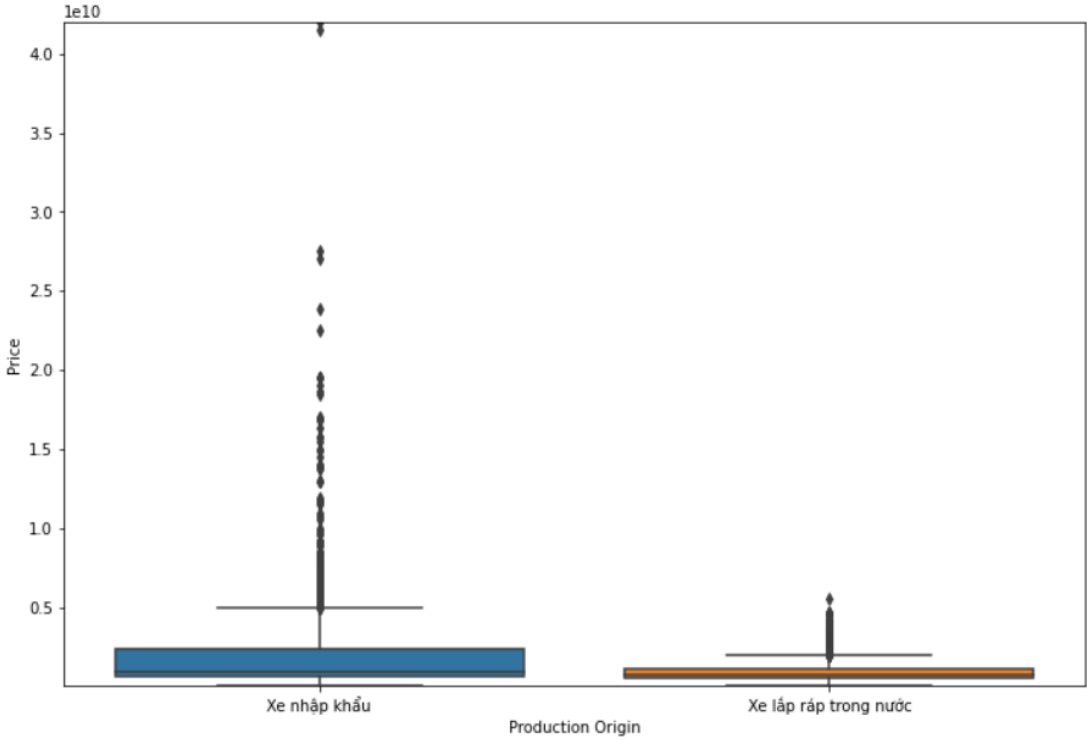
***Hình19**:Biểu đồ thể hiện sự phân bố giá xe theo các loại xe*

2.2.14 Phân bố giá trị của cột “ Production Origin “

```
var = 'Production Origin'
data = pd.concat([df['Price'], df[var]], axis=1)
f, ax = plt.subplots(figsize=(12, 8))
fig = sns.boxplot(x=var, y="Price", data=data)
fig.axis(ymin=6.300000e+07, ymax=4.199000e+10);
```

✓ 0.6s

1e10

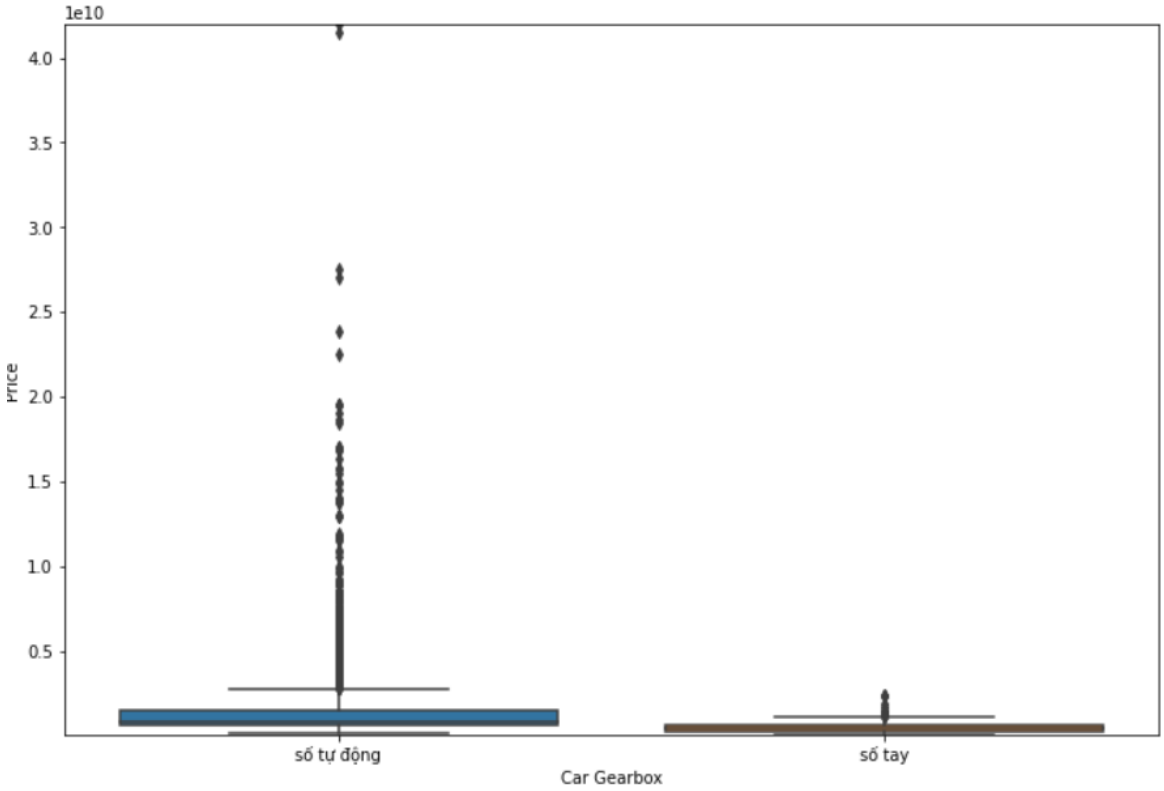


***Hình 20**:Biểu đồ thể hiện sự phân bố giá của các xe theo nguồn gốc xuất sứ*

2.2.15 Phân bố giá trị của cột " Car Gearbox "

```
var = 'Car Gearbox'
data = pd.concat([df['Price'], df[var]], axis=1)
f, ax = plt.subplots(figsize=(12, 8))
fig = sns.boxplot(x=var, y="Price", data=data)
fig.axis(ymin=6.300000e+07, ymax=4.199000e+10);
```

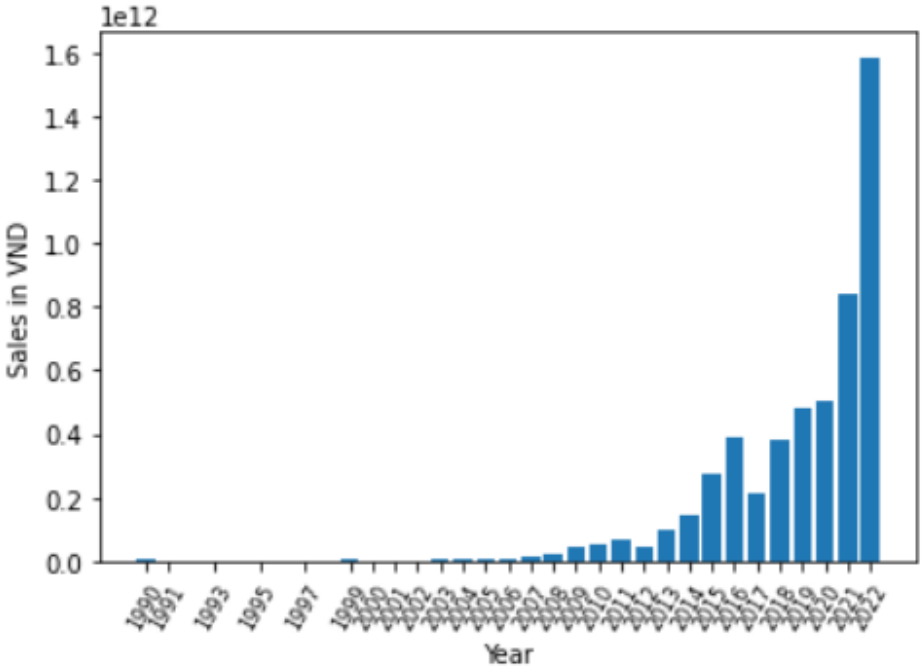
✓ 0.6s



***Hình 21**;Biểu đồ thể hiện sự phân bố tổng giá của xe theo loại hộp số*

2.2.16 Phân bố giá trị của cột " Year "

```
# groupby giá trị của cột 'Price' theo 'Year'
sales_value = df.groupby('Year').sum()['Price']
years = list(set(df['Year']))
plt.bar(x=years, height=sales_value)
plt.xticks(years, rotation= 60, size=8)
plt.xlabel('Year')
plt.ylabel('Sales in VND')
plt.show()
# Điều ở trên chưa chính xác khi mà có thể có các loại xe có giá trị đắt đỏ.
```

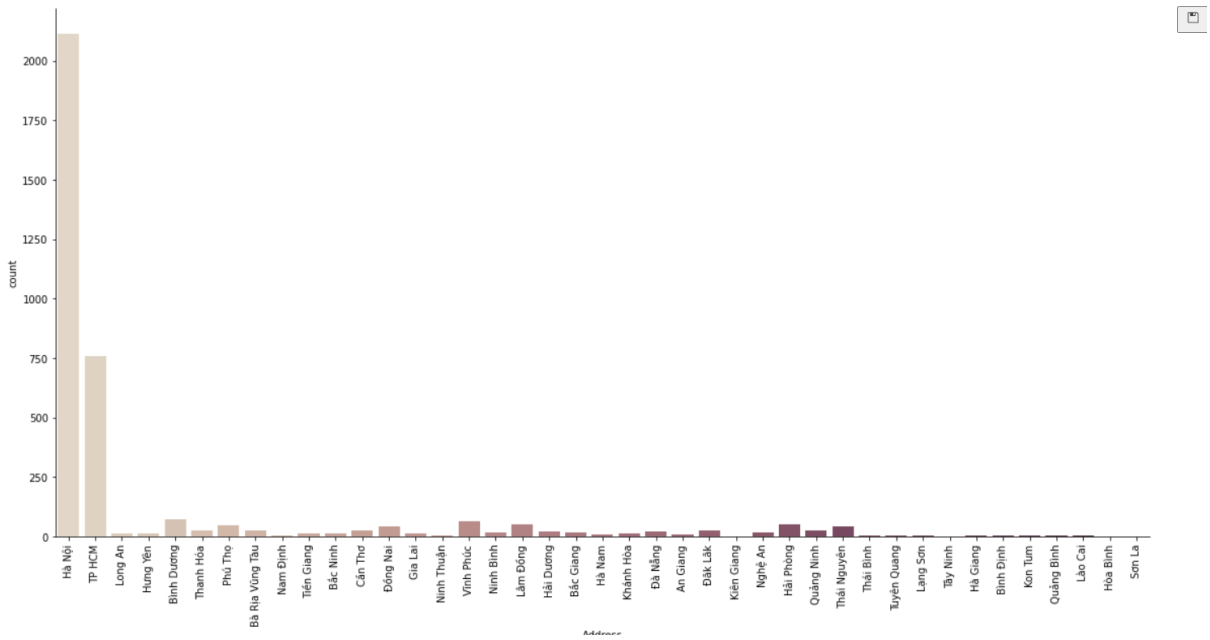


Hình22: Biểu đồ thể hiện tổng số xe bán đc theo số xe bán được theo từng năm

2.2.17 Phân bố giá trị của cột " Address "

```
var = "Address"
plt.figure(figsize=(20, 10))
sns.catplot(x=var, kind="count", palette="ch:.25", height=8, aspect=2, data=df)
plt.xticks(rotation=90);
```

```
<Figure size 1440x720 with 0 Axes>
```

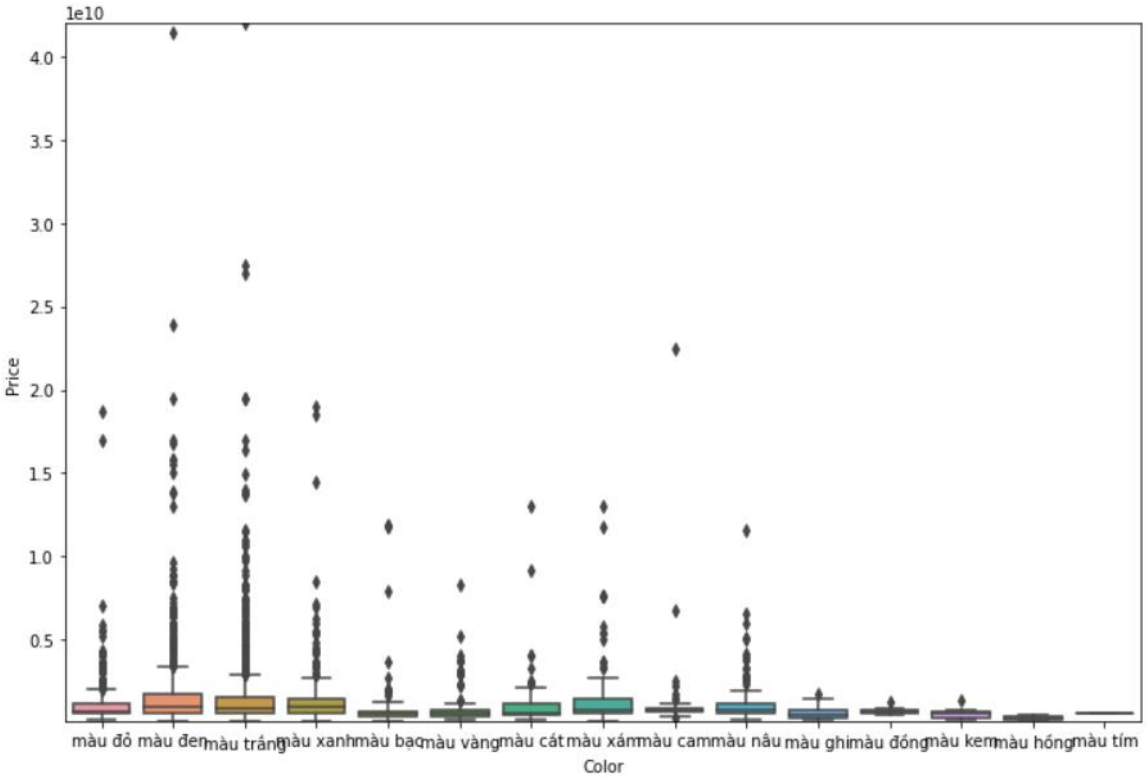


***Hình23**:Biểu đồ thể hiện sự phân bố số lượng xe theo địa chỉ*

2.2.18 Phân bố giá trị của cột " color "

```
var = 'Color'
data = pd.concat([df['Price'], df[var]], axis=1)
f, ax = plt.subplots(figsize=(12, 8))
fig = sns.boxplot(x=var, y="Price", data=data)
fig.axis(ymin=6.300000e+07, ymax=4.199000e+10);
```

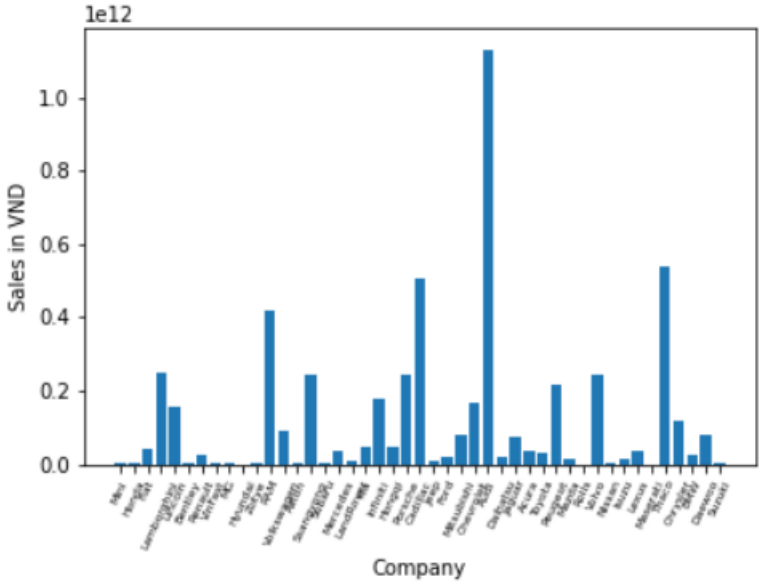
✓ 1.4s



***Hình24**: Biểu đồ thể hiện sự phân bố theo màu sắc*

2.2.19 Phân bố giá trị của cột " Company "

```
sales_value = df.groupby('Company').sum()['Price']
Company = list(set(df['Company']))
plt.bar(x=Company, height=sales_value)
plt.xticks(Company, rotation= 65, size=6)
plt.xlabel('Company')
plt.ylabel('Sales in VND')
plt.show()
```



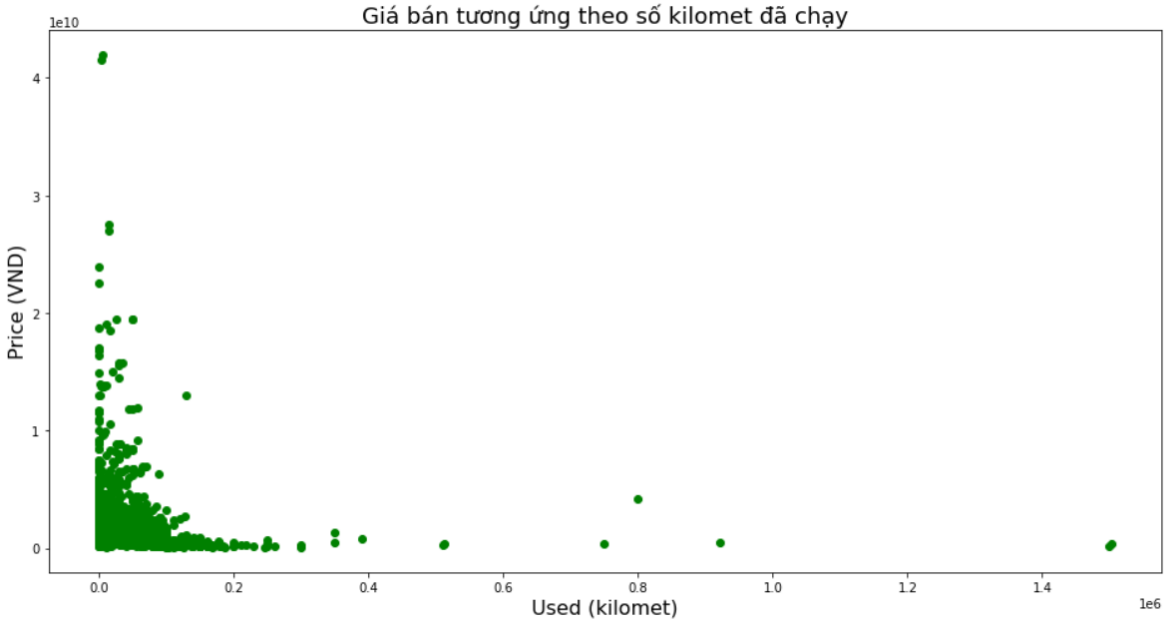
***Hình25**:Biểu đồ thể hiện sự phân bố của tổng giá xe theo công ty sản xuất xe*

2.2.20 Price

```
x = df['Used (kilomet)'].values
y = df['Price'].values

plt.figure(figsize=(16, 8))
plt.scatter(x, y, color='green', )
plt.xlabel('Used (kilomet)', fontsize=16)
plt.ylabel('Price (VND)', fontsize=16)
plt.title("Giá bán tương ứng theo số kilomet đã chạy", fontsize=18)
```

```
Text(0.5, 1.0, 'Giá bán tương ứng theo số kilomet đã chạy')
```

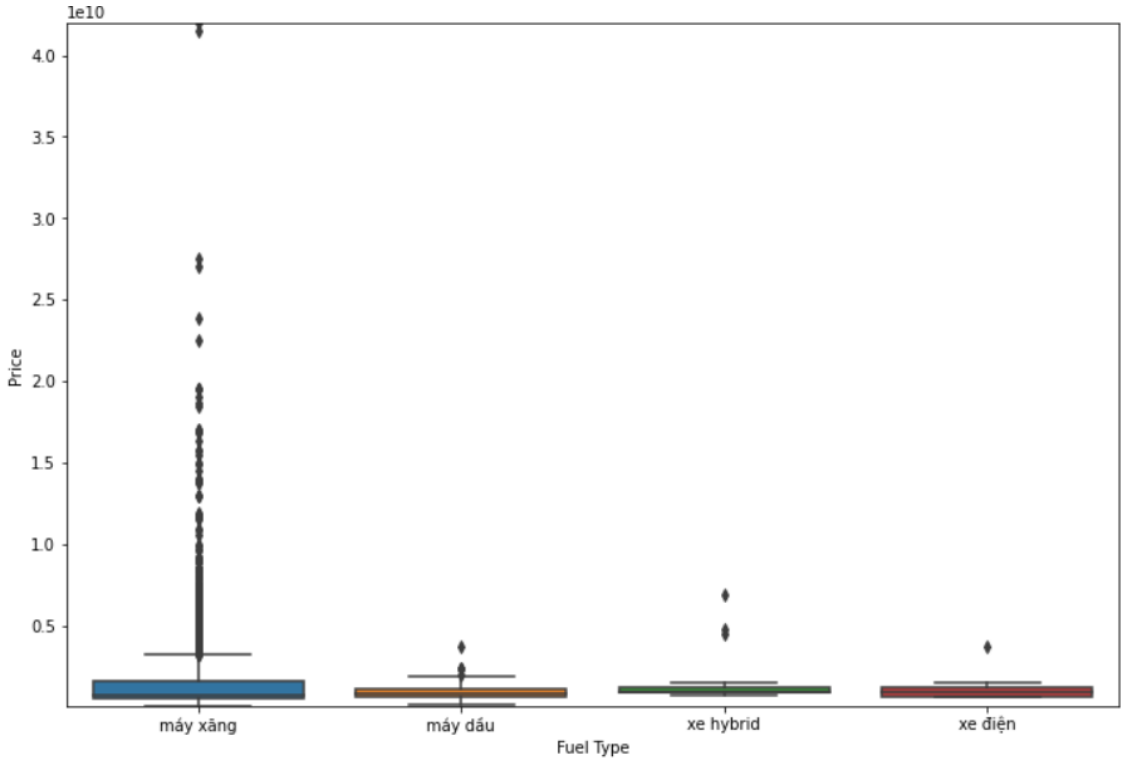


***Hình 26**: Biểu đồ thể hiện sự tương quan giá xe theo số km đi được*

2.2.21 Fuel Type

```
var = 'Fuel Type'
data = pd.concat([df['Price'], df[var]], axis=1)
f, ax = plt.subplots(figsize=(12, 8))
fig = sns.boxplot(x=var, y="Price", data=data)
fig.axis(ymin=6.300000e+07, ymax=4.199000e+10);
```

✓ 0.3s



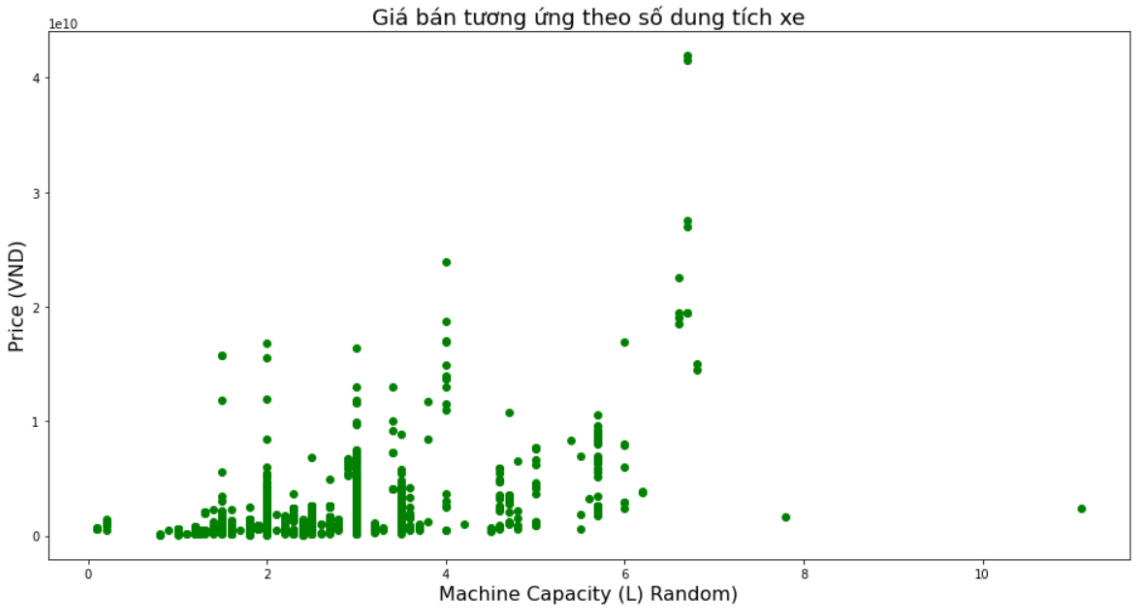
Hình 27 Biểu đồ thể hiện sự tương quan giữa giá xe theo loại nhiên liệu

2.2.22 Machine Capacity (L)

```
x = df['Machine Capacity (L) Random'].values
y = df['Price'].values

plt.figure(figsize=(16, 8))
plt.scatter(x, y, color='green', )
plt.xlabel('Machine Capacity (L) Random)', fontsize=16)
plt.ylabel('Price (VND)', fontsize=16)
plt.title("Giá bán tương ứng theo số dung tích xe", fontsize=18)
```

```
Text(0.5, 1.0, 'Giá bán tương ứng theo số dung tích xe')
```



***Hình 28**: Biểu đồ thể hiện giá bán xe theo dung tích*

3. Trích xuất đặc trưng

3.1 Encoding for 'New or Old' column

Encoding for 'New or Old' column

```
for i in range(df.shape[0]):
    if(df['New Or Old'][i] == 'Xe cũ <'):
        df['New Or Old'][i] = df['New Or Old'][i].replace('Xe cũ <', '0')
    elif (df['New Or Old'][i] == 'Xe cũ'):
        df['New Or Old'][i] = df['New Or Old'][i].replace('Xe cũ', '0')
    elif (df['New Or Old'][i] == 'Xe mới'):
        df['New Or Old'][i] = df['New Or Old'][i].replace('Xe mới', '1')

df['New Or Old'] = pd.to_numeric(df['New Or Old'], downcast='integer')

print(set(df['New Or Old']));
```

✓ 13.3s

Hình 29:code chuyển xe cũ thành giá trị "0" và "1"

```
df.head()
```

✓ 0.8s

	Year	New Or Old	Production Origin	Color	Car Gearbox	Used (kilomet)	Company	Fuel Type	Price
0	2022	1	0	màu đỏ	1	0	Mercedes	máy xăng	4.359000e+09
1	2022	1	1	màu đỏ	1	0	Mercedes	máy xăng	2.398000e+09
2	2022	1	1	màu đen	1	0	Mercedes	máy xăng	2.398000e+09
3	2019	0	0	màu đen	1	10000	Lexus	máy xăng	3.080000e+09
4	2022	1	0	màu trắng	1	0	Toyota	máy xăng	6.880000e+08

Hình 30data

```
print(set(df['Car Gearbox']))
```
✓ 0.1s

```
{'số tự động', 'số tay'}
```

```
for i in range(df.shape[0]):
    if(df['Car Gearbox'][i] == 'số tự động'):
        df['Car Gearbox'][i] = df['Car Gearbox'][i].replace('số tự động', '1')
    elif (df['Car Gearbox'][i] == 'số tay'):
        df['Car Gearbox'][i] = df['Car Gearbox'][i].replace('số tay', '0')

df['Car Gearbox'] = pd.to_numeric(df['Car Gearbox'], downcast='integer')

print(set(df['Car Gearbox']));
```
✓ 0.1s

```
{0, 1}
```

Hình 31:Code chuyển loại số của xe thành "0" và "1"

```
df.head()
```
✓ 0.8s

	Year	New Or Old	Production Origin	Color	Car Gearbox	Used (kilomet)	Company	Fuel Type	Price
0	2022	1	0	màu đỏ	1	0	Mercedes	máy xăng	4.359000e+09
1	2022	1	1	màu đỏ	1	0	Mercedes	máy xăng	2.398000e+09
2	2022	1	1	màu đen	1	0	Mercedes	máy xăng	2.398000e+09
3	2019	0	0	màu đen	1	10000	Lexus	máy xăng	3.080000e+09
4	2022	1	0	màu trắng	1	0	Toyota	máy xăng	6.880000e+08

Hình 32data

```
var = 'Color'
Color = df[[var]]
Color = pd.get_dummies(Color,drop_first=True)
var = 'Company'
Company = df[[var]]
Company = pd.get_dummies(Company,drop_first=True)
var = 'Fuel Type'
Fuel_t = df[[var]]
Fuel_t = pd.get_dummies(Fuel_t,drop_first=True)
```
✓ 0.9s

+ Code + Markdown

```
data= pd.concat([df,Color,Company,Fuel_t],axis=1)
data.drop(["Color"],axis=1,inplace=True)
data.drop(["Company"],axis=1,inplace=True)
data.drop(["Fuel Type"],axis=1,inplace=True)
data.head()
```
✓ 0.1s Python

	Year	New Or Old	Production Origin	Car Gearbox	Used (kilomet)	Price	Color_màu cam	Color_màu cát	Color_màu ghi	Color_màu hồng	...	Company_Suzuki	Company_Thaco	Company_Toyota	Com
0	2022	1	0	1	0	4.359000e+09	0	0	0	0	...	0	0	0	
1	2022	1	1	1	0	2.398000e+09	0	0	0	0	...	0	0	0	
2	2022	1	1	1	0	2.398000e+09	0	0	0	0	...	0	0	0	
3	2019	0	0	1	10000	3.080000e+09	0	0	0	0	...	0	0	0	
4	2022	1	0	1	0	6.880000e+08	0	0	0	0	...	0	0	1	

5 rows × 67 columns

Hình 33data

3.2 One-hot-encoding

```
var = 'Color'
Color = df[[var]]
Color = pd.get_dummies(Color,drop_first=True)
var = 'Company'
Company = df[[var]]
Company = pd.get_dummies(Company,drop_first=True)
var = 'Fuel Type'
Fuel_t = df[[var]]
Fuel_t = pd.get_dummies(Fuel_t,drop_first=True)
```

```
data= pd.concat([df,Color,Company,Fuel_t],axis=1)
data.drop(["Color"],axis=1,inplace=True)
data.drop(["Company"],axis=1,inplace=True)
data.drop(["Fuel Type"],axis=1,inplace=True)
data.head()
```

Hình 34*code onehot*

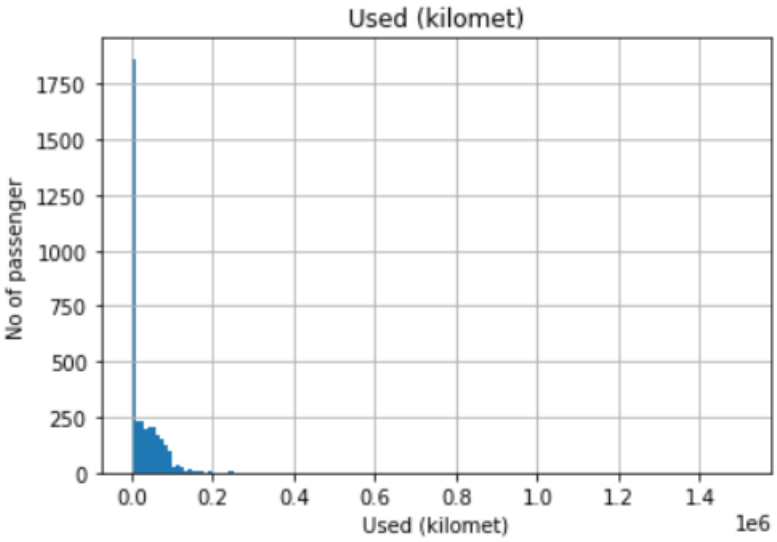
	Year	New Or Old	Production Origin	Car Gearbox	Used (kilomet)	Price	Color_màu cam	Color_màu cát	Color_màu ghi	Color_màu hồng	...	Company_Suzuki	Company_Thaco	Company_Toyota
0	2022	1	0	1	0	4.359000e+09	0	0	0	0	...	0	0	0
1	2022	1	1	1	0	2.398000e+09	0	0	0	0	...	0	0	0
2	2022	1	1	1	0	2.398000e+09	0	0	0	0	...	0	0	0
3	2019	0	0	1	10000	3.080000e+09	0	0	0	0	...	0	0	0
4	2022	1	0	1	0	6.880000e+08	0	0	0	0	...	0	0	1

5 rows × 67 columns

Hình 35*data*

3.3 Outliers

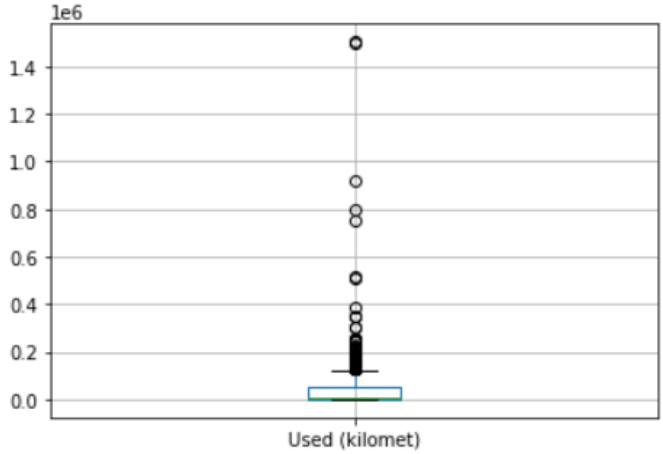
Used



Hình 36 *Biểu đồ thể hiện dữ liệu lệch*

```
data.boxplot(column="Used (kilomet)")
```

```
<AxesSubplot:>
```

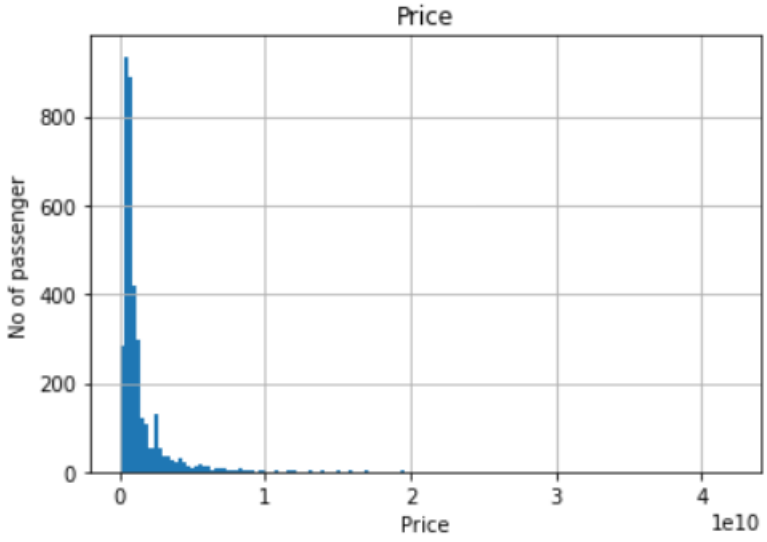


Hình 37*Biểu đồ boxplot thể hiện giá trị ngoại lai*

Price

```
figure=data['Price'].hist(bins=150)
figure.set_title('Price')
figure.set_xlabel('Price')
figure.set_ylabel('No of passenger')
```

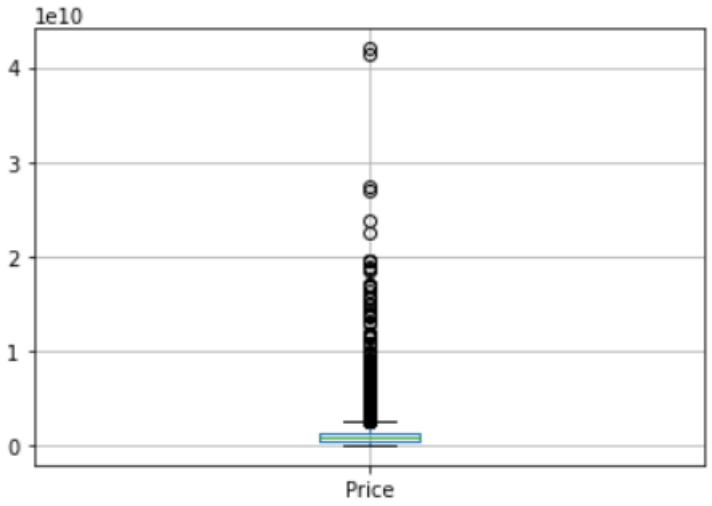
```
Text(0, 0.5, 'No of passenger')
```



Hình 38*Biểu đồ thể hiện giá trị lệch*

```
data.boxplot(column="Price")
```

```
<AxesSubplot:>
```



Hình 39*Biểu đồ boxplot thể hiện giá trị ngoại lai*

```
def outliers(data):
    # Star is skewed to the left
    q3_1 , q1_1 = np.percentile(data['Used (kilomet)'], [75,25])
    IQR_1 = q3_1 - q1_1
    upper_Used = q3_1 + 3 * IQR_1
    lower_Used = q1_1 - 3 * IQR_1
    data.loc[data['Used (kilomet)'] >= upper_Used, 'Used (kilomet)'] = round(lower_Used)

    # Fork is skewed to the left
    q3_2 , q1_2 = np.percentile(data['Price'], [75,25])
    IQR_2 = q3_2 - q1_2
    upper_Price = q3_2 + 3 * IQR_2
    lower_Price = q1_2 - 3 * IQR_2
    data.loc[data['Price'] >= upper_Price, 'Price'] = round(lower_Price)
    return data
```

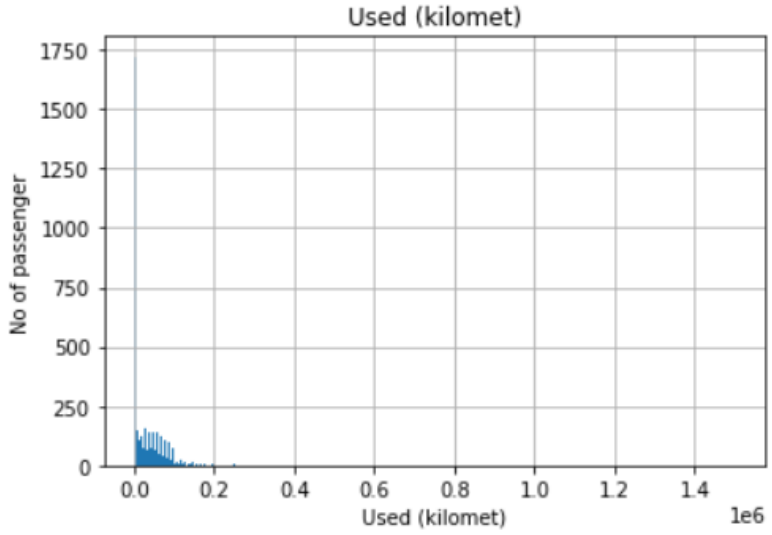
```
figure=data['Used (kilomet)'].hist(bins=300)
figure.set_title('Used (kilomet)')
figure.set_xlabel('Used (kilomet)')
figure.set_ylabel('No of passenger')
```

```
Text(0, 0.5, 'No of passenger')
```

Hình 40: Xử lý ngoại lệ

```
figure=data['Used (kilomet)'].hist(bins=300)
figure.set_title('Used (kilomet)')
figure.set_xlabel('Used (kilomet)')
figure.set_ylabel('No of passenger')
```

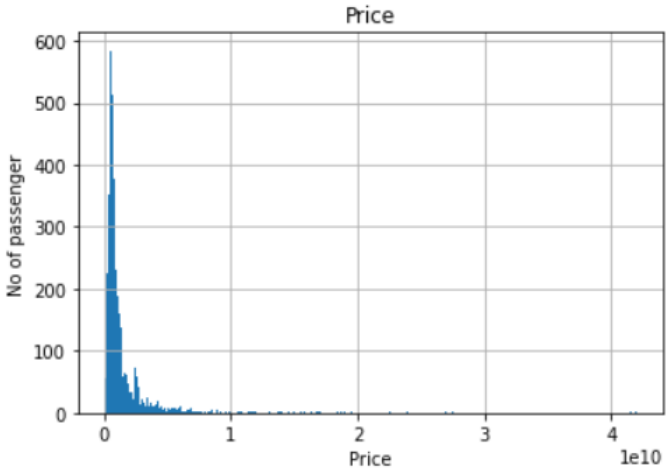
```
Text(0, 0.5, 'No of passenger')
```



Hình 41Biểu đồ sau khi xử lý outlires

```
figure=data['Price'].hist(bins=300)
figure.set_title('Price')
figure.set_xlabel('Price')
figure.set_ylabel('No of passenger')
```

```
Text(0, 0.5, 'No of passenger')
```



Hình 42*Biểu đồ sau khi xử lý outlires*

3.4 Feature Transformation

Min-Max Scaling

Min Max Scaling scales the values between 0 to 1: X_scaled = (X - X.min / (X.max - X.min)

```
from sklearn.preprocessing import MinMaxScaler
min_max=MinMaxScaler()
final_data=pd.DataFrame(min_max.fit_transform(data),columns=data.columns)
final_data.head()
```

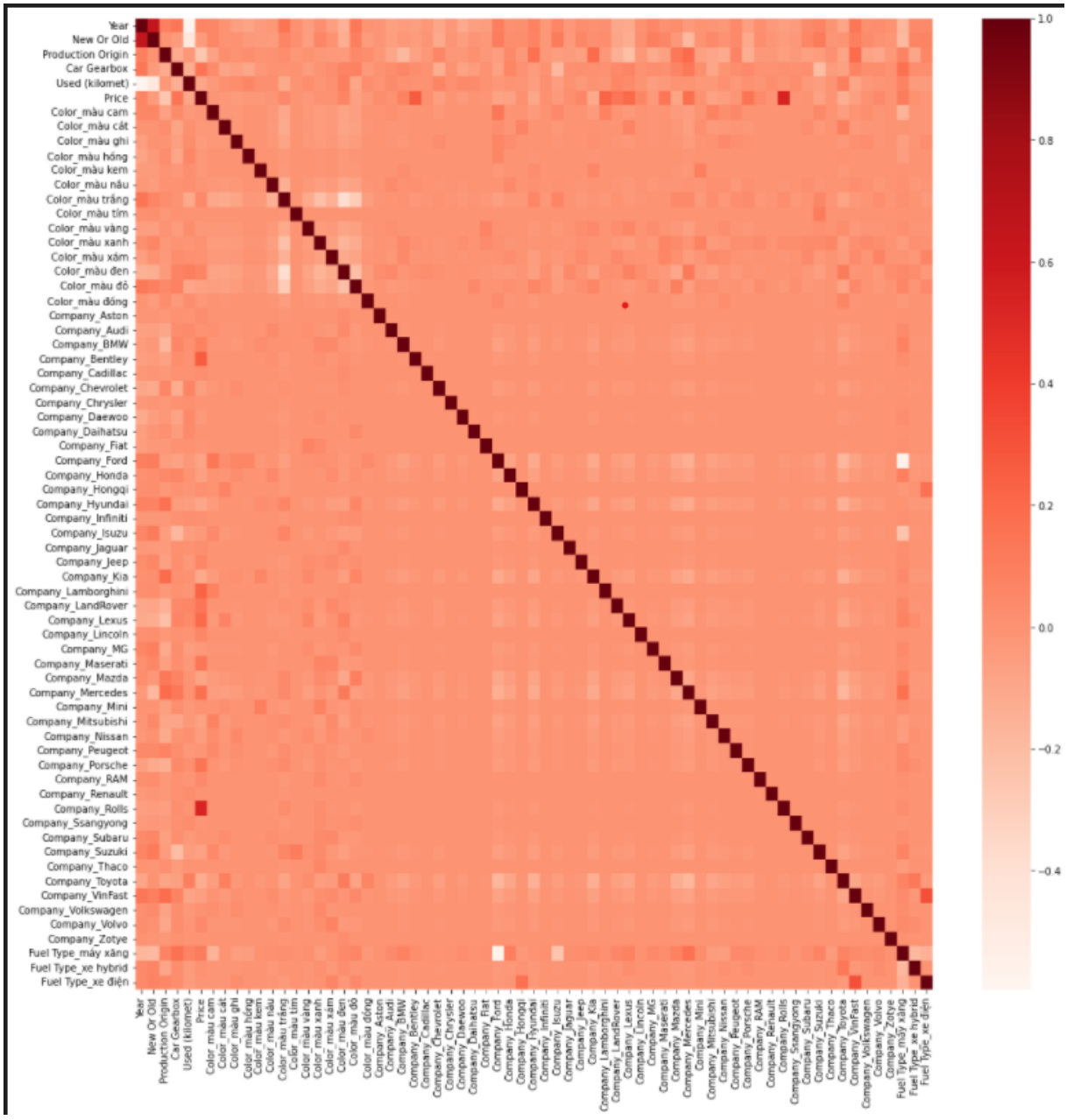
Pythc

	Year	New Or Old	Production Origin	Car Gearbox	Used (kilomet)	Price	Color_màu cam	Color_màu cát	Color_màu ghi	Color_màu hồng	...	Company_Suzuki	Company_Thaco	Company_Toyota
0	1.00000	1.0	0.0	1.0	0.000000	0.102464	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0
1	1.00000	1.0	1.0	1.0	0.000000	0.055692	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0
2	1.00000	1.0	1.0	1.0	0.000000	0.055692	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0
3	0.90625	0.0	0.0	1.0	0.006656	0.071958	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0
4	1.00000	1.0	0.0	1.0	0.000000	0.014907	0.0	0.0	0.0	0.0	...	0.0	0.0	1.0

5 rows × 67 columns

Hình 43*Kĩ thuật min-max*

Ma trận tương quan



Hình 44:Biểu đồ ma trận tương quan thể hiện sự tương quan giữa các biến

```
y = final_data.loc[:,['Price']]
y.head()
```

	Price
0	0.102464
1	0.055692
2	0.055692
3	0.071958
4	0.014907

4. Mô hình hóa dữ liệu

Ta đã định nghĩa xong bài toán, thu thập và khám phá dữ liệu, lấy mẫu tập huấn luyện và tập kiểm tra, rồi viết pipeline chứa các phép biến đổi để làm sạch và chuẩn bị dữ liệu cho các thuật toán Học máy một cách tự động. Cuối cùng thì ta đã sẵn sàng để chọn và huấn luyện hai mô hình Học máy và đánh giá so sánh giữa hai thuật toán. Ta chọn mô hình Linear Regression và Random Forest

Cơ sở lý thuyết

- Mô hình Linear Regression
    + Một cách tổng quát, mô hình tuyến tính đưa ra dự đoán bằng cách tính tổng trọng số các đặc trưng đầu vào, sau đó cộng tổng này với một hằng số gọi là hệ số điều chỉnh, ta có thể thấy công thức như sau:

$$\hat{y} = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \cdots + \theta_n I_n$$

hương trình 4.1. Dự đoán của mô hình Hồi quy Tuyến tính

    + Vậy là ta đã biết khái niệm mô hình Hồi quy tuyến tính, vậy còn phương thức huấn luyện, huấn luyện mô hình là tìm ra bộ giá trị tham số mô hình để khớp tập huấn luyện một cách tốt nhất. Để làm được điều này, trước tiên ta cần một phép đo phổ biến nhất của mô hình Hồi quy tuyến tính là Căn trung bình Bình phương sai số sẽ được nói ở phần sau.
- Mô hình Random Forest
    + Rừng ngẫu nhiên là ensemble của cây quyết định, thường được huấn luyện thông quan phương pháp bagging với max_samples thường được đặt bằng

với kích thước bộ huấn luyện. Thay vì phải tạo một Bagging Classifier và truyền một Decision Tree Classifier, sử dụng lớp RainForest Classifier sẽ tiện lợi và tối ưu hơn cho thuật toán Cây Quyết Định(tương tự, ta có lớp Random Forest Regression dành cho những tác vụ hồi quy)

+ Một ưu điểm khác của rừng ngẫu nhiên là dễ dàng đo lường được độ quan trọng tương đối của mỗi đặc trưng. Scikit Learn tính toán độ quan trọng của đặc trưng bằng cách xét xem các nút sử dụng đặc trưng đó có thể giảm độ pha tạp đi bao nhiêu
+ Scikit Learn tính độ quan trọng một cách tự động cho mỗi đặc trưng sau khi huấn luyện, tiếp đó co giãn kết quả sao cho tổng của tất cả độ quan trọng bằng 1. ta có thể thu được kết quả này thông qua thuộc tính feature_importances_.
+ Thuật toán toán này áp dụng tốt cho những bài toán có số chiều cao
+ Lấy ngẫu nhiên n dữ liệu từ bộ dữ liệu với kĩ thuật Bootstrapping, hay còn gọi là random sampling with replacement. Tức khi mình sample được 1 dữ liệu thì mình không bỏ dữ liệu đấy ra mà vẫn giữ lại trong tập dữ liệu ban đầu, rồi tiếp tục sample cho tới khi sample đủ n dữ liệu. Khi dùng kĩ thuật này thì tập n dữ liệu mới của mình có thể có những dữ liệu bị trùng nhau.
+ Sau khi sample được n dữ liệu từ bước 1 thì mình chọn ngẫu nhiên ở k thuộc tính (k < n). Giờ mình được bộ dữ liệu mới gồm n dữ liệu và mỗi dữ liệu có k thuộc tính.
+ Dùng thuật toán Decision Tree để xây dựng cây quyết định với bộ dữ liệu ở bước 2.

Do quá trình xây dựng mỗi cây quyết định đều có yếu tố ngẫu nhiên (random) nên kết quả là các cây quyết định trong thuật toán Random Forest có thể khác nhau.

- Chia tập dữ liệu train và test với tỷ lệ 0.9, 90% dữ liệu sẽ được dùng train model và 10% để test model. Với tham số random state bằng 25.

- Trình bày các tham số của quá trình huấn luyện mô hình (ví dụ: loss function, learning rate, number of epochs,…).

- Tham số để train model là bộ dữ liệu x_train, y_train

- Tham số để test model là bộ dữ liệu x_test, y_test

- Trình bày các đồ thị thể hiện hiệu suất (đánh giá bằng Loss hoặc Accuracy) của các mô hình trên các tập Huấn luyện/Xác thực/Kiểm thử.

- Với Model Linear Regression cho ra kết quả độ chính xác khá thấp:

```
from sklearn.linear_model import LinearRegression
linear_reg = LinearRegression()
linear_reg.fit(X_train, y_train)
y_pred= linear_reg.predict(X_test)
print("Accuracy on Traing set: ",linear_reg.score(X_train,y_train))
print("Accuracy on Testing set: ",linear_reg.score(X_test,y_test))
```

✓ 0.2s

```
Accuracy on Traing set:  0.65904026415604
Accuracy on Testing set:  0.7080980881758259
```

- Với Model RandomForest Regression cho ra kết quả độ chính xác ổn hơn:

```
from sklearn.ensemble import RandomForestRegressor
rf_reg = RandomForestRegressor()
rf_reg.fit(X_train, y_train)
y_pred= rf_reg.predict(X_test)
print("Accuracy on Traing set: ",rf_reg.score(X_train,y_train))
print("Accuracy on Testing set: ",rf_reg.score(X_test,y_test))
```

✓ 2.1s

```
Accuracy on Traing set:  0.9014936928996964
Accuracy on Testing set:  0.7832417907863388
```

```
from sklearn import metrics
from sklearn.metrics import mean_squared_error, mean_absolute_error

print("\t\tError Table Model Linear")
print('Mean Absolute Error      : ', metrics.mean_absolute_error(y_test, y_pred1))
print('Mean Squared  Error      : ', metrics.mean_squared_error(y_test, y_pred1))
```

✓ 0.8s

```
                Error Table Model Linear
Mean Absolute Error      :  0.017121724990811923
Mean Squared  Error      :  0.000856784322507682
```

```
print("\t\tError Table Model Random")
print('Mean Absolute Error      : ', metrics.mean_absolute_error(y_test, y_pred2))
print('Mean Squared  Error      : ', metrics.mean_squared_error(y_test, y_pred2))
```

✓ 0.9s

```
                Error Table Model Random
Mean Absolute Error      :  0.011500876983348033
Mean Squared  Error      :  0.0007143962928587906
```

5. Kết luận

Nhóm chúng em đã làm được những công việc như là crawl dữ liệu về, làm sạch dữ liệu, kĩ thuật xử lý các đặc trưng, khám phá dữ liệu và trực quan hóa dữ liệu, sau đó chọn ra những đặc trưng có ảnh hưởng tới dự đoán giá. Cuối cùng là train model và đánh giá model.

Về hướng phát triển, nhóm em chọn lọc tốt hơn những đặc trưng có mức ảnh hưởng, loại bỏ những đặc trưng không cần thiết và giảm chiều dữ liệu. Và Train thêm những model khác để cho ra kết quả chính xác nhất.

Do chưa có kiến thức hay kinh nghiệm về phần nhận thấy những quy luật tiềm ẩn của các biến đặc trưng nên trong quá trình làm còn nhiều cái chưa đúng đắn. Mong thầy đóng góp ý kiến để nhóm em biết cải thiện!!!

6. Tài liệu tham khảo

-EDA (Exploratory Data Analysis) tham khảo : https://www.youtube.com/watch?v=SMnv8h7gGp8&list=PLJcWUrckOCKKwjjHALg6fnyQCHv8z92rs&index=10

-Cách thu thập dữ liệu : https://www.youtube.com/watch?v=SMnv8h7gGp8&list=PLJcWUrckOCKKwjjHALg6fnyQCHv8z92rs&index=10

-Cách thu thập dữ liệu : https://www.youtube.com/watch?v=S_A3Hyg-1bU