TRƯỜNG ĐẠI HỌC BÁCH KHOA

KHOA CÔNG NGHỆ THÔNG TIN

TIỂU LUẬN CUỐI KỲ

HỌC PHẦN: KHOA HỌC DỮ LIỆU

ĐỀ TÀI:
DỰ ĐOÁN GIÁ XE MÁY CŨ

HỌ VÀ TÊN SINH VIÊN	LỚP HỌC PHẦN	ĐIỂM BẢO VỆ
Trần Thị Thảo	19N13	
Nguyễn Thanh Toàn	19N13	
Đặng Thanh Tuyên	19N13	

Đà Nẵng, 06/2022

TÓM TẮT

Hiện nay, nhu cầu sử dụng xe máy đang tăng cao. Có nhiều dòng xe với các mức giá, công suất hoạt động khác nhau. Tuy nhiên, việc lựa chọn một chiếc xe tốt nhưng lại phù hợp với túi tiền của mỗi người thì không hề dễ dàng. Để giải quyết vấn đề này, nhóm chùng em lựa chọn đề tài ***"Dự đoán giá xe máy cũ"***. Đề tài này sẽ giúp mọi người dự đoán được giá của một chiếc xe máy theo yêu cầu về công năng cũng như hiệu suất của xe, hơn hết, đây là xe máy cũ nên giá cả rất phải chăng, giải quyết được vấn đề tài chính.

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)
Trần Thị Thảo	- Mô hình hóa dữ liệu	- Đã hoàn thành
Nguyễn Thanh Toàn	- Thu thập dữ liệu	- Đã hoàn thành
Đặng Thanh Tuyên	- Trich xuất đặc trưng	- Đã hoàn thành

MỤC LỤC

1. GIỚI THIỆU

2. THU THẬP VÀ MÔ TẢ DỮ LIỆU

2.1. Thu thập dữ liệu

2.1.1. Nguồn dữ liệu:

Nguồn dữ liệu lấy từ 2 trang Web chuyên bán xe mô tô phân khối lớn cũ tại châu Âu:

https://www.usedmotorcyclestore.com/

https://www.autoscout24.com/

2.1.2. Công cụ thu thập: Selenium Web Driver

Selenium là một bộ công cụ kiểm thử tự động open source, dành cho các ứng dụng web, hỗ trợ trên Windows, Mac, Linux...

Selenium WebDriver được dùng để tự động các thao tác với trình duyệt Web, giúp giả lập lại các tương tác trên trình duyệt như một người dùng thực. [1]

2.1.3. Đầu vào, đầu ra của quá trình thu thập dữ liệu

- Đầu vào gồm 2 trang Web:
  - https://www.usedmotorcyclestore.com/,
  - https://www.autoscout24.com/
- Đầu ra: 2 file CSV chứa dữ liệu sau khi thu thập được.

2.1.4. Cách thức sử dụng công cụ:

Tải Chrome Drive về từ địa chỉ website: https://sites.google.com/a/chromium.org/chromedriver/downloads sẽ thu được file chromedriver.exe

Chạy chương trình:

(1) Import thư viện WebDriver: from selenium import webdriver

(2) Khởi tạo đối tượng browser để điều khiển trang web Chrome
Browser = webdriver.Chrome(executable_path = "./chromedriver.exe")

(3) Mở trang web muốn crawl với url: Browser.get(url)

(4) Sử dụng các method do thư viện selenium cung cấp để lấy dữ liệu cần thiết:

a. Tìm kiếm phần tử bằng class name:

find_element_by_class_name(*name*)
→selenium.webdriver.remote.webelement.WebElement
Tham số "name": class name của phần tử muốn tìm kiếm.

b. Tìm kiếm phần tử bằng tag name
find_element_by_tag_name(*name*)
→selenium.webdriver.remote.webelement.WebElement
Tham số name: Tên của HTML tag (h1,a,h2, span,…)

c. Tìm kiếm phần tử bằng Xpath
find_element_by_xpath(*xpath*)
→selenium.webdriver.remote.webelement.WebElement
Tham số name: Định vị xpath của phần tử cần tìm.

- Trả về:
  - WebElement – phần tử web muốn tìm kiếm nếu phần tử đó đã được tìm thấy.
  - NoSuchElementException – Nếu không tìm thấy phần tử

d. Tìm kiếm list phần tử bằng class name
find_elements_by_class_name(*name*) →
List[selenium.webdriver.remote.webelement.WebElement]
Tham số "name": class name của những phần tử muốn tìm kiếm.

e. Tìm kiếm list phần tử bằng tag name
find_elements_by_tag_name(*name*)
→List[selenium.webdriver.remote.webelement.WebElement]
Tham số name: Tên của HTML tag (h1,a,h2, span,…)

f. Tìm kiếm list phần tử bằng xPath
find_elements_by_xpath(*xpath*)
→List[selenium.webdriver.remote.webelement.WebElement]
Tham số name: Định vị xpath của phần tử cần tìm.

- Trả về:
  - Một mảng chứa WebElement – nếu những phần tử web muốn tìm đó đã được tìm thấy.
  - Trả về một mảng rỗng nếu không tìm thấy phần tử nào.

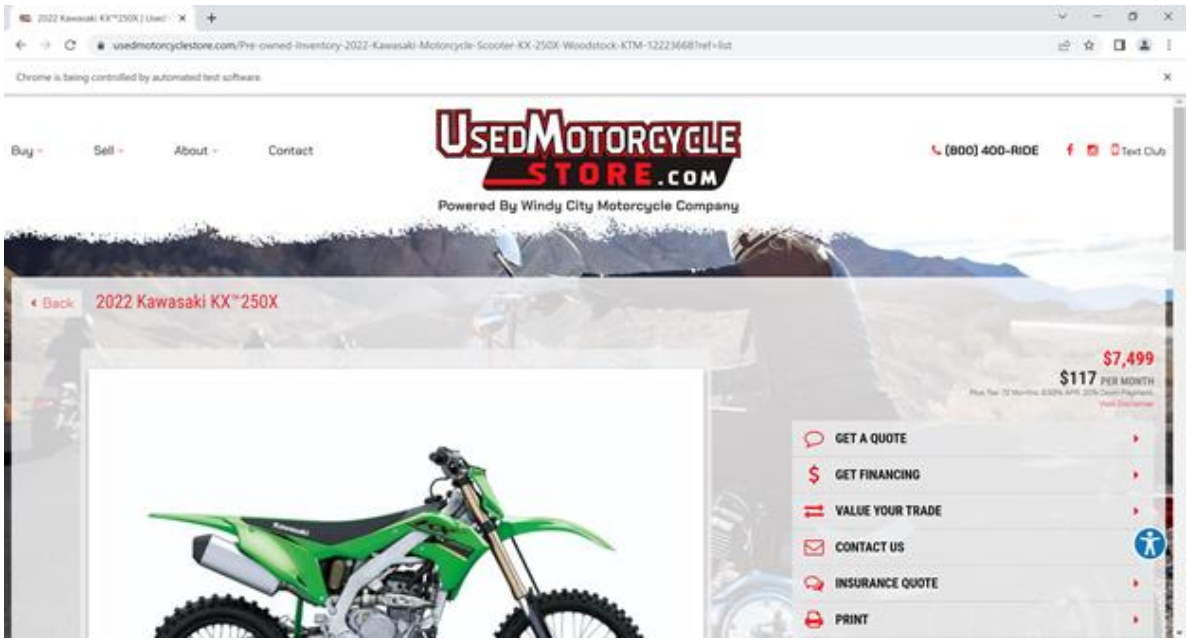
g. Tắt trang web điều khiển tự động:
Browser.close()

2.1.5. Ví dụ minh họa

Lấy dữ liệu từ trang có URL:https://www.usedmotorcyclestore.com/Pre-owned-Inventory-2022-Kawasaki-Motorcycle-Scooter-KX-250X-Woodstock-KTM-12223668?ref=list

Browser = webdriver.Chrome(executable_path =“./chromedriver.exe”)

Browser.get(“https://www.usedmotorcyclestore.com/Pre-owned-Inventory-2022-Kawasaki-Motorcycle-Scooter-KX-250X-Woodstock-KTM-12223668?ref=list”)



Hình 1. Trang web dùng để thu thập dữ liệu

Chương trình sẽ tự mở một trang web theo đường link ở chế độ tự điều khiển bởi phần mềm tự động

Lấy phần tử của 1 trang web sử dụng classname hoặc Xpath (phần tử price-value)

```python
findUsingClassName = browser.find_element_by_class_name('price-value')
findUsingXpath = browser.find_element_by_xpath('//*[@id="template"]/div[2]/div[2]/div[1]/div/div/a/div')
print("Element Using ClassName")
print(findUsingClassName)
print("Element Using Xpath")
print(findUsingXpath)
```

```
Text of Element Using ClassName
$7,499
Text of Element Using Xpath
$7,499

C:\Users\Admin\AppData\Local\Temp\ipykernel_5716\4069353475.py:1: DeprecationWarning: find_element_by_class_name is
deprecated. Please use find_element(by=By.CLASS_NAME, value=name) instead
  findUsingClassName = browser.find_element_by_class_name('price-value')
C:\Users\Admin\AppData\Local\Temp\ipykernel_5716\4069353475.py:2: DeprecationWarning: find_element_by_xpath is deprecated.
Please use find_element(by=By.XPATH, value=xpath) instead
  findUsingXpath = browser.find_element_by_xpath('//*[@id="template"]/div[2]/div[2]/div[1]/div/div/a/div')
```

Hình 2. Minh họa code và kết quả khi lấy dữ liệu bằng class name

Để lấy text trong class “price-value”

```python
findUsingClassName = browser.find_element_by_class_name('price-value')
findUsingXpath = browser.find_element_by_xpath('//*[@id="template"]/div[2]/div[2]/div[1]/div/div/a/div')
print("Text of Element Using ClassName")
print(findUsingClassName.text)
print("Text of Element Using Xpath")
print(findUsingXpath.text)
```

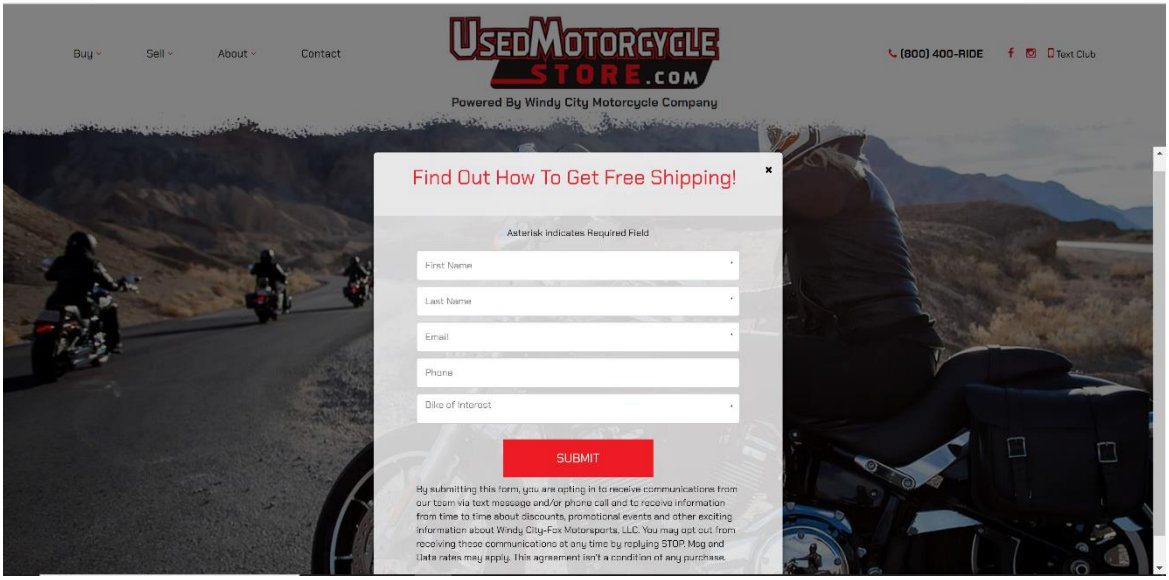
```
Text of Element Using ClassName
$7,499
Text of Element Using Xpath
$7,499

C:\Users\Admin\AppData\Local\Temp\ipykernel_5716\4069353475.py:1: DeprecationWarning: find_element_by_class_name is
deprecated. Please use find_element(by=By.CLASS_NAME, value=name) instead
  findUsingClassName = browser.find_element_by_class_name('price-value')
C:\Users\Admin\AppData\Local\Temp\ipykernel_5716\4069353475.py:2: DeprecationWarning: find_element_by_xpath is deprecated.
Please use find_element(by=By.XPATH, value=xpath) instead
  findUsingXpath = browser.find_element_by_xpath('//*[@id="template"]/div[2]/div[2]/div[1]/div/div/a/div')
```

Hình 3. Minh họa code và kết quả khi lấy TEXT thông qua class name đã tìm được

Đối với cách lấy dữ liệu bằng tag name hoặc Xpath cũng lấy dữ liệu theo các bước như vậy, chỉ thay đổi hàm và tham số đầu vào.

Có một số trang Web sẽ hiển thị một Frame yêu cầu nhập thông tin, chúng ta phải tắt nó mới có thể crawl dữ liệu.



Hình 4. Trang web yêu cầu nhập thông tin mới được lấy dữ liệu

WebDriverWait(browser,20).until(EC.frame_to_be_available_and_switch_to_it((By.XPATH,"//*[@id='gdpr-consent-notice']")))

(Chờ 20s nếu hiển thị Frame thì sẽ lấy Frame và trỏ vào nó)

WebDriverWait(browser,20).until(EC.element_to_be_clickable((By.CLASS_NAME,"mat-focus-indicator.solo-button.mat-button.mat-button-base.mat-raised-button"))).click()

(Tìm phần tử close(dấu X) rồi click vào để tắt Frame)

2.2. Mô tả dữ liệu

- Dữ liệu sau khi crawl của trang https://www.usedmotorcyclestore.com/ được lưu vào file CSV

Vì nhóm đã crawl tất cả các thông số kĩ thuật của mỗi xe máy nên lượng đặc trưng tương đối lớn.

Số mẫu thu được 1066 mẫu với 204 đặc trưng khác nhau.

Để thuận tiện cho việc dự đoán giá xe máy, nhóm đã thống kê ra các đặc trưng có số lượng mẫu lớn (giá trị NULL <40%)

Sau khi thống kê và lọc ra các đặc trưng, cuối cùng thu được 53 đặc trưng có số lượng NULL < 40% với tỉ lệ NULL trung bình là 25%

- Dữ liệu sau khi crawl của trang https://www.autoscout24.com/ được lưu vào file CSV

Với dữ liệu thu được sau khi crawl là 398 mẫu với 34 đặc trưng(5 đặc trưng Float và 29 đặc trưng Object)

Tương tự với dữ liệu thu được từ trang https://www.usedmotorcyclestore.com/, với nhiều đặc trưng có số lượng mẫu thấp nên phải lọc giúp việc dự đoán chính xác hơn.

Sau khi lọc dữ liệu cuối cùng là 398 mẫu với 10 đặc trưng với tỉ lệ NULL trung bình là 6%

- Sau khi đã xóa các đặc trưng không cần thiết và có số lượng NULL cao, nhóm sẽ gộp 2 dataset thành một dataset để làm đầu vào cho bài toán dự đoán.

3. TRÍCH XUẤT ĐẶC TRƯNG

3.1. Lựa chọn đặc trưng

Việc lựa chọn đặc trưng phải làm sao thỏa mãn được cả 2 dataset crawl về từ 2 trang web khác nhau, nên dẫn đến nhiều khó khăn.

Vì dataset1(crawl từ trang web https://www.usedmotorcyclestore.com/) có số lượng mẫu lớn hơn(1066 mẫu), nên ta sẽ ưu tiên chọn các đặc trưng ở dataset1 có nhiều hơn.

Với dataset1 chọn các cột: Price(0.0% null), Displacement(21% null), Odometer (16% null), Stroke(20% null), Year(12% null), 'Weight, In Running Order (36% null)', Engine Torque(29% null), Fuel Capacity(20% null). Đa số các đặc trưng chọn ở trên có số lượng mẫu mất dữ liệu ít hơn 35%, riêng đặc trưng ‘Weight, In Running Order’ còn có ở cả dataset2 với đặc trưng tương ứng là ‘Empty Weight’ nên vẫn chọn dù có 36% số lượng mẫu là bị mất. Và các đặc trưng này đều là dạng số thập phân nhưng khi crawl data có cả đơn vị nên thành kiểu dữ liệu object trong Python có thể dễ dàng làm sạch.

3.2. Làm sạch và chuẩn hóa dữ liệu

3.2.1. Xử lí dữ liệu ở dataset1:

Đổi tên đặc trưng ‘Weight, In Running Order’ thành Weight để dễ dàng thao tác.

Xử lí các dữ liệu số nhưng do crawl data biến thành dữ liệu không chuẩn:

- Đối với đặc trưng price:
    - Xóa bỏ các kí tự '$' trước giá trị và thay dấu ',' thành dấu '.'.
    - Sau đó đọc các giá trị và chuyển kiểu về số thập phân.
- Đối với đặc trưng 'Displacement':
    - Xác định định dạng đơn vị phía sau các giá trị là : cc, cu in, cm3.
    - Đọc các giá trị trước đơn vị và đổi về chung 1 loại đơn vị là cc sau đó chuyển thành giá trị số thập phân.
- Đối với đặc trưng 'Stroke':
    - Xác định định dạng đơn vị phía sau các giá trị là : in và mm.
    - Đọc các giá trị trước đơn vị và đổi về chung 1 loại đơn vị là in sau đó chuyển thành giá trị số thập phân.
- Đối với đặc trưng 'Weight':
    - Xác định định dạng đơn vị phía sau các giá trị là : lb và lbs.
    - Đọc các giá trị trước đơn vị và đổi về chung 1 loại đơn vị là lb(pound) sau đó chuyển thành giá trị số thập phân.
- Đối với đặc trưng 'Engine Torque':
    - Xác định định dạng đơn vị phía sau các giá trị là : ft-lb, lb-ft, ft-lbs,FT-lbs, ft. Lbs., ft.lbs và Nm.
    - Đọc các giá trị trước đơn vị và đổi về chung 1 loại đơn vị là ft-lb(Foot-pound force) sau đó chuyển thành giá trị số thập phân.
- Đối với đặc trưng 'Fuel Capacity':
    - Xác định định dạng đơn vị phía sau các giá trị là : gal, gallons, us gallon và l.
    - Đọc các giá trị trước đơn vị và đổi về chung 1 loại đơn vị là gal(Gallon) sau đó chuyển thành giá trị số thập phân.

3.2.2. Xử lí dữ liệu ở dataset2:

Đổi tên đặc trưng 'Engine size', 'Mileage', 'Empty weight' thành 'Odometer', 'Displacement', 'Weight' để tương ứng với dataset1.

Xử lí các dữ liệu số nhưng do crawl data biến thành dữ liệu không chuẩn:

- Đối với đặc trưng price:
    - Đổi đơn vị của price về Dollars và chuyển về kiểu float.
- Đối với đặc trưng ‘Displacement’:
    - Xác định định dạng đơn vị phía sau các giá trị là : cc.
    - Đọc các giá trị trước đơn vị và đổi về chung 1 loại đơn vị là cc sau đó chuyển thành giá trị số thập phân.
- Đối với đặc trưng ‘Odometer’:
    - Xác định định dạng đơn vị phía sau các giá trị là :km.
    - Đọc các giá trị trước đơn vị và đổi về chung 1 loại đơn vị là km sau đó chuyển thành giá trị số thập phân.
- Đối với đặc trưng ‘Weight’:
    - Xác định định dạng đơn vị phía sau các giá trị là : kg.
    - Đọc các giá trị trước đơn vị và đổi về chung 1 loại đơn vị là lb(pound) sau đó chuyển thành giá trị số thập phân. [2]

3.2.3. Kết hợp 2 dataset và làm sạch dữ liệu, xử lí ngoại lệ:

Nối 2 dataset lại với nhau ta thu được 1 dataset duy nhất: *(*1464 rows x 8 columns)

```
RangeIndex: 1464 entries, 0 to 1463
Data columns (total 8 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   price           1404 non-null   float64
 1   Displacement    1141 non-null   float64
 2   Odometer        1288 non-null   float64
 3   Stroke          830 non-null    float64
 4   Year            934 non-null    float64
 5   Weight          724 non-null    float64
 6   Engine Torque   759 non-null    float64
 7   Fuel Capacity   832 non-null    float64
```

Hình 5. Dữ liệu thu thập sau khi xử lý thô

- Xử lí các mẫu có giá trị bị trống:

```python
def processing_technique(name_tech, columns):
    data_temp = data.copy()
    if name_tech == 'MEAN_VALUE':
        mean = data_temp[columns].mean()
        data_temp[columns] = data_temp[columns].fillna(mean)
    if name_tech == 'MEDIAN_VALUE':
        median = data_temp[columns].median()
        data_temp[columns] = data_temp[columns].fillna(median)
    if name_tech == 'MODE_VALUE':
        mode = data_temp[columns].mode()[0]
        data_temp[columns] = data_temp[columns].fillna(mode)
    if name_tech == 'RANDOM_VALUE':
        random = data_temp[columns].dropna().sample(n=data_temp[columns].isnull().sum(),random_state=0)
        random.index = data_temp[data_temp[columns].isnull()].index
        data_temp[columns].update(random)
    return data_temp

data = processing_technique("RANDOM_VALUE", "price")#phân phối chuẩn
data = processing_technique("RANDOM_VALUE", "Displacement")#phân phối lệch
data = processing_technique("MEAN_VALUE", "Odometer")#phân phối lệch
data = processing_technique("RANDOM_VALUE", "Stroke")#phân phối lệch
data = processing_technique("RANDOM_VALUE", "Year")#phân phối lệch
data = processing_technique("MEAN_VALUE", "Weight")#phân phối lệch
data = processing_technique("RANDOM_VALUE", "Engine Torque")#phân phối lệch
data = processing_technique("RANDOM_VALUE", "Fuel Capacity")#phân phối lệch
# data
data.info()
data.describe()
```

Hình 6. Minh họa code hàm xử lí các giá trị trống và cách xử lí giá trị với từng đặc trưng

- Kết quả quá trình xử lí các mẫu có giá trị bị trống:

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1464 entries, 0 to 1463
Data columns (total 8 columns):
 #   Column         Non-Null Count  Dtype
---  ------         --------------  -----
 0   price          1464 non-null   float64
 1   Displacement   1464 non-null   float64
 2   Odometer       1464 non-null   float64
 3   Stroke         1464 non-null   float64
 4   Year           1464 non-null   float64
 5   Weight         1464 non-null   float64
 6   Engine Torque  1464 non-null   float64
 7   Fuel Capacity  1464 non-null   float64
dtypes: float64(8)
memory usage: 91.6 KB
```

	price	Displacement	Odometer	Stroke	Year	Weight	Engine Torque	Fuel Capacity
count	1464.000000	1464.000000	1464.000000	1464.000000	1464.000000	1464.000000	1464.000000	1464.000000
mean	12848.499180	1311.561256	20357.466615	4.131847	2012.426230	754.832604	97.331307	5.106900
std	7668.102181	578.846829	18927.061804	0.529488	6.606509	104.116146	18.310157	1.042441
min	472.500000	0.000000	0.000000	1.560000	1976.000000	211.680000	0.000000	0.500000
25%	6999.000000	937.000000	6266.750000	3.962000	2008.000000	754.832604	92.000000	4.700000
50%	11549.475000	1573.152000	18165.500000	4.374000	2014.000000	754.832604	100.000000	5.000000
75%	16999.000000	1689.500000	27655.000000	4.380000	2018.000000	755.000000	110.000000	6.000000
max	44995.000000	7690.000000	195491.000000	4.500000	2022.000000	1243.000000	129.000000	10.600000

Hình 7. Mô tả dữ liệu sau khi đã xử lý xá dữ liệu có giá trị bị trống

- Xử lí giá trị ngoại lệ: Xác định phân phối của các đặc trưng sau đó áp dụng phương pháp thích hợp cho các phân phối.

```python
#True danh cho phan bo chuan, False danh cho phan bo Lech
def exception_handling(column, cd, data):
  if cd == True:
    upper_column = data[column].mean() + 3* data[column].std()
    lower_column = data[column].mean() - 3* data[column].std()
    data.loc[data[column] >= round(upper_column), column] = round(upper_column)
    data.loc[data[column] <= round(lower_column), column] = round(lower_column)
  else:
    q1, q3 = np.percentile(data[column], [25, 75])
    IQR = q3 - q1
    upper_column = q3 + 3 * IQR
    lower_column = q1 - 3 * IQR
    data.loc[data[column] >= upper_column, column] = round(upper_column)
    data.loc[data[column] <= lower_column, column] = round(lower_column)
  return data

data = exception_handling("price", True, data)#phân phối chuẩn
data = exception_handling("Displacement", False, data)#phân phối lệch
data = exception_handling("Odometer", False, data)#phân phối lệch
data = exception_handling("Stroke", False, data) #phân phối lệch
data = exception_handling("Year", False, data)#phân phối lệch
data = exception_handling("Weight", False, data)#phân phối lệch
data = exception_handling("Engine Torque", False, data)#phân phối lệch
data = exception_handling("Fuel Capacity", False, data)#phân phối lệch
```

Hình 8. Minh họa code hàm xử lí ngoại lệ dùng cho phân phôi chuẩn hoặc phân phối lệch.

- Kết quả sau khi xử lí ngoại lệ:

	price	Displacement	Odometer	Stroke	Year	Weight	Engine Torque	Fuel Capacity
count	1464.00000	1464.000000	1464.000000	1464.000000	1464.000000	1464.000000	1464.000000	1464.000000
mean	12792.56612	1306.447868	20213.593664	4.158845	2012.428962	754.924267	97.418056	5.106764
std	7474.31254	537.507702	18029.646605	0.444250	6.591831	0.706062	17.963383	1.035475
min	472.50000	0.000000	0.000000	2.769000	1978.000000	754.000000	38.000000	1.000000
25%	6999.00000	937.000000	6266.750000	3.962000	2008.000000	754.832604	92.000000	4.700000
50%	11549.47500	1573.152000	18165.500000	4.374000	2014.000000	754.832604	100.000000	5.000000
75%	16999.00000	1689.500000	27655.000000	4.380000	2018.000000	755.000000	110.000000	6.000000
max	35853.00000	3947.000000	91820.000000	4.500000	2022.000000	756.000000	129.000000	10.000000

Hình 9. Mô tả dữ liệu sau khi xử lý ngoại lệ

- Chuẩn hóa dữ liệu với chuẩn hóa Min-Max:

```python
min_max=MinMaxScaler()
df_minmax =pd.DataFrame(min_max.fit_transform(data),columns
=data.columns)
```

Hình 10. Minh họa code chuẩn hóa Min-max

- Kết quả sau khi chuẩn hóa Min-Max:

	price	Displacement	Odometer	Stroke	Year	Weight	Engine Torque	Fuel Capacity
count	1464.000000	1464.000000	1464.000000	1464.000000	1464.000000	1464.000000	1464.000000	1464.000000
mean	0.348216	0.330998	0.220144	0.802914	0.782476	0.462133	0.652946	0.456307
std	0.211255	0.136181	0.196359	0.256644	0.149814	0.353031	0.197400	0.115053
min	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
25%	0.184466	0.237395	0.068250	0.689197	0.681818	0.416302	0.593407	0.411111
50%	0.313081	0.398569	0.197838	0.927210	0.818182	0.416302	0.681319	0.444444
75%	0.467108	0.428047	0.301187	0.930676	0.909091	0.500000	0.791209	0.555556
max	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000

Hình 11. Mô tả dữ liệu sau khi chuẩn hóa

3.2.4. Kết quả của quá trình trên*:*

- Từ 2 dataset1 và dataset2 có nhiều loại là object:

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 1066 entries, 1 to 2131
Data columns (total 8 columns):
 #   Column         Non-Null Count  Dtype
---  ------         --------------  -----
 0   price          1066 non-null   object
 1   Displacement   847 non-null    object
 2   Odometer       893 non-null    float64
 3   Stroke         857 non-null    object
 4   Year           934 non-null    float64
 5   Weight         678 non-null    object
 6   Engine Torque  762 non-null    object
 7   Fuel Capacity  851 non-null    object
```

Hình 12. Mô tả tập dữ liệu dataset1 sau khi chọn đặc trưng ban đầu

```
---  ------         --------------  -----
0   price          398 non-null    object
1   Displacement   323 non-null    object
2   Odometer       395 non-null    object
3   Weight         48 non-null     object
```

Hình 13. Mô tả tập dữ liệu dataset2 sau khi chọn đặc trưng ban đầu

- Ta thu được 1 dataset duy nhất không còn dữ liệu trống và cùng 1 kiểu float:

```
RangeIndex: 1464 entries, 0 to 1463
Data columns (total 8 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   price           1464 non-null   float64
 1   Displacement    1464 non-null   float64
 2   Odometer        1464 non-null   float64
 3   Stroke          1464 non-null   float64
 4   Year            1464 non-null   float64
 5   Weight          1464 non-null   float64
 6   Engine Torque   1464 non-null   float64
 7   Fuel Capacity   1464 non-null   float64
```

Hình 14. Tập dữ liệu cuối cùng thu được

4. MÔ HÌNH HÓA DỮ LIỆU

4.1. Mô hình hồi quy tuyến tính đa biến (Multiple Regression Linear)

4.1.1. Cơ sở lý thuyết

"Hồi quy tuyến tính" là một phương pháp thống kê để hồi quy dữ liệu với biến phụ thuộc có giá trị liên tục trong khi các biến độc lập có thể có một trong hai giá trị liên tục hoặc là giá trị phân loại. Nói cách khác "Hồi quy tuyến tính" là một phương pháp để dự đoán biến phụ thuộc (Y) dựa trên giá trị của biến độc lập (X). Nó có thể được sử dụng cho các trường hợp chúng ta muốn dự đoán một số lượng liên tục. Ví dụ, dự đoán giao thông ở một cửa hàng bán lẻ, dự đoán thời gian người dùng dừng lại một trang nào đó hoặc số trang đã truy cập vào một website nào đó v.v...

Hiệu suất của mô hình được tính bằng R – Square (R^2)

$$R^2 = \frac{TSS - RSS}{TSS}$$

- Tổng các diện tích (TSS): TSS là một phép đo tổng biến thiên trong tỷ lệ đáp ứng / biến phụ thuộc Y và có thể được coi là số lượng biến thiên vốn có trong đáp ứng trước khi hồi quy được thực hiện.
- Sum of Squares (RSS): RSS đo lường lượng biến đổi còn lại không giải thích được sau khi thực hiện hồi quy.
- (TSS - RSS) đo lường mức độ thay đổi trong đáp ứng được giải thích (hoặc loại bỏ) bằng cách thực hiện hồi quy

Root Mean Square Error (RMSE) RMSE cho biết mức độ phân tán các giá trị dự đoán từ các giá trị thực tế. Công thức tính RMSE là:

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

Với n là tổng số quan sát

4.1.2. Mô hình hồi quy tuyến tính trong Python

Trong python, chúng ta có thể sử dụng thư viện sklearn để dự đoán hồi quy tuyến tính.

LinearRegression(*, fit_intercept=True, copy_X=True, n_jobs=None , positive=False)

Trong đó:

- fit_intercept (bool), default=True: Tính toán điểm chặn cho mô hình hay không
- copy_X (bool), default=True: Sao chép X
- positive (bool), default=False: Buộc các hệ số phải dương nếu là True

Câu lệnh tính R^2: r2 = r2_score(Y_test, predictions)

Câu lệnh tính RMSE: rmse = np.sqrt(mean_squared_error(Y_test, predictions))

4.2. Mô hình K-means

4.2.1. Cơ sở lý thuyết

K-means là một thuật toán phân cụm đơn giản thuộc loại học không giám sát (tức là dữ liệu không có nhãn) và được sử dụng để giải quyết bài toán phân cụm. Ý tưởng của thuật toán phân cụm k-means là phân chia 1 bộ dữ liệu thành các cụm khác nhau. Trong đó số lượng cụm được cho trước là k. Công việc phân cụm được xác lập dựa trên nguyên lý: Các điểm dữ liệu trong cùng 1 cụm thì phải có cùng 1 số tính chất nhất định. Tức là giữa các điểm trong cùng 1 cụm phải có sự liên quan lẫn nhau. Đối với máy tính thì các điểm trong 1 cụm đó sẽ là các điểm dữ liệu gần nhau.

Thuật toán phân cụm k-means thường được sử dụng trong các ứng dụng cỗ máy tìm kiếm, phân đoạn khách hàng, thống kê dữ liệu,…

Ý tưởng của thuật toán K-means:

- Khởi tạo K điểm dữ liệu trong bộ dữ liệu và tạm thời coi nó là tâm của các cụm dữ liệu của chúng ta.
- Với mỗi điểm dữ liệu trong bộ dữ liệu, tâm cụm của nó sẽ được xác định là 1 trong K tâm cụm gần nó nhất.
- Sau khi tất cả các điểm dữ liệu đã có tâm, tính toán lại vị trí của tâm cụm để đảm bảo tâm của cụm nằm ở chính giữa cụm.
- Bước 2 và bước 3 sẽ được lặp đi lặp lại cho tới khi vị trí của tâm cụm không thay đổi hoặc tâm của tất cả các điểm dữ liệu không thay đổi. [3]

4.2.2. Mô hình phân cụm K-means trong python

Trong python, chúng ta có thể sử dụng thư viện sklearn để phân cụm K-means.

MiniBatchKMeans(*n_clusters=8, max_iter=100, batch_size=1024, random_state=None,*)

Trong đó:

- n_clusters (int), default=8: Số lượng cụm hình thành
- max_iter (int), default = 100: Số lần lặp lại tối đa trên tập dữ liệu hoàn chỉnh trước khi dừng
- batch_size (int), default =1024: Kích thước của các lô nhỏ
- random_state (int), RandomState instance or None, default=None: Xác định tạo số ngẫu nhiên để khởi tạo điểm trung tâm và gán lại ngẫu nhiên.

4.3. Quá trình mô hình hóa dữ liệu

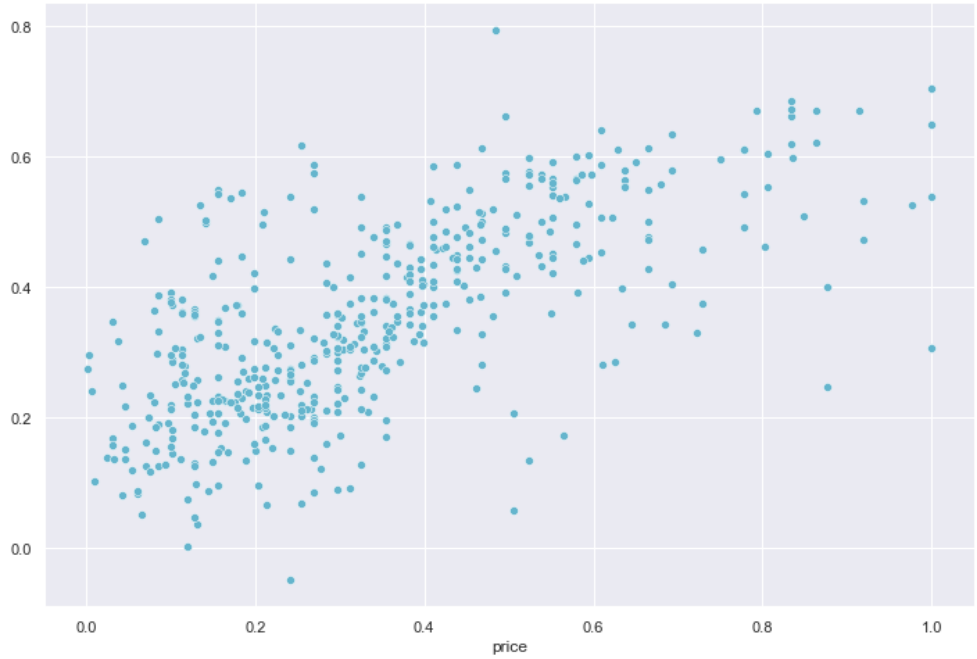
Sau khi làm sạch và chuẩn hóa dữ liệu, nhóm đã áp dụng mô hình hồi quy tuyến tính đa biến vào dữ liệu để dự đoán.

Dữ liệu được chia thành:

1. 67% thuộc tập huấn luyện: 980 hàng, 7 cột
2. 33% thuộc tập kiểm thử: 484 hàng, 7 cột

Khi áp dụng mô hình dự đoán vào tập dữ liệu thì nhận được các kết quả sau:

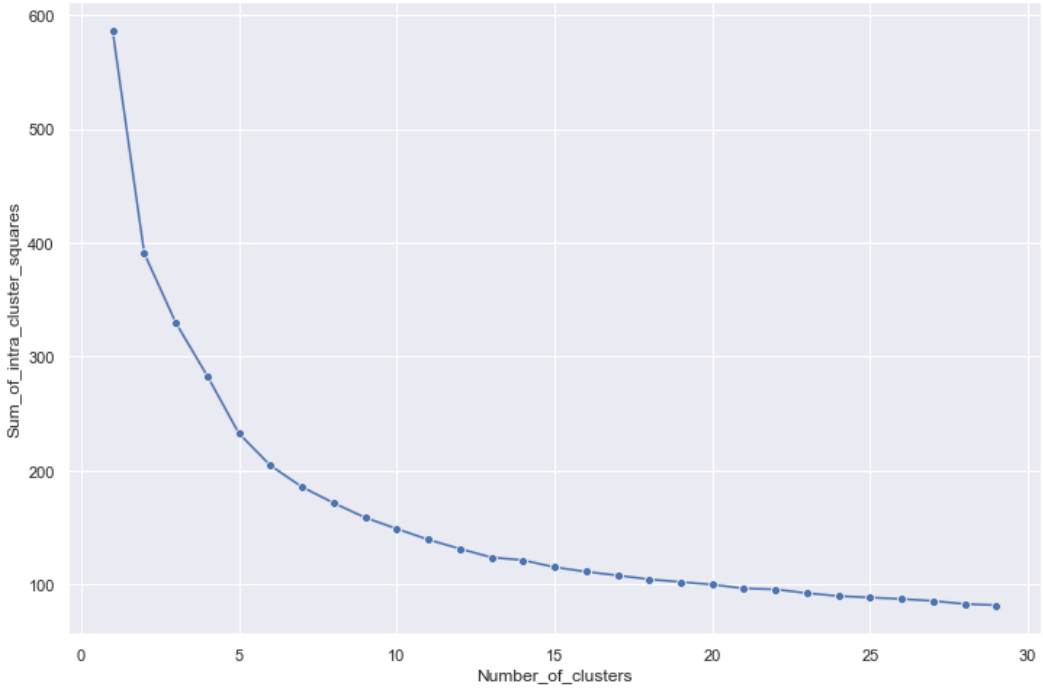
- RMSE: 0.1546
- R^2: 45.78 %



Hình 15. Mô tả kết quả dự đoán khi chưa áp dụng phân cụm

Kết quả không được cao, nên nhóm quyết định cải thiện bằng cách sử dụng mô hình phân cụm K-means trước khi dự đoán để tăng thêm dữ liệu dự đoán.

Đầu tiên, khảo sát dữ liệu để chọn số cụm. Nhóm sử dụng phương pháp Elbow: Chọn số cụm sao cho khi tăng giá trị cụm thì trung bình cộng khoảng cách từ các điểm xung quanh đến cụm không giảm hoặc giảm không đáng kể.



Hình 16. Đồ thị khảo sát số cụm

Dựa vào đồ thị khảo sát, chọn k = 20. Sau đó, thực hiện phân cụm với số cụm là 20.

Kết quả sau khi phân cụm:

	price	Displacement	Odometer	Fuel Economy	Year	Weight	Engine Torque	Fuel Capacity	k_means_clusters
0	0.325674	0.031670	0.000381	0.619724	1.000000	0.416302	0.442066	0.555556	4
1	0.890957	0.473271	0.000054	0.500000	1.000000	1.000000	0.923077	0.555556	2
2	0.947485	0.473271	0.000882	0.500000	1.000000	1.000000	0.879121	0.555556	2
3	0.325674	0.487205	0.014746	1.000000	1.000000	0.000000	0.956044	0.444444	11
4	0.198598	0.063086	0.221711	0.619724	1.000000	0.416302	0.600000	0.071111	4
...	...	...	...	...	...	...	...	...	...
1459	0.031161	0.365355	0.016336	0.619724	0.659091	0.416302	0.442066	0.444444	7
1460	0.135032	0.101343	0.100196	0.619724	0.863636	0.416302	0.628571	0.255556	4
1461	0.004452	0.038004	0.087127	0.619724	0.840909	0.416302	0.956044	0.555556	17
1462	0.117226	0.252850	0.118711	0.619724	0.590909	0.000000	0.600000	0.533333	9
1463	0.120193	0.126678	0.800479	0.619724	0.590909	0.416302	0.668132	0.444444	12

1464 rows × 9 columns

Hình 17. Mô tả tập dữ liệu sau khi được phân cụm

Mã hóa cụm thành các cột dữ liệu điều kiện.

k_means_clusters	k_means_clusters_0	...	k_means_clusters_10	k_means_clusters_11	k_means_clusters_12
4	0	...	0	0	0
2	0	...	0	0	0
2	0	...	0	0	0
11	0	...	0	1	0
4	0	...	0	0	0
...	...	...	...	...	...
7	0	...	0	0	0
4	0	...	0	0	0
17	0	...	0	0	0
9	0	...	0	0	0
12	0	...	0	0	1

Hình 18. Mã hóa các cụm thành các cột dữ liệu

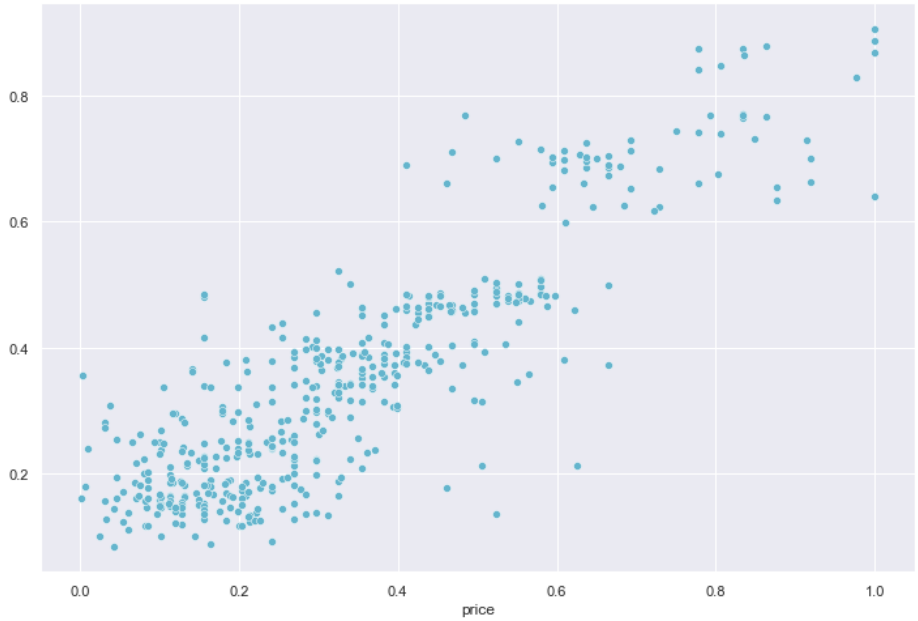
Sau đó, sử dụng mô hình hồi quy tuyến tính đa biến để dự đoán thì được kết quả như sau:

- Khi so sánh ngẫu nhiên giá xe dự đoán và thực tế:

	Target	Prediction
599	0.523523	0.136928
709	0.325645	0.342507
797	0.283278	0.320125
258	0.664844	0.689442
994	0.291757	0.402010
887	0.240881	0.092882
1319	0.728253	0.623184
1456	0.120193	0.295944
941	0.240881	0.173759
718	0.433078	0.373509

Hình 19. So sánh kết quả dự đoán và kết quả thực tế

- RMSE: 0.1021
- R^2: 76.36 %



Hình 20. Mô tả kết quả dự đoán sau khi áp dụng phân cụm

Có thể thấy, hiệu suất của mô hình đã được cải thiện đáng kể, tăng ~30%.

5. KẾT LUẬN

5.1. Mô hình đã sử dụng

- Linear Regression
- K-means

5.2. Kết quả

Đã xây dựng thành công mô hình dự đoán giá xe máy cũ với độ chính xác khá cao 76,36%

5.3. Hướng phát triển

Sau khi xây dựng mô hình, nhóm đã có hướng phát triển để nâng cao độ chính xác dự đoán như sau:

- Thu thập dữ liệu ở nhiều sàn khác nhau và uy tín hơn
- Sử dụng nhiều trường dữ liệu hơn để dự đoán
- Tăng cường các phương pháp làm sạch và chuẩn hóa dữ liệu
- Sử dụng các mô hình dự đoán khác như: Hồi quy Logistic (Logistic Regression), Stepwise Regression, …

TÀI LIỆU THAM KHẢO

[1] Baiju Muthukadan, Selenium with Python

< https://selenium-python.readthedocs.io/?fbclid=IwAR01nCiVpWUoidVwygRnO6C-UHIm1Te81nlgFEIYuKBu1w5GbYaBp6ckjFQ>

[2] W3School, Python RegEx

<https://www.w3schools.com/python/python_regex.asp?fbclid=IwAR1t4B3Yv0r--xYqoFqT-ERXWpE1u4RSZ-ENrOXxHyK56hqgfz-JwcUlBh8>

[3] Nguyễn Văn Hiếu, Thuật toán K-Means (K-Means clustering) và ví dụ

<https://nguyenvanhieu.vn/thuat-toan-phan-cum-k-means/#:~:text=Thu%E1%BA%ADt%20to%C3%A1n%20ph%C3%A2n%20c%E1%BB%A5m%20k%2Dmeans%20l%C3%A0%20m%E1%BB%99t%20ph%C6%B0%C6%A1ng%20ph%C3%A1p,s%E1%BB%AD%20thu%E1%BB%99c%20v%E1%BB%81%20nh%C3%B3m%20n%C3%A0o.>

[4] scikit-learn, sklearn.cluster.MiniBatchKMeans

<https://scikit-learn.org/stable/modules/generated/sklearn.cluster.MiniBatchKMeans.html >

[5] NguyenDuong (2017), Linear Regression - Hồi quy tuyến tính trong Machine Learning

<https://viblo.asia/p/linear-regression-hoi-quy-tuyen-tinh-trong-machine-learning-4P856akRlY3>

[6] scikit-learn, sklearn.linear_model.LinearRegression

<https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LinearRegression.html?highlight=linear#sklearn.linear_model.LinearRegression>