ĐẠI HỌC QUỐC GIA TP.HCM
TRƯỜNG ĐAI HỌC CÔNG NGHỆ THÔNG TIN

KHOA: HỆ THỐNG THÔNG TIN

----📖-----

BÁO CÁO ĐỒ ÁN

KHO DỮ LIỆU VÀ OLAP

Đề tài:

GIÁO VIÊN HƯỚNG DẪN:

KS. Phạm Nguyễn Thanh Bình

SINH VIÊN THỰC HIỆN:

Phùng Thiên Phúc – 21521297

Phạm Mạnh Hùng – 21520901

Nguyễn Quang Lâm - 21521057

Thành Phố Hồ Chí Minh, Tháng 5 Năm 2024

NHẬN XÉT CỦA GIÁO VIÊN HƯỚNG DẪN

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

……
……
……
……
……
……
……
……
……
……
……
……
……
……
……
……
……
……
……

……., ngày……...tháng……năm 2024

Người nhận xét

(Ký tên và ghi rõ họ tên)

LỜI CẢM ƠN

Lời đầu tiên, nhóm chúng em xin gửi lời cảm ơn chân thành đến toàn thể giảng viên trường Đại học Công nghệ thông tin – Đại học Quốc gia TP.HCM cũng như là nhà trường vì đã giúp nhóm em có những kiến thức cơ bản làm nền tảng để thực hiện đồ án môn học này.

Đặc biệt hơn nữa, chúng em xin gửi lời cảm ơn chân thành đến thầy Phạm Nguyễn Thanh Bình (Giảng viên thực hành môn Khai thác dữ liệu). Thời gian vừa qua, thày đã trực tiếp giảng dạy, truyền đạt kinh nghiệm, hướng dẫn chúng em một cách tận tình giúp nhóm chúng em hoàn thành tốt đồ án môn học của mình. Chúc thầy sẽ luôn dồi dào sức khỏe, tràn đầy nhiệt huyết để có thể tiếp tục giảng dạy, dìu dắt những thế hệ sinh viên tiếp theo.

Ngoài ra, chúng em cũng gửi lời cảm ơn đến tập thể lớp IS252.O21vì thời gian qua đã đồng hành cùng nhau. Cùng nhau học tập, tranh luận một cách sôi nổi để xây dựng bài học một cách tốt nhất. Cảm ơn các bạn đã cùng thảo luận, đánh giá và đóng góp ý kiến, cùng học hỏi nghiên cứu để thực hiện đồ án một cách tốt nhất có thể.

Mặc dù đã vận dụng tối đa những gì đã học được nhưng chúng em vẫn khó có thể tránh khỏi những sai sót. Chính vì vậy, nhóm em rất mong nhận được sự góp ý từ phía thầy để có thể hoàn thiện một cách tốt nhất có thể. Qua đó cũng tích lũy và học hỏi kinh nghiệm để làm hành trang cho tương lai.

Chúng em xin hết và xin một lần nữa gửi lời cảm ơn chân thành đến với thầy!

Chương 1. Nhận diện bài toán khai thác dữ liệu

1. Tổng quan về dữ liệu

a. Lý do chọn đề tài

Trong những năm qua với sự phát triển của thị trường công nghiệp oto trên thế giới nói chung và Việt Nam nói riêng đã góp phần vào sự phát triển nền kinh tế toàn cầu. hiểu được vấn đề đó. Nhóm đã quyết định xây dựng chọn đề tài "Dự đoán giá bán xe thực tế trên thị trường"

Hơn nữa, trong thời đại số hóa hiện nay, việc ứng dụng các kỹ thuật khai thác dữ liệu và học máy để dự đoán thị trường không chỉ mang lại lợi ích về mặt lý thuyết mà còn có giá trị thực tiễn cao. Các doanh nghiệp có thể sử dụng những kết quả từ nghiên cứu này để điều chỉnh chiến lược kinh doanh, tối ưu hóa sản xuất và phân phối, cũng như cải thiện dịch vụ khách hàng.

Đồng thời, những thông tin dự đoán này cũng hỗ trợ người tiêu dùng trong việc đưa ra quyết định mua sắm thông minh và hiệu quả hơn.

Nhóm mong muốn đề tài này không chỉ mang lại những hiểu biết sâu sắc về thị trường ô tô mà còn cung cấp các công cụ phân tích mạnh mẽ để dự đoán và thích ứng với những thay đổi trong tương lai, qua đó góp phần vào sự phát triển bền vững của ngành công nghiệp ô tô.

b. Giới thiệu nguồn dữ liệu

Dataset được lấy từ nền tảng Kaggle, được xây dựng bởi SYED ANWAR. Được cập nhật gần nhất vào tháng 2/2024.

Dataset gồm có 16 cột và 558844 dòng về thông tin, giá bán, hang sản xuất

Dataset car_prices cung cấp một bộ sưu tập thông tin toàn diện liên quan đến các giao dịch bán các loại xe khác nhau. Bộ dữ liệu này bao gồm các chi tiết

như năm, hãng sản xuất, mẫu xe, trang trí, loại thân xe, loại hộp số, số VIN (Số nhận dạng xe), trạng thái đăng ký, xếp hạng tình trạng, đọc đồng hồ đo đường, màu sắc ngoại thất và nội thất, thông tin người bán, Báo cáo thị trường Manheim (MMR), giá bán và ngày bán.

Link tải dataset: https://www.kaggle.com/datasets/syedanwarafridi/vehicle-sales-data

2. Mô tả thuộc tính

STT	Tên thuộc tính	Ý nghĩa	Kiểu dữ liệu
1	Year	Năm sản xuất	
2	Make	Tên hãng sản xuất xe	
3	Model	Mẫu xe cụ thể do hang sản xuất	
4	Trim	Mỗi mẫu xe sẽ có các phiên bản khác nhau, mỗi phiên bản sẽ được thể hiện trong cột trim	
5	Body	Loại than xe, vd như SUV, Sedan, coupe…	
6	Transmission	Hộp số của xe. Có giá trị có thể là tự động (automatic) hoặc số sàn(manual)	
7	Vin	Số nhận dạng xe (Vehicle Identification Number). Đây là mã số duy nhất cho mỗi xe và	

		có thể dùng để tra cứu thông tin chi tiết về xe.	
8	State	Bang hoặc khu vực nơi xe được bán.	
9	Condition	Tình trạng của xe, ví dụ như mới, đã qua sử dụng, hoặc bị hư hỏng.	
10	Odometer	Số dặm hoặc số km mà xe đã chạy. Odometer càng cao thường cho thấy xe đã qua sử dụng nhiều và có thể giảm giá trị.	
11	Color	Màu sơn bên ngoài của xe.	
12	interior	Màu hoặc tình trạng của nội thất xe.	
13	Seller	Người bán hoặc đại lý bán xe.	
14	Mmr	Giá trị thị trường dự đoán (Manheim Market Report) của xe. Đây là giá trị tham khảo để đánh giá giá trị thị trường của xe, giúp phân tích xu hướng và sự biến động của thị trường.	

15	Sellingprice	Giá bán thực tế của xe trong giao dịch. Giá này cho biết mức giá mà người mua đã trả cho xe.	
16	saledate	Ngày diễn ra giao dịch bán xe. Ngày này có thể dùng để phân tích xu hướng bán hàng theo thời gian.	

3. Phát biểu bài toán

Dựa vào những thuộc tính của dataset, chúng ta sẽ dự đoán xu hướng mua xe của người tiêu dùng để giúp các đại lý bán xe đề ra chiến lược nhập hàng bán và cho các doanh nghiệp vạch ra chiến lược sản xuất

4. Công cụ khai thác dữ liệu

Các công cụ khai thác dữ liệu:

- Google colab
- Anaconda – Jupiter notebook
- Python

5. Thư viện kèm theo

- **NumPy:** được sử dụng để thực hiện các phép tính số hiệu quả và làm việc với mảng đa chiều.
- **Pandas:** được sử dụng để thao tác, tiền xử lý và phân tích dữ liệu, cung cấp các cấu trúc và chức năng dữ liệu mạnh mẽ.
- **Seaborn:** được sử dụng để tạo trực quan hóa dữ liệu mang tính thông tin, chất lượng cao, xây dựng trên thư viện matplotlib.
- **Matplotlib:** là một thư viện toàn diện để tạo trực quan hóa dữ liệu một cách tĩnh, hoạt họa và tương tác trong môi trường Python.

- **Scikit-learn (sklearn):** là một thư viện máy học cung cấp nhiều thuật toán học có giám sát và không giám sát, cũng như các công cụ để tiền xử lý dữ liệu và đánh giá mô hình.

Chương 2. Trực quan hóa dữ liệu

1. Thống kê hãng xe nào có doanh số bán xe nhiều nhất

Tính tổng doanh số bán xe của từng hãng

```
[*]: # tính tổng doanh số bán xe theo từng hãng
tong_doanh_so_tung_hang = df.groupby('make')['sellingprice'].sum().reset_index()
```

Chúng ta thực hiện đổi tên cột make trong biến tong_doanh_so_tung_hang thành total_sales để dễ hình dung

```
[12]: tong_doanh_so_tung_hang.columns = ['make', 'total_sales']
```

Thực hiện sắp xếp hang xe theo thứ tự giảm dần theo giá trị total_sales

```
[13]: tong_doanh_so_sapxep = tong_doanh_so_tung_hang.sort_values(by='total_sales', ascending= False)
```

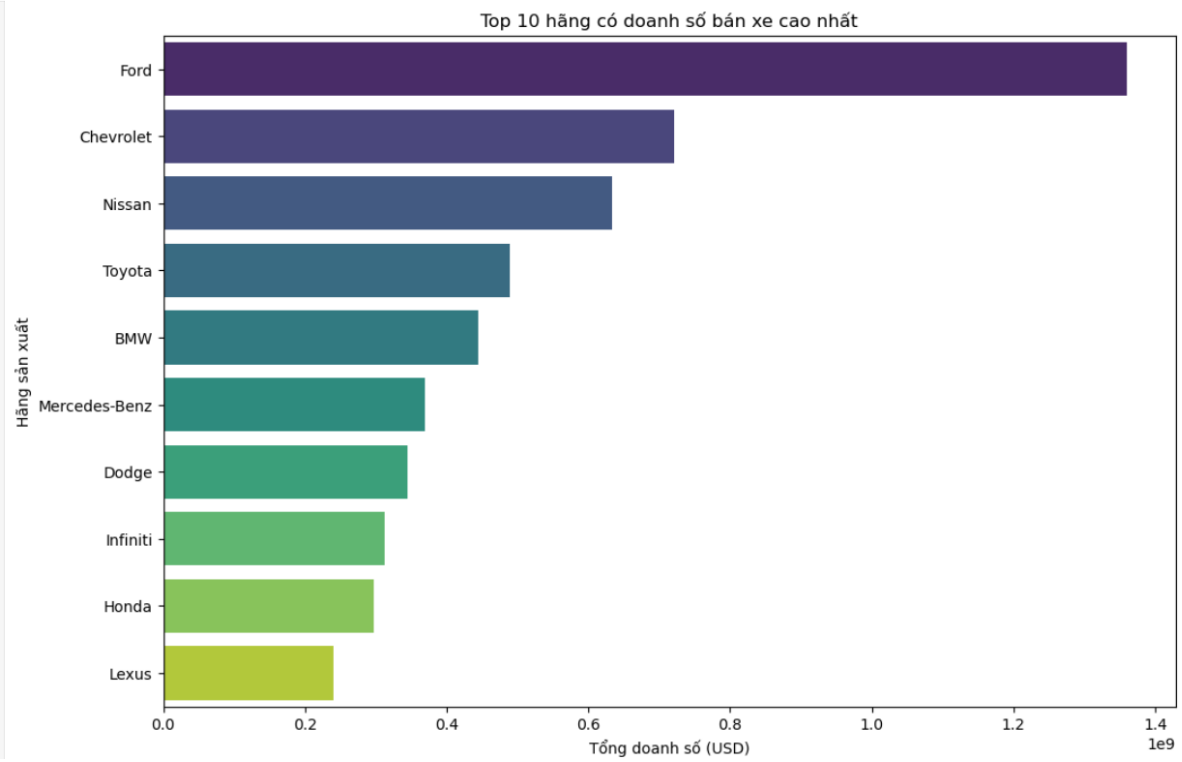
Gán 10 giá trị lớn nhất vào biến gia_tri_x

```
[15]: gia_tri_x = tong_doanh_so_sapxep.head(10)
```

Thực hiện vẽ biểu đồ cột

```
[16]: # Vẽ biểu đồ cột
plt.figure(figsize=(12, 8))
sns.barplot(x='total_sales', y='make', data=gia_tri_x, palette='viridis')
plt.title('Top 10 hãng có doanh số bán xe cao nhất')
plt.xlabel('Tổng doanh số (USD)')
plt.ylabel('Hãng sản xuất')
plt.show()
```

Kết quả chúng ta có thể thấy hãng xe ford có doanh thu cao nhất(gấp đôi hãng Chevrolet đứng thứ 2). Đứng cuối bảng xếp hạng là Honda và Lexus



2. Trực quan hóa được xu hướng mua xe của người tiêu dùng từ năm 2005 đến 2015

Trích xuất năm từ cột saledate

```
df['year'] = df['saledate'].str.extract(r'(\d{4})')
```

Xử lý các giá trị NaN trước khi chuyển đổi sang kiểu số nguyên

```
df['year'] = df['year'].fillna(0).astype(int)
```

Loại bỏ các hàng có giá trị năm không hợp lệ (ví dụ: năm 0)

```
df = df[df['year'] != 0]
```

Lọc dữ liệu từ năm 2005 đến 2015

```
df_filtered = df[(df['year'] >= 2005) & (df['year'] <= 2015)]
```

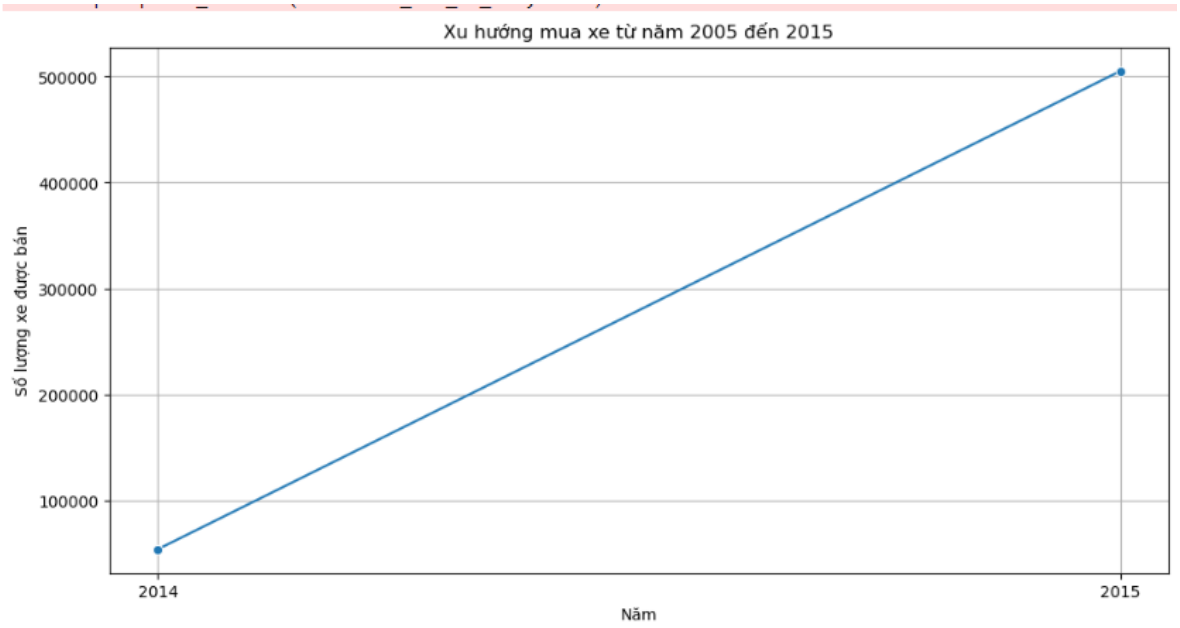
Nhóm dữ liệu theo năm và đếm số lượng xe được bán ra mỗi năm

```
sales_by_year = df_filtered.groupby('year').size().reset_index(name='sales_count')
```

Vẽ biểu đồ đường

```
plt.figure(figsize=(12, 6))
sns.lineplot(x='year', y='sales_count', data=sales_by_year, marker='o')
plt.title('Xu hướng mua xe từ năm 2005 đến 2015')
plt.xlabel('Năm')
plt.ylabel('Số lượng xe được bán')
plt.xticks(sales_by_year['year'])  # Đảm bảo các năm được hiển thị đầy đủ
plt.grid(True)
plt.show()
```

Kết quả



3. Phân tích giá bán theo theo từng địa điểm của 10 hãng sản xuất oto có doanh số bán xe nhiều nhất

Tính doanh số bán xe theo từng hãng

```
[10]: total_sales_by_make = df.groupby('make')['sellingprice'].sum().reset_index()
      total_sales_by_make.columns = ['make', 'total_sales']
      total_sales_by_make = total_sales_by_make.sort_values(by='total_sales', ascending=False)
```

Chọn ra 10 hãng có doanh số cao nhất và lưu nó vào biến top_10_makes

```
[11]: # Chọn ra 10 hãng có doanh số bán xe cao nhất
      top_10_makes = total_sales_by_make.head(10)['make']
```

Lọc dữ liệu để chỉ bao gồm các hãng xe nằm trong top 10.

[12]:
```
# Lọc dữ liệu để chỉ bao gồm các hãng xe nằm trong top 10
filtered_df = df[df['make'].isin(top_10_makes)]
```

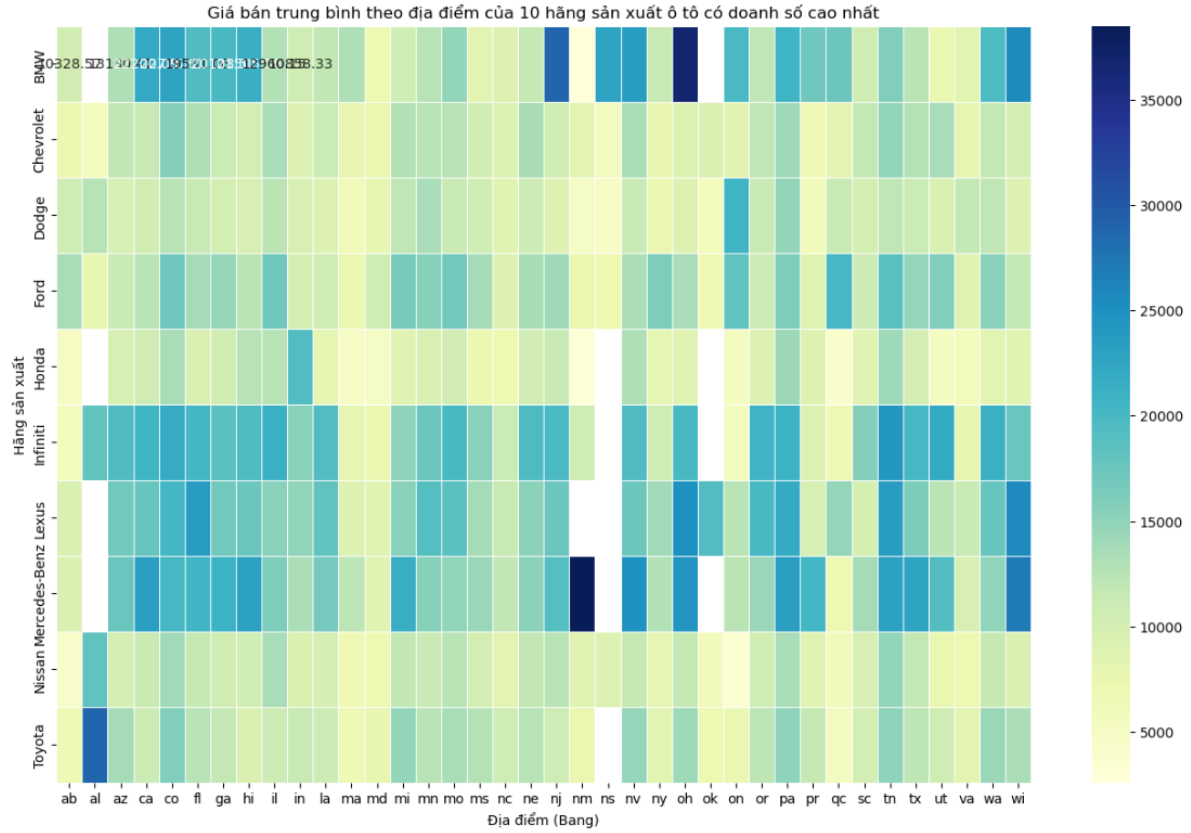
Nhóm dữ liệu theo hãng sản xuất và địa điểm, sau đó tính giá bán trung bình.

[15]:
```
# Nhóm dữ liệu theo hãng sản xuất và địa điểm, sau đó tính giá bán trung bình
avg_price_by_make_and_state = filtered_df.groupby(['make', 'state'])['sellingprice'].mean().unstack()
```

Trực quan hóa kết quả bằng biểu đồ hộp (box plot)

[16]:
```
# Trực quan hóa bằng biểu đồ nhiệt
plt.figure(figsize=(16, 10))
sns.heatmap(avg_price_by_make_and_state, annot=True, fmt=".2f", cmap="YlGnBu", linewidths=.5)
plt.title('Giá bán trung bình theo địa điểm của 10 hãng sản xuất ô tô có doanh số cao nhất')
plt.xlabel('Địa điểm (Bang)')
plt.ylabel('Hãng sản xuất')
plt.show()
```

Kết quả

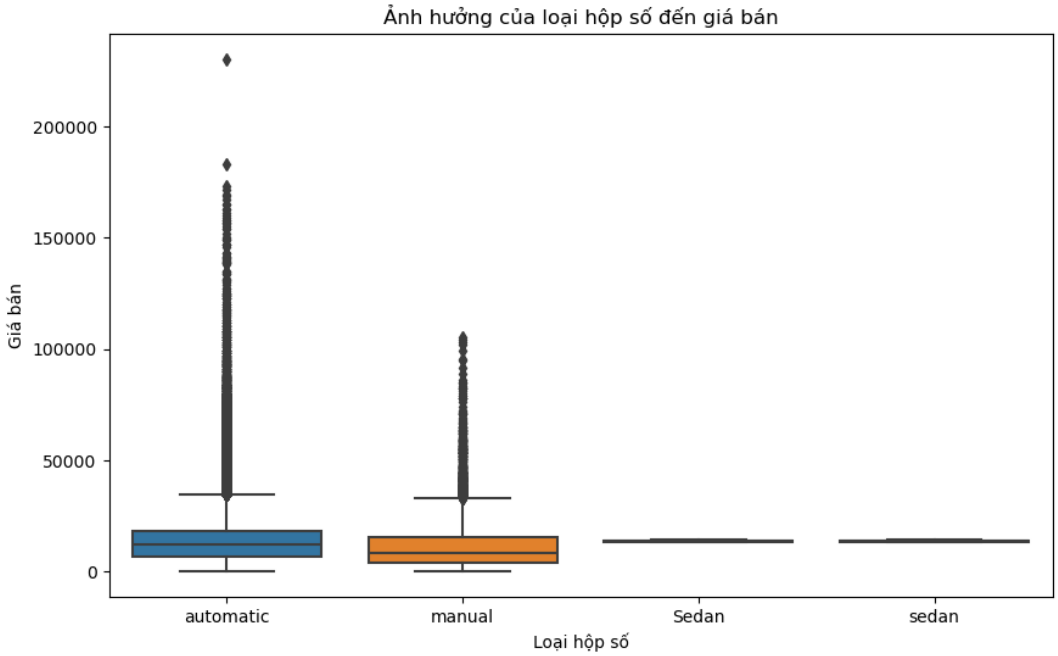


4. Ảnh hưởng của loại hộp số đến giá bán

Vẽ biểu đồ boxplot trực quan hóa ảnh hưởng của odoometer

```
[8]: plt.figure(figsize=(10, 6))
     sns.boxplot(x='transmission', y='sellingprice', data=df)
     plt.title('Ảnh hưởng của loại hộp số đến giá bán')
     plt.xlabel('Loại hộp số')
     plt.ylabel('Giá bán')
     plt.show()
```

Kết quả



5. Giá trị thị trường dự đoán (MMR) so với giá bán thực tế

Tính tổng doanh thu cho mỗi hãng xe

```
# Bước 1: Tính tổng doanh thu cho mỗi hãng xe
total_revenue = df.groupby('make')['sellingprice'].sum().reset_index()
```

Lọc ra 10 hãng xe có doanh thu cao nhất

```
# Bước 2: Lọc ra 10 hãng xe có doanh thu cao nhất
top_10_makes = total_revenue.nlargest(10, 'sellingprice')['make']
```

Tính giá trị trung bình của MMR và giá bán thực tế cho các hãng xe này

```
# Bước 3: Tính giá trị trung bình của MMR và giá bán thực tế cho các hãng xe này
top_10_data = df[df['make'].isin(top_10_makes)]
avg_prices = top_10_data.groupby('make').agg({'mmr': 'mean', 'sellingprice': 'mean'}).reset_index()
```

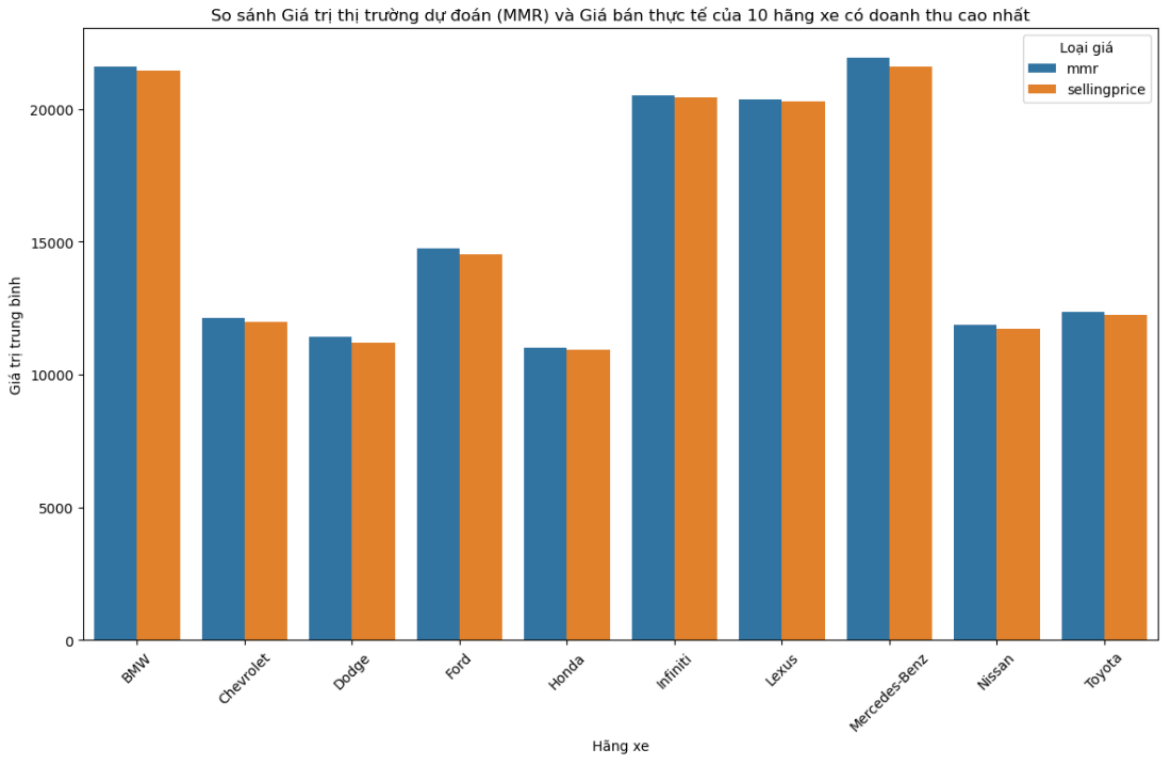
Vẽ biểu đồ cột để so sánh giá trị MMR trung bình và giá bán thực tế trung bình

```
# Bước 4: Vẽ biểu đồ cột để so sánh giá trị MMR trung bình và giá bán thực tế trung bình
plt.figure(figsize=(14, 8))

# Tạo hai cột cho MMR và Selling Price
avg_prices_melted = avg_prices.melt(id_vars='make', value_vars=['mmr', 'sellingprice'],
                                    var_name='Price Type', value_name='Price')

# Vẽ biểu đồ cột
sns.barplot(x='make', y='Price', hue='Price Type', data=avg_prices_melted)
plt.title('So sánh Giá trị thị trường dự đoán (MMR) và Giá bán thực tế của 10 hãng xe có doanh thu cao nhất')
plt.xlabel('Hãng xe')
plt.ylabel('Giá trị trung bình')
plt.xticks(rotation=45)
plt.legend(title='Loại giá', loc='upper right')
plt.show()
```

Kết quả



6. Tình trạng xe ảnh hưởng đến giá bán:

Loại bỏ các hàng có giá trị thiếu trong cột ‘condition’ hoặc ‘sellingprice’

```
[19] # Loại bỏ các hàng có giá trị thiếu trong cột 'condition' hoặc 'sellingprice'
     cleaned_car_prices = df.dropna(subset=['condition', 'sellingprice'])
```

Kết quả

cleaned_car_prices

	year	make	model	trim	body	transmission	vin	state	condition	odometer	color	interior	seller	mmr	sellingprice	saledate
0	2015	Kia	Sorento	LX	SUV	automatic	5xyktca69fg566472	ca	5.0	16639.0	white	black	kia motors america inc	20500.0	21500.0	Tue Dec 16 2014 12:30:00 GMT-0800 (PST)
1	2015	Kia	Sorento	LX	SUV	automatic	5xyktca69fg561319	ca	5.0	9393.0	white	beige	kia motors america inc	20800.0	21500.0	Tue Dec 16 2014 12:30:00 GMT-0800 (PST)
2	2014	BMW	3 Series	328i SULEV	Sedan	automatic	wba3c1c51ek116351	ca	45.0	1331.0	gray	black	financial services remarketing (lease)	31900.0	30000.0	Thu Jan 15 2015 04:30:00 GMT-0800 (PST)
3	2015	Volvo	S60	T5	Sedan	automatic	yv1612tb4f1310987	ca	41.0	14282.0	white	black	volvo na rep/world omni	27500.0	27750.0	Thu Jan 29 2015 04:30:00 GMT-0800 (PST)
4	2014	BMW	6 Series Gran Coupe	650i	Sedan	automatic	wba6b2c57ed129731	ca	43.0	2641.0	gray	black	financial services remarketing (lease)	66000.0	67000.0	Thu Dec 18 2014 12:30:00 GMT-0800 (PST)
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
40068	2014	Ford	Taurus	Limited	Sedan	automatic	1fahp2f81eg180004	tn	34.0	16533.0	—	gray	avis budget group	20500.0	19400.0	Tue Dec 23 2014 10:30:00 GMT-0800 (PST)
40069	2014	Ford	Focus	SE	Sedan	automatic	1fadp3f29el302602	tx	41.0	24382.0	black	black	avis budget group	12000.0	13600.0	Tue Jan 07 2014 11:00:00 GMT-0800 (PST)
40070	2014	Ford	Taurus	SEL	Sedan	NaN	1fahp2h88eg138877	qc	43.0	18239.0	silver	black	ford motor company of canada limited	21900.0	22500.0	Tue Feb 10 2015 02:00:00 GMT-0800 (PST)
40071	2014	Ford	Mustang	GT	Convertible	automatic	1zvbp8ff1e5253205	fl	48.0	6448.0	blue	black	tdaf remarketing	27100.0	26000.0	Mon Dec 22 2014 09:00:00 GMT-0800 (PST)
40072	2014	Ford	Taurus	Limited	Sedan	automatic	1fahp2f85eg172066	tx	37.0	33630.0	—	black	avis budget group	18150.0	17800.0	Tue Dec 23 2014 10:30:00 GMT-0800 (PST)

Vẽ biểu đồ phân tán để cho thấy ảnh hưởng của tình trạng xe đến giá bán

- Thiết lập kiểu của biểu đồ:

```
[20] # Thiết lập kiểu của biểu đồ
     sns.set(style="whitegrid")
```

2 giây

- Tạo biểu đồ phân tán

```
# Tạo biểu đồ phân tán
plt.figure(figsize=(12, 6))
sns.scatterplot(x='condition', y='sellingprice', data=cleaned_car_prices, alpha=0.5)
```

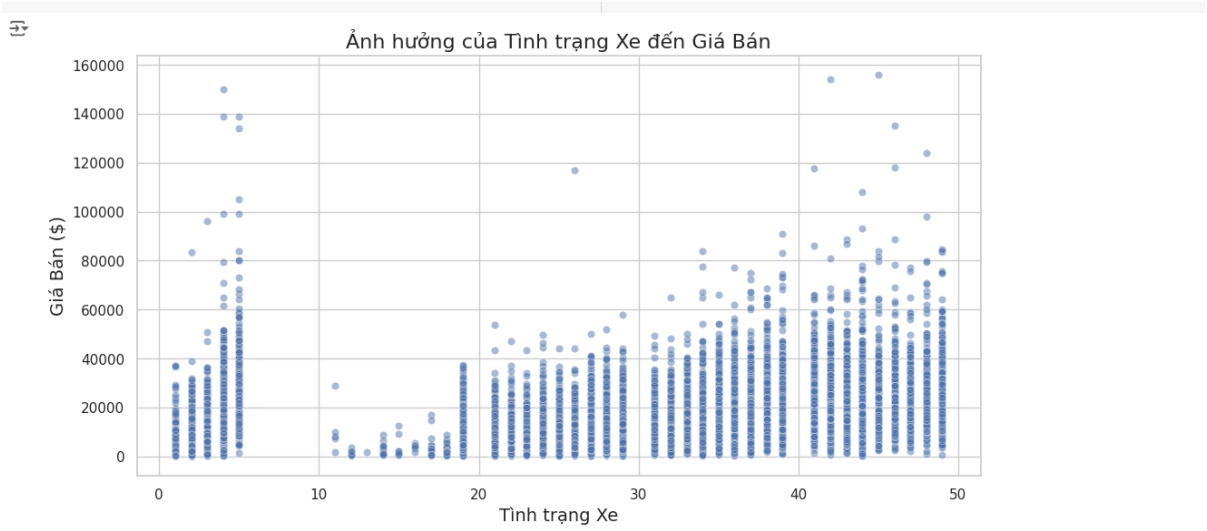
- Thiết lập tiêu đề và nhãn cho trục biểu đồ

```
# Thiết lập tiêu đề và nhãn cho trục
plt.title('Ảnh hưởng của Tình trạng Xe đến Giá Bán', fontsize=16)
plt.xlabel('Tình trạng Xe', fontsize=14)
plt.ylabel('Giá Bán ($)', fontsize=14)
```

- Hiển thị biểu đồ

```
# Hiển thị biểu đồ
plt.show()
```

Kết quả



7. Ảnh hưởng của số km đã đi (odometer) đến giá bán:

Loại bỏ các giá trị thiếu trong cột odometer và sellingprice

```
[22] # Loại bỏ các hàng có giá trị thiếu trong cột 'odometer' hoặc 'sellingprice'
     cln = df.dropna(subset=['odometer', 'sellingprice'])
```

Kết quả

```
cln
```

	year	make	model	trim	body	transmission	vin	state	condition	odometer	color	interior	seller	mmr	sellingprice	saledate
0	2015	Kia	Sorento	LX	SUV	automatic	5xyktca69fg566472	ca	5.0	16639.0	white	black	kia motors america inc	20500.0	21500.0	Tue Dec 16 2014 12:30:00 GMT-0800 (PST)
1	2015	Kia	Sorento	LX	SUV	automatic	5xyktca69fg561319	ca	5.0	9393.0	white	beige	kia motors america inc	20800.0	21500.0	Tue Dec 16 2014 12:30:00 GMT-0800 (PST)
2	2014	BMW	3 Series	328i SULEV	Sedan	automatic	wba3c1c51ek116351	ca	45.0	1331.0	gray	black	financial services remarketing (lease)	31900.0	30000.0	Thu Jan 15 2015 04:30:00 GMT-0800 (PST)
3	2015	Volvo	S60	T5	Sedan	automatic	yv1612tb4f1310987	ca	41.0	14282.0	white	black	volvo na rep/world omni	27500.0	27750.0	Thu Jan 29 2015 04:30:00 GMT-0800 (PST)
4	2014	BMW	6 Series Gran Coupe	650i	Sedan	automatic	wba6b2c57ed129731	ca	43.0	2641.0	gray	black	financial services remarketing (lease)	66000.0	67000.0	Thu Dec 18 2014 12:30:00 GMT-0800 (PST)
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
40069	2014	Ford	Focus	SE	Sedan	automatic	1fadp3f29el302602	tx	41.0	24382.0	black	black	avis budget group	12000.0	13600.0	Tue Jan 07 2014 11:00:00 GMT-0800 (PST)
40070	2014	Ford	Taurus	SEL	Sedan	NaN	1fahp2h88eg138877	qc	43.0	18239.0	silver	black	ford motor company of canada limited	21900.0	22500.0	Tue Feb 10 2015 02:00:00 GMT-0800 (PST)
40071	2014	Ford	Mustang	GT	Convertible	automatic	1zvbp8ff1e5253205	fl	48.0	6448.0	blue	black	tdaf remarketing	27100.0	26000.0	Mon Dec 22 2014 09:00:00 GMT-0800 (PST)
40072	2014	Ford	Taurus	Limited	Sedan	automatic	1fahp2f85eg172066	tx	37.0	33630.0	—	black	avis budget group	18150.0	17800.0	Tue Dec 23 2014 10:30:00 GMT-0800 (PST)

Vẽ biểu đồ biểu thị ảnh hưởng của số km đến giá bán

- Thiết lập kiểu của biểu đồ

```
# Thiết lập kiểu của biểu đồ
sns.set(style="whitegrid")
```

- Tạo biểu đồ phân tán

```
# Tạo biểu đồ phân tán
plt.figure(figsize=(12, 6))
sns.scatterplot(x='odometer', y='sellingprice', data=cleaned_car_prices, alpha=0.5)
```

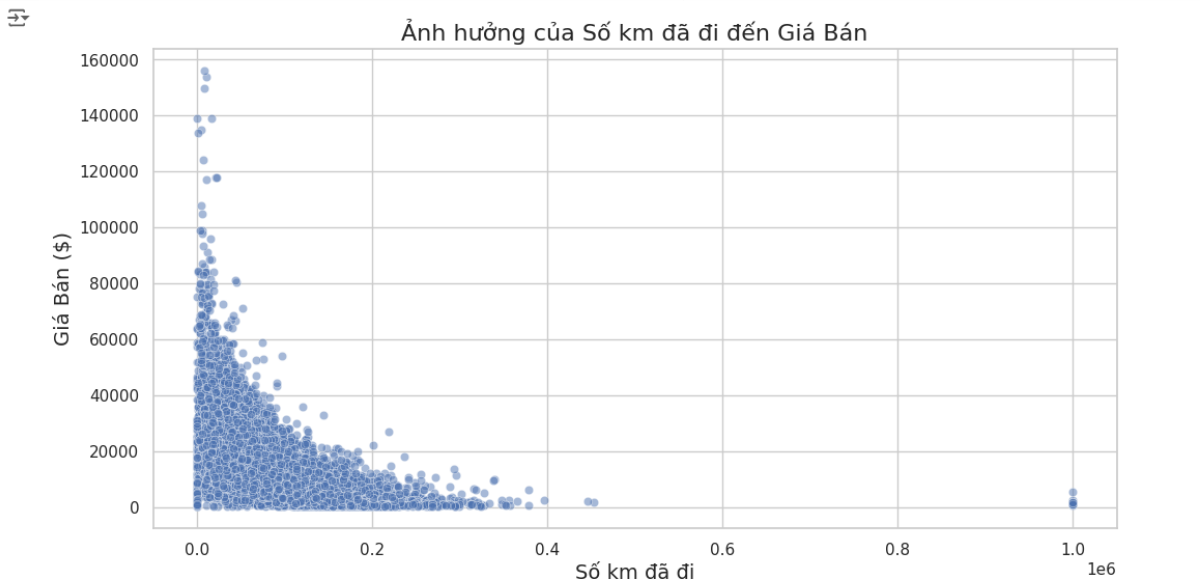
- Thiết lập tiêu đề và nhãn cho trục

```
# Thiết lập tiêu đề và nhãn cho trục
plt.title('Ảnh hưởng của Số km đã đi đến Giá Bán', fontsize=16)
plt.xlabel('Số km đã đi', fontsize=14)
plt.ylabel('Giá Bán ($)', fontsize=14)
```

- Hiển thị biểu đồ

```
# Hiển thị biểu đồ
plt.show()
```

Kết quả



8. Giá bán theo các loại thân xe (body):

Loại bỏ các giá trị thiếu trong cột body và sellingprice

```
[25] # Loại bỏ các hàng có giá trị thiếu trong cột 'body' hoặc 'sellingprice'
     cln_body = df.dropna(subset=['body', 'sellingprice'])
```

Kết quả

```
cln_body
```

	year	make	model	trim	body	transmission	vin	state	condition	odometer	color	interior	seller	mmr	sellingprice	saledate
0	2015	Kia	Sorento	LX	SUV	automatic	5xyktca69fg566472	ca	5.0	16639.0	white	black	kia motors america inc	20500.0	21500.0	Tue Dec 16 2014 12:30:00 GMT-0800 (PST)
1	2015	Kia	Sorento	LX	SUV	automatic	5xyktca69fg561319	ca	5.0	9393.0	white	beige	kia motors america inc	20800.0	21500.0	Tue Dec 16 2014 12:30:00 GMT-0800 (PST)
2	2014	BMW	3 Series	328i SULEV	Sedan	automatic	wba3c1c51ek116351	ca	45.0	1331.0	gray	black	financial services remarketing (lease)	31900.0	30000.0	Thu Jan 15 2015 04:30:00 GMT-0800 (PST)
3	2015	Volvo	S60	T5	Sedan	automatic	yv1612tb4f1310987	ca	41.0	14282.0	white	black	volvo na rep/world omni	27500.0	27750.0	Thu Jan 29 2015 04:30:00 GMT-0800 (PST)
4	2014	BMW	6 Series Gran Coupe	650i	Sedan	automatic	wba6b2c57ed129731	ca	43.0	2641.0	gray	black	financial services remarketing (lease)	66000.0	67000.0	Thu Dec 18 2014 12:30:00 GMT-0800 (PST)
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
40069	2014	Ford	Focus	SE	Sedan	automatic	1fadp3f29el302602	tx	41.0	24382.0	black	black	avis budget group	12000.0	13600.0	Tue Jan 07 2014 11:00:00 GMT-0800 (PST)
40070	2014	Ford	Taurus	SEL	Sedan	NaN	1fahp2h88eg138877	qc	43.0	18239.0	silver	black	ford motor company of canada limited	21900.0	22500.0	Tue Feb 10 2015 02:00:00 GMT-0800 (PST)
40071	2014	Ford	Mustang	GT	Convertible	automatic	1zvbp8ff1e5253205	fl	48.0	6448.0	blue	black	tdaf remarketing	27100.0	26000.0	Mon Dec 22 2014 09:00:00 GMT-0800 (PST)
40072	2014	Ford	Taurus	Limited	Sedan	automatic	1fahp2f85eg172066	tx	37.0	33630.0	—	black	avis budget group	18150.0	17800.0	Tue Dec 23 2014 10:30:00 GMT-0800 (PST)

Vẽ biểu đồ hiển thị giá bán theo thân xe

- Thiết lập kiểu biểu đồ

```
# Thiết lập kiểu của biểu đồ
sns.set(style="whitegrid")
```

- Tạo biểu đồ hộp

```
# Tạo biểu đồ hộp
plt.figure(figsize=(14, 8))
sns.boxplot(x='body', y='sellingprice', data=cleaned_car_prices)
```

- Thiết lập tiêu đề và nhãn cho trục

```
# Thiết lập tiêu đề và nhãn cho trục
plt.title('Giá Bán theo Các Loại Thân Xe', fontsize=16)
plt.xlabel('Loại Thân Xe', fontsize=14)
plt.ylabel('Giá Bán ($)', fontsize=14)
```

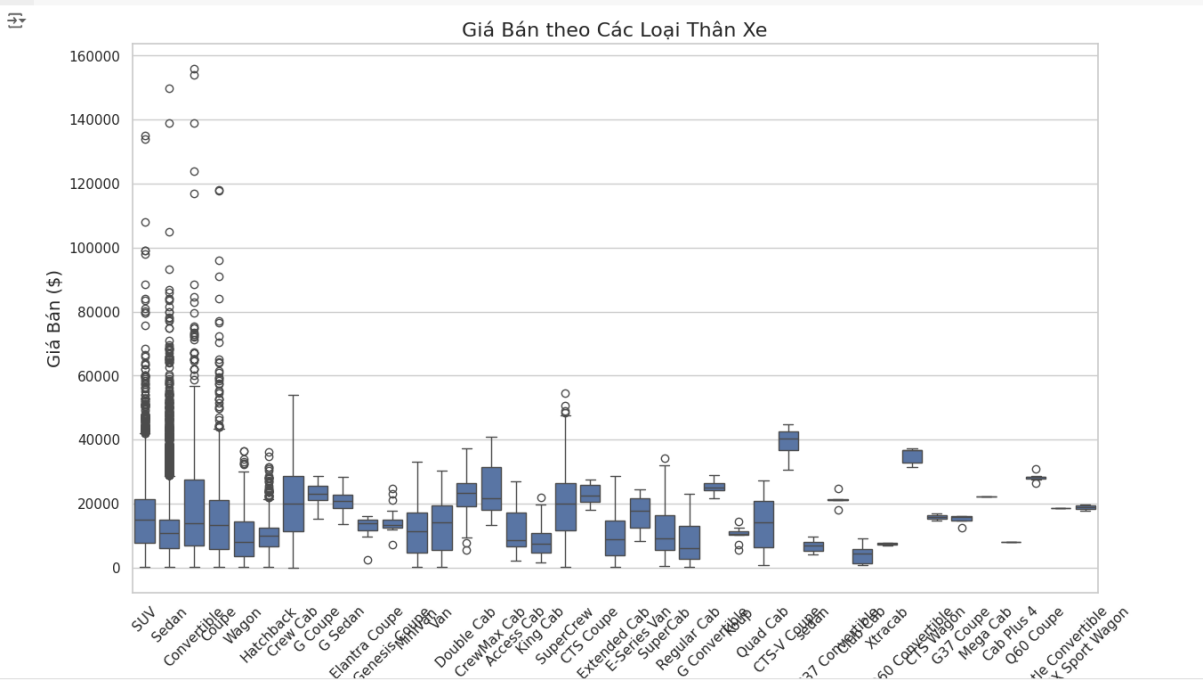
- Xoay nhãn trục x cho dễ cho đọc hơn (Optional)

```
# Xoay nhãn trục x để dễ đọc hơn
plt.xticks(rotation=45)
```

- Hiển thị biểu đồ

```
# Hiển thị biểu đồ
plt.show()
```

Kết quả



9. Phân bố giá bán theo màu sắc xe:

Loại bỏ các giá trị thiếu trong cột color và sellingprice

```
# Loại bỏ các hàng có giá trị thiếu trong cột 'color' hoặc 'sellingprice'
cln_color = df.dropna(subset=['color', 'sellingprice'])
```

Kết quả

```
cln_color
```

	year	make	model	trim	body	transmission	vin	state	condition	odometer	color	interior	seller	mmr	sellingprice	saledate
0	2015	Kia	Sorento	LX	SUV	automatic	5xyktca69fg566472	ca	5.0	16639.0	white	black	kia motors america inc	20500.0	21500.0	Tue Dec 16 2014 12:30:00 GMT-0800 (PST)
1	2015	Kia	Sorento	LX	SUV	automatic	5xyktca69fg561319	ca	5.0	9393.0	white	beige	kia motors america inc	20800.0	21500.0	Tue Dec 16 2014 12:30:00 GMT-0800 (PST)
2	2014	BMW	3 Series	328i SULEV	Sedan	automatic	wba3c1c51ek116351	ca	45.0	1331.0	gray	black	financial services remarketing (lease)	31900.0	30000.0	Thu Jan 15 2015 04:30:00 GMT-0800 (PST)
3	2015	Volvo	S60	T5	Sedan	automatic	yv1612tb4f1310987	ca	41.0	14282.0	white	black	volvo na rep/world omni	27500.0	27750.0	Thu Jan 29 2015 04:30:00 GMT-0800 (PST)
4	2014	BMW	6 Series Gran Coupe	650i	Sedan	automatic	wba6b2c57ed129731	ca	43.0	2641.0	gray	black	financial services remarketing (lease)	66000.0	67000.0	Thu Dec 18 2014 12:30:00 GMT-0800 (PST)
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
40069	2014	Ford	Focus	SE	Sedan	automatic	1fadp3f29el302602	tx	41.0	24382.0	black	black	avis budget group	12000.0	13600.0	Tue Jan 07 2014 11:00:00 GMT-0800 (PST)
40070	2014	Ford	Taurus	SEL	Sedan	NaN	1fahp2h88eg138877	qc	43.0	18239.0	silver	black	ford motor company of canada limited	21900.0	22500.0	Tue Feb 10 2015 02:00:00 GMT-0800 (PST)
40071	2014	Ford	Mustang	GT	Convertible	automatic	1zvbp8ff1e5253205	fl	48.0	6448.0	blue	black	tdaf remarketing	27100.0	26000.0	Mon Dec 22 2014 09:00:00 GMT-0800 (PST)
40072	2014	Ford	Taurus	Limited	Sedan	automatic	1fahp2f85eg172066	tx	37.0	33630.0	—	black	avis budget group	18150.0	17800.0	Tue Dec 23 2014 10:30:00 GMT-0800 (PST)

Vẽ biểu đồ phân bố giá bán theo màu sắc xe

- Thiết lập kiểu biểu đồ

```
# Thiết lập kiểu của biểu đồ
sns.set(style="whitegrid")
```

- Tạo biểu đồ hộp

```
# Tạo biểu đồ hộp
plt.figure(figsize=(14, 8))
sns.boxplot(x='color', y='sellingprice', data=cleaned_car_prices)
```

- Thiết lập tiêu đề và nhãn cho trục

```
# Thiết lập tiêu đề và nhãn cho trục
plt.title('Phân bố Giá Bán theo Màu Sắc Xe', fontsize=16)
plt.xlabel('Màu Sắc Xe', fontsize=14)
plt.ylabel('Giá Bán ($)', fontsize=14)
```

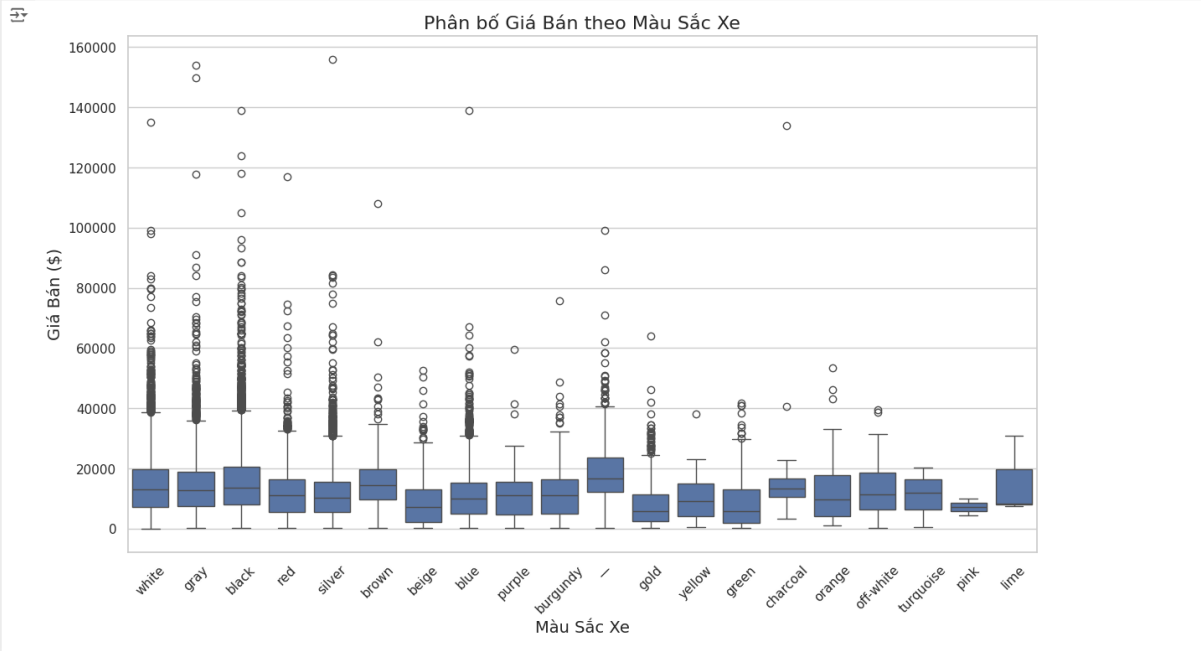
- Xoay nhãn trục x cho dễ cho đọc hơn (Optional)

```
# Xoay nhãn trục x để dễ đọc hơn
plt.xticks(rotation=45)
```

- Hiển thị biểu đồ

```
# Hiển thị biểu đồ
plt.show()
```

Kết quả



10. Xu hướng giá bán theo ngày bán (saledate):

Loại bỏ các giá trị thiếu trong cột saledate và sellingprice

```
# Loại bỏ các hàng có giá trị thiếu trong cột 'saledate' hoặc 'sellingprice'
cln_date = df.dropna(subset=['saledate', 'sellingprice'])
```

Chuyển cột saledate sang định dạng datetime

- Kiểm tra và lọc các giá trị ngày tháng không hợp lệ

```
# Kiểm tra và lọc các giá trị ngày tháng không hợp lệ
def is_valid_date(date_str):
    try:
        pd.to_datetime(date_str, errors='raise')
        return True
    except:
        return False
```

- Lọc các giá trị ngày tháng hợp lệ

```
# Lọc các hàng có giá trị ngày tháng hợp lệ
cln_date = cln_date[cln_date['saledate'].apply(is_valid_date)]
```

- Chuyển đổi cột saledate sang định dạng datetime

```
# Chuyển đổi cột 'saledate' thành định dạng datetime
cln_date['saledate'] = pd.to_datetime(cln_date['saledate'])
```

Vẽ biểu đồ thể hiện giá bán theo ngày bán

- Thiết lập kiểu biểu đồ

```
# Thiết lập kiểu của biểu đồ
sns.set(style="whitegrid")
```

- Tạo biểu đồ đường

```
# Tạo biểu đồ đường
plt.figure(figsize=(50, 8))
sns.lineplot(x='saledate', y='sellingprice', data=df)
```

- Thiết lập tiêu đề và nhãn cho trục

```
# Thiết lập tiêu đề và nhãn cho trục
plt.title('Xu hướng Giá Bán theo Ngày Bán', fontsize=16)
plt.xlabel('Ngày Bán', fontsize=14)
plt.ylabel('Giá Bán($)', fontsize=14)
```

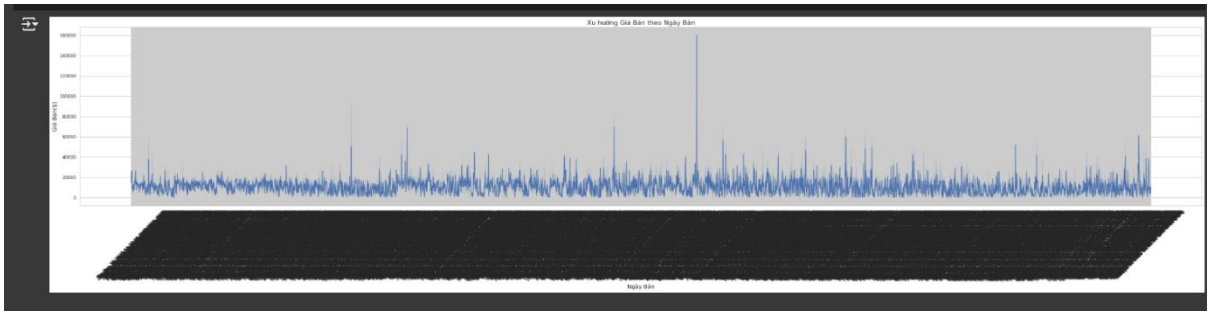
- Xoay nhãn trục x để dễ đọc hơn

```
# Xoay nhãn trục x để dễ đọc hơn
plt.xticks(rotation=45)
```

- Hiển thị đồ thị

```
# Hiển thị biểu đồ
plt.show()
```

Kết quả



11. Phân bố giá bán theo bang

Loại bỏ các giá trị thiếu trong cột state và sellingprice

```
# Loại bỏ các hàng có giá trị thiếu trong cột 'state' hoặc 'sellingprice'
cln_state = df.dropna(subset=['state','sellingprice'])
```

Kết quả

```
cln_state
```

	year	make	model	trim	body	transmission	vin	state	condition	odometer	color	interior	seller	mmr	sellingprice	saledate
0	2015	Kia	Sorento	LX	SUV	automatic	5xyktca69fg566472	ca	5.0	16639.0	white	black	kia motors america inc	20500.0	21500.0	Tue Dec 16 2014 12:30:00 GMT-0800 (PST)
1	2015	Kia	Sorento	LX	SUV	automatic	5xyktca69fg561319	ca	5.0	9393.0	white	beige	kia motors america inc	20800.0	21500.0	Tue Dec 16 2014 12:30:00 GMT-0800 (PST)
2	2014	BMW	3 Series	328i SULEV	Sedan	automatic	wba3c1c51ek116351	ca	45.0	1331.0	gray	black	financial services remarketing (lease)	31900.0	30000.0	Thu Jan 15 2015 04:30:00 GMT-0800 (PST)
3	2015	Volvo	S60	T5	Sedan	automatic	yv1612tb4f1310987	ca	41.0	14282.0	white	black	volvo na rep/world omni	27500.0	27750.0	Thu Jan 29 2015 04:30:00 GMT-0800 (PST)
4	2014	BMW	6 Series Gran Coupe	650i	Sedan	automatic	wba6b2c57ed129731	ca	43.0	2641.0	gray	black	financial services remarketing (lease)	66000.0	67000.0	Thu Dec 18 2014 12:30:00 GMT-0800 (PST)
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
558832	2015	Kia	K900	Luxury	Sedan	NaN	knalw4d4xf6019304	in	45.0	18255.0	silver	black	avis corporation	35300.0	33000.0	Thu Jul 09 2015 07:00:00 GMT-0700 (PDT)
558833	2012	Ram	2500	Power Wagon	Crew Cab	automatic	3c6td5et6cg112407	wa	5.0	54393.0	white	black	i -5 uhlmann rv	30200.0	30800.0	Wed Jul 08 2015 09:30:00 GMT-0700 (PDT)
558834	2012	BMW	X5	xDrive35d	SUV	automatic	5uxzw0c58cl668465	ca	48.0	50561.0	black	black	financial services remarketing (lease)	29800.0	34000.0	Wed Jul 08 2015 09:30:00 GMT-0700 (PDT)
558835	2015	Nissan	Altima	2.5 S	sedan	automatic	1n4al3ap0fc216050	ga	38.0	16658.0	white	black	enterprise vehicle exchange / tra /	15100.0	11100.0	Thu Jul 09 2015 06:45:00 GMT-0700

Vẽ biểu đồ hiển thị giá bán phân bố theo bang:

- Thiết lập kiểu biểu đồ

```
# Biểu đồ phân tán
plt.figure(figsize=(20, 10))
```

- Tạo biểu đồ phân tán

```
# Tạo biểu đồ phân tán (scatter plot)
sns.scatterplot(x='state', y='sellingprice', data=df)
```

- Thêm tiêu đề và nhãn trục

```
# Thêm tiêu đề và nhãn trục
plt.title('Phân tán giá bán theo bang')
plt.xlabel('Bang')
plt.ylabel('Giá bán')
```

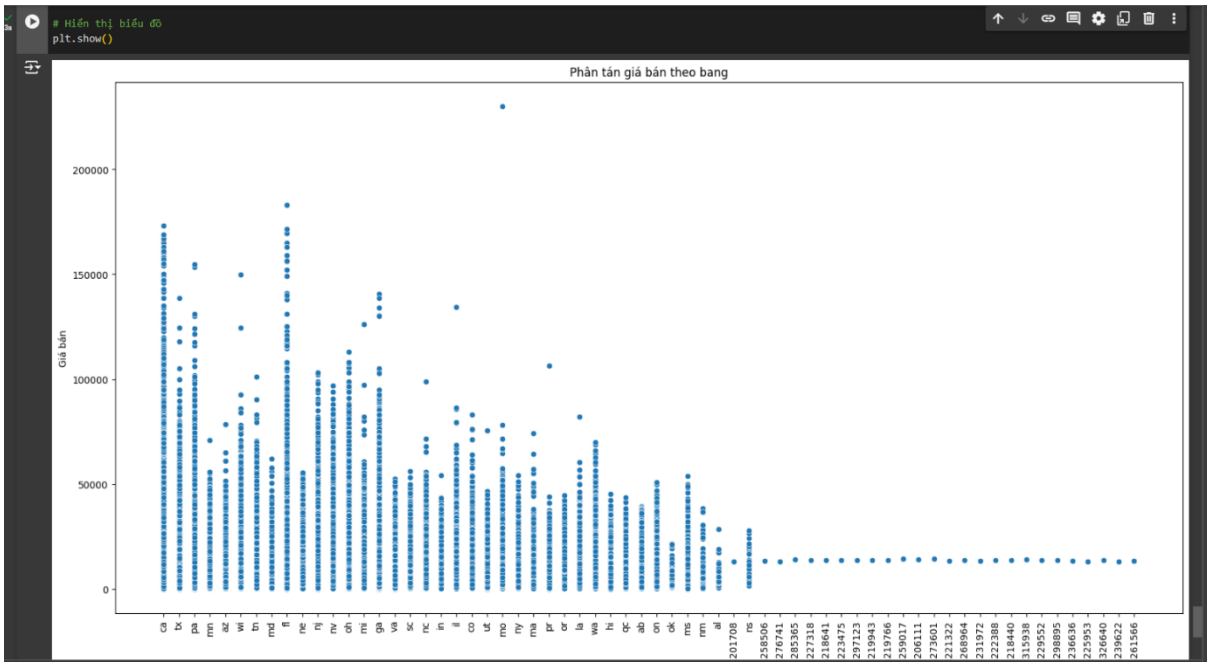
- Xoay nhãn trục x để dễ đọc (Optional)

```
# Xoay nhãn trục x để dễ đọc hơn
plt.xticks(rotation=90)
```

- Hiển thị biểu đồ

```
# Hiển thị biểu đồ
plt.show()
```

Kết quả



Chương 3. Tiền xử lý dữ liệu

1. Mô tả dữ liệu

Thống kê các chỉ số định tính và định lượng có trong bộ dữ liệu

Định lượng: giá trị lớn nhất, nhỏ nhất, giá trị trung bình, độ lệch chuẩn và tứ phân vị

```
import pandas as pd
df = pd.read_csv('car_prices.csv')
df.describe().T
```

	count	mean	std	min	25%	50%	75%	max
year	558837.0	2010.038927	3.966864	1982.0	2007.0	2012.0	2013.0	2015.0
condition	547017.0	30.672365	13.402832	1.0	23.0	35.0	42.0	49.0
odometer	558743.0	68320.017767	53398.542821	1.0	28371.0	52254.0	99109.0	999999.0
mmr	558799.0	13769.377495	9679.967174	25.0	7100.0	12250.0	18300.0	182000.0
sellingprice	558825.0	13611.358810	9749.501628	1.0	6900.0	12100.0	18200.0	230000.0

Thống kê thông tin chi tiết dữ liệu định lượng

Định tính: tính số giá trị khác nhau trong bộ dữ liệu và giá trị có tần suất cao nhất và tần số của giá trị đó

```
import pandas as pd
df = pd.read_csv('car_prices.csv')
df.describe(include = "object").T
```

	count	unique	top	freq
make	548536	96	Ford	93554
model	548438	973	Altima	19349
trim	548186	1963	Base	55817
body	545642	87	Sedan	199437
transmission	493485	4	automatic	475915
vin	558833	550297	automatic	22
state	558837	64	fl	82945
color	558088	46	black	110970
interior	558088	17	black	244329
seller	558837	14263	nissan-infiniti lt	19693
saledate	558825	3766	Tue Feb 10 2015 01:30:00 GMT-0800 (PST)	5334

Thống kê thông tin chi tiết dữ liệu định tính

2. Làm sạch dữ liệu

2.1. Nhận diện và xử lý các dữ liệu bị nhiễu

Sử dụng phương pháp IQR để tìm các ngoại lệ có trong các thuộc tính định lượng bằng cách tìm ra phân vị tứ thứ nhất và thứ ba rồi tính ra giới hạn giá trị trên và giới hạn giá trị dưới của từng thuộc tính. Cuối cùng là loại bỏ các dòng dữ liệu không nằm trong vùng giá trị vừa tạo

Thuộc tính year:

```
q1=df.year.quantile(0.25)
q3=df.year.quantile(0.75)
q1,q3
IQR=q3-q1
lower_year=q1-IQR*1.5
upper_year=q3+IQR*1.5
print("IQR:", IQR)
print("q1:", q1)
print("q3:", q3)
print("Lower limit:", lower_year)
print("Upper limit:", upper_year)
```

```
IQR: 6.0
q1: 2007.0
q3: 2013.0
Lower limit: 1998.0
Upper limit: 2022.0
```

Giới hạn trên và dưới của thuộc tính year

557327	1994	Cadillac	Seville	Base	sedan	automatic	1g6ks52y6ru826119	fl	2.0	147219.0	blue	â€"	ed morse auto plaza	750.0	900.0	Thu 18 2 08:3 GMT-0 (P
557328	1994	Honda	Civic	LX	sedan	automatic	1hgeg8650rl003433	fl	2.0	208218.0	blue	gray	coggin honda of ft pierce	575.0	350.0	Thu 18 2 06:0 GMT-0 (P
557329	1995	Chevrolet	Camaro	Base	hatchback	automatic	2g1fp22s3s2209758	tx	3.0	118442.0	black	gray	autonation ford south fort worth	1225.0	1400.0	Thu 18 2 04:0 GMT-0 (P
557863	1996	NaN	NaN	NaN	NaN	automatic	1b4gp54lxtb344615	fl	3.0	107504.0	green	gray	brandon auto mall fiat	650.0	1400.0	Thu 18 2 08:3 GMT-0 (P
558650	1997	Mercedes-Benz	S-Class	S320 LWB	sedan	automatic	wdbga33g9va372927	pa	28.0	94947.0	â€"	tan	r hollenshead auto sales inc	2550.0	3200.0	Fri Jur 2 02:0 GMT-0 (P

4021 rows × 16 columns

Các dòng dữ liệu ngoại lệ của thuộc tính year

4	2014	BMW	6 Series Gran Coupe	650i	Sedan	automatic	wba6b2c57ed129731	ca	43.0	2641.0	gray	black	financial services remarketing (lease)	66000.0	67000.0	18 2014 12:30:0(GMT-080((PST
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
558832	2015	Kia	K900	Luxury	Sedan	NaN	knalw4d4xf6019304	in	45.0	18255.0	silver	black	avis corporation	35300.0	33000.0	Thu Jul 0(2015 07:00:0(GMT-070((PDT
558833	2012	Ram	2500	Power Wagon	Crew Cab	automatic	3c6td5et6cg112407	wa	5.0	54393.0	white	black	i -5 uhlmann rv	30200.0	30800.0	Wed Ju 08 201(09:30:0(GMT-070((PDT
558834	2012	BMW	X5	xDrive35d	SUV	automatic	5uxzw0c58cl668465	ca	48.0	50561.0	black	black	financial services remarketing (lease)	29800.0	34000.0	Wed Ju 08 201(09:30:0(GMT-070((PDT
558835	2015	Nissan	Altima	2.5 S	sedan	automatic	1n4al3ap0fc216050	ga	38.0	16658.0	white	black	enterprise vehicle exchange / tra / rental / t...	15100.0	11100.0	Thu Jul 0(2015 06:45:0(GMT-070((PDT
558836	2014	Ford	F-150	XLT	SuperCrew	automatic	1ftfw1et2eke87277	ca	34.0	15008.0	gray	gray	ford motor credit company llc pd	29600.0	26700.0	Thu May 28 201(05:30:0(GMT-070((PDT

554816 rows × 16 columns

Các dòng dữ liệu còn lại

Thuộc tính condition

```
q1=df_updated.condition.quantile(0.25)
q3=df_updated.condition.quantile(0.75)
IQR=q3-q1
lower_cond=q1-IQR*1.5
upper_cond=q3+IQR*1.5
print("IQR:", IQR)
print("q1:", q1)
print("q3:", q3)
print("Lower limit:", lower_cond)
print("Upper limit:", upper_cond)
```

```
IQR: 18.0
q1: 24.0
q3: 42.0
Lower limit: -3.0
Upper limit: 69.0
```

Giới hạn trên và dưới của thuộc tính condition

year	make	model	trim	body	transmission	vin	state	condition	odometer	color	interior	seller	mmr	sellingprice	saledate

Các dòng dữ liệu ngoại lệ của thuộc tính condition

Thuộc tính odometer

```
q1=df_updated.odometer.quantile(0.25)
q3=df_updated.odometer.quantile(0.75)
IQR=q3-q1
lower_odo=q1-IQR*1.5
upper_odo=q3+IQR*1.5
print("IQR:", IQR)
print("q1:", q1)
print("q3:", q3)
print("Lower limit:", lower_odo)
print("Upper limit:", upper_odo)
```

```
IQR: 69967.0
q1: 28242.0
q3: 98209.0
Lower limit: -76708.5
Upper limit: 203159.5
```

Giới hạn trên và dưới của thuộc tính odometer

558468	2002	Chevrolet	Tahoe	LS	suv	automatic	1gnek13z52r240327	fl	27.0	209889.0	black	gray	zimmerman auto brokers inc	1875.0	24(
558470	2002	Honda	Odyssey	LX	minivan	automatic	2hkrl18512h555288	ma	2.0	215038.0	red	gray	boch new to you superstore	975.0	3(
558717	2000	Ford	Explorer	XLT	suv	automatic	1fmzu73e7yza37772	tx	27.0	220815.0	blue	gray	classic chevrolet inc	475.0	6(
558768	2011	Ford	Fusion Hybrid	Base	sedan	automatic	3fadp0l38br272114	fl	26.0	204835.0	blue	beige	firkins chrysler jeep dodge	7750.0	74(
558791	2012	Ford	Transit Connect	Wagon XLT	Minivan	automatic	nm0ks9bnxct078155	mn	27.0	262065.0	silver	gray	saxon fleet services	10100.0	30(

9918 rows × 16 columns

Các dòng dữ liệu ngoại lệ của thuộc tính odometer

558832	2015	Kia	K900	Luxury	Sedan	NaN	knalw4d4xf6019304	in	45.0	18255.0	silver	black	avis corporation	35300.0	33000.0
558833	2012	Ram	2500	Power Wagon	Crew Cab	automatic	3c6td5et6cg112407	wa	5.0	54393.0	white	black	i -5 uhlmann rv	30200.0	30800.0
558834	2012	BMW	X5	xDrive35d	SUV	automatic	5uxzw0c58cl668465	ca	48.0	50561.0	black	black	financial services remarketing (lease)	29800.0	34000.0
558835	2015	Nissan	Altima	2.5 S	sedan	automatic	1n4al3ap0fc216050	ga	38.0	16658.0	white	black	enterprise vehicle exchange / tra / rental / t...	15100.0	11100.0
558836	2014	Ford	F-150	XLT	SuperCrew	automatic	1ftfw1et2eke87277	ca	34.0	15008.0	gray	gray	ford motor credit company llc pd	29600.0	26700.0

534089 rows × 16 columns

Các dòng dữ liệu còn lại

Thuộc tính mmr

```
q1=df_updated.mmr.quantile(0.25)
q3=df_updated.mmr.quantile(0.75)
IQR=q3-q1
lower_mmr=q1-IQR*1.5
upper_mmr=q3+IQR*1.5
print("IQR:", IQR)
print("q1:", q1)
print("q3:", q3)
print("Lower limit:", lower_mmr)
print("Upper limit:", upper_mmr)
```

```
IQR: 11075.0
q1: 7275.0
q3: 18350.0
Lower limit: -9337.5
Upper limit: 34962.5
```

Giới hạn trên và dưới của thuộc tính mmr

19	2014	BMW	6 Series	650i	Convertible	automatic	wbayp9c53ed169260	ca	34.0	8819.0	black	black	the hertz corporation	68000.0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
558777	2012	Maserati	Quattroporte	S	sedan	automatic	zam39jka1c0060738	ga	39.0	21923.0	blue	â€"	maserati north america inc	53300.0
558807	2014	Mercedes-Benz	E-Class	E63 AMG 4MATIC	Sedan	automatic	wddhf9cb9ea917688	ca	45.0	17518.0	black	black	the hertz corporation	64000.0
558816	2013	Mercedes-Benz	G-Class	G63 AMG	suv	automatic	wdcyc7df4dx207385	fl	5.0	26799.0	black	black	fields bmw	104000.0
558824	2013	Audi	S5	Premium Plus quattro	convertible	automatic	waucgafh6dn005382	fl	5.0	20158.0	silver	black	prestige audi	43900.0
558832	2015	Kia	K900	Luxury	Sedan	NaN	knalw4d4xf6019304	in	45.0	18255.0	silver	black	avis corporation	35300.0

16624 rows × 16 columns

Các dòng dữ liệu ngoại lệ của thuộc tính odometer

5	2015	Nissan	Altima	2.5 S	Sedan	automatic	1n4al3ap1fn326013	ca	1.0	5554.0	gray	black	vehicle exchange / tra / rental / t...	15350.0	10900.0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
558831	2011	BMW	5 Series	528i	Sedan	automatic	wbafr1c53bc744672	fl	39.0	66403.0	white	brown	lauderdale imports ltd bmw pembrok pines	20300.0	22800.0
558833	2012	Ram	2500	Power Wagon	Crew Cab	automatic	3c6td5et6cg112407	wa	5.0	54393.0	white	black	i -5 uhlmann rv	30200.0	30800.0
558834	2012	BMW	X5	xDrive35d	SUV	automatic	5uxzw0c58cl668465	ca	48.0	50561.0	black	black	financial services remarketing (lease)	29800.0	34000.0
558835	2015	Nissan	Altima	2.5 S	sedan	automatic	1n4al3ap0fc216050	ga	38.0	16658.0	white	black	enterprise vehicle exchange / tra / rental / t...	15100.0	11100.0
558836	2014	Ford	F-150	XLT	SuperCrew	automatic	1ftfw1et2eke87277	ca	34.0	15008.0	gray	gray	ford motor credit company llc pd	29600.0	26700.0

517453 rows × 16 columns

Các dòng dữ liệu còn lại

Thuộc tính sellingprice

```
q1=df_updated.sellingprice.quantile(0.25)
q3=df_updated.sellingprice.quantile(0.75)
IQR=q3-q1
lower_sell=q1-IQR*1.5
upper_sell=q3+IQR*1.5
print("IQR:", IQR)
print("q1:", q1)
print("q3:", q3)
print("Lower limit:", lower_sell)
print("Upper limit:", upper_sell)
```

```
IQR: 10400.0
q1: 7400.0
q3: 17800.0
Lower limit: -8200.0
Upper limit: 33400.0
```

Giới hạn trên và dưới của thuộc tính sellingprice

424	2013	Infiniti	M	M37	Sedan	automatic	jn1by1ap3dm514252	ca	41.0	9338.0	white	black	nissan north america inc.	34700.0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
558553	2013	Jeep	Wrangler	Unlimited Sahara	suv	NaN	1c4bjweg9dl505974	pa	46.0	37453.0	white	black	r hollenshead auto sales inc	28000.0
558557	2012	Dodge	Challenger	SRT8 392	coupe	manual	2c3cdycj0ch286330	nv	5.0	25237.0	yellow	black	lotus of las vegas	33700.0
558587	2010	Ford	F-150	SVT Raptor	supercab	automatic	1ftex1ev0afa23790	pa	43.0	57388.0	orange	black	r hollenshead auto sales inc	34400.0
558665	2013	GMC	Sierra 1500	SLE	crew cab	automatic	3gtp2ve7xdg335201	pa	5.0	19239.0	black	black	imports of lancaster county inc	28600.0
558834	2012	BMW	X5	xDrive35d	SUV	automatic	5uxzw0c58cl668465	ca	48.0	50561.0	black	black	financial services remarketing (lease)	29800.0

4050 rows × 16 columns

Các dòng dữ liệu ngoại lệ của thuộc tính sellingprice

5	2015	Nissan	Altima	2.5 S	Sedan	automatic	1n4al3ap1fn326013	ca	1.0	5554.0	gray	black	exchange / tra / rental / t...	15350.0	10900.0	G
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
558830	2012	Nissan	Sentra	2.0 SR	Sedan	NaN	3n1ab6ap3cl622485	tn	26.0	35858.0	white	gray	nissan-infiniti lt	9950.0	10400.0	G
558831	2011	BMW	5 Series	528i	Sedan	automatic	wbafr1c53bc744672	fl	39.0	66403.0	white	brown	lauderdale imports ltd bmw pembrok pines	20300.0	22800.0	G
558833	2012	Ram	2500	Power Wagon	Crew Cab	automatic	3c6td5et6cg112407	wa	5.0	54393.0	white	black	i -5 uhlmann rv	30200.0	30800.0	G
558835	2015	Nissan	Altima	2.5 S	sedan	automatic	1n4al3ap0fc216050	ga	38.0	16658.0	white	black	enterprise vehicle exchange / tra / rental / t...	15100.0	11100.0	G
558836	2014	Ford	F-150	XLT	SuperCrew	automatic	1ftfw1et2eke87277	ca	34.0	15008.0	gray	gray	ford motor credit company llc pd	29600.0	26700.0	G

513403 rows × 16 columns

Các dòng dữ liệu còn lại

2.2 Nhận biết các dữ liệu bị thiếu

Trong quá trình xử lý dữ liệu bị nhiễu, các giá trị ở dạng số cũng đã được xử lý nên khi tìm các dòng dữ liệu null thì chỉ còn ở trong các cột dữ liệu dạng object.

Vì vậy sử dụng hàm isnull để tìm các giá trị dạng object có chứa giá trị null sau đó là hàm sum để tính tổng các dòng tồn tại rỗng trong mỗi cột

```
[28]: df_updated.isnull().sum()

[28]: year              0
      make            8723
      model           8735
      trim            8974
      body           10802
      transmission   58645
      vin                0
      state              0
      condition          0
      odometer           0
      color            478
      interior         478
      seller             0
      mmr                0
      sellingprice       0
      saledate           0
      dtype: int64
```

Tính tổng số dòng null bằng axis=1 để tránh trường hợp 1 dòng có 2 giá trị null làm nhiễu dữ liệu

```
[19]: null_rows = df_updated.isnull().any(axis=1).sum()
      null_rows

[19]: 68058
```

Kết luận: bộ dữ liệu chứa tới hơn 68000 dòng dữ liệu rỗng, chiếm gần 8% bộ dữ liệu

2.3 Xử lý ý nghĩa của dữ liệu

2.4 Xử lý dữ liệu bị thiếu

Để xử lý dữ liệu null trong các cột định tính, ta có thể thay thế điều này bằng giá trị Unknown và gom nhóm các dữ liệu rỗng vào đây cho phép ta vẫn sử dụng được các dòng dữ liệu này dù bị khiếm khuyết dữ liệu.

Kết quả là trừ đi hết các dòng dữ liệu rỗng.

```
df_updated=df_updated.fillna("Unknown")
print(df_updated.isnull().sum() )
```

```
year            0
make            0
model           0
trim            0
body            0
transmission    0
vin             0
state           0
condition       0
odometer        0
color           0
interior        0
seller          0
mmr             0
sellingprice    0
saledate        0
dtype: int64
```

2.5 Kiểm tra thuộc tính của dữ liệu

```
df_updated.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 513301 entries, 0 to 558836
Data columns (total 16 columns):
 #   Column        Non-Null Count   Dtype  
---  ------        --------------   -----  
 0   year          513301 non-null  int64  
 1   make          513301 non-null  object 
 2   model         513301 non-null  object 
 3   trim          513301 non-null  object 
 4   body          513301 non-null  object 
 5   transmission  513301 non-null  object 
 6   vin           513301 non-null  object 
 7   state         513301 non-null  object 
 8   condition     513301 non-null  float64
 9   odometer      513301 non-null  float64
 10  color         513301 non-null  object 
 11  interior      513301 non-null  object 
 12  seller        513301 non-null  object 
 13  mmr           513301 non-null  float64
 14  sellingprice  513301 non-null  float64
 15  saledate      513301 non-null  object 
dtypes: float64(4), int64(1), object(11)
memory usage: 66.6+ MB
```

Nhìn nhận các cột dữ liệu ta thấy dữ liệu tồn tại ở 3 dạng là int64, float64 và object. Nhằm làm cho việc tương tác với dữ liệu đơn giản và dễ nhìn ta đổi thuộc tính về kiểu dư liệu int

```
df_updated["condition"]=df_updated["condition"].astype(np.int64)
df_updated["mmr"]=df_updated["mmr"].astype(np.int64)
df_updated["odometer"]=df_updated["odometer"].astype(np.int64)
df_updated["sellingprice"]=df_updated["sellingprice"].astype(np.int64)
```

Đồng thời, đối với cột saledate thì chỉ cần lưu ở dạng dd/mm/yyyy nên sẽ bỏ bớt các yếu tố giờ:phút:giây và múi giờ.
Do hàm ép kiểu parser chỉ ứng dụng đươc cho từng chuỗi đơn nên cần chạy vòng lặp for để ép kiểu dữ liệu cho cột dữ liệu mới mang tên saledate_formatted

```
for i, row in df_updated.iterrows():
    date_string = str(df_updated.at[i, 'saledate'])
    parsed_date = parser.parse(date_string)
    df_updated.at[i, 'saledate_formatted'] = parsed_date.strftime('%d/%m/%Y')
```

```
df_updated.saledate_formatted.head(10)
```

```
0     16/12/2014
1     16/12/2014
2     15/01/2015
3     29/01/2015
5     30/12/2014
7     16/12/2014
8     18/12/2014
9     20/01/2015
11    16/12/2014
12    13/01/2015
Name: saledate_formatted, dtype: object
```

Kết quả cột dữ liệu mới như trên:

2.6 Xử lý các dữ liệu bị trùng lặp

Các dòng dữ liệu trùng có thể bị phát hiện bằng hàm duplicated().any().
Tùy thuộc vào kết trả về mà thay đổi phương pháp xử lý cho phù hợp.
True: Xóa dữ liệu trùng bằng hàm drop_duplicates
False: Không có dữ liệu trùng nên tiếp tục bình thường

```
df_updated.duplicated().any()
```

```
False
```

3. Tích hợp dữ liệu

Tương quan giữa các thuộc tính và thuộc tính quyết định Sellingprice

```
# Odometer vs Selling Price
plt.subplot(1, 3, 1)
sns.scatterplot(x='odometer', y='sellingprice', data=df)
plt.title('Odometer vs Selling Price')
plt.xlabel('Odometer')
plt.ylabel('Selling Price')
```

```
# MMR vs Selling Price
plt.subplot(1, 3, 2)
sns.scatterplot(x='mmr', y='sellingprice', data=df)
plt.title('MMR vs Selling Price')
plt.xlabel('MMR')
plt.ylabel('Selling Price')
```

```
# Make vs Selling Price
plt.subplot(2, 3, 1)
sns.scatterplot(x='make', y='sellingprice', data=df)
plt.title('Make vs Selling Price')
plt.xlabel('Make')
plt.ylabel('Selling Price')
plt.xticks(rotation=90)
```

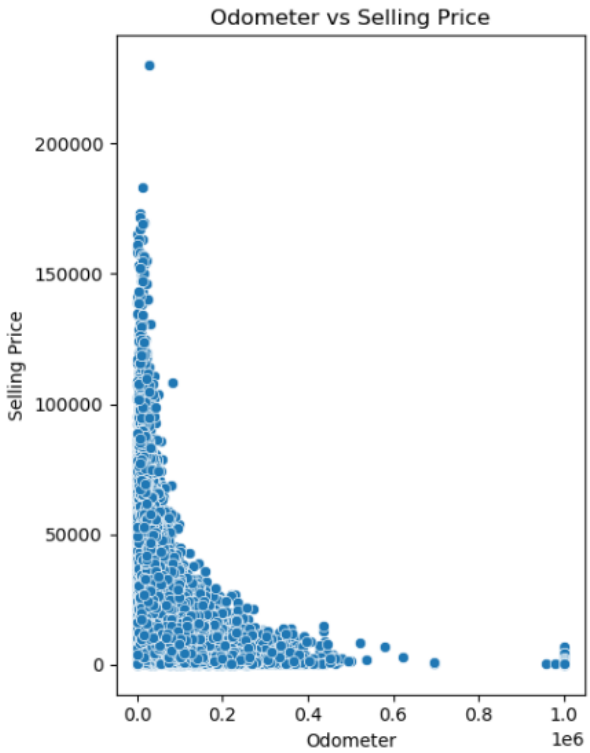
```
# Transmission vs Selling Price
plt.subplot(2, 3, 3)
sns.scatterplot(x='transmission', y='sellingprice', data=df)
plt.title('Transmission vs Selling Price')
plt.xlabel('Transmission')
plt.ylabel('Selling Price')
```

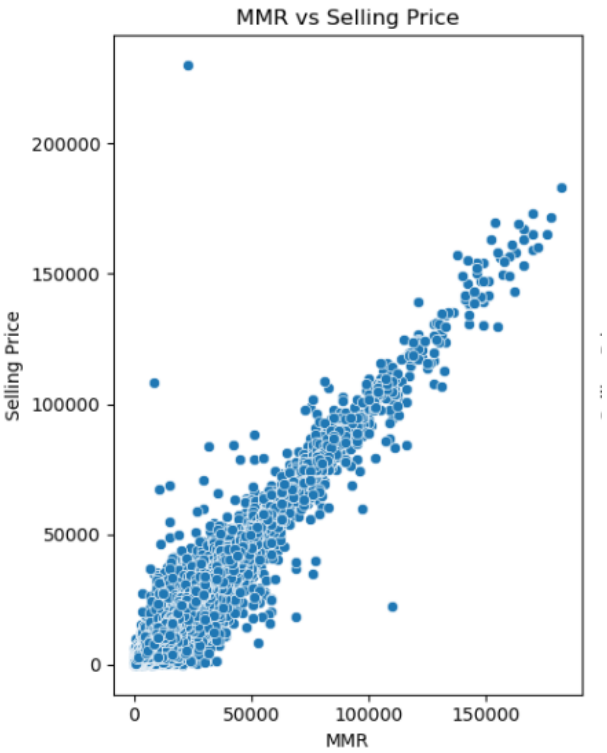
```
# Condition vs Selling Price
plt.subplot(2, 3, 4)
sns.scatterplot(x='condition', y='sellingprice', data=df)
plt.title('Condition vs Selling Price')
plt.xlabel('Condition')
plt.ylabel('Selling Price')
```

```
# State vs Selling Price
plt.subplot(2, 3, 5)
sns.scatterplot(x='state', y='sellingprice', data=df)
plt.title('State vs Selling Price')
plt.xlabel('State')
plt.ylabel('Selling Price')
plt.xticks(rotation=90)
```

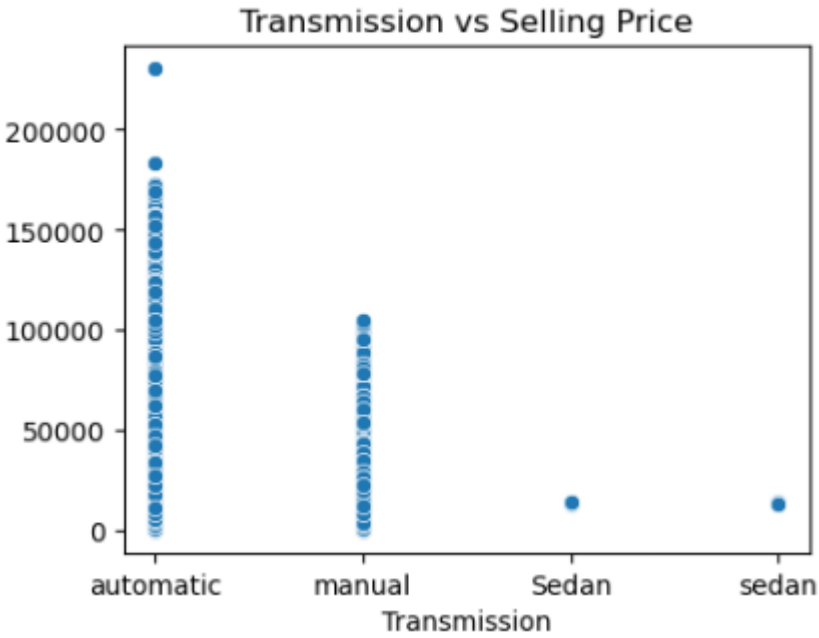
```
# Color vs Selling Price
plt.subplot(2, 3, 6)
sns.scatterplot(x='color', y='sellingprice', data=df)
plt.title('Color vs Selling Price')
plt.xlabel('Color')
plt.ylabel('Selling Price')
plt.xticks(rotation=90)
```

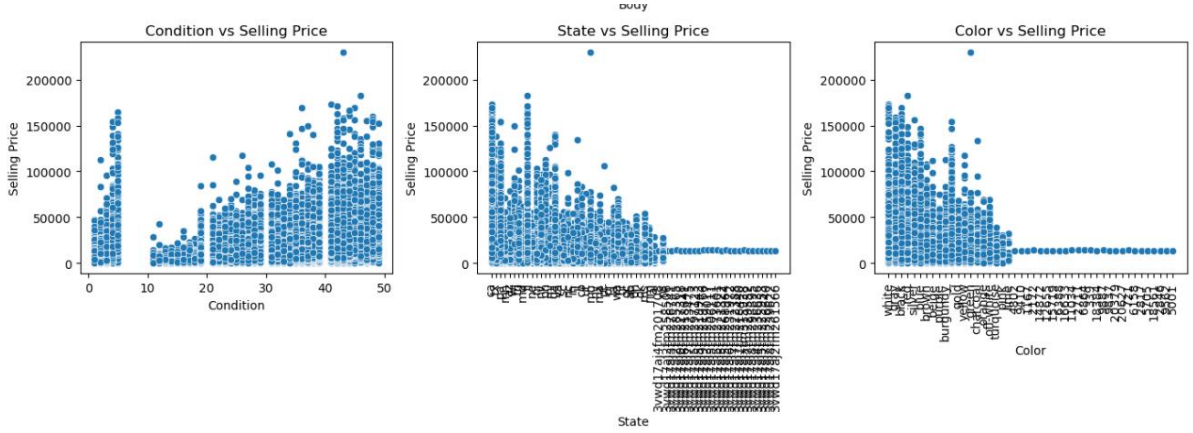
Từ đó chúng ta có các kết quả là biểu đồ scatterplot tương ứng


Odometer vs Selling Price
Selling Price
Odometer
1e6
0
50000
100000
150000
200000
0.0
0.2
0.4
0.6
0.8
1.0


MMR vs Selling Price
Selling Price
MMR
0
50000
100000
150000
200000


Make vs Selling Price
Selling Price
Make
0
50000
100000
150000
200000





Nhận xét:

- Với thuộc tính odometer: Biểu đồ phân tán cho thấy một mối quan hệ nghịch giữa số km đã đi (odometer) và giá bán (sellingprice). Khi số km đã đi tăng lên, giá bán của xe có xu hướng giảm. Điều này là hợp lý vì xe có số km đã đi ít thường có giá trị cao hơn do ít hao mòn.
- Với thuộc tính mmr: Biểu đồ phân tán cho thấy một mối quan hệ tuyến tính dương giữa giá trị thị trường dự đoán (MMR) và giá bán (sellingprice). Điều này có nghĩa là MMR là một ước tính tốt cho giá bán thực tế, và giá trị thị trường cao thì giá bán cũng cao.
- Với thuộc tính make: Biểu đồ phân tán cho thấy giá bán của các xe từ các hãng khác nhau có sự biến động lớn. Một số hãng như Mercedes-Benz và BMW có xu hướng có giá bán cao hơn

- Với thuộc tính transmission: Biểu đồ phân tán cho thấy có sự khác biệt về giá bán giữa các loại hộp số. Xe hộp số tự động (automatic) có xu hướng có giá bán cao hơn so với xe hộp số tay (manual).
- Với thuộc tính condition: Biểu đồ phân tán cho thấy một mối quan hệ dương giữa tình trạng xe (condition) và giá bán (sellingprice). Xe có tình trạng tốt hơn (điểm condition cao hơn) có xu hướng có giá bán cao hơn.
- Với thuộc tính state: Biểu đồ phân tán cho thấy sự khác biệt về giá bán giữa các bang (state).
- Với thuộc tính color: Biểu đồ phân tán cho thấy màu sắc xe (color) có ảnh hưởng nhất định đến giá bán. Một số màu sắc có xu hướng có giá bán cao hơn, đứng đầu là white rồi tới gray, black

4. .Biến đổi dữ liệu và thu giảm dữ liệu

4.1. Thu giảm dữ liệu

- Xóa các thuộc tính không cần thiết:
    - Xóa thuộc tính saledate vì đã xử lý sang saledate_formatted (chuyển sang đoạn datetime)
        - Trước:

```
df=pd.read_csv("/content/df_updated.csv")
df
```

	year	make	model	trim	body	transmission	vin	state	condition	odometer	color	interior	seller	mmr	sellingprice	saledate	saledate_formatte
0	2015	Kia	Sorento	LX	SUV	automatic	5xyktca69fg566472	ca	1	1	white	black	kia motors america inc	4.0	4.0	Tue Dec 16 2014 12:30:00 GMT-0800 (PST)	16/12/201
1	2015	Kia	Sorento	LX	SUV	automatic	5xyktca69fg561319	ca	1	1	white	beige	kia motors america inc	4.0	4.0	Tue Dec 16 2014 12:30:00 GMT-0800 (PST)	16/12/201
2	2014	BMW	3 Series	328i SULEV	Sedan	automatic	wba3c1c51ek116351	ca	4	1	gray	black	financial services remarketing (lease)	4.0	4.0	Thu Jan 15 2015 04:30:00 GMT-0800 (PST)	15/01/201
3	2015	Volvo	S60	T5	Sedan	automatic	yv1612tb4f1310987	ca	3	1	white	black	volvo na rep/world omni	4.0	4.0	Thu Jan 29 2015 04:30:00 GMT-0800 (PST)	29/01/201

        - Sau:

```
[4] df=df.drop(columns=['saledate'])
```

```
df
```

	year	make	model	trim	body	transmission	vin	state	condition	odometer	color	interior	seller	mmr	sellingprice	saledate_formatted
0	2015	Kia	Sorento	LX	SUV	automatic	5xyktca69fg566472	ca	5	16639	white	black	kia motors america inc	20500	21500	16/12/2014
1	2015	Kia	Sorento	LX	SUV	automatic	5xyktca69fg561319	ca	5	9393	white	beige	kia motors america inc	20800	21500	16/12/2014
2	2014	BMW	3 Series	328i SULEV	Sedan	automatic	wba3c1c51ek116351	ca	45	1331	gray	black	financial services remarketing (lease)	31900	30000	15/01/2015
3	2015	Volvo	S60	T5	Sedan	automatic	yv1612tb4f1310987	ca	41	14282	white	black	volvo na rep/world omni	27500	27750	29/01/2015
4	2015	Nissan	Altima	2.5 S	Sedan	automatic	1n4al3ap1fn326013	ca	1	5554	gray	black	enterprise vehicle exchange / tra / rental / t...	15350	10900	30/12/2014
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
42980	2007	Chevrolet	Tahoe	LT	SUV	automatic	1gnfk13047j112818	nc	19	132314	silver	gray	westlake financial services	12950	11900	29/12/2014

- Xóa khoảng trắng thừa

```
df['make']=df['make'].str.strip()
df['model']=df['model'].str.strip()
df['trim']=df['trim'].str.strip()
df['body']=df['body'].str.strip()
df['transmission']=df['transmission'].str.strip()
df['vin']=df['vin'].str.strip()
df['state']=df['state'].str.strip()
df['color']=df['color'].str.strip()
df['interior']=df['interior'].str.strip()
df['seller']=df['seller'].str.strip()
df
```

	year	make	model	trim	body	transmission	vin	state	condition	odometer	color	interior	seller	mmr	sellingprice	saledate_formatted
0	2015	Kia	Sorento	LX	SUV	automatic	5xyktca69fg566472	ca	5	16639	white	black	kia motors america inc	20500	21500	16/12/2014
1	2015	Kia	Sorento	LX	SUV	automatic	5xyktca69fg561319	ca	5	9393	white	beige	kia motors america inc	20800	21500	16/12/2014
2	2014	BMW	3 Series	328i SULEV	Sedan	automatic	wba3c1c51ek116351	ca	45	1331	gray	black	financial services remarketing (lease)	31900	30000	15/01/2015
3	2015	Volvo	S60	T5	Sedan	automatic	yv1612tb4f1310987	ca	41	14282	white	black	volvo na rep/world omni	27500	27750	29/01/2015
4	2015	Nissan	Altima	2.5 S	Sedan	automatic	1n4al3ap1fn326013	ca	1	5554	gray	black	enterprise vehicle exchange / tra / rental / t...	15350	10900	30/12/2014
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
42980	2007	Chevrolet	Tahoe	LT	SUV	automatic	1gnfk13047j112818	nc	19	132314	silver	gray	westlake financial services	12950	11900	29/12/2014
42981	2007	Chrysler	Aspen	Limited	SUV	automatic	1a8hx58297f508825	az	21	142317	burgundy	gray	altier credit union	6150	4600	30/12/2014
42982	2007	Chevrolet	Malibu	LS	Sedan	automatic	1g1zs58f37f316984	az	27	96584	silver	gray	camelback finance co	3350	3500	30/12/2014

4.2. Chuẩn hóa dữ liệu

- Import MinMaxScaler từ thư viện sklearn

```
[5]: import pandas as pd
     import seaborn as sns
     import matplotlib.pyplot as plt
     %matplotlib inline
     from sklearn.preprocessing import MinMaxScaler
```

- Thống kê các thuộc tính định lượng trong dataframe

[4]:
```
df.describe().T
```

[4]:

	count	mean	std	min	25%	50%	75%	max
year	513301.0	2010.241679	3.641531	1998.0	2008.0	2012.0	2013.0	2015.0
condition	513301.0	2.437020	1.120242	1.0	1.0	2.0	3.0	4.0
odometer	513301.0	2.499980	1.118031	1.0	1.0	2.0	3.0	4.0
mmr	513301.0	2.497254	1.117604	1.0	1.0	2.0	3.0	4.0
sellingprice	513301.0	2.496765	1.118108	1.0	1.0	2.0	3.0	4.0

- Tiến hành chuẩn hóa dữ liệu bằng phương pháp Min_Max Normalization

```
# Lấy các cột cần chuẩn hóa
cols_to_normalize = ['year', 'condition', 'odometer', 'mmr', 'sellingprice']

# Tạo đối tượng scaler
scaler = MinMaxScaler()

# Chuẩn hóa các cột
df[cols_to_normalize] = scaler.fit_transform(df[cols_to_normalize])
```

- Sau khi chuẩn hóa, độ lệch chuẩn std của các thuộc tính định lượng đã thay đổi (nhỏ hơn 1)

```
df.describe().T
```

[7]:

	count	mean	std	min	25%	50%	75%	max
year	513301.0	0.720099	0.214208	0.0	0.588235	0.823529	0.882353	1.0
condition	513301.0	0.479007	0.373414	0.0	0.000000	0.333333	0.666667	1.0
odometer	513301.0	0.499993	0.372677	0.0	0.000000	0.333333	0.666667	1.0
mmr	513301.0	0.499085	0.372535	0.0	0.000000	0.333333	0.666667	1.0
sellingprice	513301.0	0.498922	0.372703	0.0	0.000000	0.333333	0.666667	1.0

- Sau khi chuẩn hóa các giá trị trong DataFrame bằng phương pháp min-max normalization, ta có thể nhận thấy:
  - Tất cả các cột đều có giá trị nằm trong khoảng từ 0 đến 1. Điều này cho thấy rằng phương pháp chuẩn hóa đã được áp dụng thành công.
  - Các giá trị ở cột "mean" và "std" đều nằm trong khoảng từ 0 đến 1, phản ánh đúng bản chất của min-max normalization là đưa các giá trị về cùng một phạm vi.
  - Các percentile như 25%, 50%, 75% cũng nằm trong khoảng từ 0 đến 1, cho thấy phân bố dữ liệu đã được chuẩn hóa.

- Giá trị min và max đều là 0 và 1 tương ứng, phù hợp với bản chất của min-max normalization.

5.Biểu diễn dữ liệu

Nhìn lại thông tin bộ dữ liệu sau quá trình tiền xử lý thay đổi ra sao

```
: df_updated.info()
  df_updated.head()
```

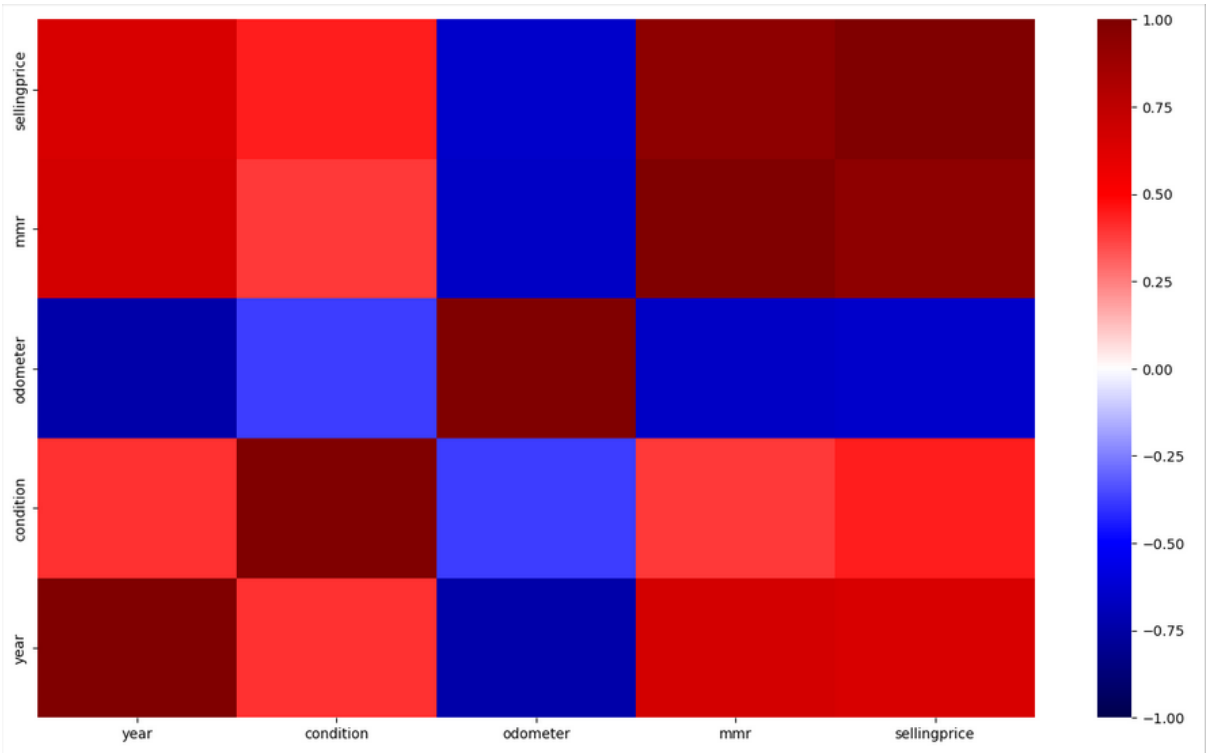
```
<class 'pandas.core.frame.DataFrame'>
Index: 513301 entries, 0 to 558836
Data columns (total 17 columns):
 #   Column              Non-Null Count   Dtype 
---  ------              --------------   ----- 
 0   year                513301 non-null  int64 
 1   make                513301 non-null  object
 2   model               513301 non-null  object
 3   trim                513301 non-null  object
 4   body                513301 non-null  object
 5   transmission        513301 non-null  object
 6   vin                 513301 non-null  object
 7   state               513301 non-null  object
 8   condition           513301 non-null  int64 
 9   odometer            513301 non-null  int64 
 10  color               513301 non-null  object
 11  interior            513301 non-null  object
 12  seller              513301 non-null  object
 13  mmr                 513301 non-null  int64 
 14  sellingprice        513301 non-null  int64 
 15  saledate            513301 non-null  object
 16  saledate_formatted  513301 non-null  object
dtypes: int64(5), object(12)
memory usage: 86.6+ MB
```

Ta có thể thấy được rằng tổng các dòng dữ liệu của các cột đã đồng nhất nhờ xử lý dữ liệu nhiễu và rỗng. Đồng thời các cột kiểu float cũng đã bị biến đổi thành int.

model	trim	body	transmission	vin	state	condition	odometer	color	interior	seller	mmr	sellingprice	saledate	saledate_formatted
Sorento	LX	SUV	automatic	5xyktca69fg566472	ca	1	1	white	black	kia motors america inc	4	4	Tue Dec 16 2014 12:30:00 GMT-0800 (PST)	16/12/2014
Sorento	LX	SUV	automatic	5xyktca69fg561319	ca	1	1	white	beige	kia motors america inc	4	4	Tue Dec 16 2014 12:30:00 GMT-0800 (PST)	16/12/2014
3 Series	328i SULEV	Sedan	automatic	wba3c1c51ek116351	ca	4	1	gray	black	financial services remarketing (lease)	4	4	Thu Jan 15 2015 04:30:00 GMT-0800 (PST)	15/01/2015
S60	T5	Sedan	automatic	yv1612tb4f1310987	ca	3	1	white	black	volvo na rep/world omni	4	4	Thu Jan 29 2015 04:30:00 GMT-0800 (PST)	29/01/2015
Altima	2.5 S	Sedan	automatic	1n4al3ap1fn326013	ca	1	1	gray	black	enterprise vehicle exchange / tra / rental / t...	3	2	Tue Dec 30 2014 12:00:00 GMT-0800 (PST)	30/12/2014

6.Thu giảm số chiều dữ liệu

Nhằm tránh tình trạng dữ liệu có tính tương cao tồn tại trong bộ dữ liệu, ta dựng lên biểu đồ nhiệt hay còn gọi là correlation matrix để tìm và loại bỏ các cột có đột tương quan trên 0,5.



Kết quả thu được cho thấy thuộc tính mmr và sellingprice có mức tương cao. Vì vậy ta sẽ loại bỏ cột mmr để làm tăng hiệu suất làm việc và giảm sự thừa thải trong dữ liệu.

Chương 4. Ứng dụng giải thuật phân lớp vào tập dữ liệu

1. Cây quyết định ID3

Khái niệm Cây quyết định ID3:

ID3 (Iterative Dichotomiser 3) là một thuật toán học cây quyết định (decision tree) được sử dụng để xây dựng mô hình phân loại. Nó được phát triển bởi Ross Quinlan vào năm 1986 và là một trong những thuật toán cây quyết định cổ điển và quan trọng nhất.

Nguyên lý hoạt động của ID3 như sau:

- Xây dựng cây quyết định theo hướng top-down, bắt đầu từ gốc và phân chia dần các nút con.
- Tại mỗi nút, ID3 sẽ chọn thuộc tính tốt nhất (dựa trên một tiêu chí đo lường độ lợi ích như Entropy hoặc Gain Information) để phân chia dữ liệu tại nút đó.
- Tiêu chí đo lường độ lợi ích được sử dụng là Entropy và Gain Information:
  - Entropy đo lường mức độ hỗn loạn/không chắc chắn của dữ liệu.
  - Gain Information đo lường lượng thông tin thu được khi sử dụng một thuộc tính để phân chia dữ liệu.
- ID3 sẽ tiếp tục chia nhánh cho đến khi đạt được các nút lá (leaf nodes) có giá trị dự đoán duy nhất.

Ưu điểm:

- Dễ hiểu và dễ diễn giải, tạo ra mô hình cây quyết định trực quan.
- Không yêu cầu giả định về phân phối dữ liệu.
- Hiệu quả về tính toán, đặc biệt với dữ liệu nhỏ và trung bình.
- Có thể xử lý các thuộc tính liên tục và không liên tục.
- Có khả năng xử lý dữ liệu bị thiếu và nhiễu.

Thuật toán Decision Tree ID3:

Import các thư viện cần thiết

[40]:
```
import pandas as pd
import time
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeRegressor, plot_tree
from sklearn.preprocessing import LabelEncoder
import matplotlib.pyplot as plt
```

Xây dựng cây quyết định hồi quy

[42]:
```
# 2. Xây dựng cây quyết định hồi quy
X = df.drop('sellingprice', axis=1)  # Các thuộc tính đặc trưng
y = df['sellingprice']  # Thuộc tính mục tiêu
```

Chia dữ liệu thành tập huấn luyện và kiểm tra

```
# Chia dữ liệu thành tập huấn luyện và tập kiểm tra
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

Khởi tạo và huấn luyện cây quyết định

```
# Khởi tạo và huấn luyện cây quyết định hồi quy
start_time = time.time()
regressor = DecisionTreeRegressor()
regressor.fit(X_train, y_train)
end_time = time.time()
```

Hiển thị cây quyết định

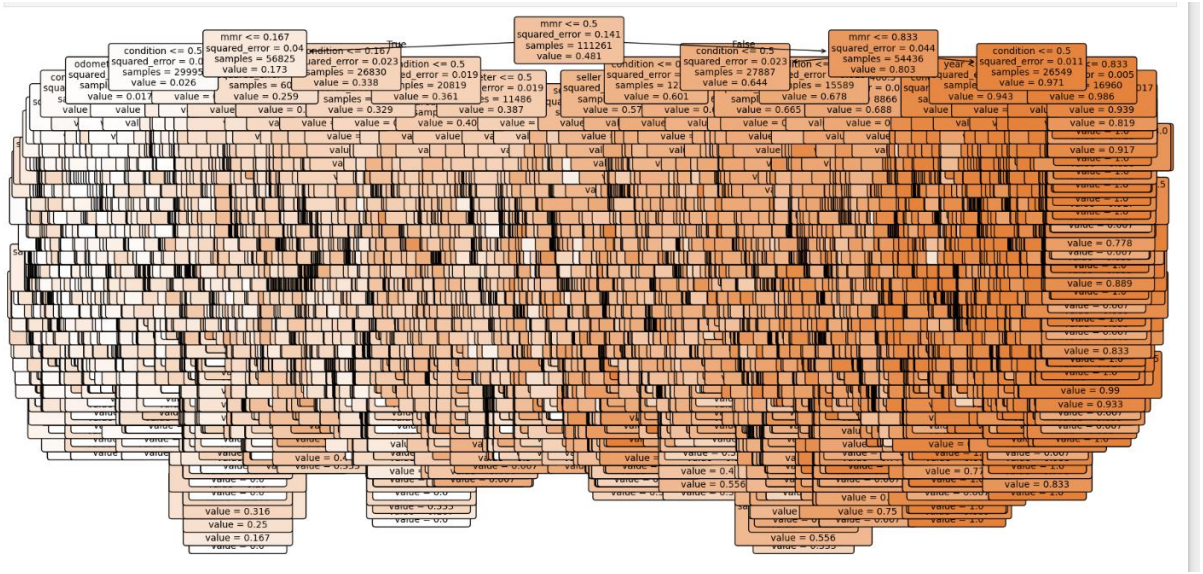
:
```
# 4. Hiển thị cây quyết định bằng biểu đồ
plt.figure(figsize=(20,10))
plot_tree(regressor, feature_names=X.columns, filled=True, rounded=True, fontsize=10)
plt.show()
```

Thời gian chạy thuật toán

[45]:
```
# 3. Lưu lại thời gian chạy thuật toán
execution_time = end_time - start_time
print(f"Thời gian chạy thuật toán: {execution_time:.4f} giây")
```

```
Thời gian chạy thuật toán: 2.1670 giây
```

Kết quả



2. KNN(K – Nearest Neighbour)

Khái niệm KNN:

- KNN là một thuật toán học máy phổ biến được sử dụng trong các bài toán phân loại và hồi quy.
- Nó hoạt động dựa trên ý tưởng rằng các điểm dữ liệu gần nhau trong không gian đặc trưng thường có cùng nhãn hoặc giá trị.
- Để phân loại một điểm dữ liệu mới, KNN tìm k điểm dữ liệu gần nhất trong tập huấn luyện, sau đó gán nhãn hoặc giá trị dự đoán dựa trên đa số hoặc trung bình của các điểm lân cận.

Ưu điểm

- Đơn giản và dễ hiểu: KNN là một thuật toán đơn giản và dễ hiểu, không yêu cầu quá nhiều tiền xử lý dữ liệu.
- Không yêu cầu huấn luyện mô hình: KNN không cần xây dựng một mô hình phức tạp, thay vào đó nó chỉ cần lưu trữ tập huấn luyện và tính toán khoảng cách đến các điểm dữ liệu mới.
- Hiệu quả với dữ liệu không tuyến tính: KNN hoạt động tốt với các bài toán có dữ liệu không tuân theo các giả định tuyến tính như hồi quy tuyến tính.
- Dễ dàng triển khai và áp dụng: KNN có thể được triển khai và áp dụng một cách nhanh chóng, đặc biệt là với các tập dữ liệu nhỏ hoặc trung bình.

- Không yêu cầu giả định về phân phối: KNN không yêu cầu bất kỳ giả định nào về phân phối dữ liệu, điều này làm cho nó trở thành một lựa chọn phù hợp khi không biết phân phối của dữ liệu.
- Dễ dàng giải thích: Việc xác định nhãn hoặc giá trị dự đoán của KNN dựa trên các điểm lân cận là rất dễ hiểu và giải thích.

Thuật toán K – Nearest Neighbour:

Import các thư viện cần thiết

```
[4]: import matplotlib.pyplot as plt
     import seaborn as sns
```

Đọc dữ liệu vào dataframe df

```
# Đọc dữ liệu từ file CSV
data = pd.read_csv('OneDrive - Trường ĐH CNTT - University of Information Technology/Đồ án Khai thác Dữ liệu/update4.csv')

# Làm sạch dữ liệu: Xử lý giá trị thiếu
data.dropna(inplace=True)
```

Thực hiện mã hóa các thuộc tính phân loại

```
# Mã hóa các thuộc tính phân loại
label_encoders = {}
categorical_columns = ['make', 'model', 'trim', 'body', 'transmission', 'state', 'color', 'interior', 'seller']

for col in categorical_columns:
    le = LabelEncoder()
    data[col] = le.fit_transform(data[col])
    label_encoders[col] = le

# Chọn các thuộc tính đầu vào và đầu ra
X = data[['year', 'make', 'model', 'trim', 'body', 'transmission', 'state', 'condition', 'odometer', 'color', 'interior', 'seller', 'mmr']]
y = data['sellingprice']
```

Thực hiện chuẩn bị dữ liệu, chia dữ liệu và chuẩn hóa dữ liệu

```
# Chia dữ liệu thành training set và test set
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Chuẩn hóa dữ liệu
scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)
```

Thực hiện khởi tạo và huấn luyện knn

```
# Triển khai thuật toán KNN
knn = KNeighborsRegressor(n_neighbors=5)
knn.fit(X_train, y_train)
```

Dự đoán và đánh giá thuật toán

```
# Dự đoán trên test set
y_pred = knn.predict(X_test)

# Đánh giá mô hình
mse = mean_squared_error(y_test, y_pred)
rmse = np.sqrt(mse)

print(f'Root Mean Squared Error: {rmse}')

# Bước 2: Vẽ biểu đồ scatter plot để so sánh giá bán thực tế và giá bán dự đoán
plt.figure(figsize=(10, 6))
plt.scatter(y_test, y_pred, alpha=0.3)
plt.xlabel('Actual Selling Price')
plt.ylabel('Predicted Selling Price')
plt.title('Comparison of Actual and Predicted Selling Prices')
plt.plot([min(y_test), max(y_test)], [min(y_test), max(y_test)], color='red')  # Đường chéo y=x để so sánh
plt.show()
```

Hiển thị kết quả bằng biểu đồ scatter

```
Root Mean Squared Error: 2145.621309860744
```

Nhận xét về biểu đồ:

- Đường chéo màu đỏ: Đường chéo màu đỏ là đường $y=x$y=x, biểu thị trường hợp lý tưởng khi giá trị dự đoán hoàn toàn trùng khớp với giá trị

thực tế. Các điểm nằm trên đường này cho thấy mô hình dự đoán chính xác giá trị thực tế.

- Các điểm dữ liệu (chấm xanh): Đa số các điểm dữ liệu nằm gần đường chéo màu đỏ, cho thấy mô hình KNN dự đoán khá chính xác giá bán của các xe. Tuy nhiên một số điểm dữ liệu nằm xa đường chéo màu đỏ, cho thấy có những trường hợp mô hình dự đoán có sai số lớn.
- Phân bố các điểm dữ liệu: Phần lớn các điểm dữ liệu tập trung ở khu vực giá bán thấp (dưới 50,000), cho thấy phần lớn các xe trong tập dữ liệu có giá bán trong khoảng này. Các điểm dữ liệu ở khu vực giá bán cao (trên 100,000) thưa thớt hơn, nhưng vẫn cho thấy mô hình hoạt động khá tốt với các mức giá cao hơn.

3. Naïve Bayes

Khái niệm Naïve Bayes

Naïve Bayes là phương pháp phân loại dựa vào xác suất được sử dụng rộng rãi trong lĩnh vực máy học và nhiều lĩnh vực khác như trong các công cụ tìm kiếm, các bộ lọc mail. Mục đích chính là làm sao tính được xác xuất Pr(Cj,d'), xác suất để tài liệu d' nằm trong lớp Cj.

Ưu điểm

- Dễ dàng áp dụng và dễ hiểu
- Hiệu suất tốt với không gian đặc trưng và dữ liệu lớn.
- Hữu dụng trong bài toán phân loại văn bản

Thuật toán Naïve Bayes

- Chạy thuật toán bằng những dòng lệnh sau:

```
import numpy as np
import pandas as pd
import seaborn as sns
%matplotlib inline
from matplotlib import pyplot as plt
from matplotlib import style
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.naive_bayes import MultinomialNB
from sklearn.metrics import accuracy_score, classification_report
from sklearn.metrics import confusion_matrix
import time

# Đọc dữ liệu từ file CSV
data = pd.read_csv("/content/df_updated2.csv")
```

```
# Đọc dữ liệu từ file CSV
data = pd.read_csv("/content/df_updated2.csv")

# Loại bỏ các hàng có giá trị NaN trong cột 'sellingprice'
data = data.dropna(subset=['sellingprice'])

# Lựa chọn các cột đặc trưng (features) và nhãn (label)
features = data[['year', 'make', 'model', 'body', 'color', 'condition']]
label = data['sellingprice']

# Chuyển đổi các cột văn bản thành các đặc trưng số
vectorizer = CountVectorizer()

# Kết hợp các cột văn bản thành một cột duy nhất để vector hóa
combined_features = features.apply(lambda x: ' '.join(x.astype(str)), axis=1)

# Vector hóa các đặc trưng
X_vectorized = vectorizer.fit_transform(combined_features)

# Chia dữ liệu thành tập huấn luyện và tập kiểm tra
X_train, X_test, y_train, y_test = train_test_split(X_vectorized, label, test_size=0.2, random_state=42)
```

```python
# Khởi tạo và huấn luyện mô hình Naive Bayes
model = MultinomialNB()

# Đo thời gian huấn luyện
start_time = time.time()
model.fit(X_train, y_train)
end_time = time.time()

# Tính thời gian huấn luyện
training_time = end_time - start_time
print(f'Thời gian huấn luyện: {training_time:.2f} giây')

# Đo thời gian dự đoán
start_time = time.time()
y_pred = model.predict(X_test)
end_time = time.time()

# Tính thời gian dự đoán
prediction_time = end_time - start_time
print(f'Thời gian dự đoán: {prediction_time:.2f} giây')

# Đánh giá mô hình
accuracy = accuracy_score(y_test, y_pred)
report = classification_report(y_test, y_pred)
print("Accuracy: ", accuracy)
print("Report: ", report)
```

- Kết quả thời gian huấn luyện, thời gian dự đoán và độ chính xác của mô hình

```
Thời gian huấn luyện: 0.04 giây
Thời gian dự đoán: 0.00 giây
Accuracy:  0.6542637331032499
Report:               precision    recall  f1-score   support

         1.0       0.76      0.79      0.78      7612
         2.0       0.58      0.49      0.53      6767
         3.0       0.56      0.53      0.54      6844
         4.0       0.69      0.79      0.73      6593

    accuracy                           0.65     27816
   macro avg       0.65      0.65      0.65     27816
weighted avg       0.65      0.65      0.65     27816
```

4.Support Vector Machine

Khái niệm:

SVM là một thuật toán giám sát, nó có thể sử dụng cho cả việc phân loại hoặc đệ quy. Tuy nhiên nó được sử dụng chủ yếu cho việc phân loại. Trong thuật toán này, chúng ta vẽ đồi thị dữ liệu là các điểm trong n chiều (ở đây n là số lượng các tính năng bạn có) với giá trị của mỗi tính năng sẽ là một phần liên kết. Sau đó chúng ta thực hiện tìm "đường bay" (Hyper-plane) phân chia các lớp. Hyper-plane nó chỉ hiểu đơn giản là 1 đường thẳng có thể phân chia các lớp ra thành hai phần riêng biệt.

Ưu điểm:

Đây là thuật toán hoạt động hiệu quả với không gian cao chiều (high dimensional spaces). Thuật toán tiêu tốn ít bộ nhớ vì chỉ sử dụng các điểm trong tập hỗ trợ để dự báo trong hàm quyết định. Chúng ta có thể tạo ra nhiều hàm quyết định từ những hàm kernel khác nhau.

Thuật toán Support Vector Machine (SVM)

Chạy thuật toán bằng các dòng lệnh sau:

```
12m
import pandas as pd

# Đọc dữ liệu từ file CSV
df = pd.read_csv('/content/update3.csv')

# Kiểm tra dữ liệu
print(df.head())
print(df.info())

from sklearn.preprocessing import LabelEncoder

# Chọn các cột phân loại cần chuyển đổi
categorical_cols = ['condition', 'make', 'body', 'color', 'sellingprice']

# Khởi tạo LabelEncoder cho từng cột
label_encoders = {col: LabelEncoder() for col in categorical_cols}

# Áp dụng LabelEncoder cho từng cột
for col in categorical_cols:
    df[col] = label_encoders[col].fit_transform(df[col])

# Kiểm tra lại dữ liệu sau khi mã hóa
print(df.head())

from sklearn.model_selection import train_test_split

# Chọn các đặc trưng (features) và nhãn (target)
X = df[['odometer', 'condition', 'make', 'year', 'body', 'color']]
y = df['sellingprice']  # 'price' là cột đích cần dự đoán

# Chia dữ liệu thành tập huấn luyện và tập kiểm tra
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

```python
from sklearn.svm import SVC
from sklearn.preprocessing import StandardScaler
from sklearn.pipeline import Pipeline

# Xây dựng pipeline với scaler và mô hình SVC
pipeline = Pipeline([
    ('scaler', StandardScaler()),
    ('svc', SVC(kernel='linear'))  # Có thể thử các kernel khác như 'rbf'
])

from sklearn.metrics import accuracy_score, classification_report
import time

# Đo thời gian huấn luyện
start_time = time.time()
pipeline.fit(X_train, y_train)
end_time = time.time()

# Tính thời gian huấn luyện
training_time = end_time - start_time
print(f'Thời gian huấn luyện: {training_time:.2f} giây')

# Đo thời gian dự đoán
start_time = time.time()
y_pred = pipeline.predict(X_test)
end_time = time.time()

# Tính thời gian dự đoán
prediction_time = end_time - start_time
print(f'Thời gian dự đoán: {prediction_time:.2f} giây')

# Tính toán và in ra độ chính xác (accuracy)
accuracy = accuracy_score(y_test, y_pred)
print(f'Accuracy: {accuracy}')

# In ra báo cáo chi tiết (classification report)
report = classification_report(y_test, y_pred)
print(f'Classification Report:\n{report}')
```

Kết quả thời gian huấn luyện, thời gian dự đoán và độ chính xác:

```
Thời gian huấn luyện: 642.62 giây
Thời gian dự đoán: 84.80 giây
Accuracy: 0.5442551049755536
Classification Report:
              precision    recall  f1-score   support

           0       0.76      0.86      0.80      7593
           1       0.43      0.41      0.42      6745
           2       0.42      0.26      0.32      6888
           3       0.48      0.62      0.54      6590

    accuracy                           0.54     27816
   macro avg       0.52      0.54      0.52     27816
weighted avg       0.53      0.54      0.53     27816
```

5.Random Forest Model

Khái niệm: Mô hình được huấn luyện dựa trên sự phối hợp giữa luật kết hợp (ensembling*)* và quá trình lấy mẫu tái lặp(boostrapping). Cụ thể thuật toán này tạo ra nhiều cây quyết định mà mỗi cây quyết định được huấn luyện dựa trên nhiều mẫu con khác nhau và kết quả dự báo là bầu cử (voting) từ toàn bộ những **cây quyết định.**

Ưu điểm:

Random Forest cho thấy hiệu quả hơn so với thuật toán phân loại thường được sử dụng vì có khả năng tìm ra thuộc tính nào quan trọng hơn so với những thuộc tính khác.

Trên thực tế, nó còn có thể chỉ ra rằng một số thuộc tính là không có tác dụng trong cây quyết định

khắc phục được hiện tượng quá khớp (overfitting) hay gặp ở cây quyết định

Thuật toán Random Forest:

- Chạy thuật toán bằng những dòng lệnh sau:

```
start_time=time.time()
rf_reg = RandomForestRegressor(random_state=42)
rf_reg.fit(X_train, y_train)
print('Training score:', rf_reg.score(X_train, y_train))
print('Testing score:', rf_reg.score(X_test, y_test))

end_time=time.time()
timeRF=(end_time-start_time)
print('Time to finish:',timeRF,'s')
```

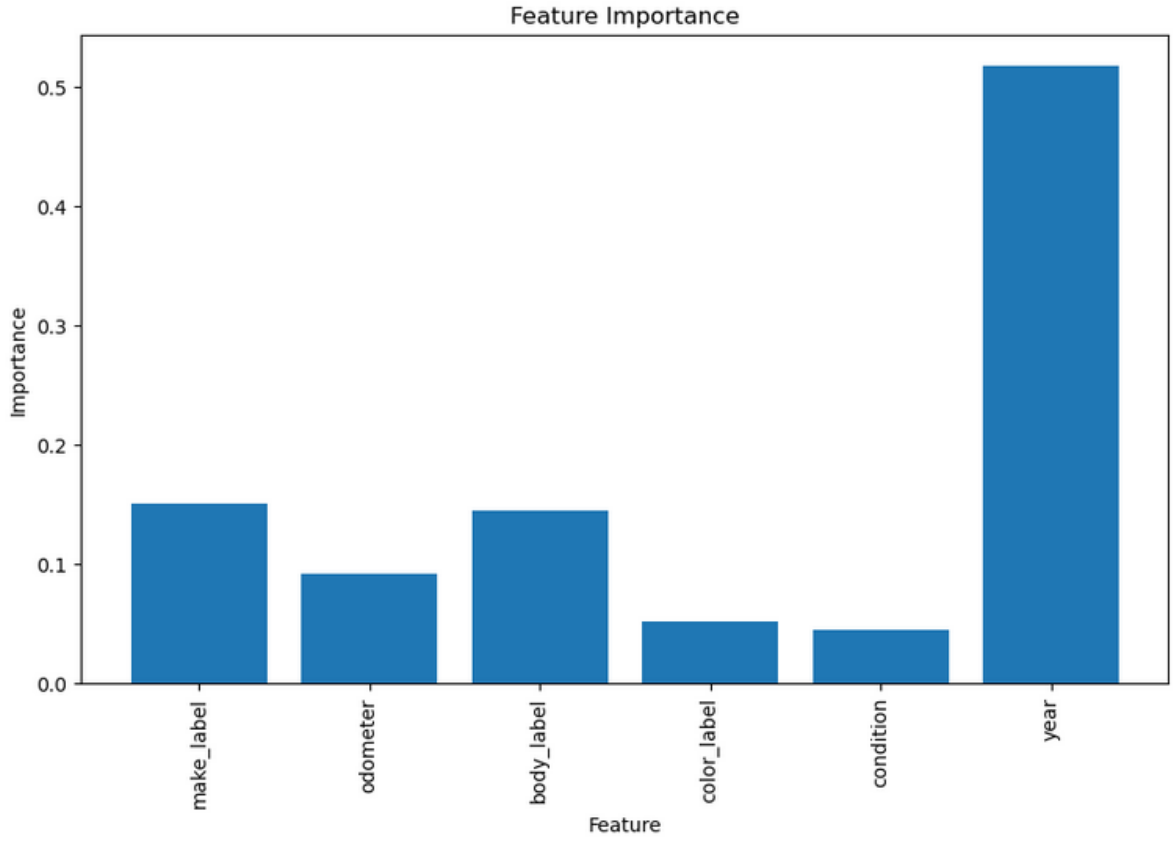
```
Training score: 0.8771707664663216
Testing score: 0.7739521304161305
Time to finish: 33.18163704872131 s
```

Kết quả dự báo

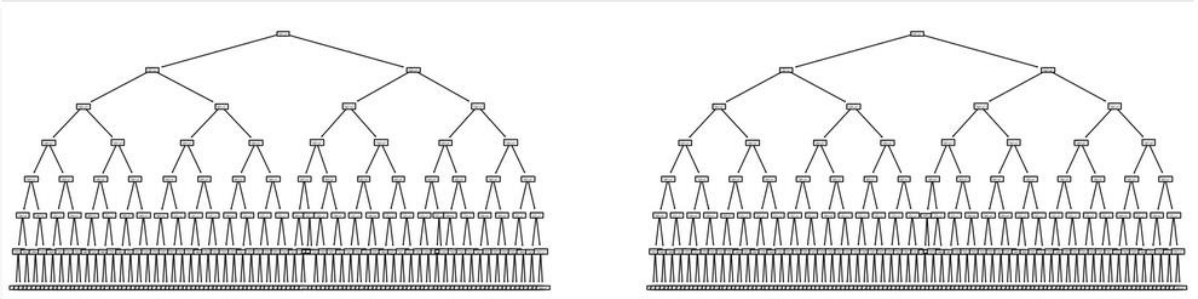
```
y_pred = rf_reg.predict(X_test)
print("Predicted Values:",y_pred)
```

```
Predicted Values: [0.02333333 0.59617752 0.68666667 ... 0.19661111 0.00666667 0.        ]
```

Xây dựng biểu đồ cột tầm quan trọng của thuộc tính được sử dụng trong thuật toán



Trực quan hóa kết quả thuật toán



6.Bagging Regressor

Khái niệm:

Bagging (Bootstrap Aggregating) là một phương pháp Ensemble Learning (học tập kết hợp) được sử dụng trong Machine Learning để nâng cao độ chính xác của mô hình dự đoán. Trong Bagging, chúng ta tạo ra nhiều mô hình dự đoán độc lập với nhau từ các tập dữ liệu con được lấy mẫu ngẫu nhiên với sự thay thế từ tập dữ liệu huấn luyện ban đầu. Sau đó, chúng ta kết hợp các dự đoán của các mô hình này để đưa ra dự đoán cuối cùng

Ưu điểm:

Giảm phương sai bằng cách đào tạo máy học trên các tập hợp con dữ liệu khác nhau, bagging mang lại sự đa dạng giữa các mô hình. Khi các mô hình đa dạng này được kết hợp, các lỗi sẽ bị loại bỏ, dẫn đến các dự đoán ổn định và đáng tin cậy hơn.

Đóng bao giúp chống lại việc quá khớp bằng cách giảm phương sai của mô hình. Bằng cách tạo ra nhiều tập hợp con của dữ liệu huấn luyện đảm bảo tập trung vào các khía cạnh hơi khác nhau của dữ liệu. Sự đa dạng này giúp nhóm khái quát hóa dữ liệu chưa nhìn thấy tốt hơn.

Bagging huấn luyện nhiều mô hình trên các tập con dữ liệu khác nhau nên nó có xu hướng ít nhạy cảm hơn với các điểm ngoại lệ và điểm dữ liệu nhiễu. Các ngoại lệ ít có khả năng tác động đến dự đoán tổng thể khi nhiều mô hình được kết hợp đáng kể.

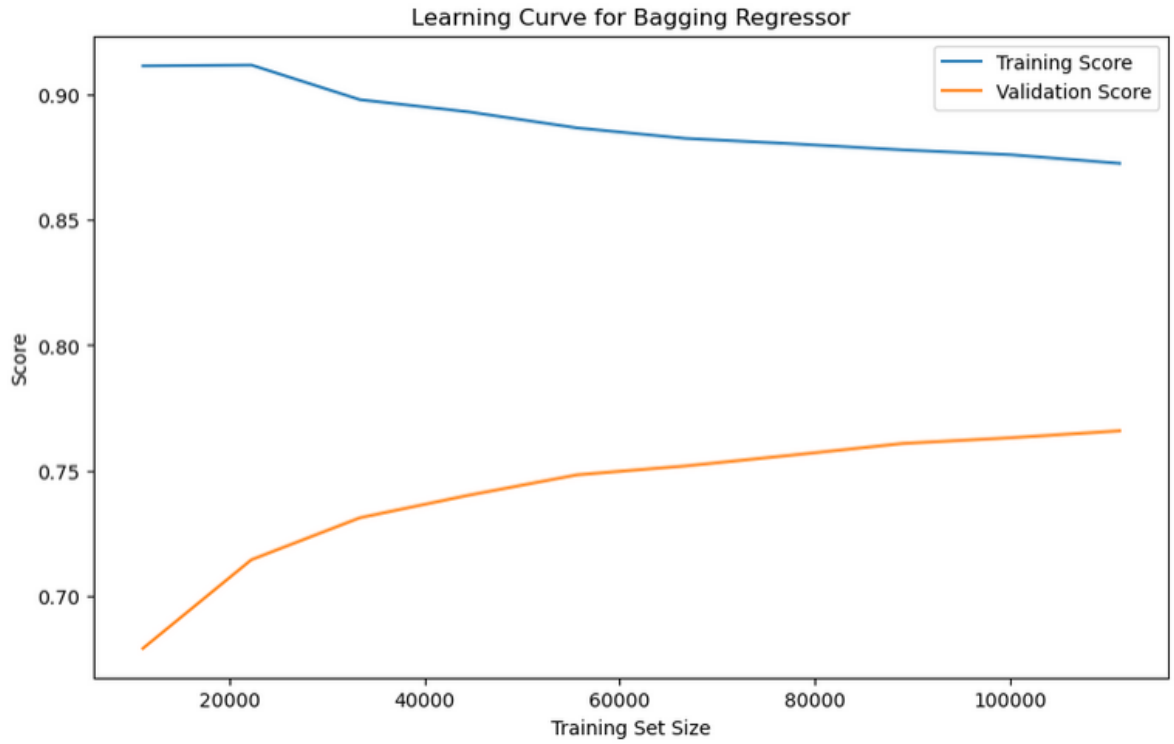
Thuật toán Bagging Regressor

- Chạy thuật toán bằng những dòng lệnh sau:

```
from sklearn.ensemble import BaggingRegressor
start_time=time.time()
bc_reg = BaggingRegressor(random_state=42)
bc_reg.fit(X_train, y_train)
print('Training score:', bc_reg.score(X_train, y_train))
print('Testing score:', bc_reg.score(X_test, y_test))

end_time=time.time()
timeBC=(end_time-start_time)
print('Time to finish:',timeBC,'s')
```

```
Training score: 0.8723431529331973
Testing score: 0.7668024680715813
Time to finish: 2.450010299682617 s
```

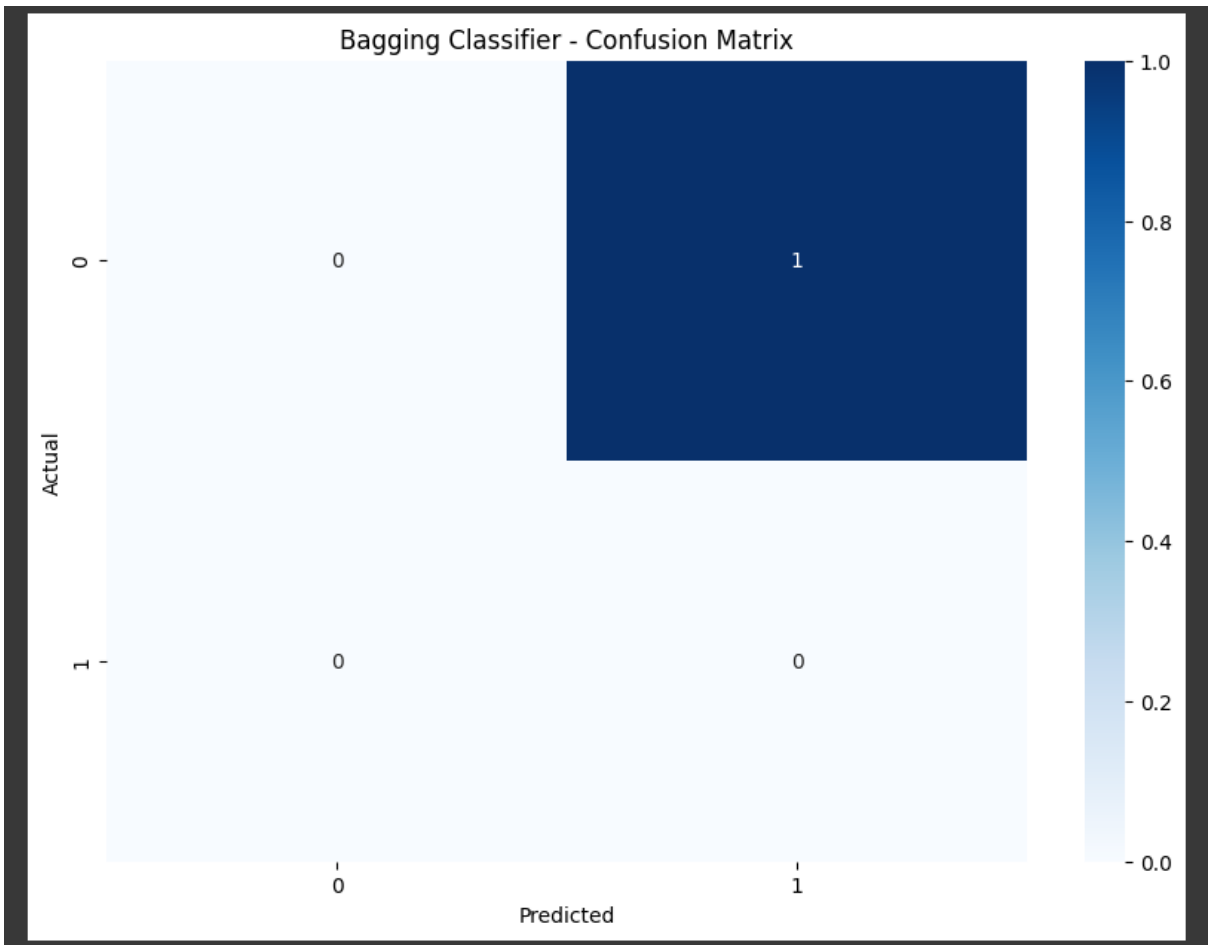


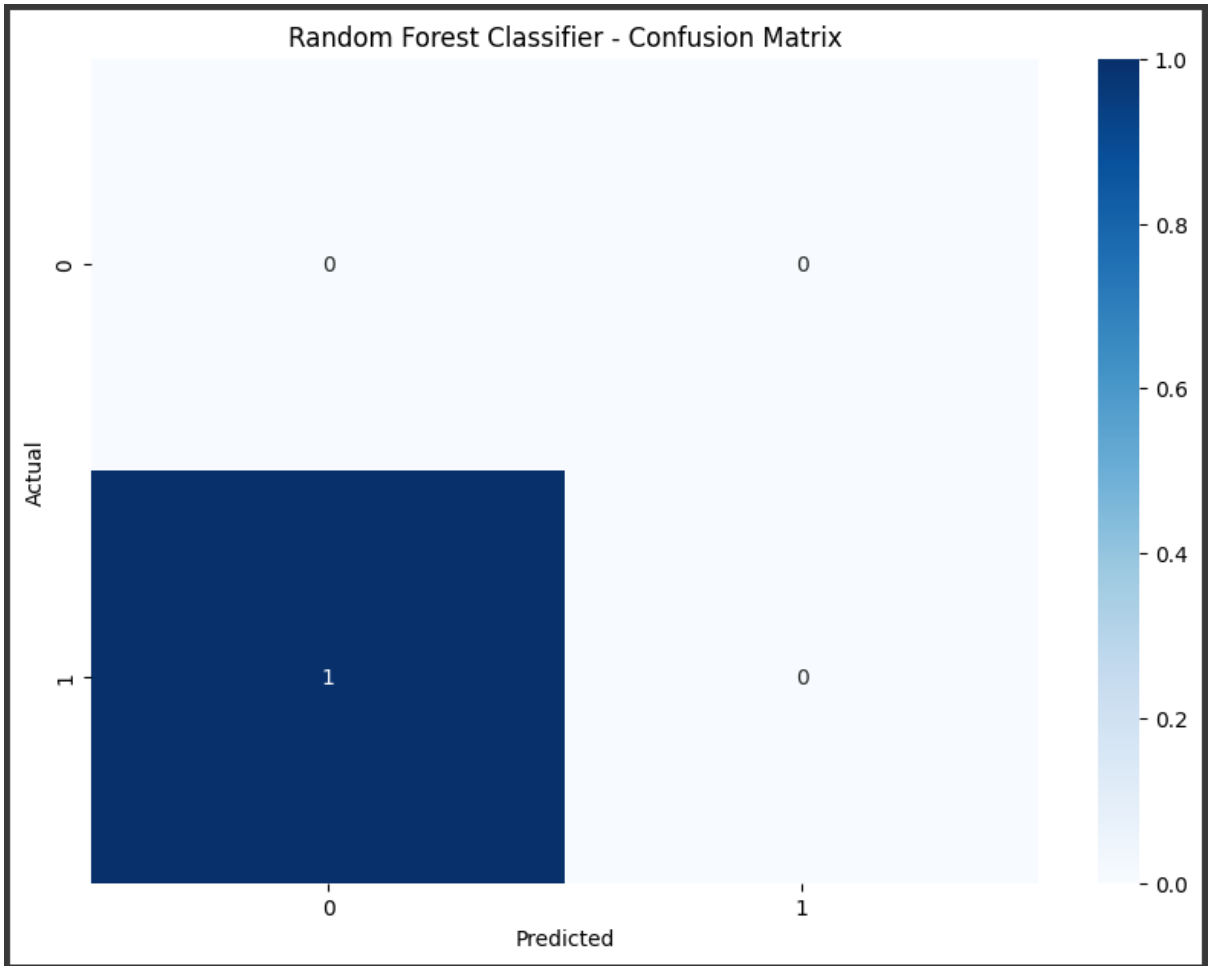
Từ biểu đó trên kết luận được rằng:

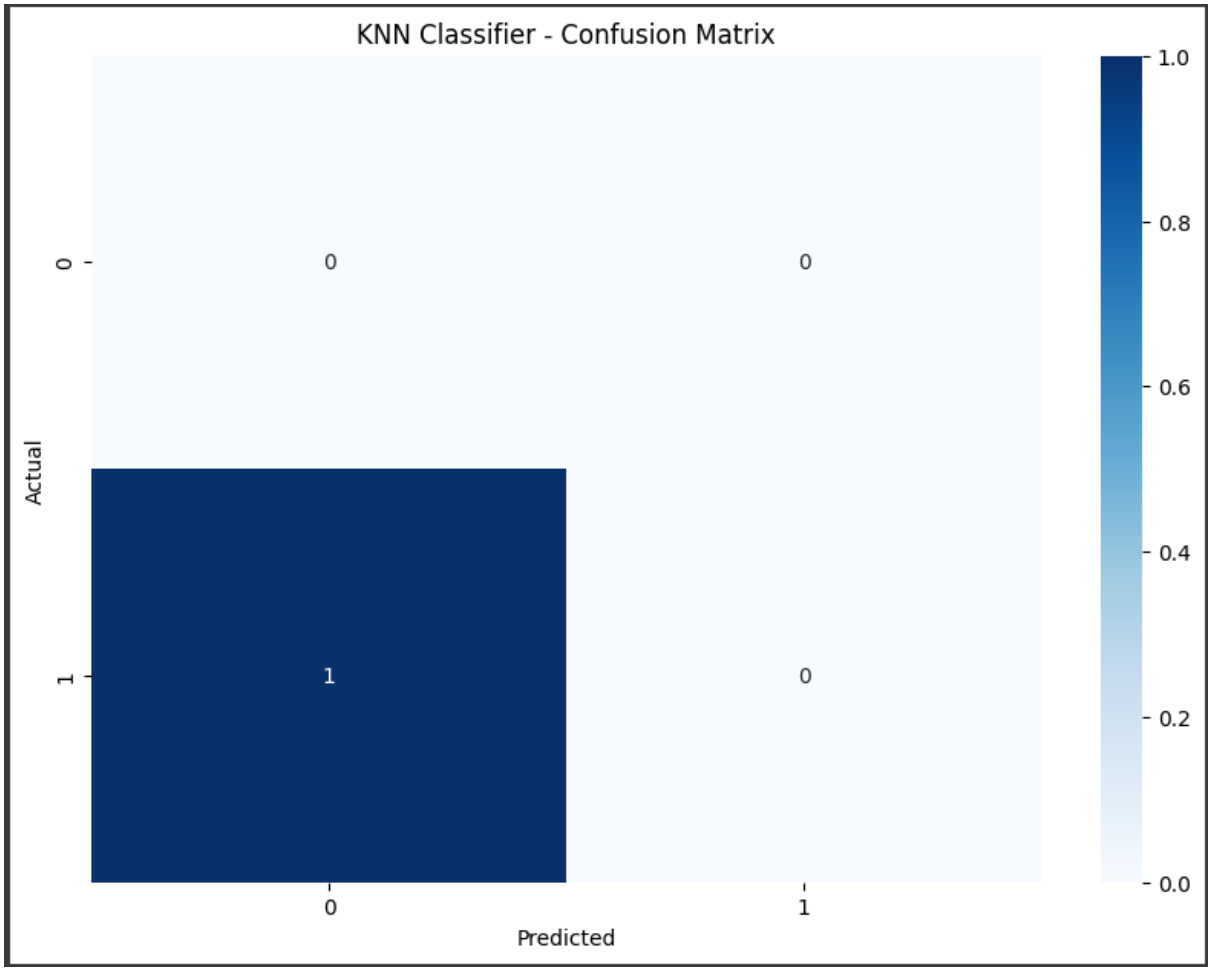
- Model Bagging có khả năng học từ dữ liệu huấn luyện, như được thể hiện qua điểm số huấn luyện cao trên 0.85
- Điểm số xác thực tăng dần khi kích thước tập huấn luyện tăng lên, cho thấy mô hình có thể tổng quát hóa tốt với tập xác thực. Điều này chứng tỏ rằng mô hình không bị quá khớp với dữ liệu huấn luyện.
- Sự giảm nhẹ trong điểm số huấn luyện và tăng trong điểm số xác thực cho thấy mô hình cân bằng tốt giữa độ lệch và phương sai. Mô hình không bị ảnh hưởng nhiều bởi tình trạng thiếu khớp hoặc quá khớp đáng kể.

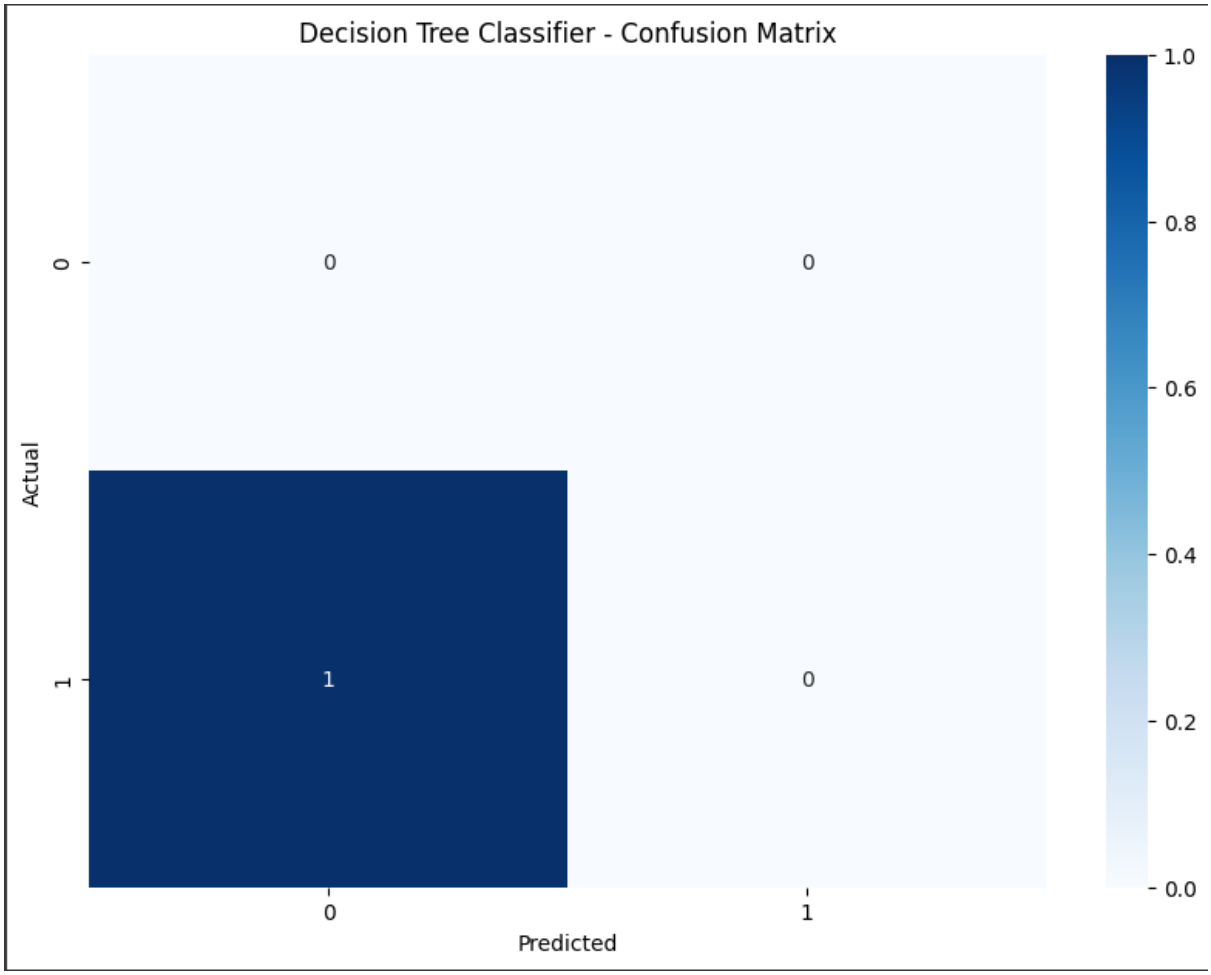
Chương 5. Phân tích đánh giá các thuật toán và dự báo

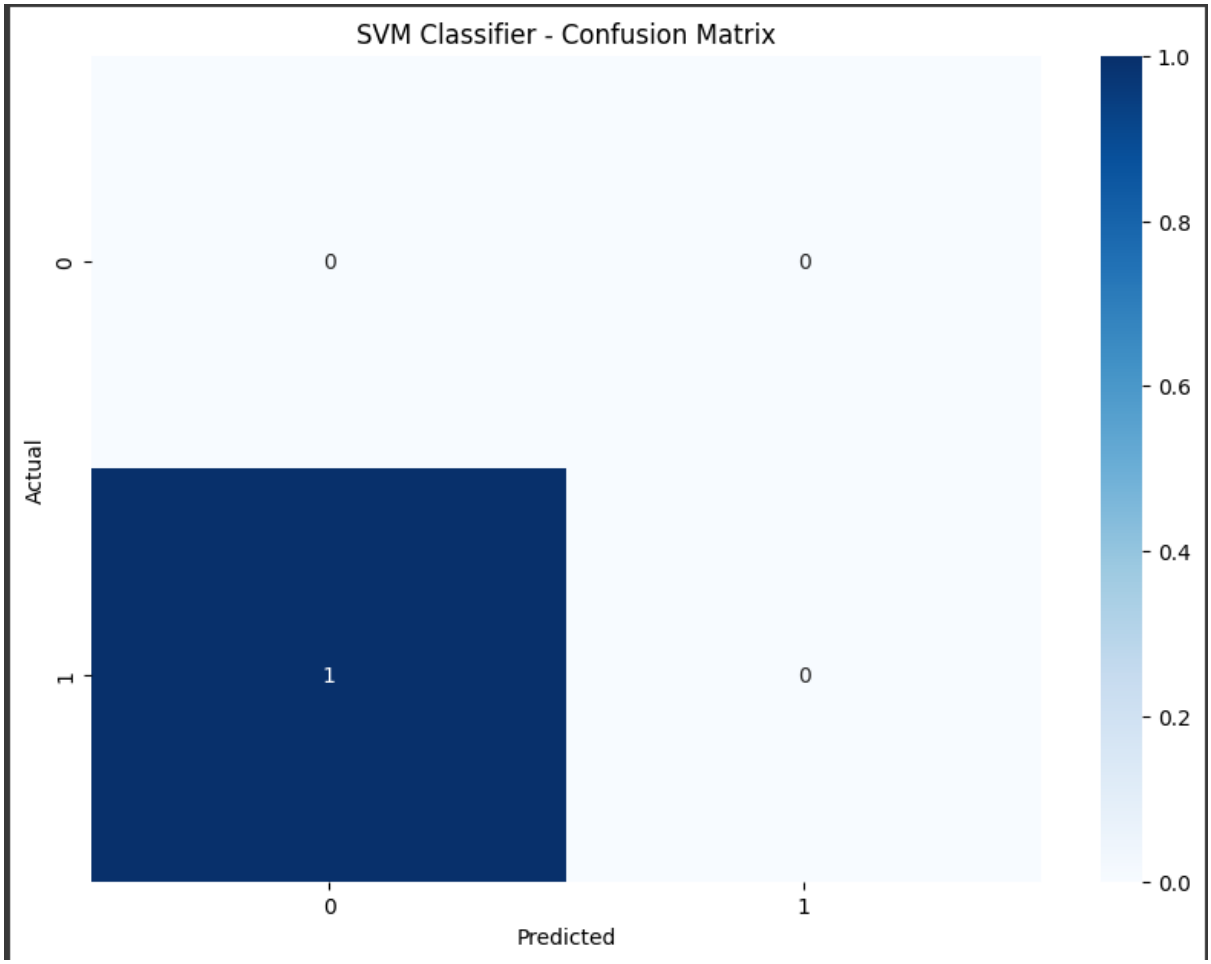
1. Ma trận nhầm nhẫn

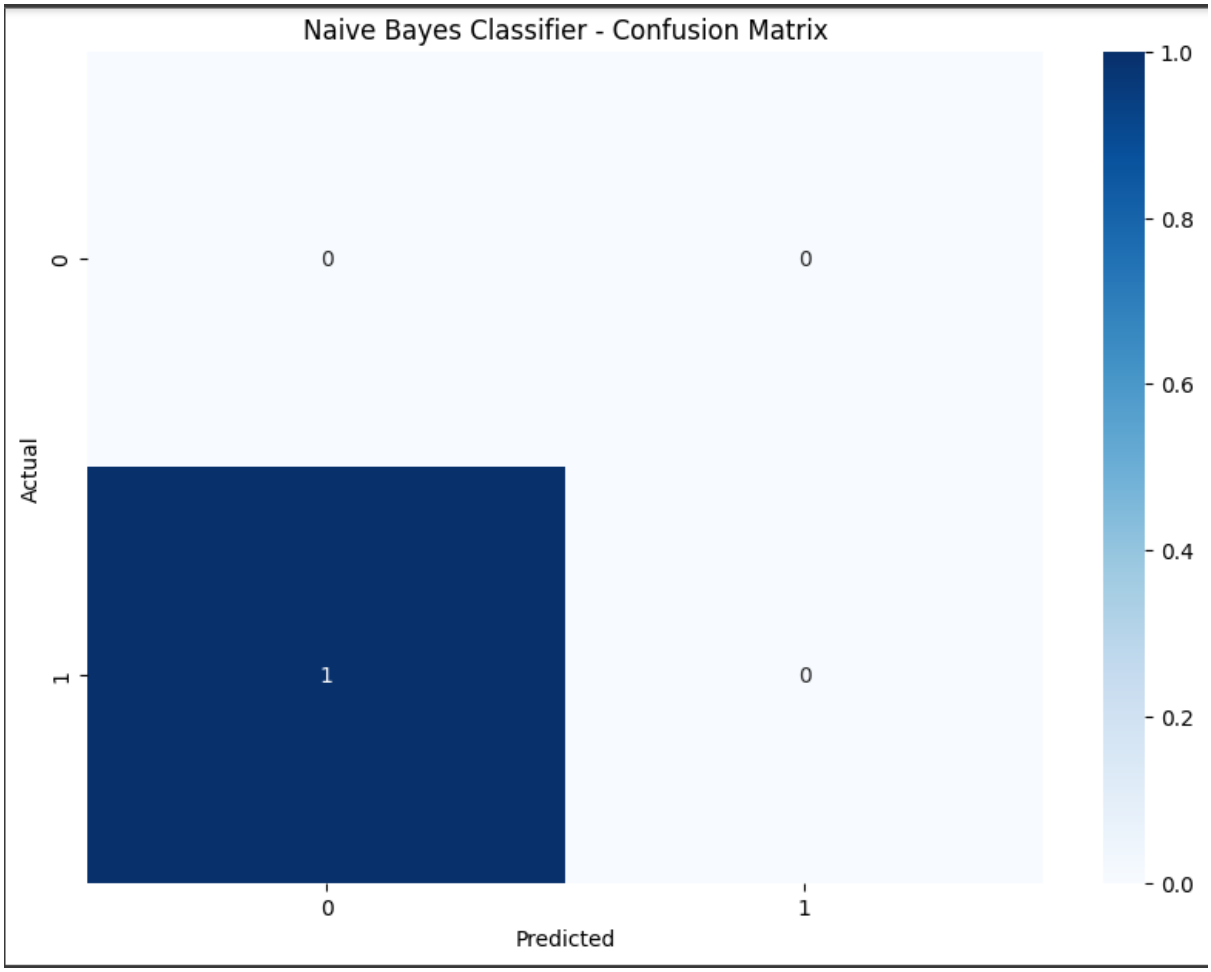



Random Forest Classifier - Confusion Matrix
Actual
Predicted
0
1
0
1
0
0
1
0
1.0
0.8
0.6
0.4
0.2
0.0

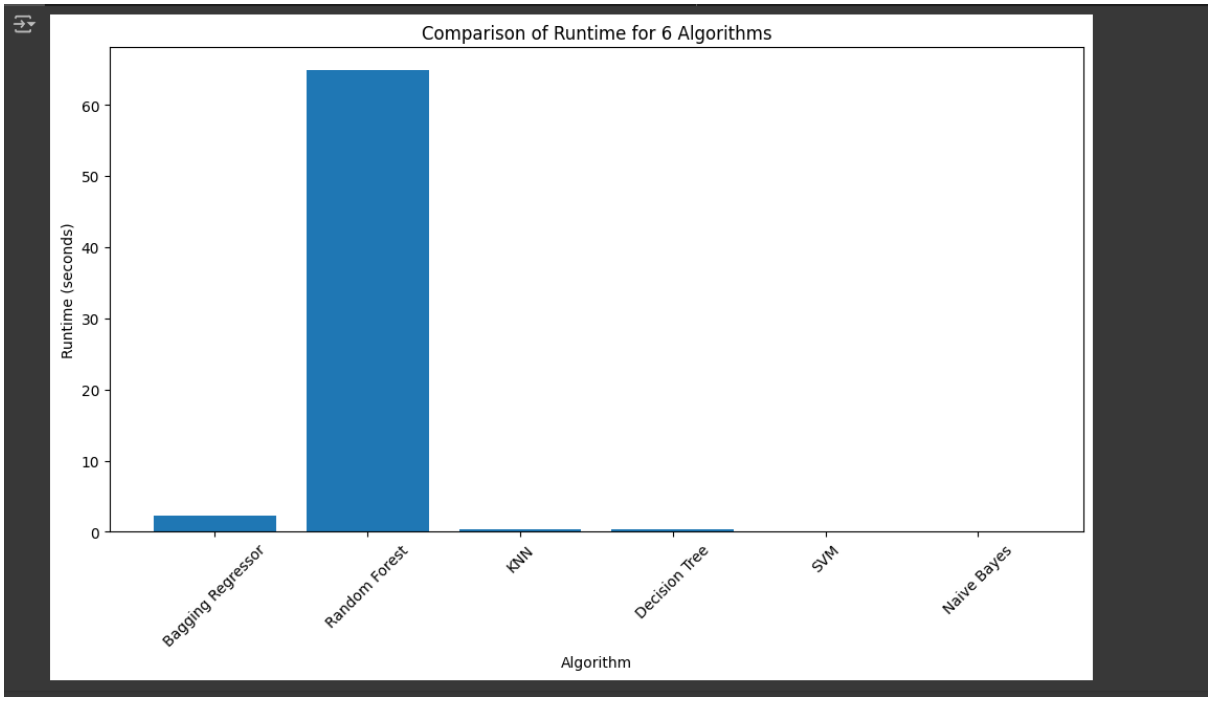

KNN Classifier - Confusion Matrix
Actual
Predicted
0
1
0
1
0
0
1
0
1.0
0.8
0.6
0.4
0.2
0.0


Decision Tree Classifier - Confusion Matrix
Actual
Predicted
0
1
0
1
0
0
1
0
1.0
0.8
0.6
0.4
0.2
0.0


SVM Classifier - Confusion Matrix
Actual
Predicted
0
1
0
1
0
0
1
0
1.0
0.8
0.6
0.4
0.2
0.0

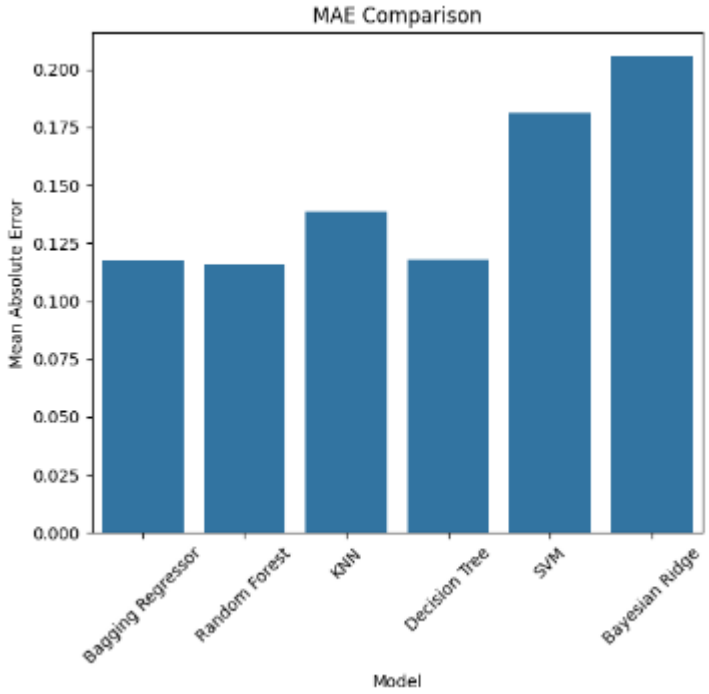


2. Đánh giá về thời gian chạy thuật toán



- Thời gian chạy thuật toán Random Forest là lớn nhất.
- Hai thuật toán SVM và Naïve Bayes có thời gian chạy gần như = 0.

3. Đánh giá về độ chính xác của thuật toán

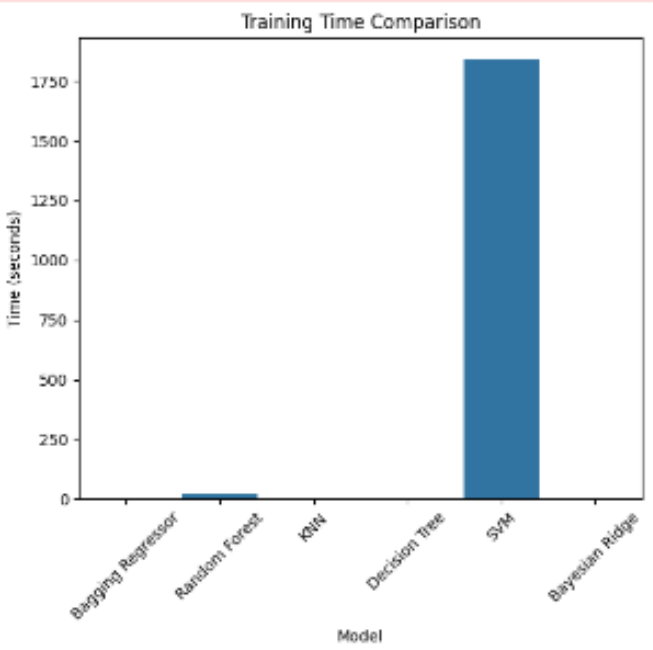


- Thuật toán Bagging Regressor và Random forest thường có độ chính xác cao nhất với chỉ số MAE thấp, cho thấy rằng chúng dự đoán rất gần với giá trị thực tế.
- Thuật toán Decision Tree và KNN có thể có MAE cao hơn một chút, chỉ ra rằng chúng có thể ít chính xác hơn trong một số trường hợp.
- Thuật toán SVM và Naïve Bayes cũng hoạt động tốt, nhưng hiệu suất của SVM phụ thuộc nhiều vào việc chọn đúng tham số, trong khi Bayesian Ridge mang lại sự ổn định trong dự đoán.

Khi so sánh các mô hình này, chỉ số MAE thấp hơn đồng nghĩa với độ chính xác cao hơn, do đó, lựa chọn mô hình có MAE thấp nhất là lựa chọn tốt nhất cho mục tiêu dự đoán chính xác.

4. Đánh giá thuật toán

1. So sánh thời gian chạy



Ta thấy được rằng thuật toán SVM tốn nhiều thời gian nhất. Điều này do rằng thuật toán không phù hợp với bộ dữ liệu quy mô lớn vì biến đổi chiều không gian O(n^2) và thời gian O(n^3) [1]

2.So sánh độ chính xác của thuật toán

Cây quyết định ID3

```
Training score: 0.78
Testing score: 0.76
```

Knn:

```
Training score: 0.8063
```

Random Forest

```
------Random Forest------
Training score: 0.8771707664663216
Testing score: 0.7739521304161305
```

SVM:

```
Accuracy: 0.5442551049755536
```

Naïve Bayes

```
Accuracy:  0.6542637331032499
```

Bagging Regressor:

```
Training score: 0.8515582746260139
Testing score: 0.7781114596132582
```

Ta thấy rằng thuật toán Bagging Regressor và Random Forest đem lại độ chính xác cao nhất nhờ vào độ phức tạc của dữ liệu cũng như đặt tính tập kết hợp và chống quá khớp của thuật toán.

Phân chia công việc

	Phùng Thiên Phúc - 21521297	Nguyễn Quang Lâm - 21521057	Phạm Mạnh Hùng - 21520901
Chọn bộ dữ liệu và ý tưởng đề tài	Hỗ trợ	Hỗ trợ	X
Chương 1: Nhận diện bài toán	Hỗ trợ	Hỗ trợ	X
Chương 2: Trực quan hóa dữ liệu	Hỗ trợ	X	X
Chương 3: Tiền xử lý dữ liệu			
Mô tả dữ liệu, Làm sạch dữ liệu, Rời rạc hóa dữ liệu, Biểu diễn dữ liệu, Thu giảm số chiều dữ liệu	X	Hỗ trợ	Hỗ trợ
Tích hợp dữ liệu	Hỗ trợ	Hỗ trợ	X
Biến đổi dữ liệu và thu giảm dữ liệu	Hỗ trợ	X	Hỗ trợ
Chương 4. Ứng dụng giải thuật phân lớp vào tập dữ liệu			
Cây quyết định ID B3 và KNN	Hỗ trợ	Hỗ trợ	X
Naïve Bayes và Support Vector Machine	Hỗ trợ	X	Hỗ trợ

Random Forest Model và Bagging	X	Hỗ trợ	Hỗ trợ
Chương 5. Phân tích đánh giá các thuật toán và dự báo			
Ma trận nhầm lẫn và đánh giá tỉ lệ TPR và FPR	Hỗ trợ	X	Hỗ trợ
Đánh giá thời gian chạy và độc chính xác của thuật toán	Hỗ trợ	Hỗ trợ	X
Đánh giá thuật toán	X	Hỗ trợ	Hỗ trợ

TÀI LIỆU THAM KHẢO

[1] https://typeset.io/papers/fast-svm-training-using-edge-detection-on-very-large-4xo9qvfp6l