Đại học Quốc gia thành phố Hồ Chí Minh
Trường Đại học Bách Khoa
Khoa Khoa học và Kỹ thuật Máy tính

HỌC MÁY (MACHINE LEARNING)
L02 - CO3117

BÁO CÁO BÀI TẬP LỚN:

"Phân tích và dự đoán giá xe Audi"

GVHD: **Th.s.Võ Thanh Hùng**

SV:
1. Lê Phương Vũ 2313954
2. Nguyễn Thanh Lộc 2311958
3. Đặng Quốc Bảo 2210200
4. Nguyễn Trọng Thắng 1915244

TP. HỒ CHÍ MINH, THÁNG 11/2025

Danh sách thành viên & Phân công

No.	Họ và tên	Nhiệm vụ	% Hoàn thành
1	Lê Phương Vũ	- Tổng hợp báo cáo - Mô hình Support Vector Machine	100%
2	Nguyễn Thanh Lộc	- Vẽ biểu đồ và đánh giá - Mô hình Linear Regression	100%
3	Đặng Quốc Bảo	- Hỗ trợ xử lý & phân tích dữ liệu - Mô hình Random Forest	100%
4	Nguyễn Trọng Thắng	- So sánh tổng hợp, quản lý dự án - Mô hình Deep Learning (MLP), Ensemble Learning	100%

Table 1: Danh sách thành viên & Phân công

Mục lục

Danh sách hình vẽ

Danh sách bảng

1 GIỚI THIỆU

1.1 Bối cảnh và Đặt vấn đề

Trong kỷ nguyên dữ liệu lớn, khả năng dự đoán các xu hướng tương lai dựa trên dữ liệu lịch sử đóng vai trò then chốt trong nhiều lĩnh vực, từ tài chính, y tế đến khoa học máy tính. Tuy nhiên, việc đưa ra các dự đoán thủ công thường tốn nhiều thời gian, mang tính chủ quan và thiếu độ chính xác cao.

Đặc biệt hiện nay thị trường ô tô đã qua sử dụng đang chứng kiến sự tăng trưởng mạnh mẽ, vượt xa thị trường xe mới về số lượng giao dịch tại nhiều quốc gia. Khác với xe mới có giá niêm yết cố định từ nhà sản xuất, giá trị của một chiếc xe cũ chịu ảnh hưởng bởi một tổ hợp phức tạp các yếu tố như: năm sản xuất, số dặm đã đi, tình trạng kỹ thuật, loại nhiên liệu và thuế phí.

Hiện nay, việc định giá xe cũ phần lớn dựa vào kinh nghiệm chủ quan của các chuyên gia thẩm định hoặc so sánh thủ công với các tin đăng bán tương tự. Phương pháp này thường thiếu nhất quán, tốn kém thời gian và dễ dẫn đến sai lệch, gây thiệt hại kinh tế cho cả người mua lẫn người bán.

Xuất phát từ thực tế đó, đề tài tập trung vào bài toán **"Dự đoán giá xe Audi đã qua sử dụng"** thông qua việc áp dụng mô hình Hồi quy tuyến tính, Rừng ngẫu nhiên (Random forest) và máy vector hỗ trợ (SVM). Vấn đề đặt ra là làm thế nào để xây dựng một mô hình toán học có khả năng học từ dữ liệu đầu vào và đưa ra kết quả dự báo định lượng với sai số thấp nhất.

1.2 Mục tiêu nghiên cứu

Mục tiêu là xây dựng một công cụ định lượng có khả năng học từ dữ liệu lịch sử để đưa ra mức giá tham khảo chính xác và khách quan.

- **Mục tiêu chính:** Xây dựng và huấn luyện mô hình hồi quy tuyến tính, Rừng ngẫu nhiên (Random forest) và máy vector hỗ trợ (SVM) để dự đoán biến mục tiêu giá xe dựa trên tập hợp các biến đặc trưng.
- **Mục tiêu phụ:** Đánh giá hiệu suất của mô hình thông qua các chỉ số sai số và phân tích mức độ ảnh hưởng của các thuộc tính dữ liệu.

1.3 Phạm vi

Nghiên cứu sử dụng bộ dữ liệu **Audi Used Car Dataset**, một phần trong bộ dữ liệu "100,000 UK Used Car Data set" được thu thập và công bố rộng rãi trên nền tảng Kaggle.

Bộ dữ liệu bao gồm thông tin chi tiết về các dòng xe Audi được rao bán tại thị trường Vương quốc Anh. Cấu trúc dữ liệu bao gồm biến mục tiêu là `price` (giá xe) và các thuộc tính đặc trưng (features) mô tả xe.

Table 2: Mô tả các thuộc tính trong bộ dữ liệu Audi

STT	Tên thuộc tính	Mô tả
1	`model`	Dòng xe (Ví dụ: A1, A3, Q5...). Đây là biến định danh (Categorical).
2	`year`	Năm sản xuất của xe.
3	`price`	**Biến mục tiêu (y):** Giá niêm yết của xe (đơn vị: Bảng Anh/USD).
4	`transmission`	Loại hộp số (Manual, Automatic, Semi-Auto).
5	`mileage`	Số dặm xe đã di chuyển. Chỉ số này phản ánh mức độ khấu hao của xe.
6	`fuelType`	Loại nhiên liệu (Petrol, Diesel, Hybrid...).
7	`tax`	Thuế đường bộ (Road Tax).
8	`mpg`	Miles Per Gallon - Mức tiêu thụ nhiên liệu.
9	`engineSize`	Dung tích động cơ (lít).

Đặc điểm của bộ dữ liệu này là sự đa dạng về các dòng xe (từ dòng phổ thông như A1 đến dòng cao cấp như R8) và dải giá rộng, tạo ra thách thức phù hợp để đánh giá độ chính xác của mô hình hồi quy.

1.4 Tầm quan trọng của bài toán

Việc giải quyết bài toán dự đoán giá xe mang lại ý nghĩa thực tiễn to lớn trên nhiều khía cạnh:

- **Đối với người tiêu dùng:** Giúp người mua tránh bị mua hớ và người bán định giá sản phẩm hợp lý, tăng tính thanh khoản cho tài sản.
- **Đối với doanh nghiệp kinh doanh ô tô:** Tự động hóa quy trình định giá đầu vào và đầu ra, tối ưu hóa biên lợi nhuận và giảm thiểu rủi ro tồn kho.
- **Đối với thị trường:** Góp phần minh bạch hóa thông tin, giảm thiểu tình trạng bất cân xứng thông tin (information asymmetry) trong giao dịch.
- **Ý nghĩa học thuật:** Là ví dụ điển hình để kiểm chứng hiệu quả của các thuật toán học máy giám sát trên dữ liệu dạng bảng (tabular data) với các đặc trưng hỗn hợp (vừa định lượng vừa định tính).

2 TỔNG QUAN DỮ LIỆU: AUDI USED CAR DATA

2.1 Ngữ cảnh

Bộ dữ liệu **audi.csv** chứa thông tin về các xe Audi đã qua sử dụng, được lấy từ https://www.kaggle.com/datasets/mysarahmadbhat/audi-used-car-listings. Mục tiêu của bộ dữ liệu là cung cấp cái nhìn tổng quan về các yếu tố ảnh hưởng đến giá bán của xe, từ đó có thể xây dựng mô hình dự đoán giá xe đã qua sử dụng.

Các thông tin có trong bộ dữ liệu bao gồm:

1. **Thông tin cơ bản về xe:**
   - **Mẫu xe (model)**: xác định kiểu xe cụ thể, ví dụ A1, A3, A4, A6.
   - **Năm đăng ký (year)**: năm sản xuất hoặc đăng ký của xe, giúp đánh giá tuổi xe, một yếu tố quan trọng ảnh hưởng đến giá.
2. **Thông tin kỹ thuật và vận hành:**
   - **Hộp số (transmission)**: loại hộp số tự động hoặc số sàn, ảnh hưởng đến hiệu năng và giá bán.
   - **Loại nhiên liệu (fuelType)**: Petrol, Diesel, v.v., giúp đánh giá chi phí vận hành và hiệu suất.
   - **Quãng đường đã đi (mileage)**: tổng số km xe đã chạy, thường tỷ lệ nghịch với giá bán.
   - **Dung tích động cơ (engineSize)**: thể hiện kích thước động cơ (lít), ảnh hưởng đến công suất và mức tiêu thụ nhiên liệu.
   - **Mức tiêu thụ nhiên liệu (mpg)**: số dặm đi được trên mỗi gallon, phản ánh hiệu quả nhiên liệu và chi phí sử dụng.
3. **Thông tin tài chính:**
   - **Giá bán (price)**: giá hiện tại của xe, là biến mục tiêu quan trọng để dự đoán.
   - **Thuế đường bộ (tax)**: chi phí thuế theo xe, có thể ảnh hưởng đến tổng chi phí sở hữu.

Mỗi bản ghi trong dữ liệu đại diện cho một xe Audi cụ thể, với các biến số cung cấp cả **thông tin định lượng** (như giá, số km đã đi, dung tích động cơ, mpg) và **thông tin phân loại** (như model, transmission, fuelType).

Bộ dữ liệu này có thể sử dụng để dự đoán giá xe đã qua sử dụng, hỗ trợ cả người bán lẫn người mua trong việc đưa ra quyết định giá hợp lý.

2.2 Làm sạch dữ liệu

Qua quan sát ban đầu, nhóm nhận thấy dữ liệu tồn tại một số điểm nhiễu cần xử lý:

- Một số tên cột chứa khoảng trắng thừa.
- Một số giá trị dạng chuỗi, như `model`, `transmission`, và `fuelType`, cũng chứa khoảng trắng thừa.

Để đảm bảo tính nhất quán và tránh lỗi trong quá trình mã hóa dữ liệu dạng phân loại, nhóm thực hiện:

- Loại bỏ khoảng trắng thừa ở tên cột bằng phương pháp `df.columns.str.strip()`.
- Chuẩn hóa dữ liệu chuỗi ở các cột dạng object bằng `df[col].str.strip()`.

```
df = pd.read_csv('audi.csv')
df.columns = df.columns.str.strip()
for col in df.select_dtypes(include='object'):
    df[col] = df[col].str.strip()
```

Sau bước làm sạch, dữ liệu đã sẵn sàng cho quá trình khám phá và phân tích.

2.3 Khám phá dữ liệu (Exploratory Data Analysis - EDA)

Nhóm tiến hành phân tích dữ liệu tổng quan nhằm hiểu rõ các đặc trưng và mối quan hệ giữa các biến:

2.3.1 Thông tin tổng quan

- **Kích thước dữ liệu:** 10,668 mẫu, 9 cột.
- **Loại dữ liệu:** 3 cột phân loại (model, transmission, fuelType), 6 cột số (year, price, mileage, tax, mpg, engineSize).
- **Giá trị thiếu:** không có giá trị bị thiếu.

2.3.2 Thống kê mô tả các biến số định lượng

Table 3: Thống kê mô tả các biến số định lượng

Statistic	Year	Price (EUR)	Mileage (km)	Engine Size (L)
Count	10,668	10,668	10,668	10,668
Mean	2,017.10	22,896.69	24,827.24	1.93
Std	2.17	11,714.84	23,505.26	0.60
Min	1,997	1,490.00	1.00	0.00
25%	2,016	15,130.75	5,968.75	1.50
50% (Median)	2,017	20,200.00	19,000.00	2.00
75%	2,019	27,990.00	36,464.50	2.00
Max	2,020	145,000.00	323,000.00	6.30

Dựa vào bảng số liệu, ta có những nhận xét sơ bộ:

- **Giá xe (Price):** Độ lệch chuẩn lớn (11,714 EUR) cho thấy sự biến động giá rất cao giữa các dòng xe.
- **Năm sản xuất (Year):** Dữ liệu tập trung vào các xe đời mới, với 75% số lượng xe được sản xuất từ năm 2016 trở lại đây.
- **Quãng đường (Mileage):** Tương tự giá xe, quãng đường đi được cũng có biên độ dao động rất lớn, từ xe "lướt" (vài km) đến xe đã vận hành cường độ cao ($> 300{,}000$ km).

2.3.3 Phân tích phân bố và giá trị ngoại lai

Để hiểu rõ hơn về cấu trúc dữ liệu, nhóm nghiên cứu tiến hành trực quan hóa thông qua biểu đồ phân phối (Histogram) và biểu đồ hộp (Boxplot).

a. Hình dáng phân phối (Histogram)

Quan sát Hình 1, ta nhận thấy:

- **Biến Price và Mileage:** Cả hai đều có phân phối **lệch phải (right-skewed)**. Đỉnh của đồ thị tập trung ở vùng giá trị thấp (Giá $< 30{,}000$ EUR và Mileage $< 40{,}000$ km), đuôi đồ thị kéo dài về phía bên phải. Điều này phản ánh thực tế thị trường xe cũ Audi chủ yếu giao dịch các dòng xe phổ thông và xe chạy lướt.
- **Biến Year:** Có phân phối lệch trái, tập trung mật độ cao ở các năm 2015-2020.

- **Biến EngineSize:** Phân phối không liên tục mà tập trung thành các đỉnh (multimodal) tại các dung tích phổ biến như 1.4L, 2.0L và 3.0L.

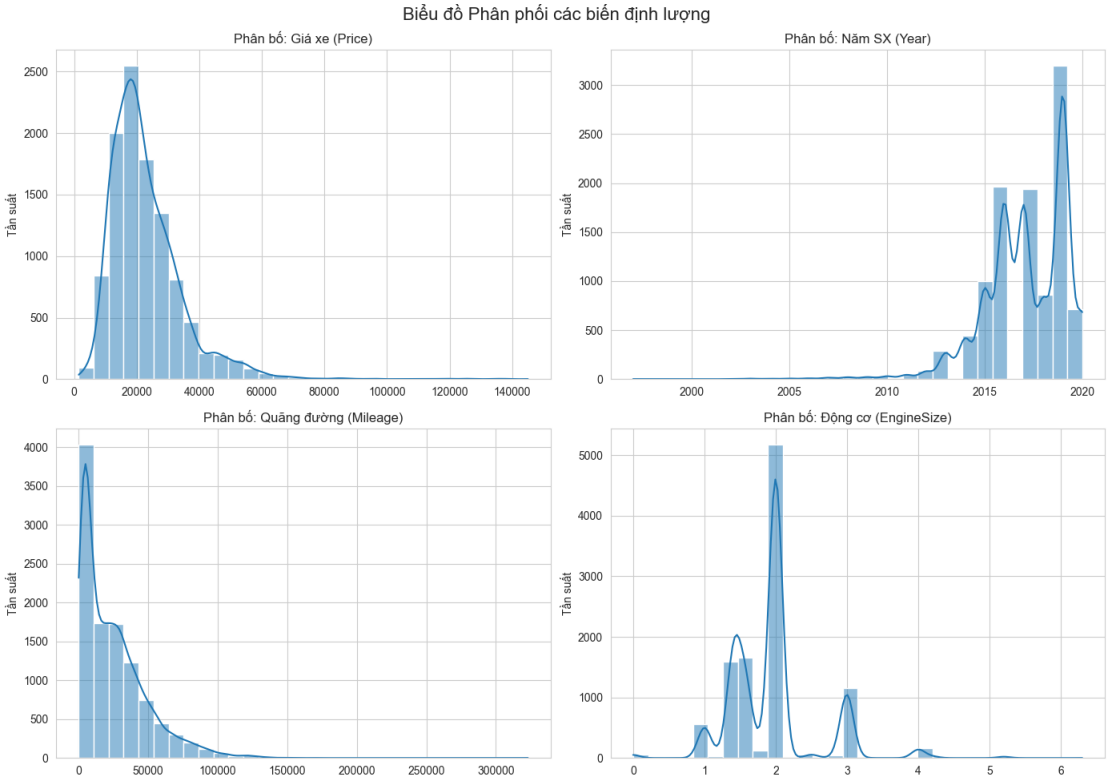

Figure 1: Biểu đồ phân phối (Histogram) của các biến định lượng.

b. Phân tích giá trị ngoại lai (Boxplot)

Biểu đồ hộp ở Hình 2 giúp nhận diện các điểm dữ liệu bất thường:

- **Price:** Xuất hiện dày đặc các điểm ngoại lai (chấm đen) ở phía trên mức giá 60,000 EUR. Đây có thể là các dòng xe hiệu suất cao (như dòng R8, RS) hoặc siêu xe.
- **Mileage:** Tương tự, có nhiều xe có số km vượt trội ($> 150{,}000$ km) so với mặt bằng chung.
- **EngineSize:** Có một số ngoại lai ở mức 0.0 (cần kiểm tra xem có phải lỗi dữ liệu không) và các động cơ lớn > 4.0L (xe thể thao).

Việc tồn tại nhiều giá trị ngoại lai ở biến mục tiêu (Price) gợi ý rằng việc sử dụng các mô hình hồi quy tuyến tính cơ bản có thể bị ảnh hưởng bởi nhiễu. Do đó, cần

cân nhắc các kỹ thuật xử lý như loại bỏ ngoại lai cực đoan hoặc sử dụng biến đổi Logarit (Log Transformation) trước khi huấn luyện mô hình.

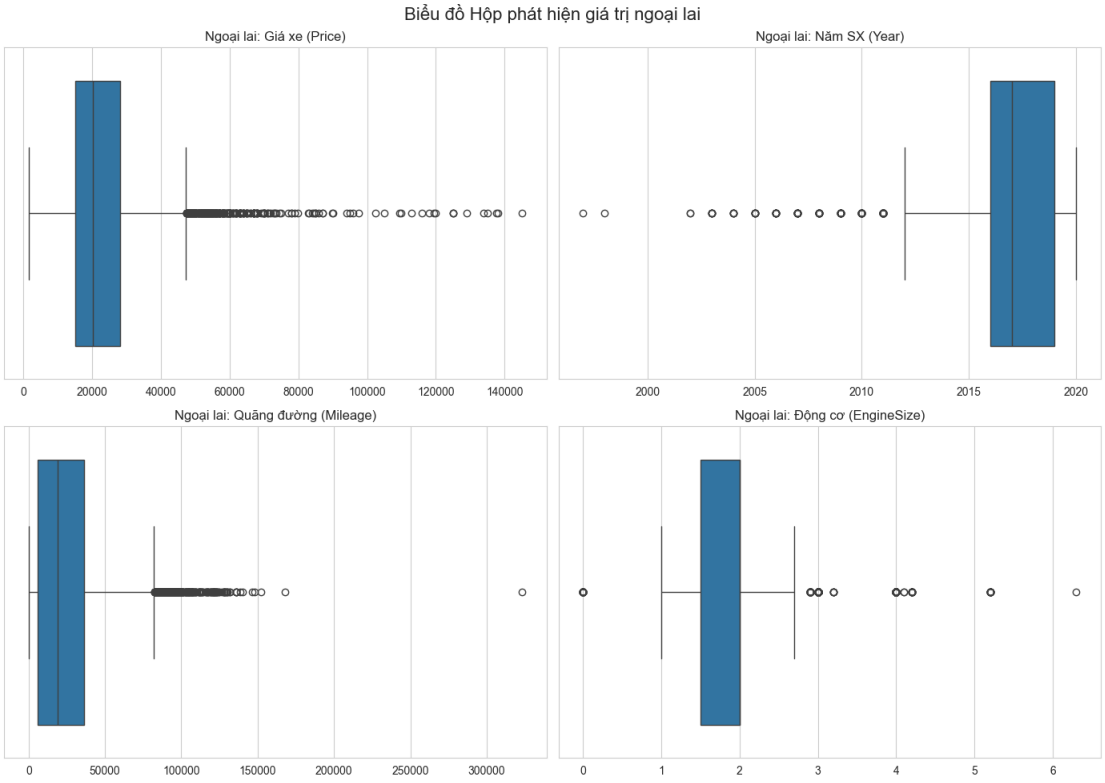

Figure 2: Biểu đồ hộp (Boxplot) phát hiện các giá trị ngoại lai.

2.3.4 Phân tích mối quan hệ giữa các biến (Bivariate Analysis)

Để đánh giá tác động của các yếu tố đầu vào đối với biến mục tiêu (Price), nhóm sử dụng biểu đồ phân tán (Scatter Plot) nhằm trực quan hóa xu hướng và độ mạnh của mối quan hệ.

a. Mối quan hệ giữa Giá xe và Năm sản xuất

Quan sát Hình 3, ta nhận thấy một mối tương quan dương mạnh mẽ giữa năm sản xuất và giá xe. Cụ thể:

- **Xu hướng tăng trưởng phi tuyến tính:** Giá xe không tăng đều theo từng năm mà có xu hướng tăng vọt theo hàm mũ ở các năm gần đây (2015 - 2020). Điều này phản ánh sự mất giá nhanh chóng của xe ô tô trong 3-5 năm đầu sử dụng.

- **Độ phân tán dữ liệu:** Ở các dòng xe đời cũ (trước 2010), mức giá dao động rất ít và tập trung ở ngưỡng thấp. Ngược lại, với các xe đời mới (2019-2020), biên độ giá dao động rất lớn (từ 20,000 đến hơn 140,000 EUR), cho thấy giá xe mới phụ thuộc nhiều vào mẫu xe (Model) và các tùy chọn đi kèm (Engine/Trim) hơn là chỉ dựa vào năm sản xuất.

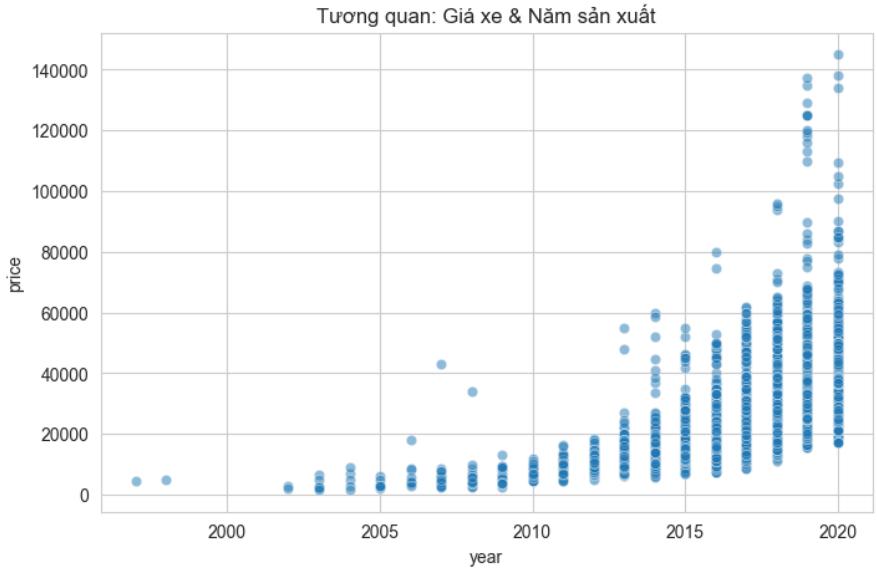

Figure 3: Biểu đồ phân tán: Giá xe theo Năm sản xuất. Xu hướng giá tăng mạnh ở các xe đời mới.

b. Mối quan hệ giữa Giá xe và Quãng đường đã đi

Hình 4 thể hiện mối tương quan âm rõ rệt giữa số km đã đi (Mileage) và giá bán:

- **Quy luật giảm giá trị:** Biểu đồ có dạng đường cong tiệm cận. Giá xe giảm rất nhanh trong khoảng quãng đường đầu tiên (0 - 50,000 dặm). Khi số km càng lớn ($>$ 100,000 dặm), mức giảm giá bắt đầu chững lại và đi ngang.
- **Mật độ tập trung:** Đa số các điểm dữ liệu tập trung ở góc trái dưới (mileage thấp, giá trung bình), phù hợp với phân phối lệch phải đã phân tích ở phần trước. Các xe có mileage cực lớn ($>$ 200,000 dặm) hầu hết đều có giá trị thấp và ít biến động.

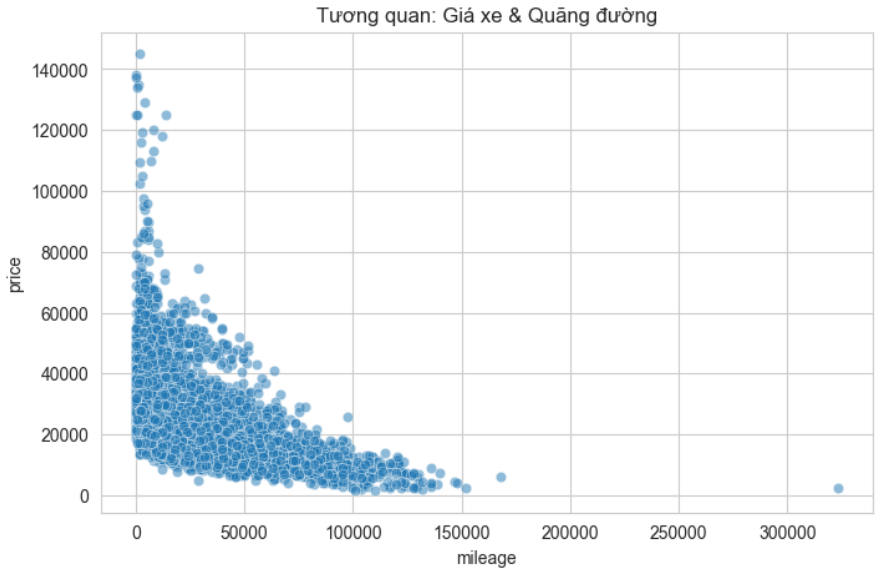

Figure 4: Biểu đồ phân tán: Giá xe theo Quãng đường. Xe đi càng nhiều, giá trị càng giảm theo đường cong.

Kết luận rút ra: Cả `Year` và `Mileage` đều là những chỉ báo (predictors) cực kỳ quan trọng cho mô hình dự đoán giá. Mối quan hệ phi tuyến tính (đường cong) gợi ý rằng các mô hình phi tuyến (như Random Forest, XGBoost) có thể sẽ hoạt động hiệu quả hơn mô hình hồi quy tuyến tính đơn thuần, hoặc cần thực hiện biến đổi đặc trưng (Feature Transformation) nếu sử dụng Linear Regression.

2.3.5 Ma trận tương quan

Tiền hành phân tích heatmap của các biến số định lượng, cho thấy:

- Năm sản xuất và giá (`year` $\sim$ `price`): tương quan dương vừa phải ($r \approx 0.59$), cho thấy xe mới thường có giá cao hơn.
- Quãng đường đã đi và giá (`mileage` $\sim$ `price`): tương quan âm vừa mạnh ($r \approx -0.54$), cho thấy xe đi nhiều km thường có giá thấp hơn.
- Mức tiêu thụ nhiên liệu và giá (`mpg` $\sim$ `price`): tương quan âm khá mạnh ($r \approx -0.60$), cho thấy xe tiết kiệm nhiên liệu thường giá thấp hơn.
- Thuế đường bộ và giá (`tax` $\sim$ `price`): tương quan dương yếu đến vừa ($r \approx 0.36$).

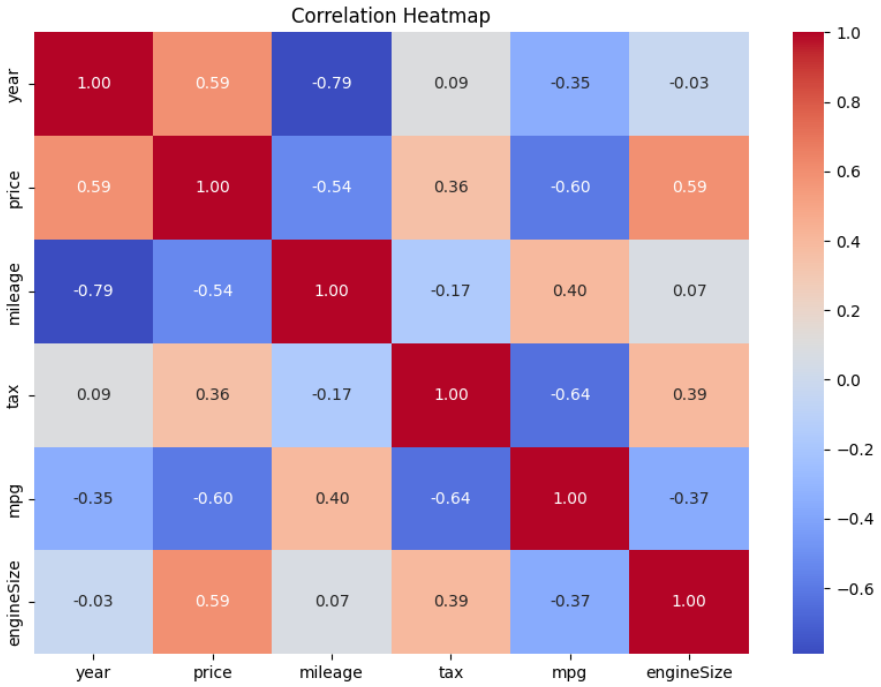

Figure 5: Ma trận tương quan giữa các biến số định lượng

2.4 Tiền xử lý và Kỹ thuật đặc trưng (Feature Engineering)

Sau khi hoàn tất quá trình khám phá dữ liệu, bước tiếp theo là chuyển đổi dữ liệu thô sang định dạng phù hợp để đưa vào các mô hình học máy. Hầu hết các thuật toán hồi quy (Regression) đều yêu cầu dữ liệu đầu vào dưới dạng số, do đó các biến phân loại (categorical variables) cần được mã hóa.

2.4.1 Mã hóa biến phân loại (One-Hot Encoding)

Trong bộ dữ liệu, có 3 biến định danh dạng chuỗi cần được xử lý:

- **model**: Mẫu xe (ví dụ: A1, A3, Q3...).
- **transmission**: Loại hộp số (Manual, Automatic, Semi-Auto).
- **fuelType**: Loại nhiên liệu (Petrol, Diesel, Hybrid).

Nhóm sử dụng phương pháp One-Hot Encoding để chuyển đổi các biến này. Phương pháp này tạo ra các cột nhị phân (0 hoặc 1) tương ứng với từng giá trị duy nhất của biến gốc.

Đặc biệt, tham số `drop_first=True` được sử dụng để loại bỏ cột đầu tiên của mỗi nhóm biến đã mã hóa. Điều này giúp giảm số lượng chiều dữ liệu không cần thiết và tránh hiện tượng đa cộng tuyến hoàn hảo (dummy variable trap) – tình trạng các biến độc lập có mối tương quan tuyến tính hoàn hảo với nhau, gây ảnh hưởng xấu đến mô hình hồi quy tuyến tính.

```
def preprocess_data(df):
    print(f"Số lượng feature ban đầu: {df.shape[1]}")

    # Thực hiện One-Hot Encoding và loại bỏ cột đầu tiên (drop_first=True)
    df_processed = pd.get_dummies(df, columns=categorical_cols,
                                                drop_first=True)

    print(f"Số lượng feature sau khi one-hot encoding:
                                                 {df_processed.shape[1]}")

    # Tách biến mục tiêu (Price) và biến đặc trưng (Features)
    X = df_processed.drop('price', axis=1)
    y = df_processed['price']

    feature_names = X.columns.tolist()
    return X, y, feature_names
```

Quá trình mã hóa đã làm thay đổi kích thước của bộ dữ liệu như sau:

- **Số lượng đặc trưng ban đầu:** 9 cột.
- **Số lượng đặc trưng sau mã hóa:** 35 cột.

Việc số lượng cột tăng lên (từ 9 lên 35) phản ánh sự đa dạng của các mẫu xe (model) và các tùy chọn cấu hình. Dữ liệu sau đó được tách biệt thành hai phần:

- **X (Features):** Chứa tất cả các thông tin kỹ thuật và thông số xe (đã được mã hóa).
- **y (Target):** Biến mục tiêu cần dự đoán là `price`.

Lúc này, dữ liệu X và y đã sẵn sàng cho việc chia tập huấn luyện/kiểm thử (Train/Test Split) và xây dựng mô hình.

3 CÁC MÔ HÌNH HỌC MÁY

3.1 Linear Regression

3.1.1 Thực nghiệm

Dữ liệu đầu vào bao gồm các đặc trưng cơ bản (như `mileage`, `engineSize`, ...). Tập dữ liệu được chia theo tỉ lệ 80% cho tập huấn luyện (train) và 20% cho tập kiểm thử (test) với `random_state=42` để đảm bảo tính tái lập.

Mô hình Linear Regression được khởi tạo và huấn luyện sử dụng thư viện `scikit-learn` với cấu hình mặc định (Ordinary Least Squares).

```
def run_linear_regression(filepath):
    print("\n=== Đang chạy thuật toán: Linear Regression (Baseline) ===")
    df = pd.read_csv(filepath)
    X = df.drop('price', axis=1)
    y = df['price']
    X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
                                                        random_state=42)

    preprocessor = ColumnTransformer(transformers=[
        ('num', StandardScaler(), ['year',
                                   'mileage',
                                   'tax',
                                   'mpg',
                                   'engineSize']),
        ('cat', OneHotEncoder(handle_unknown='ignore'), ['model',
                                                         'transmission',
                                                         'fuelType'])
    ])

    model = Pipeline(steps=[('preprocessor', preprocessor),
                            ('regressor', LinearRegression())])
    model.fit(X_train, y_train)
    return model, X_test, y_test
```

3.1.2 Kết quả

Từ các hệ số học được, phương trình ước lượng giá xe ($\hat{y}$) có dạng:

Kết quả đánh giá mô hình Hồi quy tuyến tính trên tập kiểm thử cho thấy hiệu năng còn hạn chế:

- **MAE:** 2633.07
- **RMSE:** 3963.66
- R^2**:** 0.8960

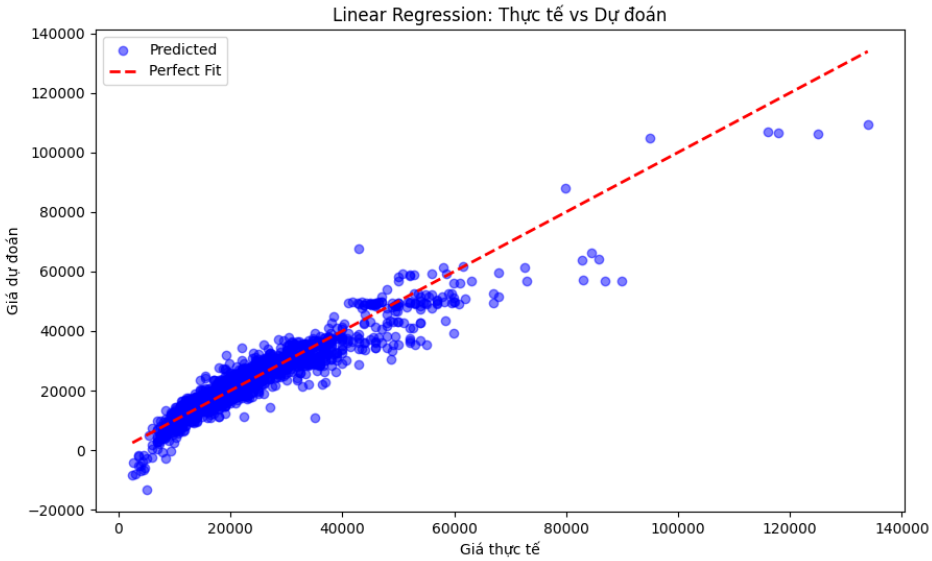

Figure 6: Biểu đồ thể hiện mối quan hệ giữa giá thực tế và giá dự đoán của LR

Giá trị R^2 tương đối cao (chỉ giải thích được khoảng 89% biến thiên của dữ liệu) và sai số RMSE thấp cho thấy mô hình tuyến tính khá ổn để nắm bắt các mối quan hệ phức tạp trong dữ liệu giá xe. Kết quả này đóng vai trò là mức cơ sở (baseline) để so sánh với các mô hình phi tuyến mạnh mẽ hơn như SVM hay Random Forest.

3.2 Random Forest

3.2.1 Thực nghiệm

Dữ liệu sau khi được tiền xử lý (One-Hot Encoding cho *model*, *transmission*, *fuelType*) đã tăng số lượng đặc trưng từ 9 lên 35. Tập dữ liệu được chia theo tỉ lệ 80% train và 20% test.

Mô hình Random Forest Regressor sau đây được huấn luyện với bộ tham số sau khi đã Tuning (Sử dụng **RandomizedSearchCV**)

```
def run_random_forest(filepath):
    print("\n=== Đang chạy thuật toán: Random Forest ===")
```

```python
# 1. Load và làm sạch cơ bản
df = pd.read_csv(filepath)
# Xử lý khoảng trắng
df.columns = df.columns.str.strip()
for col in df.select_dtypes(include='object').columns:
    df[col] = df[col].str.strip()

# 2. Chia dữ liệu
X = df.drop('price', axis=1)
y = df['price']
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
                                                    random_state=42)

# 3. Pipeline xử lý
categorical_cols = ['model', 'transmission', 'fuelType']
numerical_cols = ['year', 'mileage', 'tax', 'mpg', 'engineSize']

preprocessor = ColumnTransformer(transformers=[
    ('num', SimpleImputer(strategy='median'), numerical_cols),
    ('cat', OneHotEncoder(handle_unknown='ignore'), categorical_cols)
])

# 4. Định nghĩa mô hình và tham số
rf = RandomForestRegressor(random_state=42, n_jobs=-1)

param_dist = {
    'regressor__n_estimators': randint(100, 300),
    'regressor__max_depth': [10, 20, 30, None],
    'regressor__min_samples_split': [2, 5, 10],
    'regressor__min_samples_leaf': [1, 2, 4],
    'regressor__max_features': ['sqrt', 'log2']
}

pipeline = Pipeline(steps=[('preprocessor', preprocessor),
                           ('regressor', rf)])

# 5. Tuning (Dùng RandomizedSearchCV)
print("Đang tối ưu tham số (Tuning)...")
search = RandomizedSearchCV(
    pipeline, param_distributions=param_dist, n_iter=5, cv=3,
    random_state=42, n_jobs=-1, verbose=1
)
search.fit(X_train, y_train)
```

```
print(f"Best params: {search.best_params_}")
return search.best_estimator_, X_test, y_test
```

3.2.2 Kết quả

Mô hình Random Forest sau Tuning đã đạt hiệu năng cao trên tập kiểm thử:

- **MAE:** 1577.94
- **RMSE:** 2417.25
- R^2**:** 0.9654

Giá trị RMSE thấp cho thấy sai số dự đoán trung bình của mô hình tương đối nhỏ. Chỉ số R^2 cao thể hiện mô hình giải thích phần lớn biến thiên của giá xe.

Biểu đồ *Actual vs Predicted* cũng cho thấy các điểm dữ liệu bám sát đường chéo, chứng tỏ mô hình hoạt động ổn định và ít bị lệch. Đây là một mô hình mạnh, phù hợp để tiếp tục tinh chỉnh nhằm cải thiện thêm hiệu suất.

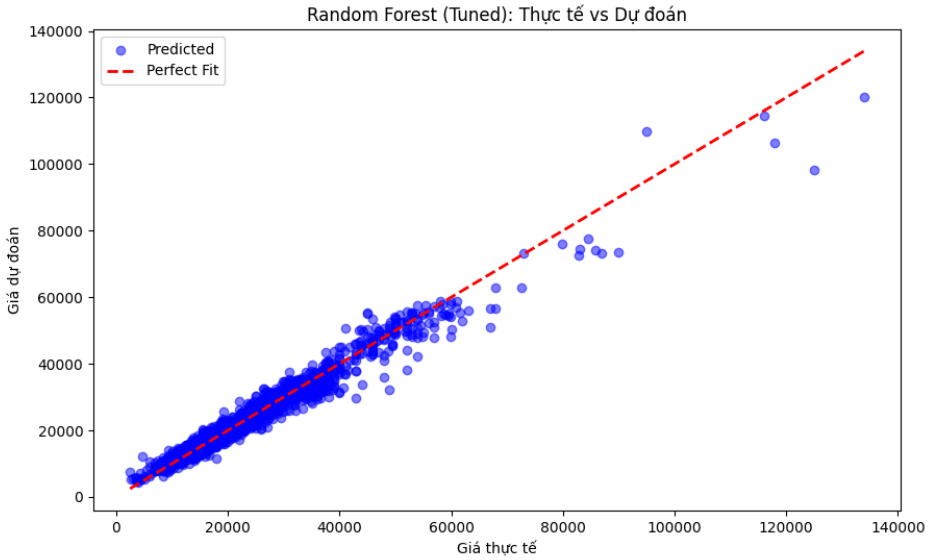

Figure 7: Biểu đồ thể hiện mối quan hệ giữa giá thực tế và giá dự đoán của RF

3.2.3 Phân tích độ quan trọng đặc trưng

Random Forest cho phép trích xuất mức độ đóng góp của từng đặc trưng vào quá trình dự đoán. Nhóm trình bày Top 15 đặc trưng quan trọng nhất nhằm hiểu rõ mô hình ra quyết định. Kết quả cho thấy các biến như *year*, *mileage*, *engineSize*

và một số biến *model* đặc thù đóng vai trò quan trọng nhất trong việc xác định giá xe.

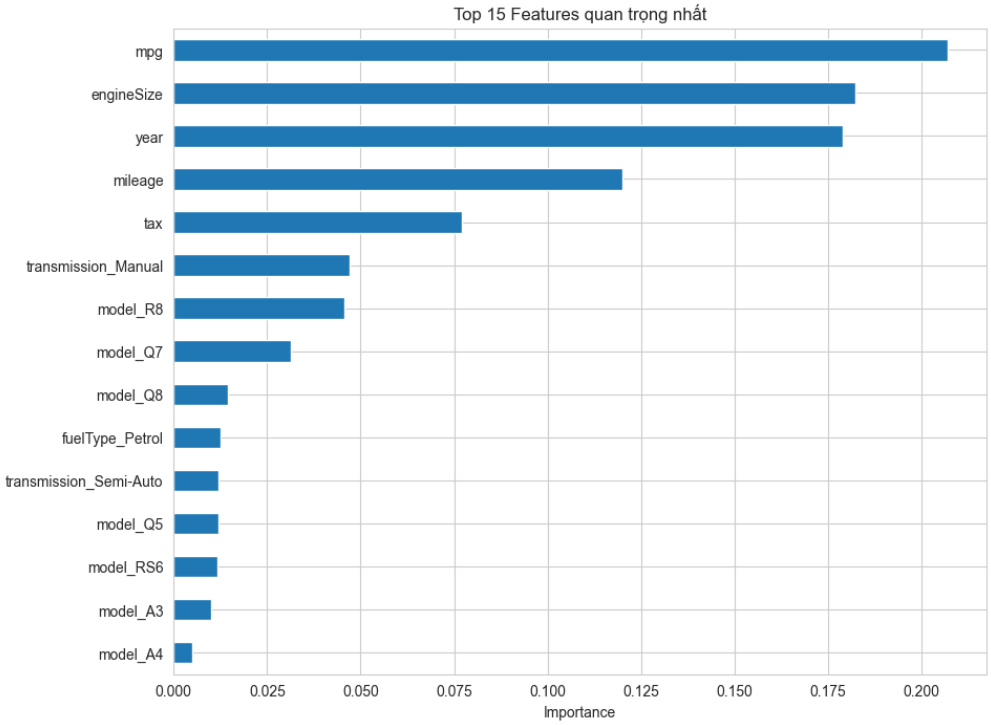

Figure 8: Top 15 đặc trưng quan trọng nhất theo Random Forest

3.3 Support Vector Machine (SVM)

3.3.1 Thực nghiệm

Dữ liệu đầu vào được giữ nguyên quy trình tiền xử lý như các mô hình khác trong nhóm: làm sạch cơ bản, xử lý thiếu dữ liệu, chia train–test theo tỉ lệ 80% train và 20% test.

Đối với mô hình SVM, việc chuẩn hóa dữ liệu là **bắt buộc** vì SVM rất nhạy với độ lớn của từng đặc trưng. Nhóm áp dụng bộ tiền xử lý bao gồm:

- **Numerical**: *median imputation* + *StandardScaler* + *SelectKBest* (dùng hàm `f_regression` để chọn đặc trưng tốt nhất).
- **Categorical**: One-Hot Encoding (không dùng dạng sparse để phù hợp với SVR).

Sau đó, mô hình được huấn luyện với pipeline chứa **SVR(kernel='rbf')** kết hợp

bộ tham số được tối ưu bằng **GridSearchCV**. Đoạn code thực nghiệm bên dưới được sử dụng trong phần chạy chính:

```
def run_svm(filepath):
    print("\n=== Đang chạy thuật toán: Support Vector Machine (SVM) ===")
    df = pd.read_csv(filepath)

    # 1. Chia dữ liệu
    X = df.drop('price', axis=1)
    y = df['price']
    X_train, X_test, y_train, y_test = train_test_split(
        X, y, test_size=0.2, random_state=42
    )

    # 2. Pipeline tiền xử lý
    numeric_features = ['year', 'mileage', 'tax', 'mpg', 'engineSize']
    categorical_features = ['model', 'transmission', 'fuelType']

    numeric_transformer = Pipeline(steps=[
        ('imputer', SimpleImputer(strategy='median')),
        ('scaler', StandardScaler()),
        ('selector', SelectKBest(score_func=f_regression, k='all'))
    ])

    categorical_transformer = OneHotEncoder(
        handle_unknown='ignore', sparse_output=False
    )

    preprocessor = ColumnTransformer(
        transformers=[
            ('num', numeric_transformer, numeric_features),
            ('cat', categorical_transformer, categorical_features)
        ]
    )

    # 3. Định nghĩa Pipeline SVM
    pipeline = Pipeline(steps=[
        ('preprocessor', preprocessor),
        ('regressor', SVR(kernel='rbf'))
    ])

    # 4. Tuning tham số bằng GridSearchCV
    param_grid = {
        'regressor__C': [100, 500],
```

```
    'regressor__epsilon': [0.1, 0.2],
    'regressor__gamma': ['scale']
}

print("Đang tối ưu tham số SVM...")
search = GridSearchCV(
    pipeline, param_grid, cv=3, n_jobs=-1, verbose=1
)

search.fit(X_train, y_train)
return search.best_estimator_, X_test, y_test
```

3.3.2 Kết quả

Sau quá trình tuning, mô hình SVM đạt hiệu năng tốt trên tập kiểm thử. Mặc dù SVM thường hoạt động mạnh hơn trong các bài toán phân loại, với hàm kernel RBF phù hợp và chuẩn hóa dữ liệu cẩn thận, mô hình vẫn cho kết quả dự đoán giá xe khả quan.

- **MAE:** 2320.44
- **RMSE:** 5538.08
- R^2**:** 0.7971

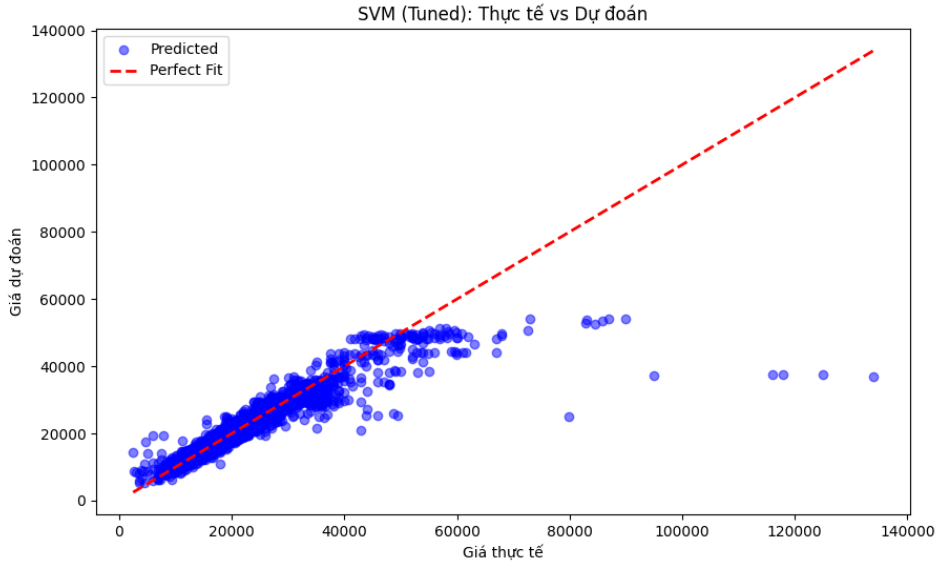

Figure 9: Mối quan hệ giữa giá thực tế và giá dự đoán của mô hình SVM

SVM thể hiện khả năng học tốt các quan hệ phi tuyến trong dữ liệu, tuy nhiên thời gian huấn luyện lâu hơn và nhạy cảm với việc chuẩn hóa dữ liệu. Điểm yếu thường gặp của SVM là khó mở rộng khi số mẫu lớn (do chi phí tính toán cao) và khó giải thích so với các mô hình dạng cây.

3.3.3 Phân tích và nhận xét

Không giống như Random Forest có thể trực tiếp trích xuất độ quan trọng đặc trưng, SVM (đặc biệt với kernel RBF) là mô hình *black-box*. Do đó, nhóm sử dụng kết quả từ `SelectKBest` để đánh giá mức độ đóng góp của từng đặc trưng đầu vào trước khi đưa vào mô hình.

Kết quả cho thấy các đặc trưng liên quan đến:

- **year**
- **mileage**
- **engineSize**

đóng vai trò quan trọng nhất trong việc dự đoán giá xe.

Nhìn chung, SVM là mô hình mạnh, ổn định, đặc biệt khi dữ liệu có quan hệ phi tuyến — nhưng vẫn không vượt qua Random Forest trong bài toán dự đoán giá xe Audi của nhóm.

4 CÁC KỸ THUẬT MÔ HÌNH HÓA NÂNG CAO

Sau khi khảo sát các thuật toán cơ bản, nhóm tiến hành thử nghiệm các kỹ thuật nâng cao nhằm cải thiện độ chính xác và khả năng tổng quát hóa của hệ thống. Phần này trình bày hai phương pháp: Mạng Nơ-ron Đa tầng (Deep Learning) và Học kết hợp (Ensemble Learning).

4.1 Mạng Nơ-ron Đa tầng (Multi-layer Perceptron - MLP)

4.1.1 Kiến trúc mạng đề xuất

Nhóm sử dụng kiến trúc **Multi-layer Perceptron (MLP)** để nắm bắt các mối quan hệ phi tuyến phức tạp trong dữ liệu giá xe. Cấu trúc mạng được thiết kế theo hình tháp ngược để trích xuất đặc trưng từ thô đến tinh:

- **Input Layer:** Tương ứng số chiều đặc trưng sau One-hot encoding.
- **Hidden Layers:** 2 lớp ẩn (100 nơ-ron và 50 nơ-ron).
- **Activation:** Sử dụng hàm **ReLU** $f(x) = \max(0, x)$ để tăng tốc độ hội tụ và tránh triệt tiêu đạo hàm.
- **Output Layer:** 1 nơ-ron (dự đoán giá trị thực).
- **Optimizer:** Adam ($\alpha = 0.001$).

4.1.2 Kết quả thực nghiệm MLP

Mô hình được huấn luyện trên thư viện `scikit-learn` với dữ liệu đã qua chuẩn hóa (`StandardScaler`).

```
# Cau hinh va huan luyen MLP
from sklearn.neural_network import MLPRegressor
from sklearn.pipeline import Pipeline

mlp_model = Pipeline(steps=[
    ('preprocessor', preprocessor),
    ('regressor', MLPRegressor(hidden_layer_sizes=(100, 50),
                               activation='relu', solver='adam',
                               max_iter=500, early_stopping=True,
                               n_iter_no_change=10, random_state=42))
```

```
])
mlp_model.fit(X_train, y_train)
```

Kết quả trên tập kiểm thử đạt **R2 Score: 0.9414** và **RMSE: 2975.71**. Biểu đồ dưới đây thể hiện quá trình hội tụ của hàm mất mát:

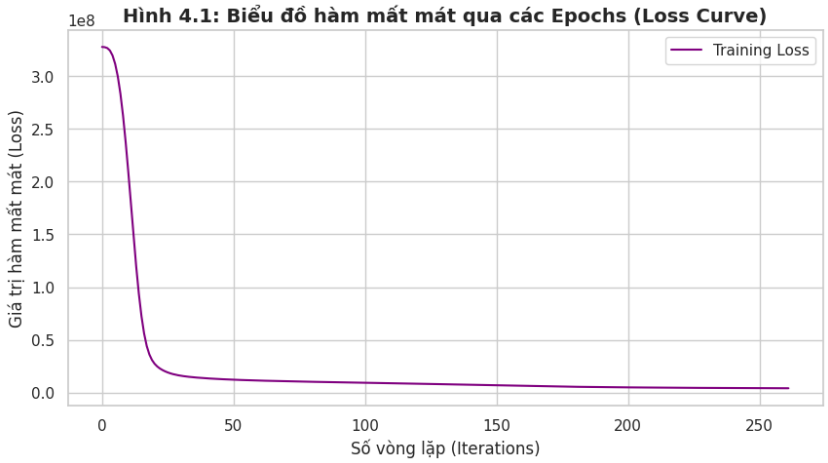

Figure 10: Biểu đồ Loss Curve của MLP: Hội tụ ổn định sau 100 epochs

4.2 Học kết hợp (Ensemble Learning - Voting Regressor)

4.2.1 Phương pháp tiếp cận

Để giảm thiểu sai số phương sai (Variance) và tận dụng ưu điểm của từng mô hình đơn lẻ, nhóm áp dụng kỹ thuật **Voting Regressor**. Kết quả dự đoán cuối cùng là trung bình cộng từ 4 mô hình thành phần ("xem nhu các chuyên gia"):

1. **Linear Regression:** Nắm bắt xu hướng tuyến tính tổng quát.
2. **SVR:** Tối ưu hóa biên quyết định trong không gian cao chiều.
3. **Random Forest:** Xử lý tốt dữ liệu phi tuyến và tương tác biến.
4. **MLP:** Học các đặc trưng ẩn sâu.

Công thức dự đoán tổng hợp: $\hat{y}_{final} = \frac{1}{N} \sum_{i=1}^{N} \hat{y}_i(x)$

4.2.2 Kết quả thực nghiệm Voting

```
from sklearn.ensemble import VotingRegressor
```

```
# Khoi tao cac estimators
estimators = [
    ('lr', LinearRegression()),
    ('svr', SVR(C=1000, kernel='rbf', gamma=0.1)),
    ('rf', RandomForestRegressor(n_estimators=100, random_state=42)),
    ('mlp', MLPRegressor(hidden_layer_sizes=(100, 50), random_state=42))
]
# Voting
voting_model = Pipeline(steps=[('pre', preprocessor),
                               ('vote', VotingRegressor(estimators))])
voting_model.fit(X_train, y_train)
```

Mô hình Voting đạt **R2 Score: 0.9398**, cho thấy độ ổn định cao và khả năng khái quát hóa tốt hơn so với việc phụ thuộc vào một mô hình duy nhất.

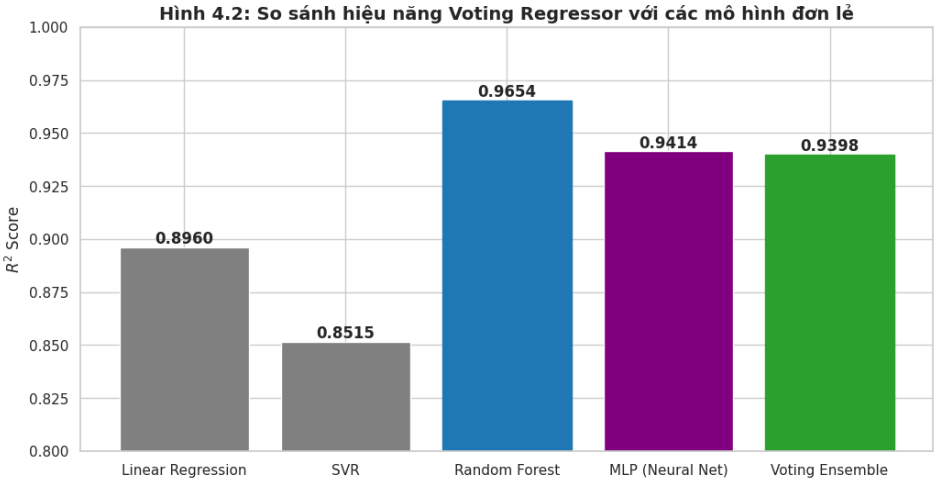

Figure 11: So sánh hiệu năng giữa Voting Regressor và các mô hình thành phần

5 SO SÁNH, ĐÁNH GIÁ VÀ KẾT LUẬN

5.1 Thiết lập kịch bản thực nghiệm

Để đảm bảo tính công bằng (fairness) và khả năng tái lập (reproducibility) của kết quả nghiên cứu, toàn bộ các mô hình từ cơ bản đến nâng cao đều được đánh giá trên cùng một kịch bản thực nghiệm chuẩn hóa:

- **Tập dữ liệu kiểm thử (Test Set):** Chiếm 20% tổng số dữ liệu (khoảng 2,134 mẫu), hoàn toàn độc lập và chưa từng được các mô hình "nhìn thấy" trong quá trình huấn luyện.
- **Tiền xử lý:** Sử dụng chung một pipeline (One-hot Encoding cho biến phân loại, StandardScaler cho biến số thực) để đảm bảo đầu vào đồng nhất.
- **Tham số ngẫu nhiên (Random State):** Cố định giá trị `random_state=42` ở tất cả các bước chia dữ liệu và khởi tạo mô hình.

Các chỉ số đánh giá (Evaluation Metrics) được sử dụng bao gồm:

- **R-squared (R^2):** Đo lường mức độ giải thích phương sai của mô hình (càng gần 1 càng tốt).
- **RMSE (Root Mean Squared Error):** Căn bậc hai của sai số bình phương trung bình, cùng đơn vị với biến mục tiêu (giá xe).
- **MAPE (Mean Absolute Percentage Error):** Sai số tuyệt đối trung bình theo phần trăm, giúp đánh giá sai số tương đối dễ hiểu hơn.

5.2 Kết quả thực nghiệm định lượng

Sau quá trình huấn luyện và tinh chỉnh, kết quả tổng hợp của 5 mô hình được trình bày trong Bảng 4 dưới đây.

Table 4: Bảng tổng hợp hiệu năng các mô hình trên tập kiểm thử

Mô hình	R2 Score	RMSE ($)	MAE ($)	MAPE (%)
Random Forest	**0.9654**	**2285.93**	**1521.84**	**7.07%**
Voting Ensemble	0.9487	3016.17	1864.50	8.14%
MLP (Deep Learning)	0.9414	2975.71	2021.12	8.89%
Linear Regression	0.8960	3963.67	2633.07	13.32%
SVM (RBF Kernel)	0.8515	4736.74	2129.82	8.73%

Nhận xét số liệu:

- **Random Forest** là mô hình vượt trội nhất ở mọi chỉ số, đạt R^2 lên tới 96.5% và sai số trung bình (MAPE) chỉ khoảng 7%.
- **Voting Regressor** và **MLP** bám đuổi rất sát nhau ở vị trí thứ 2 và 3. Điều này cho thấy việc kết hợp nhiều mô hình hoặc sử dụng mạng nơ-ron sâu đều mang lại cải thiện đáng kể so với các phương pháp truyền thống.
- **Linear Regression** và **SVM** có hiệu năng thấp nhất, đặc biệt là RMSE của SVM rất cao (4736$), cho thấy mô hình này mắc phải một số sai số rất lớn (outliers) khi dự đoán các dòng xe giá trị cao.

5.3 So sánh và Phân tích trực quan

Để có cái nhìn sâu sắc hơn về hành vi của từng mô hình, nhóm thực hiện các phân tích trực quan sau:

5.3.1 So sánh độ chính xác tổng thể (R^2 Score)

Biểu đồ thanh dưới đây so sánh trực quan điểm số R^2 của các mô hình.

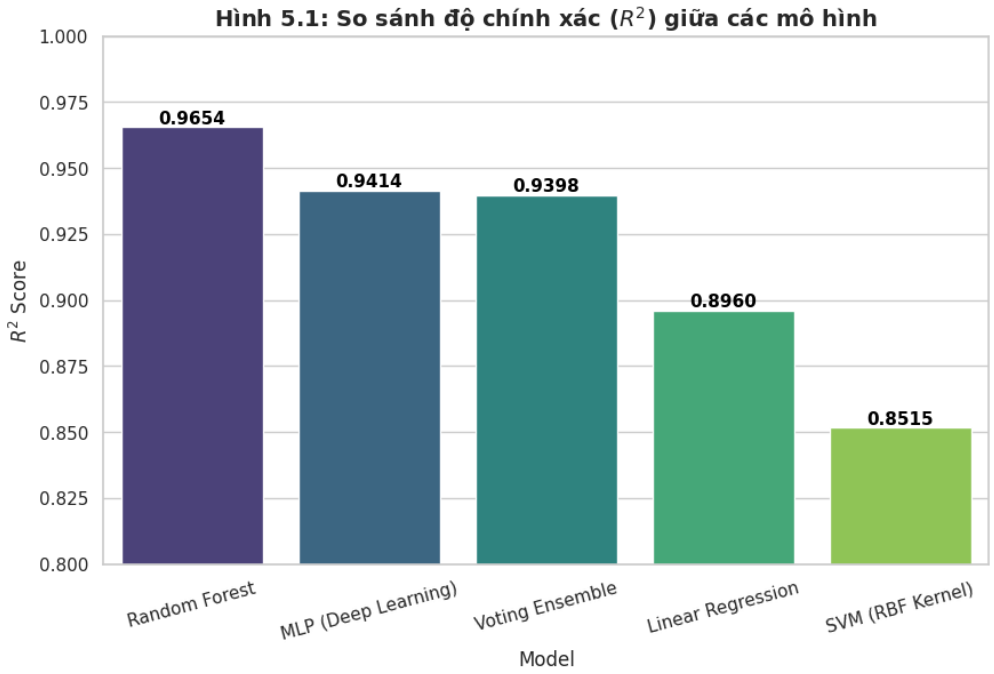

Figure 12: So sánh mức độ giải thích dữ liệu (R^2) của các mô hình

Có thể thấy rõ sự phân cấp: Nhóm mô hình phi tuyến (Random Forest, Voting,

MLP) vượt trội hoàn toàn so với nhóm mô hình tuyến tính hoặc dựa trên khoảng cách đơn thuần (Linear, SVM).

5.3.2 Độ khớp giữa Giá trị Thực tế và Dự đoán

Biểu đồ phân tán (Scatter Plot) giúp quan sát mức độ bám sát của các điểm dữ liệu dự đoán so với đường chéo lý tưởng $y = x$ (Perfect Fit).

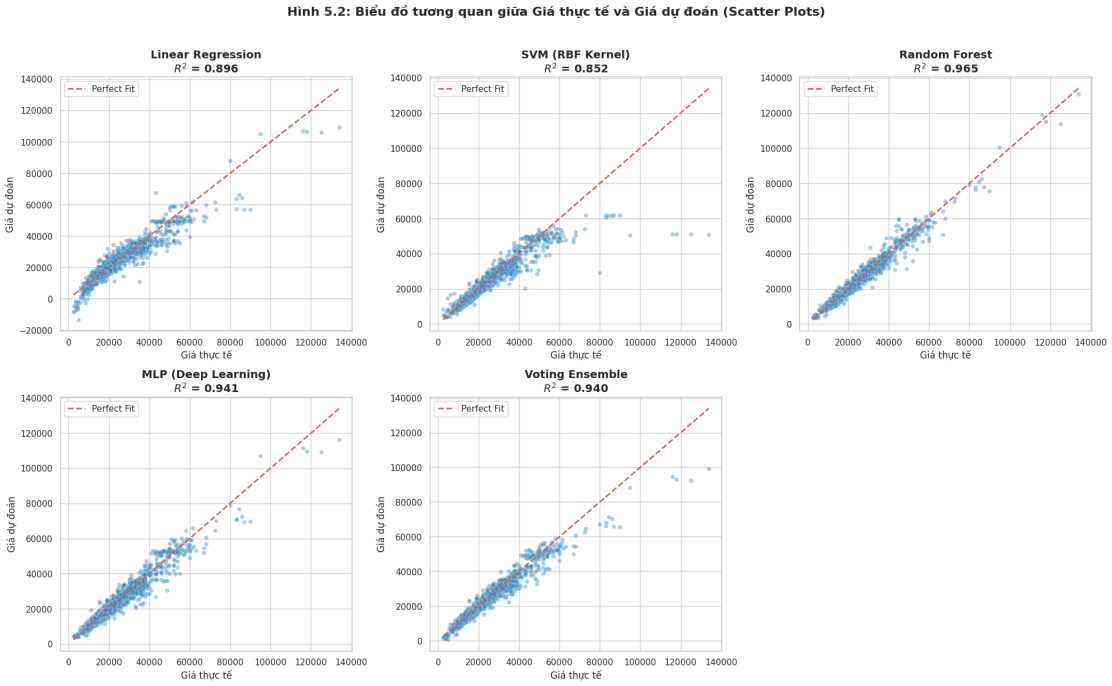

Figure 13: Biểu đồ phân tán Giá trị Thực tế vs. Giá trị Dự đoán

Phân tích:

- **Random Forest:** Các điểm dữ liệu tập trung rất dày đặc quanh đường đỏ, ít bị phân tán, chứng tỏ độ chính xác cao trên toàn bộ dải giá trị.
- **Linear Regression:** Có xu hướng dự đoán sai lệch nhiều ở phân khúc giá cao (các điểm bị tản mát xa đường đỏ ở góc trên bên phải), do hạn chế của đường thẳng hồi quy không thể uốn cong theo dữ liệu.
- **SVM:** Thể hiện rõ sự "đuối sức" khi gặp các mẫu dữ liệu nhiễu hoặc giá trị ngoại lai.

5.3.3 Phân tích sự ổn định và Sai số (Residual Analysis)

Biểu đồ hộp (Boxplot) của sai số tuyệt đối giúp phát hiện các mô hình có độ ổn định kém (nhiều outliers).

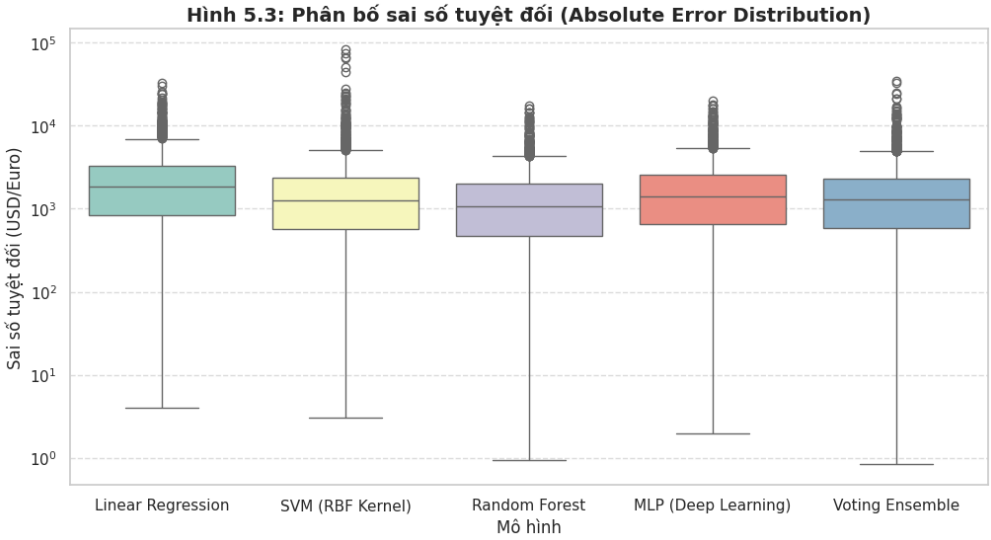

Figure 14: Phân bố sai số tuyệt đối của các mô hình (Thang đo Log)

Hộp của **Random Forest** nằm ở vị trí thấp nhất trên trục tung, đồng nghĩa với việc sai số trung vị (median error) của nó là nhỏ nhất. Ngược lại, Linear Regression có "đuôi" sai số kéo dài lên phía trên, cho thấy rủi ro dự đoán sai lệch lớn là cao nhất.

5.4 Thảo luận và Kết luận

5.4.1 Thảo luận về sự đánh đổi

1. **Độ chính xác vs. Tốc độ:**
   - *Linear Regression* có tốc độ huấn luyện và dự đoán cực nhanh (gần như tức thời), nhưng độ chính xác chỉ ở mức trung bình.
   - *Random Forest* và *Voting* tốn nhiều thời gian huấn luyện hơn (do phải xây dựng nhiều cây hoặc train nhiều model con), nhưng đổi lại là độ chính xác vượt trội.
2. **Khả năng giải thích (Interpretability):**

- *Random Forest* cung cấp tính năng "Feature Importance", giúp ta hiểu được đặc trưng nào (Năm sản xuất, Động cơ...) quan trọng nhất.
- *MLP (Neural Network)* và *SVM* thường được coi là "hộp đen" (black-box), khó giải thích lý do tại sao mô hình lại đưa ra mức giá đó.

5.4.2 Kết luận và Hướng phát triển

Dựa trên toàn bộ quá trình nghiên cứu, nhóm rút ra các kết luận sau:

- **Mô hình tối ưu: Random Forest Regressor** là lựa chọn tốt nhất cho bài toán định giá xe Audi cũ này, nhờ khả năng xử lý tốt dữ liệu dạng bảng, nắm bắt các mối quan hệ phi tuyến và ít bị ảnh hưởng bởi nhiễu.
- **Giá trị thực tiễn:** Với sai số trung bình (MAPE) khoảng **7%**, mô hình hoàn toàn có thể ứng dụng để xây dựng công cụ tham khảo giá cho người mua/bán xe hoặc các đại lý ô tô cũ.

Đề xuất hướng phát triển trong tương lai:

1. Mở rộng tập dữ liệu sang các hãng xe khác (BMW, Mercedes, Toyota...) để tăng tính tổng quát hóa.
2. Thử nghiệm các thuật toán Boosting hiện đại hơn như **XGBoost, LightGBM, CatBoost** để so sánh với Random Forest.
3. Áp dụng kỹ thuật **XAI (Explainable AI)** như thư viện SHAP để giải thích chi tiết từng dự đoán của mô hình Black-box như MLP.
4. Xây dựng ứng dụng Web (sử dụng Streamlit hoặc Flask) để triển khai mô hình phục vụ người dùng cuối.

6 TÀI LIỆU THAM KHẢO

Tài liệu tham khảo

[1] G. James, D. Witten, T. Hastie, and R. Tibshirani, *An Introduction to Statistical Learning: with Applications in R*. New York: Springer, 2013.

[2] C. M. Bishop, *Pattern Recognition and Machine Learning*. Springer, 2006.

[3] Scikit-learn Developers, "Linear Models - Scikit-learn documentation," [Online]. Available: https://scikit-learn.org/stable/modules/linear_model.html.

[4] A. Ng, "Machine Learning Course Materials," Stanford University, Coursera, 2012.

[5] Scikit-learn documentation.

[6] Sources Code, Github. [Online]. Available: https://github.com/ntthang-dev/CO3117-ML-Project.