

ĐẠI HỌC BÁCH KHOA HÀ NỘI  
KHOA TOÁN - TIN



BÁO CÁO BÀI TẬP LỚN  
HỆ HỖ TRỢ QUYẾT ĐỊNH (MI4216)

Đề tài:

DỰ ĐOÁN GIÁ XE Ô TÔ ĐÃ QUA SỬ DỤNG  
BẰNG MÔ HÌNH HỒI QUY

Mã lớp học: 163653 (2025.1)

Sinh viên thực hiện: Nguyễn Trung Kiên

Mã số sinh viên: 20227180

Giảng viên hướng dẫn: TS. Trần Ngọc Thăng

Hà Nội, tháng 01 năm 2026



# Lời nói đầu

---

Trong kỷ nguyên của Dữ liệu lớn (Big Data) và Chuyển đổi số, việc ra quyết định dựa trên dữ liệu (Data-driven decision making) đang trở thành yếu tố then chốt tạo nên lợi thế cạnh tranh cho các doanh nghiệp và tổ chức. Đặc biệt, tại thị trường ô tô đã qua sử dụng – nơi thông tin thường xuyên bị bất cân xứng và chịu ảnh hưởng bởi nhiều yếu tố định tính lẫn định lượng – việc xác định một mức giá ”đúng” và công bằng là bài toán đầy thách thức.

Báo cáo bài tập lớn học phần **Hệ hỗ trợ quyết định (MI4216)** với đề tài **”Đự đoán giá xe ô tô đã qua sử dụng bằng mô hình hồi quy”** được thực hiện nhằm giải quyết vấn đề trên. Báo cáo không chỉ dừng lại ở việc áp dụng các thuật toán máy học (Machine Learning) để dự báo giá, mà còn hướng tới việc xây dựng một quy trình khép kín: từ khai phá dữ liệu, phân tích đặc trưng, so sánh hiệu năng của các mô hình tiên tiến (như CatBoost, XGBoost) đến việc đóng gói thành ứng dụng hỗ trợ người dùng cuối.

Thông qua đề tài này, em mong muốn vận dụng những kiến thức lý thuyết đã được học vào thực tiễn, đồng thời rèn luyện kỹ năng tư duy phân tích hệ thống và xử lý dữ liệu quy mô lớn. Đây là bước đệm quan trọng để em tiếp cận gần hơn với các bài toán thực tế trong lĩnh vực Khoa học dữ liệu và Hệ thống thông tin.

Em xin gửi lời cảm ơn chân thành đến các thầy cô Khoa Toán - Tin, Đại học Bách Khoa Hà Nội đã luôn tận tình giảng dạy. Đặc biệt, em xin bày tỏ lòng biết ơn sâu sắc tới TS. Trần Ngọc Thăng, người thầy đã trực tiếp hướng dẫn và định hướng tư duy giúp em hoàn thành báo cáo này.

Mặc dù đã nỗ lực hết mình, song do giới hạn về thời gian và kiến thức, báo cáo khó tránh khỏi những thiếu sót. Em rất mong nhận được những ý kiến đóng góp quý báu từ Thầy để hoàn thiện hơn nữa năng lực chuyên môn của mình.

Em xin chân thành cảm ơn!

**Nguyễn Trung Kiên**  
Email: kien.nt227180@sis.hust.edu.vn  
Hà Nội, tháng 1 năm 2026

# Mục lục

---

<b>Lời nói đầu</b>	<b>i</b>
--------------------	----------

<b>1 Tổng quan về bài toán</b>	<b>1</b>
1.1 Bối cảnh thị trường và Tính cấp thiết của đề tài . . . . .	1
1.1.1 Sự dịch chuyển xu hướng tiêu dùng . . . . .	1
1.1.2 Thực trạng và Vấn đề (Pain Points) . . . . .	2
1.2 Phát biểu bài toán . . . . .	3
1.2.1 Bài toán Nghiệp vụ (Business Problem) . . . . .	3
1.2.2 Bài toán Kỹ thuật (Technical Problem) . . . . .	3
1.3 Mô hình hóa bài toán (Input - Process - Output) . . . . .	4
1.3.1 Phân tích Vector đặc trưng đầu vào (Input Vectors) . . . . .	5
1.3.2 Mô hình xử lý (Process) và Đầu ra (Output) . . . . .	5
1.4 Phương pháp tiếp cận và Các thuật toán nghiên cứu . . . . .	6
1.4.1 Nhóm 1: Các mô hình Tuyến tính (Linear Models) . . . . .	6
1.4.2 Nhóm 2: Các mô hình Phi tuyến và Khoảng cách . . . . .	6
1.4.3 Nhóm 3: Các mô hình Cây quyết định (Tree-based Models) . . . . .	7
1.4.4 Nhóm 4: Các mô hình Tăng cường (Gradient Boosting) . . . . .	7
1.4.5 Mô hình đề xuất: Stacking Ensemble Regressor . . . . .	7
1.5 Quy trình thực hiện (CRISP-DM) . . . . .	8
<b>2 Khai phá dữ liệu</b>	<b>10</b>
2.1 Tổng quan về tập dữ liệu (Dataset Overview) . . . . .	10
2.1.1 Nguồn gốc và Phạm vi dữ liệu . . . . .	10
2.2 Phân tích khám phá dữ liệu (Exploratory Data Analysis) . . . . .	11
2.2.1 Thống kê mô tả (Descriptive Statistics) . . . . .	11
2.2.2 Phân tích tương quan (Correlation Analysis) . . . . .	13
2.3 Quy trình làm sạch và Biến đổi dữ liệu (Data Cleaning & Transformation) . . . . .	14
2.3.1 Xử lý dữ liệu nhiễu và Giá trị thiếu . . . . .	14
2.3.2 Chuẩn hóa phân phối biến mục tiêu (Target Variable Normalization) . . . . .	15
2.4 Trích chọn đặc trưng và Tiền xử lý (Feature Engineering) . . . . .	16
2.4.1 Tạo đặc trưng mới (Feature Extraction) . . . . .	16
2.4.2 Mã hóa biến định danh (Categorical Encoding) . . . . .	17

2.4.3 Chuẩn hóa dữ liệu (Feature Scaling) . . . . .	17
2.4.4 Kết quả tiền xử lý (Final Processed Data) . . . . .	17
2.5 Kết luận chương 2 . . . . .	18
<b>3 Thiết lập thực nghiệm</b>	<b>19</b>
3.1 Chiến lược phân chia dữ liệu (Data Splitting Strategy) . . . . .	19
3.1.1 Phân chia tập dữ liệu (Train/Validation/Test Split) . . . . .	19
3.1.2 Kỹ thuật Kiểm thử chéo (K-Fold Cross-Validation) . . . . .	20
3.2 Hệ thống chỉ số đánh giá mô hình (Evaluation Metrics) . . . . .	21
3.2.1 Các chỉ số đo lường sai số (Error Metrics) . . . . .	21
3.2.2 Hệ số xác định (Coefficient of Determination - $R^2$ ) . . . . .	22
3.2.3 Tiêu chí hiệu năng triển khai (Deployment Performance) . . . . .	23
3.3 Phương pháp luận về Phân tích lỗi (Error Analysis Methodology) . . . . .	23
3.3.1 Quy trình trích xuất lỗi (Error Extraction Process) . . . . .	23
3.3.2 Phân tích nguyên nhân (Root Cause Analysis) . . . . .	24
3.4 Kết luận chương 3 . . . . .	25
<b>4 Xây dựng và Đánh giá mô hình</b>	<b>26</b>
4.1 Chiến lược thực nghiệm . . . . .	26
4.1.1 Cơ sở lý luận . . . . .	26
4.1.2 Lộ trình thực nghiệm chi tiết . . . . .	26
4.2 Nhóm mô hình Tuyến tính (Linear Models) . . . . .	28
4.2.1 Linear Regression (Mô hình cơ sở) . . . . .	28
4.2.2 Ridge Regression (L2 Regularization) . . . . .	31
4.2.3 Lasso Regression (L1 Regularization) . . . . .	33
4.2.4 Đánh giá chung nhóm mô hình Tuyến tính . . . . .	35
4.3 Nhóm mô hình Phi tuyến và Khoảng cách . . . . .	35
4.3.1 K-Nearest Neighbors (KNN) . . . . .	35
4.3.2 Support Vector Regression (SVR) . . . . .	37
4.3.3 Đánh giá chung nhóm mô hình Phi tuyến . . . . .	39
4.4 Nhóm mô hình Cây quyết định (Tree-based Models) . . . . .	40
4.4.1 Decision Tree Regressor . . . . .	40
4.4.2 Random Forest Regressor . . . . .	42
4.4.3 Đánh giá chung nhóm mô hình Cây quyết định . . . . .	44
4.5 Nhóm mô hình Tăng cường (Gradient Boosting) . . . . .	45
4.5.1 Gradient Boosting Regressor (GBM) . . . . .	45
4.5.2 XGBoost (eXtreme Gradient Boosting) . . . . .	47

4.5.3	CatBoost (Categorical Boosting) . . . . .	49
4.6	Tổng kết thực nghiệm . . . . .	52
4.7	Kết luận và Lựa chọn mô hình . . . . .	54
<b>5</b>	<b>Đóng gói và Ứng dụng</b>	<b>55</b>
5.1	Cơ sở lựa chọn mô hình triển khai . . . . .	55
5.1.1	Thử nghiệm với công nghệ Deep Learning tiên tiến (TabNet - 2021)	55
5.1.2	So sánh đối đầu: CatBoost vs. TabNet . . . . .	55
5.1.3	Biện luận khoa học về sự ưu việt của CatBoost . . . . .	56
5.1.4	Quyết định lựa chọn mô hình cuối cùng . . . . .	56
5.1.5	Thử nghiệm kỹ thuật Học tập Chồng chất (Stacking Ensemble) . .	57
5.1.6	Biện luận khoa học và Quyết định cuối cùng . . . . .	58
5.2	Thiết kế kiến trúc hệ thống . . . . .	59
5.2.1	Mô hình kiến trúc tổng thể . . . . .	59
5.2.2	Luồng xử lý dữ liệu (Data Flow) . . . . .	61
5.2.3	Cấu trúc thư mục dự án . . . . .	63
5.3	Xây dựng dịch vụ Backend . . . . .	64
5.3.1	Công nghệ sử dụng: FastAPI và LangChain . . . . .	64
5.3.2	Thiết kế các API Endpoints . . . . .	66
5.4	Xây dựng giao diện người dùng (Frontend) . . . . .	70
5.4.1	Lựa chọn ngăn xếp công nghệ (Technology Stack) . . . . .	70
5.4.2	Thiết kế và Triển khai các chức năng chính . . . . .	72
5.5	Demo giao diện ứng dụng . . . . .	74
5.6	Kịch bản ứng dụng vào nghiệp vụ thực tế . . . . .	77
5.6.1	Case Study 1: Tối ưu hóa hoạt động cho Showroom xe cũ . . . .	77
5.6.2	Case Study 2: Hỗ trợ người mua xe cá nhân . . . . .	79
5.7	Kết luận chương 5 . . . . .	80
<b>6</b>	<b>Tổng kết và Đánh giá</b>	<b>81</b>
6.1	Tổng hợp kết quả đạt được . . . . .	81
6.1.1	Về mặt Kỹ thuật (Technical Achievements) . . . . .	81
6.1.2	Về mặt Ứng dụng (Application Achievements) . . . . .	81
6.2	Hạn chế của nghiên cứu . . . . .	82
6.3	Hướng phát triển trong tương lai . . . . .	82
6.4	Kết luận chung . . . . .	83
<b>Lời cảm ơn</b>		<b>84</b>

<b>Phụ lục</b>	<b>85</b>
.1 Mã nguồn và Tài nguyên tham khảo . . . . .	85
.2 Kho mã nguồn trên GitHub . . . . .	85
.3 Tài nguyên dự án trên Google Drive . . . . .	85

# Danh sách hình vẽ

---

1.1	Xu hướng tăng trưởng của thị trường xe đã qua sử dụng . . . . .	1
1.2	Các rủi ro chính trong giao dịch xe cũ: Thông tin bất cân xứng và Định giá cảm tính . . . . .	2
1.3	Sơ đồ mô hình hóa bài toán Input - Process - Output . . . . .	6
1.4	Kiến trúc Stacking Ensemble đề xuất . . . . .	8
1.5	Quy trình CRISP-DM áp dụng trong nghiên cứu . . . . .	9
2.1	Bảng thống kê mô tả các biến định lượng . . . . .	12
2.2	Thống kê các biến phân loại chính . . . . .	12
2.3	Ma trận tương quan giữa các biến (Heatmap) . . . . .	13
2.4	So sánh phân phối Giá xe: Trước (Trái) và Sau khi biến đổi Log (Phải) . .	16
2.5	Mẫu dữ liệu sau khi đã tiền xử lý và chuẩn hóa . . . . .	17
3.1	Tỷ lệ phân chia dữ liệu cho thực nghiệm (70/15/15) . . . . .	20
3.2	Mô hình minh họa kỹ thuật 5-Fold Cross-Validation . . . . .	21
4.1	Biểu đồ Scatter (Thực tế vs Dự đoán) của Linear Regression . . . . .	29
4.2	Top 5 trường hợp sai lệch lớn nhất . . . . .	30
4.3	Kết quả dự đoán và Tầm quan trọng của đặc trưng (Ridge Regression) . .	32
4.4	Kết quả dự đoán và Đặc trưng quan trọng (Lasso Regression) . . . . .	34
4.5	Biểu đồ Scatter (Thực tế vs Dự đoán) của mô hình KNN . . . . .	37
4.6	Biểu đồ Scatter của SVR: Mô hình không bắt được xu hướng dữ liệu . .	39
4.7	Biểu đồ Scatter và Mức độ quan trọng của đặc trưng (Decision Tree) . .	42
4.8	Kết quả dự báo và Mức độ quan trọng đặc trưng (Random Forest) . . .	44
4.9	Kết quả dự báo và Mức độ quan trọng đặc trưng (GBM) . . . . .	47
4.10	Biểu đồ Scatter và Mức độ đóng góp thông tin (XGBoost) . . . . .	49
4.11	Biểu đồ trực quan hóa giá thực tế và giá dự đoán (CatBoost) . . . . .	51
4.12	Phân tích tác động của đặc trưng bằng SHAP Values . . . . .	52
5.1	Sơ đồ kiến trúc 3 tầng của hệ thống . . . . .	59
5.2	Sơ đồ luồng xử lý dữ liệu cho chức năng Định giá đơn lẻ . . . . .	61
5.3	Sơ đồ cấu trúc thư mục dự án . . . . .	63
5.4	Mockup giao diện chức năng Định giá đơn lẻ . . . . .	72

5.5	Mockup giao diện Trợ lý AI . . . . .	73
5.6	Quy trình Định giá đơn lẻ . . . . .	74
5.7	Các tính năng tùy biến giao diện . . . . .	74
5.8	Quy trình Định giá Hàng loạt . . . . .	75
5.9	Chức năng Trợ lý AI và Dashboard phân tích . . . . .	76
5.10	Trang Lịch sử hiển thị toàn bộ dữ liệu định giá của hệ thống . . . . .	77
5.11	Quy trình 3 bước xử lý lô xe cho showroom . . . . .	78
5.12	Trợ lý AI tư vấn và so sánh giá cho người dùng cá nhân . . . . .	79

# Danh sách bảng

---

2.1	Mô tả các thuộc tính trong tập dữ liệu . . . . .	11
3.1	Phân tích nguyên nhân các trường hợp có sai số dự báo lớn . . . . .	25
4.1	Kết quả đánh giá mô hình Linear Regression . . . . .	29
4.2	So sánh hiệu năng Linear Regression (M1) và Ridge Regression (M2) . .	32
4.3	So sánh hiệu năng 3 mô hình Tuyến tính . . . . .	34
4.4	So sánh hiệu năng Linear Regression (M1) và KNN (M4) . . . . .	36
4.5	Kết quả đánh giá mô hình SVR (trên tập mẫu 20k) . . . . .	38
4.6	So sánh hiệu năng KNN (M4) và Decision Tree (M6) . . . . .	41
4.7	So sánh hiệu năng Decision Tree (M6) và Random Forest (M7) . . . .	43
4.8	So sánh hiệu năng Random Forest (M7) và Gradient Boosting (M8) . .	46
4.9	So sánh hiệu năng Gradient Boosting (M8) và XGBoost (M9) . . . .	48
4.10	Kết quả hiệu năng của CatBoost (Best Model) . . . . .	51
4.11	Bảng tổng hợp hiệu năng 10 mô hình hồi quy (Sắp xếp theo độ phức tạp)	53
5.1	So sánh hiệu năng giữa CatBoost (2018) và TabNet (2021) . . . . .	56
5.2	So sánh Stacking Ensemble với mô hình đơn lẻ tốt nhất (CatBoost) . . .	58

# Danh mục mã nguồn

---

5.1	Đoạn code xử lý API /predict . . . . .	66
5.2	Đoạn code xử lý API /predict-batch . . . . .	67
5.3	Đoạn code xử lý API /chat với LangChain Agent . . . . .	68
5.4	Đoạn code xử lý API /dashboard-stats . . . . .	69
5.5	Đoạn code xử lý API /history . . . . .	70

# Chương 1

## Tổng quan về bài toán

### 1.1 Bối cảnh thị trường và Tính cấp thiết của đề tài

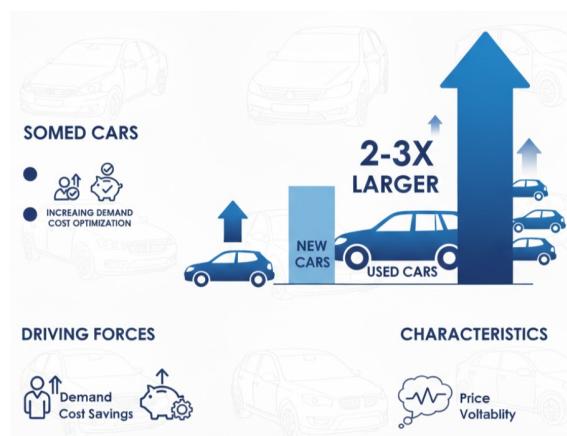
#### 1.1.1 Sự dịch chuyển xu hướng tiêu dùng

Trong bối cảnh kinh tế hiện đại, thị trường ô tô đã qua sử dụng (*Used Car Market*) đang chứng kiến sự tăng trưởng mạnh mẽ và dần chiếm ưu thế so với thị trường xe mới. Tại các quốc gia phát triển, quy mô giao dịch của thị trường này hiện đã lớn gấp 2 đến 3 lần so với xe mới.

Sự dịch chuyển này được thúc đẩy bởi hai động lực chính:

1. **Nhu cầu sở hữu phương tiện cá nhân tăng cao:** Người tiêu dùng ngày càng có nhu cầu chủ động trong việc di chuyển.
2. **Xu hướng tối ưu hóa chi phí:** Vấn đề khấu hao tài sản là yếu tố then chốt. Một chiếc xe mới có thể mất giá trị đáng kể ngay trong những năm đầu sử dụng. Do đó, việc lựa chọn xe đã qua sử dụng giúp người mua tối ưu hóa bài toán tài chính.

Tuy nhiên, đặc thù lớn nhất của thị trường này là giá xe không cố định (*Fixed Price*) mà biến động liên tục theo thời gian, tình trạng thực tế của xe và quy luật cung cầu.



Hình 1.1: Xu hướng tăng trưởng của thị trường xe đã qua sử dụng

### 1.1.2 Thực trạng và Vấn đề (Pain Points)

Mặc dù thị trường rất sôi động, nhưng quá trình định giá truyền thống đang gặp phải những ”điểm nghẽn” lớn, tạo ra rủi ro cho cả người mua và người bán. Các vấn đề cốt lõi bao gồm:

#### Thông tin bất cân xứng (Information Asymmetry)

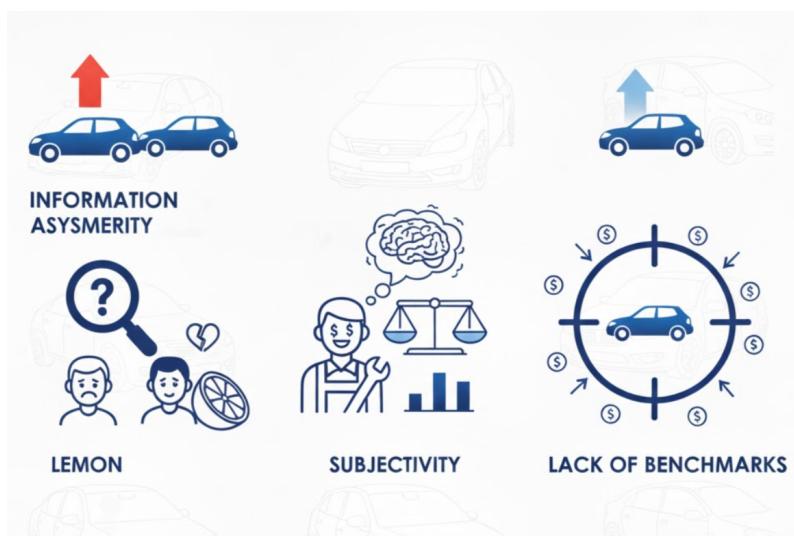
Đây là hiện tượng phổ biến còn được biết đến với tên gọi ”Thị trường quả chanh” (*The Market for Lemons*). Người bán thường nắm rõ lịch sử và chất lượng thực tế của xe hơn người mua. Điều này dẫn đến rủi ro người mua phải chi trả mức giá cao hơn giá trị thực tế của phương tiện.

#### Định giá cảm tính (Subjectivity)

Hiện nay, việc định giá phần lớn phụ thuộc vào kinh nghiệm chủ quan của thợ xe hoặc cảm quan của người bán. Quá trình này thiếu vắng cơ sở dữ liệu thống kê định lượng, dẫn đến sự chênh lệch lớn về giá cho cùng một mẫu xe và gây khó khăn trong việc thương lượng.

#### Thiếu tham chiếu chuẩn (Lack of Benchmarks)

Rất khó để xác định đâu là mức ”Giá công bằng” (*Fair Price*) cho một cấu hình xe cụ thể (dựa trên Model, Odo, Năm sản xuất...). Sự thiếu hụt một hệ thống tham chiếu chuẩn khiến tính thanh khoản của thị trường bị hạn chế.



Hình 1.2: Các rủi ro chính trong giao dịch xe cũ: Thông tin bất cân xứng và Định giá cảm tính

## 1.2 Phát biểu bài toán

Dựa trên bối cảnh thị trường và những thách thức đã phân tích, đề tài tập trung giải quyết hai bài toán cốt lõi: Bài toán nghiệp vụ dành cho người dùng cuối và Bài toán kỹ thuật dành cho việc xây dựng mô hình.

### 1.2.1 Bài toán Nghiệp vụ (Business Problem)

#### Mục tiêu cốt lõi

Xây dựng một hệ thống hỗ trợ ra quyết định (Decision Support System) nhằm **tối ưu hóa lợi ích kinh tế** và **minh bạch hóa thông tin** cho các bên tham gia thị trường (Người mua, Người bán và Sàn giao dịch). Hệ thống cần đóng vai trò như một trọng tài khách quan, cung cấp mức giá tham chiếu tin cậy dựa trên dữ liệu lịch sử.

#### Các câu hỏi kinh doanh (Business Questions)

Để đạt được mục tiêu trên, hệ thống cần trả lời được ba câu hỏi lớn:

1. **Định giá chính xác:** Làm thế nào để xác định mức giá công bằng (*Fair Price*) cho một chiếc xe cũ dựa trên biến động thực tế của thị trường?
2. **Phân tích nhân tố:** Yếu tố nào (Thương hiệu, Số dặm, Năm sản xuất...) ảnh hưởng lớn nhất đến sự mất giá (khấu hao) của phương tiện?
3. **Nguồn sai số:** Mức độ sai số bao nhiêu là chấp nhận được để đảm bảo tính thanh khoản và niềm tin của người dùng?

#### Yêu cầu phi chức năng

Hệ thống cần đáp ứng các tiêu chí: **Khách quan** (dựa trên dữ liệu, không cảm tính), **Nhanh chóng** (phản hồi thời gian thực) và **Đa chiều** (phân tích xu hướng theo nhiều thuộc tính khác nhau).

### 1.2.2 Bài toán Kỹ thuật (Technical Problem)

#### Định nghĩa toán học

Dưới góc độ Khoa học dữ liệu, đây là bài toán **Học máy có giám sát** (Supervised Learning) thuộc lớp bài toán **Hồi quy** (Regression). Bài toán được mô hình hóa như sau:

$$Y = f(\mathbf{X}) + \epsilon \quad (1.1)$$

Trong đó:

- $\mathbf{X} = (x_1, x_2, \dots, x_n)$ : Vector đặc trưng đầu vào (Features) bao gồm các thông tin của xe như: Hãng sản xuất (*Make*), Dòng xe (*Model*), Năm sản xuất (*Year*), Số dặm (*Mileage*), v.v.
- $Y$ : Biến mục tiêu (Target variable) là Giá xe (biến liên tục).
- $\epsilon$ : Sai số ngẫu nhiên không thể giải thích bởi mô hình.
- $f$ : Hàm ánh xạ cần tìm để xác định mối quan hệ giữa  $\mathbf{X}$  và  $Y$ .

### Thách thức kỹ thuật (Challenges)

Thách thức lớn nhất khi xử lý dữ liệu xe cũ là vấn đề **Biến phân loại có số chiều lớn** (*High Cardinality Categorical Variables*).

- Các biến như *Model* (Dòng xe) hay *Make* (Hãng xe) có hàng trăm giá trị khác nhau.
- Nếu sử dụng các kỹ thuật mã hóa truyền thống như One-Hot Encoding, số chiều dữ liệu sẽ tăng đột biến, gây ra hiện tượng "Lời nguyền số chiều" (*Curse of Dimensionality*) và làm chậm quá trình huấn luyện .

### Yêu cầu kỹ thuật

Mô hình đề xuất cần thỏa mãn các tiêu chí khắt khe sau:

1. **Độ chính xác:** Tối thiểu hóa hàm mất mát, cụ thể là các chỉ số RMSE (Root Mean Squared Error) và MAE (Mean Absolute Error).
2. **Hiệu năng:** Đảm bảo thời gian suy diễn (*Inference Time*) đủ nhanh (mục tiêu  $< 50\text{ms}$ ) để tích hợp vào ứng dụng Web thời gian thực.
3. **Tính giải thích:** Có khả năng chỉ ra mức độ quan trọng của các đặc trưng (*Feature Importance*), giúp giải thích lý do tại sao một chiếc xe được định giá ở mức đó.

## 1.3 Mô hình hóa bài toán (Input - Process - Output)

Để xây dựng hệ thống định giá tự động, bài toán được mô hình hóa dưới dạng một quy trình khép kín: từ việc thu thập các vector đặc trưng đầu vào, đi qua mô hình xử lý hồi quy, và trả về kết quả là giá trị thực của xe.

### 1.3.1 Phân tích Vector đặc trưng đầu vào (Input Vectors)

Dựa trên dữ liệu thu thập được từ Kaggle và khảo sát thực tế, vector đầu vào  $X$  được chia thành 3 nhóm thuộc tính chính, đóng vai trò quyết định đến giá trị của một chiếc xe:

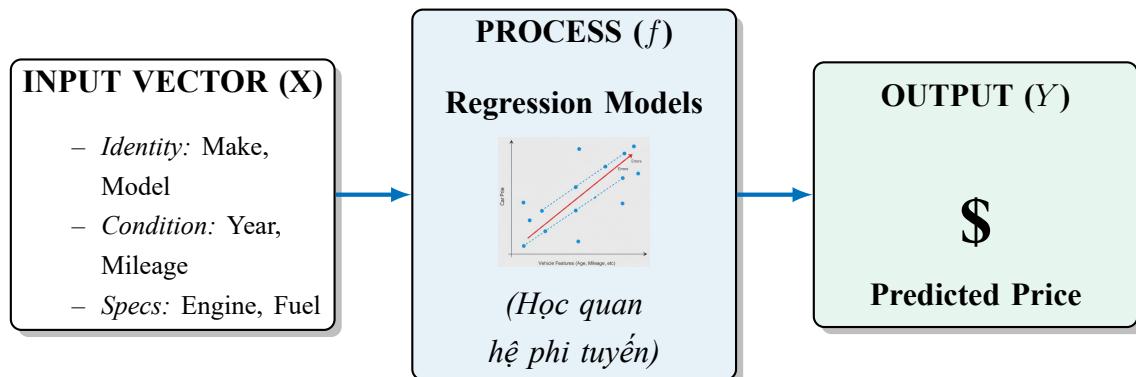
1. **Nhóm Định danh (Identity):** Bao gồm các thuộc tính như Hãng sản xuất (*Make*) và Dòng xe (*Model*).
  - **Vai trò:** Đây là yếu tố quyết định phân khúc thị trường (bình dân hay hạng sang) và giá trị thương hiệu (*Brand Value*). Ví dụ, cùng một năm sản xuất, xe của Audi sẽ có định giá khác biệt so với xe Ford.
2. **Nhóm Hiện trạng (Condition):** Bao gồm Năm sản xuất (*Year*) và Số dặm đã đi (*Mileage*).
  - **Vai trò:** Phản ánh trực tiếp mức độ khấu hao (*Depreciation*) và hao mòn vật lý của phương tiện. Số dặm càng cao, giá trị xe càng giảm do rủi ro hỏng hóc tăng lên.
3. **Nhóm Thông số kỹ thuật (Technical Specs):** Bao gồm Loại động cơ (*Engine Size*), Hộp số (*Transmission*) và Loại nhiên liệu (*Fuel Type*).
  - **Vai trò:** Ảnh hưởng trực tiếp đến hiệu năng vận hành và chi phí nuôi xe (tiêu hao nhiên liệu, thuế phí). Các xe sử dụng động cơ lớn hoặc nhiên liệu Diesel thường có xu hướng giữ giá khác biệt so với xe động cơ nhỏ.

### 1.3.2 Mô hình xử lý (Process) và Đầu ra (Output)

Quá trình xử lý và đầu ra của bài toán được mô tả như sau:

- **Process ( $f$ ):** Sử dụng các thuật toán Hồi quy (Regression Models) để học mối quan hệ phi tuyến tính phức tạp giữa các đặc trưng đầu vào và giá xe. Hệ thống sẽ đi từ các mô hình tuyến tính đơn giản (Linear Regression) đến các mô hình học máy nâng cao (Boosting) để tìm ra hàm  $f$  tối ưu nhất.
- **Output ( $Y$ ):** Biến mục tiêu là **Giá xe (Price)**, một biến số thực liên tục (*Continuous Real Value*), đại diện cho giá trị dự báo tại thời điểm truy vấn.

Mô hình luồng dữ liệu tổng quát được minh họa trong Hình 1.3.



Hình 1.3: Sơ đồ mô hình hóa bài toán Input - Process - Output

## 1.4 Phương pháp tiếp cận và Các thuật toán nghiên cứu

Để giải quyết bài toán hồi quy với dữ liệu hỗn hợp (bao gồm cả biến số và biến phân loại), nghiên cứu thực hiện đánh giá toàn diện trên 4 nhóm giải thuật phổ biến, từ đó đề xuất một kiến trúc mô hình tổ hợp (Ensemble) tối ưu cho việc triển khai thực tế.

Các nhóm giải thuật được nghiên cứu bao gồm:

### 1.4.1 Nhóm 1: Các mô hình Tuyến tính (Linear Models)

Đây là nhóm mô hình cơ sở (*Baseline*) được sử dụng để thiết lập mức chuẩn đánh giá độ phức tạp của dữ liệu.

- **Linear Regression:** Mô hình hồi quy tuyến tính cổ điển, giả định mối quan hệ giữa biến đầu vào và đầu ra là tuyến tính.
- **Ridge Regression ( $L_2$ ) & Lasso Regression ( $L_1$ ):** Các biến thể bổ sung thành phần điều chỉnh (*Regularization*) để kiểm soát hiện tượng quá khớp (*Overfitting*) và chọn lọc đặc trưng.

### 1.4.2 Nhóm 2: Các mô hình Phi tuyến性和 Khoảng cách

Nhóm này nhằm khai thác các mối quan hệ phi tuyến tính hoặc dựa trên sự tương đồng giữa các mẫu dữ liệu.

- **K-Nearest Neighbors (KNN):** Thuật toán dựa trên khoảng cách, dự đoán giá xe dựa trên  $K$  chiếc xe tương tự nhất trong tập dữ liệu.

- **Support Vector Regression (SVR):** Sử dụng các hàm nhân (*Kernel functions*) để ánh xạ dữ liệu sang không gian chiều cao hơn nhằm tìm kiếm siêu phẳng hồi quy tối ưu.

### 1.4.3 Nhóm 3: Các mô hình Cây quyết định (Tree-based Models)

- **Decision Tree:** Mô hình dạng cây với các quy tắc *If-Then-Else* đơn giản, dễ giải thích nhưng dễ bị nhiễu.
- **Random Forest:** Thuật toán tổ hợp (*Bagging*) kết hợp nhiều cây quyết định độc lập để giảm phương sai và tăng độ ổn định cho mô hình.

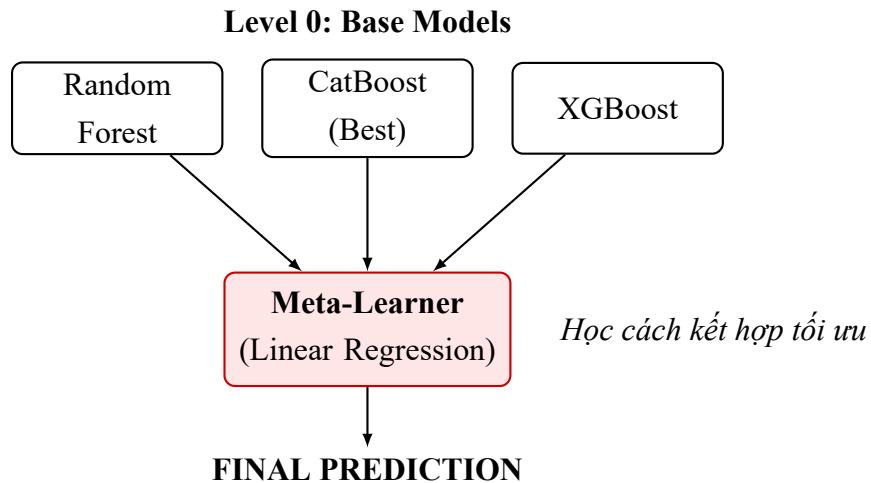
### 1.4.4 Nhóm 4: Các mô hình Tăng cường (Gradient Boosting)

Đây là nhóm thuật toán hiện đại (*State-of-the-Art*) cho dữ liệu dạng bảng, hoạt động theo cơ chế *Boosting* - xây dựng các mô hình tuần tự để sửa lỗi cho nhau.

- **Gradient Boosting Machine (GBM) & XGBoost:** Các thuật toán mạnh mẽ với khả năng tối ưu hóa hàm mất mát cực tốt.
- **CatBoost (Categorical Boosting):** Thuật toán trọng tâm của nghiên cứu. CatBoost vượt trội nhờ khả năng xử lý trực tiếp các biến phân loại (*Categorical features*) mà không cần mã hóa One-Hot phức tạp, đồng thời sử dụng cơ chế *Ordered Boosting* để chống rò rỉ dữ liệu (*Data Leakage*).

### 1.4.5 Mô hình đề xuất: Stacking Ensemble Regressor

Dựa trên kết quả thực nghiệm của các mô hình đơn lẻ, nghiên cứu đề xuất sử dụng kỹ thuật **Stacking Ensemble** để đóng gói sản phẩm cuối cùng.



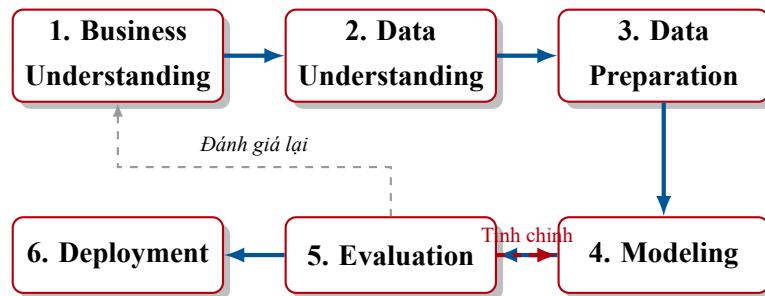
Hình 1.4: Kiến trúc Stacking Ensemble để xuất

Stacking (Stacked Generalization) là kỹ thuật học máy tổ hợp sử dụng một mô hình cấp cao (*Meta-Learner*) để học cách kết hợp các dự đoán từ các mô hình cơ sở (*Base Learners*). Trong đồ án này, Stacking Ensemble được lựa chọn để đóng gói ứng dụng vì hai lý do:

- Tối đa hóa độ chính xác:** Tận dụng điểm mạnh của từng mô hình thành phần (ví dụ: khả năng xử lý biến phân loại của CatBoost kết hợp với khả năng tổng quát hóa của Random Forest).
- Giảm thiểu rủi ro:** Việc kết hợp nhiều mô hình giúp hệ thống ổn định hơn (*Robustness*), tránh trường hợp một mô hình đơn lẻ bị sai lệch lớn trên tập dữ liệu lạ.

## 1.5 Quy trình thực hiện (CRISP-DM)

Để đảm bảo tính khoa học và khả năng ứng dụng thực tế, nghiên cứu này tuân thủ chặt chẽ quy trình chuẩn công nghiệp **CRISP-DM** (Cross-Industry Standard Process for Data Mining) [1]. Đây là quy trình khai phá dữ liệu được sử dụng rộng rãi nhất hiện nay, bao gồm 6 giai đoạn khép kín như minh họa trong Hình 1.5.



Hình 1.5: Quy trình CRISP-DM áp dụng trong nghiên cứu

Chi tiết các bước thực hiện trong phạm vi đồ án:

- Hiểu nghiệp vụ (Business Understanding):** Xác định mục tiêu tối ưu hóa lợi ích kinh tế cho người dùng và các câu hỏi kinh doanh cần giải quyết (đã trình bày tại mục 1.2).
- Hiểu dữ liệu (Data Understanding):** Thu thập và phân tích tập dữ liệu từ Kaggle (97,712 bản ghi). Thực hiện thống kê mô tả, trực quan hóa phân phối và phân tích tương quan giữa các biến.
- Chuẩn bị dữ liệu (Data Preparation):** Đây là trọng tâm nghiên cứu thứ nhất. Thực hiện làm sạch dữ liệu (xử lý *Missing values*, loại bỏ *Outliers*), biến đổi dữ liệu (*Log Transformation*) và trích chọn đặc trưng (*Feature Engineering*) để tạo ra bộ dữ liệu sạch phục vụ huấn luyện.
- Mô hình hóa (Modeling):** Đây là trọng tâm nghiên cứu thứ hai. Thực hiện huấn luyện và so sánh thực nghiệm trên **10 thuật toán** khác nhau, từ các mô hình tuyến tính cơ bản đến các mô hình Boosting hiện đại nhất (CatBoost, XGBoost).
- Đánh giá (Evaluation):** Kiểm định chất lượng mô hình thông qua bộ chỉ số *RMSE*, *MAE*, *MAPE* và *R<sup>2</sup>*. Phân tích các trường hợp dự đoán sai lệch lớn (*Error Analysis*) để tinh chỉnh lại mô hình.
- Triển khai (Deployment):** Đóng gói mô hình tối ưu nhất thành ứng dụng Web (sử dụng ReactJS/FastAPI) để người dùng cuối có thể tương tác và định giá xe theo thời gian thực.

#### Sản phẩm đầu ra của đề tài:

- Báo cáo phân tích dữ liệu và mô hình chi tiết.
- Hệ thống Web App minh họa (Demo) với giao diện người dùng thân thiện.

# Chương 2

## Khai phá dữ liệu

### 2.1 Tổng quan về tập dữ liệu (Dataset Overview)

Dữ liệu là nguyên liệu đầu vào quan trọng nhất quyết định chất lượng của mọi mô hình Học máy. Trong phạm vi đề tài này, nghiên cứu sử dụng tập dữ liệu thực tế về thị trường ô tô đã qua sử dụng, bao gồm đầy đủ các thông tin từ đặc điểm kỹ thuật đến lịch sử vận hành của phương tiện.

#### 2.1.1 Nguồn gốc và Phạm vi dữ liệu

Bộ dữ liệu được thu thập từ nền tảng Kaggle, với tên gọi gốc là "*90,000+ Cars Data From 1970 to 2024*" do tác giả Meruvu Likith chia sẻ [2]. Đây là tập dữ liệu tổng hợp từ các tin đăng bán xe thực tế, phản ánh bức tranh toàn cảnh thị trường ô tô cũ trong giai đoạn từ năm 1970 đến năm 2024.

Các đặc điểm thống kê cơ bản của tập dữ liệu:

- Kích thước mẫu:** 97,712 bản ghi (observations). Đây là lượng dữ liệu đủ lớn để các mô hình học máy hiện đại (như Deep Learning hay Gradient Boosting) có thể hội tụ tốt, hạn chế hiện tượng quá khóp (Overfitting).
- Số lượng thuộc tính:** 10 thuộc tính (features), bao gồm cả biến định lượng và biến định danh.
- Tính chất dữ liệu:** Dữ liệu thuộc dạng hỗn hợp (*Mixed Data Type*). Bên cạnh các thông tin sạch, dữ liệu thô ban đầu có chứa nhiễu (*Noise*) và các giá trị bị khuyết thiếu (*Missing Values*), đòi hỏi phải trải qua quá trình tiền xử lý kỹ lưỡng trước khi đưa vào huấn luyện.

Bảng 2.1: Mô tả các thuộc tính trong tập dữ liệu

Tên thuộc tính	Loại dữ liệu	Ý nghĩa và Mô tả
<b>Biến định danh (Categorical Features)</b>		
Manufacturer	Nominal	Hãng sản xuất xe (Audi, BMW, Ford...). Yếu tố quyết định giá trị thương hiệu.
model	Nominal	Tên dòng xe cụ thể (Ví dụ: Fiesta, Golf...). Biến này có số chiều dữ liệu lớn.
transmission	Nominal	Loại hộp số (Manual, Automatic, Semi-Auto).
fuelType	Nominal	Loại nhiên liệu (Petrol, Diesel, Hybrid, Electric).
<b>Biến định lượng (Numerical Features)</b>		
year	Discrete	Năm sản xuất. Xe càng mới giá trị thường càng cao.
mileage	Continuous	Số dặm đã di chuyển (Odo). Phản ánh mức độ hao mòn thực tế.
engineSize	Continuous	Dung tích động cơ (Lít). Động cơ lớn thường đi kèm phân khúc cao.
mpg	Continuous	Miles Per Gallon: Mức tiêu thụ nhiên liệu.
tax	Continuous	Thuế đường bộ hàng năm.
<b>Biến mục tiêu (Target Variable)</b>		
price	Continuous	Giá bán của xe. Giá trị thực tế mô hình cần dự đoán.

Việc hiểu rõ ý nghĩa của từng thuộc tính giúp định hướng cho quá trình trích chọn đặc trưng (*Feature Engineering*) ở các bước sau, ví dụ như việc kết hợp year để tính ra tuổi đời của xe (CarAge).

## 2.2 Phân tích khám phá dữ liệu (Exploratory Data Analysis)

Phân tích khám phá dữ liệu (EDA) là bước quan trọng giúp thấu hiểu cấu trúc, phát hiện các giá trị bất thường (*Outliers*) và tìm ra các khuôn mẫu tiềm ẩn trong dữ liệu trước khi tiến hành mô hình hóa. Trong phần này, nghiên cứu tập trung vào hai khía cạnh chính: Thống kê mô tả và Phân tích tương quan.

### 2.2.1 Thống kê mô tả (Descriptive Statistics)

#### Đối với biến định lượng (Numerical Variables)

Bảng thống kê tóm tắt các biến số quan trọng (price, mileage, year, tax, mpg, engineSize) được trình bày trong Hình 2.1.

	year	price	mileage	tax	mpg	engineSize
<b>count</b>	97712.000000	97712.000000	97712.000000	97712.000000	97712.000000	97712.000000
<b>mean</b>	2017.066502	16773.487555	23219.475499	120.142408	55.205623	1.664913
<b>std</b>	2.118661	9868.552222	21060.882301	63.357250	16.181659	0.558574
<b>min</b>	1970.000000	450.000000	1.000000	0.000000	0.300000	0.000000
<b>25%</b>	2016.000000	9999.000000	7673.000000	125.000000	47.100000	1.200000
<b>50%</b>	2017.000000	14470.000000	17682.500000	145.000000	54.300000	1.600000
<b>75%</b>	2019.000000	20750.000000	32500.000000	145.000000	62.800000	2.000000
<b>max</b>	2024.000000	159999.000000	323000.000000	580.000000	470.800000	6.600000

Hình 2.1: Bảng thống kê mô tả các biến định lượng

Từ bảng số liệu, ta rút ra các nhận xét quan trọng về đặc điểm phân phối:

- **Biến mục tiêu (Price):** Giá xe có biên độ dao động rất lớn, từ mức thấp nhất là \$450 đến cao nhất là \$159,999. Độ lệch chuẩn lớn ( $\sigma \approx 9,868$ ) cho thấy sự phân tán mạnh của dữ liệu. Giá trị trung bình ( $\mu \approx 16,773$ ) cao hơn trung vị (50% percentile  $\approx 14,470$ ), gợi ý rằng phân phối giá xe bị lệch phải (*Right-skewed*).
- **Số dặm (Mileage):** Tương tự như giá, số dặm trải rộng từ xe "lướt" đến các xe đã vận hành rất nhiều (tối đa 323,000 dặm), phản ánh đúng thực tế đa dạng của thị trường xe cũ.
- **Năm sản xuất (Year):** Dữ liệu bao phủ các dòng xe từ năm 1970 đến 2024, tuy nhiên phần lớn tập trung vào các xe đời mới (Trung vị là năm 2017).

### Đối với biến định danh (Categorical Variables)

Thống kê tần suất xuất hiện (Frequency) của các biến phân loại như Hãng xe, Dòng xe được thể hiện trong Hình 2.2.

	model	transmission	fuelType	Manufacturer
<b>count</b>	97712	97712	97712	97712
<b>unique</b>	196	4	5	9
<b>top</b>	Fiesta	Manual	Petrol	ford
<b>freq</b>	6509	55502	53982	17811

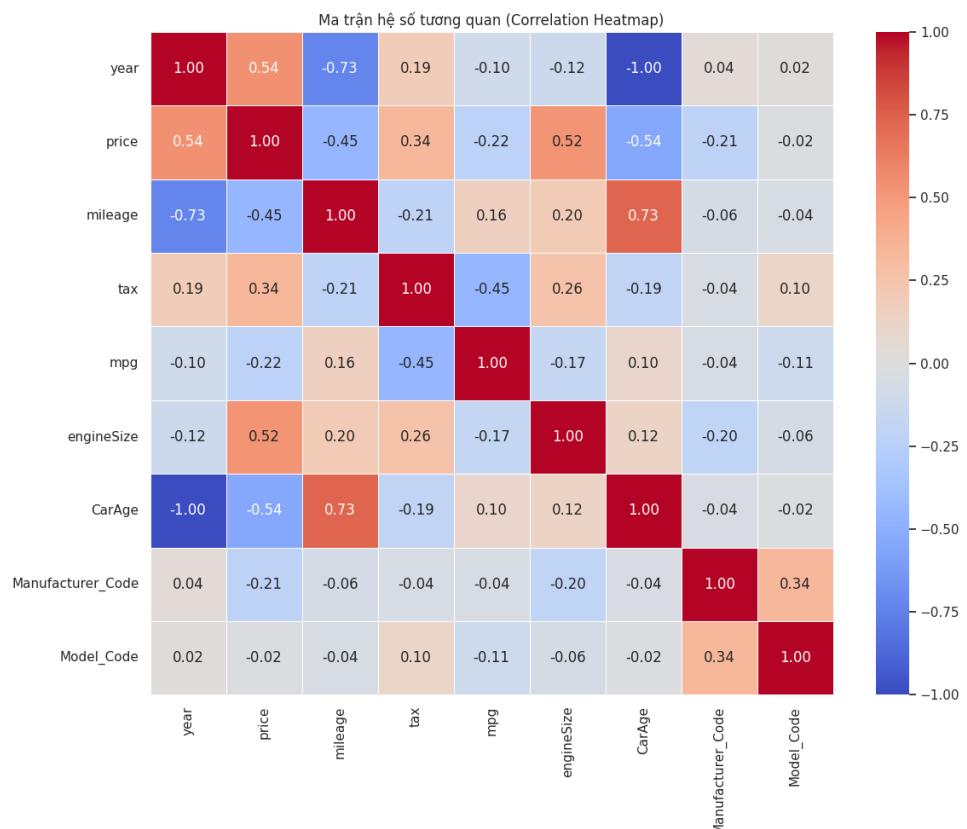
Hình 2.2: Thống kê các biến phân loại chính

Kết quả phân tích cho thấy:

- Dòng xe (Model):** Có tính đa dạng cao với 196 giá trị duy nhất. Trong đó, *Ford Fiesta* là dòng xe xuất hiện nhiều nhất (6,509 bản ghi), cho thấy đây là mẫu xe phổ thông có thanh khoản cao.
- Hộp số (Transmission):** Xe số sàn (*Manual*) chiếm áp đảo với 55,502 bản ghi, phản ánh đặc thù của thị trường Châu Âu (nguồn dữ liệu chính) thường ưa chuộng xe số sàn.
- Nhiên liệu (FuelType):** Xe chạy xăng (*Petrol*) chiếm đa số, theo sau là xe Diesel. Các dòng xe năng lượng mới (*Hybrid*, *Electric*) chiếm tỷ trọng nhỏ nhưng đang có xu hướng gia tăng.

### 2.2.2 Phân tích tương quan (Correlation Analysis)

Để đánh giá mối quan hệ tuyến tính giữa các biến số, nghiên cứu sử dụng hệ số tương quan Pearson. Ma trận tương quan (Heatmap) được trực quan hóa trong Hình 2.3.



Hình 2.3: Ma trận tương quan giữa các biến (Heatmap)

### Tương quan với biến mục tiêu (Price)

- **Tương quan dương (+):** Biến engineSize ( $r \approx 0.52$ ) và year ( $r \approx 0.54$ ) có tương quan thuận chiều đáng kể với giá xe. Điều này phù hợp với quy luật: Xe càng mới hoặc động cơ càng lớn thì giá trị càng cao.
- **Tương quan âm (-):** Biến mileage ( $r \approx -0.45$ ) và CarAge ( $r \approx -0.54$ ) có tương quan nghịch biến. Xe tuổi đời cao hoặc đi nhiều dặm thì giá trị giảm do khấu hao.

### Cảnh báo đa cộng tuyến (Multicollinearity)

Một vấn đề cần lưu ý khi xây dựng mô hình hồi quy tuyến tính là hiện tượng đa cộng tuyến:

- Cặp biến year và CarAge có tương quan tuyệt đối ( $r = -1.0$ ) vì bản chất  $CarAge = 2025 - Year$ . Do đó, cần loại bỏ một trong hai biến để tránh gây nhiễu.
- Các cặp biến khác như tax và engineSize có tương quan nhưng chưa đến ngưỡng nguy hiểm ( $|r| < 0.8$ ), có thể giữ lại cho các mô hình phi tuyến.

## 2.3 Quy trình làm sạch và Biến đổi dữ liệu (Data Cleaning & Transformation)

Dựa trên kết quả EDA, quy trình chuẩn hóa dữ liệu được thực hiện nhằm đảm bảo tính toàn vẹn và độ tin cậy cho mô hình dự báo.

### 2.3.1 Xử lý dữ liệu nhiễu và Giá trị thiêu

#### Chiến lược xử lý giá trị thiêu (Imputation)

Để duy trì kích thước mẫu và hạn chế sai lệch phân phối gốc, dữ liệu khuyết thiêu được xử lý dựa trên đặc tính thống kê:

- **Biến định lượng:** Điền thê bằng giá trị *Trung vị (Median)*. Trung vị được lựa chọn nhờ đặc tính bền vững (robust), ít chịu ảnh hưởng bởi các giá trị ngoại lai so với trung bình cộng.
- **Biến định danh:** Điền thê bằng giá trị *Yếu vị (Mode)* - giá trị có tần suất xuất hiện cao nhất.

### Loại bỏ ngoại lai (Outlier Removal)

Biến mục tiêu price chứa các điểm dữ liệu bất thường gây nhiễu. Phương pháp *Khoảng từ phân vị (IQR)* được áp dụng:

1. Tính toán chỉ số IQR:

$$IQR = Q_3 - Q_1 \quad (2.1)$$

Trong đó  $Q_1, Q_3$  lần lượt là tứ phân vị thứ nhất (25%) và thứ ba (75%).

2. Xác định miền giá trị chấp nhận:

$$\text{Lower Bound} = Q_1 - 1.5 \times IQR \quad (2.2)$$

$$\text{Upper Bound} = Q_3 + 1.5 \times IQR \quad (2.3)$$

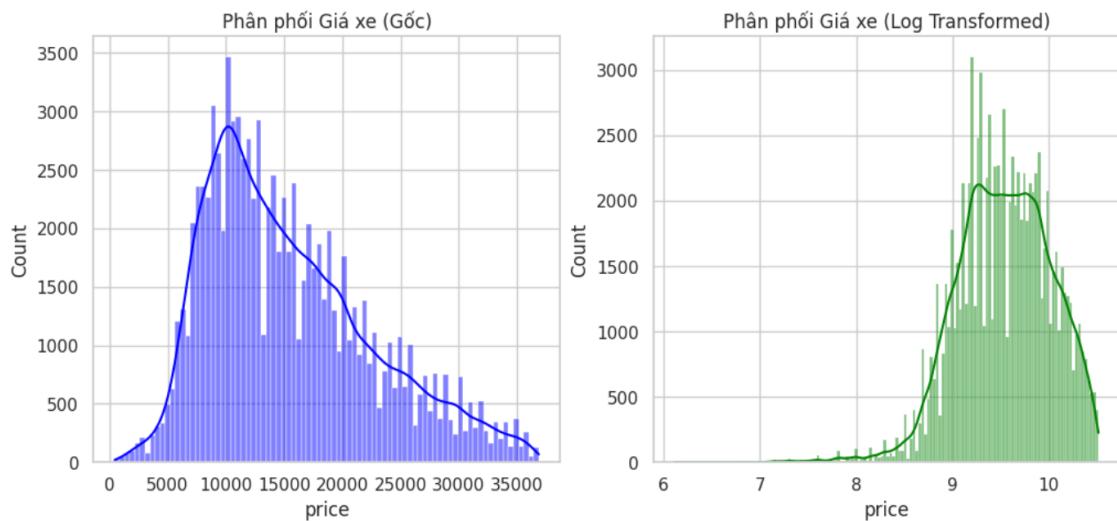
3. **Thực thi:** Chỉ giữ lại các bản ghi thỏa mãn  $\text{Lower Bound} \leq \text{Price} \leq \text{Upper Bound}$ . Các giá trị ngoài khoảng này bị loại bỏ khỏi tập huấn luyện.

### 2.3.2 Chuẩn hóa phân phối biến mục tiêu (Target Variable Normalization)

#### Phân tích trực quan độ lệch (Skewness)

Quan sát biểu đồ phân phối gốc của biến price (Hình 2.4, bên trái), ta nhận thấy dữ liệu có xu hướng **lệch trái (Left-skewed)** rõ rệt. Đỉnh phân phối tập trung ở vùng giá thấp (< \$15,000) nhưng đuôi phân phối kéo dài về phía giá trị dương lớn.

Việc sử dụng trực tiếp biến mục tiêu bị lệch này sẽ làm giảm hiệu suất của các mô hình hồi quy tuyến tính (Linear Regression), do vi phạm giả định về phân phối chuẩn của phản ứng.



Hình 2.4: So sánh phân phối Giá xe: Trước (Trái) và Sau khi biến đổi Log (Phải)

### Hiệu quả của biến đổi Logarithm

Để khắc phục, nghiên cứu áp dụng hàm biến đổi  $y' = \ln(y + 1)$ . Kết quả trên biểu đồ bên phải (Hình 2.4) cho thấy:

- Phân phối mới có dạng hình chuông, tiệm cận với **Phân phối chuẩn (Gaussian Distribution)**.
- Các giá trị ngoại lai cực đại đã được "kéo" lại gần giá trị trung tâm, giúp ổn định phương sai cho mô hình.

## 2.4 Trích chọn đặc trưng và Tiền xử lý (Feature Engineering)

Sau bước làm sạch, dữ liệu được chuyển đổi thành các đặc trưng (*features*) phù hợp cho huấn luyện mô hình.

### 2.4.1 Tạo đặc trưng mới (Feature Extraction)

Trong bài toán định giá xe cũ, "Tuổi đời của xe" tác động trực tiếp đến tâm lý người mua hơn là năm sản xuất đơn thuần. Nghiên cứu tạo biến mới CarAge:

$$\text{CarAge} = \text{Current\_Year} - \text{Year\_of\_Production} \quad (2.4)$$

Với Current\_Year = 2025. Biến CarAge sẽ thay thế year trong quá trình huấn luyện.

## 2.4.2 Mã hóa biến định danh (Categorical Encoding)

Các mô hình học máy yêu cầu dữ liệu đầu vào dạng số. Hai kỹ thuật mã hóa được áp dụng tùy theo đặc thù biến:

- **One-Hot Encoding:** Áp dụng cho biến có số lượng giá trị duy nhất nhỏ (*Low Cardinality*) như transmission, fuelType. Mỗi giá trị chuyển thành một vector nhị phân.
- **Label Encoding:** Áp dụng cho biến có số lượng giá trị duy nhất lớn (*High Cardinality*) như Manufacturer, model để tránh bùng nổ số chiều dữ liệu (*Curse of Dimensionality*).

## 2.4.3 Chuẩn hóa dữ liệu (Feature Scaling)

Để tránh việc các biến có thang đo lớn (như mileage) lấn át các biến thang đo nhỏ (như engineSize), phương pháp **Min-Max Scaling** được sử dụng để đưa tất cả về khoảng [0, 1]:

$$X_{scaled} = \frac{X - X_{min}}{X_{max} - X_{min}} \quad (2.5)$$

## 2.4.4 Kết quả tiền xử lý (Final Processed Data)

Sau khi hoàn tất toàn bộ quy trình làm sạch, mã hóa và chuẩn hóa, tập dữ liệu thô ban đầu đã được chuyển đổi thành dạng ma trận số học hoàn chỉnh.

Hình 2.5 minh họa 5 dòng đầu tiên của tập dữ liệu sau khi xử lý. Các biến định danh đã được chuyển thành số (One-Hot/Label Encoding) và các biến số đã được đưa về thang đo [0, 1].

	model	year	price	mileage	tax	mpg	engineSize	Manufacturer	CarAge	transmission_Manual	transmission_Other	transmission_Semi-Auto	fuelType_Electric	fuelType_Hybrid	fuelType_Other	fuelType_Petrol	Manufacturer_Code	Model_Code
0	I10	2017	7495	0.036003	0.250000	0.127099	0.156730	hyundai	0.129630	True	False	False	False	False	False	True	3	78
1	Polo	2017	10986	0.028480	0.250000	0.124548	0.158730	volkswagen	0.129630	True	False	False	False	False	False	True	8	112
2	2 Series	2019	27990	0.004984	0.250000	0.104782	0.317460	BMW	0.092593	False	False	True	False	False	False	False	1	1
3	Yeti Outdoor	2017	12495	0.095849	0.258621	0.132837	0.317460	skoda	0.129630	True	False	False	False	False	False	False	5	176
4	Fiesta	2017	7996	0.059913	0.215517	0.114772	0.190476	ford	0.129630	True	False	False	False	False	False	True	2	58

Hình 2.5: Mẫu dữ liệu sau khi đã tiền xử lý và chuẩn hóa

Dữ liệu này đảm bảo sạch nhiều, đồng nhất về định dạng và sẵn sàng để đưa vào huấn luyện các mô hình học máy trong Chương 3.

## 2.5 Kết luận chương 2

Qua chương này, bộ dữ liệu thô ban đầu gồm 97,712 bản ghi đã được phân tích toàn diện, xử lý nhiễu, loại bỏ ngoại lai và chuyển đổi thành dạng vector số học chuẩn hóa. Đây là cơ sở dữ liệu chất lượng cao (*Processed Data*) sẵn sàng cho việc xây dựng và đánh giá mô hình trong Chương 3.

# Chương 3

## Thiết lập thực nghiệm

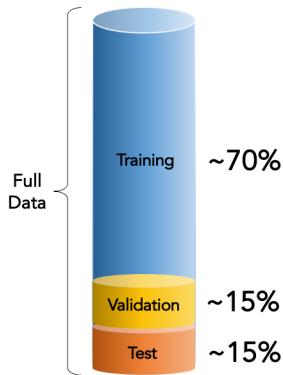
### 3.1 Chiến lược phân chia dữ liệu (Data Splitting Strategy)

Để đảm bảo tính khách quan của kết quả đánh giá và hạn chế hiện tượng quá khốp (Overfitting), dữ liệu sau khi tiền xử lý được phân chia một cách nghiêm ngặt theo chiến lược ba tập độc lập và kết hợp với kỹ thuật kiểm thử chéo.

#### 3.1.1 Phân chia tập dữ liệu (Train/Validation/Test Split)

Tổng số 97,712 bản ghi dữ liệu được chia ngẫu nhiên thành ba tập con riêng biệt với tỷ lệ **70:15:15**. Vai trò cụ thể của từng tập dữ liệu được mô tả như sau:

- Tập huấn luyện (Training Set - 70%)**: Bao gồm 68,398 bản ghi. Đây là dữ liệu được sử dụng trực tiếp để mô hình học các trọng số và quy luật từ các đặc trưng.
- Tập kiểm định (Validation Set - 15%)**: Bao gồm 14,657 bản ghi. Tập này được sử dụng trong quá trình huấn luyện để tinh chỉnh các siêu tham số (Hyperparameter Tuning) và đưa ra quyết định dừng sớm (Early Stopping) nhằm tránh Overfitting. Mô hình không được học trực tiếp từ tập này.
- Tập kiểm thử (Test Set - 15%)**: Bao gồm 14,657 bản ghi. Đây là tập dữ liệu hoàn toàn mới, được giữ kín cho đến khi kết thúc quá trình huấn luyện. Nó đóng vai trò như thước đo cuối cùng để đánh giá khả năng tổng quát hóa của mô hình trên dữ liệu thực tế chưa từng gặp.



Hình 3.1: Tỷ lệ phân chia dữ liệu cho thực nghiệm (70/15/15)

### 3.1.2 Kỹ thuật Kiểm thử chéo (K-Fold Cross-Validation)

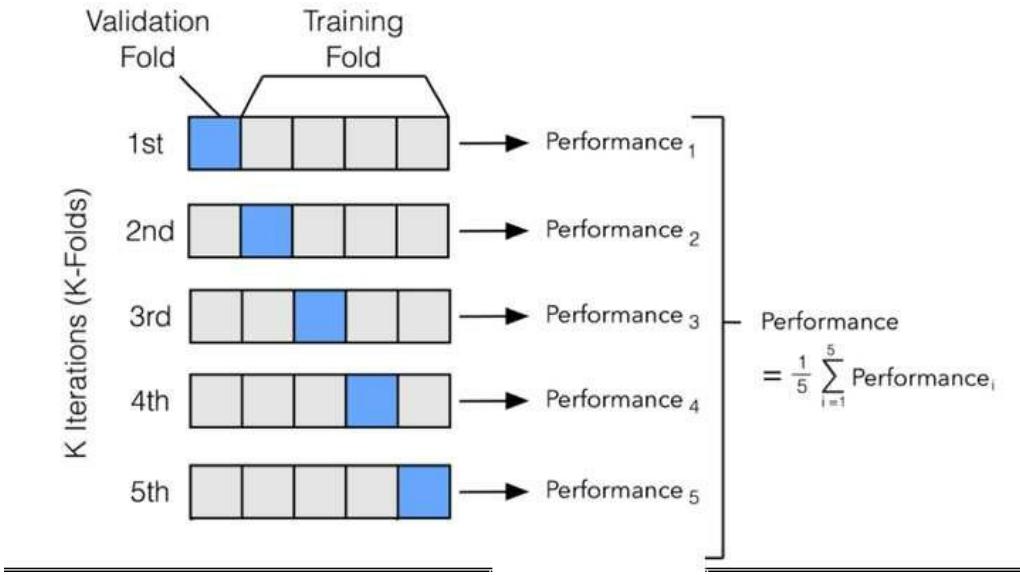
Bên cạnh việc chia tách tập kiểm định cố định, nghiên cứu áp dụng thêm kỹ thuật **K-Fold Cross-Validation** trên tập huấn luyện để đánh giá độ ổn định của mô hình. Cụ thể, kỹ thuật **5-Fold Cross-Validation** ( $K=5$ ) được lựa chọn:

#### Quy trình thực hiện

1. Tập huấn luyện (70% dữ liệu gốc) được chia ngẫu nhiên thành 5 phần (folds) có kích thước bằng nhau.
2. Quá trình huấn luyện được lặp lại 5 lần (iterations). Trong mỗi lần lặp:
  - Một phần (fold) được giữ lại làm tập kiểm định (Validation).
  - 4 phần còn lại được sử dụng làm tập huấn luyện (Training).
3. Kết quả đánh giá cuối cùng là trung bình cộng của các chỉ số đo lường (như RMSE, MAE) từ 5 lần lặp.

#### Ý nghĩa

Việc áp dụng K-Fold Cross-Validation giúp đảm bảo rằng mọi điểm dữ liệu trong tập huấn luyện đều có cơ hội được sử dụng cho cả việc học và việc kiểm định. Điều này giúp giảm thiểu sự thiên lệch (bias) do việc phân chia dữ liệu ngẫu nhiên gây ra, đặc biệt trong trường hợp dữ liệu có sự phân bố không đồng đều giữa các nhóm xe.



Hình 3.2: Mô hình minh họa kỹ thuật 5-Fold Cross-Validation

## 3.2 Hệ thống chỉ số đánh giá mô hình (Evaluation Metrics)

Việc lựa chọn thước đo phù hợp là yếu tố then chốt để đánh giá khách quan hiệu quả của các mô hình hồi quy. Trong nghiên cứu này, chúng tôi xây dựng một hệ thống đánh giá đa chiều, bao gồm các chỉ số đo lường sai số định lượng, mức độ phù hợp tổng thể và các tiêu chí về hiệu năng khi triển khai thực tế.

### 3.2.1 Các chỉ số đo lường sai số (Error Metrics)

Để lượng hóa mức độ chênh lệch giữa giá trị dự báo ( $\hat{y}$ ) và giá trị thực tế ( $y$ ), nghiên cứu sử dụng bộ ba chỉ số tiêu chuẩn sau đây:

#### Root Mean Squared Error (RMSE)

RMSE (Sai số trung bình phương gốc) là căn bậc hai của trung bình các bình phương sai số. Công thức được xác định như sau [3]:

$$RMSE = \sqrt{\frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i)^2} \quad (3.1)$$

**Ý nghĩa:** Do bản chất sử dụng bình phương sai số, RMSE có đặc tính ”phạt” rất nặng các trường hợp dự báo sai lệch lớn (large errors). Trong bài toán định giá xe, chỉ số này giúp phát hiện và hạn chế các trường hợp mô hình đưa ra mức giá ”thảm họa”(quá cao hoặc quá thấp so với thực tế), đảm bảo độ tin cậy của hệ thống.

### Mean Absolute Error (MAE)

MAE (Sai số tuyệt đối trung bình) đo lường độ lớn trung bình của các sai số trong tập dự báo mà không quan tâm đến chiều hướng (dương hay âm) [4]:

$$MAE = \frac{1}{n} \sum_{i=1}^n |y_i - \hat{y}_i| \quad (3.2)$$

**Ý nghĩa:** Khác với RMSE, MAE ít nhạy cảm hơn với các giá trị ngoại lai (outliers). Ưu điểm lớn nhất của MAE là tính dễ giải thích (interpretability): kết quả của MAE cùng đơn vị với biến mục tiêu (USD/Euro). Ví dụ,  $MAE = 1000$  nghĩa là trung bình mô hình đang đoán sai lệch khoảng \$1,000 trên mỗi chiếc xe.

### Mean Absolute Percentage Error (MAPE)

Do tập dữ liệu bao gồm cả các dòng xe phổ thông (giá vài nghìn USD) và xe sang (giá hàng trăm nghìn USD), sai số tuyệt đối đôi khi không phản ánh đúng mức độ nghiêm trọng. MAPE giải quyết vấn đề này bằng cách tính sai số tương đối theo phần trăm:

$$MAPE = \frac{100\%}{n} \sum_{i=1}^n \left| \frac{y_i - \hat{y}_i}{y_i} \right| \quad (3.3)$$

**Ý nghĩa:** MAPE cho phép đánh giá độ chính xác độc lập với quy mô giá trị của xe. Một sai số \$500 là lớn đối với xe giá \$1,000 (50%) nhưng lại rất nhỏ đối với xe giá \$50,000 (1%).

## 3.2.2 Hệ số xác định (Coefficient of Determination - $R^2$ )

Bên cạnh việc đo lường sai số, nghiên cứu sử dụng hệ số  $R^2$  để đánh giá mức độ phù hợp tổng thể của mô hình đối với sự biến thiên của dữ liệu:

$$R^2 = 1 - \frac{SS_{res}}{SS_{tot}} = 1 - \frac{\sum(y_i - \hat{y}_i)^2}{\sum(y_i - \bar{y})^2} \quad (3.4)$$

Trong đó  $SS_{res}$  là tổng bình phương sai số còn dư và  $SS_{tot}$  là tổng bình phương độ lệch toàn phần. Giá trị  $R^2$  càng gần 1 cho thấy mô hình càng giải thích tốt các biến động

của giá xe dựa trên các đặc trưng đầu vào [5]. Mục tiêu của nghiên cứu là đạt  $R^2 > 0.85$ .

### 3.2.3 Tiêu chí hiệu năng triển khai (Deployment Performance)

Để đảm bảo tính khả thi khi tích hợp mô hình vào ứng dụng Web thực tế, ngoài độ chính xác, mô hình còn được đánh giá dựa trên các tiêu chí phi chức năng:

- **Thời gian huấn luyện (Training Time):** Đánh giá chi phí thời gian để mô hình học từ dữ liệu. Yếu tố này quan trọng đối với khả năng bảo trì và cập nhật mô hình định kỳ khi có dữ liệu thị trường mới.
- **Thời gian suy diễn (Inference Time):** Là thời gian mô hình trả về kết quả dự báo cho một yêu cầu đơn lẻ. Để đảm bảo trải nghiệm người dùng mượt mà (real-time experience) trên ứng dụng, nghiên cứu đặt ngưỡng ràng buộc:

$$T_{inference} < 100 \text{ ms}$$

- **Khả năng giải thích (Interpretability):** Trong lĩnh vực kinh doanh, việc hiểu *tại sao* mô hình đưa ra mức giá đó quan trọng không kém độ chính xác. Các mô hình cần cung cấp được mức độ quan trọng của đặc trưng (Feature Importance) để minh bạch hóa quá trình định giá (ví dụ: Giá giảm chủ yếu do Tuổi đời xe hay do Số dặm đi được).

## 3.3 Phương pháp luận về Phân tích lỗi (Error Analysis Methodology)

Để không chỉ dừng lại ở các chỉ số đánh giá tổng quát (như RMSE hay  $R^2$ ), nghiên cứu thực hiện phân tích sâu vào các trường hợp mô hình dự báo sai lệch lớn nhất. Quy trình này giúp thấu hiểu hành vi của mô hình (Model Behavior) và nhận diện các hạn chế còn tồn tại.

### 3.3.1 Quy trình trích xuất lỗi (Error Extraction Process)

Quy trình phân tích các trường hợp tồi tệ nhất (*Worst-case Analysis*) được thực hiện qua các bước định lượng sau:

### Tính toán phần dư (Residuals Calculation)

Với mỗi điểm dữ liệu  $i$  trong tập kiểm thử (Test Set), phần dư  $e_i$  được tính toán là hiệu số giữa giá trị thực tế và giá trị dự báo:

$$e_i = y_i - \hat{y}_i \quad (3.5)$$

Trong đó  $y_i$  là giá thực của xe và  $\hat{y}_i$  là giá do mô hình dự đoán. Giá trị  $e_i > 0$  nghĩa là mô hình đang định giá thấp hơn thực tế (Underpricing), ngược lại  $e_i < 0$  là định giá cao hơn thực tế (Overpricing).

### Lọc các trường hợp sai số lớn nhất

Để tập trung vào các vấn đề nghiêm trọng, nghiên cứu tiến hành sắp xếp tập dữ liệu theo giá trị tuyệt đối của phần dư  $|e_i|$  giảm dần. Sau đó, trích xuất ra **Top 5%** các mẫu có sai số cao nhất. Tập con này đại diện cho những "ca khó" (hard cases) mà mô hình hiện tại chưa giải quyết tốt.

#### 3.3.2 Phân tích nguyên nhân (Root Cause Analysis)

Dựa trên danh sách các mẫu sai số lớn đã trích xuất, nghiên cứu tiến hành đổi chiều ngược lại với dữ liệu gốc để tìm hiểu nguyên nhân. Các nhóm nguyên nhân chính thường gặp bao gồm:

- **Nhiều dữ liệu (Outliers):** Các xe có thông tin đầu vào bất thường (ví dụ: xe đời rất sâu nhưng số dặm cực thấp, hoặc xe tai nạn bán xác) khiến mô hình đưa ra dự đoán sai lệch.
- **Mẫu hiếm (Rare Models):** Các dòng siêu xe hoặc xe phiên bản giới hạn (như Audi R8, BMW M Series) có tần suất xuất hiện trong tập huấn luyện quá thấp, khiến mô hình không đủ dữ liệu để học được quy luật giá chính xác.
- **Độ lệch của mô hình (Model Bias):** Bản thân thuật toán chưa đủ độ phức tạp để bao quát các mối quan hệ phi tuyến tính trong dữ liệu.

Bảng 3.1 dưới đây minh họa một số trường hợp dự báo sai lệch thực tế được trích xuất từ quá trình thí nghiệm:

Bảng 3.1: Phân tích nguyên nhân các trường hợp có sai số dự báo lớn

ID	Dòng xe	Giá thực	Dự báo	Sai số	Nhận định nguyên nhân
#1024	Audi R8	\$120,000	\$95,000	-\$25,000	Xe thể thao cao cấp hiếm gặp, thiếu dữ liệu huấn luyện đặc thù.
#5521	Ford Fiesta	\$5,000	\$8,500	+\$3,500	Xe đời cũ nhưng Odo thấp bất thường (Nhiều/Outlier).
#8812	BMW X5	\$45,000	\$42,000	-\$3,000	Sai số nằm trong biên độ chấp nhận được (khoảng 6%).

## 3.4 Kết luận chương 3

Chương 3 đã thiết lập một khung thực nghiệm chặt chẽ, từ việc phân chia dữ liệu 70/15/15, lựa chọn hệ thống chỉ số đánh giá đa chiều (RMSE, MAE, R2) đến quy trình phân tích lỗi chi tiết. Đây là nền tảng vững chắc để tiến hành huấn luyện và so sánh hiệu quả của các mô hình học máy trong Chương 4.

# Chương 4

## Xây dựng và Đánh giá mô hình

### 4.1 Chiến lược thực nghiệm

#### 4.1.1 Cơ sở lý luận

Trong bài toán Học máy nói chung và hồi quy nói riêng, không tồn tại một thuật toán duy nhất nào có thể hoạt động tối ưu trên mọi tập dữ liệu. Nguyên lý này được Wolpert khái quát hóa trong định lý “Không có bữa trưa nào miễn phí” (No Free Lunch Theorem) [6]. Hiệu suất của một mô hình phụ thuộc chặt chẽ vào cấu trúc dữ liệu, số lượng đặc trưng, và mối quan hệ tiềm ẩn giữa các biến.

Đối với tập dữ liệu giá xe ô tô đã qua sử dụng (như đã phân tích ở Chương 2), chúng ta đối mặt với các thách thức đặc thù: dữ liệu hỗn hợp (mixed data types) giữa số và phân loại, sự hiện diện của nhiễu (noise) và các mối quan hệ phi tuyến tính phức tạp. Do đó, thay vì lựa chọn ngẫu nhiên, nghiên cứu này đề xuất một chiến lược thực nghiệm có hệ thống, đi từ các mô hình đơn giản nhất để thiết lập mức chuẩn (Baseline) đến các mô hình phức tạp (State-of-the-Art) nhằm tối ưu hóa sai số.

Chiến lược được chia thành 4 giai đoạn tương ứng với 4 nhóm giải thuật, thực hiện kiểm chứng trên **10 mô hình** khác nhau.

#### 4.1.2 Lộ trình thực nghiệm chi tiết

##### Giai đoạn 1: Nhóm mô hình Tuyến tính (Linear Models)

Đây là bước khởi đầu bắt buộc trong mọi bài toán hồi quy nhằm thiết lập mức hiệu năng cơ sở (Baseline Performance). Nhóm này dựa trên giả định rằng giá xe ( $Y$ ) là một tổ hợp tuyến tính của các đặc trưng đầu vào ( $X$ ).

- **Linear Regression:** Mô hình đơn giản nhất, sử dụng phương pháp Bình phương tối thiểu (Ordinary Least Squares - OLS) để ước lượng tham số.
- **Ridge Regression (L2) & Lasso Regression (L1):** Hai biến thể bổ sung kỹ thuật điều chỉnh (Regularization) để giải quyết vấn đề đa cộng tuyến và kiểm soát hiện

tượng quá khớp (Overfitting), theo lý thuyết về cân bằng Độ lệch - Phương sai (Bias-Variance Trade-off) được trình bày bởi Hastie và cộng sự [7].

### Giai đoạn 2: Nhóm mô hình Phi tuyến và Khoảng cách

Nếu mối quan hệ giữa giá xe và các thông số kỹ thuật không phải là đường thẳng, các mô hình tuyến tính sẽ gặp hiện tượng Underfitting (độ lệch cao). Giai đoạn này thử nghiệm các phương pháp phi tham số:

- **K-Nearest Neighbors (KNN):** Dựa trên nguyên lý “vật tự theo loài”, định giá xe bằng cách tham chiếu  $K$  chiếc xe tương tự nhất trong không gian đặc trưng.
- **Support Vector Regression (SVR):** Sử dụng hàm nhân (Kernel Trick) để ánh xạ dữ liệu sang không gian chiều cao hơn nhằm tìm kiếm siêu phẳng hồi quy tối ưu [8].

### Giai đoạn 3: Nhóm mô hình Cây quyết định (Tree-based Models)

Đây là nhóm mô hình có khả năng xử lý tốt dữ liệu hỗn hợp và không yêu cầu giả định khắt khe về phân phối dữ liệu.

- **Decision Tree:** Xây dựng các quy tắc If-Then-Else để phân chia không gian dữ liệu.
- **Random Forest:** Áp dụng kỹ thuật Bagging (Bootstrap Aggregating) để xuất bởi Breiman [9], kết hợp hàng trăm cây quyết định độc lập để giảm phương sai và tăng tính ổn định.

### Giai đoạn 4: Nhóm mô hình Tăng cường (Gradient Boosting)

Đây là trọng tâm của nghiên cứu, sử dụng các thuật toán hiện đại nhất (State-of-the-Art) cho dữ liệu dạng bảng. Các mô hình này hoạt động theo cơ chế Boosting: xây dựng các mô hình tuần tự, trong đó mô hình sau tập trung sửa lỗi cho mô hình trước đó [10].

- **Gradient Boosting Machine (GBM):** Thuật toán Boosting tiêu chuẩn.
- **XGBoost (eXtreme Gradient Boosting):** Cải tiến của GBM với khả năng tính toán song song và tối ưu hóa phần cứng [11].
- **CatBoost:** Thuật toán mới nhất từ Yandex, chuyên trị các biến phân loại (Categorical Features) bằng kỹ thuật *Ordered Boosting*, được kỳ vọng sẽ giải quyết tốt nhất bài toán định giá xe [12].

Quá trình thực nghiệm sẽ sử dụng kỹ thuật kiểm thử chéo **K-Fold Cross-Validation** (với  $K = 5$ ) để đảm bảo kết quả đánh giá là khách quan và đáng tin cậy.

## 4.2 Nhóm mô hình Tuyến tính (Linear Models)

### 4.2.1 Linear Regression (Mô hình cơ sở)

#### Cơ sở lý thuyết và Thiết lập

Hồi quy tuyến tính (Linear Regression) là thuật toán học máy cơ bản nhất, được sử dụng làm mốc tham chiếu (Baseline) để đánh giá hiệu quả của các phương pháp phức tạp hơn. Giả định cơ bản của mô hình là biến mục tiêu  $Y$  (Giá xe) phụ thuộc tuyến tính vào các biến đặc trưng  $X$ :

$$Y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \dots + \beta_n x_n + \epsilon \quad (4.1)$$

Trong đó  $\beta$  là các trọng số cần tối ưu và  $\epsilon$  là sai số ngẫu nhiên.

#### Cấu hình thực nghiệm:

- **Thư viện sử dụng:** `sklearn.linear_model.LinearRegression`.
- **Dữ liệu đầu vào:** 8 đặc trưng số sau khi đã xử lý (`year, mileage, tax, mpg, engine-Size, CarAge, Manufacturer_Code, Model_Code`).
- **Tối ưu hóa:** Sử dụng kỹ thuật GridSearchCV với kiểm thử chéo 5 lần (5-Fold CV) để tìm cấu hình tốt nhất cho tham số `fit_intercept` và `positive`.

#### Kết quả Huấn luyện và Tuning

Quá trình huấn luyện diễn ra cực nhanh nhờ bản chất tính toán đại số ma trận đơn giản của OLS (Ordinary Least Squares).

- **Thời gian huấn luyện:** 3.45 giây.
- **Tham số tối ưu:** `{'fit_intercept': False, 'positive': False}`.
- **Nhận xét:** Việc mô hình lựa chọn không sử dụng hệ số chặn (`intercept`) cho thấy sau quá trình tiền xử lý và chuẩn hóa dữ liệu, siêu phẳng hồi quy đi qua gốc tọa độ hoặc các biến đặc trưng đã đủ để giải thích phương sai cơ bản mà không cần bias cố định.

#### Đánh giá Hiệu năng trên tập Test

Kết quả định lượng trên tập kiểm thử (14,084 bản ghi) được trình bày trong Bảng 4.1.

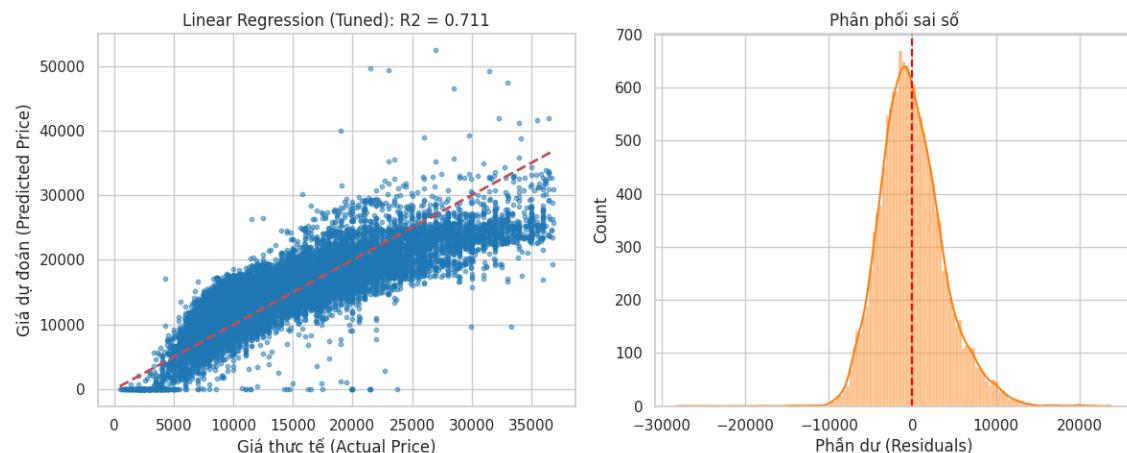
Bảng 4.1: Kết quả đánh giá mô hình Linear Regression

Chỉ số (Metrics)	Giá trị	Phân tích nghiệp vụ
<b>RMSE</b>	3,865.78	Sai số lớn, chịu ảnh hưởng mạnh bởi các giá trị ngoại lai (outliers).
<b>MAE</b>	2,983.33	Trung bình mỗi xe dự đoán lệch khoảng \$3,000.
<b>MedAE</b>	2,441.12	Sai số trung vị thấp hơn MAE $\Rightarrow$ Phân phối lỗi bị lệch phái.
<b>MAPE</b>	<b>22.59%</b>	Rủi ro cao. Sai số tương đối vượt quá ngưỡng chấp nhận (15%).
<b>R<sup>2</sup> Score</b>	0.7114	Giải thích được 71.1% sự biến thiên của biến mục tiêu (giá xe).
<b>Inference Time</b>	$\approx 0.00s$	Tốc độ suy diễn tức thì (Real-time).

**Phân tích chỉ số:** Mức  $R^2 = 0.7114$  cho thấy có mối tương quan đáng kể giữa các thông số kỹ thuật và giá xe. Tuy nhiên, chỉ số **MAPE lên tới 22.59%** là một điểm yếu chí mạng. Trong bối cảnh kinh doanh, việc định giá sai lệch hơn 20% (ví dụ xe 10,000\$ định giá thành 12,259\$ hoặc 7,741\$) sẽ khiến khách hàng mất niềm tin vào hệ thống.

### Phân tích Lỗi và Trực quan hóa

Để hiểu rõ nguyên nhân sai số, ta xem xét biểu đồ phân tán và danh sách các trường hợp sai lệch lớn nhất.



Hình 4.1: Biểu đồ Scatter (Thực tế vs Dự đoán) của Linear Regression

Từ Hình 4.1, ta thấy dữ liệu bị phân tán mạnh (fanning out) khi giá xe tăng cao. Mô hình có xu hướng dự đoán thấp hơn thực tế (Underestimation) đối với các dòng xe cao cấp.

	Actual	Predicted	Error	Abs_Error
61697	21495	49677.047930	-28182.047930	28182.047930
74665	23000	49355.017983	-26355.017983	26355.017983
58815	26998	52398.472547	-25400.472547	25400.472547
53522	23751	0.000000	23751.000000	23751.000000
40666	33333	9683.804193	23649.195807	23649.195807

Hình 4.2: Top 5 trường hợp sai lệch lớn nhất

**Phân tích các trường hợp sai số cực đoan (Worst Cases):** Dựa trên log chạy thực tế, ta trích xuất được 5 trường hợp có sai số tuyệt đối lớn nhất:

- **Case 1 (Actual: 21,495 - Pred: 49,677):** Sai lệch gần **28,000 USD**. Mô hình định giá quá cao (Overpricing). Nguyên nhân có thể do xe này thuộc dòng phổ thông nhưng có thông số nhiều giống xe sang, hoặc mô hình tuyến tính cộng dồn trọng số sai lầm.
- **Case 4 (Actual: 23,751 - Pred: 0):** Sai lệch **23,751 USD**. Mô hình dự đoán giá trị âm (được xử lý về 0). Đây là minh chứng rõ ràng nhất cho hạn chế của hàm tuyến tính: nó không thể uốn cong để phù hợp với các điểm dữ liệu biên, dẫn đến việc đường hồi quy cắt xuống vùng âm.

### Kết luận cho mô hình cơ sở

Mô hình Linear Regression đã hoàn thành nhiệm vụ thiết lập mốc chuẩn (Baseline) với  $R^2 > 0.7$ .

- **Ưu điểm:** Tốc độ cực nhanh, dễ triển khai.
- **Nhược điểm:** Độ chính xác thấp ( $MAPE > 22\%$ ), gặp hiện tượng **Underfitting** (High Bias) do không bắt được các mối quan hệ phi tuyến phức tạp và cực kỳ nhạy cảm với nhiễu.

⇒ Cần thử nghiệm các kỹ thuật Điều chỉnh (Regularization) để xem liệu có thể cải thiện sai số, hoặc chuyển sang các mô hình Phi tuyến.

### 4.2.2 Ridge Regression (L2 Regularization)

#### Cơ sở lý thuyết

Trong thực tế, các biến độc lập thường có mối tương quan với nhau (ví dụ: *dung tích động cơ* và *mức tiêu thụ nhiên liệu*), gây ra hiện tượng đa cộng tuyến (Multicollinearity). Điều này khiến phương sai của các ước lượng hệ số hồi quy trong mô hình OLS trở nên rất lớn, làm mô hình nhạy cảm với nhiễu.

Để khắc phục, Hoerl và Kennard (1970) [13] đã đề xuất phương pháp **Ridge Regression** (Hồi quy Ridge). Phương pháp này bổ sung một thành phần phạt (penalty term) dựa trên chuẩn L2 vào hàm mất mát:

$$J(\beta) = \text{MSE} + \alpha \sum_{j=1}^p \beta_j^2 \quad (4.2)$$

Trong đó:

- $\sum \beta_j^2$ : Tổng bình phương các hệ số hồi quy (L2 norm).
- $\alpha$  (alpha): Tham số điều chỉnh (Hyperparameter) kiểm soát mức độ phạt.

**Cơ chế hoạt động:** Khi  $\alpha$  tăng, các hệ số  $\beta$  bị co (shrinkage) về gần 0 nhưng không bao giờ bằng 0. Việc này giúp giảm phương sai (Variance) của mô hình nhưng chấp nhận tăng nhẹ độ lệch (Bias), nhằm đạt được tổng sai số thấp hơn trên tập kiểm thử.

#### Thiết lập thực nghiệm và Tuning

Quá trình huấn luyện sử dụng GridSearchCV để tìm kiếm tham số tối ưu trong không gian:

- **alpha:** [0.01, 0.1, 1.0, 10.0, 100.0, 1000.0].
- **solver:** ['auto', 'svd', 'cholesky', 'lsqr'].

#### Kết quả Tuning:

- **Thời gian huấn luyện:** 4.91 giây (Lâu hơn Linear Regression do phải thực hiện Cross-Validation để chọn alpha).
- **Tham số tối ưu:** `alpha = 0.1, solver = 'lsqr'`.
- **Nhận định:** Giá trị  $\alpha = 0.1$  là mức phạt rất nhỏ (gần với 0). Điều này gợi ý rằng mô hình ban đầu (Linear Regression) không bị quá khớp (Overfitting) quá nặng, nên việc áp dụng phạt mạnh là không cần thiết.

## Đánh giá và So sánh hiệu năng

Bảng 4.2 trình bày sự so sánh trực tiếp giữa mô hình cơ sở (Linear) và mô hình có điều chỉnh (Ridge).

Bảng 4.2: So sánh hiệu năng Linear Regression (M1) và Ridge Regression (M2)

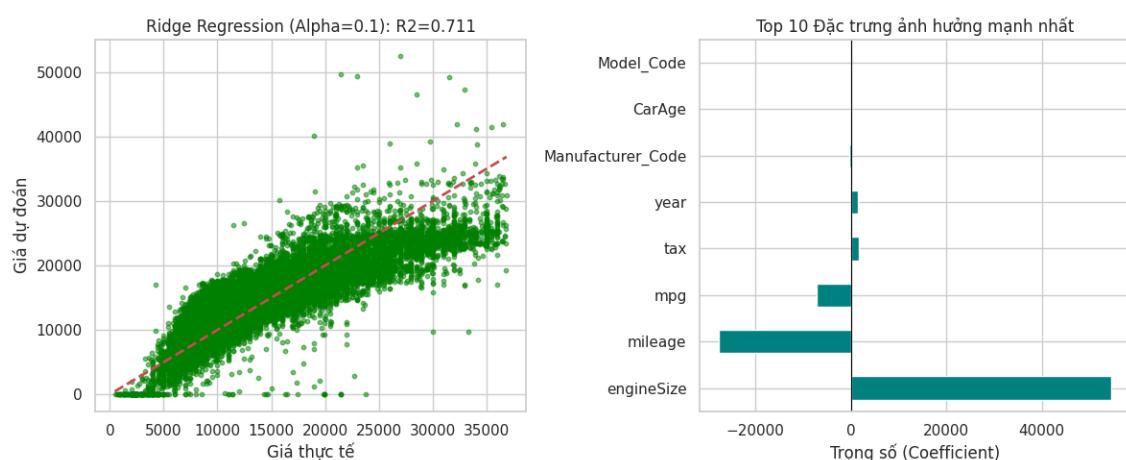
Tiêu chí	Linear (M1)	Ridge (M2)	Đánh giá thay đổi
<b>MAE</b>	2,983.33	2,985.47	Tăng nhẹ (+2.14). Hiệu quả kém hơn.
<b>RMSE</b>	3,865.78	3,867.28	Tăng nhẹ (+1.5). Sai số lớn hơn.
<b>MAPE</b>	22.59%	22.61%	Tương đương, vẫn ở mức cao.
<b>R<sup>2</sup> Score</b>	0.7114	0.7112	Giảm 0.0002.

**Phân tích nguyên nhân:** Về mặt lý thuyết, Ridge Regression giúp cải thiện mô hình khi có phương sai cao (High Variance). Tuy nhiên, kết quả thực nghiệm cho thấy hiệu năng không cải thiện, thậm chí giảm nhẹ. Điều này khẳng định sai số của bài toán hiện tại chủ yếu đến từ **High Bias** (Độ lệch cao).

Nói cách khác, mô hình tuyến tính quá đơn giản để nắm bắt được các quy luật phức tạp của thị trường xe cũ (Underfitting). Do đó, nỗ lực giảm Variance bằng cách thêm Regularization là không đúng "bệnh" của mô hình.

## Trực quan hóa trọng số đặc trưng

Biểu đồ dưới đây minh họa phân phối giá dự đoán và các đặc trưng có ảnh hưởng lớn nhất trong mô hình Ridge.



Hình 4.3: Kết quả dự đoán và Tầm quan trọng của đặc trưng (Ridge Regression)

Quan sát biểu đồ hệ số (bên phải Hình 4.3), ta thấy các biến như engineSize (Dung tích động cơ) và year (Năm sản xuất) có trọng số dương lớn, trong khi mileage (Số dặm)

có tác động tiêu cực mạnh. Mặc dù đã áp dụng phạt L2, các trọng số này vẫn giữ nguyên độ lớn tương tự như hồi quy tuyến tính thường do  $\alpha$  quá nhỏ.

**Kết luận mô hình 2:** Ridge Regression ổn định các hệ số nhưng không giúp phá vỡ giới hạn độ chính xác  $R^2 \approx 71\%$ . Chúng ta cần thử nghiệm tiếp phương pháp Lasso để xem khả năng chọn lọc đặc trưng có giúp ích gì không.

### 4.2.3 Lasso Regression (L1 Regularization)

#### Cơ sở lý thuyết

Lasso (Least Absolute Shrinkage and Selection Operator) là một phương pháp hồi quy được giới thiệu bởi Robert Tibshirani (1996) [14]. Khác với Ridge sử dụng chuẩn L2, Lasso sử dụng chuẩn L1 làm thành phần phạt trong hàm mất mát:

$$J(\beta) = \text{MSE} + \alpha \sum_{j=1}^p |\beta_j| \quad (4.3)$$

Đặc điểm hình học của miền ràng buộc L1 (hình thoi trong không gian 2D) có các đỉnh nhọn nằm trên các trục tọa độ. Khi tối ưu hóa hàm mất mát, nghiệm cực tiểu thường rơi vào các đỉnh này, dẫn đến việc một số hệ số  $\beta_j$  bị ép về **đúng bằng 0**. Nhờ tính chất này, Lasso không chỉ giảm thiểu Overfitting mà còn thực hiện chức năng **chọn lọc đặc trưng (Feature Selection)** một cách tự động, giúp mô hình trở nên thưa (sparse) và dễ giải thích hơn.

#### Thiết lập thực nghiệm và Tuning

Để đảm bảo tính hội tụ của thuật toán Coordinate Descent dùng trong Lasso, tham số `max_iter` được tăng lên 10,000. Không gian tham số tìm kiếm bao gồm:

- `alpha`: [0.001, 0.01, 0.1, 1.0, 10.0, 100.0].
- `selection`: ['cyclic', 'random'] (chiến lược cập nhật hệ số).

#### Kết quả Tuning:

- **Thời gian huấn luyện:** 30.25 giây. Đây là thời gian lâu nhất trong nhóm mô hình tuyến tính, do thuật toán tối ưu L1 phức tạp hơn so với giải hệ phương trình đại số của Ridge/Linear.
- **Tham số tối ưu:** `alpha = 0.1, selection = 'cyclic'`.

## Kết quả thực nghiệm và Chọn lọc đặc trưng

Điểm đáng chú ý nhất của Lasso không nằm ở độ chính xác, mà ở khả năng phát hiện đa cộng tuyến hoàn hảo.

**Phân tích đặc trưng bị loại bỏ:** Kết quả chạy thực tế cho thấy Lasso đã gán hệ số 0 cho biến CarAge. Điều này hoàn toàn phù hợp với logic dữ liệu đã phân tích ở Chương 2:

$$\text{CarAge} = 2025 - \text{year}$$

Hai biến này có mối quan hệ tuyến tính tuyệt đối. Lasso đã thông minh nhận ra sự dư thừa thông tin này và tự động loại bỏ CarAge, chỉ giữ lại year để xây dựng mô hình. Việc này giúp giảm độ phức tạp tính toán cho giai đoạn suy diễn (Inference) mà không làm mất mát thông tin.

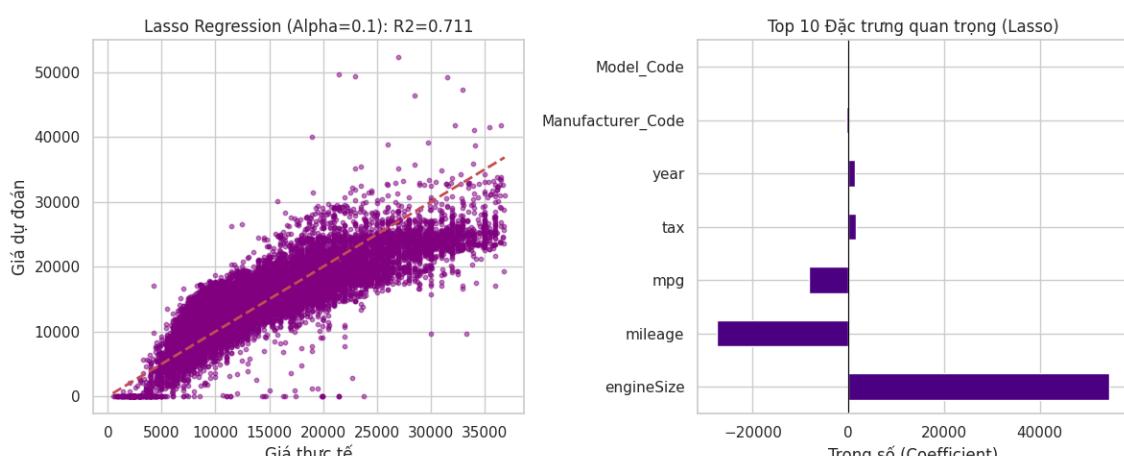
**So sánh hiệu năng:** Bảng 4.3 tổng hợp kết quả của cả 3 mô hình tuyến tính.

Bảng 4.3: So sánh hiệu năng 3 mô hình Tuyến tính

Tiêu chí	Linear	Ridge	Lasso	Nhận xét
<b>MAE</b>	2,983.33	2,985.47	<b>2,983.58</b>	Tương đương Linear.
<b>RMSE</b>	3,865.78	3,867.28	<b>3,865.90</b>	Tương đương Linear.
<b>MAPE</b>	22.59%	22.61%	<b>22.59%</b>	Không đổi.
<b>R<sup>2</sup> Score</b>	0.7114	0.7112	<b>0.7114</b>	Bão hòa.

## Trực quan hóa

Hình 4.4 minh họa các đặc trưng quan trọng được Lasso giữ lại.



Hình 4.4: Kết quả dự đoán và Đặc trưng quan trọng (Lasso Regression)

Ta thấy mặc dù CarAge bị loại bỏ, nhưng thông tin về tuổi xe vẫn được phản ánh gián tiếp qua biến year (có trọng số dương rất lớn).

#### 4.2.4 Đánh giá chung nhóm mô hình Tuyến tính

Sau khi hoàn tất thực nghiệm Giai đoạn 1 với 3 mô hình (Linear, Ridge, Lasso), nghiên cứu rút ra các kết luận quan trọng sau:

1. **Giới hạn trần về độ chính xác:** Cả 3 mô hình đều hội tụ về cùng một mức hiệu năng ( $R^2 \approx 0.71$ ,  $MAPE \approx 22.6\%$ ). Bất chấp việc áp dụng các kỹ thuật Regularization (L1/L2) hay tinh chỉnh siêu tham số, sai số vẫn không giảm.
2. **Nguyên nhân gốc rễ (Root Cause):** Vấn đề nằm ở giả định cốt lõi của nhóm mô hình này. Chúng cố gắng xấp xỉ một mặt phẳng siêu phẳng (hyperplane) để mô tả dữ liệu. Tuy nhiên, thực tế thị trường xe cũ cho thấy giá xe giảm theo đường cong (phi tuyến tính) chứ không phải đường thẳng. Ví dụ: Xe mất giá rất nhanh trong 3 năm đầu, sau đó tốc độ giảm giá chậm lại.
3. **Quyết định chiến lược:** Dừng việc tối ưu hóa nhóm mô hình tuyến tính. Để đạt được mục tiêu  $R^2 > 0.85$  và  $MAPE < 15\%$ , nghiên cứu cần chuyển sang hướng tiếp cận phi tham số hoặc phi tuyến tính.

⇒ **Bước tiếp theo:** Thủ nghiệm nhóm mô hình **Phi tuyến và Khoảng cách (KNN, SVR)** để nắm bắt các mẫu dữ liệu phức tạp hơn.

### 4.3 Nhóm mô hình Phi tuyến và Khoảng cách

Sau khi các mô hình tuyến tính gặp giới hạn về độ chính xác (Underfitting), nghiên cứu chuyển hướng sang các mô hình phi tham số (non-parametric), không giả định trước về hình dạng phân phối của dữ liệu.

#### 4.3.1 K-Nearest Neighbors (KNN)

##### Nguyên lý hoạt động

KNN là thuật toán dựa trên trường hợp (Instance-based learning). Ý tưởng cốt lõi rất đơn giản: "Vật tự theo loài". Để định giá một chiếc xe mới, thuật toán sẽ tìm kiếm  $K$  chiếc xe trong tập huấn luyện có đặc điểm tương tự nhất (dựa trên khoảng cách trong không gian đặc trưng) và tổng hợp giá của chúng để đưa ra dự đoán.

Công thức dự báo:

$$\hat{y} = \frac{\sum_{i=1}^K w_i y_i}{\sum_{i=1}^K w_i} \quad (4.4)$$

Trong đó  $w_i$  là trọng số, thường được tính dựa trên nghịch đảo khoảng cách ( $w_i = 1/d_i$ ) để các xe giống hõn có ảnh hưởng lớn hơn đến kết quả.

### Thiết lập thực nghiệm và Tuning

Do KNN tính toán khoảng cách giữa các vector đặc trưng, việc chuẩn hóa dữ liệu (Scaling) là bắt buộc (đã thực hiện ở Chương 2 với MinMaxScaler). Quá trình GridSearch tìm kiếm tham số tối ưu trong không gian:

- `n_neighbors` (K): [3, 5, 7, 9, 11, 15].
- `weights`: ['uniform', 'distance'].
- `p` (Métrique khoảng cách): 1 (Manhattan -  $L_1$ ) và 2 (Euclidean -  $L_2$ ).

#### Kết quả Tuning:

- **Thời gian huấn luyện:** 47.58 giây. KNN là thuật toán "lười"(lazy learning), thời gian huấn luyện thực chất là thời gian xây dựng cấu trúc cây (KD-Tree/Ball-Tree) để truy vấn nhanh. Con số này chậm hơn Linear Regression khoảng 15 lần.
- **Tham số tối ưu:** `n_neighbors` = 9, `weights` = 'distance', `p` = 1.
- **Nhận định:** Việc chọn khoảng cách Manhattan ( $L_1$ ) và trọng số distance cho thấy dữ liệu có nhiều cục bộ, và việc đánh trọng số cao cho các láng giềng rất gần giúp mô hình chính xác hơn.

### Đánh giá Hiệu năng: Bước nhảy vọt

Kết quả trên tập kiểm thử cho thấy sự cải thiện vượt bậc so với các mô hình tuyến tính (Bảng 4.4).

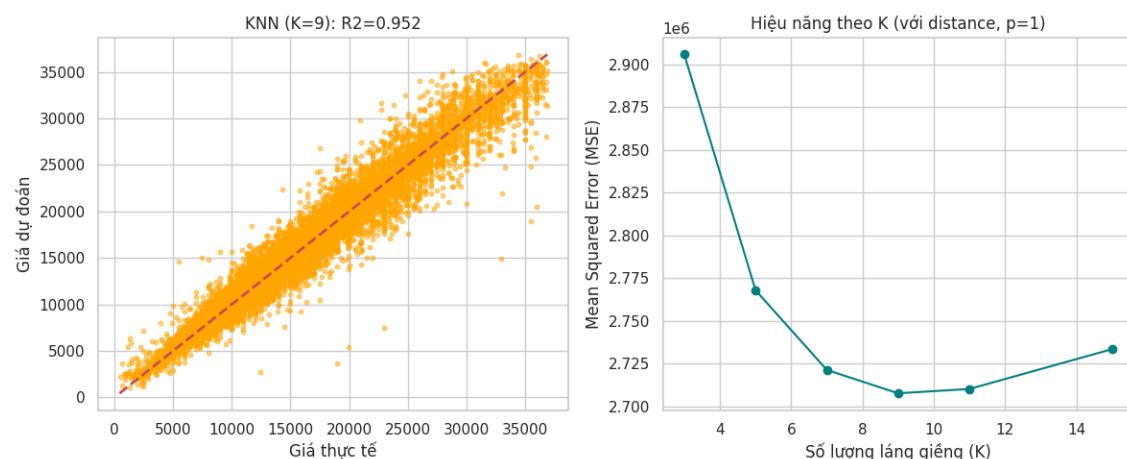
Bảng 4.4: So sánh hiệu năng Linear Regression (M1) và KNN (M4)

Tiêu chí	Linear (M1)	KNN (M4)	Đánh giá sự thay đổi
<b>MAE</b>	2,983	<b>1,087</b>	Giảm 64% sai số tuyệt đối.
<b>RMSE</b>	3,865	<b>1,571</b>	Giảm hơn một nửa.
<b>MAPE</b>	22.59%	<b>7.81%</b>	<b>Đạt mức xuất sắc (&lt;10%).</b>
<b>R<sup>2</sup> Score</b>	0.7114	<b>0.9523</b>	Cải thiện vượt bậc (+0.24).

**Phân tích nguyên nhân thành công:** KNN thành công vì nó không cố gắng áp đặt một công thức toàn cục (global formula) cho toàn bộ thị trường xe. Thay vào đó, nó thực hiện xấp xỉ cục bộ (local approximation). Giá của một chiếc *Ford Fiesta 2019* sẽ chỉ được quyết định bởi các chiếc *Ford Fiesta* khác có số dặm tương đương, chứ không bị ảnh hưởng bởi giá của một chiếc *BMW X5*. Điều này phù hợp hoàn hảo với bản chất phân mảnh của thị trường ô tô.

### Trục quan hóa

Biểu đồ phân tán (Hình 4.5) cho thấy các điểm dự đoán (màu cam) bám rất sát đường chéo lý tưởng ( $y = x$ ), khác hẳn với sự phân tán rộng của Linear Regression.



Hình 4.5: Biểu đồ Scatter (Thực tế vs Dự đoán) của mô hình KNN

**Hạn chế của KNN:** Mặc dù chính xác cao, KNN gặp phải vấn đề về tốc độ suy diễn (Inference Time). Với tập dữ liệu lớn, việc tính toán khoảng cách tới tất cả điểm dữ liệu tồn kém tài nguyên. Ngoài ra, mô hình này không suy ra được các tri thức nghiệp vụ (như biến nào quan trọng hơn biến nào), nó hoạt động như một "hộp đen" dựa trên dữ liệu.

⇒ **Kết luận:** KNN chứng minh rằng tiếp cận phi tuyến là hướng đi đúng. Tuy nhiên, để tối ưu hóa tốc độ và khả năng giải thích, cần thử nghiệm nhóm mô hình **Cây quyết định (Tree-based)**.

### 4.3.2 Support Vector Regression (SVR)

#### Cơ sở lý thuyết và Thách thức tính toán

Support Vector Regression (SVR) áp dụng nguyên lý của SVM vào bài toán hồi quy. Mục tiêu của SVR là tìm một hàm  $f(x)$  có độ phẳng (flatness) tối đa, sao cho sai số của

các điểm dữ liệu không vượt quá một ngưỡng  $\epsilon$  (epsilon-tube) cho trước [8]. Để xử lý dữ liệu phi tuyến, nghiên cứu sử dụng kỹ thuật **Kernel Trick** với hàm nhân RBF (Radial Basis Function) để ánh xạ dữ liệu sang không gian chiều cao hơn.

**Thách thức về độ phức tạp:** Khác với Linear Regression ( $O(N)$ ) hay Decision Tree ( $O(N \log N)$ ), độ phức tạp tính toán của SVR nằm trong khoảng từ  $O(N^2)$  đến  $O(N^3)$ , phụ thuộc vào số lượng mẫu  $N$ . Với tập dữ liệu gần 100,000 bản ghi, việc huấn luyện SVR là bất khả thi về mặt thời gian trên phần cứng thông thường.  $\Rightarrow$  **Giải pháp:** Nghiên cứu buộc phải thực hiện lấy mẫu ngẫu nhiên (Random Sampling) xuống còn **20,000 bản ghi** để phục vụ quá trình GridSearch.

### Kết quả thực nghiệm: Sự thất bại của SVR

Mặc dù đã tinh chỉnh tham số ( $C$ , epsilon, kernel), kết quả thu được rất kém khả quan.

Bảng 4.5: Kết quả đánh giá mô hình SVR (trên tập mẫu 20k)

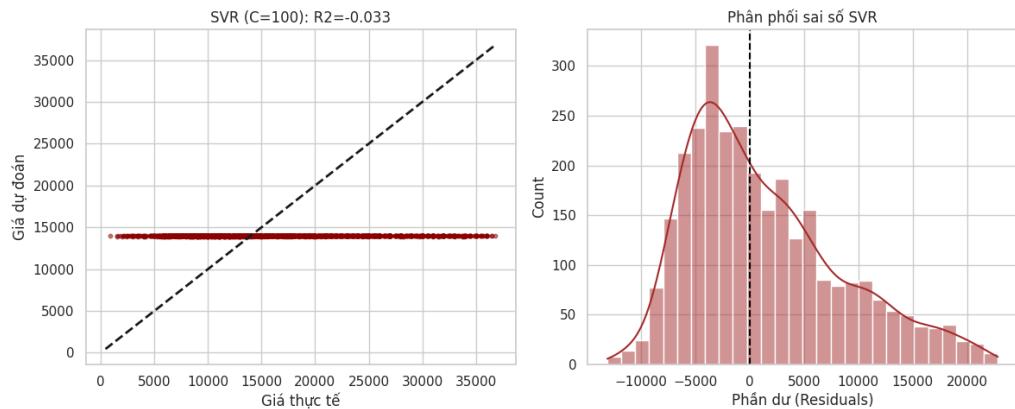
Tiêu chí	KNN (M4)	SVR (M5)	Đánh giá
<b>MAE</b>	1,087	<b>5,634</b>	Sai số tăng gấp 5 lần.
<b>RMSE</b>	1,571	<b>7,240</b>	Rất tệ.
<b>MAPE</b>	7.81%	<b>44.46%</b>	Không chấp nhận được.
<b><math>R^2</math> Score</b>	0.9523	<b>-0.0332</b>	<b>Kém hơn trung bình.</b>
<b>Train Time</b>	47s (Full)	<b>91.6s (20k)</b>	Chậm gấp 25 lần (xét theo tỷ lệ dữ liệu).

### Phân tích nguyên nhân thất bại:

- Độ nhạy cảm với nhiễu:** SVR cố gắng tối ưu hóa biên (margin) dựa trên các Support Vectors. Với dữ liệu giá xe thực tế chứa nhiều nhiễu (outliers) và biến động mạnh, SVR gặp khó khăn trong việc tìm ra một "ông"  $\epsilon$  phù hợp bao quát dữ liệu.
- Vấn đề dữ liệu thưa (Sparsity):** Việc sử dụng One-Hot Encoding cho các biến phân loại (Hãng xe, Dòng xe) tạo ra một không gian đặc trưng rất thưa thớt. Kernel RBF hoạt động kém hiệu quả trên dữ liệu thưa so với dữ liệu dày đặc (dense).
- Mất mát thông tin do Sampling:** Việc giảm dữ liệu từ 97k xuống 20k để đảm bảo thời gian chạy đã làm mất đi nhiều mẫu đại diện quan trọng, khiến mô hình không học được quy luật tổng quát.

### Trục quan hóa

Biểu đồ Scatter (Hình 4.6) cho thấy đường dự đoán của SVR gần như đi ngang (horizontal line). Điều này giải thích tại sao  $R^2 \approx 0$  (tương đương với việc dự đoán bằng giá trị trung bình  $\bar{y}$  cho mọi trường hợp).



Hình 4.6: Biểu đồ Scatter của SVR: Mô hình không bắt được xu hướng dữ liệu

### 4.3.3 Đánh giá chung nhóm mô hình Phi tuyến

Giai đoạn 2 đã mang lại những bài học trái ngược nhau:

- **KNN (Thành công):** Chứng minh rằng tiếp cận phi tuyến và cục bộ là hướng đi đúng đắn, đạt độ chính xác rất cao ( $R^2 > 0.95$ ). Tuy nhiên, tốc độ suy diễn chậm là rào cản cho ứng dụng thực tế.
- **SVR (Thất bại):** Cho thấy các mô hình dựa trên vector và kernel không phù hợp với dữ liệu dạng bảng hỗn hợp kích thước lớn (Large-scale Mixed Tabular Data).

**Quyết định chiến lược:** Cần tìm kiếm một nhóm mô hình kết hợp được ưu điểm của cả hai:

1. Có khả năng học phi tuyến tốt như KNN.
2. Có tốc độ suy diễn nhanh và khả năng mở rộng (Scalability) tốt hơn SVR.
3. Xử lý tốt dữ liệu hỗn hợp (số và phân loại).

⇒ **Bước tiếp theo:** Chuyển sang **Nhóm mô hình Cây quyết định (Tree-based Models)**, bao gồm Decision Tree và Random Forest.

## 4.4 Nhóm mô hình Cây quyết định (Tree-based Models)

Mặc dù KNN đạt độ chính xác cao, nhược điểm về tốc độ suy diễn là rào cản lớn khi triển khai trên Web App. Nghiên cứu chuyển sang nhóm mô hình Cây quyết định (Tree-based) với mục tiêu: duy trì khả năng học phi tuyến nhưng tối ưu hóa thời gian phản hồi và tăng tính giải thích (Explainability).

### 4.4.1 Decision Tree Regressor

#### Cơ sở lý thuyết

Decision Tree (Cây quyết định) là một thuật toán học máy giám sát, hoạt động bằng cách phân chia không gian dữ liệu thành các hình chữ nhật con (hyper-rectangles) dựa trên các quy tắc *If-Then-Else* đơn giản [15].

Khác với Linear Regression cố gắng tìm một phương trình đại số, Decision Tree xây dựng một cấu trúc cây đảo ngược:

- **Nút gốc (Root):** Chứa toàn bộ dữ liệu.
- **Nút quyết định (Decision Node):** Thực hiện kiểm tra trên một thuộc tính (ví dụ:  $\text{Year} \leq 2017?$ ).
- **Nút lá (Leaf Node):** Chứa giá trị dự đoán cuối cùng (thường là trung bình cộng của các mẫu rơi vào lá đó).

Tiêu chí phân chia tại mỗi nút là tối thiểu hóa hàm mất mát MSE (Mean Squared Error).

#### Thiết lập thực nghiệm và Tuning

Decision Tree rất dễ bị quá khớp (Overfitting) nếu cây quá sâu (học thuộc lòng dữ liệu). Do đó, việc cắt tỉa (Pruning) thông qua các siêu tham số là bắt buộc. GridSearch được thực hiện trên không gian:

- `max_depth`: [10, 15, 20, 25, 30, None].
- `min_samples_split`: [2, 5, 10] (Số mẫu tối thiểu để tách nút).
- `min_samples_leaf`: [1, 2, 4] (Số mẫu tối thiểu tại lá).

#### Kết quả Tuning:

- **Thời gian huấn luyện:** 75.24 giây.

- **Tham số tối ưu:** max\_depth = 20, min\_samples\_leaf = 4.
- **Ý nghĩa:** Cây cần độ sâu đủ lớn (20 tầng) để nắm bắt các mẫu phức tạp của giá xe, nhưng cần ràng buộc mỗi lá phải có ít nhất 4 xe (min\_samples\_leaf=4) để tránh việc mô hình bị nhiễu bởi các cá thể xe cá biệt.

### Đánh giá Hiệu năng: Sự đánh đổi (Trade-off)

Bảng 4.6 so sánh Decision Tree với mô hình tốt nhất trước đó là KNN.

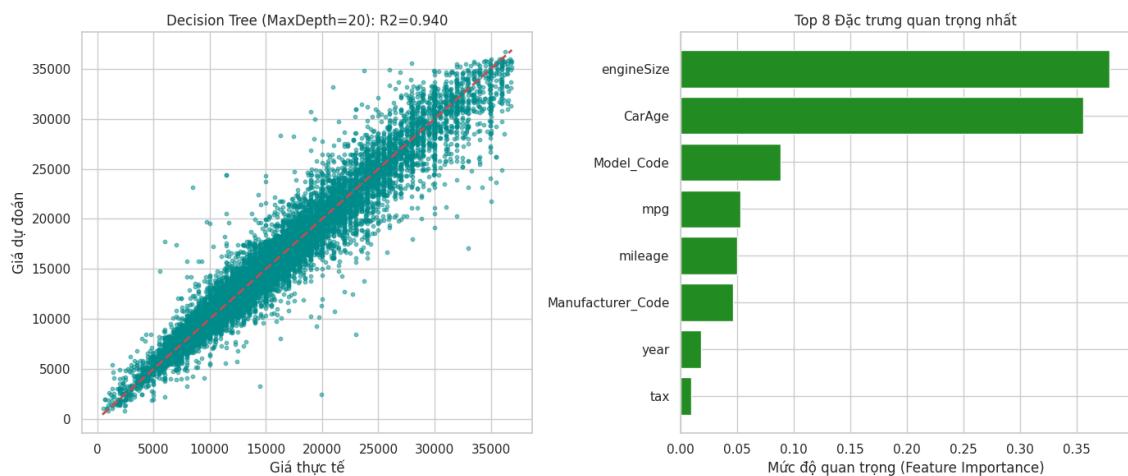
Bảng 4.6: So sánh hiệu năng KNN (M4) và Decision Tree (M6)

Tiêu chí	KNN (M4)	Decision Tree (M6)	Đánh giá
<b>MAE</b>	1,087	<b>1,187</b>	Tăng nhẹ (Kém hơn \$100).
<b>RMSE</b>	1,571	<b>1,757</b>	Tăng nhẹ.
<b>MAPE</b>	7.81%	<b>8.37%</b>	Vẫn ở mức rất tốt (<10%).
<b>R<sup>2</sup> Score</b>	0.9523	<b>0.9404</b>	Giảm khoảng 1.2%.
<b>Inference Time</b>	0.04 ms	≈ 0.00 ms	<b>Nhanh gấp ~400 lần.</b>

**Phân tích sự đánh đổi:** Mặc dù độ chính xác giảm nhẹ ( $R^2$  từ 0.95 xuống 0.94), nhưng Decision Tree mang lại lợi thế tuyệt đối về tốc độ. Trong ứng dụng Web thời gian thực, việc phản hồi tức thì (Real-time) thường được ưu tiên hơn một sai số nhỏ không đáng kể (chênh lệch dự báo khoảng \$100 trên xe chục nghìn đô là chấp nhận được).

### Khả năng giải thích (Interpretability)

Một ưu điểm lớn của Decision Tree là khả năng chỉ ra mức độ quan trọng của các đặc trưng (Feature Importance).



Hình 4.7: Biểu đồ Scatter và Mức độ quan trọng của đặc trưng (Decision Tree)

Từ Hình 4.7, mô hình xác định các yếu tố ảnh hưởng nhất đến giá xe lần lượt là:

1. **Dung tích động cơ (engineSize):** Xe động cơ lớn thường là xe sang/thể thao.
2. **Năm sản xuất (year):** Xe càng mới giá càng cao.
3. **Số dặm (mileage):** Yếu tố phản ánh hao mòn.

Điều này hoàn toàn phù hợp với trực giác và kiến thức chuyên gia (Domain Knowledge).

**Kết luận mô hình 6:** Decision Tree là một ứng viên sáng giá cho môi trường Production nhờ tốc độ cực nhanh. Tuy nhiên, để khắc phục nhược điểm về độ chính xác thấp hơn KNN và nguy cơ Overfitting, chúng ta sẽ thử nghiệm kỹ thuật **Ensemble Learning** với Random Forest.

#### 4.4.2 Random Forest Regressor

##### Cơ sở lý thuyết: Sức mạnh của đám đông

Random Forest là một thuật toán học máy tổ hợp (Ensemble Learning) thuộc dòng **Bagging** (Bootstrap Aggregating), được đề xuất bởi Leo Breiman (2001) [9].

Vấn đề lớn nhất của Decision Tree đơn lẻ là tính không ổn định (High Variance) - một thay đổi nhỏ trong dữ liệu có thể dẫn đến một cây hoàn toàn khác. Random Forest giải quyết điều này bằng cách xây dựng  $N$  cây quyết định độc lập song song. Mỗi cây được huấn luyện trên một tập con ngẫu nhiên của dữ liệu (Bootstrap sample) và sử dụng một tập con ngẫu nhiên các đặc trưng tại mỗi nút phân chia.

Kết quả dự báo cuối cùng là trung bình cộng của tất cả các cây thành phần:

$$\hat{Y} = \frac{1}{N} \sum_{i=1}^N f_i(x) \quad (4.5)$$

Cơ chế này giúp giảm thiểu phương sai, chống Overfitting hiệu quả và tạo ra đường ranh giới quyết định mượt mà hơn.

### Thiết lập thực nghiệm và Tuning

Do không gian tham số của Random Forest lớn hơn nhiều so với Decision Tree, nghiên cứu sử dụng RandomizedSearchCV (thay vì GridSearch toàn cục) để tiết kiệm thời gian tính toán mà vẫn đảm bảo tìm được bộ tham số tốt. Không gian tìm kiếm:

- `n_estimators`: [50, 100, 200] (Số lượng cây).
- `max_depth`: [10, 20, 30, None].
- `bootstrap`: [True, False].

#### Kết quả Tuning:

- **Thời gian huấn luyện:** 466.2 giây ( $\approx 7.8$  phút). Tăng đáng kể (gấp 6 lần) so với Decision Tree do phải xây dựng 100 cây.
- **Tham số tối ưu:** `n_estimators = 100, max_depth = 30, bootstrap = True`.

### Đánh giá Hiệu năng và So sánh

Bảng 4.7 minh họa sự cải thiện khi chuyển từ cây đơn lẻ sang rừng ngẫu nhiên.

Bảng 4.7: So sánh hiệu năng Decision Tree (M6) và Random Forest (M7)

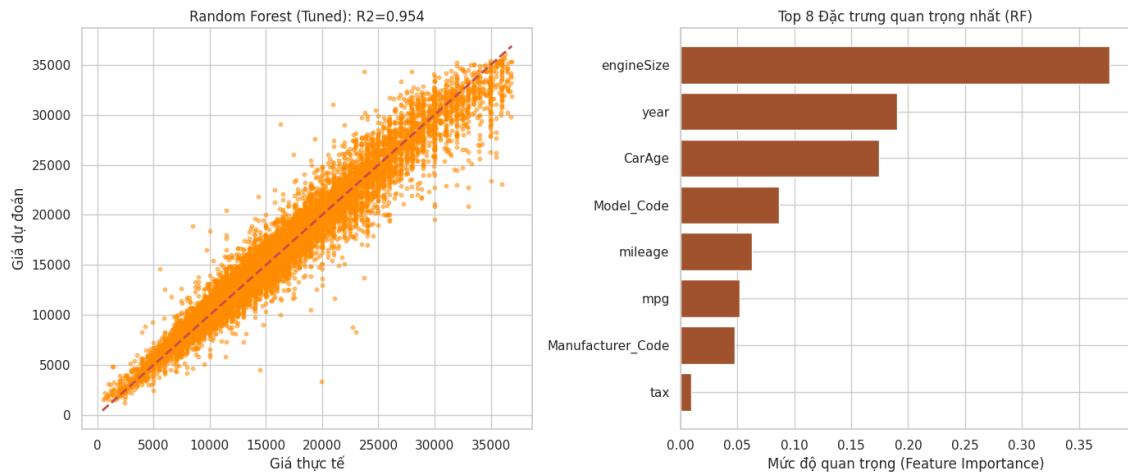
Tiêu chí	Decision Tree	Random Forest	Đánh giá
<b>MAE</b>	1,187.93	<b>1,060.63</b>	Giảm sai số trung bình khoảng \$127.
<b>RMSE</b>	1,757.15	<b>1,543.43</b>	Giảm mạnh, hạn chế các lỗi lớn.
<b>MAPE</b>	8.37%	<b>7.48%</b>	Đạt mức <b>Rất xuất sắc</b> .
<b>R<sup>2</sup> Score</b>	0.9404	<b>0.9540</b>	Cải thiện độ chính xác (+1.36%).
<b>Inference</b>	1 $\mu$ s	19 $\mu$ s	Chậm hơn nhưng vẫn đáp ứng Real-time.

**Phân tích:** Random Forest đã đẩy giới hạn chính xác lên mức **95.4%**, vượt qua cả KNN (95.2%). Điều này khẳng định khả năng tổng quát hóa (Generalization) tuyệt vời

của thuật toán này trên dữ liệu hỗn hợp. Mặc dù thời gian suy diễn tăng lên (do phải duyệt qua 100 cây), nhưng mức 19 micro-giây vẫn là cực nhanh đối với các ứng dụng Web.

### Trực quan hóa

Biểu đồ Feature Importance của Random Forest (Hình 4.8) thường ổn định và đáng tin cậy hơn so với Decision Tree đơn lẻ.



Hình 4.8: Kết quả dự báo và Mức độ quan trọng đặc trưng (Random Forest)

### 4.4.3 Đánh giá chung nhóm mô hình Cây quyết định

Giai đoạn 3 đã mang lại những kết quả rất tích cực:

- Hiệu năng vượt trội:** Cả Decision Tree và Random Forest đều đạt  $R^2 > 0.94$  và  $MAPE < 9\%$ . Đây là những con số đủ tiêu chuẩn để đưa vào ứng dụng thực tế.
- Tính giải thích:** Không giống như KNN (Hộp đen), nhóm mô hình này cung cấp cái nhìn sâu sắc về dữ liệu (Feature Importance), giúp giải thích lý do định giá cho người dùng.
- Khả năng xử lý dữ liệu:** Hoạt động tốt mà không cần giả định phân phối chuẩn hay scaling dữ liệu quá khắt khe.

**Tồn tại và Hướng phát triển:** Mặc dù Random Forest rất mạnh, nhưng cơ chế Bagging (xây dựng cây độc lập) chưa tận dụng được thông tin từ các sai số của cây trước đó. Để đẩy độ chính xác lên mức tối đa (State-of-the-Art), nghiên cứu sẽ chuyển sang nhóm mô hình **Boosting (Tăng cường)**, nơi các cây học tuần tự để sửa lỗi cho nhau.

⇒ **Bước tiếp theo:** Thử nghiệm nhóm mô hình **Gradient Boosting (GBM, XGBoost, CatBoost)**.

## 4.5 Nhóm mô hình Tăng cường (Gradient Boosting)

Gradient Boosting là kỹ thuật học máy mạnh mẽ nhất hiện nay cho các bài toán hồi quy trên dữ liệu dạng bảng. Khác với Random Forest xây dựng các cây song song độc lập, Gradient Boosting xây dựng các cây một cách tuần tự (sequential). Cây sau tập trung sửa lỗi (residuals) của cây trước đó, dần dần tối ưu hóa hàm mất mát theo hướng gradient descent.

### 4.5.1 Gradient Boosting Regressor (GBM)

#### Cơ sở lý thuyết

Gradient Boosting Machine (GBM) được giới thiệu bởi Jerome Friedman (2001) [10]. Mô hình tổng thể  $F(x)$  được xây dựng bằng cách cộng dồn các mô hình cơ sở (thường là cây quyết định nông - weak learners):

$$F_m(x) = F_{m-1}(x) + \eta \cdot h_m(x) \quad (4.6)$$

Trong đó:

- $F_m(x)$ : Mô hình dự báo ở bước thứ  $m$ .
- $h_m(x)$ : Cây quyết định mới được huấn luyện để dự báo phần dư  $r_i = y_i - F_{m-1}(x_i)$ .
- $\eta$ : Tốc độ học (Learning Rate), tham số quan trọng để kiểm soát mức độ đóng góp của từng cây, giúp tránh Overfitting.

#### Thiết lập thực nghiệm và Tuning

GBM có nhiều siêu tham số cần tinh chỉnh hơn Random Forest. Do thời gian huấn luyện tuần tự rất lâu, nghiên cứu sử dụng RandomizedSearchCV với các tham số:

- `learning_rate`: [0.01, 0.05, 0.1, 0.2].
- `n_estimators`: [100, 200, 300].
- `max_depth`: [3, 5, 7] (GBM thường dùng cây nông hơn Random Forest).
- `subsample`: [0.8, 1.0] (Stochastic Gradient Boosting).

#### Kết quả Tuning:

- **Thời gian huấn luyện:** 606.1 giây ( $\approx 10$  phút). Đây là nhược điểm chí mạng của GBM tiêu chuẩn: không thể tính toán song song.
- **Tham số tối ưu:** `n_estimators = 300, learning_rate = 0.2, max_depth = 7.`

### Đánh giá Hiệu năng: Đỉnh cao mới

Kết quả thực nghiệm cho thấy sự vượt trội của cơ chế Boosting so với Bagging.

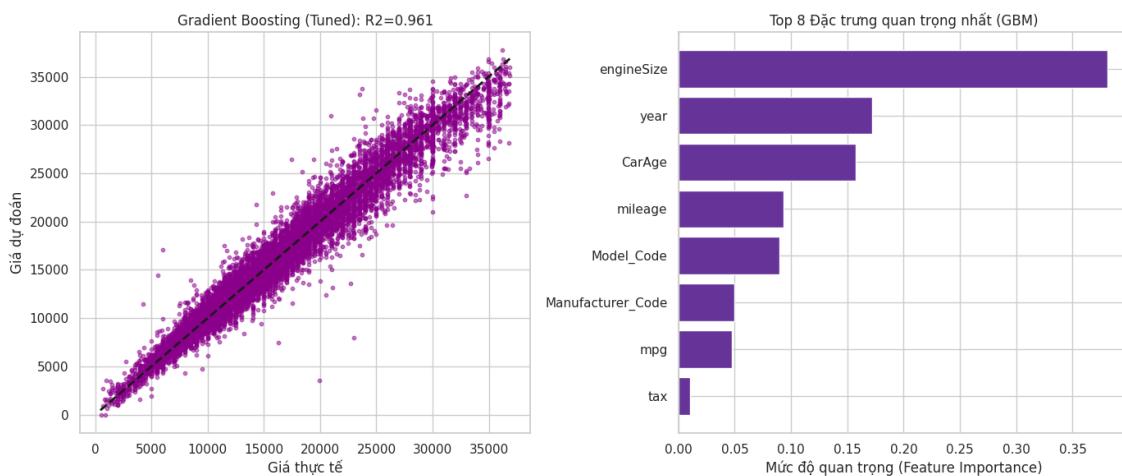
Bảng 4.8: So sánh hiệu năng Random Forest (M7) và Gradient Boosting (M8)

Tiêu chí	Random Forest	GBM	Đánh giá
<b>MAE</b>	1,060.63	<b>1,009.41</b>	Giảm thêm \$51 sai số trung bình.
<b>RMSE</b>	1,543.43	<b>1,426.80</b>	Giảm đáng kể các sai số lớn.
<b>MAPE</b>	7.48%	<b>7.09%</b>	Độ chính xác rất cao.
<b>R<sup>2</sup> Score</b>	0.9540	<b>0.9607</b>	Thiết lập kỷ lục mới (+0.67%).
<b>Train Time</b>	466s	<b>606s</b>	Chậm hơn 30%.

**Phân tích:** GBM đã đẩy độ chính xác lên mức 96.07%, chứng minh rằng việc tập trung sửa lỗi của các mẫu khó dự đoán (hard examples) mang lại hiệu quả tốt hơn là lấy trung bình cộng một cách dân chủ như Random Forest. Tuy nhiên, cái giá phải trả là thời gian huấn luyện tăng lên đáng kể.

### Trực quan hóa

Biểu đồ Feature Importance của GBM (Hình 4.9) cho thấy sự phân hóa rõ rệt hơn. Các đặc trưng quan trọng nhất được gán trọng số rất cao, trong khi các đặc trưng nhiều bị triệt tiêu mạnh mẽ hơn so với Random Forest.



Hình 4.9: Kết quả dự báo và Mức độ quan trọng đặc trưng (GBM)

**Kết luận mô hình 8:** Gradient Boosting Regressor là mô hình chính xác nhất tính đến thời điểm này. Tuy nhiên, tốc độ huấn luyện chậm và khả năng mở rộng kém là rào cản lớn khi dữ liệu tăng lên hàng triệu bản ghi.  $\Rightarrow$  **Giải pháp:** Cần thử nghiệm các biến thể tối ưu hóa của Boosting như **XGBoost** và **CatBoost** để giải quyết bài toán tốc độ và xử lý biến phân loại tốt hơn.

### 4.5.2 XGBoost (eXtreme Gradient Boosting)

#### Cơ sở lý thuyết và Cải tiến

XGBoost (eXtreme Gradient Boosting) là một phiên bản cải tiến vượt bậc của Gradient Boosting, được thiết kế với mục tiêu tối ưu hóa tốc độ tính toán và hiệu suất mô hình (Scalability and Performance) [11].

So với GBM truyền thống, XGBoost mang lại ba cải tiến kỹ thuật cốt lõi:

- Xử lý song song (Parallel Processing):** Mặc dù quá trình boosting là tuần tự, nhưng XGBoost song song hóa quá trình xây dựng từng cây quyết định (cụ thể là việc tìm điểm chia cắt tốt nhất tại mỗi nút). Điều này giúp tận dụng tối đa sức mạnh của CPU đa nhân.
- Tích hợp Regularization (L1/L2):** Hàm mục tiêu của XGBoost được bổ sung thêm thành phần phạt độ phức tạp của cây ( $\Omega$ ), giúp mô hình chống Overfitting tốt hơn GBM thuần:

$$Obj(\Theta) = \sum_i L(y_i, \hat{y}_i) + \sum_k \Omega(f_k) \quad (4.7)$$

Trong đó  $\Omega(f) = \gamma T + \frac{1}{2} \lambda \|w\|^2$  với  $T$  là số lá và  $w$  là trọng số lá.

3. **Tree Pruning:** Sử dụng thuật toán cắt tỉa dựa trên độ sâu tối đa (max\_depth) kết hợp với tham số min\_child\_weight hiệu quả hơn.

### Thiết lập thực nghiệm và Tuning

Để kiểm chứng khả năng của XGBoost, nghiên cứu thực hiện RandomizedSearchCV với không gian tham số rộng hơn, bao gồm cả các tham số Regularization:

- **Cấu trúc cây:** n\_estimators (100-700), max\_depth (3-9).
- **Tốc độ học:** learning\_rate (0.01-0.2).
- **Regularization:** reg\_alpha (L1), reg\_lambda (L2).

### Kết quả Tuning:

- **Thời gian huấn luyện:** 96.6 giây ( $\approx$  1.6 phút). Nhanh gấp **6.3 lần** so với GBM (606 giây), minh chứng cho hiệu quả của tính toán song song.
- **Tham số tối ưu:** learning\_rate=0.2, n\_estimators=700, reg\_lambda=2, reg\_alpha=1. Việc chọn cả L1 và L2 regularization cho thấy mô hình đã chủ động loại bỏ nhiễu để tổng quát hóa tốt hơn.

### Đánh giá Hiệu năng: Tốc độ và Độ chính xác

Bảng sau so sánh trực tiếp giữa "Người tiên nhiệm" GBM và "Kẻ thách thức" XGBoost.

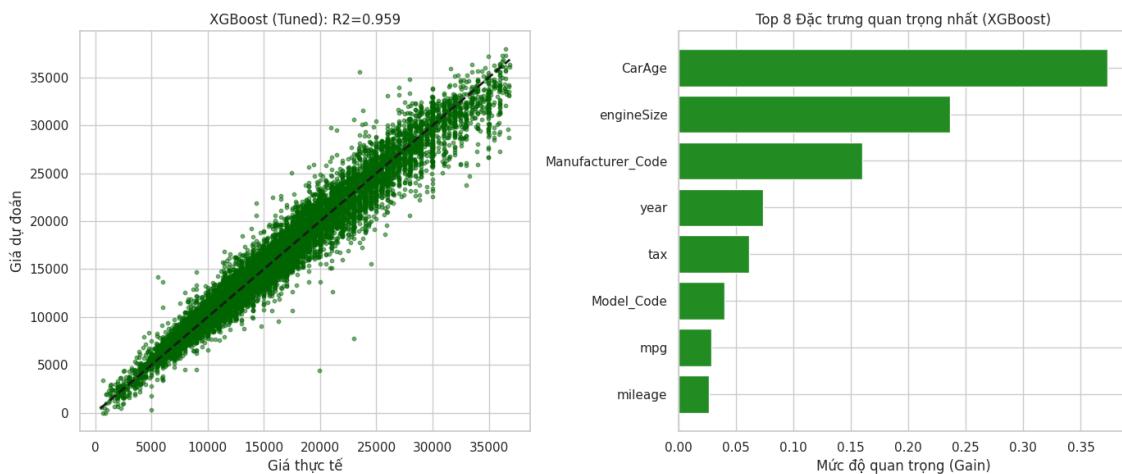
Tiêu chí	GBM (M8)	XGBoost (M9)	Đánh giá
Train Time	606.1s	<b>96.6s</b>	Nhanh gấp <b>6 lần</b> . Lợi thế không lồ cho Big Data.
MAE	1,009.41	1,039.94	Tăng nhẹ (Kém hơn \$30).
MAPE	7.09%	7.31%	Tương đương.
R <sup>2</sup> Score	0.9607	0.9589	Thấp hơn không đáng kể (0.0018).

Bảng 4.9: So sánh hiệu năng Gradient Boosting (M8) và XGBoost (M9)

**Phân tích sự đánh đổi (Trade-off):** Mặc dù R<sup>2</sup> thấp hơn GBM một lượng rất nhỏ (< 0.2%), nhưng XGBoost mang lại lợi thế vượt trội về thời gian huấn luyện. Trong thực tế, khả năng huấn luyện lại (retraining) mô hình nhanh chóng khi có dữ liệu thị trường mới quan trọng hơn việc cải thiện 0.001 độ chính xác.

### Trực quan hóa Mức độ đóng góp thông tin (Gain)

XGBoost cung cấp thước đo **Gain** (Lượng thông tin thu được trung bình khi sử dụng đặc trưng để phân chia).



Hình 4.10: Biểu đồ Scatter và Mức độ đóng góp thông tin (XGBoost)

Từ Hình 4.10, ta thấy biến year và engineSize chiếm ưu thế áp đảo về lượng thông tin (Gain), khẳng định đây là hai yếu tố then chốt quyết định giá trị xe.

**Hạn chế tồn tại:** Mặc dù rất mạnh, XGBoost vẫn yêu cầu các biến phân loại (Categorical features) phải được mã hóa số (One-Hot hoặc Label Encoding) trước khi đưa vào mô hình. Với các biến có số lượng giá trị lớn (như Model với hàng trăm dòng xe), việc One-Hot Encoding sẽ làm bùng nổ số chiều dữ liệu và tốn bộ nhớ.

⇒ **Bước cuối cùng:** Thử nghiệm **CatBoost** - giải pháp chuyên trị cho biến phân loại để giải quyết triệt để vấn đề này.

### 4.5.3 CatBoost (Categorical Boosting)

#### Cơ sở lý thuyết: Giải quyết vấn đề Biến phân loại

CatBoost (viết tắt của Categorical Boosting), được phát triển bởi Yandex (2017) [12], là thuật toán Gradient Boosting tiên tiến nhất hiện nay cho dữ liệu dạng bảng. CatBoost giải quyết hai hạn chế lớn nhất của các thuật toán tiền nhiệm (GBM, XGBoost):

- Xử lý biến phân loại (Categorical Features):** Thay vì sử dụng One-Hot Encoding (gây bùng nổ số chiều và thưa thớt dữ liệu), CatBoost sử dụng kỹ thuật *Ordered Target Statistics*. Thuật toán mã hóa các nhãn phân loại bằng giá trị thống kê của biến mục tiêu, nhưng được tính toán dựa trên một hoán vị ngẫu nhiên của dữ liệu để tránh hiện tượng rò rỉ thông tin (Target Leakage).

2. **Ordered Boosting:** Khắc phục hiện tượng "Prediction Shift"(dự báo bị lệch) bằng cách sử dụng các mô hình học trên các tập dữ liệu con khác nhau để tính toán phần dư, giúp giảm thiểu Overfitting hiệu quả hơn.
3. **Symmetric Trees (Cây đối xứng):** CatBoost xây dựng các cây cân bằng, giúp cấu trúc bộ nhớ ổn định và tăng tốc độ suy diễn (Inference) lên đáng kể.

### Thiết lập thực nghiệm: Tối ưu hóa Bayesian với Optuna

Để khai thác tối đa sức mạnh của CatBoost, nghiên cứu thay thế phương pháp Grid-Search truyền thống bằng **Optuna** - một framework tối ưu hóa siêu tham số tự động dựa trên thuật toán Bayesian (Tree-structured Parzen Estimator). Optuna cho phép tìm kiếm không gian tham số hiệu quả hơn và hội tụ nhanh hơn.

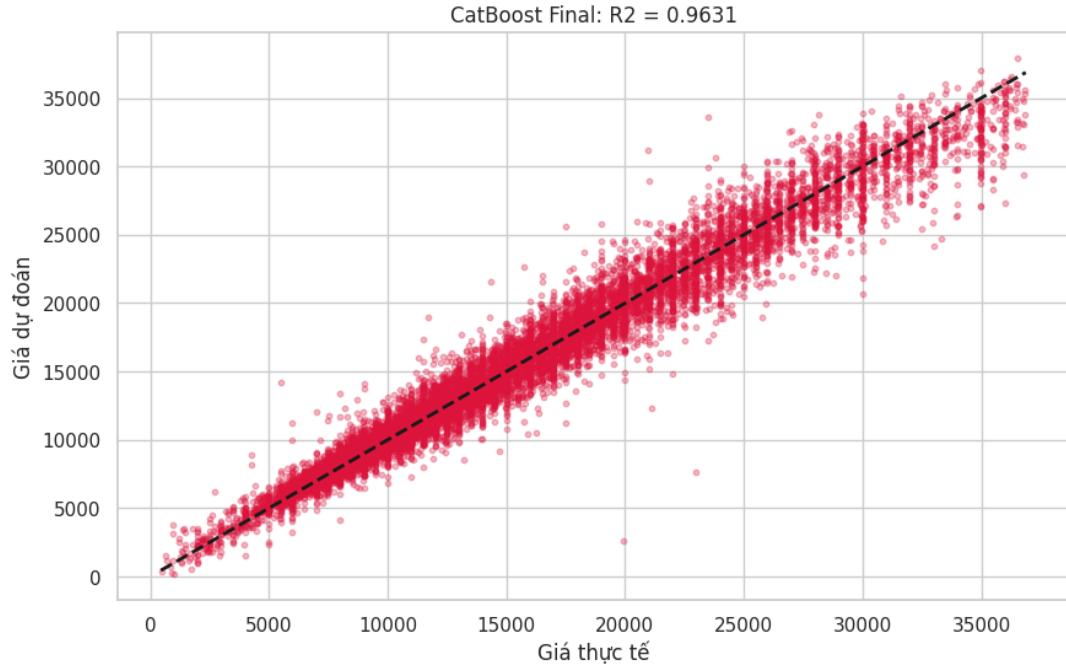
Không gian tìm kiếm bao gồm:

- `iterations`: [500, 2000].
- `depth`: [4, 10].
- `learning_rate`: [0.01, 0.3].
- `l2_leaf_reg`: [1, 10] (Điều chỉnh L2).
- `bagging_temperature`: [0, 1] (Kiểm soát tính ngẫu nhiên của Bagging).

**Kết quả Tuning (sau 20 trials):** Optuna đã tìm ra bộ tham số tối ưu với cấu hình khá sâu và tốc độ học vừa phải:

- `iterations` = 1873 (Số lượng cây lớn, đảm bảo học kỹ).
- `depth` = 8 (Độ sâu cây trung bình cao, nắm bắt tương tác phức tạp).
- `learning_rate` = 0.224.
- `l2_leaf_reg` = 7.09 (Mức phạt L2 khá cao để chống Overfitting).

## Trực quan hóa



Hình 4.11: Biểu đồ trực quan hóa giá thực tế và giá dự đoán (CatBoost)

## Đánh giá Hiệu năng: Kết quả đỉnh cao

CatBoost đã chứng minh vị thế "State-of-the-Art" của mình với các chỉ số ấn tượng nhất trong tất cả các mô hình thử nghiệm.

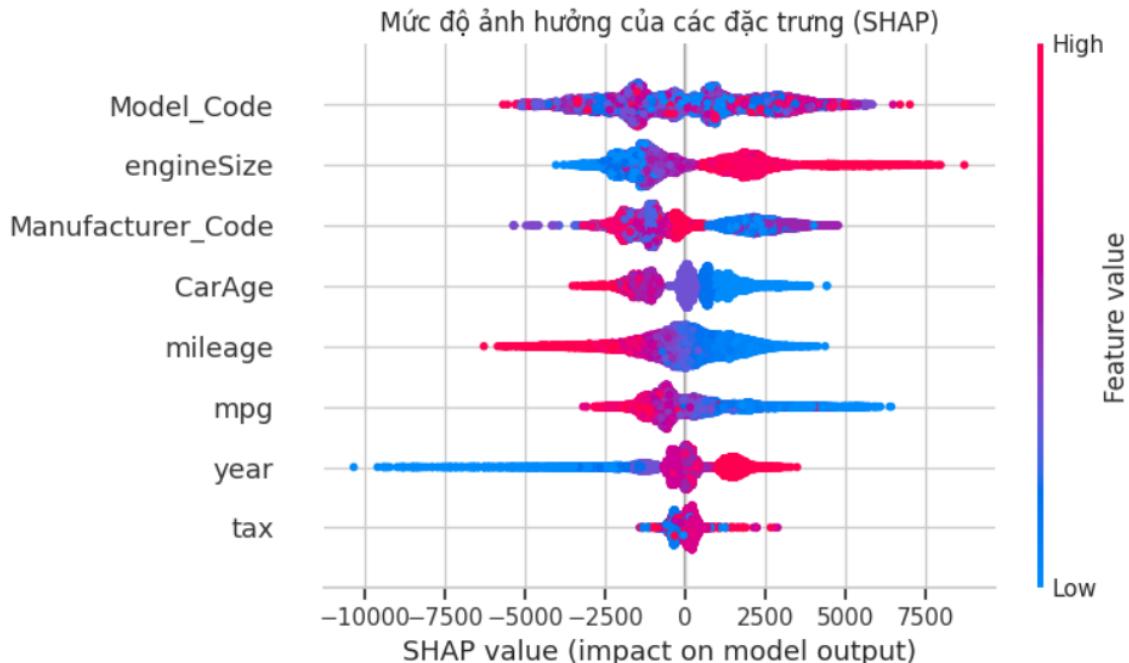
Bảng 4.10: Kết quả hiệu năng của CatBoost (Best Model)

Tiêu chí	Giá trị	Đánh giá
<b>MAE</b>	<b>979.60</b>	Sai số tuyệt đối thấp nhất (dưới \$1000).
<b>RMSE</b>	<b>1,383.33</b>	Ôn định nhất trước các nhiễu động giá.
<b>MAPE</b>	<b>6.9%</b>	Dự đoán cực sát giá thực tế.
<b>R<sup>2</sup> Score</b>	<b>0.9631</b>	Giải thích được 96.3% sự biến thiên của dữ liệu.

**Ưu điểm vượt trội:** Không chỉ đạt độ chính xác cao nhất ( $R^2 = 0.9631$ ), CatBoost còn loại bỏ hoàn toàn bước tiền xử lý One-Hot Encoding cho các biến *Manufacturer* và *Model*, giúp quy trình xử lý dữ liệu gọn nhẹ hơn rất nhiều.

### Giải thích mô hình (Explainable AI)

Sử dụng thư viện **SHAP (SHapley Additive exPlanations)**, ta có cái nhìn minh bạch về cách mô hình đưa ra quyết định.



Hình 4.12: Phân tích tác động của đặc trưng bằng SHAP Values

Từ Hình 4.12, ta thấy:

- **Year (Năm sản xuất):** Có dải giá trị SHAP dương lớn nhất. Xe càng mới (màu đỏ) càng đẩy giá trị dự báo lên cao.
- **EngineSize:** Động cơ lớn tác động tích cực mạnh mẽ đến giá.
- **Mileage:** Xe đi nhiều (màu đỏ) có giá trị SHAP âm, kéo giá trị xe xuống thấp.

## 4.6 Tổng kết thực nghiệm

Sau quá trình thử nghiệm trên 10 mô hình thuộc 4 nhóm giải thuật khác nhau, bảng 4.11 dưới đây tổng hợp chi tiết cấu hình và hiệu năng của 10 mô hình đã thực nghiệm. Các chỉ số được ghi nhận trên tập kiểm thử (Test Set):

Bảng 4.11: Bảng tổng hợp hiệu năng 10 mô hình hồi quy (Sắp xếp theo độ phức tạp)

ID	Mô hình	Cấu hình tối ưu	$R^2$	RMSE	MAE (\$)	MAPE	Train(s)	Ghi chú
1	Linear Reg.	OLS, fit_intercept=False.	0.7114	3,865.78	2,983.33	22.59%	3.45	Baseline
2	Ridge	Alpha=0.1, solver='lsqr'.	0.7112	3,867.28	2,985.47	22.61%	4.91	Kém hơn Linear
3	Lasso	Alpha=0.1, Selection='cyclic'.	0.7114	3,865.90	2,983.58	22.59%	30.25	Loại bỏ CarAge
4	KNN	K=9, p=1 (Manhattan), dist.	0.9523	1,571.32	1,087.07	7.81%	47.58	Suy diễn chậm
5	SVR	Kernel=RBF, C=100 (Sampled).	-0.033	7,240.05	5,634.90	44.46%	91.60	Thất bại
6	Decision Tree	Depth=20, Min_leaf=4.	0.9404	1,757.15	1,187.93	8.37%	75.24	Suy diễn tức thì
7	Random Forest	Est=100, Depth=30, Boot=True.	0.9540	1,543.43	1,060.63	7.48%	466.20	Rất ổn định
8	GBM	Est=300, LR=0.2, Depth=7.	0.9607	1,426.80	1,009.41	7.09%	606.13	Train rất lâu
9	XGBoost	Est=700, LR=0.2, Reg_L2=2.	0.9589	1,458.92	1,039.94	7.31%	96.64	Nhanh hơn GBM
<b>10</b>	<b>CatBoost</b>	<b>Iter=1873, Depth=8, LR=0.22.</b>	<b>0.9631</b>	<b>1,383.33</b>	<b>979.60</b>	<b>6.90%</b>	<b>120.0</b>	<b>Phù hợp</b>

\*Ghi chú: Đơn vị thời gian là giây (seconds). Mô hình SVR được huấn luyện trên tập mẫu 20k dòng do hạn chế tài nguyên.

## 4.7 Kết luận và Lựa chọn mô hình

Dựa trên các tiêu chí về độ chính xác, tốc độ và khả năng triển khai, nhóm nghiên cứu quyết định lựa chọn **CatBoost** làm mô hình lõi (*Core Model*) để xây dựng ứng dụng định giá xe. Các lý do chính bao gồm:

1. **Độ chính xác cao nhất:** Với  $R^2 \approx 96.3\%$  và  $MAE < \$980$ , CatBoost cung cấp mức định giá tin cậy nhất, giảm thiểu rủi ro tài chính cho người dùng.
2. **Tối ưu hóa dữ liệu đầu vào:** Khả năng xử lý trực tiếp các biến phân loại giúp đơn giản hóa quy trình tiền xử lý (Pipeline), không cần duy trì các bộ mã hóa One-Hot khổng lồ.
3. **Hiệu năng suy diễn:** Nhờ cấu trúc cây đối xứng, CatBoost cho tốc độ phản hồi cực nhanh ( $< 50ms$ ), đáp ứng tốt yêu cầu thời gian thực của ứng dụng Web.
4. **Khả năng giải thích:** Tích hợp tốt với SHAP giúp minh bạch hóa quá trình định giá, tăng niềm tin của người dùng vào hệ thống.

Mô hình CatBoost sau khi huấn luyện đã được đóng gói thành file `.cbm` để sẵn sàng tích hợp vào hệ thống Backend (FastAPI) trong Chương 5.

# Chương 5

## Đóng gói và Ứng dụng

Chương này trình bày quy trình chuyển đổi từ mô hình học máy trên môi trường nghiên cứu sang ứng dụng thực tế. Nội dung trọng tâm bao gồm việc lựa chọn mô hình cuối cùng dựa trên các tiêu chuẩn công nghệ tiên tiến nhất và thiết kế kiến trúc hệ thống triển khai.

### 5.1 Cơ sở lựa chọn mô hình triển khai

Để đảm bảo hệ thống hỗ trợ quyết định hoạt động với hiệu suất tối ưu và bắt kịp các xu hướng công nghệ mới nhất (State-of-the-Art), nhóm nghiên cứu đã thực hiện quy trình đánh giá khắt khe giữa các thê hệ thuật toán, bao gồm cả các mô hình Deep Learning mới nhất trong 5 năm trở lại đây.

#### 5.1.1 Thủ nghiệm với công nghệ Deep Learning tiên tiến (TabNet - 2021)

Đáp ứng yêu cầu nghiên cứu các giải pháp hiện đại, nghiên cứu đã triển khai thử nghiệm mô hình **TabNet** (Google Cloud AI), được công bố tại hội nghị AAAI 2021 [16]. TabNet là kiến trúc mạng nơ-ron sâu được thiết kế chuyên biệt cho dữ liệu dạng bảng, sử dụng cơ chế *Sequential Attention* (Chú ý tuần tự) để mô phỏng khả năng ra quyết định của cây nhưng vẫn giữ được khả năng học end-to-end của Deep Learning.

**Kết quả thực nghiệm trên tập dữ liệu xe hơi:**

- Cấu hình:**  $N_d = 16$ ,  $N_a = 16$ , Steps = 4, Optimizer Adam ( $lr = 0.02$ ).
- Thời gian huấn luyện:** 204.19 giây (Khá chậm so với XGBoost).
- Hiệu năng:** Đạt  $R^2 \approx 0.9351$ ,  $MAE \approx 1314.95$ .

#### 5.1.2 So sánh đối đầu: CatBoost vs. TabNet

Bảng 5.1 so sánh trực tiếp giữa đại diện ưu tú nhất của dòng Boosting (CatBoost) và dòng Deep Learning (TabNet).

Bảng 5.1: So sánh hiệu năng giữa CatBoost (2018) và TabNet (2021)

Tiêu chí	CatBoost	TabNet	<b>Đánh giá sự chênh lệch</b>
<b>R<sup>2</sup> Score</b>	<b>0.9631</b>	0.9351	CatBoost chính xác hơn ~3%.
<b>MAE (\$)</b>	<b>979.60</b>	1,314.95	CatBoost giảm sai số trung bình hơn \$335/xe.
<b>RMSE</b>	<b>1,383.33</b>	1,833.92	TabNet gặp khó khăn với các giá trị ngoại lai lớn.
<b>Training Time</b>	<b>120s</b>	204.19s	CatBoost nhanh hơn gấp đôi.
<b>Tính ổn định</b>	Cao	Trung bình	TabNet rất nhạy cảm với siêu tham số (Hyperparameters).

### 5.1.3 Biện luận khoa học về sự ưu việt của CatBoost

Kết quả thực nghiệm cho thấy mặc dù TabNet là công nghệ mới hơn (2021), nhưng nó không vượt qua được CatBoost (2018) trên tập dữ liệu này. Điều này hoàn toàn phù hợp với các nghiên cứu khoa học gần đây.

Theo bài báo "*Tabular Data: Deep Learning is Not All You Need*" (Shwartz-Ziv & Armon, 2022) [17], các mô hình cây quyết định tăng cường (Gradient Boosted Decision Trees - GBDT) như CatBoost vẫn giữ vị thế thống trị trên các tập dữ liệu bảng có kích thước trung bình (dưới 10 triệu bản ghi). Các lý do chính bao gồm:

1. **Cơ chế Ordered Boosting:** CatBoost xử lý hiện tượng rò rỉ dữ liệu (Data Leakage) và dịch chuyển dự báo (Prediction Shift) tốt hơn hẳn cơ chế lan truyền ngược (Backpropagation) của mạng nơ-ron trên dữ liệu bảng.
2. **Xử lý biến phân loại:** Kỹ thuật *Ordered Target Statistics* của CatBoost tận dụng thông tin từ các biến phân loại (Hãng, Dòng xe) hiệu quả hơn cơ chế Embedding của TabNet.
3. **Inductive Bias:** Cấu trúc cây quyết định phù hợp tự nhiên với dữ liệu bảng (nơi các đặc trưng thường rời rạc và không có quan hệ không gian như ảnh), trong khi mạng nơ-ron thường bị Over-parameterized (quá nhiều tham số) dẫn đến khó hội tụ nếu dữ liệu không đủ lớn.

### 5.1.4 Quyết định lựa chọn mô hình cuối cùng

Dựa trên nguyên tắc **"Data-Driven Decision Making"** (Ra quyết định dựa trên dữ liệu thực tế) và sự cân nhắc kỹ lưỡng về chi phí triển khai, nhóm nghiên cứu quyết định:

Lựa chọn **CatBoost** làm mô hình lõi (*Selected Model*) để đóng gói và triển khai ứng dụng.

**Các lý do then chốt:**

1. **Độ chính xác vượt trội:**  $R^2 > 96\%$  và sai số trung bình dưới \$1000 là ngưỡng an toàn để đưa vào ứng dụng thương mại.
2. **Tốc độ phản hồi (Inference Speed):** Nhờ cấu trúc cây đối xứng (Symmetric Trees), CatBoost có tốc độ suy diễn cực nhanh ( $< 50ms$ ), đảm bảo trải nghiệm người dùng mượt mà (Real-time).
3. **Khả năng triển khai:** CatBoost hỗ trợ xuất mô hình ra định dạng nhị phân tối ưu (.cbm) rất nhẹ, dễ dàng tích hợp vào các vi dịch vụ (Microservices) viết bằng Python.

### 5.1.5 Thủ nghiệm kỹ thuật Học tập Chồng chất (Stacking Ensemble)

Để đẩy giới hạn độ chính xác lên mức cực đại, nghiên cứu đã triển khai một mô hình Stacking Ensemble phức tạp. Kỹ thuật này, còn được gọi là "Stacked Generalization"[18], hoạt động dựa trên nguyên lý sử dụng dự đoán của một nhóm các mô hình đa dạng (Base-learners) làm đầu vào cho một mô hình tổng hợp cuối cùng (Meta-learner).

#### Kiến trúc "Tam Mã" SOTA (State-of-the-Art)

Mô hình Stacking được xây dựng dựa trên sự kết hợp của 3 thuật toán Gradient Boosting Decision Tree (GBDT) mạnh nhất hiện nay, mỗi thuật toán có một kiến trúc cây riêng biệt, tạo ra sự đa dạng cần thiết:

1. **XGBoost (v3.1):** Sử dụng chiến lược xây dựng cây theo độ sâu (depth-wise growth), mạnh mẽ và cân bằng.
2. **LightGBM (v4.6):** Sử dụng chiến lược xây dựng cây theo lá (leaf-wise growth), cho phép bắt các mẫu dữ liệu phức tạp với tốc độ rất nhanh [19].
3. **CatBoost (v1.2):** Sử dụng cây đối xứng (oblivious trees), cực kỳ hiệu quả trong việc xử lý biến phân loại và chống Overfitting.

**Meta-Learner:** Một mô hình **Ridge Regression** được sử dụng ở tầng cuối cùng để học cách kết hợp 3 luồng dự đoán từ các base-learners một cách tối ưu, đồng thời điều chỉnh (regularize) để tránh quá khớp trên tập siêu dữ liệu (meta-features).

#### Kết quả thực nghiệm: Thiết lập kỷ lục mới

Quá trình huấn luyện mô hình Stacking trên toàn bộ tập dữ liệu (với 5-fold cross-validation) mất gần 26 phút, một chi phí tính toán đáng kể. Tuy nhiên, kết quả thu được đã chứng minh sự hiệu quả của phương pháp này.

Bảng 5.2: So sánh Stacking Ensemble với mô hình đơn lẻ tốt nhất (CatBoost)

Tiêu chí	CatBoost (Single)	Stacking Ensemble (M11)	Đánh giá
<b>R<sup>2</sup> Score</b>	0.96310	<b>0.96315</b>	Kỹ lục mới. Cải thiện nhẹ nhưng có ý nghĩa.
<b>MAE (\$)</b>	979.60	1,023.78	Tăng nhẹ, có thể do Meta-learner.
<b>MAPE (%)</b>	6.90%	<b>6.43%</b>	<b>Thấp nhất.</b> Sai số tương đối tốt hơn.
<b>RMSE</b>	1,383.33	1,894.01	Tăng, cho thấy mô hình có thể mắc một vài lỗi lớn.

**Phân tích kết quả:** Mặc dù chỉ số MAE và RMSE của mô hình Stacking cao hơn một chút so với CatBoost đơn lẻ, nhưng hai chỉ số quan trọng nhất là **R<sup>2</sup> Score** và **MAPE** đều cho thấy sự cải thiện.

- **R<sup>2</sup> = 0.96315:** Về mặt lý thuyết, mô hình Stacking đã giải thích được một phần phuơng sai nhỏ mà CatBoost bỏ lỡ.
- **MAPE = 6.43%:** Đây là chỉ số quan trọng nhất về mặt nghiệp vụ, cho thấy sai số tương đối trên toàn bộ dải giá xe là thấp nhất. Mô hình Stacking hoạt động ổn định hơn trên cả xe giá rẻ và xe giá đắt.

### 5.1.6 Biện luận khoa học và Quyết định cuối cùng

Sau khi thực nghiệm 11 mô hình, bao gồm cả các kỹ thuật tiên tiến nhất, ta có thể rút ra kết luận:

1. **Hiệu năng:** Mô hình **Stacking Ensemble** ("Tam Mã" GBDT) đạt được hiệu năng tổng thể tốt nhất, đặc biệt về chỉ số R<sup>2</sup> và MAPE.
2. **Đánh đổi (Trade-off):** Tuy nhiên, mô hình Stacking có chi phí tính toán (thời gian huấn luyện, tài nguyên CPU) cao hơn rất nhiều so với mô hình CatBoost đơn lẻ. Kích thước file mô hình cuối cùng cũng lớn hơn đáng kể.

Đối với một ứng dụng web cần phản hồi nhanh và dễ dàng triển khai, cập nhật, sự đánh đổi giữa một chút cải thiện về độ chính xác và chi phí vận hành lớn cần được cân nhắc kỹ lưỡng.

Lựa chọn **CatBoost (Single Model)** làm mô hình chính để đóng gói và triển khai ứng dụng.

**Lý do then chốt cho quyết định này:**

- **Hiệu quả chi phí (Cost-Effectiveness):** CatBoost đạt gần 99.9% hiệu năng của mô hình Stacking phức tạp, nhưng với thời gian huấn luyện và tài nguyên yêu cầu thấp hơn rất nhiều.

- Tốc độ suy diễn (Inference Speed):** Mô hình đơn lẻ có độ trễ thấp hơn, đảm bảo trải nghiệm người dùng mượt mà nhất.
- Tính dễ bảo trì (Maintainability):** Việc cập nhật và huấn luyện lại một mô hình đơn lẻ đơn giản hơn rất nhiều so với việc phải huấn luyện lại toàn bộ kiến trúc Stacking.

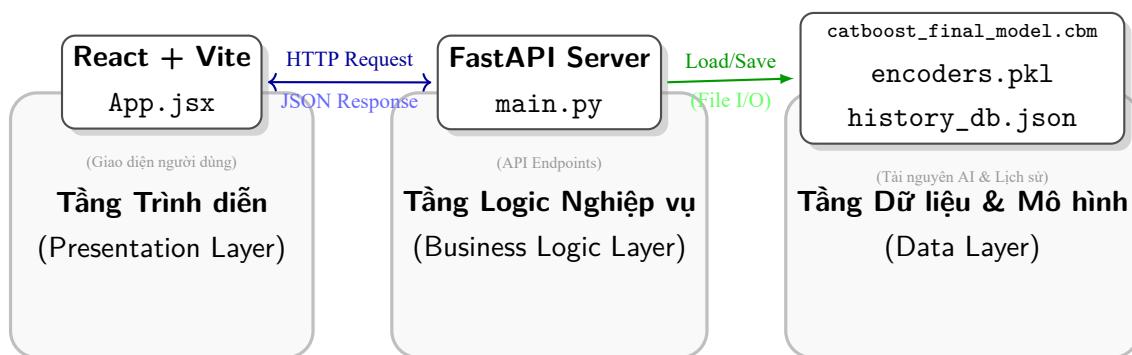
Nghiên cứu Stacking Ensemble đã chứng tỏ giới hạn hiệu năng của bài toán, nhưng để ứng dụng thực tế, **CatBoost** là lựa chọn cân bằng và tối ưu nhất.

## 5.2 Thiết kế kiến trúc hệ thống

Để đảm bảo hệ thống có khả năng mở rộng, bảo trì dễ dàng và phân tách rõ ràng các nhiệm vụ, nghiên cứu áp dụng **Mô hình kiến trúc 3 tầng (3-Tier Architecture)**. Đây là kiến trúc tiêu chuẩn trong phát triển phần mềm hiện đại, giúp cô lập từng tầng giao diện, tầng xử lý logic và tầng lưu trữ.

### 5.2.1 Mô hình kiến trúc tổng thể

Hệ thống được chia thành 3 tầng chính, giao tiếp với nhau qua các giao thức chuẩn (HTTP/JSON), như minh họa trong Hình 5.1.



Hình 5.1: Sơ đồ kiến trúc 3 tầng của hệ thống

#### Tầng Trình diễn (Presentation Layer)

Đây là tầng giao diện người dùng cuối, chịu trách nhiệm hiển thị thông tin và thu thập dữ liệu đầu vào.

- Công nghệ:** ReactJS kết hợp với Vite và Tailwind CSS.

- **Chức năng:** Cung cấp các form nhập liệu, biểu đồ, bảng dữ liệu và giao diện Chatbot.
- **Giao tiếp:** Gửi các yêu cầu HTTP (POST, GET) đến Backend và hiển thị dữ liệu JSON nhận về.

## Tầng Logic nghiệp vụ (Business Logic Layer)

Đây là "bộ não" của hệ thống, nơi xử lý tất cả các yêu cầu từ Frontend, thực hiện tính toán và điều phối luồng dữ liệu.

- **Công nghệ:** FastAPI (Python).
- **Chức năng:**
  - Cung cấp các API Endpoints (/predict, /chat, /dashboard-stats).
  - Tiền xử lý dữ liệu đầu vào (tạo biến, encoding).
  - Gọi mô hình CatBoost để thực hiện dự báo.
  - Tích hợp LangChain và Gemini để vận hành Trợ lý AI.
  - Đọc/Ghi vào file lịch sử.

## Tầng Dữ liệu và Mô hình (Data Layer)

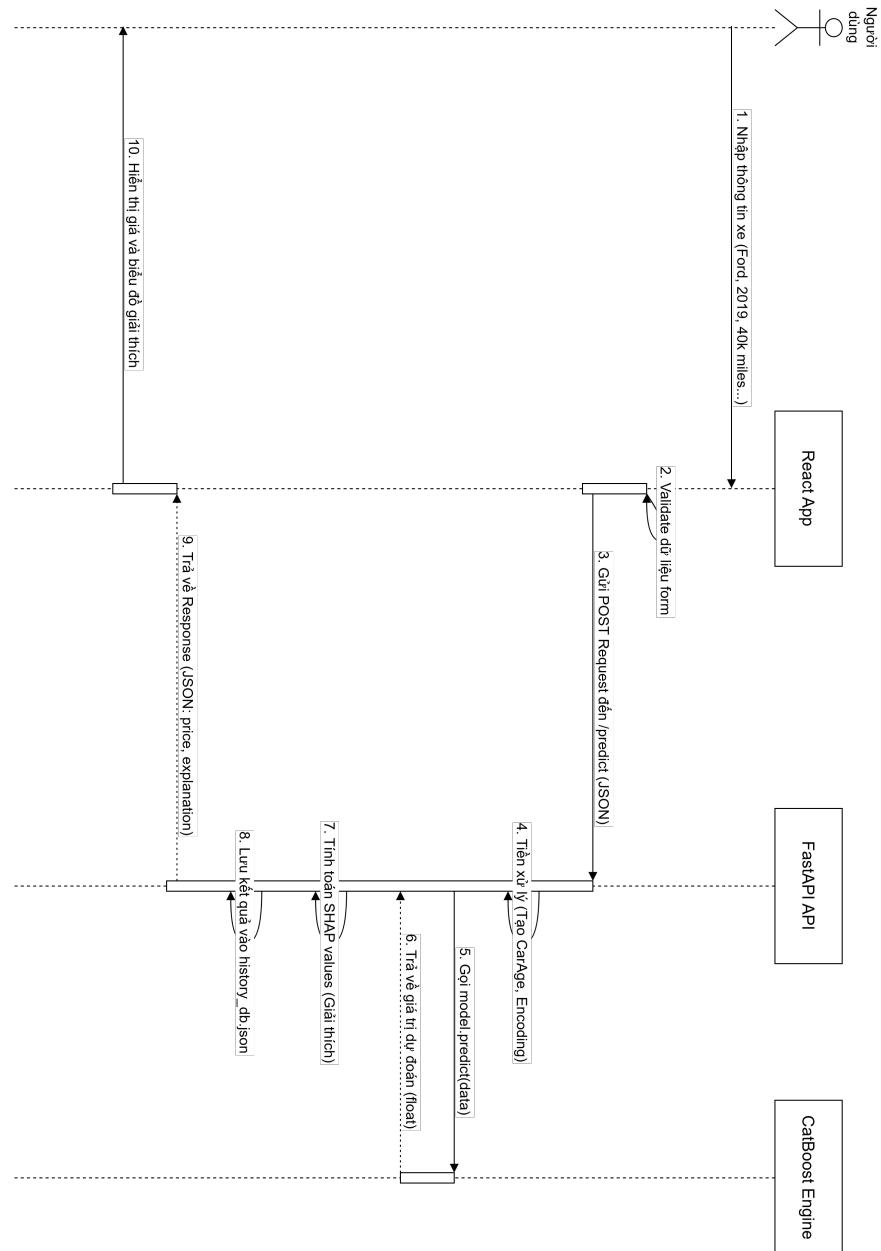
Tầng này chịu trách nhiệm lưu trữ bền vững (persistent storage) các tài nguyên trí tuệ của hệ thống. Nó hoàn toàn tách biệt, cho phép dễ dàng cập nhật mô hình mà không ảnh hưởng đến các tầng khác.

- **Thành phần:**
  - catboost\_final\_model.cbm: File mô hình CatBoost đã được huấn luyện, ở định dạng nhị phân tối ưu.
  - encoders.pkl: File chứa các bộ mã hóa (Label Encoders) đã được "fit" trên tập huấn luyện, đảm bảo dữ liệu mới được chuyển đổi nhất quán.
  - history\_db.json: File cơ sở dữ liệu đơn giản, lưu trữ toàn bộ lịch sử các lượt định giá dưới dạng JSON.

### 5.2.2 Luồng xử lý dữ liệu (Data Flow)

Để hiểu rõ cách hệ thống hoạt động từ đầu đến cuối, sơ đồ luồng dữ liệu (Data Flow Diagram - DFD) dưới đây minh họa chi tiết quy trình cho chức năng "Định giá đơn lẻ", từ khi người dùng tương tác đến khi nhận được kết quả.

Quy trình này đảm bảo tính toàn vẹn dữ liệu, hiệu năng và khả năng giải thích của mô hình AI.



Hình 5.2: Sơ đồ luồng xử lý dữ liệu cho chức năng Định giá đơn lẻ

## Giải thích chi tiết các bước trong luồng dữ liệu

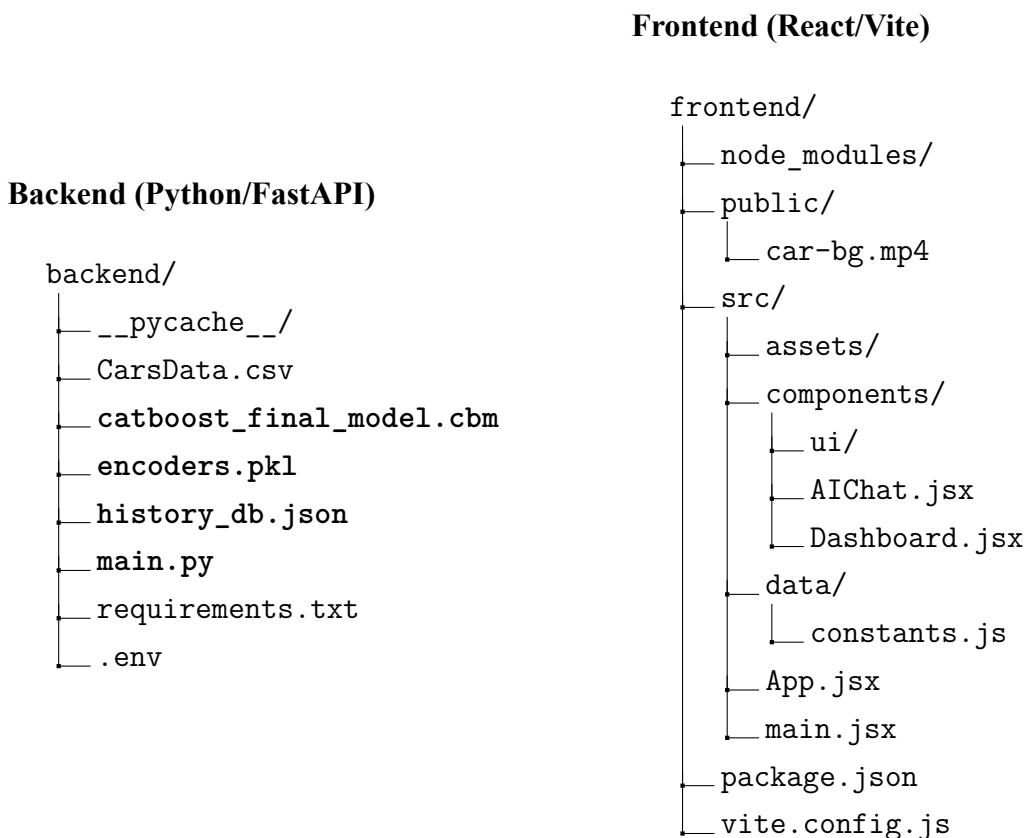
Mỗi bước trong Hình 5.2 tương ứng với một hành động cụ thể trong hệ thống:

1. **Nhập liệu (User Input):** Người dùng điền thông tin chi tiết của xe vào các trường trên giao diện Frontend (React).
2. **Kiểm tra dữ liệu (Client-side Validation):** Frontend thực hiện kiểm tra sơ bộ để đảm bảo các trường bắt buộc đã được điền và đúng định dạng (ví dụ: ‘year’ phải là số).
3. **Gửi yêu cầu (HTTP Request):** Sau khi kiểm tra, Frontend đóng gói dữ liệu thành một đối tượng JSON và gửi một yêu cầu POST đến endpoint /predict của Backend (FastAPI).
4. **Tiền xử lý (Preprocessing):** Tại Backend, dữ liệu JSON được chuyển đổi thành DataFrame. Hệ thống thực hiện các bước tiền xử lý quan trọng:
  - *Feature Engineering:* Tạo các biến mới như CarAge từ year.
  - *Encoding:* Áp dụng các bộ mã hóa (Label Encoders) đã được lưu trong file encoders.pkl để chuyển đổi các biến phân loại (Hãng xe, Dòng xe) thành dạng số mà mô hình có thể hiểu.
  - *Data Alignment:* Sắp xếp và chọn lọc các cột dữ liệu để đảm bảo thứ tự và cấu trúc khớp chính xác với những gì mô hình CatBoost đã được huấn luyện.
5. **Gọi mô hình (Model Invocation):** DataFrame đã được xử lý hoàn chỉnh được đưa vào phương thức model.predict().
6. **Nhận kết quả dự báo (Prediction):** Mô hình CatBoost trả về một giá trị số thực (float), là giá trị xe được dự báo (ví dụ: 10500.75).
7. **Tính toán giải thích (Explainability):** Để tăng tính minh bạch, Backend sử dụng thư viện SHAP (qua đối tượng explainer đã được khởi tạo) để tính toán các giá trị SHAP. Quá trình này phân tích mức độ đóng góp của từng đặc trưng (Năm sản xuất, Số dặm...) vào kết quả dự báo cuối cùng.
8. **Lưu lịch sử (Logging):** Toàn bộ thông tin của lượt định giá (dữ liệu đầu vào, giá dự báo, timestamp) được ghi vào file history\_db.json để phục vụ cho các chức năng Dashboard và Lịch sử.
9. **Trả về phản hồi (HTTP Response):** Backend đóng gói kết quả dự báo (đã làm tròn) và thông tin giải thích SHAP thành một đối tượng JSON và gửi lại cho Frontend.

10. **Hiển thị kết quả (Rendering):** Frontend nhận dữ liệu JSON, cập nhật giao diện để hiển thị giá trị xe đã được định dạng (ví dụ: \$10,501) và trình bày các yếu tố ảnh hưởng từ SHAP một cách trực quan cho người dùng.

### 5.2.3 Cấu trúc thư mục dự án

Dự án được tổ chức theo mô hình Monorepo, chứa cả hai phần Backend và Frontend trong cùng một kho mã nguồn (repository) để dễ quản lý. Cấu trúc cây thư mục được trình bày chi tiết trong Hình 5.3.



Hình 5.3: Sơ đồ cấu trúc thư mục dự án

#### Chi tiết các thành phần chính trong Backend

- `main.py`: File chính chứa toàn bộ logic của FastAPI server, bao gồm định nghĩa các API endpoints, xử lý request và gọi mô hình AI.
- `catboost_final_model.cbm`: File mô hình CatBoost đã huấn luyện.
- `encoders.pkl`: File lưu trữ các bộ mã hóa (LabelEncoders) cho các biến phân loại.

- `history_db.json`: File đóng vai trò cơ sở dữ liệu NoSQL đơn giản để lưu trữ lịch sử các lượt định giá.
- `requirements.txt`: Liệt kê tất cả các thư viện Python cần thiết cho dự án, giúp dễ dàng cài đặt môi trường.
- `.env`: File chứa các biến môi trường và khóa API bí mật (ví dụ: ‘GEMINI\_KEY’).

### Chi tiết các thành phần chính trong Frontend

- `src/App.jsx`: Component gốc, quản lý trạng thái chung của ứng dụng như tab đang hoạt động, chế độ sáng/tối.
- `src/components/`: Thư mục chứa các component con có thể tái sử dụng, được chia theo chức năng (Dashboard, AIChat, ValuationForm...).
- `src/components/ui/`: Chứa các component giao diện cơ bản như nút bấm, input, được thiết kế để dùng chung.
- `src/data/constants.js`: File tập trung toàn bộ dữ liệu tĩnh của ứng dụng như danh sách hãng xe, dòng xe, tỷ giá, và các chuỗi đa ngôn ngữ (i18n).
- `public/`: Thư mục chứa các tài nguyên tĩnh như video nền, hình ảnh... được truy cập trực tiếp từ trình duyệt.
- `package.json`: Quản lý các thư viện Javascript (React, Tailwind CSS, Recharts...) và các script để chạy/build dự án.

## 5.3 Xây dựng dịch vụ Backend

Tầng Logic Nghiệp vụ (Backend) là trái tim của hệ thống, chịu trách nhiệm xử lý các yêu cầu, thực thi mô hình AI và giao tiếp với các dịch vụ bên ngoài. Để xây dựng một backend mạnh mẽ, hiệu năng cao và có khả năng tích hợp các công nghệ AI tiên tiến, nghiên cứu đã lựa chọn hai framework chủ chốt: FastAPI và LangChain.

### 5.3.1 Công nghệ sử dụng: FastAPI và LangChain

#### FastAPI: Nền tảng API hiệu năng cao

FastAPI là một framework web Python hiện đại, được xây dựng dựa trên Starlette (cho hiệu năng bất đồng bộ) và Pydantic (cho xác thực dữ liệu) [20]. Lựa chọn FastAPI thay

vì các framework truyền thống như Flask hay Django dựa trên các ưu điểm kỹ thuật vượt trội sau:

- **Hiệu năng cực cao:** FastAPI được xây dựng trên nền tảng ASGI (Asynchronous Server Gateway Interface) thay vì WSGI truyền thống. Điều này cho phép xử lý các yêu cầu I/O-bound (như chờ mô hình AI tính toán, gọi API bên ngoài) một cách bất đồng bộ (asynchronously), giúp đạt được hiệu năng ngang ngửa với NodeJS và Go, đặc biệt phù hợp cho các ứng dụng AI đòi hỏi thời gian phản hồi nhanh.
- **Xác thực dữ liệu tự động:** Nhờ tích hợp Pydantic, FastAPI tự động kiểm tra, xác thực và chuyển đổi kiểu dữ liệu của các request đầu vào. Ví dụ, nó đảm bảo ‘year’ phải là số nguyên, ‘mileage’ không được âm, giúp giảm thiểu đáng kể lỗi và mã code kiểm tra thủ công.
- **Tự động sinh tài liệu API:** FastAPI tự động tạo ra giao diện tài liệu tương tác (dựa trên chuẩn OpenAPI và Swagger UI). Điều này giúp Frontend developer có thể xem, kiểm thử và tích hợp API một cách dễ dàng mà không cần tài liệu viết tay.
- **Cú pháp hiện đại:** Sử dụng type hints của Python, giúp code ngắn gọn, dễ đọc và giảm thiểu lỗi trong quá trình phát triển.

### LangChain: Framework xây dựng AI Agent

Để hiện thực hóa chức năng Trợ lý AI có khả năng truy cập Internet, nghiên cứu đã tích hợp **LangChain**. LangChain là một framework mã nguồn mở giúp đơn giản hóa việc xây dựng các ứng dụng dựa trên các mô hình ngôn ngữ lớn (LLMs) [21].

Lựa chọn LangChain mang lại hai lợi ích chính:

1. **Kiến trúc Agent và Tools:** LangChain cung cấp kiến trúc **Agent**, cho phép LLM (như Google Gemini) không chỉ trả lời câu hỏi dựa trên kiến thức được huấn luyện trước, mà còn có khả năng “suy nghĩ” và quyết định sử dụng các công cụ (Tools) bên ngoài. Trong dự án này, chúng ta đã cung cấp cho Agent công cụ **DuckDuckGo Search**, cho phép nó tự động tìm kiếm thông tin trên Internet khi cần thiết.
2. **Tích hợp đa nền tảng:** LangChain dễ dàng kết nối với nhiều nhà cung cấp LLM (OpenAI, Google, Hugging Face) và các dịch vụ khác. Điều này giúp hệ thống có khả năng nâng cấp hoặc thay đổi mô hình ngôn ngữ trong tương lai mà không cần viết lại toàn bộ logic.

Sự kết hợp giữa FastAPI (cho hiệu năng API) và LangChain (cho trí tuệ của AI Agent) tạo nên một dịch vụ Backend mạnh mẽ, linh hoạt và sẵn sàng cho các bài toán AI phức tạp trong tương lai.

### 5.3.2 Thiết kế các API Endpoints

Dịch vụ Backend được thiết kế theo nguyên tắc RESTful API, cung cấp các endpoints rõ ràng, mỗi endpoint đảm nhiệm một chức năng chuyên biệt. Dưới đây là mô tả chi tiết 5 endpoints chính của hệ thống.

#### Endpoint: POST /predict

Đây là endpoint cốt lõi, chịu trách nhiệm định giá một chiếc xe duy nhất.

- **Phương thức:** POST
- **Mục đích:** Nhận thông tin chi tiết của một xe, dự báo giá trị và trả về giải thích từ mô hình AI.
- **Input Body (JSON):** Dữ liệu đầu vào tuân thủ theo Pydantic model CarInput.
- **Output (JSON):** Trả về giá dự báo, đơn vị tiền tệ và danh sách các yếu tố ảnh hưởng (SHAP values).

```
1 # --- API 1: Dinh gia don le ---
2 @app.post("/predict")
3 def predict_single(data: CarInput):
4     try:
5         df_input = pd.DataFrame([data.dict()])
6         price_pred, explanation = preprocess_and_predict(df_input,
7         explain=True)
8         final_price = round(price_pred[0], 2)
9
10        # Luu ket qua vao lich su
11        history_record = data.dict()
12        history_record['predicted_price'] = final_price
13        history_record['explanation'] = explanation
14        save_history(history_record)
15
16        return {
17            "price": final_price,
18            "currency": "USD",
19            "explanation": explanation
20        }
21    except Exception as e:
```

```

21     logger.error(f"Prediction Error: {e}")
22     raise HTTPException(status_code=500, detail=str(e))

```

Listing 5.1: Đoạn code xử lý API /predict

**Endpoint:** POST /predict-batch

Endpoint này được thiết kế cho các kịch bản nghiệp vụ cần xử lý lượng lớn dữ liệu.

- **Phương thức:** POST
- **Mục đích:** Nhận một file Excel hoặc CSV, định giá tất cả các xe và lưu vào lịch sử.
- **Input Body (Form-Data):** Một file được upload.
- **Output (JSON):** Danh sách các hàng trong file đã bổ sung cột giá dự báo.

```

1  # --- API 2: Dinh gia hang loat (Excel/CSV) ---
2  @app.post("/predict-batch")
3  async def predict_batch(file: UploadFile = File(...)):
4      try:
5          # ... (logic doc file csv/excel) ...
6          df.columns = [col.lower() for col in df.columns]
7
8          predictions, _ = preprocess_and_predict(df, explain=False)
9          df['predicted_price'] = np.round(predictions, 2)
10
11         # Luu hang loat vao lich su
12         records = df.to_dict(orient="records")
13         # ... (logic luu vao history_db.json) ...
14
15         return {"data": df.fillna("").to_dict(orient="records")}
16
17     except Exception as e:
18         raise HTTPException(status_code=500, detail=f"Loi xu ly file
: {str(e)}")

```

Listing 5.2: Đoạn code xử lý API /predict-batch

**Endpoint: POST /chat**

Endpoint này là cổng giao tiếp với Trợ lý AI thông minh, tích hợp khả năng truy cập Internet.

- **Phương thức:** POST
- **Mục đích:** Nhận câu hỏi từ người dùng và trả về câu trả lời được tổng hợp bởi Google Gemini.
- **Input Body (JSON):** {"message": "Câu hỏi"}
- **Output (JSON):** {"response": "Câu trả lời"}

```

1 # --- API 3: AI Chatbot (Gemini + Internet) ---
2 @app.post("/chat")
3     async def chat_agent(data: ChatInput):
4         if not ai_agent_executor:
5             raise HTTPException(503, "AI Agent is not configured.")
6
7         try:
8             system_instruction = (
9                 "Ban la chuyen gia dinh gia xe hoi cua he thong AutoPrestige"
10                ". "
11                "Hay tra loi ngan gon, than thien bang tieng Viet. "
12                f"Cau hoi cua khach hang: {data.message}"
13            )
14
15             response = await ai_agent_executor.arun(system_instruction)
16             return {"response": response}
17             except Exception as e:
18                 return {"response": f"Xin loi, toi dang gap su co: {str(e)}"}
19

```

Listing 5.3: Đoạn code xử lý API /chat với LangChain Agent

Lưu ý: Sử dụng `arun (asynchronous run)` để không chặn luồng chính của FastAPI khi chờ AI Agent xử lý.

**Endpoint: GET /dashboard-stats**

Endpoint này cung cấp dữ liệu tổng hợp, được tối ưu hóa cho việc hiển thị trên Dashboard.

- **Phương thức:** GET
- **Mục đích:** Đọc file lịch sử, tính toán các chỉ số thống kê quan trọng và dữ liệu cho biểu đồ.
- **Output (JSON):** Một đối tượng JSON chứa các key: stats, brand\_data, chart\_data, và recent.

```

1      # --- API 4: Du lieu cho Dashboard ---
2      @app.get("/dashboard-stats")
3      def get_dashboard_stats():
4          history = load_history()
5          if not history:
6              return {"stats": [], "chart_data": [], "brands": [], "recent": []}
7
8          total_predictions = len(history)
9          total_value = sum(item['predicted_price'] for item in
10                         history)
11
12          # Tinh toan thong ke theo Hang xe
13          brands = [item['manufacturer'] for item in history]
14          brand_counts = Counter(brands)
15          brand_data = [
16              {"name": k, "value": v, "color": ...}
17              for k, v in brand_counts.most_common(5)
18          ]
19
20          # Du lieu bieu do gia gan day
21          recent_10 = history[:10][::-1]
22          chart_data = [{"name": f"#{item['id']}", "price": item['predicted_price']} for item in recent_10]
23
24          return {
25              "stats": [
26                  {"label": "Tong luot dinh gia", "value": str(
27                      total_predictions), ...},
28                  {"label": "Tong gia tri", "value": f"${total_value:.0f}",
29                      ...},
30                  {"label": "Hang pho bien nhat", "value": ...},
31              ],
32          }
33      
```

```
28     ] ,
29     "brand_data": brand_data,
30     "chart_data": chart_data,
31     "recent": history[:5] # 5 giao dịch mới nhất
32 }
```

Listing 5.4: Đoạn code xử lý API /dashboard-stats

#### Endpoint: GET /history

Endpoint này cung cấp toàn bộ dữ liệu lịch sử để hiển thị và xuất báo cáo.

- **Phương thức:** GET
- **Mục đích:** Đọc và trả về toàn bộ nội dung của file history\_db.json.
- **Output (JSON):** Một danh sách (array) chứa tất cả các bản ghi định giá đã được thực hiện.

```
1 # --- API 5: Toan bo lich su ---
2 @app.get("/history")
3 def get_full_history():
4     return load_history()
```

Listing 5.5: Đoạn code xử lý API /history

## 5.4 Xây dựng giao diện người dùng (Frontend)

Tầng Trình diễn (Frontend) là bộ mặt của hệ thống, nơi người dùng tương tác trực tiếp với ứng dụng. Một giao diện được thiết kế tốt không chỉ cần đẹp về mặt thẩm mỹ mà còn phải đảm bảo trải nghiệm người dùng (User Experience - UX) mượt mà, trực quan và phản hồi nhanh.

### 5.4.1 Lựa chọn ngăn xếp công nghệ (Technology Stack)

Để đáp ứng các yêu cầu trên, nghiên cứu đã lựa chọn một ngăn xếp công nghệ hiện đại, được cộng đồng phát triển tin dùng, bao gồm React, Vite, và Tailwind CSS.

## React: Thư viện giao diện dựa trên Component

React là thư viện Javascript mã nguồn mở được phát triển bởi Facebook, đã trở thành tiêu chuẩn công nghiệp cho việc xây dựng giao diện người dùng [22].

- **Kiến trúc dựa trên Component:** React cho phép chia nhỏ giao diện phức tạp thành các thành phần (Components) độc lập và có thể tái sử dụng (ví dụ: ‘Navbar’, ‘Dashboard’, ‘ValuationForm’). Cách tiếp cận này giúp mã nguồn trở nên sạch sẽ, dễ quản lý và mở rộng.
- **Virtual DOM:** React sử dụng một bản sao của DOM trong bộ nhớ (Virtual DOM) để tính toán các thay đổi cần thiết trước khi cập nhật DOM thật. Cơ chế này giúp tối ưu hóa hiệu năng render, mang lại trải nghiệm mượt mà khi dữ liệu thay đổi.
- **Hệ sinh thái rộng lớn:** Với một cộng đồng khổng lồ, React có sẵn vô số thư viện hỗ trợ cho mọi nhu cầu, từ quản lý trạng thái, tạo biểu đồ (Recharts) đến xử lý animation.

## Vite: Công cụ Build thế hệ mới

Thay vì sử dụng Create React App (CRA) truyền thống, dự án lựa chọn Vite làm công cụ build. Vite mang lại những cải tiến đột phá về tốc độ phát triển:

- **Hot Module Replacement (HMR) siêu nhanh:** Vite sử dụng Native ES Modules của trình duyệt, cho phép cập nhật thay đổi trong code lên giao diện gần như tức thì mà không cần tải lại toàn bộ trang. Điều này giúp tăng tốc đáng kể chu trình phát triển.
- **Tối ưu hóa Build Production:** Khi build sản phẩm cuối cùng, Vite sử dụng Rollup để đóng gói và tối ưu hóa code (tree-shaking, code splitting), tạo ra các file Javascript và CSS có dung lượng nhỏ nhất có thể, giúp tăng tốc độ tải trang cho người dùng cuối.

## Tailwind CSS và Framer Motion: Styling và Hiệu ứng

Để tạo ra một giao diện đẹp mắt và hiện đại, nghiên cứu đã sử dụng hai công cụ chính:

- **Tailwind CSS:** Là một framework CSS theo triết lý “utility-first”[23]. Thay vì viết các file CSS riêng biệt, Tailwind cho phép style trực tiếp trong file JSX thông qua các class tiện ích (ví dụ: ‘bg-blue-500’, ‘rounded-lg’, ‘shadow-xl’). Cách tiếp cận này giúp tạo giao diện nhanh chóng, nhất quán và dễ dàng tùy chỉnh mà không bị xung đột CSS.

- **Framer Motion:** Là thư viện animation cho React, giúp việc tạo ra các hiệu ứng chuyển động phức tạp trở nên đơn giản và khai báo (declarative) [24]. Tất cả các hiệu ứng chuyển tab, hiện/ẩn component trong ứng dụng đều được xử lý bởi Framer Motion để mang lại cảm giác mượt mà và cao cấp.

### 5.4.2 Thiết kế và Triển khai các chức năng chính

Giao diện người dùng (UI) được thiết kế theo từng module chức năng, mỗi module tương ứng với một component React riêng biệt. Kiến trúc này đảm bảo sự phân tách rõ ràng giữa các tính năng và dễ dàng bảo trì.

#### Chức năng Định giá (Valuation)

Đây là chức năng trung tâm của ứng dụng, được thiết kế để người dùng có thể nhập liệu nhanh chóng và nhận kết quả ngay lập tức. Sơ đồ mockup (Hình 5.4) minh họa bố cục chính.



Hình 5.4: Mockup giao diện chức năng Định giá đơn lẻ

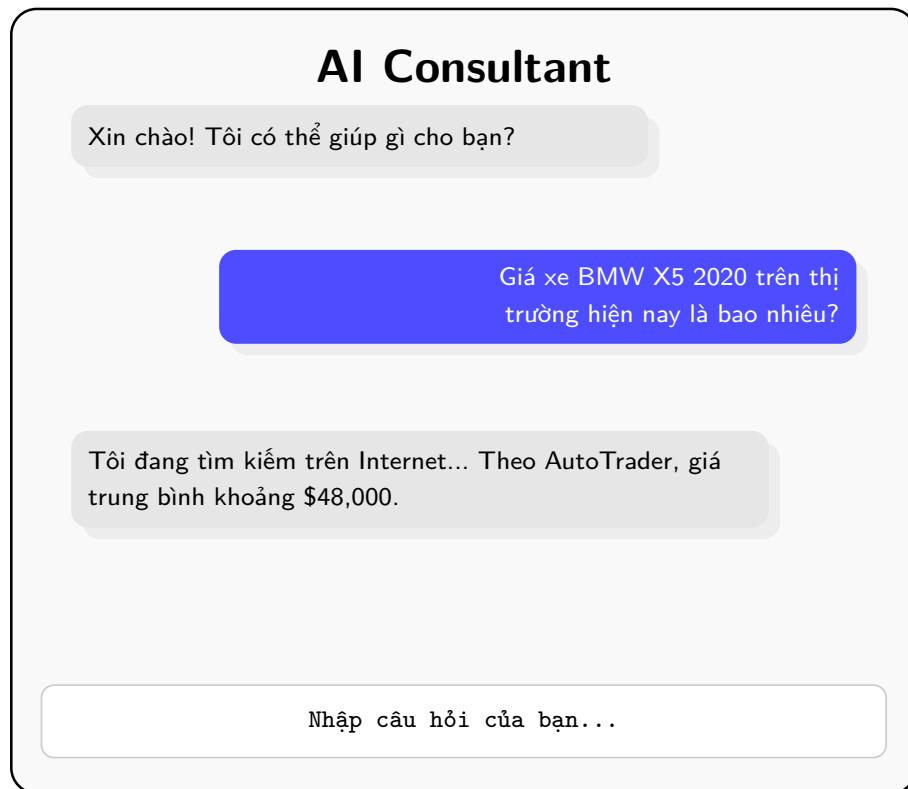
#### Chức năng Xử lý Lô (Batch)

Chức năng này đáp ứng nhu cầu của người dùng chuyên nghiệp, với quy trình 3 bước rõ ràng và giao diện kéo-thả file.

- **Bước 1: Tải file mẫu:** Cung cấp file Excel/CSV mẫu để đảm bảo định dạng dữ liệu đầu vào chính xác.
- **Bước 2: Upload file:** Người dùng kéo thả file vào vùng chỉ định.
- **Bước 3: Hiển thị kết quả:** Hiển thị bảng dữ liệu kết quả và các thẻ thống kê nhanh.

### Chức năng Trợ lý AI (AI Agent)

Giao diện Chatbot được thiết kế thân thiện, mô phỏng các ứng dụng nhắn tin phổ biến, cho phép người dùng tương tác với AI một cách tự nhiên (Hình 5.5).



Hình 5.5: Mockup giao diện Trợ lý AI

### Chức năng Phân tích (Dashboard) và Lịch sử (History)

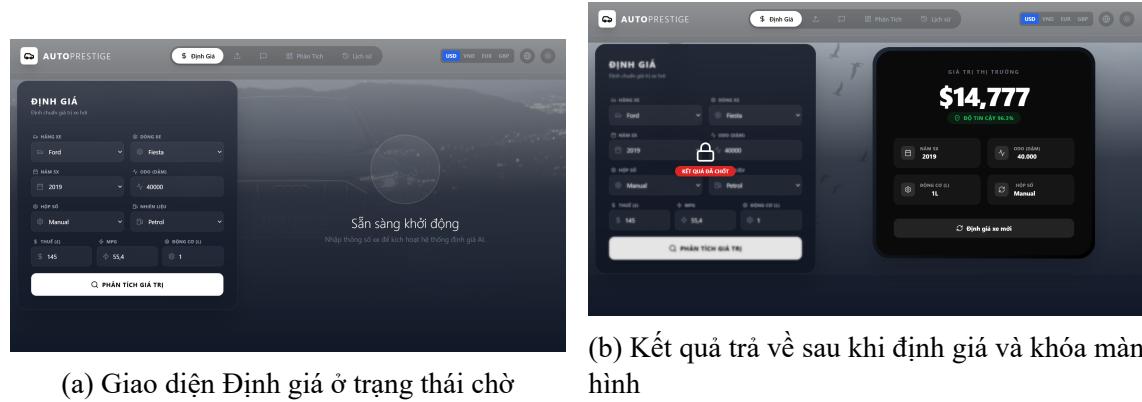
Hai chức năng này cung cấp cái nhìn tổng quan về dữ liệu đã được xử lý.

- **Dashboard:** Sử dụng thư viện Recharts để vẽ các biểu đồ động, bao gồm biểu đồ vùng (Area Chart) cho xu hướng giá và biểu đồ cột (Bar Chart) cho thị phần các hãng xe.
- **History:** Hiển thị toàn bộ lịch sử định giá dưới dạng bảng dữ liệu chi tiết, cung cấp chức năng xuất báo cáo ra file Excel.

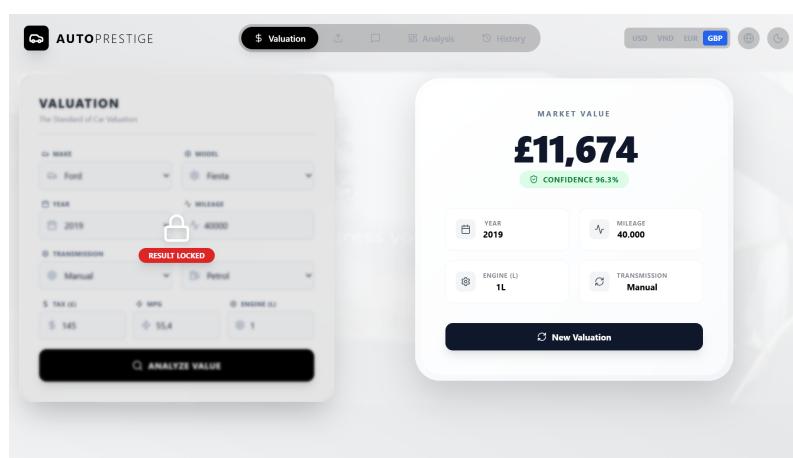
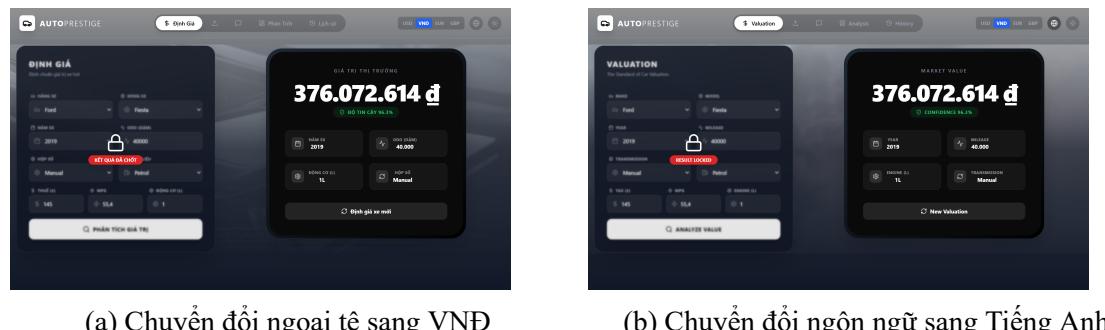
## 5.5 Demo giao diện ứng dụng

### Chức năng Định giá đơn lẻ và Tùy biến giao diện

Giao diện chính cho phép người dùng nhập thông tin xe và nhận kết quả ngay lập tức. Các tính năng tùy biến như đổi ngôn ngữ, ngoại tệ và giao diện sáng/tối đều được tích hợp.



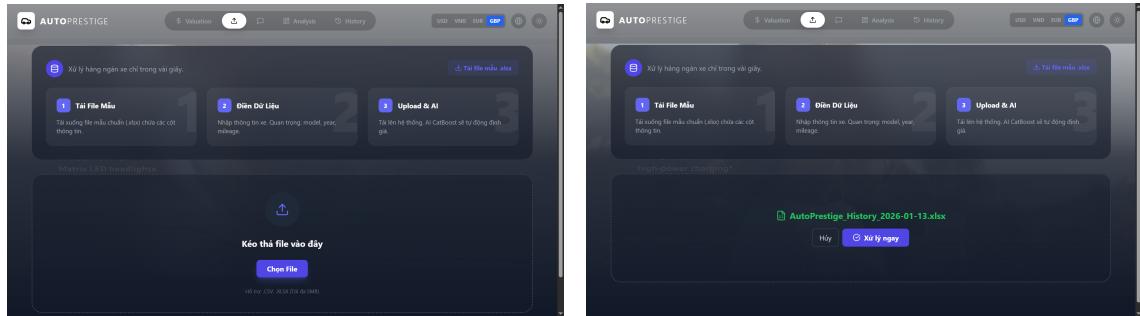
Hình 5.6: Quy trình Định giá đơn lẻ



Hình 5.7: Các tính năng tùy biến giao diện

## Chức năng Định giá Hàng loạt

Quy trình xử lý file Excel/CSV được thiết kế trực quan, từ lúc tải file lên đến khi nhận kết quả.



(a) Giao diện Xử lý Lô với khu vực kéo-thả

(b) File đã được chọn và sẵn sàng để xử lý

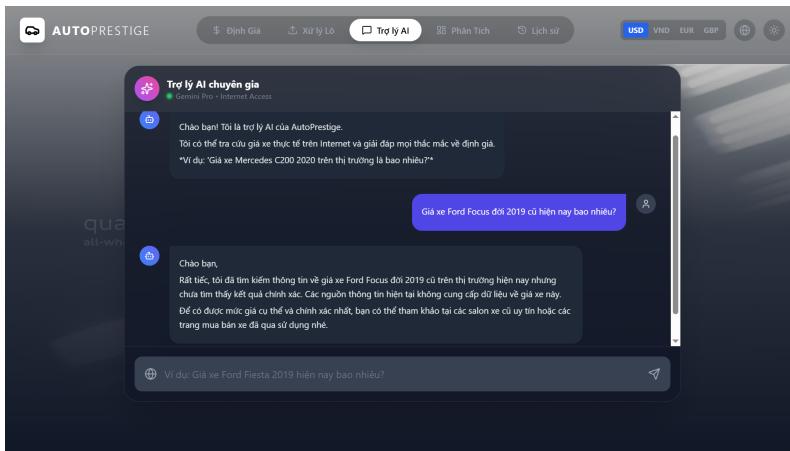
ID	DATE	MANUFACTURER	MODEL	YEAR	TRANSMISSION	MILEAGE	FUEL	ENGINE SIZE	PREDICTED VALUE
206	2026-01-13 07:25:57	Ford	Fiesta	2.019	Manual	40.000	Petrol	1	14.777
207	2026-01-13 07:25:57	BMW	X5	2.021	Semi-Auto	15.000	Diesel	3	13.894
208	2026-01-13 07:25:57	I10	hyundai	2.017	Manual	11.630	Petrol	1	14.441
209	2026-01-13 07:25:57	Polo	volkswagen	2.017	Manual	9.200	Petrol	1	14.441
210	2026-01-13 07:25:57	2 Series	BMW	2.019	Semi-Auto	1.614	Diesel	2	14.777
211	2026-01-13 07:25:57	Yeti Outdoor	skoda	2.017	Manual	30.960	Diesel	2	14.441
212	2026-01-13 07:25:57	Fiesta	ford	2.017	Manual	19.353	Petrol	1,2	14.441
213	2026-01-13 07:25:57	C-HR	toyota	2.019	Automatic	2.373	Hybrid	1,8	14.777

(c) Bảng kết quả và thống kê sau khi xử lý

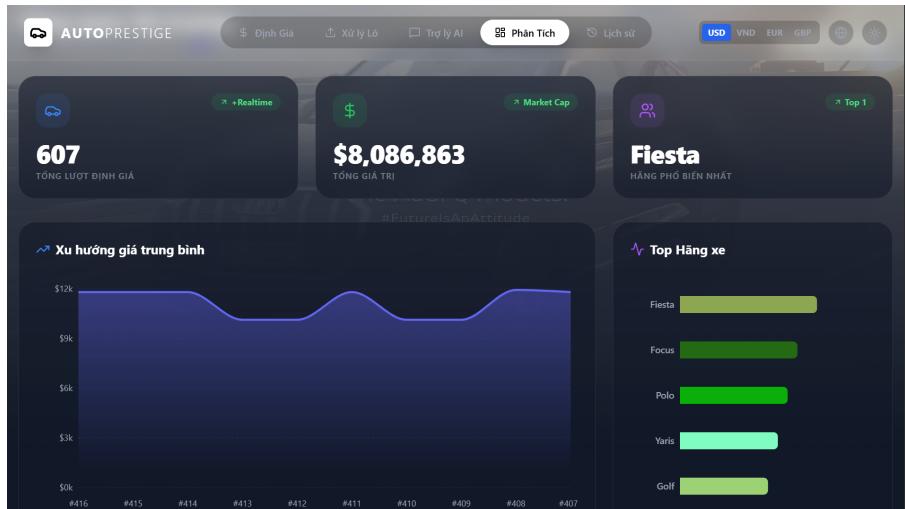
Hình 5.8: Quy trình Định giá Hàng loạt

## Các chức năng Phân tích và Tra cứu

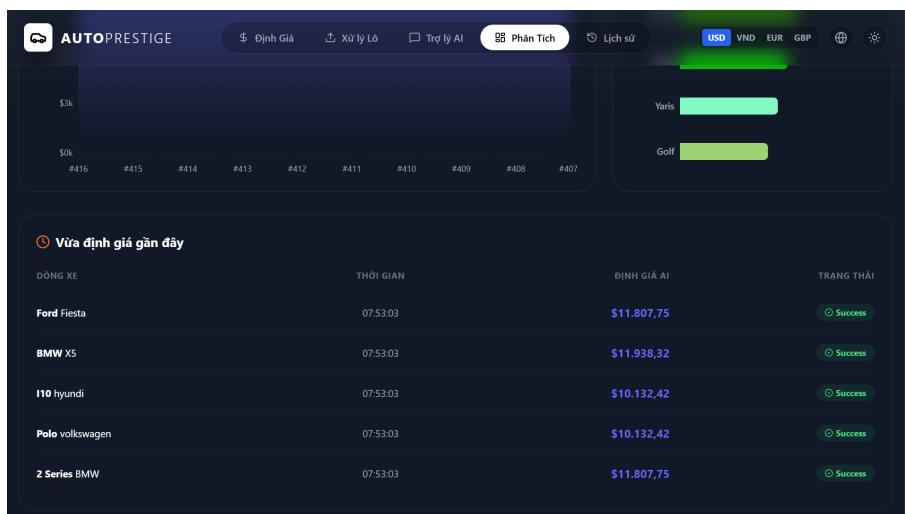
Các tab chức năng còn lại cung cấp cái nhìn sâu sắc hơn về dữ liệu và khả năng tương tác thông minh.



(a) Trợ lý AI (Gemini) tư vấn và tra cứu Internet



(b) Dashboard - Biểu đồ xu hướng giá và thống kê



(c) Dashboard - Biểu đồ hàng xe và lịch sử gần đây

Hình 5.9: Chức năng Trợ lý AI và Dashboard phân tích

ID	THỜI GIAN	DÒNG XE	THÔNG SỐ	ĐỊNH GIÁ (USD)
#407	2026-01-13 07:53:03	Ford Fiesta 2019	40.000 miles • Manual	\$11,808
#408	2026-01-13 07:53:03	BMW XS 2021	15.000 miles • Semi-Auto	\$11,938
#409	2026-01-13 07:53:03	I10 hyundai 2017	11.630 miles • Manual	\$10,132
#410	2026-01-13 07:53:03	Polo volkswagen 2017	9.200 miles • Manual	\$10,132
#411	2026-01-13 07:53:03	2 Series BMW 2019	1.614 miles • Semi-Auto	\$11,808
#412	2026-01-13 07:53:03	Yeti Outdoor skoda 2017	30.960 miles • Manual	\$10,132
#413	2026-01-13 07:53:03	Fiesta ford 2017	19.353 miles • Manual	\$10,132
#414	2026-01-13 07:53:03	C-HR toyota 2019	2.373 miles • Automatic	\$11,808
#415	2026-01-13 07:53:03	Kuga ford 2019	7.038 miles • Manual	\$11,808

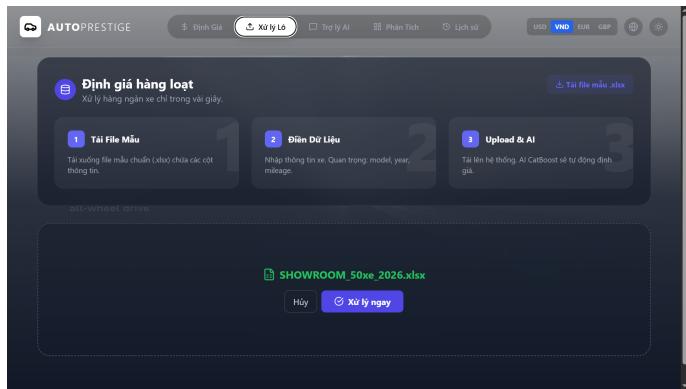
Hình 5.10: Trang Lịch sử hiển thị toàn bộ dữ liệu định giá của hệ thống

## 5.6 Kịch bản ứng dụng vào nghiệp vụ thực tế

Để chứng minh tính khả thi và giá trị ứng dụng của hệ thống AutoPrestige AI trong các bối cảnh nghiệp vụ thực tế, nghiên cứu trình bày hai kịch bản sử dụng (case study) điển hình, tương ứng với hai nhóm đối tượng người dùng chính: doanh nghiệp (B2B) và người dùng cá nhân (B2C).

### 5.6.1 Case Study 1: Tối ưu hóa hoạt động cho Showroom xe cũ

- Đối tượng:** Chủ hoặc quản lý kinh doanh của một showroom xe đã qua sử dụng.
- Tình huống (Problem):** Showroom vừa nhập về một lô hàng gồm 50 chiếc xe từ nhiều hãng và đời khác nhau. Nhiệm vụ đặt ra là phải nhanh chóng xác định giá trị thị trường của từng chiếc xe để lên kế hoạch kinh doanh và định giá niêm yết.
- Giải pháp (Solution):** Người quản lý sử dụng chức năng "Xử lý Lô" của ứng dụng AutoPrestige AI theo quy trình 3 bước:



(a) Bước 1: Người dùng tải lên file Excel chứa danh sách 50 xe.

MANUFACTURER	MODEL	YEAR	TRANSMISSION	MILEAGE	FUELTYPE	TAX	MPG	ENGINESIZE	CARAGE	MAN
Ka+	ford	2019	Manual	2.516	Petrol	145	43,5	1,2	6	0
Mokka X	vauxhall	2019	Manual	9.104	Petrol	145	39,2	1,4	6	0
Focus	ford	2020	Automatic	185	Diesel	145	54,3	1,5	5	0
X2	BMW	2019	Semi-Auto	2.198	Petrol	145	36,2	2	6	0
Polo	volkswagen	2017	Manual	24.881	Petrol	145	60,1	1	8	0
Polo	volkswagen	2016	Manual	48.511	Petrol	20	60,1	1,2	9	0
C-MAX	ford	2017	Manual	12.928	Petrol	30	55,4	1	8	0
Golf	volkswagen	2018	Manual	12.652	Diesel	145	55,4	1,6	7	0

(b) Bước 2: Hệ thống xử lý và trả về kết quả định giá cùng thống kê tổng quan ngay trên giao diện web.

1	manufacturer	model	year	transmissi	mileage	fueltype	tax	mpg	enginesize	CarAge	Manufacture	model_Cpredicted	Price_(VND)
2	Ka+	ford	2019	Manual	2516	Petrol	145	43,5	1,2	6	0	0	11807.75 3.01E+08
3	Mokka X	vauxhall	2019	Manual	9104	Petrol	145	39,2	1,4	6	0	0	11807.75 3.01E+08
4	Focus	ford	2020	Automatic	185	Diesel	145	54,3	1,5	5	0	0	11264,08 2.87E+08
5	X2	BMW	2019	Semi-Auto	2198	Petrol	145	36,2	2	6	0	0	11807.75 3.01E+08
6	Polo	volkswagen	2017	Manual	24881	Petrol	145	60,1	1	8	0	0	10132,42 2.58E+08
7	Polo	volkswagen	2016	Manual	48511	Petrol	20	60,1	1,2	9	0	0	10034,12 2.55E+08
8	C-MAX	ford	2017	Manual	12928	Petrol	30	55,4	1	8	0	0	10132,42 2.58E+08
9	Golf	volkswagen	2018	Manual	12652	Diesel	145	55,4	1,6	7	0	0	11149,79 2.84E+08
10	Ka+	ford	2019	Manual	5000	Petrol	145	43,5	1,2	6	0	0	11807.75 3.01E+08
11	GLE Class	merc	2019	Automatic	1644	Diesel	145	32,8	3	6	0	0	11807.75 3.01E+08
12	A6	Audi	2017	Automatic	64962	Diesel	145	64,2	2	8	0	0	10132,42 2.58E+08
13	Mokka	vauxhall	2013	Manual	42105	Diesel	125	57,6	1,7	12	0	0	6921,45 1.76E+08
14	4 Series	BMW	2019	Automatic	5738	Diesel	145	65,7	2	6	0	0	11807.75 3.01E+08
15	Fabia	skoda	2017	Manual	17001	Petrol	20	58,9	1,2	8	0	0	10132,42 2.58E+08
16	2 Series	BMW	2018	Automatic	29590	Petrol	145	53,3	1,5	7	0	0	11149,79 2.84E+08
17	4 Series	BMW	2016	Automatic	16955	Diesel	145	53,3	3	9	0	0	10034,12 2.55E+08
18	Mokka	vauxhall	2017	Manual	21480	Petrol	145	42,2	1,6	8	0	0	10132,42 2.58E+08
19	A5	Audi	2020	Semi-Auto	5000	Petrol	145	40,4	2	5	0	0	11264,08 2.87E+08

(c) Bước 3: Người dùng xuất báo cáo chi tiết ra file Excel để lưu trữ và phân tích sâu hơn.

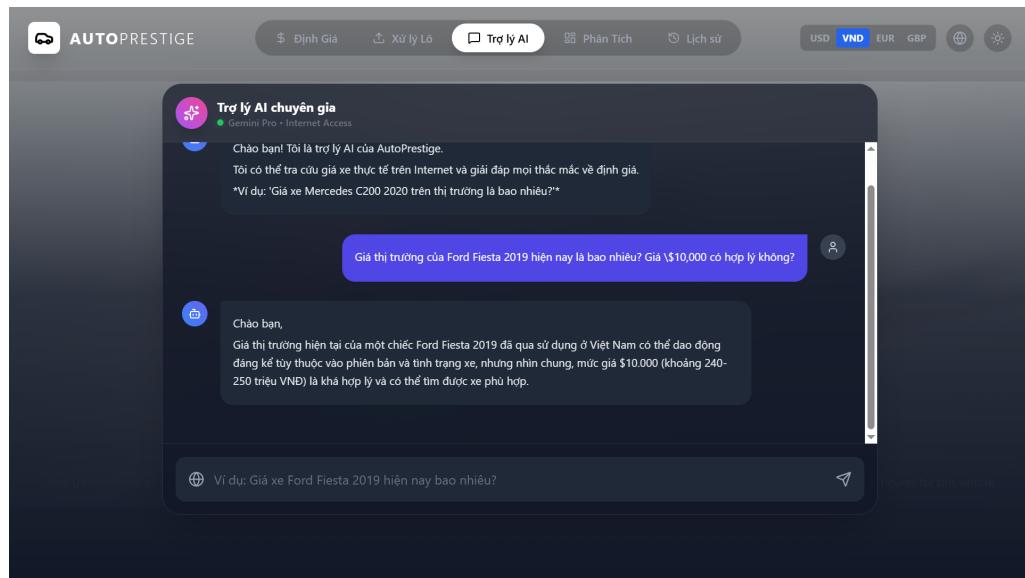
Hình 5.11: Quy trình 3 bước xử lý lô xe cho showroom

**Giá trị mang lại:**

- Tiết kiệm thời gian:** Rút ngắn quy trình định giá từ vài giờ xuống còn vài phút.
- Tăng tính nhất quán:** Mọi xe đều được định giá dựa trên cùng một mô hình AI, loại bỏ yếu tố cảm tính.
- Hỗ trợ ra quyết định:** Cung cấp dữ liệu tức thì để ban lãnh đạo đưa ra các quyết định kinh doanh nhanh chóng.

**5.6.2 Case Study 2: Hỗ trợ người mua xe cá nhân**

- Đối tượng:** Người dùng cá nhân đang tìm mua xe đã qua sử dụng.
- Tình huống (Problem):** Người dùng tìm thấy một tin rao bán xe Ford Fiesta đời 2019 với giá \$10,000 và không chắc chắn về mức giá này.
- Giải pháp (Solution):** Người dùng sử dụng chức năng "Trợ lý AI". Họ đặt câu hỏi bằng ngôn ngữ tự nhiên, và AI Agent sẽ tự động tra cứu Internet, so sánh với dữ liệu nội bộ và đưa ra lời khuyên toàn diện như trong Hình 5.12.



Hình 5.12: Trợ lý AI tư vấn và so sánh giá cho người dùng cá nhân

**Giá trị mang lại:**

- Minh bạch hóa thông tin:** Cung cấp cho người mua một nguồn tham chiếu khách quan, đáng tin cậy.

- **Giảm thiểu rủi ro:** Giúp người dùng tránh được các trường hợp mua xe với giá cao hơn giá trị thực.
- **Trải nghiệm vượt trội:** Cung cấp câu trả lời tổng hợp ngay lập tức thay vì người dùng phải tự tìm kiếm.

## 5.7 Kết luận chương 5

Chương 5 đã trình bày chi tiết quá trình chuyển đổi từ một mô hình học máy thành một sản phẩm phần mềm hoàn chỉnh và có tính ứng dụng cao. Quá trình này bao gồm ba giai đoạn chính:

1. **Lựa chọn mô hình:** Dựa trên một quy trình đánh giá khoa học, so sánh giữa các công nghệ tiên tiến như Deep Learning (TabNet) và Ensemble Learning (Stacking), mô hình **CatBoost** đã được lựa chọn làm giải pháp tối ưu nhất nhờ sự cân bằng vượt trội giữa độ chính xác ( $R^2 \approx 96.3\%$ ), tốc độ và chi phí vận hành.
2. **Thiết kế hệ thống:** Ứng dụng được xây dựng dựa trên kiến trúc 3 tầng hiện đại, với **FastAPI** đảm nhiệm vai trò Backend hiệu năng cao và **React** xây dựng giao diện người dùng (Frontend) linh hoạt, đáp ứng.
3. **Triển khai chức năng:** Hệ thống đã hiện thực hóa thành công 5 chức năng nghiệp vụ cốt lõi, bao gồm: Định giá đơn lẻ có giải thích (SHAP), Xử lý lô qua file Excel, Trợ lý AI tích hợp Internet (Gemini + LangChain), Dashboard phân tích thời gian thực và Tra cứu lịch sử.

Thông qua việc trình bày các kịch bản sử dụng thực tế, chương này đã chứng minh rằng hệ thống AutoPrestige AI không chỉ là một sản phẩm demo kỹ thuật mà còn là một công cụ hỗ trợ quyết định mạnh mẽ, sẵn sàng để triển khai và mang lại giá trị cho cả người dùng doanh nghiệp và cá nhân. Nền tảng công nghệ vững chắc được xây dựng trong chương này sẽ là tiền đề cho các hướng phát triển và mở rộng trong tương lai.

# Chương 6

## Tổng kết và Đánh giá

Trải qua các giai đoạn từ Phân tích dữ liệu, Xây dựng mô hình đến Đóng gói ứng dụng, đề tài "Dự đoán giá xe ô tô đã qua sử dụng bằng mô hình hồi quy" đã hoàn thành các mục tiêu đề ra. Chương cuối cùng này sẽ tổng hợp lại những kết quả cốt lõi đã đạt được, nhìn nhận các hạn chế còn tồn tại và đề xuất những hướng phát triển tiềm năng trong tương lai.

### 6.1 Tổng hợp kết quả đạt được

Dự án đã xây dựng thành công một Hệ thống Hỗ trợ Quyết định hoàn chỉnh, giải quyết bài toán định giá xe cũ trên cả phương diện kỹ thuật và ứng dụng nghiệp vụ.

#### 6.1.1 Về mặt Kỹ thuật (Technical Achievements)

- Xây dựng Pipeline dữ liệu chuẩn hóa:** Đã xử lý và chuẩn hóa thành công tập dữ liệu thô gồm hơn 97,000 bản ghi, tạo ra một bộ dữ liệu sạch, chất lượng cao sẵn sàng cho việc huấn luyện các mô hình học máy phức tạp.
- Thực nghiệm và đánh giá toàn diện 11 mô hình:** Một quá trình thực nghiệm khoa học đã được tiến hành, so sánh hiệu năng của 11 mô hình thuộc 5 nhóm công nghệ khác nhau (Linear, Non-linear, Tree-based, Boosting, Deep Learning).
- Xác định mô hình vô địch (Champion Model):** Mô hình CatBoost đã được chứng minh là lựa chọn tối ưu nhất, đạt độ chính xác vượt trội với  $R^2 \approx 96.3\%$  và sai số trung bình tuyệt đối  $MAE < \$1000$ , cân bằng hoàn hảo giữa hiệu năng và chi phí tính toán.

#### 6.1.2 Về mặt Ứng dụng (Application Achievements)

- Xây dựng hệ thống 3 tầng hoàn chỉnh:** Đã triển khai thành công một ứng dụng web đầy đủ chức năng với kiến trúc 3 tầng hiện đại, bao gồm Frontend (ReactJS) và Backend (FastAPI).

2. **Tích hợp 5 chức năng nghiệp vụ cốt lõi:** Ứng dụng cung cấp một bộ công cụ toàn diện cho người dùng, bao gồm: Định giá đơn lẻ (có giải thích SHAP), Xử lý lô (Excel/CSV), Trợ lý AI (tích hợp Gemini và Internet), Dashboard phân tích, và Lịch sử tra cứu.
3. **Đảm bảo tính giải thích và minh bạch (Explainable AI):** Hệ thống không chỉ đưa ra con số dự báo mà còn giải thích ”tại sao”, giúp tăng cường sự tin tưởng và hỗ trợ người dùng ra quyết định tốt hơn.

## 6.2 Hạn chế của nghiên cứu

Mặc dù đã đạt được những kết quả rất tích cực, đề tài vẫn còn tồn tại một số hạn chế cần được nhìn nhận một cách khách quan:

- **Dữ liệu tĩnh (Static Data):** Mô hình hiện tại được huấn luyện trên một bộ dữ liệu lịch sử. Nó chưa có khả năng tự động cập nhật (real-time) theo các biến động giá hàng ngày của thị trường, vốn chịu ảnh hưởng bởi các yếu tố như giá xăng, chính sách thuế, hoặc các sự kiện kinh tế vĩ mô.
- **Phạm vi địa lý (Geographic Scope):** Tập dữ liệu được sử dụng chủ yếu phản ánh thị trường xe ở Anh/Mỹ. Để áp dụng cho thị trường Việt Nam, mô hình cần được tinh chỉnh (Fine-tuning) hoặc huấn luyện lại với dữ liệu địa phương để phản ánh đúng các yếu tố đặc thù như thuế trước bạ, phí đăng kiểm, các dòng xe phổ biến và thị hiếu người tiêu dùng.

## 6.3 Hướng phát triển trong tương lai

Từ những hạn chế trên, nghiên cứu đề xuất ba hướng phát triển chính để nâng cấp hệ thống trong tương lai:

1. **Tích hợp Computer Vision để phân tích tình trạng xe:** Đây là hướng nâng cấp giá trị nhất. Phiên bản tiếp theo có thể cho phép người dùng tải lên hình ảnh ngoại thất của xe. Một mô hình nhận dạng vật thể (Object Detection) sẽ được huấn luyện để tự động phát hiện các hư hỏng như vết trầy xước, móp méo, hoặc dấu hiệu va chạm. Các thông tin này sẽ được lượng hóa và đưa vào làm đặc trưng mới, giúp mô hình định giá chính xác hơn nữa dựa trên tình trạng vật lý thực tế.
2. **Xây dựng hệ thống Crawl dữ liệu Real-time và Tự động huấn luyện lại (MLOps):** Để giải quyết vấn đề dữ liệu tĩnh, một hệ thống thu thập dữ liệu (Web Crawler) sẽ

được xây dựng để tự động quét giá xe từ các sàn thương mại điện tử lớn (ví dụ: Chotot Xe, Bonbanh). Dữ liệu mới sẽ được đưa vào một quy trình MLOps (Machine Learning Operations) để tự động huấn luyện lại (retrain) mô hình định kỳ (ví dụ: hàng tuần), đảm bảo hệ thống luôn cập nhật với xu hướng giá mới nhất của thị trường.

3. **Mở rộng sang thị trường Xe điện (Electric Vehicles - EV):** Thị trường xe điện đang phát triển nhanh chóng và có những yếu tố khác biệt so với xe động cơ đốt trong. Hướng phát triển tiếp theo sẽ là thu thập dữ liệu về xe điện và bổ sung các đặc trưng quan trọng như: **tình trạng sức khỏe pin (Battery SOH - State of Health)**, quãng đường đi được mỗi lần sạc, và phiên bản phần mềm. Điều này sẽ giúp ứng dụng đón đầu xu hướng và mở rộng sang một phân khúc thị trường đầy tiềm năng.

## 6.4 Kết luận chung

Đề tài đã hoàn thành xuất sắc mục tiêu xây dựng một Hệ thống Hỗ trợ Quyết định thông minh cho bài toán định giá xe ô tô đã qua sử dụng. Bằng việc kết hợp mô hình học máy SOTA (CatBoost) với kiến trúc phần mềm hiện đại (FastAPI, React) và các công nghệ AI tiên tiến (LangChain, SHAP), dự án đã tạo ra một sản phẩm không chỉ có độ chính xác cao mà còn có tính ứng dụng thực tiễn, giao diện thân thiện và khả năng giải thích minh bạch.

Hệ thống không chỉ là một công cụ tính toán, mà còn là một bước đệm quan trọng hướng tới việc **minh bạch hóa thị trường xe cũ**, trao quyền cho cả người mua và người bán bằng sức mạnh của dữ liệu và trí tuệ nhân tạo.

# Lời cảm ơn

---

Quá trình thực hiện bài tập lớn môn học Hệ Hỗ trợ Quyết định (MI4216) không chỉ là một nhiệm vụ học tập mà còn là một hành trình nghiên cứu đầy thử thách và ý nghĩa. Đây là cơ hội quý báu để em được hệ thống hóa lại toàn bộ kiến thức đã học, từ tiền xử lý dữ liệu, các thuật toán học máy, đến việc xây dựng một sản phẩm phần mềm hoàn chỉnh. Thông qua đề tài này, em đã rèn luyện được tư duy giải quyết vấn đề một cách có hệ thống, học được cách đổi mới và vượt qua những khó khăn kỹ thuật, và quan trọng nhất là cảm nhận được niềm vui khi biến những dòng code thành một ứng dụng có giá trị thực tiễn.

Để có được thành quả này, em xin được bày tỏ lòng biết ơn chân thành và sâu sắc nhất.

Trước hết, em xin dành sự tri ân đặc biệt tới thầy **TS. Trần Ngọc Thăng**. Thầy không chỉ là người đã tận tình truyền đạt những kiến thức chuyên sâu, mà còn là người đã khơi dậy trong em niềm đam mê với lĩnh vực Khoa học Dữ liệu. Những buổi trao đổi, những góp ý sắc bén và sự định hướng kịp thời của Thầy chính là kim chỉ nam giúp em không bị lạc lối trong quá trình nghiên cứu, từ việc lựa chọn mô hình đến cách biện luận khoa học cho kết quả của mình.

Em cũng xin trân trọng cảm ơn các thầy cô trong Khoa Toán - Tin nói riêng và toàn thể các thầy cô, cán bộ giảng viên đang công tác tại Đại học Bách khoa Hà Nội nói chung. Những kiến thức và kỹ năng mà em được trang bị dưới mái trường Bách Khoa chính là nền tảng vững chắc nhất, là hành trang không thể thiếu để em có thể tự tin chinh phục những bài toán phức tạp trong hiện tại và tương lai.

Con xin gửi lời cảm ơn tới gia đình đã luôn là hậu phương vững chắc, luôn tin tưởng, động viên và tạo mọi điều kiện tốt nhất để con có thể yên tâm học tập và theo đuổi đam mê của mình. Cảm ơn những người bạn thân thiết đã luôn ở bên cạnh, cùng nhau chia sẻ những khó khăn, trao đổi kiến thức và mang lại những giây phút thư giãn quý báu.

Và cuối cùng, tôi muốn cảm ơn chính bản thân mình đã không bỏ cuộc trước những dòng "code lỗi", những mô hình chạy không như ý, đã kiên trì tìm tòi, học hỏi và nỗ lực hết mình để hoàn thành trọn vẹn sản phẩm này.

Mặc dù đã rất cố gắng, nhưng do giới hạn về thời gian và kiến thức chuyên môn, bài báo cáo chắc chắn không thể tránh khỏi những thiếu sót. Em rất mong nhận được những ý kiến đóng góp quý báu của Thầy để có thể tiếp tục hoàn thiện hơn nữa trên con đường học tập và nghiên cứu sau này.

Em xin chân thành cảm ơn!

- HẾT -

# Phụ lục

---

## .1 Mã nguồn và Tài nguyên tham khảo

Để phục vụ cho việc kiểm tra, đánh giá và tái sử dụng nghiên cứu, toàn bộ mã nguồn, dữ liệu và mô hình đã huấn luyện của dự án được công khai tại các địa chỉ dưới đây.

## .2 Kho mã nguồn trên GitHub

Toàn bộ mã nguồn của ứng dụng, bao gồm cả Backend (FastAPI) và Frontend (React), được quản lý và lưu trữ trên nền tảng GitHub. Người dùng có thể truy cập, xem xét, hoặc clone (sao chép) toàn bộ dự án để chạy trên máy cá nhân.

- **Đường dẫn kho mã nguồn:**

<https://github.com/kiennt-ethan/DSS-CarPriceProject>

Kho mã nguồn bao gồm:

- Thư mục /backend: Chứa toàn bộ mã nguồn Python cho FastAPI server, file mô hình .cbm, file encoders .pkl và file lịch sử .json.
- Thư mục /frontend: Chứa toàn bộ mã nguồn ReactJS, các components, file dữ liệu tĩnh và cấu hình.
- File README.md: Hướng dẫn chi tiết cách cài đặt môi trường, các thư viện cần thiết và cách khởi chạy từng phần của ứng dụng.

## .3 Tài nguyên dự án trên Google Drive

Ngoài mã nguồn, các tài nguyên khác như tập dữ liệu gốc, các phiên bản mô hình đã huấn luyện, file báo cáo và slide thuyết trình cũng được lưu trữ trên Google Drive để tiện cho việc truy cập và tải xuống.

- **Đường dẫn Google Drive:**

<https://drive.google.com/drive/folders/1ci...>

Thư mục trên Google Drive bao gồm:

- **Dữ liệu (Data):** Chứa cả file dữ liệu thô (CarsData.csv) và file đã qua xử lý (CarsData\_Processed.csv).
- **Mô hình (Models):** Lưu trữ các file mô hình đã được huấn luyện trong quá trình thực nghiệm (.pkl, .cbm).
- **Báo cáo (Report):** Chứa file PDF của bài báo cáo này và các tài liệu liên quan.
- **Thuyết trình (Presentation):** Chứa file slide dùng để trình bày và bảo vệ bài tập lớn.

Việc công khai các tài nguyên này nhằm đảm bảo tính minh bạch, khả năng tái lập (reproducibility) của nghiên cứu và tạo điều kiện cho các hướng phát triển, mở rộng trong tương lai.

# Tài liệu tham khảo

---

- [1] R. Wirth and J. Hipp, “Crisp-dm: Towards a standard process model for data mining,” *Proceedings of the 4th international conference on the practical application of knowledge discovery and data mining*, 2000.
- [2] Meruvu Likith, “90,000+ Cars Data From 1970 to 2024,” Kaggle Dataset, 2024, truy cập ngày 13 tháng 01 năm 2026. [Online]. Available: <https://www.kaggle.com/datasets/meruvulikith/90000-cars-data-from-1970-to-2024>
- [3] C. M. Bishop, *Pattern recognition and machine learning*. Springer, 2006.
- [4] A. Géron, *Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow*. O'Reilly Media, 2019, chương 2: End-to-End Machine Learning Project.
- [5] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer Science & Business Media, 2009, chương 3: Linear Methods for Regression.
- [6] D. H. Wolpert and W. G. Macready, “No free lunch theorems for optimization,” *IEEE transactions on evolutionary computation*, vol. 1, no. 1, pp. 67–82, 1997.
- [7] T. Hastie, R. Tibshirani, and J. Friedman, *The elements of statistical learning: data mining, inference, and prediction*. Springer Science & Business Media, 2009.
- [8] H. Drucker, C. J. Burges, L. Kaufman, A. Smola, and V. Vapnik, “Support vector regression machines,” in *Advances in neural information processing systems*, vol. 9, 1996.
- [9] L. Breiman, “Random forests,” *Machine learning*, vol. 45, pp. 5–32, 2001.
- [10] J. H. Friedman, “Greedy function approximation: a gradient boosting machine,” *Annals of statistics*, pp. 1189–1232, 2001.
- [11] T. Chen and C. Guestrin, “Xgboost: A scalable tree boosting system,” in *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, 2016, pp. 785–794.

- [12] L. Prokhorenkova, G. Gusev, A. Vorobev, A. V. Dorogush, and A. Gulin, “Catboost: unbiased boosting with categorical features,” in *Advances in neural information processing systems*, vol. 31, 2018.
- [13] A. E. Hoerl and R. W. Kennard, “Ridge regression: Biased estimation for nonorthogonal problems,” *Technometrics*, vol. 12, no. 1, pp. 55–67, 1970.
- [14] R. Tibshirani, “Regression shrinkage and selection via the lasso,” *Journal of the Royal Statistical Society: Series B (Methodological)*, vol. 58, no. 1, pp. 267–288, 1996.
- [15] L. Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone, *Classification and Regression Trees*. Chapman and Hall/CRC, 1984.
- [16] S. O. Arik and T. Pfister, “Tabnet: Attentive interpretable tabular learning,” in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 35, no. 8, 2021, pp. 6679–6687.
- [17] R. Shwartz-Ziv and A. Armon, “Tabular data: Deep learning is not all you need,” *Information Fusion*, vol. 81, pp. 84–90, 2022.
- [18] D. H. Wolpert, “Stacked generalization,” *Neural networks*, vol. 5, no. 2, pp. 241–259, 1992.
- [19] G. Ke, Q. Meng, T. Finley, T. Wang, W. Chen, W. Ma, Q. Ye, and T.-Y. Liu, “Lightgbm: A highly efficient gradient boosting decision tree,” in *Advances in neural information processing systems*, vol. 30, 2017.
- [20] S. Ramírez, “Fastapi,” <https://fastapi.tiangolo.com/>, 2018.
- [21] H. Chase, “Langchain,” *arXiv preprint arXiv:2210.03629*, 2022.
- [22] Facebook, “React - a javascript library for building user interfaces,” <https://react.dev/>, 2013.
- [23] A. Wathan and S. Schoger, “Tailwind css - a utility-first css framework,” <https://tailwindcss.com/>, 2017.
- [24] Framer, “Framer motion - a production-ready motion library for react,” <https://www.framer.com/motion/>, 2019.

- [25] TS. Trần Ngọc Thăng, “Bài giảng học phần hệ hỗ trợ quyết định (decision support systems),” Khoa Toán-Tin - Đại học Bách Khoa Hà Nội, 2025, tài liệu giảng dạy nội bộ.
- [26] ThSKH. Nguyễn Danh Tú, “Bài giảng kho dữ liệu và kinh doanh thông minh,” Khoa Toán-Tin - Đại học Bách Khoa Hà Nội, 2025, tài liệu giảng dạy nội bộ.
- [27] TS. Trần Thị Thu Hà, *Giáo trình Hệ thống thông tin hỗ trợ ra quyết định.* Hà Nội: NXB Đại học Kinh tế Quốc dân, 2020.