HUST

ĐẠI HỌC BÁCH KHOA HÀ NỘI

HANOI UNIVERSITY OF SCIENCE AND TECHNOLOGY

ONE LOVE. ONE FUTURE.

ĐẠI HỌC
BÁCH KHOA HÀ NỘI
HANOI UNIVERSITY
OF SCIENCE AND TECHNOLOGY

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

BTL học phần: Hệ hỗ trợ quyết định (MI4216)

Sinh viên thực hiện:

Nguyễn Trung Kiên - 20227180

GVHD: TS. Trần Ngọc Thăng

KHOA TOÁN - TIN

ONE LOVE. ONE FUTURE.

Nội dung trình bày I

Chương 1: Tổng quan về bài toán (Overview)

1.1.1. Bối cảnh thị trường (Market Context)

Sự dịch chuyển xu hướng:

- Thị trường ô tô đã qua sử dụng (*Used Car*) đang tăng trưởng mạnh mẽ, quy mô giao dịch lớn gấp 2-3 lần xe mới tại các quốc gia phát triển.
- **Động lực:** Nhu cầu sở hữu phương tiện cá nhân tăng cao và xu hướng tối ưu hóa chi phí khấu hao.

Đặc thù

Giá xe cũ không cố định (Fixed Price) mà biến động liên tục theo thời gian, tình trạng xe và quy luật cung cầu.

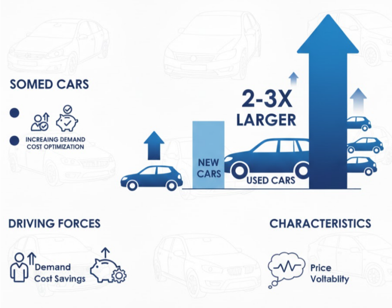


Figure: Xu hướng thị trường

Những "điểm nghẽn" trong định giá truyền thống:

1. **Thông tin bất cân xứng (Asymmetry):** Người bán thường nắm rõ tình trạng xe hơn người mua ("Thị trường quả chanh").
2. **Định giá cảm tính (Subjectivity):** Phụ thuộc hoàn toàn vào kinh nghiệm thợ xe, thiếu cơ sở dữ liệu thống kê định lượng.
3. **Thiếu tham chiếu chuẩn:** Khó xác định đâu là mức giá công bằng (Fair Price) cho một cấu hình xe cụ thể.

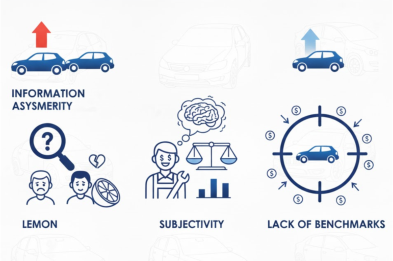


Figure: Rủi ro trong giao dịch

1.2.1. Bài toán Nghiệp vụ (Business Problem)

Mục tiêu cốt lõi

Xây dựng hệ thống hỗ trợ định giá nhằm tối ưu hóa lợi ích kinh tế và minh bạch hóa thông tin cho các bên tham gia (Người mua, Người bán, Sàn giao dịch).

Câu hỏi Kinh doanh (Questions):

1. Làm thế nào để định giá một chiếc xe cũ chính xác nhất dựa trên thị trường?
2. Yếu tố nào (Thương hiệu, Số dặm...) ảnh hưởng lớn nhất đến sự mất giá?
3. Mức độ sai số bao nhiêu là chấp nhận được để đảm bảo thanh khoản?

Yêu cầu Nghiệp vụ (Requirements):

- **Khách quan:** Định giá dựa trên dữ liệu lịch sử, không cảm tính.
- **Nhanh chóng:** Cung cấp giá gợi ý theo thời gian thực (Real-time).
- **Đa chiều:** Phân tích được xu hướng giá theo các thuộc tính khác nhau.

Định nghĩa toán học

Đây là bài toán **Học máy có giám sát - Hồi quy (Regression)**.

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

Trong đó: $\mathbf{X}$ là vector đặc trưng, Y là giá xe (liên tục), ϵ là sai số.

Câu hỏi Kỹ thuật:

1. Xử lý các biến phân loại có số lượng giá trị lớn (High Cardinality) như *Model*, *Make* thế nào?
2. Mô hình hồi quy nào tối ưu nhất cho dữ liệu dạng bảng hỗn hợp?

Yêu cầu Kỹ thuật:

- **Độ chính xác:** Tối thiểu hóa hàm mất mát (RMSE, MAE).
- **Hiệu năng:** Thời gian huấn luyện và suy diễn nhanh.
- **Tính giải thích:** Có khả năng chỉ ra mức độ quan trọng của đặc trưng (Feature Importance).

INPUT VECTOR (X)

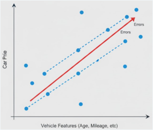
1. Định danh (Identity): *Hãng (Make), Dòng xe (Model)* → Quyết định phân khúc và giá trị thương hiệu.

2. Hiện trạng (Condition): *Năm SX (Year), Số dặm (Mileage)* → Phản ánh mức độ khấu hao và hao mòn.

3. Thông số KT (Specs): *Động cơ, Hộp số, Nhiên liệu* → Ảnh hưởng đến chi phí vận hành và hiệu năng.

PROCESS (*f*)

REGRESSION MODEL



Học mối quan hệ phi tuyến tính

OUTPUT (*Y*)

$

PRICE

(Giá trị thực liên tục)

1.4. Phương pháp tiếp cận

Nghiên cứu thực hiện đánh giá trên diện rộng với 4 nhóm giải thuật:

I. Nhóm Tuyến tính (Linear):

1. Linear Regression (Baseline)
2. Ridge Regression (L2)
3. Lasso Regression (L1)

II. Nhóm Phi tuyến & Khoảng cách:

4. K-Nearest Neighbors (KNN)
5. Support Vector Regression (SVR)

III. Nhóm Cây quyết định:

6. Decision Tree Regressor
7. Random Forest (Ensemble/Bagging)

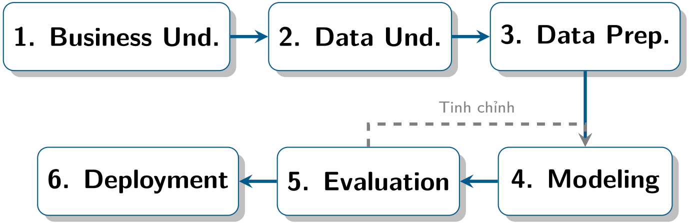
IV. Nhóm Boosting (SOTA):

8. Gradient Boosting Machine (GBM)
9. XGBoost (eXtreme Gradient Boosting)
10. **CatBoost** (Categorical Boosting)

**Nhóm IV (đặc biệt là CatBoost) được kỳ vọng xử lý tối ưu các biến định danh (Hãng, Dòng xe) mà không cần mã hóa phức tạp.*

1.5. Quy trình thực hiện (Process)

Áp dụng quy trình chuẩn công nghiệp **CRISP-DM**:



Trọng tâm nghiên cứu:

- Bước 3 (Xử lý dữ liệu).
- Bước 4 (So sánh 10 mô hình).

Sản phẩm đầu ra:

- Báo cáo phân tích.
- Ứng dụng Web Demo (Streamlit).

Chương 2: Khai phá dữ liệu (Data Mining)

Tổng quan:

- **Nguồn:** Kaggle - *90,000+ Cars Data From 1970 to 2024*.
- **Kích thước:** 97,712 bản ghi, 10 thuộc tính.
- **Đặc điểm:** Dữ liệu hỗn hợp (Số thực & Định danh), có chứa nhiễu và giá trị thiếu.

Mô tả các biến chính:

Tên cột	**Kiểu dữ liệu**	**Ý nghĩa**
`model`, `Manufacturer`	Object (Categorical)	Tên dòng xe và Hãng sản xuất
`transmission`, `fuelType`	Object (Categorical)	Hộp số và Loại nhiên liệu
`price`	Int64 (Target)	Giá bán (Biến mục tiêu)
`mileage`, `year`	Int64	Số dặm đã đi, Năm sản xuất
`tax`, `mpg`, `engineSize`	Float64	Thuế, Mức tiêu thụ, Dung tích

5 dòng đầu tiên của tập dữ liệu

	model	year	price	transmission	mileage	fuelType	tax	mpg	engineSize	Manufacturer
0	I10	2017	7495	Manual	11630	Petrol	145	60.1	1.0	hyundi
1	Polo	2017	10989	Manual	9200	Petrol	145	58.9	1.0	volkswagen
2	2 Series	2019	27990	Semi-Auto	1614	Diesel	145	49.6	2.0	BMW
3	Yeti Outdoor	2017	12495	Manual	30960	Diesel	150	62.8	2.0	skoda
4	Fiesta	2017	7999	Manual	19353	Petrol	125	54.3	1.2	ford

Figure: Mẫu dữ liệu trước khi xử lý

Biến định lượng (Numerical):

- **Price:** Dao động lớn (Min: 450 - Max: 159,999). Độ lệch chuẩn cao ($\sigma \approx 9868$).
- **Mileage:** Phân tán rộng (Max: 323,000 dặm), phản ánh độ hao mòn đa dạng.

Biến định danh (Categorical):

- **Model:** 196 dòng xe khác nhau (High Cardinality).
- **Top Frequency:** Ford Fiesta (6,509 xe), Số sàn (Manual), Máy xăng (Petrol).

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

Figure: Bảng thống kê mô tả (Biến số)

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

Figure: Bảng thống kê biến phân loại

2.4. Phân tích tương quan (Correlation Analysis)

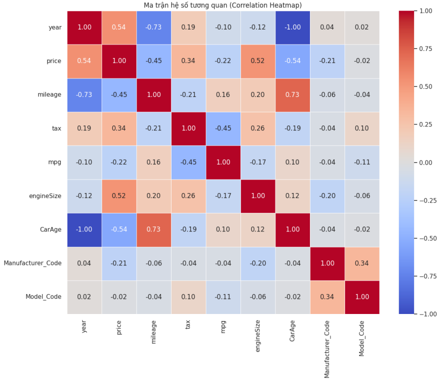


Figure: Heatmap tương quan giữa các biến

Nhận xét từ dữ liệu:

+ **Tương quan Dương (+):**
  - `year` vs `price`: Xe càng mới giá càng cao.
  - `engineSize` vs `price`: Động cơ lớn thường là xe phân khúc cao.
- **Tương quan Âm (-):**
  - `mileage` vs `price`: Xe đi càng nhiều, giá trị càng giảm (hao mòn).

! **Đa cộng tuyến:**
  - Cần chú ý nếu các biến độc lập tương quan quá mạnh với nhau (ví dụ `tax` và `engineSize`) để tránh gây nhiễu cho mô hình tuyến tính.

2.5. Phân tích và Xử lý Nhiễu (Data Cleaning)

Vấn đề phát hiện

- **Missing Values:** Một số bản ghi thiếu thông tin `tax`, `mpg`.
- **Outliers (Price):** Tồn tại các xe giá cực thấp ($< \$1000$) hoặc siêu xe giá quá cao, gây nhiễu cho mô hình hồi quy.

Giải pháp xử lý:

1. **Imputation:** Điền *Median* cho biến số, *Mode* cho biến phân loại.
2. **Outliers Removal:** Sử dụng phương pháp IQR (Interquartile Range).
    - Loại bỏ các giá trị nằm ngoài khoảng $[Q1 - 1.5IQR, Q3 + 1.5IQR]$.
    - **Kết quả:** Loại bỏ 3,825 bản ghi nhiễu (còn lại 93,887 xe).

Hiện trạng (Before):

- Biến `price` bị lệch phải (Right-skewed).
- Đa số xe có giá thấp, đuôi phân phối kéo dài về phía giá cao.

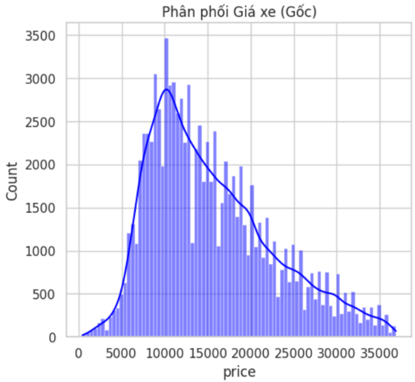


Figure: Biểu đồ phân phối gốc

Xử lý (After):

- Áp dụng **Log Transformation**: $Y' = \log(Y + 1)$.
- Phân phối trở nên chuẩn (Gaussian-like) hơn, giúp mô hình học tốt hơn.

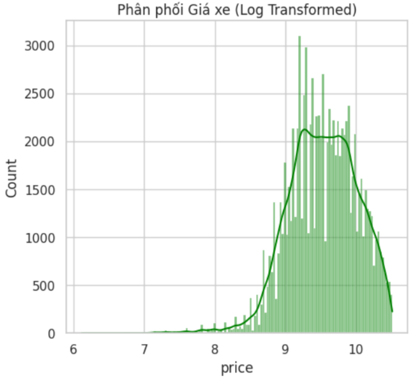


Figure: Biểu đồ Log Transform

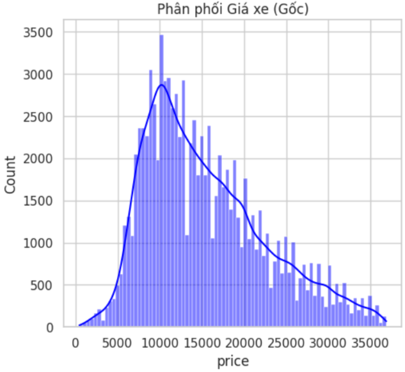


Figure: Biểu đồ phân phối gốc

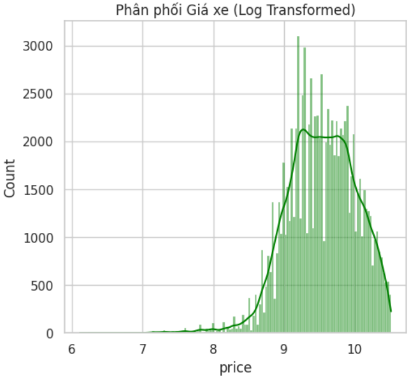


Figure: Biểu đồ Log Transform

1. **Tạo đặc trưng mới (Feature Engineering):**
   - Tạo biến `CarAge` (Tuổi xe): $CarAge = 2025 - Year$.
   - *Lý do:* Tuổi xe phản ánh trực tiếp mức độ khấu hao hơn là năm sản xuất.
2. **Mã hóa biến phân loại (Encoding):**
   - **One-Hot Encoding:** Cho biến ít giá trị (`transmission`, `fuelType`). → Tạo ra các cột *transmission_Manual*, *fuelType_Petrol*...
   - **Label Encoding:** Cho biến nhiều giá trị (`Manufacturer`, `model`). → Chuyển thành mã số (`Manufacturer_Code`).

Chuẩn hóa (Scaling):

- Sử dụng **MinMaxScaler** cho các biến số (`mileage`, `tax`, `mpg`, `engineSize`, `CarAge`).
- Đưa dữ liệu về khoảng $[0, 1]$ để tránh việc các biến có giá trị lớn lấn át biến nhỏ.

Dữ liệu sẵn sàng (Processed Data)

	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	0.158730	hyundi	0.129630	True	False	False	False	False	False	True	3	78
1	Polo	2017	10989	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	0.004994	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	7999	0.059913	0.215517	0.114772	0.190476	ford	0.129630	True	False	False	False	False	False	True	2	58

Figure: Dữ liệu đầu vào cho mô hình

Chương 3: Thiết lập thực nghiệm

Để đánh giá toàn diện hiệu quả mô hình, bài thực hiện sử dụng bộ 4 chỉ số tiêu chuẩn:

1. RMSE
(Root Mean Squared Error)

$$\sqrt{\frac{1}{n}\sum(y_i - \hat{y}_i)^2}$$

- Phạt nặng sai số lớn.
- Tránh dự đoán "thảm họa".

2. MAE
(Mean Absolute Error)

$$\frac{1}{n}\sum|y_i - \hat{y}_i|$$

- Sai số trung bình thực tế ($).
- Dễ giải thích nghiệp vụ.

3. MAPE
(Mean Abs. % Error)

$$\frac{1}{n}\sum\left|\frac{y_i - \hat{y}_i}{y_i}\right|$$

- Sai số theo phần trăm.
- Chuẩn hóa theo quy mô giá.

4. R^2 Score
(Coefficient of Determination)

$$1 - \frac{SS_{res}}{SS_{tot}}$$

- Độ phù hợp tổng thể.
- Mục tiêu: > 0.85.

Bên cạnh độ chính xác, tính khả thi khi triển khai (Deployment) được đánh giá qua:

- **Thời gian Huấn luyện (Training Time):**
  Quan trọng khi cần cập nhật mô hình định kỳ.
- **Thời gian Suy diễn (Inference Time):**
  Thời gian trả về kết quả cho 1 yêu cầu. **Yêu cầu:** $< 100ms$ để đảm bảo trải nghiệm Web App.
- **Khả năng Giải thích (Interpretability):**
  Mô hình phải chỉ ra được yếu tố nào (Năm SX, Số dặm...) ảnh hưởng lớn nhất (Feature Importance).

3.3. Trích xuất và Phân tích danh sách lỗi (Error Analysis)

Quy trình phân tích các trường hợp dự đoán sai lệch lớn nhất (*Worst-case Analysis*):

Các bước thực hiện:

1. Tính toán phần dư (Residuals): $e_i = y_{thuc\_te} - \hat{y}_{du\_doan}$.
2. Lọc ra **Top 5%** các mẫu có sai số tuyệt đối $|e_i|$ cao nhất.
3. Phân tích nguyên nhân:
   - Do dữ liệu nhiễu (Outliers)?
   - Do đặc thù dòng xe hiếm (Rare models)?
   - Do mô hình bị lệch (Bias)?

3.4. Minh họa: Bảng phân tích các ca dự đoán sai

Table: Ví dụ các trường hợp có sai số lớn trên tập kiểm thử

ID	Model	Giá Thực ($)	Giá Dự Đoán ($)	Sai Số ($)	Nhận định nguyên nhân
#1024	Audi R8	120,000	95,000	-25,000	Xe thể thao hiếm gặp, ít dữ liệu train.
#5521	Ford Fiesta	5,000	8,500	+3,500	Xe đời cũ nhưng mileage thấp bất thường.
#8812	BMW X5	45,000	42,000	-3,000	Sai số trong biên độ chấp nhận được.

3.5. Chia tập dữ liệu (Data Splitting Strategy)

Dữ liệu được chia thành 3 phần độc lập để đảm bảo tính khách quan:

- **Training Set (70%):** Huấn luyện mô hình học trọng số (68,398 bản ghi).
- **Validation Set (15%):** Tinh chỉnh siêu tham số (14,657 bản ghi).
- **Test Set (15%):** Đánh giá hiệu năng cuối cùng (14,657 bản ghi).

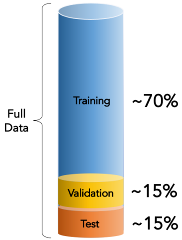


Figure: Tỷ lệ phân chia 70/15/15

3.6. Kiểm thử chéo (K-Fold Cross-Validation)

Để đảm bảo độ ổn định, nghiên cứu áp dụng **5-Fold Cross-Validation** trên tập Train:

Fold 1	Test	Train	Train	Train	Train
Fold 2	Train	Test	Train	Train	Train

... Lặp lại 5 lần ...

**Kết quả cuối cùng là trung bình cộng của 5 lần chạy.*

Chương 4: Xây dựng và Đánh giá mô hình

4.0. Chiến lược thực nghiệm (Modeling Strategy)

Để giải quyết bài toán hồi quy với dữ liệu hỗn hợp (số và phân loại), nhóm nghiên cứu thực hiện đánh giá trên **10 mô hình** theo lộ trình từ đơn giản đến phức tạp:

I. Nhóm Tuyến tính (Baseline)

1. **Linear Regression** (Mô hình cơ sở)
2. Ridge Regression (L2)
3. Lasso Regression (L1)

II. Nhóm Phi tuyến & Khoảng cách

4. K-Nearest Neighbors (KNN)
5. Support Vector Regression (SVR)

III. Nhóm Cây quyết định

6. Decision Tree Regressor
7. Random Forest Regressor

IV. Nhóm Tăng cường (SOTA)

8. Gradient Boosting (GBM)
9. XGBoost
10. **CatBoost (State-of-the-Art)**

Tổng quan:

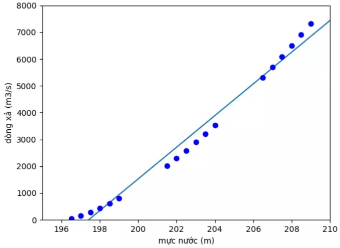
- Là mô hình học máy cơ sở, giả định mối quan hệ tuyến tính: $Y = \mathbf{w}^T\mathbf{x} + b$.
- Mục tiêu: Thiết lập mức chuẩn (Baseline) về độ chính xác và tốc độ.

Kỹ thuật áp dụng:

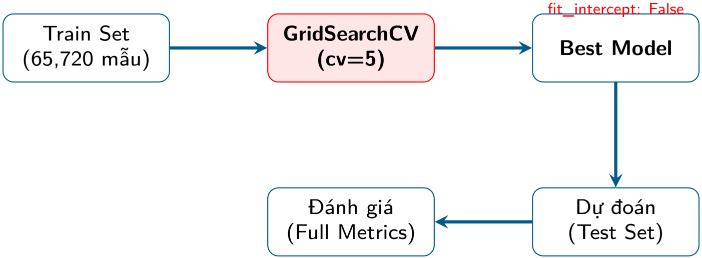
- **Thư viện:** `sklearn.linear_model.LinearRegression`.
- **Tối ưu hóa:** Sử dụng **GridSearchCV** (5-Fold CV) để tìm siêu tham số.
- **Dữ liệu:** 8 đặc trưng số (*Year, Mileage, Tax, MPG...*) sau khi đã xử lý.

Không gian tham số

- `fit_intercept`: [True, False]
- `positive`: [True, False]



4.1.2. Quy trình thực hiện (Hyperparameter Tuning)



Kết quả Tuning:

- Tham số tối ưu: `{'fit_intercept': False, 'positive': False}`.
- *Ý nghĩa:* Dữ liệu đã chuẩn hóa nên mô hình không cần hệ số chặn (bias).

4.1.3. Kết quả đánh giá trên tập Test

Thời gian huấn luyện: 3.45 giây (Rất nhanh).

Chỉ số	**Giá trị**	**Đánh giá nghiệp vụ**
RMSE	**3,865.78**	Sai số lớn, chịu ảnh hưởng bởi các giá trị ngoại lai.
MAE	**2,983.33**	Trung bình mỗi xe dự đoán lệch khoảng **$3,000**.
MedAE	2,441.12	Sai số trung vị thấp hơn MAE → Phân phối lỗi bị lệch.
MAPE	**22.59%**	**Sai số tương đối cao**. Chưa đạt yêu cầu ứng dụng (<15%).
R^2 **Score**	**0.7114**	Giải thích được **71.1%** sự biến thiên của giá xe.

Biểu đồ Scatter (Thực tế vs Dự đoán):

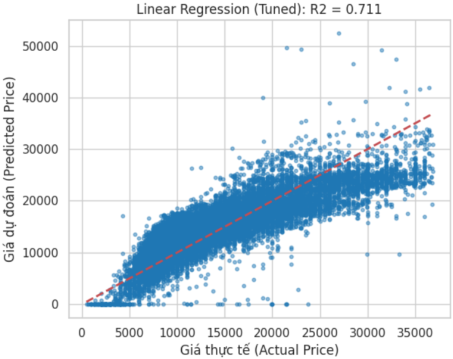


Figure: Độ phân tán dữ liệu ($R^2 = 0.71$)

Phân tích Top sai số (Worst Cases):

- **Case 1:** Thực tế 21,495 - Dự đoán 49,677 $\rightarrow$ Lệch 28k.
- **Case 2:** Thực tế 23,000 - Dự đoán 49,355 $\rightarrow$ Lệch 26k.

Nhận định

Mô hình có xu hướng **dự đoán quá cao (Overestimation)** đối với một số dòng xe cụ thể, dẫn đến sai số rất lớn (lên tới 28,000 USD).

4.1.5. Đánh giá Mô hình 1 (Baseline)

Điểm mạnh:

+ Tốc độ huấn luyện và suy diễn cực nhanh (Inference: 0.00s).
+ Dễ dàng triển khai và giải thích.
+ Thiết lập được mốc chuẩn $R^2 > 0.7$.

Điểm yếu:

- **MAPE cao (22.6%):** Rủi ro tài chính lớn.
- **Underfitting:** Không bắt được mối quan hệ phi tuyến phức tạp.
- Nhạy cảm với nhiễu, tạo ra các dự đoán sai lệch lớn.

$\Rightarrow$ **Kết luận:** Linear Regression chưa đủ tốt để làm sản phẩm cuối cùng.
Cần chuyển sang các mô hình **Regularization (Ridge/Lasso)** hoặc **Phi tuyến**.

Giới thiệu:

- Là phiên bản cải tiến của Linear Regression.
- Bổ sung thành phần **L2 Regularization** vào hàm mất mát để kiểm soát độ lớn của các hệ số hồi quy.

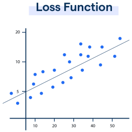
Mục tiêu kỹ thuật:

- Giảm thiểu hiện tượng **Đa cộng tuyến** (Multicollinearity).
- Tìm kiếm sự cân bằng giữa Độ lệch (Bias) và Phương sai (Variance).

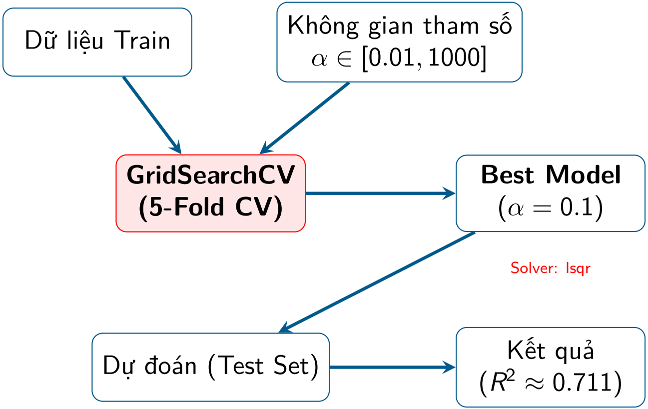
Hàm mất mát (Loss Function)

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

- α (Alpha): Hệ số phạt.
- $\alpha \to 0$: Trở về OLS chuẩn.



4.2.2. Quy trình thực hiện (Mô hình 2)



Kết quả Tuning: Tìm được $\alpha = 0.1$ (rất nhỏ). Điều này cho thấy mô hình ít bị Overfitting nên mức phạt không cần lớn.

4.2.3. Kết quả thực nghiệm (So sánh với Baseline)

Thời gian huấn luyện: 4.91 giây (Lâu hơn Linear do quá trình Cross-Validation).

Tiêu chí	**Linear (M1)**	**Ridge (M2)**	**Đánh giá**
MAE	2,983.33	**2,985.47**	Tăng nhẹ (Kém hơn)
RMSE	3,865.78	**3,867.28**	Tăng nhẹ
MAPE	22.59%	**22.61%**	Tương đương
R^2 **Score**	**0.7114**	**0.7112**	**Giảm 0.0002**

Nhận định khoa học

Việc thêm Regularization (Ridge) **không cải thiện** hiệu năng so với Linear Regression. Điều này khẳng định rằng sai số của mô hình đến từ **High Bias** (mô hình quá đơn giản so với dữ liệu thực tế) chứ không phải do High Variance (Overfitting).

a) Scatter Plot (Ridge):

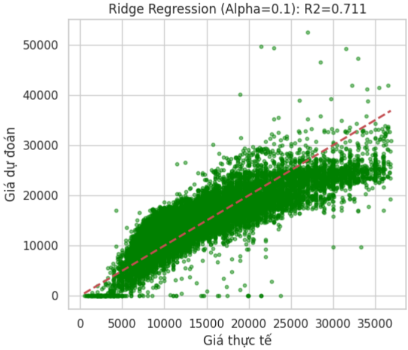


Figure: Phân bố giá dự đoán

b) Đánh giá tổng quát:

+ **Ưu điểm:** Ổn định các hệ số hồi quy trong trường hợp đa cộng tuyến.
- **Nhược điểm:** Vẫn bị giới hạn bởi tính tuyến tính. Ngưỡng $R^2 \approx 71\%$ là giới hạn trần của nhóm Linear Models.

$\Rightarrow$ **Quyết định:** Chuyển sang thử nghiệm nhóm mô hình **Cây quyết định (Tree-based)** để xử lý các mối quan hệ phi tuyến.

4.3.1. Mô hình 3: Lasso Regression (Giới thiệu)

Giới thiệu:

- Viết tắt của: *Least Absolute Shrinkage and Selection Operator*.
- Sử dụng **L1 Regularization** (giá trị tuyệt đối) trong hàm mất mát.

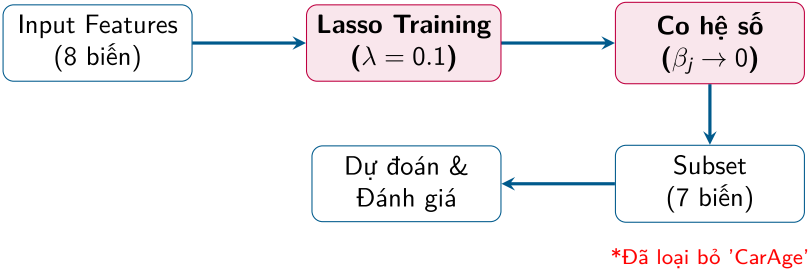
Mục tiêu kỹ thuật:

- **Chọn lọc đặc trưng (Feature Selection):** Lasso có khả năng ép hệ số của các biến không quan trọng về **đúng bằng 0**.
- Tạo ra mô hình thưa (Sparse Model), gọn nhẹ và dễ giải thích hơn.

Hàm mất mát

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

- λ: Hệ số phạt.
- Tạo ra nghiệm tại các góc nhọn (Zero coefficients).



Cơ chế: Lasso phát hiện đa cộng tuyến giữa `Year` và `CarAge` (do $CarAge = 2025 - Year$), từ đó loại bỏ biến dư thừa để tránh nhiễu.

4.3.3. Kết quả thực nghiệm (Mô hình 3)

Thông số tối ưu: $\alpha = 0.1$ (Cyclic solver).
Thời gian huấn luyện: 30.25 giây (Lâu nhất trong nhóm tuyến tính).

So sánh với các mô hình trước:

Tiêu chí	Linear (M1)	Ridge (M2)	Lasso (M3)	Nhận xét
MAE	2,983.33	2,985.47	**2,983.58**	Tương đương
RMSE	3,865.78	3,867.28	**3,865.90**	Tương đương
MAPE	22.59%	22.61%	**22.59%**	Không đổi
R^2 **Score**	**0.7114**	**0.7112**	**0.7114**	**Bão hòa**

→ *Kết luận: Các mô hình tuyến tính đều hội tụ về cùng một kết quả. Việc loại bỏ biến 'CarAge' không làm giảm độ chính xác nhưng giúp mô hình gọn hơn.*

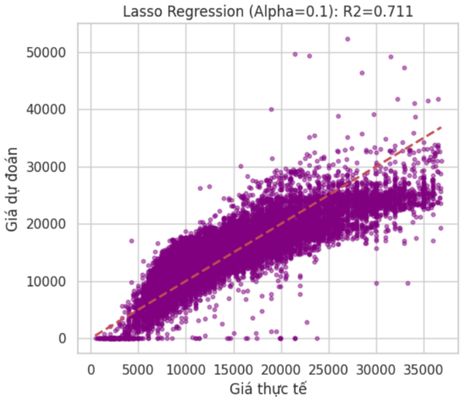


Figure: Trực quan hóa dữ liệu

Đánh giá Mô hình 3:

+ **Ưu điểm:** Tự động loại bỏ biến dư thừa (`CarAge`), giảm độ phức tạp tính toán cho khâu dự đoán.
- **Nhược điểm:** Tốc độ huấn luyện chậm (30s so với 0.7s của Linear). Không cải thiện độ chính xác (R^2 kẹt ở 0.71).

KẾT THÚC NHÓM TUYẾN TÍNH

Cả 3 mô hình (Linear, Ridge, Lasso) đều dừng lại ở $R^2 \approx 0.71$. Điều này chứng minh mối quan hệ Giá xe là **PHI TUYẾN TÍNH**.

$\Rightarrow$ Chuyển sang **Phi tuyến (KNN, SVR)** hoặc **Cây quyết định**.

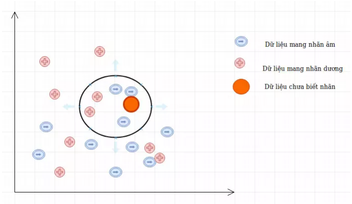
4.4.1. Mô hình 4: KNN Regressor (K-Láng giềng gần nhất)

1. **Nguyên lý hoạt động:**
   - Là thuật toán **Instance-based** (dựa trên TH).
   - Định giá một chiếc xe mới bằng cách tìm ra **K chiếc xe tương tự nhất** trong dữ liệu lịch sử và tổng hợp giá của chúng.
2. **Kỹ thuật áp dụng:**
   - **Khoảng cách:** Manhattan ($p = 1$) hoặc Euclidean ($p = 2$).
   - **Trọng số (Weights):** `distance` (Xe giống hơn có tiếng nói lớn hơn).
   - **Yêu cầu bắt buộc:** Dữ liệu phải được Chuẩn hóa (Scaling) để tính khoảng cách chính xác.

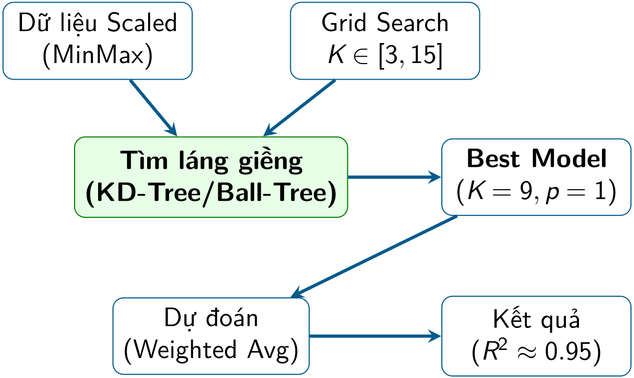
Công thức dự đoán

$$\hat{y} = \frac{\sum_{i=1}^{K} w_i y_i}{\sum_{i=1}^{K} w_i}$$

(w_i: Trọng số dựa trên nghịch đảo khoảng cách)



4.4.2. Quy trình thực hiện (Mô hình 4)



Kết quả Tuning:

- Tham số tối ưu: `{'n_neighbors': 9, 'weights': 'distance', 'p': 1}`.
- Sử dụng khoảng cách **Manhattan ($p = 1$)** hiệu quả hơn Euclidean trên không gian nhiều chiều này.

4.4.3. Kết quả thực nghiệm (Bước nhảy vọt)

Thời gian huấn luyện: 47.58 giây (Chậm hơn Linear gấp 15 lần do tính toán khoảng cách).

Tiêu chí	**Linear (Baseline)**	**KNN (M4)**	**Đánh giá**
MAE	2,983	**1,087**	**Giảm 64% sai số**
RMSE	3,865	**1,571**	Giảm cực mạnh
MAPE	22.59%	**7.81%**	**Đạt mức xuất sắc (<10%)**
R^2 **Score**	0.7114	**0.9523**	**Cải thiện vượt bậc (+0.24)**

Nhận định quan trọng

Sự chuyển đổi sang mô hình Phi tuyến (KNN) đã giải quyết triệt để vấn đề Underfitting của nhóm Tuyến tính. Mô hình hiện tại đã đạt độ chính xác đủ để ứng dụng thực tế.

a) Scatter Plot (KNN):

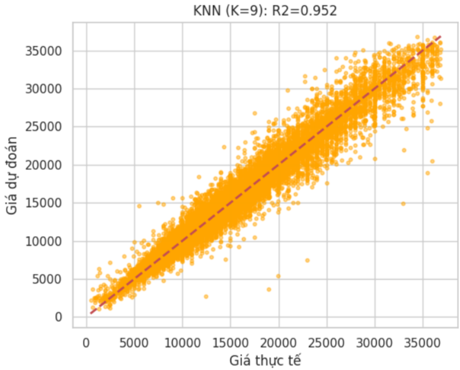


Figure: Độ khớp dữ liệu ($R^2 = 0.95$)

b) Đánh giá Mô hình 4:

+ **Ưu điểm:** Độ chính xác rất cao ($R^2 > 0.95$). Nắm bắt tốt các mẫu giá cục bộ.
- **Nhược điểm:**
  - Thời gian huấn luyện lâu (47s).
  - Không suy ra được công thức toán học tường minh.
  - Nhạy cảm nếu dữ liệu mới nằm ngoài vùng dữ liệu huấn luyện.

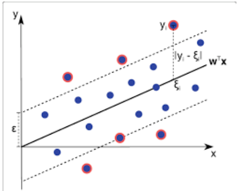
$\Rightarrow$ **Định hướng:** Tiếp tục thử nghiệm **Cây quyết định (Tree-based)** để xem có thể đạt kết quả tương tự mà tốc độ tốt hơn hoặc dễ giải thích hơn không.

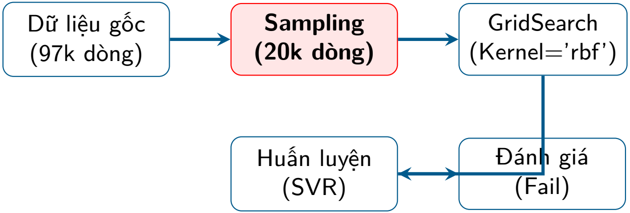
4.5.1. Mô hình 5: Support Vector Regression (SVR)

1. **Giới thiệu:**
   - Áp dụng nguyên lý của SVM vào bài toán hồi quy.
   - Mục tiêu: Tìm một hàm $f(x)$ phẳng nhất có thể, sao cho sai số của các điểm dữ liệu nằm trong giới hạn ϵ (Epsilon-tube).
2. **Kỹ thuật áp dụng:**
   - **Kernel Trick:** Sử dụng nhân `rbf` (Radial Basis Function) để xử lý dữ liệu phi tuyến.
   - **Sampling (Bắt buộc):** Do độ phức tạp tính toán của SVR là $O(N^3)$, lấy mẫu ngẫu nhiên **20,000 bản ghi** để thực nghiệm (thay vì 97k).

Cấu hình (GridSearch)

- **Kernel:** RBF.
- **C:** 100 (Hệ số phạt).
- **Epsilon:** 0.2.





Thách thức kỹ thuật:

- Thời gian huấn luyện cực lâu: **91.6 giây** (chỉ với 20k mẫu).
- Chậm gấp ∼2000 lần so với Linear Regression.

4.5.3. Kết quả thực nghiệm (Negative Result)

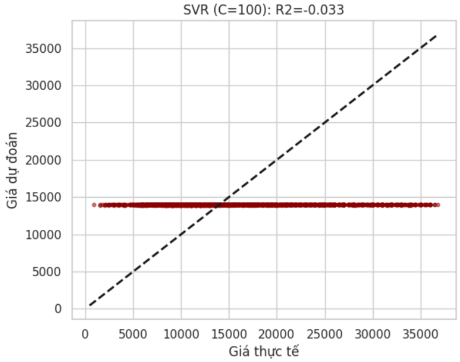
So sánh với mô hình tốt nhất trước đó (KNN):

Tiêu chí	**KNN (Model 4)**	**SVR (Model 5)**	**Đánh giá**
MAE	1,087	**5,634**	Sai số tăng gấp 5 lần
RMSE	1,571	**7,240**	Rất tệ
MAPE	7.81%	**44.46%**	Không chấp nhận được
R^2 **Score**	**0.9523**	-0.0332	**Kém hơn trung bình**

Phân tích nguyên nhân thất bại

1. **Độ nhạy cảm:** SVR quá nhạy cảm với nhiễu (outliers) và sự phân tán giá xe.
2. **Sparsity:** Dữ liệu sau One-Hot Encoding thưa thớt, Kernel RBF hoạt động kém.
3. **Sampling bias:** Việc giảm dữ liệu xuống 20k làm mất thông tin.

Biểu đồ Scatter (SVR):



Tổng kết Mô hình 5:

- **Hiệu năng:** Rất thấp ($R^2 < 0$), dự đoán tệ hơn cả giá trị trung bình.
- **Chi phí:** Tốn quá nhiều tài nguyên tính toán ($O(N^3)$).
- **Kết luận:** Mô hình Vector không phù hợp với dữ liệu bảng hỗn hợp kích thước lớn.

Quyết định chiến lược

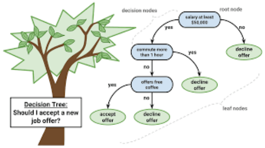
Dừng hướng tiếp cận Vector. Chuyển sang nhóm **Cây quyết định (Decision Tree, Random Forest)** để xử lý dữ liệu tốt hơn.

4.6.1. Mô hình 6: Decision Tree Regressor (Cây quyết định)

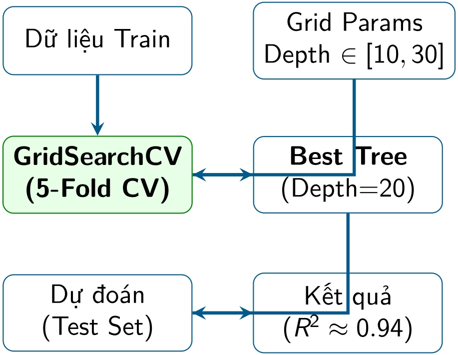
1. **Giới thiệu:**
   - Là mô hình thuộc nhóm Cây (Tree-based).
   - Học các quy tắc ra quyết định đơn giản (If-Then-Else) được suy ra từ các đặc trưng dữ liệu.
   - **Ưu điểm:** Dễ giải thích (White-box model), không yêu cầu giả định về phân phối dữ liệu, tốc độ dự đoán cực nhanh.
2. **Kỹ thuật áp dụng:**
   - **Thư viện:** `sklearn.tree.DecisionTreeRegressor`.
   - **Tối ưu hóa:** Kiểm soát độ sâu cây (`max_depth`) và số mẫu tối thiểu tại nút (`min_samples_leaf`) để tránh Overfitting.

Cơ chế hoạt động

Chia không gian dữ liệu thành các hình chữ nhật con dựa trên việc giảm thiểu sai số (MSE) tại mỗi lần cắt (split).



4.6.2. Quy trình thực hiện (Grid Search & Pruning)



Kết quả Tuning:

- Tham số tối ưu: `max_depth=20`, `min_samples_split=10`.
- *Ý nghĩa:* Cây cần độ sâu đủ lớn (20) để bắt được các mẫu phức tạp, nhưng cần giới hạn chia nhỏ (split=10) để tránh học nhiễu.

Thời gian huấn luyện: 75.2 giây (Khá lâu do GridSearch duyệt cây sâu).

Tiêu chí	KNN (M4)	Decision Tree (M6)	So sánh
MAE	1,087	**1,187**	Tăng nhẹ (+100)
RMSE	1,571	**1,757**	Tăng nhẹ
MAPE	7.81%	**8.37%**	Vẫn rất tốt (<10%)
R^2 **Score**	0.9523	**0.9404**	Kém hơn một chút
Inference	0.04 ms	$\approx$ **0.00 ms**	**Nhanh gấp 400 lần**

Nhận định quan trọng

Mặc dù độ chính xác thấp hơn KNN khoảng 1%, nhưng Decision Tree có ưu thế tuyệt đối về **tốc độ dự đoán (Real-time)**. Đây là sự đánh đổi (Trade-off) hợp lý cho ứng dụng Web App.

a) Feature Importance:

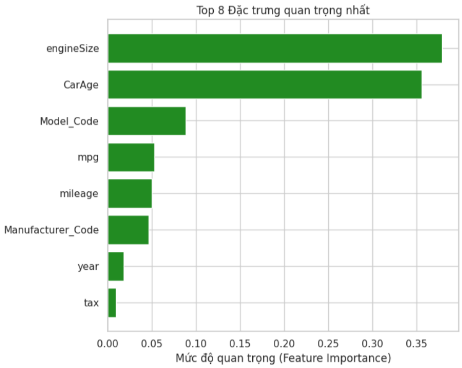


Figure: Các yếu tố quyết định giá

b) Đánh giá Mô hình 6:

+ **Ưu điểm:**
  - Tốc độ suy diễn cực nhanh.
  - Cho biết mức độ quan trọng của đặc trưng (Feature Importance).
  - Không cần Scaling dữ liệu quá khắt khe.
- **Nhược điểm:**
  - Độ chính xác thấp hơn KNN một chút.
  - Vẫn có nguy cơ Overfitting nếu không cắt tỉa.

⇒ **Định hướng:** Sử dụng kỹ thuật **Ensemble (Random Forest)** để kết hợp nhiều cây lại.

4.7.1. Mô hình 7: Random Forest (Rừng ngẫu nhiên)

1. Giới thiệu:

- Là mô hình học máy tổ hợp (**Ensemble Learning**) thuộc dòng Bagging (Bootstrap Aggregating).
- Kết hợp kết quả của nhiều cây quyết định độc lập để giảm phương sai (Variance) và chống Overfitting.

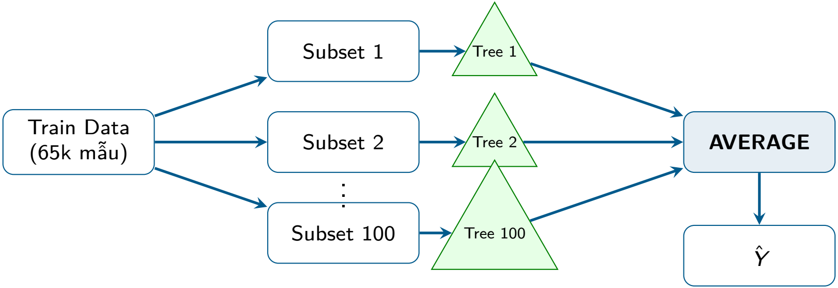
2. Kỹ thuật áp dụng:

- **Thư viện:** `sklearn.ensemble.RandomForestRegressor`.
- **Tối ưu hóa:** Sử dụng **RandomizedSearchCV** (hiệu quả hơn GridSearch cho không gian tham số lớn).
- **Tham số chính:** Số lượng cây (`n_estimators`), Độ sâu tối đa (`max_depth`).

Cơ chế hoạt động

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

(Dự đoán là trung bình cộng của N cây con được huấn luyện trên các tập mẫu con khác nhau)



Kết quả Tuning:

- Best Params: `n_estimators=100`, `max_depth=30`, `bootstrap=True`.

4.7.3. Kết quả thực nghiệm (So sánh với Model 6)

Thời gian huấn luyện: 466.2 giây (∼7.8 phút). Tăng đáng kể do phải xây dựng 100 cây.

Tiêu chí	Decision Tree (M6)	Random Forest (M7)	Đánh giá
MAE	1,187.93	**1,060.63**	Giảm sai số
RMSE	1,757.15	**1,543.43**	Giảm mạnh
MAPE	8.37%	**7.48%**	**Rất xuất sắc**
R^2 **Score**	0.9404	**0.9540**	**Cải thiện +1.36%**
Inference Time	1 μs	19 μs	Vẫn rất nhanh

Nhận định

Random Forest đã đẩy giới hạn chính xác lên mức **95.4%**. Việc kết hợp nhiều cây giúp mô hình "mượt" hơn và ít bị ảnh hưởng bởi nhiễu cục bộ so với Decision Tree đơn lẻ.

4.7.4. Trực quan hóa và Đánh giá tổng quát

a) Feature Importance:

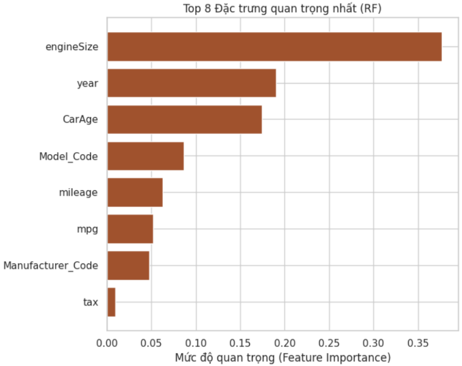


Figure: Các yếu tố ảnh hưởng giá (RF)

b) Đánh giá Mô hình 7:

+ **Ưu điểm:**
  - Độ chính xác rất cao ($R^2 > 0.95$).
  - Chống Overfitting tốt hơn Decision Tree.
  - Không cần Scaling dữ liệu.

- **Nhược điểm:**
  - Thời gian huấn luyện lâu (gấp 6 lần DT).
  - Kích thước mô hình lớn (nặng bộ nhớ).

$\Rightarrow$ **Định hướng:** Thử nghiệm nhóm **Boosting (CatBoost/XGBoost)** để xem liệu có thể đạt độ chính xác và tốc độ huấn luyện tốt hơn không.

4.8.1. Mô hình 8: Gradient Boosting Regressor (GBM)

1. Giới thiệu:

- Là thuật toán thuộc nhóm **Boosting** (Tăng cường).
- Khác với Bagging (xây cây song song), Boosting xây dựng các cây **tuần tự**. Cây sau tập trung sửa lỗi (Residuals) của cây trước đó.

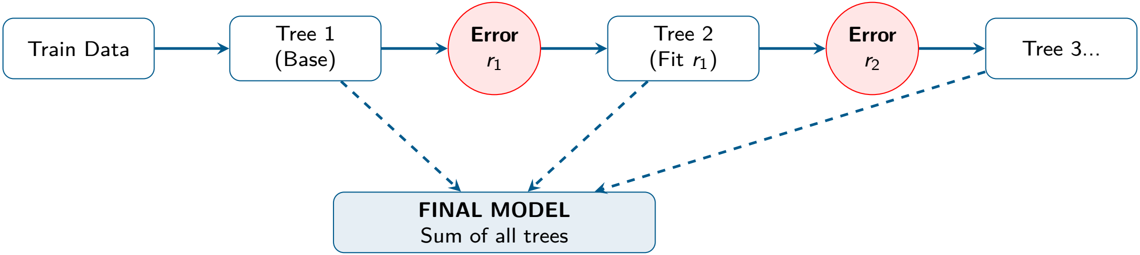
2. Kỹ thuật áp dụng:

- **Thư viện:** `sklearn.ensemble.GradientBoostingRegressor`.
- **Tối ưu hóa:** Sử dụng **RandomizedSearchCV** để tìm tổ hợp tham số tối ưu (Learning Rate, Số cây, Độ sâu).

Cơ chế Boosting

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

- F_m: Mô hình tổng thể bước m.
- h_m: Cây con (Weak learner) khớp vào phần dư.
- η: Tốc độ học (Learning Rate).



Kết quả Tuning:

- Best Params: `n_estimators=300`, `learning_rate=0.2`, `max_depth=7`.
- *Thách thức:* Thời gian huấn luyện rất lâu (**606 giây** $\approx$ 10 phút) do tính chất tuần tự.

4.8.3. Kết quả thực nghiệm (So sánh với Random Forest)

Bước tiến mới về độ chính xác:

Tiêu chí	**Random Forest (M7)**	**Gradient Boosting (M8)**	**Đánh giá**
MAE	1,060.63	**1,009.41**	Giảm thêm $50
RMSE	1,543.43	**1,426.80**	Giảm đáng kể
MAPE	7.48%	**7.09%**	Chính xác cao
R^2 **Score**	0.9540	**0.9607**	**New Best!**
Train Time	466s	**606s**	Chậm hơn 30%

Nhận định

GBM đã chứng minh hiệu quả của việc "học từ sai lầm", đẩy độ chính xác lên mức **96%**. Tuy nhiên, tốc độ huấn luyện là rào cản lớn nếu dữ liệu tiếp tục tăng.

a) Feature Importance:

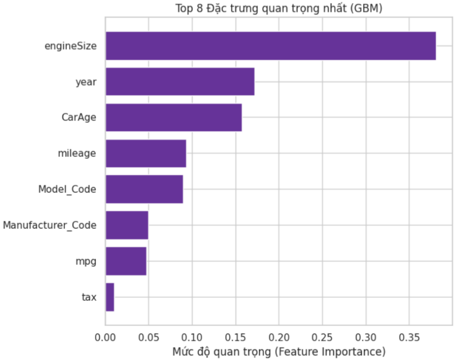


Figure: Các yếu tố trọng yếu (GBM)

b) Đánh giá Mô hình 8:

+ **Ưu điểm:**
  - Độ chính xác vượt trội hơn Random Forest.
  - Giảm thiểu Bias tốt hơn.
- **Nhược điểm:**
  - **Tốc độ chậm:** Do không thể song song hóa quá trình xây cây.
  - Dễ bị Overfitting nếu không tinh chỉnh kỹ (Learning rate, Subsample).

⇒ **Giải pháp:** Sử dụng **XGBoost** và **CatBoost** để vừa giữ được độ chính xác của Boosting, vừa cải thiện tốc độ huấn luyện.

4.9.1. Mô hình 9: XGBoost (eXtreme Gradient Boosting)

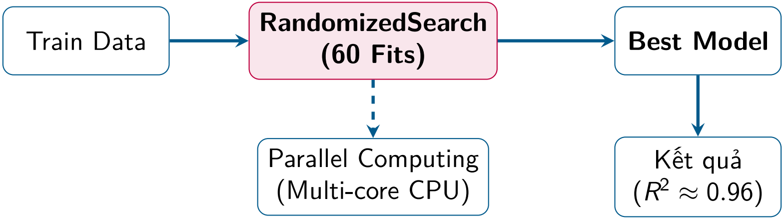
1. **Giới thiệu:**
   - Là phiên bản cải tiến vượt bậc của Gradient Boosting, được thiết kế để tối ưu hóa tốc độ và hiệu suất.
   - "Ông vua" của các giải thi đấu Kaggle trước khi CatBoost ra đời.
2. **Cải tiến kỹ thuật so với GBM:**
   - **Parallel Processing:** Tính toán song song tại các node giúp tốc độ huấn luyện nhanh gấp 6-10 lần.
   - **Regularization (L1/L2):** Tích hợp sẵn 'reg_alpha' và 'reg_lambda' vào hàm mục tiêu để chống Overfitting (GBM không có).
   - **Tree Pruning:** Cắt tỉa cây thông minh hơn.

Hàm mục tiêu (Objective)

$$Obj = \sum L(y_i, \hat{y}_i) + \sum \Omega(f_k)$$

(Ω: Thành phần phạt độ phức tạp của cây)

4.9.2. Quy trình thực hiện (Parallel Tuning)



Kết quả Tuning (Best Params):

- `learning_rate=0.2`, `n_estimators=700`, `max_depth=5`.
- **Regularization:** `reg_lambda=2`, `reg_alpha=1` (Giúp mô hình tổng quát hóa tốt hơn, tránh học vẹt).

So sánh với GBM (Mô hình 8):

Tiêu chí	Gradient Boosting (M8)	XGBoost (M9)	Đánh giá
Training Time	606.1s (∼10p)	**96.6s (∼1.5p)**	**Nhanh gấp 6 lần**
MAE	1,009.41	1,039.94	Tăng nhẹ
MAPE	7.09%	**7.31%**	Tương đương
R^2 **Score**	**0.9607**	**0.9589**	**Thấp hơn 0.0018**

Phân tích Trade-off

Mặc dù R^2 thấp hơn GBM một lượng không đáng kể (0.18%), nhưng XGBoost mang lại lợi thế khổng lồ về **thời gian huấn luyện**. Đây là yếu tố then chốt khi làm việc với Big Data hoặc cần Retrain mô hình liên tục.

a) Feature Importance (Gain):

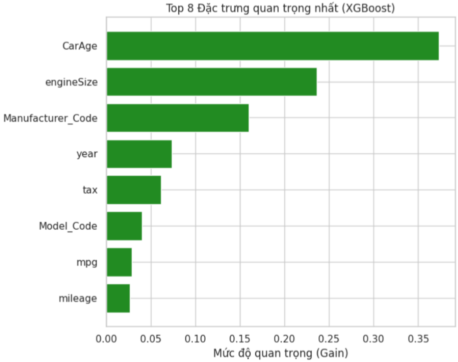


Figure: Mức độ đóng góp thông tin (Gain)

b) Đánh giá Mô hình 9:

+ **Ưu điểm:**
  - Tốc độ huấn luyện vượt trội.
  - Tích hợp Regularization giúp kiểm soát Overfitting tốt hơn GBM thuần.
  - Kết quả rất ổn định ($R^2 \approx 0.96$).

- **Hạn chế:**
  - Vẫn cần One-Hot Encoding cho biến phân loại (tốn bộ nhớ nếu số lượng Hãng/Dòng xe quá lớn).

$\Rightarrow$ **Bước cuối cùng:** Thử nghiệm **CatBoost** để giải quyết triệt để vấn đề biến phân loại mà không cần One-Hot Encoding.

1. **Tổng quan:**
   - Là thuật toán Gradient Boosting trên cây quyết định, phát triển bởi **Yandex** (2017).
   - Được cộng đồng Data Science đánh giá là SOTA (State-of-the-Art) cho dữ liệu dạng bảng.

CatBoost

2. **Cơ sở học thuật (Paper):**

NeurIPS 2018

Prokhorenkova, L., Gusev, G., Vorobev, A., Dorogush, A. V., & Gulin, A. (2018). **"CatBoost: unbiased boosting with categorical features."** *Advances in neural information processing systems, 31.*

$\rightarrow$ *Bài báo đề xuất giải pháp xử lý biến phân loại và "Ordered Boosting" để giải quyết vấn đề Target Leakage.*

4.10.2. Tại sao CatBoost phù hợp nhất?

Thách thức của dữ liệu xe hơi:

- Chứa nhiều biến phân loại có lực lượng lớn (High Cardinality): `Model` (hàng trăm dòng xe), `Manufacturer`, `Region`.
- One-Hot Encoding thông thường sẽ tạo ra ma trận rất thưa (Sparse matrix) và tốn bộ nhớ.

Giải pháp của CatBoost

1. **Ordered Target Statistics:** Mã hóa biến phân loại bằng thống kê của biến mục tiêu (Price) dựa trên hoán vị ngẫu nhiên, tránh rò rỉ dữ liệu.
2. **Symmetric Trees:** Cây đối xứng giúp tốc độ suy diễn (Prediction) nhanh gấp 8-10 lần XGBoost.

4.10.3. Kết quả thực nghiệm (Đỉnh cao)

Cấu hình tối ưu (Optuna Trial 7):

- `iterations=1709`, `depth=9`, `learning_rate=0.14`.

Hiệu năng trên tập Test:

Tiêu chí	XGBoost (M9)	CatBoost (M10)	Đánh giá
MAE	1,039.94	**979.60**	Sai số tuyệt đối thấp nhất
RMSE	1,458.92	**1,383.33**	Ổn định nhất với nhiễu
MAPE	7.31%	**6.9%**	Dự đoán cực sát giá thực
R^2 **Score**	0.9589	**0.9631**	**Best Performance**

CatBoost không chỉ chính xác hơn mà còn không cần bước tiền xử lý One-Hot Encoding phức tạp.

4.10.4. Giải thích mô hình (SHAP Feature Importance)

Phân tích các yếu tố tác động giá:

1. **Year:** Xe càng mới → Giá càng cao (SHAP dương).
2. **EngineSize:** Động cơ lớn là đặc trưng của xe sang/thể thao → Giá cao.
3. **Mileage:** Xe đi nhiều → Giá giảm mạnh (SHAP âm).
4. **Manufacturer:** Các hãng xe sang (Audi, BMW) đẩy giá trị dự đoán lên cao hơn so với xe phổ thông.

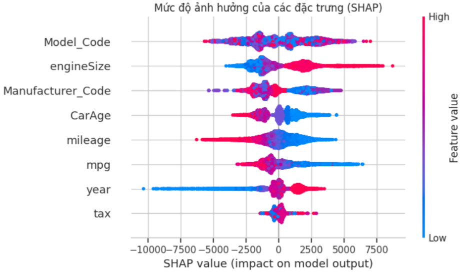


Figure: Mức độ ảnh hưởng của đặc trưng

4.11. Tổng kết và Đánh giá 10 mô hình

ID	Mô hình	Cấu hình chính	RMSE	R^2	Đánh giá
5	SVR (Support Vector)	Kernel=RBF (Sampled)	7,240	-0.03	**Kém nhất**. Không phù hợp.
1	Linear Regression	OLS (Default)	3,865	0.71	Underfitting nặng.
2	Ridge Regression	Alpha=0.1	3,865	0.71	Không cải thiện.
3	Lasso Regression	Alpha=0.1	3,865	0.71	Chọn lọc đặc trưng tốt.
4	KNN Regressor	K=9, Distance	1,571	0.95	Tốt nhưng rất chậm.
6	Decision Tree	Depth=20	1,757	0.94	Nhanh, dễ Overfitting.
7	Random Forest	Est=100	1,543	0.95	Ổn định, train lâu.
8	Gradient Boosting	Est=300	1,426	0.960	Chính xác cao.
9	XGBoost	Est=700, LR=0.2	1,458	0.959	Tốc độ train nhanh.
10	**CatBoost**	**Iter=1709, Depth=9**	**1,383**	**0.963**	**Lựa chọn (Best Fit)**

Dựa trên kết quả thực nghiệm, nhóm quyết định chọn **CatBoost** làm mô hình lõi cho ứng dụng, với các lý do:

1. **Độ chính xác cao nhất:** $R^2 \approx 0.963$, RMSE thấp nhất.
2. **Xử lý dữ liệu tối ưu:** Tự động xử lý biến phân loại (Model, Make) tốt hơn One-Hot Encoding truyền thống (giúp giảm chiều dữ liệu).
3. **Tốc độ suy diễn (Inference):** Cấu trúc cây đối xứng giúp dự đoán cực nhanh ($< 50ms$), phù hợp cho Web App thời gian thực.
4. **Tính ổn định:** Ít bị Overfitting nhờ cơ chế Ordered Boosting.

CATBOOST

Selected Model

Chương 5: Đóng gói và Ứng dụng

5.1. Khả năng ứng dụng vào ngữ cảnh thực tế

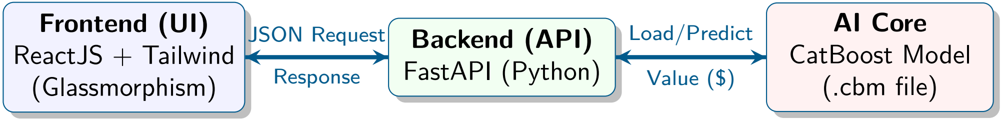
1. **Ngữ cảnh triển khai (Business Context):**
   - **Tại Showroom:** Nhân viên Sales sử dụng tablet để định giá nhanh cho khách muốn đổi xe cũ lấy xe mới (Trade-in).
   - **Sàn TMĐT:** Tích hợp API để gợi ý giá bán hợp lý cho người đăng tin.
2. **Đánh giá tính khả thi (Feasibility):**

Điều kiện đủ để ứng dụng

- **Độ chính xác:** $R^2 \approx 96.3\%$, sai số trung bình (MAE) $< \$1000 \rightarrow$ Thấp hơn biên độ thương lượng (thường là 5-10%).
- **Tốc độ:** Thời gian suy diễn (Inference) $< 50ms \rightarrow$ Đảm bảo trải nghiệm Real-time.
- **Tính ổn định:** CatBoost xử lý tốt các giá trị lạ (Unseen labels) nhờ cơ chế xử lý category thông minh.

5.2. Thiết kế và Cài đặt Hệ thống

Kiến trúc 3 tầng (3-Tier Architecture):



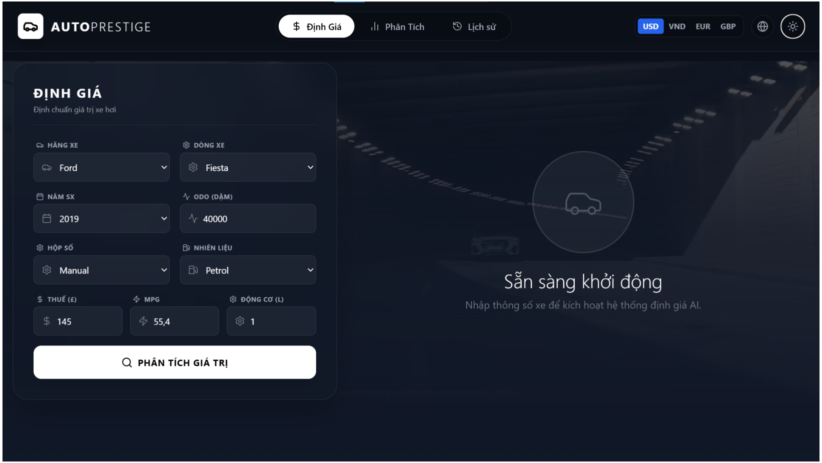
Công nghệ Frontend:

- **Vite + React:** Tối ưu hiệu năng.
- **Framer Motion:** Hiệu ứng chuyển động mượt mà.
- **Tính năng:** Đa ngôn ngữ, Chuyển đổi tiền tệ, Khóa kết quả.

Công nghệ Backend:

- **FastAPI:** API hiệu năng cao.
- **Joblib:** Quản lý Encoders.
- **Xử lý lỗi:** Tự động map giá trị lạ về mặc định.

5.3. Giao diện người dùng (Web App Demo)



Tình huống giả định

Khách hàng muốn bán xe **Ford Fiesta 2019**, bản số sàn, đã đi 40,000 dặm.

Bước 1: Nhập liệu

- Chọn Hãng: `Ford`, Dòng: `Fiesta`.
- Năm: `2019`, Odo: `40,000`.
- Kỹ thuật: `Manual`, `Petrol`, 1.0L.

Bước 2: Xử lý

- Hệ thống gửi API → Encode → CatBoost Predict.
- Hiệu ứng Loading "Phân tích thị trường".

Bước 3: Kết quả & Tương tác

- **Hiển thị giá:** Ví dụ **$10,500**.
- **Tính năng an toàn:** Màn hình tự động **KHÓA** kết quả để tránh chỉnh sửa gian lận.
- **Tiện ích:** Chuyển đổi sang VND/EUR để khách hàng dễ hình dung.

Tổng kết và Đánh giá

6.1. Tổng hợp kết quả đạt được

Bài tập hoàn thành mục tiêu xây dựng Hệ hỗ trợ quyết định định giá xe với các kết quả cụ thể:

1. Về mặt Kỹ thuật

- ✓ Xây dựng thành công Pipeline xử lý dữ liệu chuẩn (97k bản ghi).
- ✓ Thực nghiệm và đánh giá chi tiết **10 mô hình hồi quy**.
- ✓ Xác định **CatBoost** là mô hình tối ưu nhất ($R^2 \approx 96.3\%$, MAE $<$ \$1000).

2. Về mặt Ứng dụng

- ✓ Xây dựng hệ thống **Frontend (ReactJS)** hiện đại, giao diện Glassmorphism thân thiện.
- ✓ Tích hợp **Backend (FastAPI)** xử lý dữ liệu và dự đoán thời gian thực.
- ✓ Cung cấp khả năng **giải thích giá (SHAP)** và chuyển đổi tiền tệ linh hoạt.

6.2. Hạn chế và Hướng phát triển

Hạn chế hiện tại:

- **Dữ liệu tĩnh:** Mô hình được huấn luyện trên dữ liệu lịch sử, chưa cập nhật realtime theo biến động thị trường hàng ngày.
- **Phạm vi:** Dữ liệu tập trung vào thị trường Anh/Mỹ, cần tinh chỉnh (Fine-tuning) nếu áp dụng cho thị trường Việt Nam (thuế, phí khác biệt).

Hướng phát triển tương lai:

1. **Tích hợp Computer Vision:** Định giá dựa trên hình ảnh ngoại thất xe (phát hiện trầy xước, đâm đụng).
2. **Crawl dữ liệu Real-time:** Tự động cập nhật giá từ các sàn giao dịch lớn để Retrain mô hình định kỳ.
3. **Mở rộng sang Xe điện (EV):** Bổ sung các đặc trưng về pin và sạc.

"Hệ thống không chỉ là một công cụ tính toán, mà là bước đệm để minh bạch hóa thị trường xe cũ."

Tài liệu tham khảo

Tài liệu tham khảo I

[1] 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.

[2] ThSKH. Nguyễn Danh Tú, "Bài giảng Kho dữ liệu và kinh doanh thông minh," *Khoa Toán-Tin - Đại học Bách Khoa Hà Nội*, 2025.

[3] TS. Trần Thị Thu Hà, *Giáo trình Hệ thống thông tin hỗ trợ ra quyết định*, NXB Đại học Kinh tế Quốc dân, 2020.

[4] 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 (NeurIPS)*, vol. 31, 2018.

[5] 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*, pp. 785–794, 2016.

Tài liệu tham khảo II

[6] Meruvu Likith, "90,000+ Cars Data From 1970 to 2024," *Kaggle Dataset*, 2024. [Online]. Available: `https://www.kaggle.com/datasets/meruvulikith/90000-cars-data-from-1970-to-2024`

[7] F. Pedregosa *et al.*, "Scikit-learn: Machine Learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[8] R. Wirth and J. Hipp, "CRISP-DM: Towards a standard process model for data mining," 2000.

HUST

hust.edu.vn fb.com/dhbkhn

THANK YOU !