TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN

KHOA KHOA HỌC MÁY TÍNH

Báo cáo kết quả
HUẤN LUYỆN, ĐÁNH GIÁ VÀ TINH CHỈNH MÔ HÌNH BÀI TOÁN DỰ ĐOÁN GIÁ NHÀ

Lớp: CS116.P21

Nhóm thực hiện:

Nguyễn Thị Lý - 22520837

Hà Ngũ Long Nguyên - 22520965

Nguyễn Thu Phương - 22521167

HỒ CHÍ MINH – 04/2025

1. Kết quả Tiền xử lý dữ liệu:

Các chiến lược tiền xử lý dữ liệu:

- **Xử lý dữ liệu bị thiếu:**
  + Loại bỏ 4 cột có tỉ lệ thiếu dữ liệu trên 80%: *PoolQC, MiscFeature, Alley, Fence*
  + Các cột còn lại nếu là kiểu số thì điền giá trị khuyết bằng **median**, các cột phân loại điền bằng giá trị **None**, cột *Electrical* chỉ có 1 giá trị thiếu nên điền bằng **mode.**
- **Xử lý outlier:**
  - Phân loại cột số dựa trên số lượng giá trị duy nhất: Chia các cột dữ liệu số thành hai nhóm:
    - **Nhóm 1:** Số lượng giá trị duy nhất nhỏ hơn hoặc bằng 12: **Loại bỏ cột lệch:** Dựa trên quan sát, loại bỏ 6 cột có độ lệch cao được xác định là: *'ScreenPorch', 'LowQualFinSF', '3SsnPorch', 'BsmtFinSF2', 'MiscVal', 'EnclosedPorch'.*
    - **Nhóm 2:** Số lượng giá trị duy nhất lớn hơn 12: Loại bỏ cột lệch: Loại bỏ cột '*PoolArea'* do nhận thấy có độ lệch cao.
  - **Xử lý đa cộng tuyến (Multicollinearity)** sau khi loại bỏ cột lệch:
    - Tính ma trận tương quan: Tính ma trận tương quan giữa các biến số còn lại.
    - Loại bỏ cột tương quan cao: Xác định và loại bỏ một cột từ mỗi cặp cột có mức độ tương quan lớn hơn 0.75 để giảm đa cộng tuyến.
  - **Xử lý outlier trực tiếp trên các cột còn lại (sau khi loại bỏ cột và giảm đa cộng tuyến):**
    - Xác định cột cần xử lý: Nhận diện 5 cột chứa outlier cần xử lý: '*LotFrontage', 'LotArea', 'BsmtFinSF1', 'TotalBsmtSF', 'TotRmsAbvGrd'*.
    - Loại bỏ dòng chứa outlier: Loại bỏ các dòng (quan sát) chứa outlier trong 5 cột này dựa trên các ngưỡng cụ thể:
      - *'LotFrontage'*: Ngưỡng > 200
      - '*LotArea'*: Ngưỡng > 100000
      - '*BsmtFinSF1*': Ngưỡng > 4000
      - '*TotalBsmtSF'*: Ngưỡng > 4000
      - '*TotRmsAbvGrd*': Ngưỡng > 4000

- Encode dữ liệu:

- Chia các cột dữ liệu có kiểu phân loại (object hoặc category) thành hai nhóm dựa trên tính chất của giá trị:
    - **Ordinal columns (biến thứ tự):** Các cột mà giá trị có thể được sắp xếp theo một thứ tự có ý nghĩa (ví dụ: chất lượng, đánh giá). Nhóm bạn đã xác định các cột sau là ordinal: *['ExterQual', 'ExterCond', 'BsmtQual', 'BsmtCond', 'HeatingQC', 'KitchenQual', 'FireplaceQu', 'GarageQual', 'GarageCond', 'PoolQC', 'BsmtExposure', 'Functional', 'LandSlope', 'PavedDrive', 'GarageFinish', 'BsmtFinType1', 'BsmtFinType2']:* **Encode cột Ordinal bằng Label Encoding:**
    - **Nominal columns (biến danh nghĩa)**: Các cột mà giá trị chỉ đại diện cho các danh mục khác nhau mà không có thứ tự cụ thể (ví dụ: tên loại, khu vực). Nhóm bạn đã xác định các cột sau là nominal: *['MSZoning', 'Street', 'LotShape', 'LandContour', 'Utilities', 'LotConfig', 'Neighborhood', 'Condition1', 'Condition2', 'BldgType', 'HouseStyle', 'RoofStyle', 'RoofMatl', 'Exterior1st', 'Exterior2nd', 'MasVnrType', 'Foundation', 'Heating', 'CentralAir', 'Electrical', 'GarageType', 'SaleType', 'SaleCondition']:* **Encode cột Nominal bằng One-Hot Encoding**

- Chuẩn hóa dữ liệu:

- Sử dụng chiến thuật **Standardization** thông qua **StandardScaler** từ thư viện *sklearn* đối với các cột dạng số trước khi thực hiện Encode để biến đổi dữ liệu sao cho nó có trung bình bằng 0 và độ lệch chuẩn bằng 1 với hy vọng các thuật toán học máy học hiệu quả hơn.

Kết quả Tiền xử lý:

- Tập *train.csv* sau khi tiền xử lý còn lại 1454 dòng và 255 cột. Trong đó có 40 cột kiểu số và 185 cột thuộc kiểu bool:

```
<class 'pandas.core.frame.DataFrame'>
Index: 1454 entries, 0 to 1453
Columns: 225 entries, Id to SalePrice
dtypes: bool(185), float64(40)
memory usage: 728.4 KB
```

-

	Id	MSSubClass	LotFrontage	LotArea	YearBuilt	YearRemodAdd	MasVnrArea	BsmtFinSF1	BsmtUnfSF	T
0	1.0	0.073836	-0.223516	-0.292989	1.049501	0.880128	0.520741	0.616203	-0.945995	
1	2.0	-0.874133	0.525544	-0.082066	0.156499	-0.428117	-0.570040	1.242232	-0.642892	
2	3.0	0.073836	-0.073704	0.220563	0.983353	0.831675	0.331524	0.109855	-0.303599	
3	4.0	0.310828	-0.473203	-0.091236	-1.861023	-0.718838	-0.570040	-0.511571	-0.063831	
4	5.0	0.073836	0.725293	0.772632	0.950278	0.734768	1.377783	0.498822	-0.176929	
...	...	...	...	...	...	...	...	...	...	
1449	1456.0	0.073836	-0.373328	-0.390747	0.917204	0.734768	-0.570040	-1.008712	0.870357	
1450	1457.0	-0.874133	0.775231	0.573630	0.222648	0.153325	0.092220	0.809535	0.047005	
1451	1458.0	0.310828	-0.173579	-0.184409	-1.001095	1.025489	-0.570040	-0.375778	0.698449	
1452	1459.0	-0.874133	-0.073704	-0.060607	-0.703428	0.540953	-0.570040	-0.895935	-1.285288	
1453	1460.0	-0.874133	0.275857	-0.020256	-0.207316	-0.961106	-0.570040	0.901598	-0.977662	

1454 rows × 225 columns

2. Các bước Feature Engineering:

2.1. Đánh giá tầm quan trọng của đặc trưng (Feature Importance) bằng mô hình Random Forest Regressor:

- Sử dụng mô hình học máy **Random Forest**, một thuật toán dựa trên cây quyết định mạnh mẽ và thường hiệu quả trong việc xác định các đặc trưng quan trọng để ước lượng mức độ đóng góp của từng đặc trưng trong việc dự đoán biến mục tiêu (*SalePrice*). Các đặc trưng được mô hình đánh giá là quan trọng hơn sẽ có giá trị độ quan trọng cao hơn.
- Các bước thực hiện:
    + Tách biến mục tiêu (*'SalePrice'*) khỏi tập dữ liệu các đặc trưng sau bước tiền xử lý.
    + Huấn luyện mô hình: Xây dựng và huấn luyện mô hình **Random Forest** trên dữ liệu.
    + Trích xuất tầm quan trọng: Lấy giá trị tầm quan trọng của từng đặc trưng từ mô hình **Random Forest** đã huấn luyện.

Feature	Importance
OverallQual	6.265994e-01
2ndFlrSF	6.293752e-02
TotalBsmtSF	5.499517e-02
BsmtFinSF1	2.894943e-02
GarageCars	2.601530e-02
LotArea	2.563818e-02
TotRmsAbvGrd	1.780144e-02
FullBath	1.496185e-02
Fireplaces	9.402185e-03
LotFrontage	7.946102e-03
YearBuilt	7.720132e-03
OpenPorchSF	7.593709e-03
MasVnrArea	7.099904e-03
YearRemodAdd	7.098768e-03
WoodDeckSF	6.651191e-03
MoSold	5.796410e-03
BsmtUnfSF	5.796030e-03
ExterQual	4.954604e-03

(18 cột có giá trị tầm quan trọng đặc trưng cao nhất)

- **Nhật xét:**
  + *'OverallQual'* là yếu tố quan trọng nhất: Với giá trị tầm quan trọng vượt trội (6.27e-01), chất lượng tổng thể của ngôi nhà là đặc quan trọng nhất trong việc dự đoán giá bán. Điều này hoàn toàn hợp lý vì chất lượng xây dựng, vật liệu và hoàn thiện có ảnh hưởng trực tiếp đến bất động sản có giá trị.
  + Diện tích là yếu tố quan trọng thứ hai: Các đặc trưng liên quan đến diện tích như *'2ndFlrSF'* (diện tích sàn tầng hai) và *'TotalBsmtSF'* (tích tích tầng hầm) có tầm quan trọng đáng kể, cho thấy kích thước và không gian sống là những yếu tố sau đó ảnh hưởng đến giá nhà.
  + Các cơ sở cơ bản của ngôi nhà có vai trò: Các đặc tính như *'BsmtFinSF1'* (diện tích tầng hầm hoàn thiện), *'LotArea'* (diện lô đất), *'TotRmsAbvGrd'* (tập số phòng mặt đất), và *'FullBath'* (số phòng tắm đầy đủ) cũng đóng góp vào dự đoán, mặc dù mức độ chất lượng và diện tích đặc biệt.
  + Các yếu tố quan trọng về nhà để xe và ngoại lệ có tác dụng ảnh hưởng: *'GarageCars'* (số lượng xe có thể chứa trong nhà để xe) và *'Heating'* (số lượng lò sưởi) cho thấy tác động của tiện ích và nội thất đến giá nhà.
  + Tuổi và thời gian cải tạo có tác động: *'YearBuilt'* (năm xây dựng) và *'YearRemodAdd'* (năm cải tạo) cũng có tầm quan trọng, cho tuổi đời và việc nâng cấp có ảnh hưởng đến giá trị ngôi nhà.

+ Diện tích mở cũng quan trọng: *'OpenPorchSF'* (diện tích hiên mở) và *'WoodDeckSF'* (diện tích sàn gỗ) cho thấy không gian sống bên ngoài cũng là yếu tố được cân nhắc.
+ Các cụ cụ thể về chất lượng và tình trạng có vai trò: Các cột đã được mã hóa như *'ExterQual'* (chất lượng ngoại thất), *'BsmtQual'* (chất lượng tầng hầm), *'KitchenQual'* (chất lượng bếp), *'FireplaceQu'* (chất lượng lò nung), và *'GarageQual'* (chất lượng nhà để xe) đều xuất hiện trong danh sách, cho thấy lượng chất lượng của chi tiết.
+ Các yếu tố vị trí và loại hình ảnh có ảnh hưởng nhưng ít hơn: Các cụ thể đã được One-Hot Encoding như *'GarageType_Attchd', 'GarageType_Detchd', 'CentralAir_Y', 'CentralAir_N', 'Neighborhood_Crawfor', 'MSZoning_RM', 'MSZoning_RL', 'Exterior1st_HdBoard', 'Exterior2nd_HdBoard'* cũng có mức độ quan trọng, chọn vị trí địa lý và loại nhà có ảnh hưởng, nhưng thường ít hơn so với các đặc tính về chất lượng và giao diện.
+ *'Id'* có mức độ quan trọng rất thấp: Việc 'Id' xuất hiện với mức độ quan trọng gần bằng 0 là điều hợp lý, vì đây chỉ là một định nghĩa duy nhất cho mỗi ngôi nhà và không mang thông tin mong đợi về giá.
+ Một số đặc điểm có ít ảnh hưởng: Nhiều đặc trưng khác trong danh sách có mức độ rất thấp, cho thấy chúng có thể không đóng góp nhiều vào công việc được mong đợi về giá dựa trên mô hình Random Forest này.

2.2 Chiến thuật thực hiện Feature Engineering:

- **Loại bỏ các đặc trưng không quan trọng:** Xác định các đặc trưng mà mô hình Random Forest đánh giá là hoàn toàn không có đóng góp vào việc dự đoán (Importance = 0), sau đó loại bỏ các đặc trưng đó, ở đây, loại được *'Id'*
- **Tạo các biến tổng hợp:** kết hợp các đặc trưng hiện có có liên quan về mặt ngữ nghĩa để tạo ra các đặc trưng mới, hy vọng sẽ nắm bắt được các mối quan hệ phức tạp hơn trong dữ liệu và cung cấp thông tin dự đoán mạnh mẽ hơn cho mô hình học máy. Cụ thể, nhóm đã tạo thêm các đặc trưng mới:
    + *TotalBath*: Tổng số phòng tắm (*FullBath* + *HalfBath*).
    + *TotalRooms*: Tổng số phòng (*TotRmsAbvGrd* + *BedroomAbvGr*).

+ *TotalArea*: Tổng diện tích (*TotalBsmtSF* + *1stFlrSF* + *2ndFlrSF* + *WoodDeckSF* + *OpenPorchSF*).
+ *LivingArea*: Tổng diện tích sinh hoạt *(GrLivArea* + *GarageArea*).
+ *Age*: Tuổi của ngôi nhà tại thời điểm giả định (2025 - *YearBuilt*).
+ *RemodAge:* Tuổi kể từ lần sửa chữa cuối cùng (2025 - *YearRemodAdd*).
+ *OverallCondition*: Đánh giá chung về chất lượng và tình trạng (trung bình của *OverallQual* và *OverallCond*).
+ *TotalBathrooms*: Tổng số phòng tắm (bao gồm cả tầng hầm).
+ *TotalFinishedArea:* Tổng diện tích hoàn thiện *(GrLivArea* + *TotalBsmtSF*).
+ *RoomsPerArea:* Tỷ lệ số phòng trên tổng diện tích.
+ *Remodeled:* Biến nhị phân cho biết nhà đã được sửa chữa sau khi xây dựng (1 nếu có, 0 nếu không).
+ *QualityCondition:* Đánh giá kết hợp chất lượng, tình trạng và số lượng phòng tắm.

Các đặc trưng sau khi thực hiện feature engineering:

	MSSubClass	LotFrontage	LotArea	YearBuilt	YearRemodAdd	MasVnrArea
0	0.073836	-0.223516	-0.292989	1.049501	0.880128	0.520741
1	-0.874133	0.525544	-0.082066	0.156499	-0.428117	-0.570040
2	0.073836	-0.073704	0.220563	0.983353	0.831675	0.331524
3	0.310828	-0.473203	-0.091236	-1.861023	-0.718838	-0.570040
4	0.073836	0.725293	0.772632	0.950278	0.734768	1.377783
...	...	...	...	...	...	...
1449	0.073836	-0.373328	-0.390747	0.917204	0.734768	-0.570040
1450	-0.874133	0.775231	0.573630	0.222648	0.153325	0.092220
1451	0.310828	-0.173579	-0.184409	-1.001095	1.025489	-0.570040
1452	-0.874133	-0.073704	-0.060607	-0.703428	0.540953	-0.570040
1453	-0.874133	0.275857	-0.020256	-0.207316	-0.961106	-0.570040

1454 rows × 230 columns

Partial	TotalBath	TotalRooms	Age	RemodAge	OverallCondition	TotalBathrooms	Remodeled	QualityCondition
False	2.017530	1.084265	2021.950499	2022.119872	0.069557	2.902223	0	0.718881
False	0.029410	-0.150720	2022.843501	2023.428117	1.055009	3.173834	0	0.713143
False	2.017530	-0.150720	2022.016647	2022.168325	0.069557	2.902223	0	0.718881
False	-1.784973	0.466772	2024.861023	2023.718838	0.069557	-0.900280	1	-0.548620
False	2.017530	2.927612	2022.049722	2022.265232	0.431698	2.902223	0	0.960309
...	...	...	...	...	...	...	...	...
False	2.017530	0.466772	2022.082796	2022.265232	-0.292585	0.957763	0	0.477454
False	0.029410	0.466772	2022.777352	2022.846675	0.156613	0.914102	0	0.114212
False	0.029410	2.927612	2024.001095	2021.974511	1.866349	-1.030358	1	1.254036
False	-1.784973	-1.994067	2023.703428	2022.459047	-0.205528	-0.900280	1	-0.732010
False	0.203148	-0.150720	2023.207316	2023.961106	-0.205528	1.087840	0	-0.069303

- Sau khi đã loại bỏ các đặc trưng không quan trọng và tạo thêm các đặc trưng mới, bạn huấn luyện lại mô hình Random Forest trên tập dữ liệu đã được cập nhật. Mục đích là để mô hình học được các mối quan hệ mới từ tập đặc trưng đã được tinh chỉnh.

2.3 Kết quả sau khi thực hiện các bước Feature Engineering:

- Sau khi thực hiện các bước trên, độ quan trọng của các đặc trưng đã thay đổi. Bảng dưới đây hiển thị một số đặc trưng quan trọng nhất sau quá trình feature engineering:

index	Feature	Importance
12	OverallQual	0.5292408348547021
229	QualityCondition	0.12919874698392045
8	TotalBsmtSF	0.07411936559122492
9	2ndFlrSF	0.055033613080759394
6	BsmtFinSF1	0.027987995005725998
2	LotArea	0.022126001590202622
22	GarageCars	0.012013513166305313
20	TotRmsAbvGrd	0.010692173119394664
227	TotalBathrooms	0.010613382244392936
223	TotalRooms	0.007756757501947662
1	LotFrontage	0.0075346016501380494
10	WoodDeckSF	0.006564255510992194
5	MasVnrArea	0.006561418164435491
11	OpenPorchSF	0.005998303840364051
23	MoSold	0.005134139237720094

- **Nhận xét:**
    + *'OverallQual'* vẫn là yếu tố dự đoán hàng đầu: Với giá trị độ quan trọng cao nhất **(0.529)**, chất lượng tổng thể của ngôi nhà tiếp tục là đặc trưng có sức ảnh hưởng lớn nhất đến giá bán. Điều này củng cố tầm quan trọng của yếu tố này trong việc định giá bất động sản.
    + Các biến tổng hợp mới thể hiện vai trò quan trọng:
        - *'QualityCondition'* **(0.129)**: Biến tổng hợp kết hợp chất lượng, tình trạng và số lượng phòng tắm cho thấy đây là một chỉ số mạnh mẽ, có tầm quan trọng thứ hai trong danh sách. Điều này cho thấy việc kết hợp các yếu tố liên quan đến chất lượng và tiện nghi có giá trị dự đoán cao.
        - *'TotalBathrooms'* **(0.011)** và *'TotalRooms'* (0.008): Các biến tổng hợp về số lượng phòng tắm và phòng cũng xuất hiện trong top 15, cho thấy quy mô và tiện nghi bên trong ngôi nhà vẫn là những yếu tố quan trọng.
    + Diện tích vẫn là yếu tố then chốt: Các đặc trưng liên quan đến diện tích như *'TotalBsmtSF'* (tổng diện tích tầng hầm) và *'2ndFlrSF'* (diện tích sàn tầng hai) tiếp tục nằm trong top đầu, khẳng định tầm quan trọng của không gian sống trong việc định giá. *'BsmtFinSF1'* (diện tích tầng hầm hoàn thiện) và *'LotArea'* (diện tích lô đất) cũng có đóng góp đáng kể.
    + Các đặc điểm về garage và số lượng phòng có ảnh hưởng: *'GarageCars'* (số lượng xe chứa trong garage) và *'TotRmsAbvGrd'* (tổng số phòng trên mặt đất) vẫn là những yếu tố được mô hình đánh giá cao.
    + Các đặc trưng về ngoại thất và kích thước lô đất đóng góp: *'WoodDeckSF'* (diện tích sàn gỗ) và *'LotFrontage'* (chiều dài mặt tiền) cũng có vai trò nhất định trong việc dự đoán giá.
    + *'MasVnrArea'* (diện tích lớp phủ tường) có ảnh hưởng: Đặc trưng này cho thấy vẻ ngoài và chất liệu hoàn thiện bên ngoài cũng đóng góp vào giá trị ngôi nhà.
- Sự xuất hiện của các biến tổng hợp cho thấy hiệu quả của **Feature Engineering:** Việc các biến tổng hợp như *'QualityCondition', 'TotalBathrooms',* và '*TotalRooms'* có độ quan trọng cao hơn một số đặc trưng gốc cho thấy chiến lược kết hợp thông tin đã thành công trong việc tạo ra các đặc trưng có khả năng dự đoán tốt hơn.

3. Huấn luyện, đánh giá và tinh chỉnh mô hình.

3.1. Giới thiệu.

Mục tiêu của việc Modeling:

- Xây dựng các model Machine Learning có khả năng dự đoán giá nhà một cách chính xác.
- So sánh hiệu suất của các model khác nhau để lựa chọn model tốt nhất.
- Kết hợp các model để cải thiện độ chính xác dự đoán thông qua kỹ thuật Ensemble Learning.

Các Model được sử dụng:

- Ridge Regression: Mô hình hồi quy tuyến tính với L2 regularization, giúp giảm overfitting.
- Lasso Regression: Mô hình hồi quy tuyến tính với L1 regularization, có khả năng lựa chọn feature.
- Elastic Net Regression: Kết hợp L1 và L2 regularization, tận dụng ưu điểm của cả hai.
- Support Vector Regression (SVR): Sử dụng hàm kernel để ánh xạ dữ liệu vào không gian chiều cao hơn, phù hợp với các mối quan hệ phi tuyến tính.
- Gradient Boosting Regression: Xây dựng model bằng cách kết hợp các cây quyết định yếu, giảm dần lỗi.
- LightGBM: Một biến thể của Gradient Boosting, sử dụng kỹ thuật leaf-wise để tăng tốc độ huấn luyện.
- XGBoost: Một thuật toán Gradient Boosting phổ biến, được biết đến với hiệu suất cao và khả năng xử lý dữ liệu lớn.

Kỹ thuật Ensemble Learning:

- Stacking: Kết hợp các model cơ sở bằng cách sử dụng một meta-model (XGBoost) để học cách kết hợp các dự đoán.
- Blending: Kết hợp các dự đoán từ các model đã huấn luyện bằng cách lấy trung bình có trọng số.

3.2. Lựa chọn và thiết lập tham số.

- Các model tuyến tính (Ridge, Lasso, Elastic Net) được sử dụng để nắm bắt các mối quan hệ tuyến tính tiềm năng giữa các features và giá nhà.
- SVR được sử dụng để khám phá các mối quan hệ phi tuyến tính phức tạp.
- Gradient Boosting, LightGBM và XGBoost được sử dụng vì chúng là các thuật toán mạnh mẽ và thường đạt được hiệu suất tốt trong các bài toán dự đoán.

Cách thiết lập tham số cho từng model:

Các Model Tuyến tính (Ridge, Lasso, Elastic Net):

- Sử dụng RobustScaler để xử lý outliers trước khi đưa vào model, giúp giảm ảnh hưởng của các giá trị ngoại lệ.
- Sử dụng RidgeCV, LassoCV, ElasticNetCV để tự động tìm các tham số regularization tốt nhất thông qua Cross-Validation.
- alphas_alt, alphas2, e_alphas, e_l1ratio: Các giá trị alpha và l1_ratio được thử nghiệm trong quá trình Cross-Validation để tìm ra các giá trị tối ưu.

SVR:

- Sử dụng RobustScaler để xử lý outliers.
- C=20, epsilon=0.008, gamma=0.0003: Các tham số này được thiết lập thủ công, có thể dựa trên kinh nghiệm hoặc thử nghiệm.

Gradient Boosting:

- n_estimators=3000: Số lượng cây quyết định trong model.
- learning_rate=0.05: Tốc độ học, kiểm soát mức độ đóng góp của mỗi cây vào model.
- max_depth=4: Độ sâu tối đa của mỗi cây, giới hạn độ phức tạp của model.
- max_features='sqrt': Số lượng features được xem xét khi chia một node, giúp giảm correlation giữa các cây.
- min_samples_leaf=15, min_samples_split=10: Các tham số này kiểm soát kích thước của các leaf node và số lượng samples tối thiểu để chia một node, giúp ngăn ngừa overfitting.

- loss='huber': Sử dụng hàm loss Huber, ít nhạy cảm với outliers hơn so với Mean Squared Error.

LightGBM:

- objective='regression': Mục tiêu là bài toán hồi quy.
- num_leaves=4: Số lượng leaf node trên mỗi cây, kiểm soát độ phức tạp của model.
- learning_rate=0.01: Tốc độ học.
- n_estimators=5000: Số lượng cây quyết định.
- max_bin=200: Số lượng bins để rời rạc hóa các giá trị features, ảnh hưởng đến tốc độ huấn luyện và hiệu suất.
- bagging_fraction=0.75, bagging_freq=5: Sử dụng bagging để giảm variance.
- feature_fraction=0.2: Sử dụng feature subsampling để giảm correlation giữa các cây.

XGBoost:

- learning_rate=0.01: Tốc độ học.
- n_estimators=3460: Số lượng cây quyết định.
- max_depth=3: Độ sâu tối đa của cây.
- min_child_weight=0: Tổng trọng lượng tối thiểu của các instance con cần thiết để chia một node.
- gamma=0: Tham số regularization.
- subsample=0.7, colsample_bytree=0.7: Sử dụng subsampling để giảm variance.
- objective='reg:linear': Mục tiêu là bài toán hồi quy tuyến tính.
- reg_alpha=0.00006: L1 regularization.

Sử dụng Cross-Validation để đánh giá và lựa chọn model:

- Sử dụng K-Fold Cross-Validation (K=10) để đánh giá hiệu suất của các model trên các phần dữ liệu khác nhau, giúp đánh giá khách quan và tránh overfitting.

- kfolds = KFold(n_splits=10, shuffle=True, random_state=42): Khởi tạo đối tượng KFold với 10 splits, shuffle dữ liệu và sử dụng random_state để đảm bảo tính nhất quán.
- cv_rmse(model): Hàm này tính RMSE trung bình trên các fold khác nhau, cung cấp một ước tính về hiệu suất của model trên dữ liệu unseen.

3.3. Kỹ thuật Ensemble Learning.

Stacking:

- Các model cơ sở (ridge, lasso, elasticnet, gbr, xgboost, lightgbm) được huấn luyện trên tập train.
- Các dự đoán từ các model cơ sở được sử dụng làm đầu vào cho model meta (XGBoost).
- Model meta học cách kết hợp các dự đoán từ các model cơ sở để tạo ra dự đoán cuối cùng.
- Sử dụng use_features_in_secondary=True để cung cấp thêm thông tin cho model meta, có thể cải thiện hiệu suất.

Blending:

- Các dự đoán từ các model đã huấn luyện được kết hợp bằng cách lấy trung bình có trọng số.
- blend_models_predict(X): Hàm này thực hiện blending bằng cách gán trọng số cho từng model.
- Các trọng số được lựa chọn có thể dựa trên hiệu suất Cross-Validation của từng model, hoặc dựa trên kinh nghiệm.

3.4. Đánh giá mô hình.

- Sử dụng cv_rmse(model) để tính RMSE cho từng model.
- In ra RMSE trung bình và độ lệch chuẩn cho từng model.

```
Kernel Ridge score: 0.1041 (0.0147)
 2025-05-05 10:21:20.205160
Lasso score: 0.1046 (0.0153)
 2025-05-05 10:21:57.187166
ElasticNet score: 0.1048 (0.0152)
 2025-05-05 10:24:31.552905
SVR score: 0.1046 (0.0132)
 2025-05-05 10:24:36.858136
GradientBoosting score: 0.1069 (0.0139)
 2025-05-05 10:26:52.486385
Lightgbm score: 0.1056 (0.0159)
 2025-05-05 10:27:21.427270
```

- **Phân tích kết quả:**
    - Dựa trên kết quả Cross-Validation, **Kernel Ridge** có hiệu suất tốt nhất với RMSE trung bình thấp nhất (0.1041) và độ lệch chuẩn tương đối thấp (0.0147). Điều này cho thấy Kernel Ridge có khả năng dự đoán chính xác và ổn định trên các phần dữ liệu khác nhau.
    - Các model **Lasso**, **ElasticNet** và **SVR** có hiệu suất tương đương với RMSE trung bình khoảng 0.1046 - 0.1048.
    - **GradientBoosting** có hiệu suất kém hơn một chút so với các model còn lại, với RMSE trung bình là 0.1069.
    - **LightGBM** có RMSE trung bình là 0.1056, nằm giữa Kernel Ridge và GradientBoosting.
    - Độ lệch chuẩn của RMSE cho các model dao động từ 0.0132 đến 0.0159, cho thấy mức độ biến động của hiệu suất giữa các fold. **SVR** có độ lệch chuẩn thấp nhất (0.0132), cho thấy tính ổn định cao.
- **Nhận xét:**
    - Các model tuyến tính (Kernel Ridge, Lasso, ElasticNet) và SVR cho thấy hiệu suất tốt trong bài toán này. Điều này có thể cho thấy rằng có một phần đáng kể của mối quan hệ giữa các features và giá nhà có thể được mô hình hóa bằng các phương pháp tuyến tính hoặc gần tuyến tính.
    - Các model boosting (GradientBoosting, LightGBM) có hiệu suất kém hơn một chút so với các model tuyến tính. Điều này có thể là do các tham số của các model boosting chưa được

tối ưu hóa đầy đủ, hoặc do các model này đang overfitting trên tập train.