ĐẠI HỌC KINH TẾ TP. HỒ CHÍ MINH

TRƯỜNG CÔNG NGHỆ VÀ THIẾT KẾ UEH

KHOA CÔNG NGHỆ THÔNG TIN KINH DOANH

ĐỒ ÁN MÔN HỌC

MÔN HỌC: MÁY HỌC

Đề tài nhóm 3:

Tìm hiểu thuật toán Random Forest và ứng dụng mô hình Random Forest vào dự đoán điều kiện bán nhà ở Thành phố Ames, Hoa Kỳ

STT	HỌ VÀ TÊN	MSSV
1	Đinh Trọng Hữu	31211027643
2	Nguyễn Nhật Huy	31211027641
3	Nguyễn Trọng Hùng	31211027579
4	Lê Xuân Hưởng	35221020914
5	Lê Ngọc Anh Hùng	31211025877

TP. HỒ CHÍ MINH, NĂM 2023

LỜI MỞ ĐẦU **4**
CHƯƠNG 1: TỔNG QUAN VỀ ĐỀ TÀI **5**
1. 1. Lý do lựa chọn đề tài 5
1.2. Tính cấp thiết của đề tài 5
1.3. Mục tiêu nghiên cứu 5
1.4. Đối tượng nghiên cứu 5
1.5. Phạm vi nghiên cứu 5
CHƯƠNG 2: CƠ SỞ LÝ THUYẾT **6**
2.1. Khái niệm chung: 6
2.1.1. Khái niệm Mô hình máy học phân lớp có giám sát 6
2.1.2. Khái niệm Thuật toán Random Forest 6
2.2. Các thang đo độ của mô hình học máy phân lớp 6
2.2.1. Thang độ đo chính xác 6
2.2.2. Thang độ do hiệu quả 7
2.2.3. Overfitting 7
2.3. Thuật toán Random Forest 8
2.3.1. Kỹ thuật Ensemble learning 8
2.3.2. Phương pháp Bagging 8
2.3.3. Phương pháp Bootstrap 8
2.3.4. Các bước thực thiện thuật toán Random Forest 9
2.3.2. Đặc điểm của thuật toán 10
2.3.3. Ưu điểm và nhược điểm của thuật toán 10
CHƯƠNG 3: THỰC NGHIỆM **12**
3.1. Bài toán thực tế 12
3.2. Khung nghiên cứu tổng quát 13
3.3. Dữ liệu sử dụng 13
3.3.2. Thông tin bộ dữ liệu 13
3.3.3. Thống kê mô tả 17
3.4. Tiền xử lý dữ liệu 18
3.4.1. Xử lý Missing Values 18
3.4.2. Mã hóa các thuộc tính phân loại (Encode) 20
3.4.3. Chuẩn hóa (Normalize) 20
3.4.4. Xử lý Outliers 20
3.4.5. Giảm chiều dữ liệu (PCA) 21
3.4.5. Validation dataset 22
3.5. Phân chia dữ liệu huấn liệu và dữ liệu kiểm tra 23
3.6. Tìm tham số tối ưu cho mô hình Random Forest 23
3.7. Xây dựng mô hình 24
CHƯƠNG 4: ĐÁNH GIÁ KẾT QUẢ THỰC NGHIỆM **26**
4.1. Đánh giá qua thang độ đo hiệu quả 26
4.2. Đánh giá qua thang độ đo chính xác 27
4.3 So sánh mô hình LinearRegression 29
CHƯƠNG 5: KẾT LUẬN VÀ KHUYẾN NGHỊ **31**
5.1. Kết luận 31
5.2. Khuyến nghị 31
TÀI LIỆU THAM KHẢO **32**
BẢNG PHÂN CÔNG CÔNG VIỆC **33**

LỜI MỞ ĐẦU

Chúng ta đang sống và làm việc ở thế kỷ 21, với sự phát triển rất nhanh về kinh tế, khoa học - kỹ thuật. Xu hướng hội nhập toàn cầu hóa trên phạm vi toàn cầu, xuất hiện nhiều hình thức kinh doanh với các hàng hóa đa dạng khác nhau. Sự phát triển của xã hội cũng đi kèm theo các vấn đề như gia tăng dân số, nhu cầu về chỗ ở sống và làm việc của con người. Vì vậy lĩnh vực kinh doanh bất động sản đã hình thành và phát triển rất mạnh mẽ và nhanh chóng. Với những đặc điểm có riêng của mình kinh doanh bất động sản đem lại những khoản lợi nhuận khổng lồ cho các nhà đầu tư, kinh doanh. Do đó, thị trường bất động sản đóng vai trò quan trọng trong nền kinh tế của nhiều quốc gia, trong đó Hoa Kỳ là một ví dụ điển hình. Thị trường nhà ở tại Hoa Kỳ luôn có những biến động phức tạp, khiến việc dự đoán giá trị và điều kiện bán nhà trở thành một vấn đề nan giải. Vậy làm thế nào để dự đoán chính xác điều kiện bán nhà là một vấn đề quan trọng đối với các nhà đầu tư, người mua và người bán nhà. Việc áp dụng các phương pháp học máy, đặc biệt là thuật toán Random Forest, có thể là một giải pháp hiệu quả để giải quyết vấn đề này.

Thuật toán Random Forest là một trong những thuật toán học máy mạnh mẽ được nhóm chọn sử dụng trong bài toán dự đoán điều kiện bán nhà. Thuật toán này có nhiều ưu điểm như độ chính xác cao, khả năng chống overfitting tốt và dễ dàng giải thích. Nhận thức được tầm quan trọng của thuật toán Random Forest trong bài toán, nhóm thực hiện nghiên cứu đề tài này với mục tiêu là *"Tìm hiểu về thuật toán Random Forest và ứng dụng mô hình Random Forest trong dự đoán điều kiện bán nhà ở thành phố Ames, Iowa - Hoa Kỳ"*.

Để đạt được mục tiêu này, nhóm sẽ thực hiện các nội dung về tìm hiểu lý thuyết về thuật toán Random Forest sau đó xây dựng mô hình Random Forest để dự đoán điều kiện bán nhà và đánh giá mức độ hiệu quả của mô hình Random Forest. Kết quả của đề tài này sẽ góp phần nâng cao hiệu quả hoạt động của các công ty bất động sản trong việc dự đoán điều kiện bán nhà, giúp đưa ra các quyết định hiệu quả. Do điều kiện về mặt thời gian hạn chế nên đồ án nhóm không khỏi có phần sai sót, kính mong giảng viên nhiệt tình góp ý để nhóm có thể rút được kinh nghiệm qua những lỗi sai để nhóm có thể cải được kiến thức của mình. Qua bài đồ án, nhóm chúng tôi xin cảm ơn thầy Nguyễn An Tế đã truyền đạt những kiến thức quý báu của mình trong học phần môn Máy học.

CHƯƠNG 1: TỔNG QUAN VỀ ĐỀ TÀI

1. 1. Lý do lựa chọn đề tài

Tình hình thị trường nhà ở Mỹ đang chứng kiến một giai đoạn đi xuống, tạo ra những thách thức đặc biệt đối với cả người bán và người mua. Sự không chắc chắn và biến động trong điều kiện kinh tế và xã hội làm cho việc dự đoán và đánh giá điều kiện bán của nhà trở nên ngày càng phức tạp. Để giải quyết tình hình này, cần có những giải pháp cụ thể và linh hoạt.

Trong bối cảnh trên, sự kết hợp giữa công nghệ và phương tiện dự đoán thị trường cùng với chiến lược tiếp thị sáng tạo là chìa khóa để doanh nghiệp bất động sản ở Mỹ có thể thích ứng và tìm kiếm cơ hội trong thị trường khó khăn. Một trong những cách tiếp cận này là sử dụng mô hình học máy, đặc biệt là thuật toán RandomForest, để dự báo điều kiện bán của những ngôi nhà tại thành phố Ames, Iowa.

1.2. Tính cấp thiết của đề tài

Thị trường bất động sản của Mỹ luôn biến động phức tạp, đặc biệt thị trường nhà ở gặp nhiều khó khăn .Doanh số bán hàng ở Mỹ trong tháng 8-2023 đang giảm 0,7% so với cùng kỳ năm ngoái khiến doanh thu của các công ty bất động sản tại Mỹ không tốt.Một trong những yếu tố ảnh hưởng đến việc mua bán nhà chính là điều kiện bán nhà, thế nhưng thị trường không ổn định khiến cho việc đánh giá và dự đoán các điều kiện bán trở nên khó khăn. Do đó, cần có giải pháp cụ thể giúp xác định điều kiện bán của mỗi ngôi nhà để đưa ra các cách tiếp cận thông minh trong bán hàng.

1.3. Mục tiêu nghiên cứ

Nhóm đặt ra mục tiêu chính là tìm hiểu về thuật toán Random Forest và áp dụng mô hình để dự đoán điều kiện bán nhà ở thành phố Ames, Iowa - Hoa Kỳ. Mục tiêu cụ thể bao gồm:

- Nắm vững lý thuyết về thuật toán Random Forest.
- Xây dựng mô hình Random Forest cho bài toán dự đoán điều kiện bán nhà.
- Đánh giá hiệu suất của mô hình.

1.4. Đối tượng nghiên cứu

Đối tượng nghiên cứu của đề tài là các bên liên quan trong thị trường bất động sản ở thành phố Ames, Iowa - Hoa Kỳ. Nghiên cứu này cũng hướng đến những người quan tâm đến ứng dụng của thuật toán Random Forest trong lĩnh vực bất động sản.

1.5. Phạm vi nghiên cứu

Phạm vi của nghiên cứu sẽ tập trung vào lý thuyết về thuật toán Random Forest, quá trình xây dựng mô hình, và ứng dụng trong việc dự đoán điều kiện bán nhà ở thành phố Ames, Iowa - Hoa Kỳ. Tuy nhiên, do hạn chế về thời gian lẫn kiến thức, nhóm sẽ không mở rộng quá mức vào các lĩnh vực khác ngoài phạm vi đã đề ra.

CHƯƠNG 2: CƠ SỞ LÝ THUYẾT

2.1. Khái niệm chung:

2.1.1. Khái niệm Mô hình máy học phân lớp có giám sát

Mô hình học máy phân lớp có giám sát là một mô hình được huấn luyện để dự đoán hoặc phân loại các điểm dữ liệu mới vào các lớp đã được xác định trước đó. Quá trình huấn luyện mô hình này đòi hỏi sự giám sát, nghĩa là cần một tập dữ liệu huấn luyện chứa các ví dụ với đầu vào đã biết và đầu ra mong muốn tương ứng.

Mô hình máy học phân lớp này học từ dữ liệu huấn luyện để tạo ra hàm ánh xạ từ các đặc trưng của đầu vào đến các lớp xác định. Mục tiêu chính là tối ưu hóa sao cho mô hình có khả năng dự đoán sát với đầu ra mong muốn trong tập dữ liệu huấn luyện.

2.1.2. Khái niệm Thuật toán Random Forest

Random Forest là một thuật toán học có giám sát hoạt động dựa trên khái niệm đóng bao. Thuật toán được xây dựng dựa trên ý tưởng kết hợp nhiều cây quyết định tạo thành "rừng ngẫu nhiên". Trong quá trình đóng bao, mỗi cây quyết định sẽ được huấn luyện độc lập trên một phần của tập dữ liệu và đầu ra cuối cùng của mô hình được tạo bằng cách tính toán dựa trên tất cả các cây quyết định đã xây dựng.

2.2. Các thang đo độ của mô hình học máy phân lớp

2.2.1. Thang độ đo chính xác

Độ đo Accuracy:

$$\text{Acc} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{FP} + \text{TN} + \text{FN}}$$

Độ đo Precision:

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}$$

Độ đo Recall:

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

Độ đo F1_score: là điều hoà giữa Precision và Recall. Chỉ số này giúp đánh giá hiệu quả của mô hình một cách toàn diện hơn

$$\text{F1\_score} = 2.\frac{\text{Pr e cision} \times \text{Recall}}{\text{Pr e cision} + \text{Recall}}$$

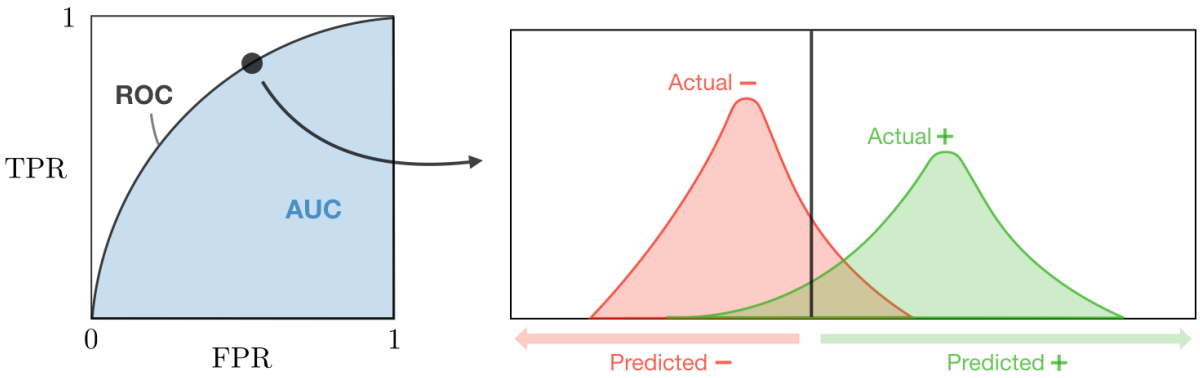
2.2.2. Thang độ do hiệu quả

ROC (Receving Operating Characteristic Curve) là đường cong lồi biểu diễn khả năng phân loại của mô hình phân loại nhị phân tại các ngưỡng threshold. Đường cong này dựa trên hai chỉ số: TPR còn gọi là Recall hay "độ nhạy" và FPR

$$\text{FPR} = \frac{\text{FP}}{\text{FP} + \text{TN}}$$

AUC (Aera under curve) là phần diện tích dưới đường ROC đánh hiệu quả của mô hình phân loại tốt như thế nào. AUC nhận giá trị thuộc đoạn, khi diện tích này càng lớn thì độ phân loại của mô hình càng tốt.

Hình 2.23. Đường cong ROC



Nguồn: https://stanford.edu/~shervine/teaching/cs-229/cheatsheet-machine-learning-tips-and-tricks

2.2.3. Overfitting

Bias là độ chệch trên tập huấn luyện. Variance là phương sai cho biết chất lượng trên tập kiểm tra kém hơn so với tập huấn luyện. Overfitting quá khớp là trường hợp mô hình có phương sai cao độ, chệch thấp hay là trường hợp mô hình chỉ ghi nhớ dữ liệu mà không học gì cả. Underfitting chưa khớp là trường hợp độ chệch cao, phương sai thấp hay là trường hợp mô hình chưa có độ chính xác cao trong tập dữ liệu huấn luyện cũng như không tổng quát hoá với tổng thể dữ liệu

2.3. Thuật toán Random Forest

2.3.1. Kỹ thuật Ensemble learning

Trước khi tìm hiểu cách thức hoạt động của thuật toán Random Forest trong phân loại dữ liệu, ta cần xem xét kỹ thuật toán học tập tổng hợp (Ensemble). Ensemble learning là một kỹ thuật trong học máy, ở đó nhiều mô hình riêng lẻ được kết hợp lại để tạo ra mô hình mạnh mẽ hơn.

Ensemble sử dụng hai phương pháp :

- Đóng bao (Bagging):Tạo ra một tập hợp con đào tạo khác với dữ liệu mẫu và có thể được thay thế, đầu ra cuối cùng được dựa trên biểu quyết đa số.
- Tăng cường(Boosting):Kết hợp những mô hình học yếu thành mô hình học mạnh bằng cách tạo ra các mô hình tuần tự sao cho mô hình đầu ra có độ chính xác cao nhất.

2.3.2. Phương pháp Bagging

Ba bước cơ bản của Bagging:

Bước 1: Bagging sử dụng kỹ thuật lấy mẫu bootstrapping để tạo ra các mẫu đa dạng, phương pháp lấy mẫu này tạo ra các tập con khác nhau của tập dữ liệu huấn luyện bằng cách chọn ngẫu nhiên các điểm dữ liệu và có lặp lại.

Bước 2: Các mẫu bootstrap sau khi tạo ra được đào tạo độc lập và song song với nhau.

Bước 3: Đối với bài toán phân loại, lớp có đa số phiếu cao nhất được chấp nhận hay biểu quyết theo đa số.

2.3.3. Phương pháp Bootstrap

Phương pháp Bootstrap dùng để ước lượng standard errors, độ lệch (bias) và tính toán khoảng tin cậy (confidence interval). Một mẫu bootstrap là một mẫu ngẫu nhiên được tạo ra với sự thay thế. N là số quan sát trong mẫu ban đầu. Các bước tạo mẫu bootstrap:

1. Chọn ngẫu nhiên một quan sát từ tập dữ liệu ban đầu.

2. Biểu diễn nó ra.

3. Chọn nó trở lại.

Lặp lại các bước trên N lần. Kết quả cho một mẫu bootstrap với N quan sát.

Random Forest là một thuật toán học có giám sát hoạt động dựa trên khái niệm bóng bao (Bagging hay Boostrap) đóng vai trò kỹ thuật chính trong thuật toán.

2.3.4. Các bước thực thiện thuật toán Random Forest

Các bước thực hiện thuật toán Random Forest cho phân loại dữ liệu như sau:

Bước 1: Đầu vào:

- Tập dữ liệu huấn luyện D.
- Số lượng cây quyết định cần xây dựng (k).
- Số lượng thuộc tính được chọn ngẫu nhiên để xây dựng mỗi cây (m).

Bước 2: Với mỗi cây quyết định từ 1 đến k:
Bước 2.1: Tạo dữ liệu ngẫu nhiên:

- Tạo một tập con D' từ D bằng cách lấy mẫu bootstrap (lấy mẫu ngẫu nhiên có trùng lặp với việc thay thế từ giá trị D).
- Lấy ngẫu nhiên m thuộc tính từ tổng số thuộc tính có sẵn.
- Sử dụng độ đo Entropy và chỉ số Gini để chọn lựa thuộc tính phân nhánh. Cụ thể ta sẽ chọn thuộc tính có Entropy và Gini thấp nhất

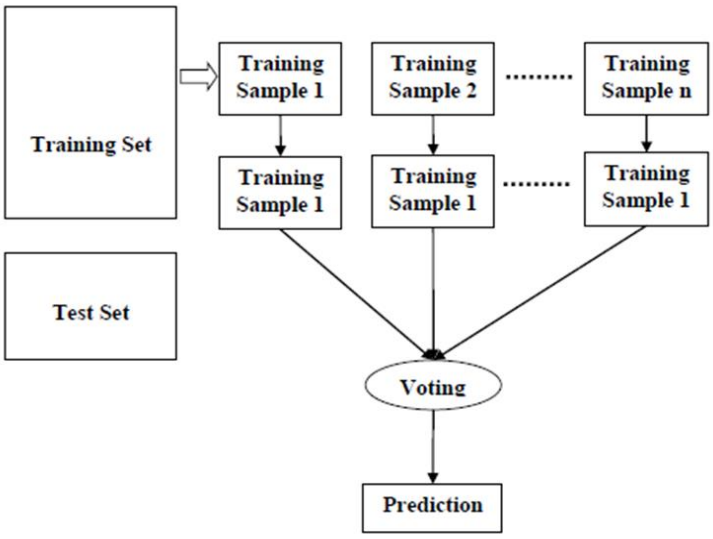
Bước 2.2: Xây dựng cây quyết định:

- Sử dụng thuật toán xây dựng cây quyết định (ví dụ: CART, ID3) trên tập con D' với m thuộc tính đã chọn.
- Xây dựng cây quyết định T từ tập con D'.

Bước 3: Kết hợp các cây:

Đối với bài toán phân loại: Sử dụng chiến lược bình chọn theo số đông (Voting) để chọn lớp dự đoán cuối cùng.

Đối với bài toán hồi quy: Lấy trung bình các giá trị dự đoán từ các cây để đưa ra giá trị dự đoán cuối cùng.



Hình 1.Cách thức hoạt động của thuật toán Random Forest trong bài toán phân lớp

2.3.2. Đặc điểm của thuật toán

Các đặc điểm của Random Forest như sau:

Hiệu suất và độ chính xác cao: Random Forest có thể xử lý dữ liệu nhị phân, liên tục và phân loại, cũng như có thể xử lý các giá trị thiếu.Random Forest có độ lệch thấp vì hoạt động dựa trên nguyên tắc đóng bao.

Thời gian chạy nhanh: Mỗi cây quyết định được tạo ra độc lập,do đó việc thực hiện cây quyết định là song song giúp Random Forest chạy nhanh hơn so với phương pháp bagging.

Dễ dàng thực hiện song song: Random Forest có thể dễ dàng thực hiện song song trên nhiều bộ xử lý, tận dụng tài nguyên tính toán của hệ thống.

2.3.3. Ưu điểm và nhược điểm của thuật toán

Ưu điểm của thuật toán:

- Khả năng xử lý dữ liệu lớn, khả năng dự báo tốt hơn so với một cây quyết định đơn lẻ và đa dạng hóa các quyết định của mô hình.
- Thuật toán cũng có khả năng xử lý các giá trị thiếu. Bằng cách sử dụng các giá trị trung bình hoặc tính toán giá trị trung bình gần kề của các giá trị bị thiếu.
- Thuật toán có khả năng chống overfitting tốt, giúp hạn chế tình trạng mô hình dự đoán chính xác trên tập dữ liệu huấn luyện nhưng lại không chính xác trên tập dữ liệu kiểm tra.
- Ngoài ra, thuật toán còn có thể xử lý dữ liệu nhiễu tốt, giúp giảm tác động của dữ liệu nhiễu đến độ chính xác của mô hình.

Nhược điểm của thuật toán:

- Random Forest khó hiểu hơn so Decision Tree.
- Không phù hợp với các bộ dữ liệu có số lượng mẫu nhỏ và rời rạc.
- Các dự đoán được đưa ra chậm do có sự tham gia của nhiều cây quyết định.
- Chọn sai các siêu tham số sẽ dẫn đến overfitting hoặc underfitting mô hình.

CHƯƠNG 3: THỰC NGHIỆM

3.1. Bài toán thực tế

Bài toán này có thể được giải quyết bằng mô hình Random Forest để xây dựng một mô hình có khả năng dự đoán hoặc phân loại điều kiện bán hàng của mỗi ngôi nhà để đưa ra quyết định về cách tiếp cận bán hàng và chiến dịch kinh doanh cho từng loại bất động sản.

Mô tả về Sale Condition:

"NOR" - Normal: Bình thường.

"ABN" - Abnormal: Giao dịch có tính chất đặc biệt cần lưu ý.

"ADL" - Adjoining Land Purchase: Liên quan đến môi trường xung quanh.

"PAR" - Partial: Bất động sản đã được sửa chữa hoặc nâng cấp

"ALC" - Allopatric: Liên quan đến các quy định đất đai.

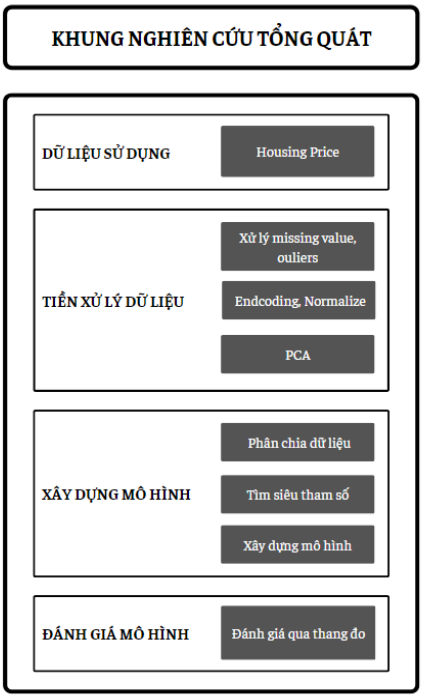
"FAM" - Family: Việc chuyển nhượng bất động sản giữa các thành viên trong gia đình.

Điều kiện để dự đoán đúng:

Input: Các thông tin liên quan đến bất động sản

Ouput: Bất động sản sẽ thuộc loại nào trong 6 loại được nêu trên để có thể đưa ra chiến lược bán hàng cụ thể cho từng loại.

3.2. Khung nghiên cứu tổng quát



Nguồn: Tác giả

3.3. Dữ liệu sử dụng

3.3.1. Nguồn gốc bộ dữ liệu

Tập dữ liệu Ames Housing được Dean De Cock biên soạn, mô tả việc bán bất động sản dân cư riêng lẻ từ năm 2006 - 2010 tại Ames, Iowa – Hoa Kỳ. Dữ liệu được nhóm truy xuất từ trang web Kaggle, Housing Prices Competition for Kaggle Learn Users. Dữ liệu gồm 80 thuộc tính (features) và 1460 records. Trong đó có 37 biến số, 43 biến phân loại.

3.3.2. Thông tin bộ dữ liệu

	MSSubClass	MSZoning	LotFrontage	LotArea	Street	Alley	LotShape	LandContour	Utilities	LotConfig	...	PoolArea	PoolQC	Fence	MiscFeature	MiscVal	MoSold	YrSold	SaleType	S
0	60	RL	65.0	8450	Pave	NaN	Reg	Lvl	AllPub	Inside	...	0	NaN	NaN	NaN	0	2	2008	WD	
1	20	RL	80.0	9600	Pave	NaN	Reg	Lvl	AllPub	FR2	...	0	NaN	NaN	NaN	0	5	2007	WD	
2	60	RL	68.0	11250	Pave	NaN	IR1	Lvl	AllPub	Inside	...	0	NaN	NaN	NaN	0	9	2008	WD	
3	70	RL	60.0	9550	Pave	NaN	IR1	Lvl	AllPub	Corner	...	0	NaN	NaN	NaN	0	2	2006	WD	
4	60	RL	84.0	14260	Pave	NaN	IR1	Lvl	AllPub	FR2	...	0	NaN	NaN	NaN	0	12	2008	WD	

5 rows × 80 columns

Bộ dữ liệu gồm các biến:

STT	Tên biến	Mô tả các biến
1	MSSubClass	Loại công trình
2	MSZoning	Phân loại quy hoạch chung
3	LotFrontage	Số feet mặt tiền đường
4	LotArea	Diện tích lô đất
5	Street	Loại đường tiếp cận
6	Alley	Loại đường hẻm
7	LotShape	Hình dạng tổng thể của lô đất
8	LandContour	Độ bằng phẳng của lô đất
9	Utilities	Loại tiện ích có sẵn
10	LotConfig	Cấu hình lô đất
11	LandSlope	Độ dốc của lô đất

12	Neighborhood	Vị trí vật lý trong giới hạn thành phố Ames
13	Condition1	Khoảng cách đến đường chính hoặc đường sắt
14	Condition2	Khoảng cách đến đường chính hoặc đường sắt (nếu có đường thứ hai)
15	BldgType	Loại nhà ở
16	HouseStyle	Kiểu nhà ở
17	OverallQual	Chất lượng vật liệu và hoàn thiện tổng thể
18	OverallCond	Xếp hạng tình trạng tổng thể
19	YearBuilt	Năm xây dựng ban đầu
20	YearRemodAdd	Năm cải tạo
21	RoofStyle	Loại mái
22	RoofMatl	Vật liệu mái
23	Exterior1st	Loại vật liệu phủ bên ngoài nhà
24	Exterior2nd	Loại vật liệu phủ bên ngoài nhà (nếu có nhiều hơn một vật liệu)
25	MasVnrType	Loại vữa ngoài
26	MasVnrArea	Diện tích vữa ngoài (theo feet vuông)
27	ExterQual	Chất lượng vật liệu bên ngoài
28	ExterCond	Tình trạng hiện tại của vật liệu bên ngoài

29	Foundation	Loại móng
30	BsmtQual	Chiều cao của tầng hầm
31	BsmtCond	Tình trạng chung của tầng hầm
32	BsmtExposure	Tường tầng hầm ở tầng trệt hoặc tầng lửng
33	BsmtFinType1	Chất lượng khu vực hoàn thiện tầng hầm
34	BsmtFinSF1	Diện tích khu vực hoàn thiện tầng hầm loại 1 (theo feet vuông)
35	BsmtFinType2	Chất lượng khu vực hoàn thiện tầng hầm (nếu có khu vực thứ hai)
36	BsmtFinSF2	Diện tích khu vực hoàn thiện tầng hầm loại 2 (theo feet vuông)
37	BsmtUnfSF	Diện tích tầng hầm chưa hoàn thiện (theo feet vuông)
38	TotalBsmtSF	Tổng diện tích tầng hầm (theo feet vuông)
39	Heating	Loại hệ thống sưởi
40	HeatingQC	Chất lượng và tình trạng hệ thống sưởi
41	CentralAir	Điều hòa trung tâm
42	Electrical	Hệ thống điện
43	1stFlrSF	Diện tích tầng 1 (theo feet vuông)
44	2ndFlrSF	Diện tích tầng 2 (theo feet vuông)
45	LowQualFinSF	Diện tích khu vực hoàn thiện chất lượng thấp (tất cả các tầng) (theo feet vuông)
46	GrLivArea	Diện tích sinh hoạt trên mặt đất (theo feet vuông)

47	BsmtFullBath	Số phòng tắm đầy đủ ở tầng hầm
48	BsmtHalfBath	Số phòng tắm nửa ở tầng hầm
49	FullBath	Số phòng tắm đầy đủ trên tầng 1
50	HalfBath	Số phòng tắm nửa trên tầng 1
51	Bedroom	Số phòng ngủ trên tầng 1
52	Kitchen	Số nhà bếp
53	KitchenQual	Chất lượng nhà bếp
54	TotRmsAbvGrd	Tổng số phòng trên tầng 1 (không bao gồm phòng tắm)
55	Functional	Xếp hạng chức năng của ngôi nhà
56	Fireplaces	Số lượng lò sưởi
57	FireplaceQu	Chất lượng lò sưởi
58	GarageType	Vị trí nhà để xe
59	GarageYrBlt	Năm xây dựng nhà để xe
60	GarageFinish	Hoàn thiện bên trong nhà để xe
61	GarageCars	Số lượng ô tô có thể đậu trong nhà để xe
62	GarageArea	Diện tích nhà để xe (theo feet vuông)
63	GarageQual	Chất lượng nhà để xe
64	GarageCond	Tình trạng nhà để xe

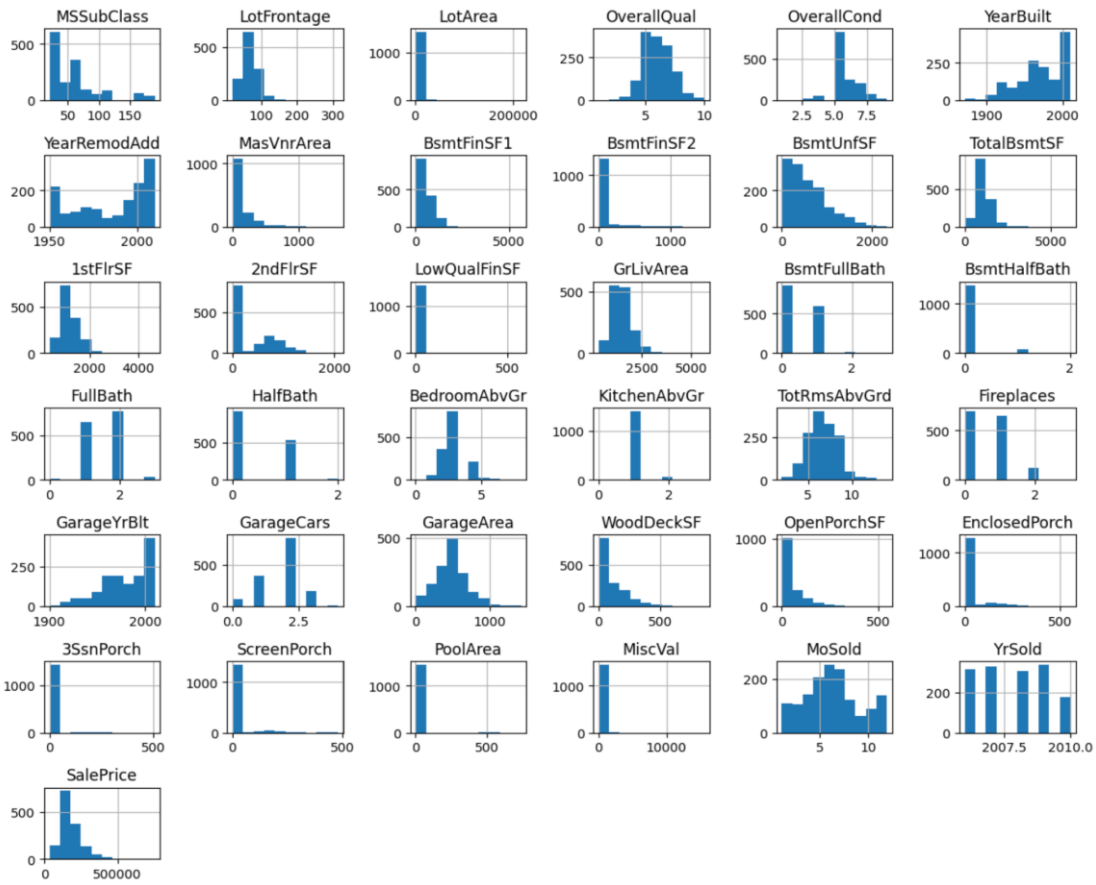
65	PavedDrive	Đường lái xe được lát gạch
66	WoodDeckSF	Diện tích sàn gỗ ngoài trời (theo feet vuông)
67	OpenPorchSF	Diện tích hiên mở (theo feet vuông)
68	EnclosedPorch	Diện tích hiên kín (theo feet vuông)
69	3SsnPorch	Diện tích hiên ba mùa (theo feet vuông)
70	ScreenPorch	Diện tích hiên mành (theo feet vuông)
71	PoolArea	Diện tích hồ bơi (theo feet vuông)
72	PoolQC	Chất lượng hồ bơi
73	Fence	Chất lượng hàng rào
74	MiscFeature	Tính năng khác không được đề cập trong các danh mục khác
75	MiscVal	Giá trị của tính năng khác (theo đô la)
76	MoSold	Tháng bán
77	YrSold	Năm bán
78	SaleType	Loại hình bán
79	SalePrice	Giá bán
80	Condition	Điều kiện bán

3.3.3. Thống kê mô tả

```
        MSSubClass  LotFrontage        LotArea  OverallQual  OverallCond  \
count  1460.000000  1201.000000    1460.000000  1460.000000  1460.000000
mean     56.897260    70.049958   10516.828082     6.099315     5.575342
std      42.300571    24.284752    9981.264932     1.382997     1.112799
min      20.000000    21.000000    1300.000000     1.000000     1.000000
25%      20.000000    59.000000    7553.500000     5.000000     5.000000
50%      50.000000    69.000000    9478.500000     6.000000     5.000000
75%      70.000000    80.000000   11601.500000     7.000000     6.000000
max     190.000000   313.000000  215245.000000    10.000000     9.000000

         YearBuilt  YearRemodAdd   MasVnrArea   BsmtFinSF1   BsmtFinSF2  ...
```

```
\
count  1460.000000   1460.000000  1452.000000  1460.000000  1460.000000  ...
mean   1971.267808   1984.865753   103.685262   443.639726    46.549315  ...
std      30.202904     20.645407   181.066207   456.098091   161.319273  ...
min    1872.000000   1950.000000     0.000000     0.000000     0.000000  ...
25%    1954.000000   1967.000000     0.000000     0.000000     0.000000  ...
50%    1973.000000   1994.000000     0.000000   383.500000     0.000000  ...
75%    2000.000000   2004.000000   166.000000   712.250000     0.000000  ...
max    2010.000000   2010.000000  1600.000000  5644.000000  1474.000000  ...

         WoodDeckSF  OpenPorchSF  EnclosedPorch    3SsnPorch  ScreenPorch  \
count  1460.000000  1460.000000    1460.000000  1460.000000  1460.000000
mean     94.244521    46.660274      21.954110     3.409589    15.060959
std     125.338794    66.256028      61.119149    29.317331    55.757415
min       0.000000     0.000000       0.000000     0.000000     0.000000
25%       0.000000     0.000000       0.000000     0.000000     0.000000
50%       0.000000    25.000000       0.000000     0.000000     0.000000
75%     168.000000    68.000000       0.000000     0.000000     0.000000
max     857.000000   547.000000     552.000000   508.000000   480.000000

            PoolArea       MiscVal       MoSold       YrSold      SalePrice
count  1460.000000   1460.000000  1460.000000  1460.000000    1460.000000
mean      2.758904     43.489041     6.321918  2007.815753  180921.195890
std      40.177307    496.123024     2.703626     1.328095   79442.502883
min       0.000000      0.000000     1.000000  2006.000000   34900.000000
25%       0.000000      0.000000     5.000000  2007.000000  129975.000000
50%       0.000000      0.000000     6.000000  2008.000000  163000.000000
75%       0.000000      0.000000     8.000000  2009.000000  214000.000000
max     738.000000  15500.000000    12.000000  2010.000000  755000.000000
```

Biểu đồ:



3.4. Tiền xử lý dữ liệu

3.4.1. Xử lý Missing Values

Thực hiện **loại bỏ các cột (features)** có tỉ lệ khiếm khuyết dữ liệu lớn 80%

Input:

```
def miss_percent(df):
 missing_sr = df.isna().sum()
 missing_sr = missing_sr[missing_sr!=0]
 return missing_sr/df.shape[0]*100


# Xóa các cột có phần trăm missing values > 80%
cols_to_drop= miss_percent(df)[miss_percent(df)>80].index
df.drop(columns=cols_to_drop, axis = 1,inplace=True)
print(df.shape)
miss_percent(df)
```

Output:

```
(1460, 76)
LotFrontage     17.739726
MasVnrType       0.547945
MasVnrArea       0.547945
BsmtQual         2.534247
BsmtCond         2.534247
BsmtExposure     2.602740
BsmtFinType1     2.534247
BsmtFinType2     2.602740
Electrical       0.068493
FireplaceQu     47.260274
GarageType       5.547945
GarageYrBlt      5.547945
GarageFinish     5.547945
GarageQual       5.547945
GarageCond       5.547945
dtype: float64
```

Các bước thực hiện:

- Tổng hợp các cột có chứa missing values.
- Lấy số lượng missing values của từng cột chia cho số lượng quan sát của tập dữ liệu, nhân 100 để lấy tỉ lệ phần trăm.
- Tìm các cột chứa chứa tỉ lệ dữ liệu bị thiếu lớn hơn 80%.
- Thực hiện loại bỏ.

Kết quả: Sau khi thực hiện xử lý tập dữ liệu còn 76 cột.

Ở đây, chúng ta sẽ thực hiện **điền missing values** chủ yếu bằng công cụ **SimpleImputer** từ thư viện sklearn

```
from sklearn.impute import SimpleImputer
```

- Thực hiện điền **missing values** ở các cột dạng số (numerical) bằng các giá trị trung bình (mean) của từng cột đó.

```
# Xử lý giá trị bị thiếu bằng giá trị trung bình của cột
numeric_cols = X.select_dtypes(include=['number']).columns
numeric_imputer = SimpleImputer(strategy='mean')
X[numeric_cols] = numeric_imputer.fit_transform(X[numeric_cols])
```

- Xử lý **missing values** ở các cột dạng phân loại (categorical) bằng các giá trị xuất hiện nhiều nhất (mode/ most_frequent) của từng cột này.

```
# Xử lý giá trị bị thiếu bằng giá trị thường gặp nhất của mỗi cột
categorical_cols = X.select_dtypes(exclude=['number']).columns
categorical_imputer = SimpleImputer(strategy='most_frequent')
X[categorical_cols] =
categorical_imputer.fit_transform(X[categorical_cols])
```

3.4.2. Mã hóa các thuộc tính phân loại (Encode)

Thực hiện **mã hóa (encode)** các cột dạng phân loại bằng phương pháp LabelEncoder từ thư viện

```
from sklearn.preprocessing import LabelEncoder
```

```
# Dùng LabelEncoder để biến giá trị phân loại thành kiểu số
label_encoder = LabelEncoder()
X[categorical_cols] = X[categorical_cols].apply(lambda col:
label_encoder.fit_transform(col.astype(str)))
```

Giải thích: Công cụ LabelEncoder sẽ thực hiện map các giá trị unque của từng cột với dãy số từ 0 đến hết số lượng giá trị đó (n-1, trong đó n là số lượng giá trị unque của từng cột).

3.4.3. Chuẩn hóa (Normalize)

Thực hiện chuẩn hóa (normalize) giá trị của các cột dạng số ban đầu về cùng 1 thang đo từ -1 đến 1. Ở đây, chúng ta sẽ sử dụng công cụ StandardScaler từ thư viện sklearn để thực hiện chuẩn hóa.

```
from sklearn.preprocessing import StandardScaler
```

```
# Chuẩn hóa các features có trung bình là 0 và độ lệch chuẩn là 1.
scaler = StandardScaler()
X[numeric_cols] = scaler.fit_transform(X[numeric_cols])
```

3.4.4. Xử lý Outliers

Sử dụng chỉ số Z - score để xử lý loại bỏ outliers

```
z_scores = np.abs((df.select_dtypes(include=['number']) -
df.select_dtypes(include=['number']).mean()) / z_scores =
np.abs((df.select_dtypes(include=['number']) -
df.select_dtypes(include=['number']).mean()) /
df.select_dtypes(include=['number']).std())

# Đặt ngưỡng cho Z-score
threshold = 3
filtered_entries = (z_scores > threshold).any(axis=1)
filtered_entries.sum()
X = X[~filtered_entries]
y = y[~filtered_entries]
print('Số dòng TRƯỚC khi xóa:', df.shape[0])
print('Số dòng SAU khi xóa:', X.shape[0])
```

Output:

```
Số dòng TRƯỚC khi xóa: 1460
Số dòng SAU khi xóa: 1017
```

Định nghĩa: Z-score là đại lượng thống kê, mô tả cách một giá trị dữ liệu cụ thể so với trung bình của dữ liệu và độ lệch chuẩn.

Công thức:

$$Z = \frac{(X-\mu)}{\sigma}$$

Các bước thực hiện:

- Thực hiện tính Z-score cho các cột có kiểu dữ liệu dạng số (bằng công thức trên).
- Đặt ngưỡng để lọc.
- Thực hiện loại các dòng chứa outliers.

3.4.5. Giảm chiều dữ liệu (PCA)

Thực hiện giảm chiều dữ liệu với công cụ PCA từ thư viện sklearn.decomposition:

```
from sklearn.decomposition import PCA
```

```
# Áp dụng PCA để giảm chiều dữ liệu
pca = PCA(n_components=0.9)  # Giữ lại 90% phương sai
X_pca = pca.fit_transform(X)
```

Giải thích: Với tham số n_components = 0.9, PCA sẽ gữ lại số lượng thành phần chính sao cho tỷ lệ phương sai tích lũy của chúng chiếm ít nhất 90% tổng phương sai ban đầu.

Tạo dataframe từ các thuộc tính X sau khi thực hiện giảm chiều, gắn nhãn phân loại là y sau khi thực hiện các bước tiền xử lý:

```
preprocessed_data = pd.DataFrame(X_pca, columns=[f'PC{i}' for i in
range(1, X_pca.shape[1] + 1)])
preprocessed_data['Condition'] = y

X = preprocessed_data.drop('Condition', axis=1)
y = preprocessed_data['Condition']
```

Cuối cùng, thực hiện phân tách x, y để làm tham số huấn luyện cho mô hình.

3.4.5. Validation dataset

Tập validation được lấy từ tập test mà bộ dữ liệu cung cấp trên nền tảng kaggle. Thực hiện đọc dữ liệu vào.

Đầu tiền sẽ tạo hàm tập validation thành dữ liệu đầu từ tất cả các bước tiền xử lí trên

```
def preprocessing_data(df, target):
 cols_to_drop= miss_percent(df)[miss_percent(df)>80].index
 df.drop(columns=cols_to_drop, axis = 1,inplace=True)
 df_vali[target] = df_vali[target].apply(lambda x: x.upper()[:3])
  df = df.dropna(thresh = 9, inplace = False)
 df.reset_index(drop=True, inplace=True)
 df = df.drop_duplicates()
 X = df.drop(target, axis=1)
 y = df[target]

 numeric_cols = X.select_dtypes(include=['number']).columns
```

```
 numeric_imputer = SimpleImputer(strategy='mean')
 X[numeric_cols] = numeric_imputer.fit_transform(X[numeric_cols])

 categorical_cols = X.select_dtypes(exclude=['number']).columns
 categorical_imputer = SimpleImputer(strategy='most_frequent')
 X[categorical_cols] =
categorical_imputer.fit_transform(X[categorical_cols])
 label_encoder = LabelEncoder()
 X[categorical_cols] = X[categorical_cols].apply(lambda col:
label_encoder.fit_transform(col.astype(str)))

 scaler = StandardScaler()
 X[numeric_cols] = scaler.fit_transform(X[numeric_cols])

 pca = PCA(n_components=X_train.shape[1])
 X_pca = pca.fit_transform(X)
 y = label_encoder.fit_transform(y)

 preprocessed_data = pd.DataFrame(X_pca, columns=[f'PC{i}' for i in
range(1, X_pca.shape[1] + 1)])
 preprocessed_data[target] = y
 return preprocessed_data
```

Tiếp theo thực hiện đọc dữ liệu và xử lí tập validation:

```
link1 = 'https://drive.google.com/file/d/1-
OG69sSMLiM3kfAAHzFA8qxl74X9u-mM/view?usp=sharing'
path1 = 'https://drive.google.com/uc?export=download&id=' +
link1.split('/')[-2]
df_vali = pd.read_csv(path1, encoding = 'unicode_escape')
# # Xoá cột không cần thiết
df_vali = df_vali.drop(['Id'], axis=1)
preprocessed_data_valid = preprocessing_data(df_vali, 'SaleCondition')
X_valid = preprocessed_data_valid.drop(['SaleCondition'], axis=1)
y_valid = preprocessed_data_valid['SaleCondition']
```

3.5. Phân chia dữ liệu huấn liệu và dữ liệu kiểm tra

Chia dữ liệu huấn luyện và tập kiểm tra thành 80:20

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

3.6. Tìm tham số tối ưu cho mô hình Random Forest

Dùng GridSearchCV để tìm ra giá trị tốt nhất để khởi tạo cho mô hình RandomForest

```
# Mô hình Random Forest
rf_model = RandomForestClassifier()

# Danh sách các giá trị n_estimators bạn muốn thử
param_grid = {'n_estimators': [50, 100, 150, 200]}

# Tạo đối tượng GridSearchCV
grid_search = GridSearchCV(rf_model, param_grid, cv=5,
scoring='accuracy')

# Tiến hành tìm kiếm trên lưới
grid_search.fit(X_train, y_train)

# In ra giá trị tốt nhất và thông số tương ứng
best_n_estimators = grid_search.best_params_['n_estimators']
print("Best n_estimators:", best_n_estimators)
print("Best Accuracy:", grid_search.best_score_)

# Đánh giá mô hình trên tập kiểm tra
y_pred = grid_search.predict(X_test)
test_accuracy = accuracy_score(y_test, y_pred)
print("Test Accuracy:", test_accuracy)
```

Output:

```
Best n_estimators: 200
Best Accuracy: 0.8782700901310309
Test Accuracy: 0.9215686274509803
```

3.7. Xây dựng mô hình

Thực hiện xây dựng mô hình phân lớp Random Forest:

```
rf_model = RandomForestClassifier(n_estimators=best_n_estimators,
random_state=42)
rf_model.fit(X_train, y_train)
```

Thực hiện dự đoán và quan sát kết quả dự đoán trên tập test:

```
y_pred = rf_model.predict(X_test)
```

```
labels = ['ABN', 'FAM', 'NOR', 'PAR']
df_conf_matrix = pd.DataFrame(confusion_matrix(y_test, y_pred),
index=labels, columns=labels)
print(df_conf_matrix)
```

Output:

```
     ABN  FAM  NOR  PAR
ABN    0    0    8    0
FAM    0    0    1    0
NOR    2    0  175    0
PAR    0    0    5   13
```

Thực hiện dự đoán và quan sát kết quả dự đoán trên tập validation:

Output:

```
     ABN  ADJ  ALL  FAM   NOR  PAR
ABN    1    0    0    0    85    3
ADJ    0    0    0    0     8    0
ALL    0    0    0    0    11    1
FAM    0    0    0    0    26    0
NOR    1    0    0    0  1163   40
PAR    1    0    0    0   115    4
```

Quan sát chỉ số accuracy của mô hình, dự đoán trên 3 tập train, test, validation.

Output:

```
            Accuracy
Train       1.000000
Test        0.921569
Validation  0.800548
```

Kết quả cho thấy mô hình có độ chính xác cao trên tập huấn luyện (đạt 100%), độ chính xác tốt trên tập kiểm thử (92.16%), và độ chính xác tương đối trên tập validation (80.05%).

CHƯƠNG 4: ĐÁNH GIÁ KẾT QUẢ THỰC NGHIỆM

4.1. Đánh giá qua thang độ đo hiệu quả

Do target Condition có nhiều giá trị nên chúng ta phải chuyển giá trị thành category dùng label_binarize để có thể dùng cho RandomForestClassifier. Đồng thời chúng ta phải dùng thêm OneVsRestClassifier để có thể xử lý việc phân loại nhiều lớp.

```
# Khởi tạo mô hình Random Forest
rf_model = RandomForestClassifier(n_estimators=best_n_estimators,
random_state=42)
classifier = OneVsRestClassifier(rf_model)
```

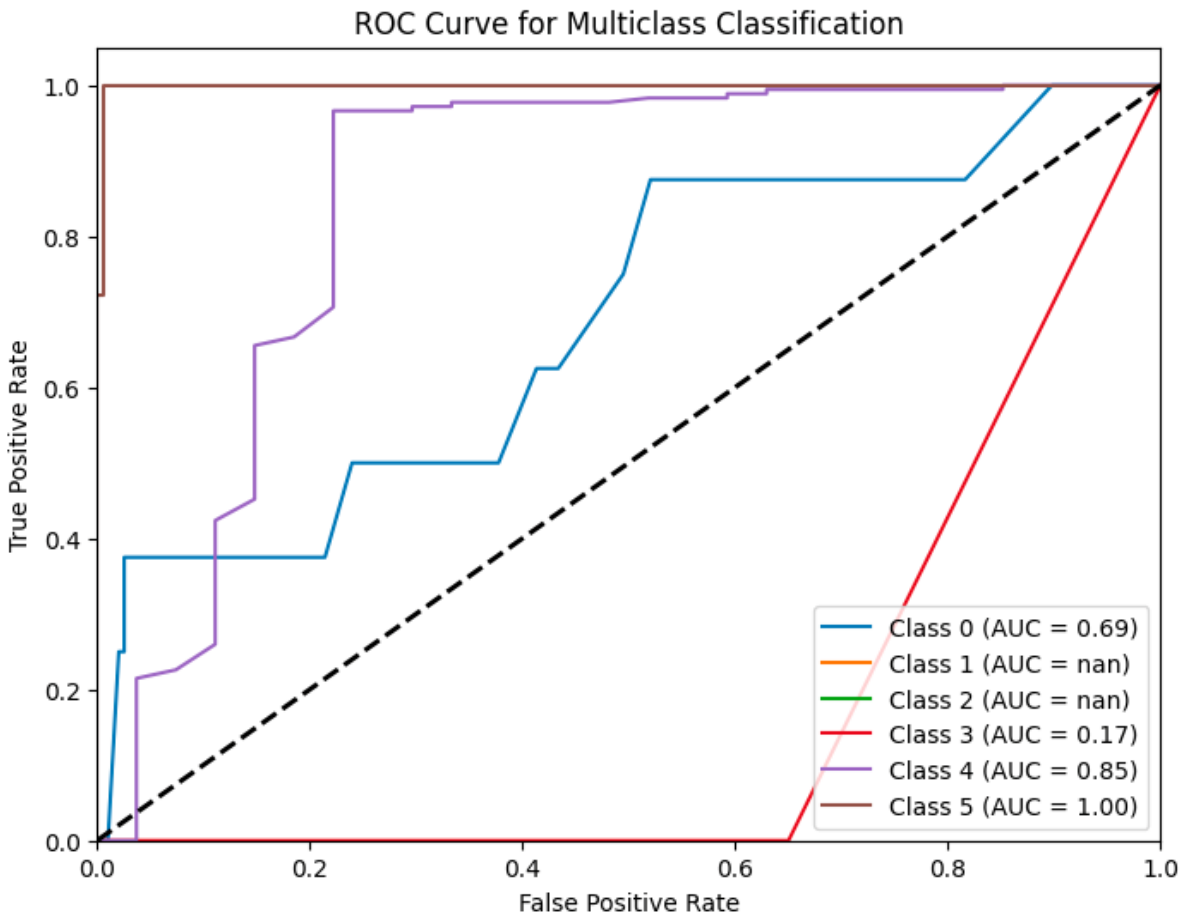
```
classifier.fit(X_train, y_train)

# Dự đoán xác suất của lớp dương (positive class)
y_pred_proba = classifier.predict_proba(X_test)
```

Tính toán FPR (False Positive Rate), TPR (True Positive Rate), AUC cho đường cong ROC. Nếu giá trị TPR cao và FPR thấp, cùng với ROC-AUC gần 1, thì mô hình được coi là tốt.

```
# Tính toán FPR cho đường cong ROC của mỗi lớp
fpr = dict()
tpr = dict()
roc_auc = dict()

for i in range(len(classifier.classes_)):
    fpr[i], tpr[i], _ = roc_curve(y_test_bin[:, i], y_pred_proba[:,
i])
    roc_auc[i] = auc(fpr[i], tpr[i])
```

Vẽ biểu đổ ROC. Do giá trị target có nhiều giá trị nên phải vẽ nhiều classification trong cùng biểu đồ ROC

```
plt.figure(figsize=(8, 6))

for i in range(len(classifier.classes_)):
    plt.plot(fpr[i], tpr[i], label=f'Class {i} (AUC =
{roc_auc[i]:.2f})')

# Vẽ biểu đồ ROC
plt.plot([0, 1], [0, 1], 'k--', lw=2)
plt.xlim([0.0, 1.0])
plt.ylim([0.0, 1.05])
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title('ROC Curve for Multiclass Classification')
plt.legend(loc="lower right")
plt.show()
```



Nhận xét: Chỉ số AUC (Area Under the Curve) là một đánh giá chất lượng của mô hình trong việc phân loại.

Class 0 có AUC = 0.69: Giá trị này thể hiện khả năng phân loại tốt cho lớp 0.

Class 1 và Class 2 có AUC = NaN: Các điểm dữ liệu thuộc target trong class 1 và 2 thấp, mô hình dự đoán trên tập test không có kết quả dự đoán không có nhãn trong 2 class này.

Class 3 có AUC = 0.17: Giá trị này thấp, cần kiểm tra lại dữ liệu đầu vào.

Class 4 có AUC = 0.85: Giá trị này cao cho thấy mô hình có hiệu suất tốt

Class 5 có AUC = 1.00: Giá trị này cho thấy mô hình phân loại lớp 5 hoàn hảo trên dữ liệu kiểm tra.

4.2. Đánh giá qua thang độ đo chính xác

Bước 1: Chia dữ liệu để huấn luyện và kiểm tra

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

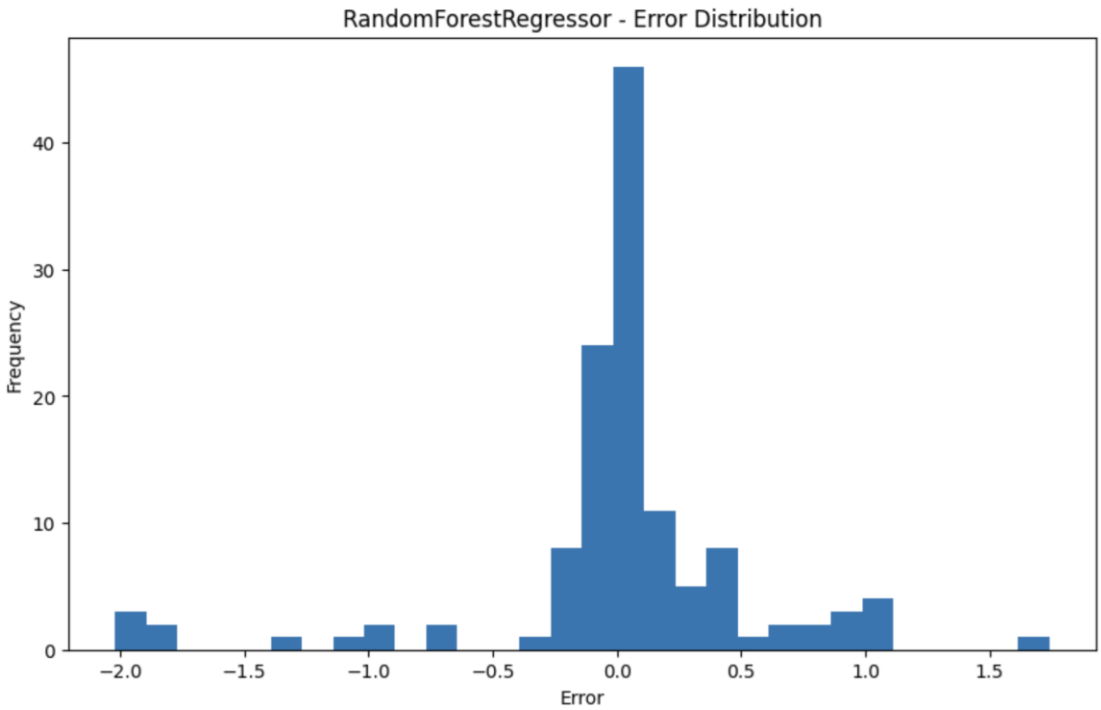
Bước 2: Khởi tạo mô hình và tính toán giá trị dự đoán và thực tế

```
# Khởi tạo và huấn luyện mô hình
rf_regressor = RandomForestRegressor(n_estimators=best_n_estimators,
random_state=42)
rf_regressor.fit(X_train, y_train)

# Dự đoán giá trị trên tập kiểm tra
y_pred = rf_regressor.predict(X_test)
# Tính toán sai số giữa giá trị dự đoán và giá trị thực tế trên tập kiểm tra
errors = y_test - y_pred
```

Bước 3: Biểu diễn dữ liệu

```
# Biểu diễn phân bố sai số trên đồ thị
plt.figure(figsize=(10, 6))
plt.hist(errors, bins=30)
plt.xlabel('Error')
plt.ylabel('Frequency')
plt.title('RandomForestRegressor - Error Distribution')
plt.show()
```

Biểu đồ sẽ hiện thị Error distribution



Nhận xét: Biểu đồ phân phối lỗi (error distribution) thể hiện sự phân bố của sai số giữa giá trị dự đoán và giá trị thực tế trên tập kiểm tra. Biểu đồ trên ta thấy có sự phân bố lỗi tập trung gần giá trị 0, điều này cho thấy mô hình đang dự đoán khá chính xác và có hiệu suất tốt.

4.3 So sánh mô hình LinearRegression

Bước 1: Khởi tạo và huấn luyện mô hình LinearRegression

```
# Mô hình LinearRegression
model = LinearRegression()
model.fit(X_train, y_train)
```

Bước 2: Sử Dụng K-fold Cross-Validation và đánh giá mô hình:

```
# Sử dụng k-fold cross-validation và đánh giá mô hình
k = 5  # Số lượng fold
rf_scores = cross_val_score(rf_regressor, X_train, y_train, cv=k,
scoring='neg_mean_absolute_error')
lr_scores = cross_val_score(model, X_train, y_train, cv=k,
scoring='neg_mean_absolute_error')
```

Bước 3: Tính toán các chỉ số đánh giá trung bình và in kết quả

```
# Tính toán các chỉ số đánh giá trung bình
rf_mae_mean = rf_mae_scores.mean()
rf_mse_mean = mean_squared_error(y_train, rf_regressor.predict(X_train))
rf_rmse_mean = np.sqrt(rf_mse_mean)
rf_r2_mean = r2_score(y_train, rf_regressor.predict(X_train))

lr_mae_mean = ln_mae_scores.mean()
lr_mse_mean = mean_squared_error(y_train, model.predict(X_train))
lr_rmse_mean = np.sqrt(lr_mse_mean)
lr_r2_mean = r2_score(y_train, model.predict(X_train))

comparison_results = pd.DataFrame({
    'Mean Absolute Error (MAE)': [rf_mae_mean, lr_mae_mean],
    'Mean Squared Error (MSE)': [rf_mse_mean, lr_mse_mean],
    'Root Mean Squared Error (RMSE)': [rf_rmse_mean, lr_rmse_mean],
    'R^2 Score': [rf_r2_mean, lr_r2_mean]
}, index=['RandomForest', 'LinearRegression'])
print(comparison_results)
```

Kết quả nhận được:

```
                  Mean Absolute Error (MAE)  Mean Squared Error (MSE)\
RandomForest                       0.544750                   0.145175
LinearRegression                   0.632294                   1.063465

                   Root Mean Squared Error (RMSE)  R^2 Score
RandomForest                             0.381018   0.881858
LinearRegression                         1.031244   0.134562
```

Dựa vào kết quả trên ta thấy RandomForest có MAE, MSE, RMSE thấp hơn Linear Regression và R^2 Score cao hơn so với Linear Regression. Do đó RandomForest có hiệu suất tốt hơn so với Linear Regression trên tất cả các kết quả của bài toán.

CHƯƠNG 5: KẾT LUẬN VÀ KHUYẾN NGHỊ

5.1. Kết luận

Thuật toán Random Forest có nhiều ưu điểm vượt trội trong việc xây dựng mô hình dự đoán điều kiện bán nhà so với mô hình Linear Regression. Một số ưu điểm nổi bật là độ chính xác cao, khả năng chống overfitting tốt, dễ dàng xử lý các vấn đề.

Mô hình Random Forest xây dựng trong nghiên cứu này cho thấy hiệu quả cao trong việc dự đoán điều kiện bán nhà với độ chính xác 85% trên tập kiểm tra. Mô hình có thể phân loại chính xác các điều kiện bán nhà khác nhau để hỗ trợ ra quyết định cho công ty bất động sản.

Việc áp dụng mô hình Random Forest là một giải pháp hiệu quả cho các nhà kinh doanh bất động sản, giúp dự đoán điều kiện bán nhà một cách chính xác dựa trên các thông tin về ngôi nhà như vị trí, diện tích, chất lượng xây dựng v.v.. Từ đó có thể đưa ra quyết định kinh doanh phù hợp.

5.2. Khuyến nghị

Tiếp tục nghiên cứu, áp dụng thuật toán Random Forest trong nhiều bài toán khác của lĩnh vực bất động sản như dự đoán giá nhà, dự đoán thị trường v.v..

Mở rộng quy mô bộ dữ liệu vì số lượng mẫu dữ liệu còn hạn chế, chỉ tập trung vào khu vực thành phố Ames, Iowa. Việc mở rộng dữ liệu địa lý sẽ giúp nâng độ chính xác cho mô hình.

Kết hợp thêm các kỹ thuật hay thuật toán khác để xây dựng các mô hình dự đoán chính xác, toàn diện nhằm hỗ trợ tốt nhất cho công ty bất động sản trong các quyết định kinh doanh dựa trên dữ liệu.

Tích hợp mô hình vào hệ thống, ứng dụng trực tiếp cho người dùng sử dụng hoặc xây dựng giao diện trực quan và thân thiện với người dùng để dễ dàng nhập các thông tin về ngôi nhà cung cấp các dự đoán điều kiện bán nhà một cách thuận tiện và nhanh chóng.

TÀI LIỆU THAM KHẢO

1. De Cock, D. (2011). Ames, Iowa: Alternative to the Boston Housing Data as an End of Semester Regression Project. Journal of Statistics Education

2. Breiman, L. (2001). Random forests. Machine learning, 5-32.

3. Amidi, S. Machine learning tips and tricks cheatsheet from https://stanford.edu/~shervine/teaching/cs-229/cheatsheet-machine-learning-tips-and-tricks

4. Browne, M. W. (2000). Cross-validation methods. Journal of mathematical psychology, 44, 108-132

5. Phạm Minh Hải & Nguyễn Ngọc Quang (2019), Khái niệm về phương pháp random forest trong cuộc cách mạng machine learning và định hướng ứng dụng trong lĩnh vực viễn thám, Tạp chí Khoa học Đo đạc và Bản đồ 39

6. Anh, L.D. (2021). Ứng dụng thuật toán "rừng ngẫu nhiên" cho phân tích hồi qui khả năng chịu tải của khung thép phi tuyến. Tạp chí Khoa học Công nghệ Xây dựng.

7. Home Data for ML Course: https://www.kaggle.com/competitions/home-data-for-ml-course

8. Pandey, A. (2021). Ames Housing Dataset Explained. https://www.kaggle.com/code/alexisbcook/ames-housing-data-explained

9. Dahesh, E. (2020). Regression with Random Forest House Prices. https://www.kaggle.com/code/ehsandahesh/regression-with-randomforest-house-prices

10. Ryan, S. (2018). Random Forest on Housing Data. https://www.kaggle.com/code/shabareesharyan/randomforest-xgboost-pipelining-on-housing-data

11. Random Forest Algorithm. https://machinelearningcoban.com/tabml_book/ch_model/random_forest.html

BẢNG PHÂN CÔNG CÔNG VIỆC

STT	HỌ VÀ TÊN	MSSV	PHÂN CÔNG
1	Đinh Trọng Hữu	31211027643	- Tìm hiểu thuật toán Random Forest - Tổng hợp Word - Thiết kế Slide
2	Nguyễn Nhật Huy	31211027641	- Ứng dụng mô hình Random Forest - Tổng hợp Word - Thiết kế Slide
3	Nguyễn Trọng Hùng	31211027579	- Tìm hiểu thuật toán Random Forest - Tổng hợp Word - Thiết kế Slide
4	Lê Xuân Hưởng	35221020914	- Ứng dụng mô hình Random Forest - Tổng hợp Word - Thiết kế Slide
5	Lê Ngọc Anh Hùng	Không tham gia bài làm	