ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC BÁCH KHOA
KHOA KHOA HỌC - KỸ THUẬT MÁY TÍNH

CÔNG NGHỆ PHẦN MỀM (MỞ RỘNG)

Bài toán Định giá

GVHD: Quản Thành Thơ
SV: Trần Hoàng Công Toại - 1912237
Nguyễn Thế Hiệp - 1913396

TP. HỒ CHÍ MINH, THÁNG 12/2021

Mục lục

1 Giới thiệu bài toán định giá nhà

Ước tính chính xác giá trị bất động sản là một vấn đề quan trọng đối với nhiều bên liên quan như chủ sở hữu nhà, người mua nhà, đại lý, nhà đầu tư,... Việc đánh giá giá trị của một bất động sản dĩ nhiên không phải là một việc dễ dàng. Bên cạnh các yếu tố ảnh hưởng chính như kích thước, số lượng phòng và vị trí, thì giá cả cũng nhạy cảm với những thay đổi của thị trường nhu cầu và đặc thù của từng tình huống, chẳng hạn như tài sản cần bán gấp. Để đánh giá chính xác giá của một căn nhà, người ta không chỉ đòi hỏi một sự hiểu biết chuyên môn về thị trường bất động sản mà còn đòi hỏi một sự hiểu biết thật sự tường tận về bản thân thuộc tính của bất động sản đó. Nếu chúng ta có thể nắm bắt kiến thức này bằng cách thu thập dữ liệu, sử dụng các dữ liệu mở, tận dụng sự giúp sức của các thuật toán, chương trình máy tính thì các kiến thức này trở nên dễ tiếp cận hơn với những người dân bình thường, giúp đưa ra quyết định dễ dàng hơn.

Vì vậy, bài toán đặt ra là: Từ dữ liệu là tập hợp những giá rao của các listings trên thị trường bất động sản, định giá tự động giá trị của một căn nhà cần bán hoặc cho thuê.

- Tham số đầu vào: tập hợp dữ liệu của nhiều loại listings (listing pool) và các loại dữ liệu liên quan.
- Tham số đầu ra: giá của một căn nhà đang cần bán hoặc cần cho thuê.

2 Mô tả dữ liệu

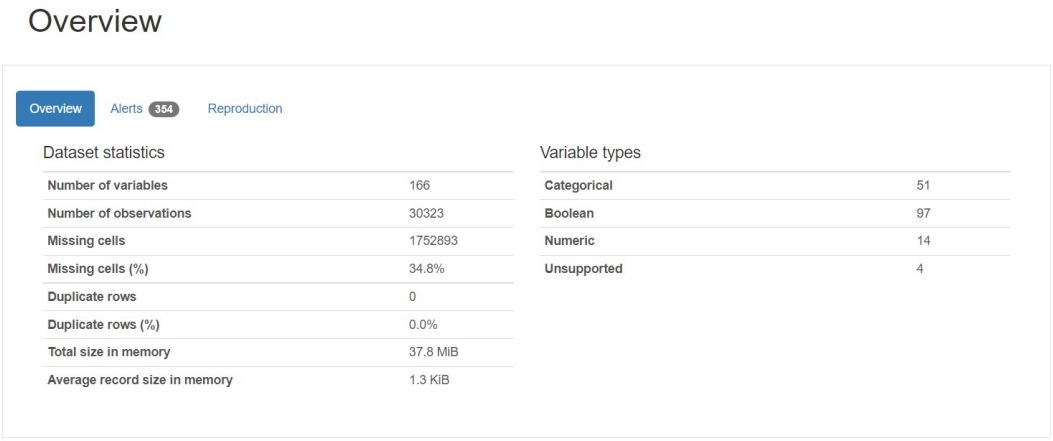
Dữ liệu dùng để huấn luyện mô hình là các thông tin chi tiết của một căn hộ như: diện tích, vị trí, số phòng ngủ,...

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
1	furniture_st	has_video	num_bed_r	area	name	created_tim	full_name	property_ty	title_en	tags	has_media	direction	published_t	alias	ownership	alias_en	is_hot	amenities	project_id	has_3d
2	Full	FALSE	1	56	Flora Fuji	1.531E+12	A-12.07 Flo	1	Căn hộ Flori	['Bán căn hộ 1 pl	1	1	1.531E+12	can-ho-flori	so-hong	can-ho-flora	FALSE	{'pool': True	151969842	FALSE
3	Empty	FALSE	1	56	Flora Fuji	1.531E+12	A-12.08 Flo	1	Căn hộ Flori	['Cho thuê căn h	1	0	1.531E+12	can-ho-flori	so-hong	can-ho-flora	FALSE	{'pool': True	151969842	FALSE
4	Full	FALSE	3	120.1	Trần Trọng (	1.595E+12	18 Trần Trọ	4		['nhà phố quận 7', 'mua bán r	2		1.595E+12	ban-biet-thu-duong-tran-trong-cung			FALSE	{'basement': False, 'smai		FALSE
5	Full	FALSE	2	63.1	The CBD Pre	1.548E+12	A7-10 The C	1	Bán căn hộ	['Căn hộ CBD', 'T	8		1.55E+12	ban-can-ho	hd-mua-bar	ban-can-ho	FALSE	{'pool': True	1482820768	FALSE
6	Full	FALSE	4	0	Đường 9, Bì	1.5E+12	34/6 Đường	8	Nhà phố đư	['1 trệt 3 lầu', 'Nl	3		1.5E+12	nha-pho-du	so-hong	nha-pho-du	FALSE	{'pool': False, 'open24h':		FALSE
7	Basic	FALSE	1	39	Tầng 21, Sur	1.596E+12	B21.25 Tầng	1		['căn hộ Sunrise CityView', 'ci	0		1.596E+12	cho-thue-can-ho-sunrise-cityview-di			FALSE	{'basement':	1540521226	FALSE
8	Empty	FALSE	2	65.5	Tầng 21, Blo	1.578E+12	B21.03 Tầng	1		['Cho thuê căn hộ Saigon Gat	7		1.578E+12	cho-thue-can-ho-saigon-gateway-2p			FALSE	{'basement':	1491562431	FALSE
9	Basic	FALSE	2	89.6	Chung cư Phú	1.597E+12	3.9.6 Chung	1		['chung cư phú mỹ', 'bán căn	1		1.597E+12	ban-can-ho-huong-dong-tang-trung-			FALSE	{'basement':	1505794874	FALSE
10	Empty	FALSE	2	90.8	Huỳnh Tấn F	1.596E+12	1549/16/5 H	8		['nhà phố huỳnh tấn phát', 'nl	2		1.596E+12	ban-nha-huynh-tan-phat-phuong-phi			FALSE	{'basement': False, 'smai		FALSE
11	unknown	FALSE	2	64	Tầng 10, M-	1.597E+12	T1A 10.13 T	1		['thuê căn hộ 2 phòng ngủ', 'c	1		1.597E+12	thue-can-ho-2-phong-ngu-thuoc-tan			FALSE	{'basement':	1503483673	FALSE
12	Empty	FALSE	2	30.9	Mai Văn Vĩn	1.595E+12	10 Mai Văn	8		['bán nhà mai văn vĩnh', 'bán	7		1.595E+12	ban-nha-duong-mai-van-vinh-phuon			FALSE	{'basement': False, 'smai		FALSE
13	Basic	FALSE	3	131.15	Tầng 10, Sky	1.596E+12	2E1.10 Tầng	1		['căn hộ Sky Garden', 'căn hộ	0		1.596E+12	ban-can-ho-sky-garden-phuong-tan-			FALSE	{'basement':	1480951151	FALSE
14	Full	FALSE	2	67	Flora Anh Đ	1.535E+12	6-17 Flora A	1	Bán căn hộ	['căn hộ Flora An	1		1.535E+12	ban-can-ho	so-hong	ban-can-ho	FALSE	{'pool': True	1521455314	FALSE
15	Empty	FALSE	2	71.1	Opal Riversi	1.524E+12	A2-14.03 Op	1	Căn hộ Opa	['Bán căn hộ 2 pl	1	5	1.524E+12	can-ho-opa	hd-mua-bar	can-ho-opa	FALSE	{'pool': True	1491878868	FALSE
16	Full	FALSE	2	68	An gia Skyli	1.596E+12	An gia Skyli	1		['Cho thuê căn hộ An Gia Sky	0		1.596E+12	cho-thue-can-ho-an-gia-skyline-thuc			FALSE	{'basement':	1495852093	FALSE
17	Basic	FALSE	1	56.15	Tầng 4, Flor	1.596E+12	A.423 Tầng	1		['Bán căn hộ Flora Fuji', 'Bán	8		1.596E+12	ban-can-ho-flora-fuji-tang-thap-1-ph			FALSE	{'basement':	1519698428	FALSE
18	Basic	FALSE	1	55	Tầng 13, Ch	1.604E+12	D.13.10 Tầ	1		['căn hộ 1 phòng ngủ', 'căn hộ	1		1.604E+12	can-ho-pham-viet-chanh-day-du-noi			FALSE	{'basement':	1509434985	FALSE
19	Full	FALSE	2	57	Tầng 4, Chu	1.604E+12	A2.A Tầng 4	1		['căn hộ 2 phòng ngủ', 'căn hộ	8		1.604E+12	can-ho-chung-cu-bui-minh-truc-tang			FALSE	{'basement':	1603524378	FALSE
20	Basic	FALSE	2	67	Tầng 4, IDIC	1.606E+12	B.4.3 Tầng 4	1		['căn hộ 2 phòng ngủ', 'căn hộ	0		1.606E+12	can-ho-idico-noi-that-co-ban-huong			FALSE	{'basement':	1508841851	FALSE
21	Basic	FALSE	1	46	Tầng 1, Chu	1.606E+12	1.11 Tầng 1	1		['căn hộ 1 phòng ngủ', 'căn hộ	0		1.606E+12	can-ho-chung-cu-khuong-viet-noi-th			FALSE	{'basement':	1520771984	FALSE
22	Basic	FALSE	2	61.62	Tầng 9, Bloc	1.606E+12	A9.16 Tầng	1		['căn hộ 2 phòng ngủ', 'căn hộ	1		1.606E+12	can-ho-moonlight-park-view-tang-tr			FALSE	{'basement':	1520867913	FALSE
23	Basic	FALSE	2	67	Tầng 14, MF	1.606E+12	MP4.14.05 T	1		['căn hộ 2 phòng ngủ', 'căn hộ	0		1.606E+12	can-ho-mizuki-park-tang-trung-san-l			FALSE	{'basement':	1520929653	FALSE
24	Full	FALSE	4	168	Lý Thường K	1.521E+12	299/19b Lý	8	Nhà phố 4 p	['Cho thuê nhà p	5		1.522E+12	nha-pho-4-phong-ngu-d	nha-pho-4-p		FALSE	{'pool': False, 'open24h':		FALSE
25	Full	FALSE	6	190	Đường Số 6(	1.523E+12	15 Đường Sc	8	Nhà phố 4 p	['Nhà hướng Bắc	4		1.527E+12	biet-thu-6-p	so-hong	nha-pho-4-p	FALSE	{'pool': False, 'open24h':		FALSE
26	Basic	FALSE	2	84.85	Tầng 21, Cri	1.606E+12	C21.15 Tầng	1		['căn hộ 2 phòng ngủ', 'căn hộ	7		1.606E+12	can-ho-feliz-en-vista-view-noi-khu-v			FALSE	{'basement':	1480951153	FALSE
27		FALSE	0	103.1	Nguyễn Thị	1.607E+12	Nguyễn Thị	16		['đất nền quận 2', 'bán đất nề	6		1.607E+12	dat-nen-huong-dong-bac-hem-xe-ho			FALSE			FALSE
28	Full	FALSE	2	90	Tầng 12, chi	1.603E+12	A.12B.02 Tầ	1		['chung cư Minh Thành', 'căn	1		1.603E+12	can-ho-chung-cu-minh-thanh-day-du			FALSE	{'basement':	1601902321	FALSE
29	Basic	FALSE	2	17	Phan Xích Lo	1.604E+12	343/97 Phar	8		['Nhà phố Phan Xích Long, Ph	8		1.604E+12	nha-pho-hem-phan-xich-long-dien-ti			FALSE	{'basement': False, 'smai		FALSE
30	Full	FALSE	1	26	Tầng 17, Riv	1.606E+12	17.10 Tầng	6		['officetel quận 4', 'rivergate	0		1.606E+12	can-ho-office-tel-rivergate-residenc			FALSE	{'basement':	1601132115	FALSE

rever_dataset

Hình 1: Dữ liệu huấn luyện mô hình

Để có thể quan sát dữ liệu một cách trực quan hơn, nhóm sử dụng thư viện pandas để hiển thị dữ liệu như hình bên dưới:



Hình 2: Tổng quan về dữ liệu huấn luyện

Dựa vào hình trên ta thấy một số đặc điểm của tập dữ liệu như sau:

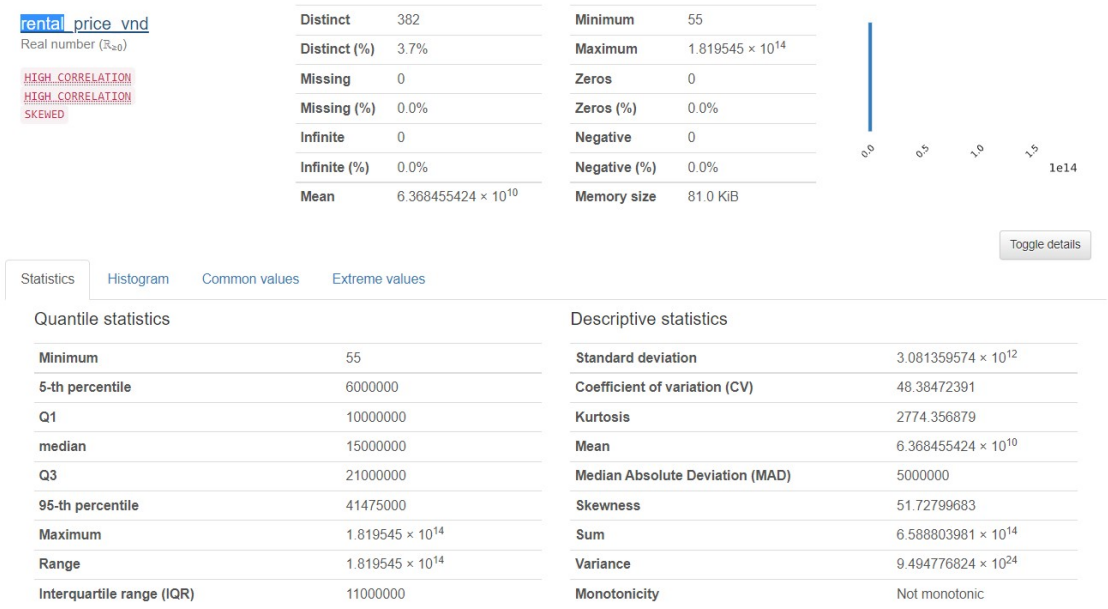
- Có 30323 dữ liệu (entries)
- Gồm 166 đặc trưng (diện tích, số phòng ngủ...)

- Tỉ lệ dữ liệu bị thiếu (mang giá trị NULL) là: 34.8%
- Có 51 đặc trưng dạng category, 97 đặc trưng dạng boolean, 14 đặc trưng dạng numeric và 4 đặc trưng chưa phân loại được (có thể do những cột này đang bị trống dữ liệu)

Vì dữ liệu này bao gồm cả cho giá thuê và giá bán nên ta sẽ tách ra làm 2 để phân tích: 1 tập dữ liệu để huấn luyện mô hình dự đoán giá thuê nhà và 1 tập dữ liệu dùng cho giá bán.

2.1 Tập dữ liệu giá thuê

Ta cũng sử dụng thư viện pandas để phân tích dữ liệu giá thuê nhà.

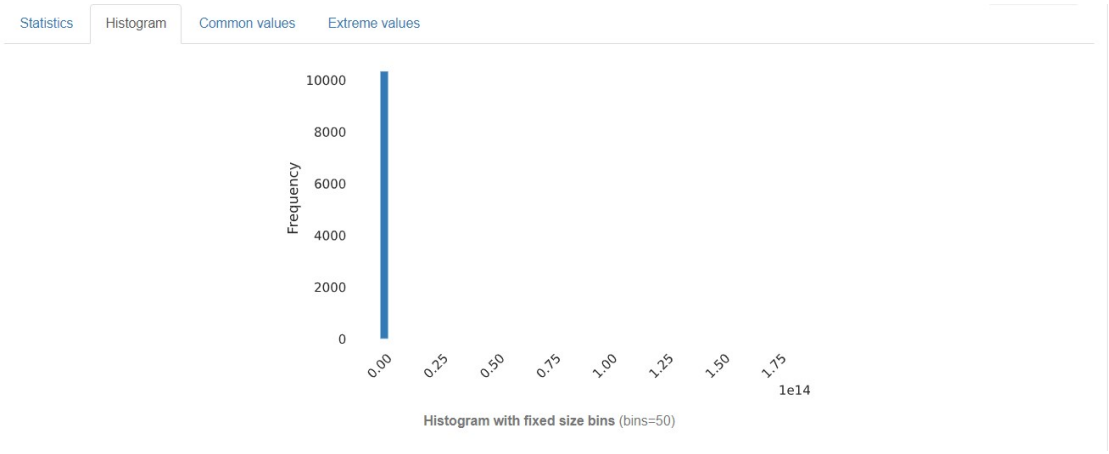


Hình 3: Dữ liệu giá thuê

Dựa vào hình trên ta có một số nhận xét sau về dữ liệu giá thuê nhà:

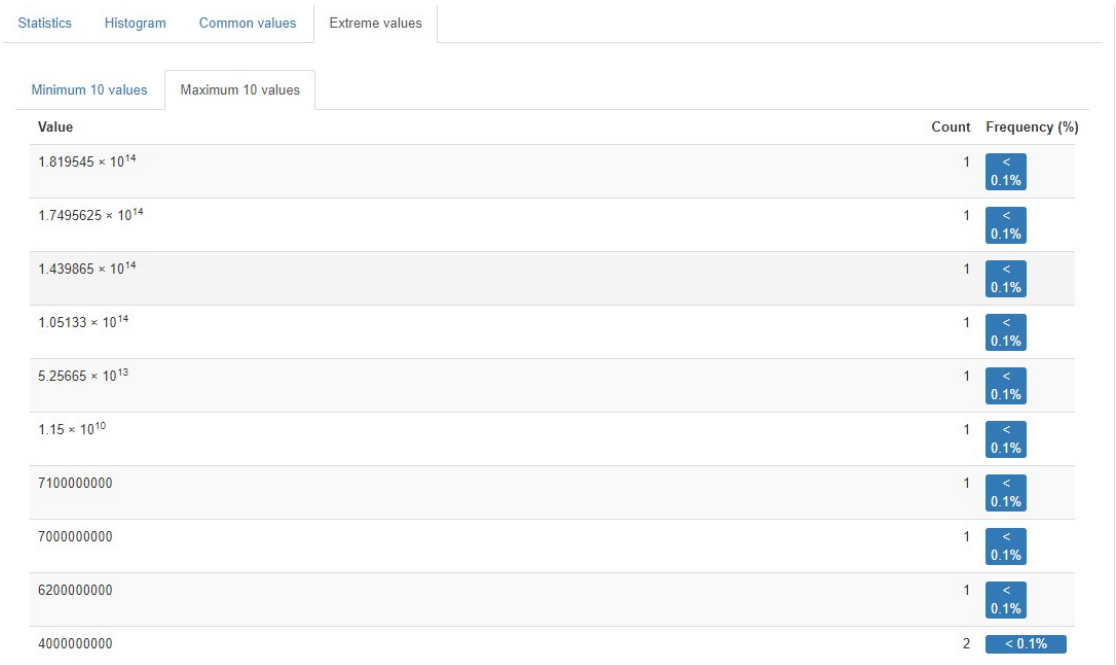
- Số lượng dữ liệu: 10346
- Giá trị nhỏ nhất: 1.819545×10^{14} (VNĐ)
- Giá trị lớn nhất: 55 (VNĐ)
- Trung bình: $6.368455424 \times 10^{10}$ (VNĐ)

Biểu đồ biểu thị giá trị của giá thuê:

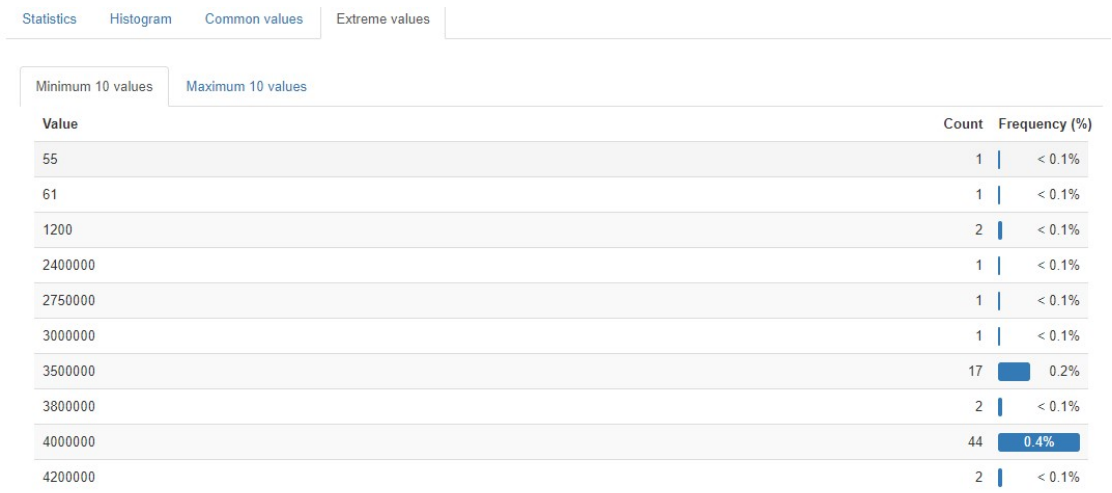


Hình 4: Histogram giá thuê

Ta thấy dữ liệu phân bố ko đồng đều, khoảng giá trị của giá thuê tương đối lớn, có thể do có 1 số điểm dữ liệu nhiễu. Để phát hiện nhiễu, ta liệt kê các điểm giá trị lớn nhất và nhỏ nhất của tập dữ liệu:



Hình 5: Các điểm giá trị lớn nhất



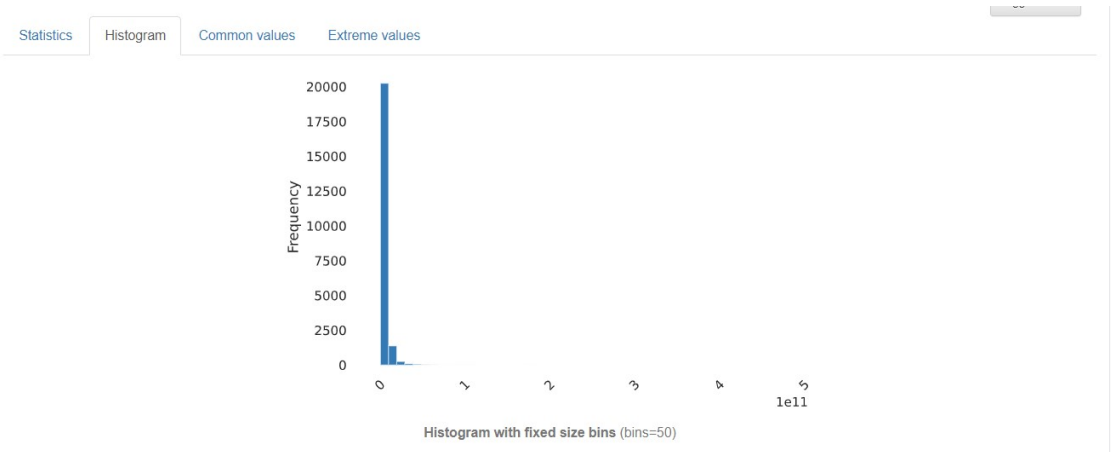
Value	Count	Frequency (%)
55	1	< 0.1%
61	1	< 0.1%
1200	2	< 0.1%
2400000	1	< 0.1%
2750000	1	< 0.1%
3000000	1	< 0.1%
3500000	17	0.2%
3800000	2	< 0.1%
4000000	44	0.4%
4200000	2	< 0.1%

Hình 6: Các điểm giá trị nhỏ nhất

Dựa vào 2 hình trên ta thấy có một số điểm giá trị rất lớn và rất nhỏ (có những giá trị lên tới mũ 14 hay những giá trị rất nhỏ như: 55) nhưng chỉ chiếm dưới 0.1% của tập dữ liệu. Đây có thể là các điểm outlier, nên ta cần loại bỏ các điểm dữ liệu này trước khi đưa vào huấn luyện mô hình.

2.2 Tập dữ liệu giá bán

Ta cũng sử dụng thư viện pandas để phân tích dữ liệu giá bán.



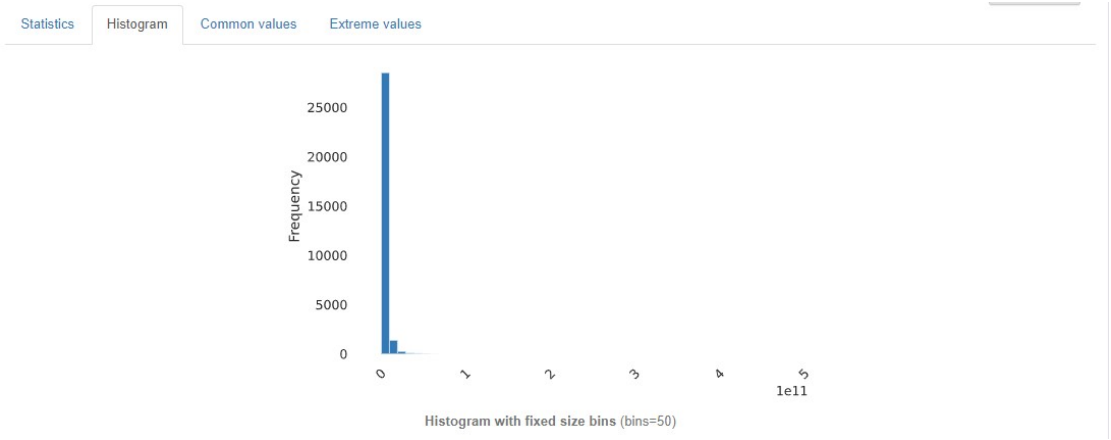
Hình 7: Dữ liệu giá bán

Dựa vào hình trên ta có một số nhận xét sau về dữ liệu giá bán:

- Số lượng dữ liệu: 22042
- Giá trị lớn nhất: 4.8×10^{11} (VNĐ)

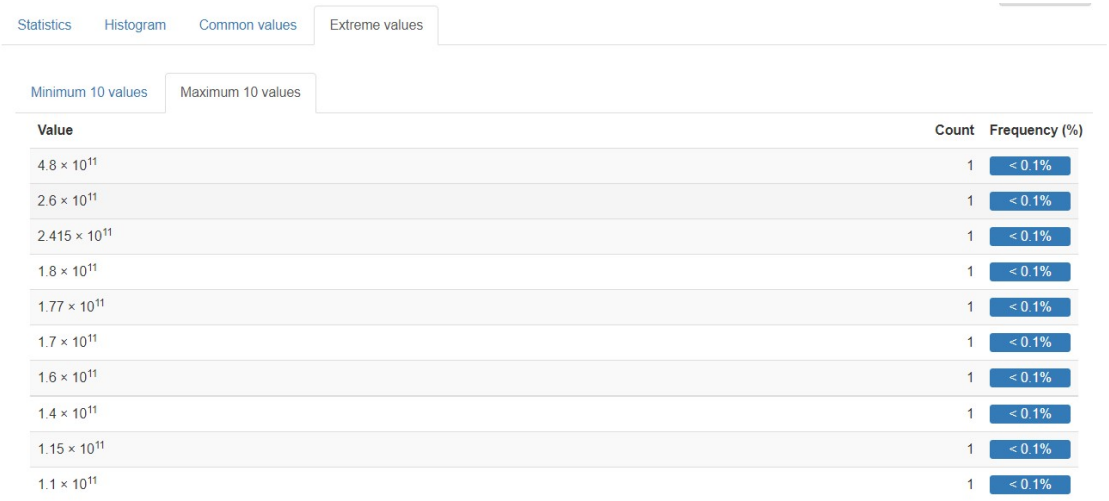
- Giá trị nhỏ nhất: 55 (VNĐ)
- Trung bình: $6.368455424 \times 10^{10}$ (VNĐ)

Biểu đồ biểu thị giá trị của giá bán:



Hình 8: Histogram giá bán

Ta thấy dữ liệu phân bố ko đồng đều, khoảng giá trị của giá thuê tương đối lớn, có thể do có 1 số điểm dữ liệu nhiễu. Để phát hiện nhiễu, ta liệt kê các điểm giá trị lớn nhất và nhỏ nhất của tập dữ liệu:



Value	Count	Frequency (%)
4.8×10^{11}	1	< 0.1%
2.6×10^{11}	1	< 0.1%
2.415×10^{11}	1	< 0.1%
1.8×10^{11}	1	< 0.1%
1.77×10^{11}	1	< 0.1%
1.7×10^{11}	1	< 0.1%
1.6×10^{11}	1	< 0.1%
1.4×10^{11}	1	< 0.1%
1.15×10^{11}	1	< 0.1%
1.1×10^{11}	1	< 0.1%

Hình 9: Các điểm giá trị lớn nhất

Statistics | Histogram | Common values | Extreme values

Minimum 10 values | Maximum 10 values

Value	Count	Frequency (%)
1	5	< 0.1%
2	2	< 0.1%
3	1	< 0.1%
6	1	< 0.1%
11	1	< 0.1%
13	1	< 0.1%
19	1	< 0.1%
69	1	< 0.1%
800	2	< 0.1%
1000	1	< 0.1%

Hình 10: Các điểm giá trị nhỏ nhất

Dựa vào 2 hình trên ta thấy có một số điểm giá trị rất nhỏ như: 1, 2, 4 và chỉ chiếm dưới 0.1% của tập dữ liệu. Đây có thể là các điểm outlier, nên ta cần loại bỏ các điểm dữ liệu này trước khi đưa vào huấn luyện mô hình.

3 Các giải thuật sử dụng

3.1 Hồi quy tuyến tính

Xét một bài toán đơn giản:
Một căn nhà rộng x_1 m^2, có x_2 phòng ngủ và cách trung tâm thành phố x_3 km có giá là bao nhiêu. Giả sử chúng ta đã có số liệu thống kê 1000 căn nhà trong thành phố thì khi có một căn nhà mới với các thông số diện tích, số phòng ngủ và khoảng cách tới trung tâm thành phố thì ta sẽ dự đoán được giá của căn nhà mới đó như thế nào ? Hàm dự đoán $y = f(x)$ sẽ có dạng ra sao ?
Ta thấy được:

- Diện tích nhà càng lớn thì giá nhà càng cao
- Số lượng phòng ngủ càng nhiều thì giá càng cao
- Càng xa trung tâm thành phố thì giá càng giảm

Khi đó, hàm số đơn giản có thể mô tả mối quan hệ giữa giá nhà và 3 đại lượng đầu vào là:

$$y \approx f(x) = \hat{y}$$

$$f(x) = w_1x_1 + w_2x_2 + w_3x_3 + w_0$$

Trong đó: w_1, w_2, w_3, w_0 là các hệ số, x là vecto (x_1, x_2, x_3)
$f(x)$ là một hàm tuyến tính theo các biến x_1, x_2, x_3 Vì vậy, ta cần tìm các hệ số w_0, w_1, w_2, w_3 sao cho giá trị của $\hat{(y)} = f(x)$ sẽ sát với giá trị y thực nhất.
Loss function
Với mỗi điểm dữ liệu thứ i thì độ chênh lệch giữa giá thật và giá nhà dự đoán được tính bằng: $\frac{1}{2}(\hat{y_i} - y_i)^2$. Vậy độ chênh lệch trung bình tại mỗi điểm tính trên toàn bộ dữ liệu là:

$$J = \frac{1}{2} * \frac{1}{N}(\sum_{1}^{N}(\hat{y_i} - y_i)^2)$$

Với N là số điểm dữ liệu.
Đây là hàm mất mát (loss function) của bài toán.
Nhận xét:

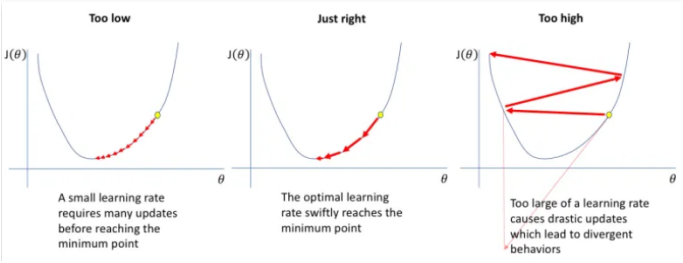
- J không âm
- J càng nhỏ thì giá trị dự đoán càng sát với giá trị thực. Nếu J = 0 thì giá trị dự đoán chính bằng giá trị thực.

Vậy nhiệm vụ của ta là đi tìm các hệ số sao cho giá trị của J là nhỏ nhất.
Thuật toán dùng để tìm giá trị nhỏ nhất của J là Gradient descent.
Gradient descent
Đây là thuật toán tìm giá trị nhỏ nhất của hàm số dựa trên đạo hàm.
Giả sử tìm giá trị của một hàm số $f(x)$ thì ta làm như sau:

1. Khởi tạo $x = x_0$ tùy ý
2. Gán $x = x - learning\_rate * f'(x)$ (learning_rate là hằng số không âm)
3. Tính lại $f(x)$. Nếu $f(x)$ đủ nhỏ thì dừng lại, ngược lại quay lại bước 2.

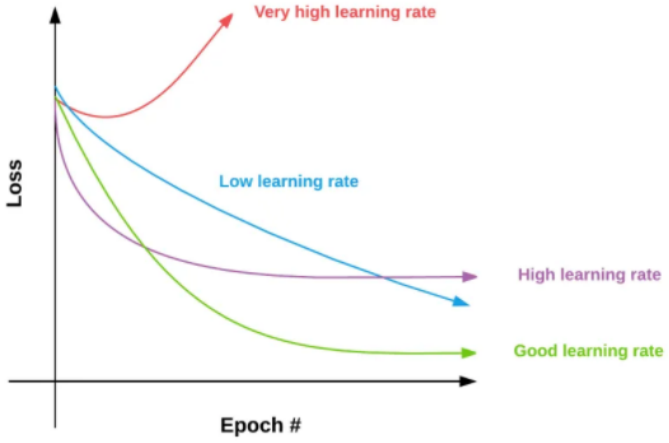
Lặp lại bước 2 với một số lần đủ lớn (100 hoặc 1000 lần tùy vào bái toán và hệ số learning_rate) cho đến khi $f(x)$ đạt giá trị đủ nhỏ.
Khi x tiến gần đến gần điểm đạt giá trị nhỏ nhất thì đạo hàm $f'(x)$ xấp xỉ 0, nên khi đó ở bước 2 giá trị của x thay đổi không đáng kể nữa và gần như là giữ nguyên giá trị của x.
Việc chọn hệ số learnin_rate cực kì quan trọng, có 3 trường hợp:

- Nếu learning_rate nhỏ: mỗi lần hàm số giảm rất ít nên cần rất nhiều lần thực hiện bước 2 để hàm số đạt giá trị nhỏ nhất.
- Nếu learning_rate hợp lý: sau một số lần lặp bước 2 vừa phải thì hàm sẽ đạt giá trị đủ nhỏ.
- Nếu learning_rate quá lớn: sẽ gây hiện tượng overshoot và không báo giờ đạt được giá trị nhỏ nhất của hàm.



Hình 11: Minh họa 3 giá trị learning_rate, Nguồn: *nttuan8.com*

Cách tốt nhát để kiểm tra learning_rate hợp lý là kiểm tra giá trị hàm f(x) sau mỗi lần thực hiện bước 2.



Hình 12: Loss là giá trị hàm f(x), Epoch là số lần thực hiện bước 2, Nguồn: *nttuan8.com*

Với trường hợp bài toán của chúng ta thì thay x ở trên bởi các hệ số w, và đạo hàm sẽ là đạo hàm riêng theo từng w.
Đánh giá hiệu suất mô hình Để biết liệu mô hình có đủ để dự đoán trong tương lại hoặc là mối quan hệ đã xây dựng giữa các biến phụ thuộc và độc lập là đủ hay không thì ta có thể đánh giá qua chỉ số R bình phương hiệu chỉnh:

$$R^2 = 1 - \frac{ESS}{TSS}$$

Trong đó:

- ESS: tổng các độ lệch bình phương phần dư
- TSS: tổng các độ lệch bình phương toàn bộ

R^2 có giá trị từ 0 đến 1, càng gần 1 thì mô hình xây dựng càng phù hợp với bộ dữ liệu.
Ý nghĩa của R^2: Ví dụ $R^2 = 0.6$ thì mô hình phù hợp với tập dữ liệu ở mức 60%. Thông thường, ngưỡng của R^2 phải trên 50% thì mô hình mới gọi là phù hợp.
Ngoài ra còn các chỉ số khác như: Root Mean Square Error (RMSE), Mean Absolute Percentage Error (MAPE), ...
Hạn chế của hồi quy tuyến tính

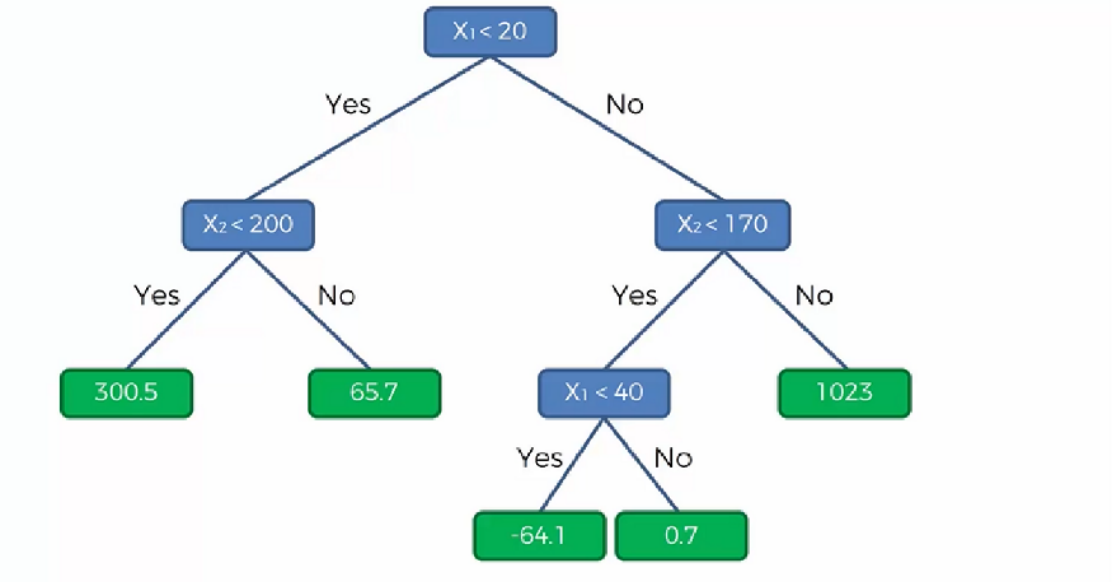
1. Rất nhạy cảm với nhiễu, dữ liệu nhiều làm mô hình sai khác đi rất nhiều. Chính vì vậy nên loại bỏ các nhiễu trước.
2. Không thể biểu diễn được các mô hình quá phức tạp.

3.2 Regression Tree

Hồi quy cây quyết định quan sát các đặc điểm của một đối tượng và đào tạo một mô hình trong cấu trúc của cây để dự đoán dữ liệu trong tương lai nhằm tạo ra đầu ra liên tục có ý nghĩa. Đầu ra liên tục có nghĩa là đầu ra / kết quả không rời rạc, tức là nó không được biểu diễn chỉ bằng một tập hợp số hoặc giá trị rời rạc, đã biết.

Trong cây hồi quy, mỗi nút (không là nút lá) tương ứng cho một quyết định, còn những nút lá tượng trưng cho đầu ra của giải thuật.

Việc xây dựng một cây hồi quy trên dữ liệu huấn luyện cho trước là việc đi xác định các câu hỏi và thứ tự của chúng.



Hình 13: Ví dụ về cây hồi quy

Trong cây hồi quy, để xác định các câu hỏi và thứ tự, chúng ta sẽ dựa vào độ lệch chuẩn (Standard Deviation) và giá trị giảm thiểu độ lệch chuẩn mỗi node phân nhánh (SDR). Việc phân nhánh trong lúc xây dựng mô hình cây quyết định đó chính là việc phân chia tập dữ liệu thành các tập con trên cơ sở chúng sẽ đồng nhất về giá trị sau cùng của biến mục tiêu.

$$SD = \sqrt{\frac{\sum(x - \bar{x})^2}{n}} \tag{1}$$

SDR sẽ bằng độ lệch chuẩn của các giá trị biến mục tiêu xét trên toàn bộ quan sát trừ cho SD của mỗi node. SDR càng lớn thì cách phân nhánh càng tối ưu.

Giá trị của biến mục tiêu sau cùng tại các nút lá (leaf node) sẽ là giá trị trung bình của các đối tượng dữ liệu bên trong node này.

Phương thức để đánh giá mô hình cây hồi quy là sử dụng công thức RMSE (Root Mean Square Error) - thường dùng để xác định tính hiệu quả của các phương trình hồi quy.

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

RMSE chính là tính toán mức độ chênh lệch giữa giá trị dự báo và giá trị quan sát thực tế, RMSE càng nhỏ thì mô hình càng hiệu quả.

Ưu điểm của cây hồi quy:

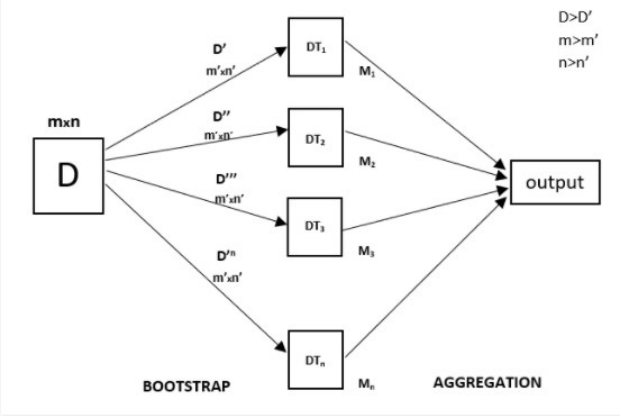
1. Mô hình sinh ra các quy tắc dễ hiểu cho người đọc, tạo ra bộ luật với mỗi nhánh lá là một luật của cây.
2. Dữ liệu đầu vào có thể là là dữ liệu missing, không cần chuẩn hóa hoặc tạo biến giả.

3. Có thể xác thực mô hình bằng cách sử dụng các kiểm tra thống kê.
4. Có khả năng làm việc với dữ liệu lớn.

3.3 Random Forest Regressor

Random là ngẫu nhiên, Forest là rừng, nên ở thuật toán Random Forest ta sẽ xây dựng nhiều cây quyết định bằng thuật toán Decision Tree, tuy nhiên mỗi cây quyết định sẽ khác nhau (có yếu tố random). Sau đó kết quả dự đoán được tổng hợp từ các cây quyết định.

Random Forest là một kỹ thuật tổng hợp có khả năng thực hiện cả nhiệm vụ hồi quy và phân loại với việc sử dụng nhiều cây quyết định và một kỹ thuật được gọi là Bootstrap and Aggregation, thường được gọi là bagging. Ý tưởng cơ bản đằng sau kĩ thuật này là kết hợp nhiều cây quyết định để xác định đầu ra cuối cùng thay vì dựa vào các cây quyết định riêng lẻ.



Hình 14: Kĩ thuật bagging

Các bước xây dựng Random Forest:

1. Lấy ngẫu nhiên n dữ liệu từ bộ dữ liệu với kĩ thuật Bootstrapping, hay còn gọi là random sampling with replacement. Tức khi mình sample được 1 dữ liệu thì mình không bỏ dữ liệu đấy ra mà vẫn giữ lại trong tập dữ liệu ban đầu, rồi tiếp tục sample cho tới khi sample đủ n dữ liệu. Khi dùng kĩ thuật này thì tập n dữ liệu mới của mình có thể có những dữ liệu bị trùng nhau.
2. Sau khi sample được n dữ liệu từ bước 1 thì mình chọn ngẫu nhiên ở k thuộc tính ($k < n$). Giờ mình được bộ dữ liệu mới gồm n dữ liệu và mỗi dữ liệu có k thuộc tính.
3. Dùng thuật toán Decision Tree để xây dựng cây quyết định với bộ dữ liệu ở bước 2.

Thuật toán Random Forest sẽ bao gồm nhiều cây quyết định, mỗi cây được xây dựng dùng thuật toán Decision Tree trên tập dữ liệu khác nhau và dùng tập thuộc tính khác nhau. Sau đó kết quả dự đoán của thuật toán Random Forest sẽ được tổng hợp từ các cây quyết định.

4 Hiện thực giải thuật

Các giải thuật được nhóm triển khai trên môi trường Google Colab.

4.1 Tiền xử lý dữ liệu

Như những phân tích từ dữ liệu trên, đầu tiên là ta loại bỏ đi những giá trị nhiễu:

```
category = ['furniture_status',
       'property_type', 'direction', 'ownership',
       'has_3d', 'service_type',
       'balcony_direction', 'content_status',
       'architectural_style', 'exclusive',
       'pool', 'open24h', 'garage', 'sauna_bath',
       'working_space', 'relax_room', 'elevator', 'gym', 'cable', 'internet', 'pet', 'steam_bath',
       'smart_home', 'tv', 'fridge', 'store_house', 'smart_drying_rig',
       'gas_stove', 'mini_bar', 'microwave', 'helper_room',
       'washing_machine', 'oven', 'fire_detection', 'water_heater',
       'password_lock', 'kitchen_hood', 'dryer', 'sound_equipment',
       'air_conditioner', 'fingerprint_lock', 'security_camera', 'garden',
       'magnetic_card_lock', 'city', 'floor_number',
       'neighborhood_id', 'district_id', 'street_id', 'city_id', 'ward_id',
       'kitchen_cabinet', 'bed', 'sofa',  'dining_table', 'balcony',
       'kitchen_equipment', 'multimedia',
       'makeup_table', 'wardrobe', 'kitchen', 'table',
       'pillow_cushions', 'shoe_cabinet', 'washstand',
       'kitchen_island', 'bathtub',
       'bedside_cupboard', 'decorative_fight',
       'wet_kitchen', 'tv_shelf',
       'bookshelf', 'wall_cabinet', 'ceiling_light', 'toilet_bowl',
       'wood_floor', 'kitchen_cabinet_above', 'table_lamp', 'dry_kitchen', 'tv_cabinet', 'liquor_cabinet']

numeric = ['num_bath_room', 'num_bed_room', 'area']

column = category + numeric + ['rental_price_vnd']
column_sell = category + numeric + ['sell_price_vnd']

df_data_rental = data.drop(index=data[data['rental_price_vnd'] == 0].index)
df_data_rental = df_data_rental.drop(index=df_data_rental[df_data_rental['rental_price_vnd'] < 2000000].index)
df_data_rental = df_data_rental.drop(index=df_data_rental[df_data_rental['rental_price_vnd'] > 500000000].index)
df_data_rental = df_data_rental[column]

df_data_sell = data.drop(index=data[data['sell_price_vnd'] == 0].index)
df_data_sell = df_data_sell.drop(index=df_data_sell[df_data_sell['sell_price_vnd'] < 100000000].index)
df_data_sell = df_data_sell.drop(index=df_data_sell[df_data_sell['sell_price_vnd'] > 100000000000].index)
df_data_sell = df_data_sell[column_sell]
```

Đầu tiên nhóm chọn thuộc tính có độ tương quan cao so với giá thuê và giá bán, đối với tập dữ liệu giá thuê (lưu trong **df_data_rental**) nhóm chỉ lấy giá trị trong khoảng (2.000.000, 500.000.000 VNĐ), đối với tập dữ liệu giá bán (lưu trong **df_data_sell**) nhóm chỉ lấy giá trị

trong khoảng (100.000.000, 100.000.000.000 VNĐ), mục đích là để loại đi những giá trị nhiễu (giá trị quá lớn hoặc quá nhỏ).

Sau khi lọc nhiễu, thì tiếp theo nhóm xử lí những giá trị NULL trong tập dữ liệu, thì phương pháp ở đây nhóm chọn là sẽ thay thế những giá trị NULL bằng những giá trị trong entrie mà có giống với nó nhất:

```
#Replace missing value by K-nearest-neighbor algorithm
_data_rental = df_data_rental.drop("rental_price_vnd", axis = 1).replace(-1, np.NaN)
rental_imputer = KNNImputer(n_neighbors=1, weights='uniform', metric='nan_euclidean')
data_rental = rental_imputer.fit_transform(_data_rental.values)
data_rental = pd.DataFrame(data_rental, index=_data_rental.index, columns=_data_rental.columns)

_data_sell = df_data_sell.drop("sell_price_vnd", axis = 1).replace(-1, np.NaN)
sell_imputer = KNNImputer(n_neighbors=1, weights='uniform', metric='nan_euclidean')
data_sell = sell_imputer.fit_transform(_data_sell.values)
data_sell = pd.DataFrame(data_sell, index=_data_sell.index, columns=_data_sell.columns)
```

Ở đây nhóm sử dụng K-neighbors-nearest để tìm entries mà gần giống với với các entries có dữ liệu bị NULL nhất, rồi sau đó thay thế các giá trị NULL bằng các giá trị trong entries tìm được.

Sau khi tiền xử lý dữ liệu, ta chuẩn bị dữ liệu để huấn luyện mô hình:

```
X_rental= data_rental
y_rental= df_data_rental['rental_price_vnd']

X_sell= data_sell
y_sell= df_data_sell['sell_price_vnd']
```

```
X_rental_train, X_rental_test, y_rental_train, y_rental_test = train_test_split(X_rental,
                                            y_rental, test_size = 0.2, random_state = 42)

X_sell_train, X_sell_test, y_sell_train, y_sell_test = train_test_split(X_sell, y_sell,
                                                  test_size = 0.2, random_state = 42)
```

4.2 Huấn luyện mô hình

Huấn luyện mô hình hồi quy tuyến tính:

```
lin_reg = LinearRegression()
lin_reg.fit(X_rental_train, y_rental_train)
```

Huấn luyện mô hình Regression Tree:

```
tree_reg = DecisionTreeRegressor()
tree_reg.fit(X_rental_train, y_rental_train)
```

Huấn luyện mô hình Random Forest Regressor:

```
forest_reg = RandomForestRegressor(max_features='log2', n_estimators=300, n_jobs=-1, random_state=13)
forest_reg.fit(X_rental_train, y_rental_train)
```

5 Kết quả

5.1 Mô hình giá thuê

Tính MSE và RMSE của từng mô hình

5.1.1 Hồi quy tuyến tính

```
import math
y_rental_test_list = y_rental_test.tolist()
y_rental_predict = lin_reg.predict(X_rental_test)
sum_rental_lin = 0
for index in range(len(y_rental_test)):
  sum_rental_lin += (y_rental_predict[index] - y_rental_test_list[index])**2
mse_rental_lin = sum_rental_lin/len(y_rental_test)
rmse_rental_lin = math.sqrt(mse_rental_lin)
```

5.1.2 Regression Tree

```
y_rental_predict = tree_reg.predict(X_rental_test)
sum_rental_tree = 0
for index in range(len(y_rental_test)):
  sum_rental_tree += (y_rental_predict[index] - y_rental_test_list[index])**2
mse_rental_tree = sum_rental_tree/len(y_rental_test)
rmse_rental_tree = math.sqrt(mse_rental_tree)
```

5.1.3 Random Forest Regressor

```
y_rental_predict = forest_reg.predict(X_rental_test)
sum_rental_forest = 0
for index in range(len(y_rental_test)):
  sum_rental_forest += (y_rental_predict[index] - y_rental_test_list[index])**2
mse_rental_forest = sum_rental_forest/len(y_rental_test)
rmse_rental_forest = math.sqrt(mse_rental_forest)
```

5.1.4 Bảng kết quả

Model	RMSE
Hồi quy tuyến tính	14828805
Random Forest Regressor	10372364
Regression Tree	15098153

Hình 15: RMSE giá thuê của từng mô hình

5.2 Mô hình giá bán

Tính MSE và RMSE của từng mô hình

5.2.1 Hồi quy tuyến tính

```
y_sell_test_list = y_sell_test.tolist()
y_sell_predict = lin_reg_sell.predict(X_sell_test)
sum_sell_lin = 0
for index in range(len(y_sell_test)):
  sum_sell_lin += (y_sell_predict[index] - y_sell_test_list[index])**2
mse_sell_lin = sum_sell_lin/len(y_sell_test)
rmse_sell_lin = math.sqrt(mse_sell_lin)
```

5.2.2 Regression Tree

```
y_sell_predict = tree_reg_sell.predict(X_sell_test)
sum_sell_tree = 0
for index in range(len(y_sell_test)):
  sum_sell_tree += (y_sell_predict[index] - y_sell_test_list[index])**2
mse_sell_tree = sum_sell_tree/len(y_sell_test)
rmse_sell_tree = math.sqrt(mse_sell_tree)
```

5.2.3 Random Forest Regressor

```
y_sell_predict = forest_reg_sell.predict(X_sell_test)
sum_sell_forest = 0
for index in range(len(y_sell_test)):
  sum_sell_forest += (y_sell_predict[index] - y_sell_test_list[index])**2
```

```
mse_sell_forest = sum_sell_forest/len(y_sell_test)
rmse_sell_forest = math.sqrt(mse_sell_forest)
```

5.2.4 Bảng kết quả

Model	RMSE
Hồi quy tuyến tính	4182188359
Random Forest Regressor	3224792924
Regression Tree	4225515443

Hình 16: RMSE giá bán của từng mô hình

6 Nhận xét

Qua kết quả trên có thể thấy mô hình Random Forest Regressor áp dụng cho bài toán hiệu quả nhất vì RMSE của mô hình này nhỏ nhất. Còn với hai mô hình còn lại là Hồi quy tuyến tính và Regression Tree cho ra kết quả khá tương đồng nhau.
Đối với giá thuê thì RMSE khoảng 10 triệu và với giá bán thì RMSE khoảng 3 tỷ. RMSE vẫn còn khá lớn vì trong tập dữ liệu có các giá trị của giá thuê cũng như giá bán có mức độ chênh lệch cao.

TÀI LIỆU THAM KHẢO

[1] Loh, W., 2011. Classification and regression trees. WIREs Data Mining and Knowledge Discovery, 1(1), pp.14-23.

[2] Chen, J., 2020. An Introduction to Machine Learning for Panel Data: Decision Trees, Random Forests, and Other Dendrological Methods. SSRN Electronic Journal.

[3] GeeksforGeeks. 2021. Random Forest Regression in Python - GeeksforGeeks. [online] Available at: <https://www.geeksforgeeks.org/random-forest-regression-in-python/> [Accessed 1 December 2021].

[4] Nttuan8's blog. 2019. Bài 1: Linear Regression và Gradient descent. [online] Available at: <http://nttuan8.com/bai-1:-linear-regression-va-gradient-descent/> [Accessed 1 December 2021].

[5] Tiep Vu's blog. 2016. Bài 3: Linear Regression. [online] Available at: <https://machinelearningcoban.com/2016/12/28/linearregression/> [Accessed 1 December 2021].