HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG
KHOA CÔNG NGHỆ THÔNG TIN

BÁO CÁO THỰC TẬP CƠ SỞ

Chuyên ngành: **CÔNG NGHỆ THÔNG TIN**
Đề tài: Data Analyst sử dụng Python
Dự đoán giá nhà

Giảng viên hướng dẫn:	TS. Đào Ngọc Phong
Sinh viên thực hiện:	Lê Hoàng Long
Mã sinh viên:	B20DCCN405
Lớp:	D20CQCN405
SĐT:	0949404106
Email:	lehoanglonght12@gmail.com

Hà Nội, 4/2023

Lời cảm ơn

Em muốn gửi lời cảm ơn tới Thầy Đào Ngọc Phong – và nhà trường đã tạo điều kiện để chúng em có môn thực hành cơ sở. Nhờ có môn học này mà em đã có thể tự thúc đẩy mình trong quá trình tiếp thu kiến thức và tự tìm hiểu các kiến thức mới. Trong quá trình tìm hiểu về đề tài Python và Data như làm báo cáo thực tập này, do trình độ và kinh nghiệm thực tiễn còn hạn chế nên không tránh khỏi những thiếu sót, em rất mong nhận được ý kiến đóng góp từ giảng viên hướng dẫn để em có thể học hỏi, rút kinh nghiệm và cải thiện trong tương lai.

Mục lục

I, Bài toán :

Chúng ta có một tập dữ liệu về giá nhà trên Kaggle gồm 21613 dòng và 21 cột dữ liệu . Tập dữ liệu này chứa giá bán nhà cho Quận King, bao gồm thành phố Seattle. Nó bao gồm các ngôi nhà được bán từ tháng 5 năm 2014 đến tháng 5 năm 2015. Và bài toán là dựa trên tập dữ liệu này ta phân tích giá nhà và chạy thử nghiệm một số mô hình để dự đoán giá nhà – xem mô hình nào là mô hình có sai số thấp nhất ?

	id	date	price	bedrooms	bathrooms	sqft_living	sqft_lot	floors	waterfront	view	...	grade	sqft_above	sqft_basement	yr_bu
0	7129300520	20141013T000000	221900.0	3	1.00	1180	5650	1.0	0	0	...	7	1180	0	19
1	6414100192	20141209T000000	538000.0	3	2.25	2570	7242	2.0	0	0	...	7	2170	400	19
2	5631500400	20150225T000000	180000.0	2	1.00	770	10000	1.0	0	0	...	6	770	0	19
3	2487200875	20141209T000000	604000.0	4	3.00	1960	5000	1.0	0	0	...	7	1050	910	19
4	1954400510	20150218T000000	510000.0	3	2.00	1680	8080	1.0	0	0	...	8	1680	0	19
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
21608	263000018	20140521T000000	360000.0	3	2.50	1530	1131	3.0	0	0	...	8	1530	0	20
21609	6600060120	20150223T000000	400000.0	4	2.50	2310	5813	2.0	0	0	...	8	2310	0	20
21610	1523300141	20140623T000000	402101.0	2	0.75	1020	1350	2.0	0	0	...	7	1020	0	20
21611	291310100	20150116T000000	400000.0	3	2.50	1600	2388	2.0	0	0	...	8	1600	0	20
21612	1523300157	20141015T000000	325000.0	2	0.75	1020	1076	2.0	0	0	...	7	1020	0	20

21613 rows × 21 columns

Mô tả các ý nghĩa các cột trong tập dữ liệu

Tên cột	**Mô tả**
Id	Id dùng để phân biệt các ngôi nhà
Date	Ngày nhà được bán
price	GIá nhà - Đây là column ta muốn dự đoán
bedrooms	Số phòng ngủ
bathrooms	Số phòng tắm
sqft_living	Diện tích sống của ngôi nhà
sqft_lot	Diện tích của khu đất
floors	Tổng số tầng trong ngôi nhà
waterfront	Nhà có tầm nhìn ra sông hoặc biển không
view	Đã được xem
condition	Tình trạng chung của ngôi nhà
grade	Mức độ tổng thể của ngôi nhà, dựa trên hệ thống xếp loại của King County
sqft_above	Diện tích của nhà ngoài tầng hầm
sqft_basement	Diện tích của tầng hầm
yr_built	Năm xây dựng
yr_renovated	Năm nhà được cải tạo
zipcode	Mã Zip
lat	Vĩ độ

long	Kinh độ
sqft_living15	Diện tích phòng khách vào năm 2015 (một số cải tạo) Điều này có thể đã ảnh hưởng hoặc không ảnh hưởng đến diện tích khu đất
sqft_lot15	Diện tích khu đất vào năm 2015 (một số cải tạo)

II, GIải pháp thực hiện :

1, Các thư viện Python sử dụng trong dự án này:
+ NumPy: là viết tắt của "Numerical Python" và cung cấp một thư viện xử lý mảng mạnh mẽ cho Python. Các mảng NumPy được sử dụng để lưu trữ các bộ số lớn và thư viện bao gồm một loạt các hàm cho các phép toán toán học trên các mảng này. Đây là một số tính năng chính của NumPy:

- Mảng hiệu quả: các mảng NumPy hiệu quả hơn nhiều so với danh sách Python thông thường để lưu trữ và xử lý lượng lớn dữ liệu số.
- Hàm toán học: NumPy bao gồm một loạt các hàm toán học cho việc làm việc với các mảng, bao gồm các hàm cho phép tính toán cơ bản, lượng giác và đại số tuyến tính.
- Broadcasting: NumPy bao gồm một tính năng gọi là "broadcasting" cho phép bạn thực hiện các phép toán số trên các mảng với các hình dạng khác nhau.
- Tạo số ngẫu nhiên: NumPy bao gồm một module để tạo ra số ngẫu nhiên, rất hữu ích cho mô phỏng và phân tích thống kê.

+ Pandas: là một thư viện phân tích dữ liệu cung cấp các công cụ để làm việc với dữ liệu có cấu trúc trong Python. Nó được xây dựng trên nền tảng NumPy và cung cấp các chức năng nâng cao hơn cho việc xử lý dữ liệu. Đây là một số tính năng chính của Pandas:

- Dataframes: Pandas cung cấp một đối tượng "DataFrame" mạnh mẽ tương tự như một bảng tính hoặc bảng SQL. DataFrames có thể được sử dụng để lưu trữ và xử lý dữ liệu có cấu trúc.
- Xử lý dữ liệu: Pandas bao gồm một loạt các hàm để xử lý dữ liệu, bao gồm các hàm để lọc, sắp xếp và nhóm dữ liệu.
- Dữ liệu bị thiếu: Pandas bao gồm các hàm để xử lý dữ liệu bị thiếu trong tập dữ liệu của bạn, chẳng hạn như điền các giá trị bị thiếu hoặc xóa các hàng có giá trị bị thiếu.
- Ghép dữ liệu: Pandas bao gồm các hàm để hợp nhất nhiều bộ dữ liệu với nhau, điều này rất hữu ích để kết hợp dữ liệu từ nhiều nguồn.

+ Matplotlib: là một thư viện vẽ đồ thị cho Python cung cấp các công cụ để tạo hình ảnh của dữ liệu của bạn. Dưới đây là một số tính năng chính của Matplotlib:

- Các hàm vẽ đồ thị: Matplotlib bao gồm một loạt các hàm để tạo ra các loại đồ thị khác nhau, bao gồm các đồ thị đường, đồ thị phân tán và đồ thị histogram.
- Tùy chỉnh: Matplotlib cung cấp một loạt các tùy chọn để tùy chỉnh các biểu đồ của bạn, chẳng hạn như thay đổi màu sắc hoặc kiểu của các đường hoặc điểm đánh dấu.
- Các bản phân đoạn (subplots): Matplotlib cho phép bạn tạo nhiều biểu đồ trong một hình sử dụng các bản phân đoạn.

+ Seaborn : là một thư viện trực quan hóa dữ liệu dựa trên Matplotlib. Nó cung cấp một giao diện cao cấp để tạo ra đồ họa thống kê hấp dẫn và thông tin. Seaborn có thể được sử dụng để trực

quan hóa phân phối đơn biến và đa biến, các mô hình hồi quy và các biến phân loại. Các tính năng chính của Seaborn bao gồm:

- Hỗ trợ cho các loại biểu đồ khác nhau như biểu đồ phân tán, biểu đồ đường, biểu đồ thanh, biểu đồ hộp, biểu đồ violin, bản đồ nhiệt và nhiều hơn nữa.
- Dễ dàng tùy chỉnh các yếu tố thẩm mỹ của biểu đồ bao gồm màu sắc, phông chữ và kích thước.
- Hỗ trợ tích hợp cho các biến phân loại và các lưới nhiều biểu đồ.
- Tích hợp với cấu trúc dữ liệu Pandas để dễ dàng xử lý và phân tích dữ liệu.
- Tích hợp với Matplotlib cho các tùy chỉnh và tùy chọn vẽ biểu đồ nâng cao hơn.

+ scikit-learn : là một thư viện học máy cho Python cung cấp một loạt các công cụ để xây dựng các mô hình dự đoán. Dưới đây là một số tính năng chính của scikit-learn:

- Các thuật toán học máy: scikit-learn bao gồm một loạt các thuật toán học máy, bao gồm các thuật toán hồi quy, phân loại và gom cụm.
- Các chức năng tiền xử lý: scikit-learn bao gồm các chức năng để tiền xử lý dữ liệu của bạn trước khi xây dựng một mô hình, chẳng hạn như tỷ lệ hoặc chuẩn hóa dữ liệu của bạn.
- Đánh giá mô hình: scikit-learn bao gồm các chức năng để đánh giá hiệu suất của mô hình của bạn, chẳng hạn như tính độ chính xác hoặc tính ma trận nhầm lẫn.
- Các đường ống (pipelines): scikit-learn bao gồm một tính năng gọi là "đường ống" cho phép bạn kết nối các bước tiền xử lý và một mô hình thành một đối tượng duy nhất.

+ Folium : là một thư viện Python để tạo bản đồ tương tác bằng cách sử dụng Leaflet.js. Nó cho phép bạn trực quan hóa dữ liệu địa lý trên bản đồ tương tác với đánh dấu, cửa sổ bật lên, bản đồ choropleth và nhiều hơn nữa. Folium được xây dựng trên các thư viện Python như Pandas, Numpy và Matplotlib và có thể được sử dụng để tạo bản đồ tương tác từ nhiều nguồn dữ liệu bao gồm các tệp CSV, tệp GeoJSON và Pandas DataFrames. Các tính năng chính của Folium bao gồm:

- API Python dễ sử dụng để tạo bản đồ tương tác.
- Hỗ trợ cho các loại bản đồ khác nhau bao gồm bản đồ tile, bản đồ choropleth và bản đồ nhiệt.
- Tích hợp với Pandas DataFrames để dễ dàng xử lý và phân tích dữ liệu.
- Hỗ trợ cho các biểu tượng và đánh dấu tùy chỉnh.
- Hỗ trợ tích hợp cho các cửa sổ bật lên và thông tin chú thích.
- Khả năng xuất bản đồ dưới dạng tệp HTML để dễ dàng

2, Các bước thực hiện :

+ Import các thư viện cần thiết và dữ liệu vào

+ Visualize bằng các biểu đồ và xem sự tương quan giữa các cột với nhau

+ Tiền xử lý dữ liệu bằng Bining

+ Thử nghiệm các mô hình như Linear Regression (Simple – Multiple), Rigde Regression, Lasso Regression, Polynomial Regression để so sánh về sai số của các mô hình!

3, Demo phần phân tích :

Lưu ý : Phần này trình hơi khó nhìn vì trình bày cả hình lẫn chữ, do các công thức toán học khá khó viết trên Word nên em chụp từ phần Markdown bên Jupiter Notebook sang luôn. Mong thầy thông cảm vì sự bất tiện này ạ .

1, Importing Modules, Reading the Dataset and Defining an Evaluation Table

+ Để thực hiện một số phân tích, chúng ta cần thiết lập môi trường làm việc của mình bằng cách Import các thư viện cần thiết. Bảng dữ liệu này bao gồm **Root Mean Squared Error (RMSE là thước đo độ lệch trung bình giữa các giá trị dự đoán và giá trị thực tế trong tập dữ liệu), R-squared (R-squared là một đại lượng thống kê đo độ lệch của dữ liệu với đường hồi quy được sử dụng để khớp với dữ liệu), Adjusted R-squared** và trung bình các giá trị **R-squared** được đạt được bằng phương pháp **Kiểm định chéo k-Fold**, đây là các chỉ số quan trọng để so sánh các mô hình khác nhau. Một giá trị **R-squared** gần với một và RMSE càng nhỏ thì sai số càng ít

+ Phương pháp kiểm định đường chéo K-fold là một phương pháp phân chia tập dữ liệu thành k phần bằng nhau và sử dụng một phần để đánh giá mô hình và các phần còn lại để huấn luyện mô hình. Quá trình này được lặp lại k lần với các phần dữ liệu khác nhau được sử dụng để đánh giá và huấn luyện mô hình. Kết quả cuối cùng là trung bình của các kết quả được tính toán. Phương pháp này giúp đánh giá mô hình một cách khách quan và giảm thiểu tác động của phân chia dữ liệu ban đầu lên kết quả đánh giá.

```
#
evaluation = pd.DataFrame({'Model': [],
                           'Details':[],
                           'Root Mean Squared Error (RMSE)':[], # Độ lệch chuẩn của các điểm dữ liệu so với đường hồi quy
                           # RMSE là thước đo độ lệch trung bình giữa các giá trị dự đoán và giá trị thực tế trong tập dữ liệu.
                           'R-squared (training)':[], # Độ chính xác của mô hình trên tập huấn luyện
                           'Adjusted R-squared (training)':[], # R-squared is a statistical measure of how close the data are to the fitted regressi
                           'R-squared (test)':[],
                           'Adjusted R-squared (test)':[],
                           '5-Fold Cross Validation':[]})

df = pd.read_csv('D:\Desktop\Data Analyst\Python\TTCS\kc_house_data copy.csv') # Đọc dữ liệu từ file csv

df # Show all dataframe
```

Out[1]:

	id	date	price	bedrooms	bathrooms	sqft_living	sqft_lot	floors	waterfront	view	...	grade	sqft_above	sqft_basement	yr_bu
0	7129300520	20141013T000000	221900.0	3	1.00	1180	5650	1.0	0	0	...	7	1180	0	19
1	6414100192	20141209T000000	538000.0	3	2.25	2570	7242	2.0	0	0	...	7	2170	400	19
2	5631500400	20150225T000000	180000.0	2	1.00	770	10000	1.0	0	0	...	6	770	0	19
3	2487200875	20141209T000000	604000.0	4	3.00	1960	5000	1.0	0	0	...	7	1050	910	19
4	1954400510	20150218T000000	510000.0	3	2.00	1680	8080	1.0	0	0	...	8	1680	0	19
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
21608	263000018	20140521T000000	360000.0	3	2.50	1530	1131	3.0	0	0	...	8	1530	0	20
21609	6600060120	20150223T000000	400000.0	4	2.50	2310	5813	2.0	0	0	...	8	2310	0	20
21610	1523300141	20140623T000000	402101.0	2	0.75	1020	1350	2.0	0	0	...	7	1020	0	20
21611	291310100	20150116T000000	400000.0	3	2.50	1600	2388	2.0	0	0	...	8	1600	0	20
21612	1523300157	20141015T000000	325000.0	2	0.75	1020	1076	2.0	0	0	...	7	1020	0	20

21613 rows × 21 columns

Defining a Function to Calculate the Adjusted R^2

Hệ số xác định R-squared tăng khi số lượng đặc trưng tăng. Vì vậy, em chọn adjusted R-squared và nó chỉ tăng khi việc thêm biến làm giảm Sai số bình phương trung bình (MSE). Công thức của adjusted R^2 là:

$\bar{R}^2 = R^2 - \frac{k-1}{n-k}(1 - R^2)$

Trong đó n là số quan sát và k là tham số

```
def adjustedR2(r2,n,k):
    return r2-(k-1)/(n-k)*(1-r2)
```

2, Creating a Simple Linear Regression

Khi chúng ta xây dựng một mối quan hệ tuyến tính giữa biến phụ thuộc và chỉ một biến giải thích, điều này được gọi là **Simple Linear Regression**. Muốn dự đoán giá nhà và sau đó, biến phụ thuộc của chúng ta là giá. Tuy nhiên, đối với một mô hình đơn giản, chúng ta cũng cần chọn một đặc trưng (feature). Khi chúng ta xem xét ma trận tương quan, chúng ta có thể quan sát thấy rằng giá có hệ số tương quan cao nhất với diện tích sống (sqft) nên sẽ chọn nó làm biến đặc trưng. **Tóm lại mô hình này dùng biến 'Price' là biến phụ thuộc, 'sqft_living'là biến giải thích - biến đặc trưng**

```
Average Price for Test Data: 539744.130
Intercept: -47235.811302901246
Coefficient: [282.2468152]
```

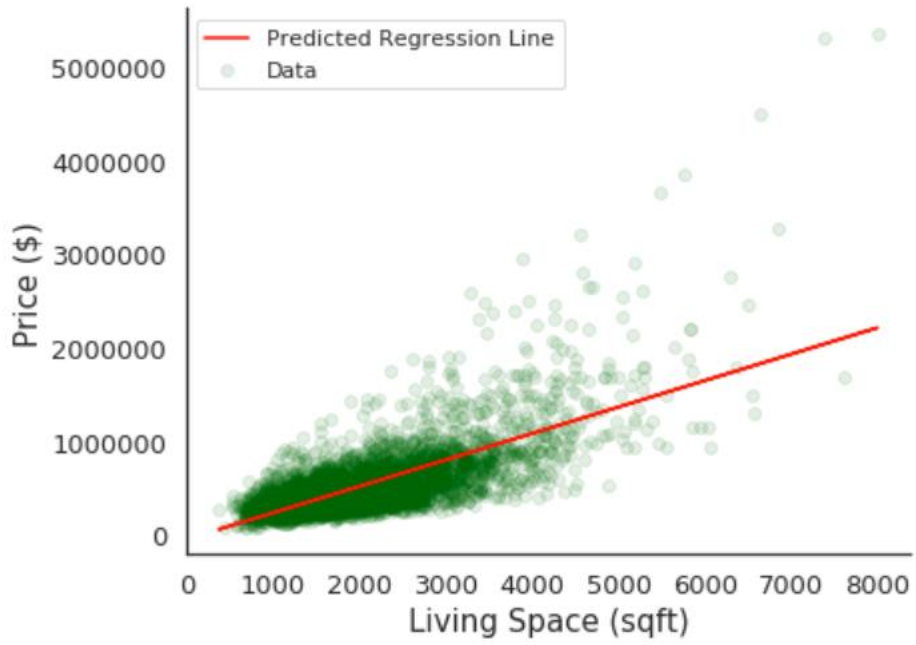
	Model	Details	Root Mean Squared Error (RMSE)	R-squared (training)	Adjusted R-squared (training)	R-squared (test)	Adjusted R-squared (test)	5-Fold Cross Validation
0	Simple Linear Regression	-	254289.149	0.492	-	0.496	-	0.491

Em cũng in ra hệ số góc (Intercept) và hệ số tương quan(Coefficient) cho hồi quy tuyến tính đơn giản. Bằng cách sử dụng những giá trị này và định nghĩa bên dưới, chúng ta có thể ước tính giá nhà thủ công. Phương trình chúng ta sử dụng để ước tính được gọi là hàm hypothesis và được định nghĩa như sau:

$$h_\theta(X) = \theta_0 + \theta_1 x$$

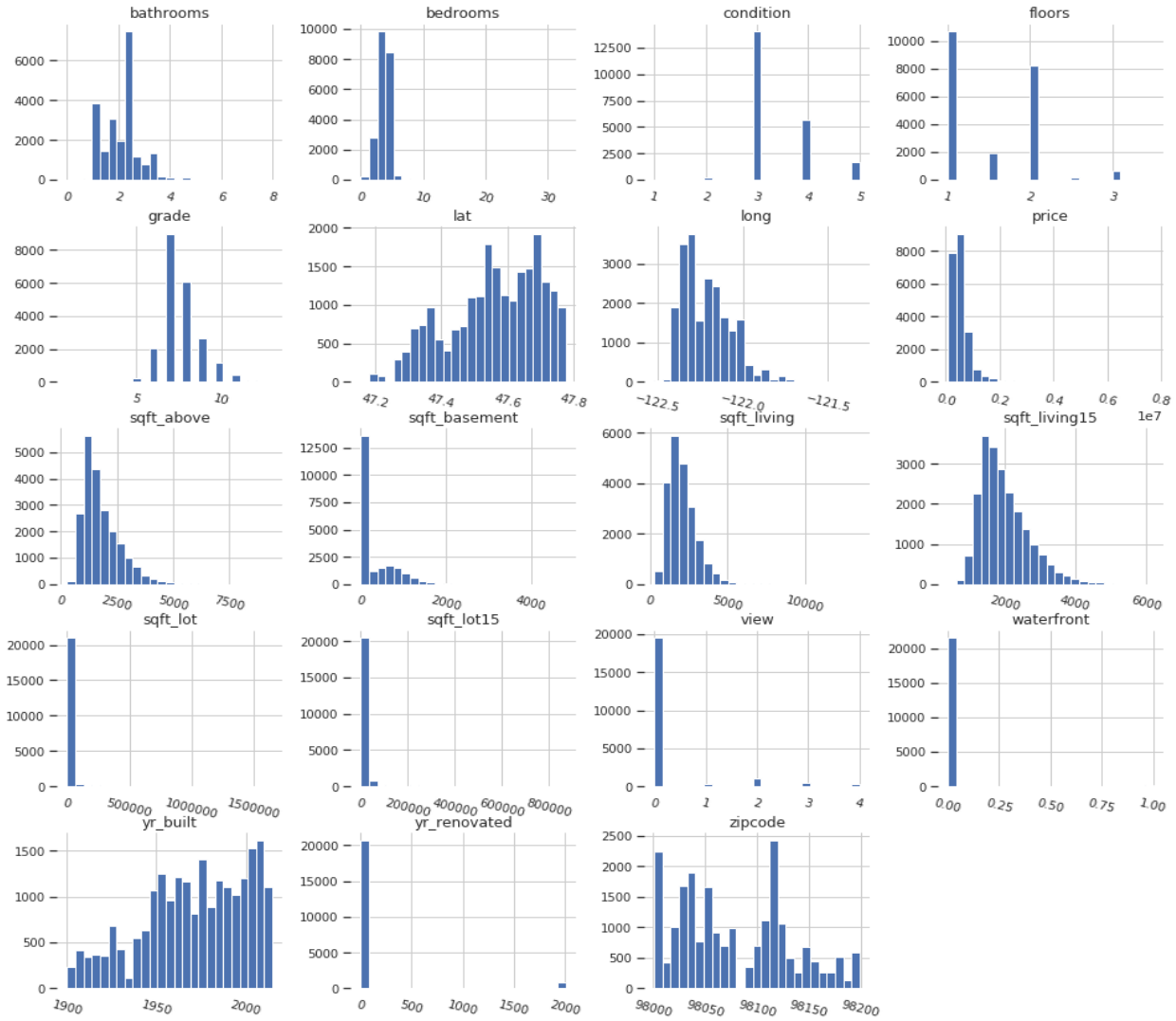
Let's Show the Result

Vì chúng ta chỉ có **hai chiều** trong mô hình hồi quy đơn giản, nên việc vẽ biểu đồ rất đơn giản. Biểu đồ dưới đây cho thấy kết quả của hồi quy đơn giản. Nó không trông hoàn hảo nhưng khi làm việc với các tập dữ liệu thực tế, việc có một sự phù hợp hoàn hảo không dễ dàng.



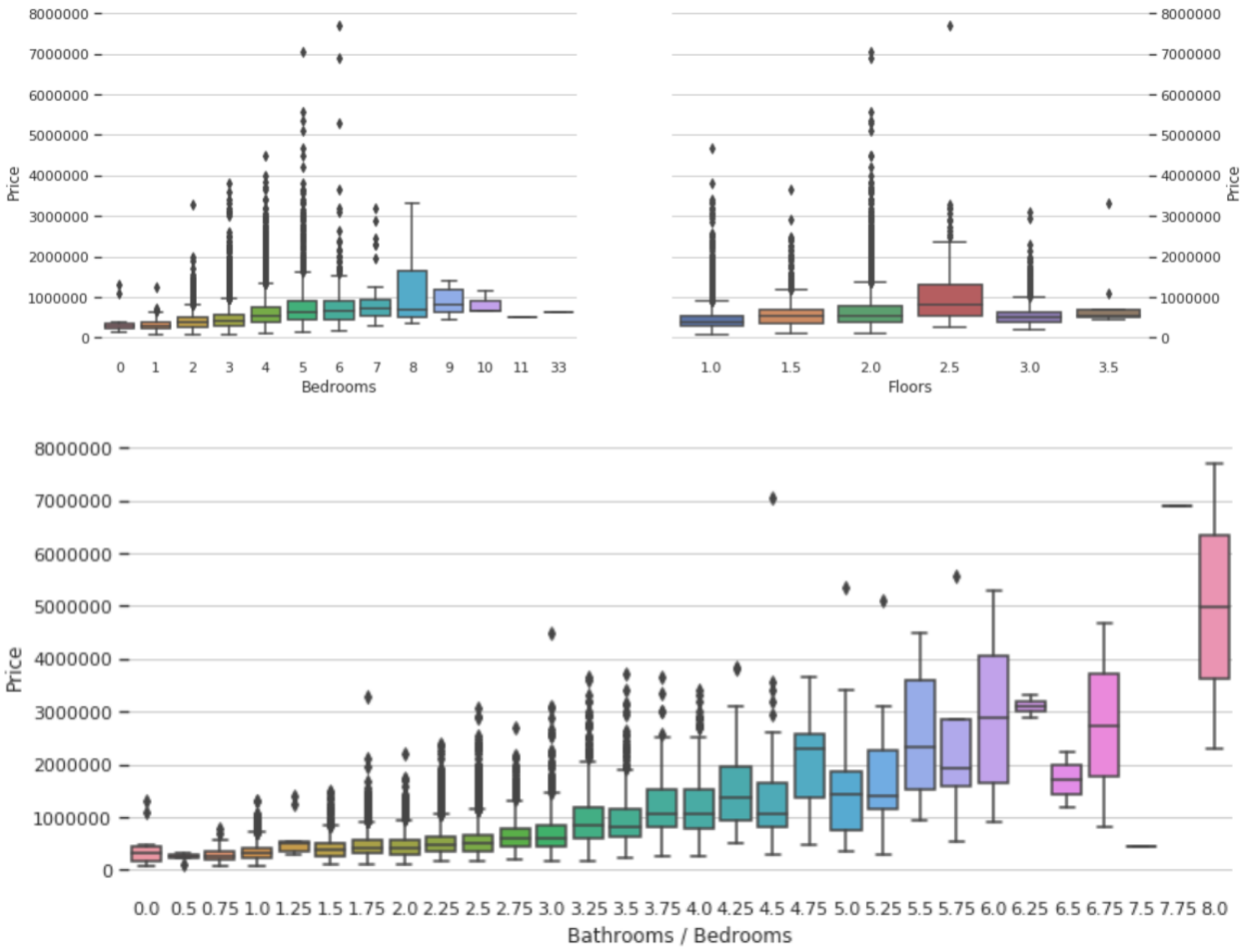
3, Visualizing and Examining Data

Đây không phải là dữ liệu quá lớn và chúng ta không có quá nhiều đặc trưng (feature). Do đó, chúng ta có cơ hội để vẽ đồ thị cho hầu hết chúng và đạt được một số kết quả phân tích hữu ích. Việc vẽ biểu đồ và kiểm tra dữ liệu trước khi áp dụng một mô hình là một thực hành tốt vì chúng ta có thể phát hiện ra một số điểm ngoại lai có thể xảy ra hoặc quyết định thực hiện chuẩn hóa dữ liệu. Điều này không bắt buộc nhưng hiểu về dữ liệu luôn là điều tốt. Sau đó, em bắt đầu với các biểu đồ histogram của khung dữ liệu. Dưới đây là biểu đồ thể hiện số lượng mẫu dữ liệu theo từng Feature – trục x là số lượng của feature đó (VD :) , Y là số lượng mẫu dữ liệu (hàng) tương ứng



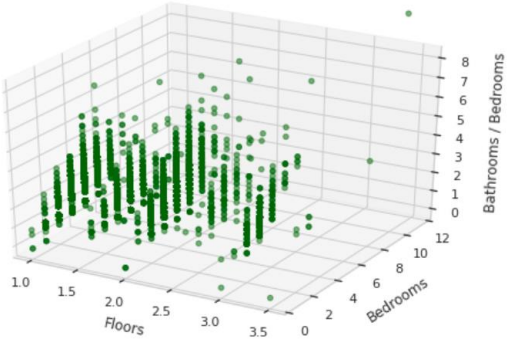
Để xác định số phòng ngủ, số tầng hoặc số phòng tắm/phòng ngủ so với giá, chọn **biểu đồ hộp** vì chúng ta có dữ liệu số nhưng chúng không liên tục như 1,2, ... phòng ngủ, 2.5, 3, ... tầng (có thể 0.5 đại diện cho phòng penthouse).

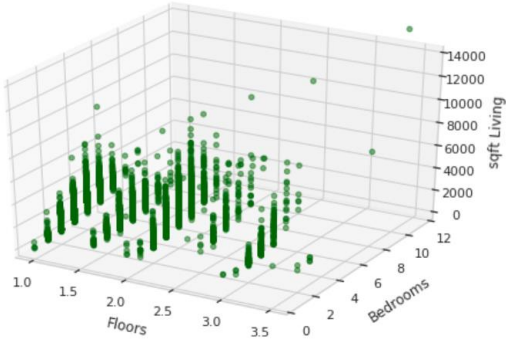
Từ các biểu đồ dưới đây, có thể thấy rằng có rất ít nhà có một số đặc điểm hoặc giá xuất hiện xa hơn so với các giá trị khác như 33 phòng ngủ hoặc giá khoảng 7000000. Tuy nhiên, xác định được tác động tiêu cực có thể mất thời gian và trong các bộ dữ liệu thực tế, luôn có một số ngoại lệ như một số giá nhà sang trọng trong bộ dữ liệu này. Đó là lý do tại sao không có kế hoạch loại bỏ các ngoại lệ.

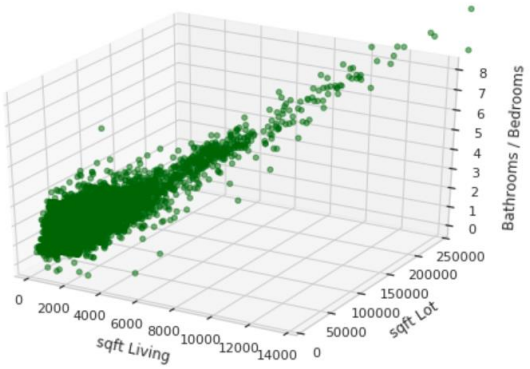


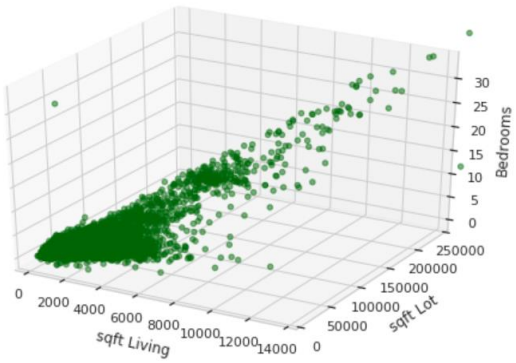
Vẽ biểu đồ giá trị của một số đặc điểm so với giá nhà và có vẻ như không có mối quan hệ tuyến tính hoàn hảo giữa giá và những đặc điểm đó. Trong khi đó, mối quan hệ giữa chúng thì sao? Để thể hiện điều này, sử dụng biểu đồ 3D. Sử dụng màu xanh nhạt làm màu cho điểm. Các phần xanh đậm có nghĩa là mật độ cao, nhiều điểm xanh nhạt chồng lên nhau và trở nên tối hơn.

Các biểu đồ dưới đây cho thấy rằng khi sqrt_living tăng, sqrt_lot và số phòng ngủ hoặc số phòng tắm/phòng ngủ cũng tăng. Tuy nhiên, số tầng, số phòng ngủ và số phòng tắm/phòng ngủ hoặc sqrt_living lại không có mối quan hệ tương tự nhau.

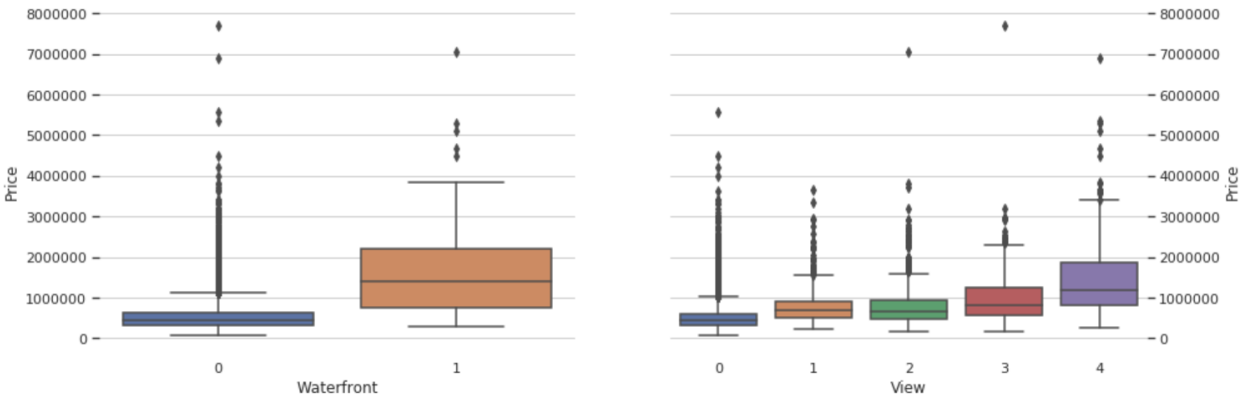


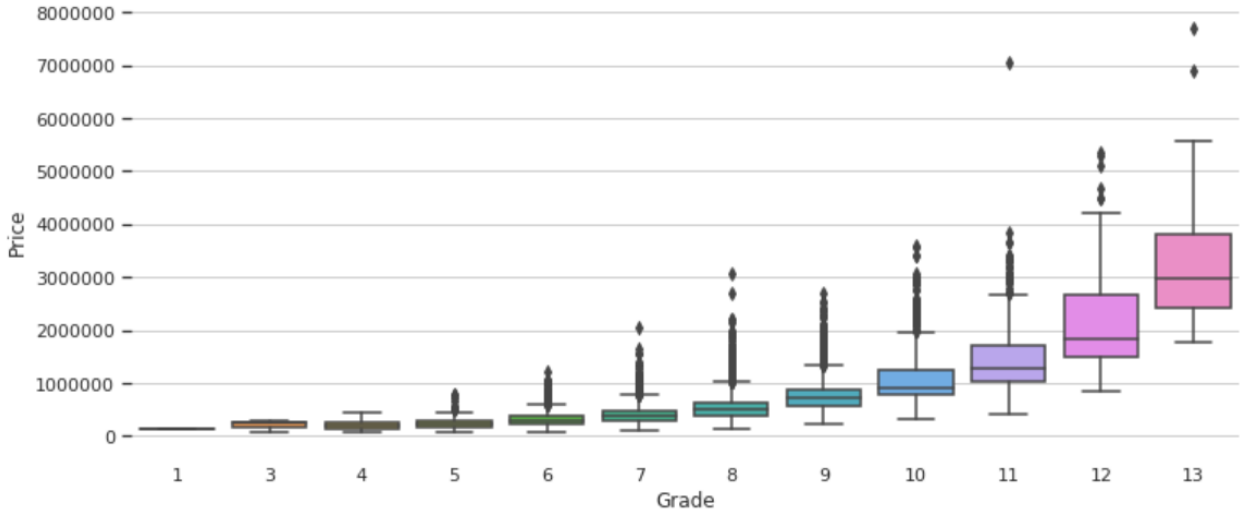




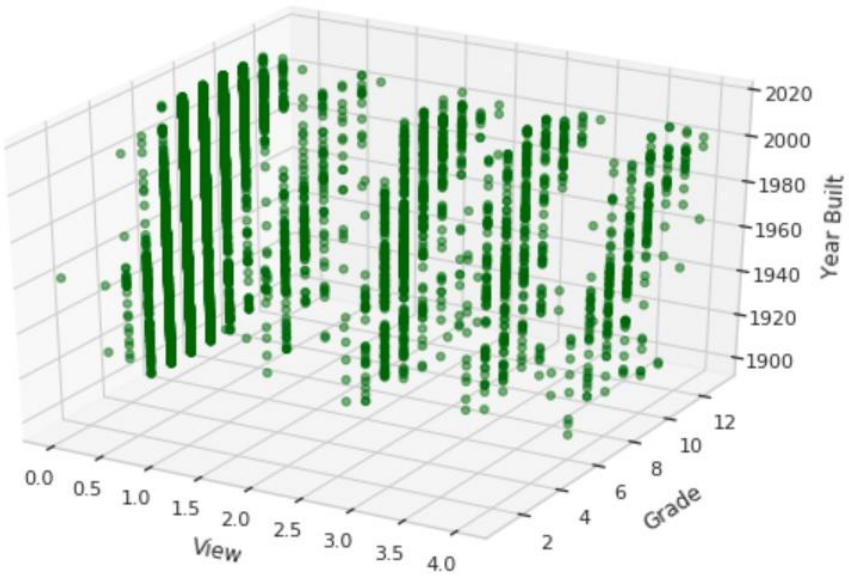


Khi chúng ta nhìn vào các biểu đồ hộp dưới đây, độ lớn của ngôi nhà và bãi biển ảnh hưởng đến giá một cách rõ rệt. Trong khi đó, quan điểm về cảnh quan có vẻ ít ảnh hưởng, nhưng nó cũng có ảnh hưởng đến giá.

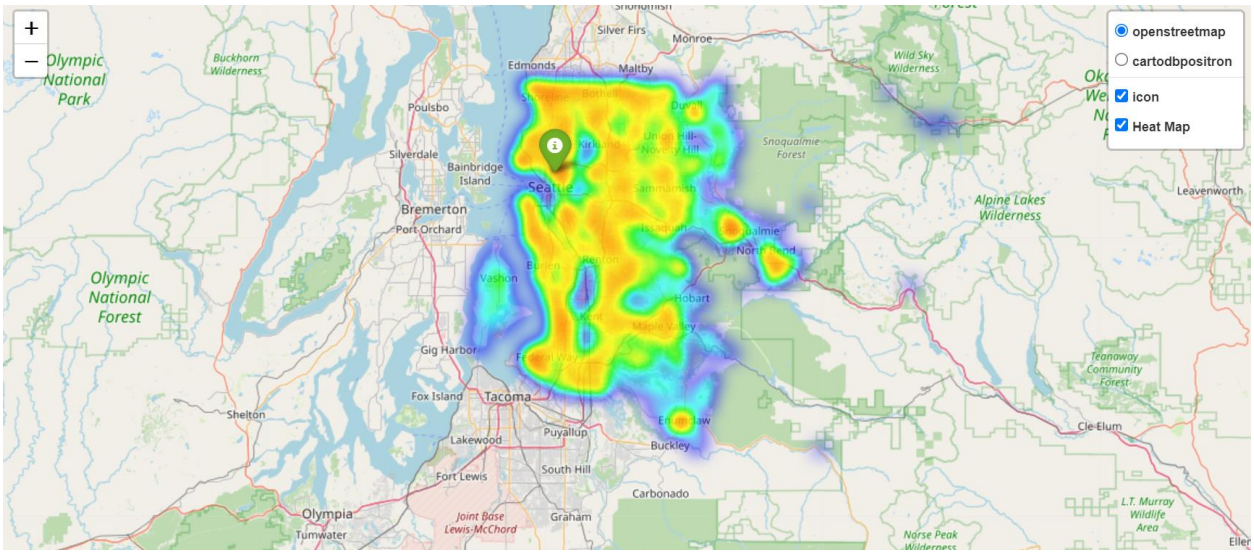




Vẽ biểu đồ 3D để xác định mối quan hệ giữa view, grade và năm xây dựng(yr_built). Biểu đồ dưới đây cho thấy những ngôi nhà mới hơn có mức độ đánh giá tốt hơn nhưng chúng ta không thể nói nhiều về sự thay đổi của view.



Trong bộ dữ liệu này, chúng ta có thông tin về vĩ độ và kinh độ của các ngôi nhà. Bằng cách sử dụng cột lat và long, đã hiển thị bản đồ nhiệt dưới đây, rất hữu ích cho những người không quen biết với Seattle. Ngoài ra, nếu bạn chọn một mã bưu chính cụ thể, bạn có thể chỉ xem bản đồ nhiệt của khu phố thuộc mã bưu chính này.

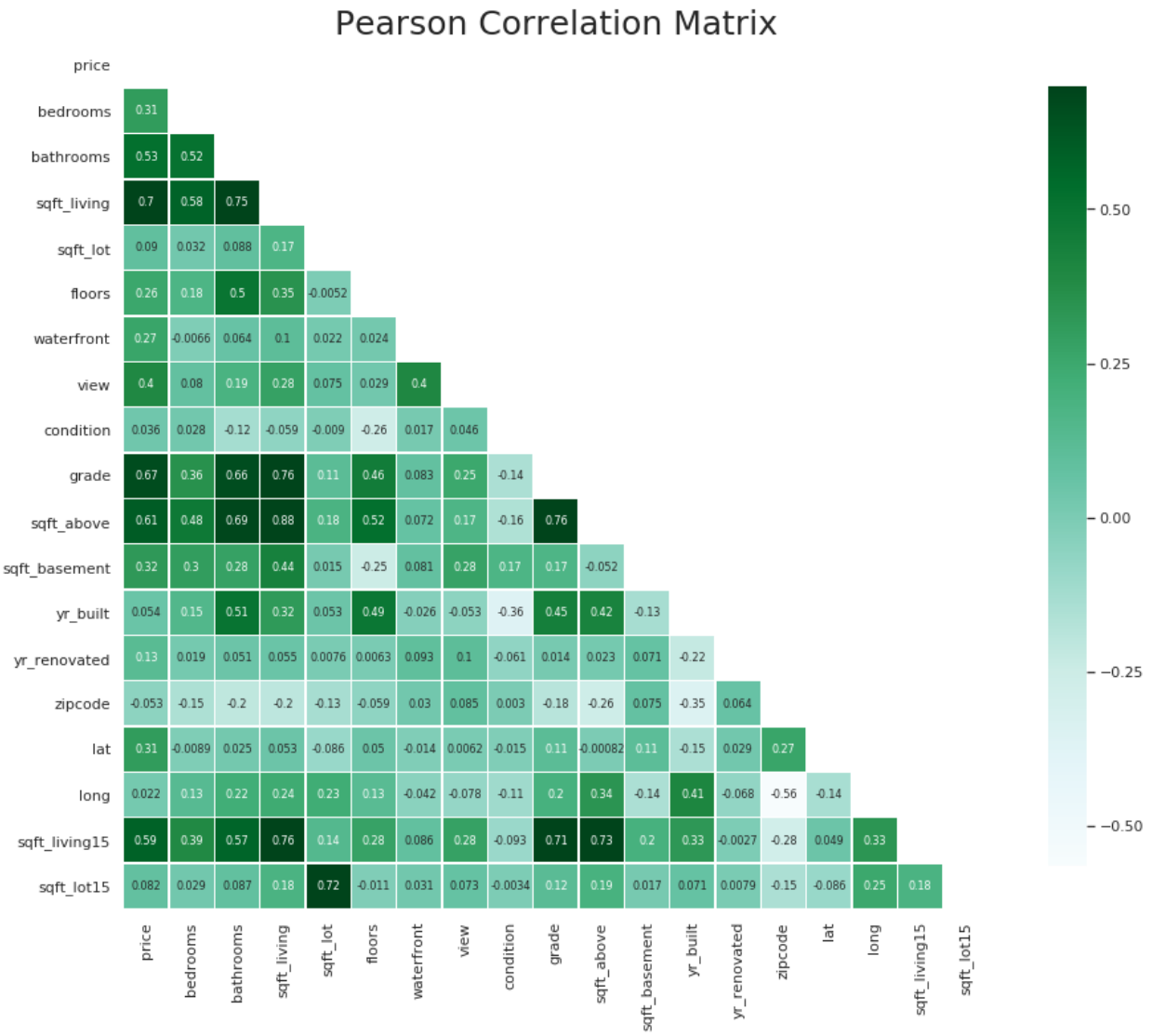


Checking Out the Correlation Among Explanatory Variables

Có quá nhiều đặc trưng trong một mô hình không phải lúc nào cũng tốt vì nó có thể gây ra tình trạng overfitting và kết quả tồi hơn khi chúng ta muốn dự đoán giá trị cho một bộ dữ liệu mới. Do đó, **nếu một đặc trưng không cải thiện mô hình một cách đáng kể, việc không thêm nó có thể là một lựa chọn tốt hơn.**

Một điều quan trọng khác là tương quan. **Nếu có mức độ tương quan rất cao giữa hai đặc trưng, giữ cả hai đặc trưng không phải lúc nào cũng là một ý tưởng tốt, để không gây ra**

tình trạng overfitting. Ví dụ, nếu có tình trạng overfitting, chúng ta có thể loại bỏ sqt_above hoặc sqt_living vì chúng có mối quan hệ cao. Mối quan hệ này có thể được ước tính khi chúng ta xem các định nghĩa trong bộ dữ liệu, nhưng để chắc chắn, ma trận tương quan nên được kiểm tra. Tuy nhiên, điều này không có nghĩa là bạn phải loại bỏ một trong những đặc trưng có mức độ tương quan cao. Ví dụ: bathrooms và sqrt_living. Chúng có mức độ tương quan cao nhưng không nghĩ rằng mối quan hệ giữa chúng giống như mối quan hệ giữa sqt_living và sqt_above.



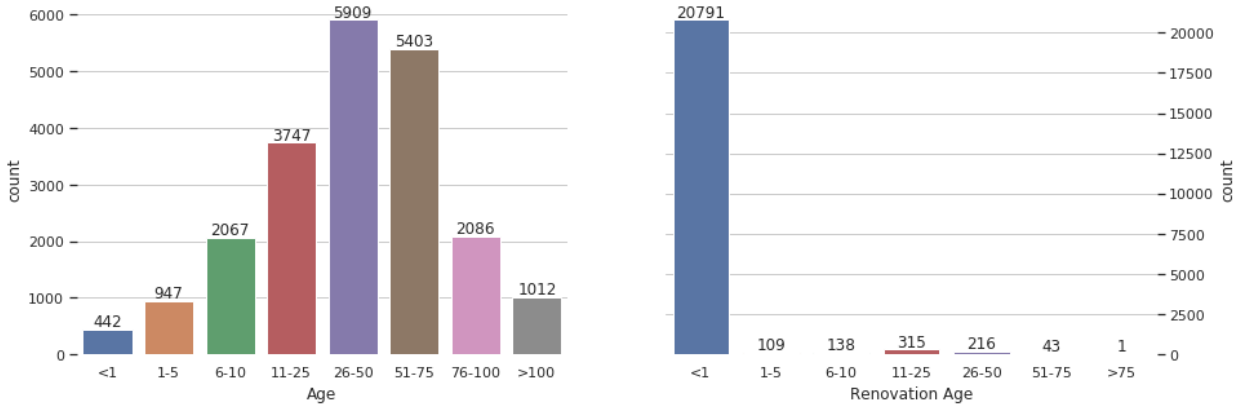
4, Data Preprocessing

Việc tiền xử lý dữ liệu có thể cải thiện độ chính xác của mô hình và làm cho mô hình đáng tin cậy hơn. Điều này không phải lúc nào cũng cải thiện kết quả của chúng ta, nhưng khi chúng ta nhận thức được các tính năng và sử dụng đầu vào thích hợp, chúng ta có thể dễ dàng đạt được một số kết quả. Sử dụng binning và tạo một khung dữ liệu mới gọi là **df_dm.**

Binning dữ liệu (hay còn gọi là discrete binning) là một kỹ thuật tiền xử lý dữ liệu được sử dụng để chia dữ liệu thành các khoảng (bins) rời rạc dựa trên giá trị của một biến liên tục. Kỹ thuật này giúp giảm độ phức tạp của dữ liệu bằng cách tạo ra các nhóm (bins) có ý nghĩa hơn, từ đó giảm thiểu ảnh hưởng của các sai số nhỏ và tăng tính tổng quát của dữ liệu.

Khi áp dụng binning, chúng ta chia đồng thời phạm vi giá trị của biến liên tục thành các khoảng (bins) rời rạc. Các giá trị trong cùng một khoảng sẽ được gom nhóm lại thành một nhóm duy nhất và được đại diện bằng một giá trị gốc. (Hoặc hiểu Binning theo kiểu Group By trong SQL)

Binning dữ liệu là một kỹ thuật tiền xử lý (Preprocessing) được sử dụng để giảm thiểu ảnh hưởng của các sai số quan sát nhỏ. Nó đáng áp dụng cho một số cột trong bộ dữ liệu này. Áp dụng binning cho yr_built và yr_renovated. Thêm tuổi(age) và tuổi cải tạo (Renovation Age) của các căn nhà khi bán. Ngoài ra, Phân vùng các cột này thành các khoảng và ta có thể quan sát điều này trong các biểu đồ **histogram** bên dưới.



5, Multiple Regression

Mô hình hồi quy tuyến tính đơn giản cho thấy thấy kết quả rất kém. Để cải thiện mô hình này, em dự định thêm nhiều tính năng hơn. Khi chúng ta có **nhiều hơn một feature** trong một mô hình hồi quy tuyến tính, nó được định nghĩa là **hồi quy đa biến(Multiple Regression)**. Sau đó, đến lúc tạo ra một số mô hình phức tạp.

Multiple Regression - 1

Xác định **features** ngay từ cái nhìn đầu tiên bằng cách xem các phần trước và sử dụng chúng trong mô hình hồi quy tuyến tính đa biến đầu tiên. Giống như trong hồi quy đơn giản, in các hệ số mà mô hình sử dụng cho các dự đoán. Tuy nhiên, lần này chúng ta phải sử dụng định nghĩa dưới đây cho các dự đoán của chúng ta nếu chúng ta muốn tính toán thủ công:

$$h_\theta(X) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \ldots + \theta_n x_n$$

```
features = ['bedrooms','bathrooms','sqft_living','sqft_lot','floors','zipcode']
```

Sử dụng 6 feature này vì chúng có độ tương quan cao nhất với Price, sử dụng zipcode vì nó là một biến phân loại

Output: Đã sai số ít hơn so với simple regression

	Model	Details	Root Mean Squared Error (RMSE)	R-squared (training)	Adjusted R-squared (training)	R-squared (test)	Adjusted R-squared (test)	5-Fold Cross Validation
1	Multiple Regression-1	selected features	248514.011	0.514	0.514	0.519	0.518	0.512
0	Simple Linear Regression	-	254289.149	0.492	-	0.496	-	0.491

Multiple Regression - 2

Thêm nhiều **feature** vào danh sách feature. Khi chúng ta đánh giá thì đã cải thiện đáng kể sai số.

```
features = ['bedrooms','bathrooms','sqft_living','sqft_lot','floors','waterfront','view',
            'grade','age_binned_<1', 'age_binned_1-5', 'age_binned_6-10','age_binned_11-25',
            'age_binned_26-50', 'age_binned_51-75','age_binned_76-100', 'age_binned_>100',
            'zipcode']
```

Output: Đã sai số ít hơn nữa

	Model	Details	Root Mean Squared Error (RMSE)	R-squared (training)	Adjusted R-squared (training)	R-squared (test)	Adjusted R-squared (test)	5-Fold Cross Validation
2	Multiple Regression-2	selected features	209712.753	0.652	0.652	0.657	0.656	0.648
1	Multiple Regression-1	selected features	248514.011	0.514	0.514	0.519	0.518	0.512
0	Simple Linear Regression	-	254289.149	0.492	-	0.496	-	0.491

Multiple Regression - 3

Tạo một mô hình với **tất cả các đặc trưng mà không có bất kỳ tiền xử lý nào**. Các chỉ số đánh giá sai số đã cải thiện đáng kể.

```
features = ['bedrooms','bathrooms','sqft_living','sqft_lot','floors','waterfront','view',
            'condition','grade','sqft_above','sqft_basement','yr_built','yr_renovated',
            'zipcode','lat','long','sqft_living15','sqft_lot15']
```

Output: Đã sai số ít hơn

	Model	Details	Root Mean Squared Error (RMSE)	R-squared (training)	Adjusted R-squared (training)	R-squared (test)	Adjusted R-squared (test)	5-Fold Cross Validation
3	Multiple Regression-3	all features, no preprocessing	193693.989	0.698	0.697	0.708	0.707	0.695
2	Multiple Regression-2	selected features	209712.753	0.652	0.652	0.657	0.656	0.648
1	Multiple Regression-1	selected features	248514.011	0.514	0.514	0.519	0.518	0.512
0	Simple Linear Regression	-	254289.149	0.492	-	0.496	-	0.491

Multiple Regression - 4

Lần này sử dụng dữ liệu được thu thập sau bước tiền xử lý preprocessing

```
# create a list of features
features = ['bedrooms','bathrooms','sqft_living','sqft_lot','floors','waterfront',
            'view','condition','grade','sqft_above','sqft_basement','age_binned_<1',
            'age_binned_1-5', 'age_binned_6-10','age_binned_11-25', 'age_binned_26-50',
            'age_binned_51-75','age_binned_76-100', 'age_binned_>100','age_rnv_binned_<1',
            'age_rnv_binned_1-5', 'age_rnv_binned_6-10', 'age_rnv_binned_11-25',
            'age_rnv_binned_26-50', 'age_rnv_binned_51-75', 'age_rnv_binned_>75',
            'zipcode','lat','long','sqft_living15','sqft_lot15']
```

Output : Đã sai số ít hơn- đây là phương pháp linear regression có sai số thấp nhất: RMSE thấp, đường chéo fold cao

	Model	Details	Root Mean Squared Error (RMSE)	R-squared (training)	Adjusted R-squared (training)	R-squared (test)	Adjusted R-squared (test)	5-Fold Cross Validation
4	Multiple Regression-4	all features	191879.550	0.701	0.7	0.713	0.711	0.698
3	Multiple Regression-3	all features, no preprocessing	193693.989	0.698	0.697	0.708	0.707	0.695
2	Multiple Regression-2	selected features	209712.753	0.652	0.652	0.657	0.656	0.648
1	Multiple Regression-1	selected features	248514.011	0.514	0.514	0.519	0.518	0.512
0	Simple Linear Regression	-	254289.149	0.492	-	0.496	-	0.491

Regularization

Regularization là một mô hình machine learning được thiết kế để giải quyết vấn đề quá khớp (overfitting) và thiếu khớp (underfitting). Quá khớp (overfitting) có nghĩa là phương sai cao và thường được gây ra bởi một hàm phức tạp tạo ra nhiều đường cong và góc không cần thiết không liên quan đến dữ liệu. Hàm này phù hợp với dữ liệu huấn luyện tốt nhưng có thể gây ra kết quả kém cho tập kiểm tra. Trong khi đó, **thiếu khớp (underfitting)** có nghĩa là phương sai thấp và mô hình rất đơn giản. Điều này cũng có thể gây ra kết quả kém. Một số biện pháp khắc phục có thể là điều chỉnh các tính năng thủ công hoặc sử dụng một số thuật toán lựa chọn mô hình để mang lại khối lượng công việc bổ sung. Tuy nhiên, khi chúng ta áp dụng Regularization, tất cả các tính năng được giữ nguyên và mô hình điều chỉnh θj. Điều này đặc biệt hữu ích khi chúng ta có rất nhiều tính năng hơi hữu ích. Có hai loại Regularization phổ biến (Ridge và Lasso Regressions) và trong phần này, em đã sử dụng chúng.

Khi nào nên sử dụng hồi quy Ridge và hồi quy Lasso:

+ Nhiều hiệu ứng nhỏ/trung bình: sử dụng hồi quy Ridge.

+ Chỉ một vài biến số với hiệu ứng trung bình/lớn: sử dụng hồi quy Lasso.

Ridge Regression

Hồi quy Ridge được gọi là **L2 regularization** để kiểm soát sự phức tạp của mô hình. Nó giảm giá trị của các hệ số (weights) trong mô hình, đồng thời duy trì tất cả các biến số trong mô hình. . Chúng ta có phương trình sau đây.

$$RSS_{RIDGE} = \sum_{i=1}^{m}(h_\theta(x_i) - y_i)^2 + \alpha \sum_{j=1}^{n} \theta_j^2$$

Bằng cách thay đổi giá trị α , chúng ta có thể kiểm soát lượng regularization. Khi tăng giá trị của α , regularization tăng và ngược lại. Vì vậy chọn các giá trị α khác nhau và sử dụng một phương trình hồi quy tuyến tính không có regularization để quan sát sự khác biệt một cách dễ dàng.

Output:

Model	Details	Root Mean Squared Error (RMSE)	R-squared (training)	Adjusted R-squared (training)	R-squared (test)	Adjusted R-squared (test)	5-Fold Cross Validation
Multiple Regression-4	all features	191879.550	0.701	0.7	0.713	0.711	0.698
Ridge Regression	alpha=1, all features	191903.548	0.701	0.7	0.713	0.711	0.698
Multiple Regression-3	all features, no preprocessing	193693.989	0.698	0.697	0.708	0.707	0.695
Ridge Regression	alpha=100, all features	195372.495	0.694	0.693	0.703	0.701	0.691
Multiple Regression-2	selected features	209712.753	0.652	0.652	0.657	0.656	0.648
Ridge Regression	alpha=1000, all features	209625.468	0.651	0.65	0.658	0.655	0.648
Multiple Regression-1	selected features	248514.011	0.514	0.514	0.519	0.518	0.512
Simple Linear Regression	-	254289.149	0.492	-	0.496	-	0.491

Lasso Regression

Lasso regression gọi là **L1 regularization** để kiểm soát sự phức tạp của mô hình. Hồi quy Lasso có khả năng thực hiện lựa chọn biến tự động bằng cách đặt một số hệ số của các biến thành 0. Điều này giúp loại bỏ các biến không quan trọng hoặc có ảnh hưởng nhỏ trong mô hình. Và công thức là :

$$RSS_{LASSO} = \sum_{i=1}^{m}(h_\theta(x_i) - y_i)^2 + \alpha \sum_{j=1}^{n} |\theta_j|$$

Sự khác nhau giữa ridge and lasso là vế thứ 2 của biểu thức

Output:

	Model	Details	Root Mean Squared Error (RMSE)	R-squared (training)	Adjusted R-squared (training)	R-squared (test)	Adjusted R-squared (test)	5-Fold Cross Validation
4	Multiple Regression-4	all features	191879.550	0.701	0.7	0.713	0.711	0.698
5	Ridge Regression	alpha=1, all features	191903.548	0.701	0.7	0.713	0.711	0.698
8	Lasso Regression	alpha=1, all features	191880.918	0.701	0.7	0.713	0.711	0.698
9	Lasso Regression	alpha=100, all features	192060.144	0.701	0.7	0.713	0.711	0.698
3	Multiple Regression-3	all features, no preprocessing	193693.989	0.698	0.697	0.708	0.707	0.695
10	Lasso Regression	alpha=1000, all features	193587.943	0.697	0.697	0.708	0.706	0.695
6	Ridge Regression	alpha=100, all features	195372.495	0.694	0.693	0.703	0.701	0.691
2	Multiple Regression-2	selected features	209712.753	0.652	0.652	0.657	0.656	0.648
7	Ridge Regression	alpha=1000, all features	209625.468	0.651	0.65	0.658	0.655	0.648
1	Multiple Regression-1	selected features	248514.011	0.514	0.514	0.519	0.518	0.512
0	Simple Linear Regression	-	254289.149	0.492	-	0.496	-	0.491

Polynomial Regression

Đối với các mô hình tuyến tính (linear model), ý tưởng chính là phù hợp với một đường thẳng với dữ liệu của chúng ta. Tuy nhiên, nếu dữ liệu có phân bố bậc hai, lần này chọn một hàm bậc hai và áp dụng biến đổi đa thức có thể mang lại kết quả tốt hơn. Lần này hàm (hypothesis) được định nghĩa như sau:

$$h_\theta(X) = \theta_0 + \theta_1 x + \theta_2 x^2 + \ldots + \theta_n x^n$$

Vì có nhiều biến thể cho hồi quy đa thức, ta muốn hiển thị kết quả bằng một bảng mới và có thể thấy từ bảng dưới đây rằng phép biến đổi đa thức đã cải thiện rất nhiều về sai số. Tuy nhiên, khi sử dụng phép biến đổi đa thức và quyết định mức độ của nó, chúng ta phải rất cẩn thận vì nó có thể **gây ra overfitting**. Ngoài ra, trong bảng dưới đây, overfitting tồn tại đối với một số mô hình. Các độ đo chéo 5 lần cho các mô hình này là âm hoặc thấp mặc dù chúng có giá trị R-squared rất cao cho tập huấn luyện. Note: degree = 2 => Hàm H0 ở bậc 2

Output: **Ta có thể thấy sai số của các Polynomial Regression là thấp nhất trong các mô hình thử nghiệm từ trước đến nay điều này thể hiện ở RMSE thấp và đường chéo fold cao và ta sẽ lựa chọn nó để train các tập bản ghi tiếp theo khi được bổ sung vào tập này để tối ưu hóa**

	Model	Details	Root Mean Squared Error (RMSE)	R-squared (training)	R-squared (test)	5-Fold Cross Validation
2	Polynomial Regression	degree=2, all features, no preprocessing	151200.970	0.830	0.822	0.813
6	Polynomial Ridge Regression	alpha=50000, degree=2, all features	159872.572	0.810	0.801	0.791
8	Polynomial Lasso Regression	alpha=50000, degree=2, all features	166020.484	0.797	0.785	0.779
7	Polynomial Lasso Regression	alpha=1, degree=2, all features	166195.984	0.807	0.785	0.778
0	Polynomial Regression	degree=2, selected features, no preprocessing	190980.547	0.730	0.716	0.714
1	Polynomial Regression	degree=3, selected features, no preprocessing	189235.269	0.749	0.721	0.595
3	Polynomial Regression	degree=3, all features, no preprocessing	186433.648	0.874	0.729	-0.927
5	Polynomial Ridge Regression	alpha=1, degree=2, all features	150177.258	0.838	0.824	-3168.943
4	Polynomial Regression	degree=2, all features	151654.993	0.840	0.821	-11230.411

III, Nhận xét – Kết quả :

+ Việc xử lý và tìm hiểu sơ qua dữ liệu để xây dựng model là cần thiết bởi vì nếu làm sạch dữ liệu thì khi chạy sẽ chính xác hơn, còn tìm hiểu qua dữ liệu thì giúp ta hiểu rõ dữ liệu để dễ build model phù hợp
+ Trong các chương trình Data Science thì người ta sẽ chạy nhiều model Machine Learning để xem sai số của model nào thấp nhất trong các model phù hợp thì chúng ta sẽ chọn để train các tập dữ liệu tiếp theo

+ VD: Ở dự án trên thì nếu ta thêm bản ghi các giá nhà từ tháng 5 2025 trở về sau thì ta có thể dùng Model Polynomial Regression để chạy !

LINK CODE : https://github.com/LongAndLe/Predict-House-Price-