**TRƯỜNG ĐẠI HỌC KINH TẾ - ĐẠI HỌC ĐÀ NẴNG**

**KHOA THỐNG KÊ – TIN HỌC**



**BÁO CÁO BÀI TẬP NHÓM**

**BÀI TOÁN HỒI QUY DỰ ĐOÁN DOANH THU VÀ LỢI NHUẬN CỦA CỬA HÀNG XE ĐẠP**

**GVHD:** TS. Lê Diên Tuấn

**Lớp học phần:** MIS3041\_47K21.2

**Nhóm:** 10

|  |  |  |
| --- | --- | --- |
| **Thành viên** | **Đóng góp (%)** | **Ghi chú** |
| Nguyễn Thị Thanh Hằng | 100% |  |
| Trần Thị Hằng | 100% |  |
| Cai Thị Thùy Kiên | 100% | Nhóm trưởng (0812485969) |
| Bùi Lê Khánh Vy | 100% |  |

**ĐÀ NẴNG 2024**

Mục lục

[1. Giới thiệu tổng quan 3](#_Toc163309181)

[2. Cơ sở lý thuyết 3](#_Toc163309182)

[2.1 Khái niệm về hồi quy tuyến tính 3](#_Toc163309183)

[2.2 Hồi quy tuyến tính đa biến 3](#_Toc163309184)

[2.3 Các phương pháp ước lượng trong mô hình hồi quy tuyến tính đa biến 4](#_Toc163309185)

[2.4 Đánh giá mô hình 4](#_Toc163309186)

[3. Triển khai đề tài 5](#_Toc163309187)

[3.1 Đặt vấn đề 5](#_Toc163309188)

[3.2 Yêu cầu cụ thể 5](#_Toc163309189)

[3.3 Mô tả dữ liệu 5](#_Toc163309190)

[3.4 Framework 7](#_Toc163309191)

[3.5 Triển khai trên python 7](#_Toc163309192)

[3.5.1 Chuẩn bị dữ liệu 7](#_Toc163309193)

[3.5.2 Tiền xử lý dữ liệu 9](#_Toc163309194)

[3.5.3 Xây dựng mô hình hồi quy tuyến tính đa biến 18](#_Toc163309195)

[3.5.4 Dự báo giá nhà bằng mô hình đã xây dựng 25](#_Toc163309196)

[3.5.5 Đánh giá mô hình 26](#_Toc163309197)

[3.5.6 Biểu đồ Scatter plot 29](#_Toc163309198)

[4. Kết luận 31](#_Toc163309199)

[5. Tài liệu tham khảo 32](#_Toc163309200)

**Danh mục hình ảnh**

[Hình 1: Framework 7](#_Toc163308187)

[Hình 2: Kết quả đọc dữ liệu từ file Excel 8](#_Toc163308188)

[Hình 3: Kết quả mô tả dữ liệu 9](#_Toc163308189)

[Hình 4: Tiền xử lý - Loại bỏ dữ liệu thừa 9](#_Toc163308190)

[Hình 5: Đồ thị phân phối dữ liệu trước khi loại bỏ outliers 11](#_Toc163308191)

[Hình 6: Đồ thị boxplot trước khi loại bỏ outliers 13](#_Toc163308192)

[Hình 7: Kết quả loại bỏ outliers bằng IQR 14](#_Toc163308193)

[Hình 8: Đồ thị boxplot sau khi loại bỏ outliers 15](#_Toc163308194)

[Hình 9: Đồ thị phân phối dữ liệu sau khi loại bỏ outliers 16](#_Toc163308195)

[Hình 10: Ma trận tương quan 18](#_Toc163308196)

[Hình 11: Xây dựng mô hình - biến độc lập 19](#_Toc163308197)

[Hình 12: Xây dựng mô hình - Biến phụ thuộc price 20](#_Toc163308198)

[Hình 13: Xây dựng tập dữ liệu huấn luyện 21](#_Toc163308199)

[Hình 14: Xây dựng tập dữ liệu kiểm tra 22](#_Toc163308200)

[Hình 15: Xây dựng mô hình hồi quy 23](#_Toc163308201)

[Hình 16: Hệ số chặn 23](#_Toc163308202)

[Hình 17: Hệ số hồi quy của từng biến độc lập 23](#_Toc163308203)

[Hình 18: Dự đoán giá nhà bằng mô hình hồi quy 26](#_Toc163308204)

[Hình 19: Đánh giá mô hình bằng MSE, R square 27](#_Toc163308205)

[Hình 20: Kết quả kiểm định t - test cho mô hình 28](#_Toc163308206)

[Hình 21: Kết quả kiểm định F - test 29](#_Toc163308207)

[Hình 22: Biểu đồ so sánh giá trị dự đoán và thực tế 30](#_Toc163308208)

[Hình 23: Biểu đồ dự đoán và sai số 31](#_Toc163308209)

# Giới thiệu tổng quan

Dự đoán giá nhà trong lĩnh vực bất động sản cũng là một khía cạnh quan trọng đối với việc quản lý và ra quyết định chiến lược. Trong thị trường bất động sản ngày nay, việc hiểu và dự đoán giá nhà dựa trên các yếu tố khác nhau đã trở thành một phần không thể thiếu trong chiến lược kinh doanh của các doanh nghiệp. Điều này đặt ra một thách thức lớn cho các nhà quản lý và nhà phân tích dữ liệu. Hồi quy là một trong những công cụ mạnh mẽ và linh hoạt được sử dụng để dự đoán giá nhà trong lĩnh vực bất động sản. Bằng cách dự đoán hoặc giải thích mối quan hệ giữa một biến phụ thuộc và một hoặc nhiều biến độc lập. Chúng ta có thể xây dựng mô hình hồi quy để dự đoán giá nhà dựa trên các yếu tố này.

Bài phân tích này sẽ giới thiệu về việc sử dụng phương pháp hồi quy tuyến tính đa biến để dự đoán giá nhà trong lĩnh vực bất động sản. Chúng ta sẽ khám phá cách xây dựng mô hình hồi quy dựa trên cách phân tích dữ liệu, mối quan hệ giữa các thuộc tính của căn nhà như diện tích, số lượng phòng, số tầng,..(biến độc lập) và giá nhà (biến phụ thuộc). Qua việc phân tích dữ liệu và áp dụng các kỹ thuật hồi quy, chúng ta có thể có cái nhìn sâu sắc hơn về cách mà các yếu tố có thể tác động đến giá nhà trong thị trường bất động sản.

# Cơ sở lý thuyết

## Khái niệm về hồi quy tuyến tính

* Là phương pháp học máy có giám sát đơn giản, được sử dụng để dự đoán

giá trị đầu ra, nó dựa trên thống kê để thiết lập mối quan hệ giữa một biến phụ thuộc và một nhóm tập hợp các biến độc lập.

## Hồi quy tuyến tính đa biến

* Trong phân tích hồi quy tuyến tính đa biến, tập dữ liệu chứa một biến phụ thuộc và nhiều biến độc lập.
* **Phương trình của mô hình hồi quy tuyến tính:**

|  |
| --- |
| Y = β0 + β1X1 + β2X2 + ... + βnXn + ε |

Trong đó: Y: Biến phụ thuộc

β0: Hệ số chặn

β1, β2, ..., βn: Hệ số hồi quy của các biến độc lập

X1, X2, ..., Xn: Các biến độc lập

ε: Sai số ngẫu nhiên

## Các phương pháp ước lượng trong mô hình hồi quy tuyến tính đa biến

* **Phương pháp bình phương nhỏ nhất (OLS):** Phương pháp OLS là phương pháp ước lượng phổ biến nhất được sử dụng để ước lượng các hệ số trong mô hình hồi quy tuyến tính. Phương pháp này nhằm mục đích tìm ra tập hợp các hệ số làm cho tổng bình phương của các sai số là nhỏ nhất.

https://lh7-us.googleusercontent.com/8JQx9pEE66AwI8UGwL1ZS_CKSLn13rggmUZavJ6q-mNj9R7trAU-wqrN7-aoZgI_0WBzHPQQPX75x71ucr5p9oKMokhM8xv3cXywtngh4cf7omxMMSzxNeCDc7jdS5_kv8CKf7mfPVmG

## Đánh giá mô hình

Sau khi ước lượng các tham số, cần đánh giá mô hình để xem nó có phù hợp với dữ liệu hay không. Các tiêu chí đánh giá mô hình bao gồm:

* Hệ số tương quan R (coefficient of correlation): yếu tố nào có R càng lớn thì ảnh hưởng càng nhiều
* Bình phương của R (R square): Hệ số xác định cho biết tỷ lệ phần trăm biến đổi của biến phụ thuộc được giải thích bởi các biến độc lập, yếu tố nào có R square càng lớn thì mối quan hệ giữa yếu tố đó và biến y càng chặt chẽ.
* Hệ số hồi quy  (regression coefficient): yếu tố nào có  cao thì ảnh hưởng nhiều hơn, tuy nhiên các yếu tố có đơn vị khác nhau (tuổi, mmol/L, mmHg....) nên không thể so sánh mức ảnh hưởng giữa các yếu tố. Nếu muốn so sánh phải đổi các yếu tố có cùng đơn vị là độ lệch chuẩn, lúc đó ta có hệ số hồi quy chuẩn hóa: https://lh7-us.googleusercontent.com/FBx0NCUE4LkVc6TYz1e57n-5FZkQrv3htShgiScRr8H1Hh0ST8PcdisOak4-9IaorIFQmHHqdt87jWpj9xpWw2w040YeWvn8zo7b5pJuXFTG_TnkIQIrKp1L6vGSJ0h5teosRhNPk3R4( Với Sx là độ lệch chuẩn của x tương ứng và Sy là độ lệch chuẩn của y).
* Trị số p (p value): cho biết mức độ tin cậy của hệ số hồi quy, p càng nhỏ mức ảnh hưởng càng mạnh và ngược lại.

**Các phương pháp đánh giá mô hình:**

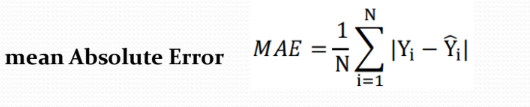
* **Root mean square error** (lỗi trung bình bình phương gốc)

Lỗi trung bình bình phương gốc (RMSE) là thước đo mức độ hiệu quả của mô hình của bạn. Nó thực hiện điều này bằng cách đo sự khác biệt giữa các giá trị dự đoán và giá trị thực tế . R-MSE càng nhỏ tức là sai số càng bé thì mức độ ước lượng cho thấy độ tin cậy của mô hình có thể đạt cao nhất.



* **Mean Absolute error** (độ lớn trung bình của các lỗi)

Mean Absolute Error (MAE) đo độ lớn trung bình của các lỗi trong một tập hợp các dự đoán mà không cần xem xét hướng của chúng. Đó là giá trị trung bình trên mẫu thử nghiệm về sự khác biệt tuyệt đối giữa dự đoán và quan sát thực tế, trong đó tất cả các khác biệt riêng lẻ có trọng số bằng nhau.

****

# Triển khai đề tài

## Đặt vấn đề

Nhằm mục đích dự đoán giá nhà dựa trên các yếu tố có sẵn, bằng việc sử dụng mô hình hồi quy tuyến tính bội, chúng tôi muốn tạo ra một công cụ dự đoán có thể ước lượng giá trị của một căn nhà dựa trên các thuộc tính như diện tích,số lượng phòng và các yếu tố khác. Mục tiêu là cung cấp cho người mua hoặc bán nhà một cái nhìn tổng quan và dự đoán chính xác về giá trị của một căn nhà dựa trên các thông tin sẵn có. Qua việc áp dụng phân tích này, chúng tôi hy vọng có thể hỗ trợ quyết định mua bán nhà và cung cấp giá trị đáng tin cậy cho người dùng.

## Yêu cầu cụ thể

* **Thu thập dữ liệu:** Xác định và thu thập dữ liệu liên quan đến giá nhà bao gồm các thông tin như diện tích, vị trí, số phòng, tiện ích xung quanh, giá bán, v.v.
* **Tiền xử lý dữ liệu:** Loại bỏ các dữ liệu không hợp lệ, xử lý dữ liệu thiếu, chuyển đổi dữ liệu về định dạng phù hợp cho việc phân tích.
* **Phân tích hồi quy tuyến tính bội:** Sử dụng thư viện Python như scikit-learn để xây dựng mô hình hồi quy tuyến tính bội trên tập dữ liệu đã được tiền xử lý.
* **Đánh giá mô hình:** Sử dụng các phép đánh giá như sai số trung bình bình phương (Mean Squared Error), hệ số xác định (R-squared) để đánh giá hiệu suất của mô hình.
* **Đưa ra dự đoán:** Sử dụng mô hình đã được huấn luyện để dự đoán giá nhà dựa trên các dữ liệu mới.

## Mô tả dữ liệu

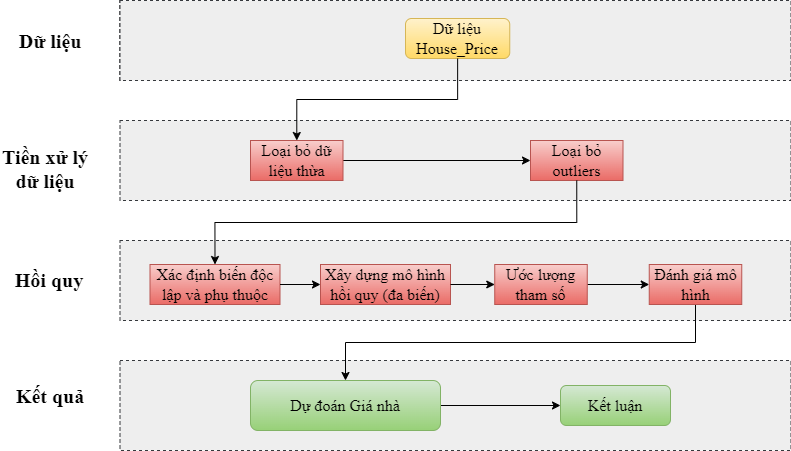
Dữ liệu: [House\_price](https://docs.google.com/spreadsheets/d/1kgdVBVjLdmQz8TKOAWYjdxEz7kIM7wp0/edit#gid=1757702685)

Bộ dữ liệu về giá nhà được thu thập từ các giao dịch bán hoặc bán đấu giá trong khoảng thời gian năm 2014, có chứa các thông tin quan trọng như price, bedrooms, sqft\_living, sqft\_lot và sqft\_basement, cũng như các yếu tố khác như condition,.... Dữ liệu bao gồm 1000 dòng và 18 cột.

**Mô tả dữ liệu**

* **date:** thời gian ghi lại hoặc cập nhật, thêm mới thông tin liên quan đến giá nhà.
* **price:** giá bán của một căn nhà.
* **bedrooms:** Số lượng phòng ngủ trong căn nhà.
* **bathrooms:** Số lượng phòng tắm trong căn nhà.
* **sqft\_living:** Tổng diện tích không gian sống trong nhà, tính bằng feet vuông.
* **sqft\_lot:** Diện tích lô đất .
* **floors:** Số tầng của căn nhà.
* **waterfront**: Căn nhà có hồ hay sông nằm cạnh không: 0 là không và 1 là có
* **view:**  Mô tả mức độ cảnh quan: 0 là có cảnh quan nhưng bị che khuất, 1 là có nhưng có thể bị che hoặc hạn chế, 3 là cảnh quan tốt nhưng không phải các phòng đều thấy, 4 là cảnh quan tuyệt vời có thể nhìn thấy ở mọi góc độ.
* **condition:** Tình trạng căn nhà: 3 là trung bình, 4 là tốt, 5 rất tốt.
* **sqft\_above:** Diện ngôi nhà mà không bao gồm tầng hầm.
* **sqft\_basement**: Diện tích của phần hầm.
* **yr\_built:** Năm căn nhà được xây dựng.
* **yr\_renovated:** Năm mà căn đã được cải tạo hoặc nâng cấp, nếu có. Nếu không, giá trị là 0.
* **street:** địa chỉ căn nhà.
* **city:** thành phố
* **statezip:** Thể hiện vị trí của căn nhà  kết hợp giữa thông tin về bang và mã zip code của mỗi tài sản
* **country:** Quốc gia

## Framework



Hình 1: Framework

## Triển khai trên python

### Chuẩn bị dữ liệu

* Khai báo thư viện cần thiết:

import pandas as pd # cung cấp các cấu trúc dữ liệu và công cụ phân tích dữ liệu

import numpy as np #cung cấp các công cụ cho tính toán

from sklearn.linear\_model import LinearRegression #cung cấp mô hình hồi quy tuyến tính

from sklearn.metrics import mean\_squared\_error #cung cấp hàm mean\_squared\_error để tính toán độ lỗi trung bình

from sklearn.metrics import r2\_score #tính toán hệ số xác định R^2

from sklearn.model\_selection import train\_test\_split # chia dữ liệu train và test

import matplotlib.pyplot as plt #công cụ để tạo biểu đồ và hình vẽ

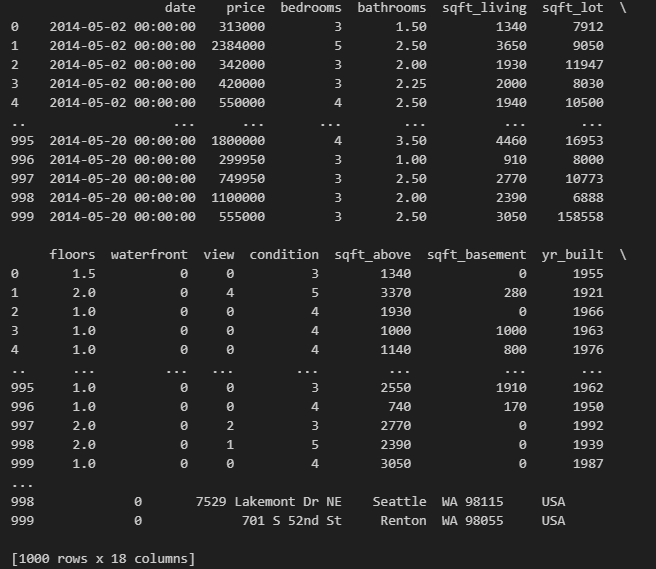
import seaborn as sns # trực quan hóa dữ liệu dựa trên matplotlib

* Đọc dữ liệu từ file excel

data= pd.read\_excel('house\_price.xls') # Đọc file excel

print(data)

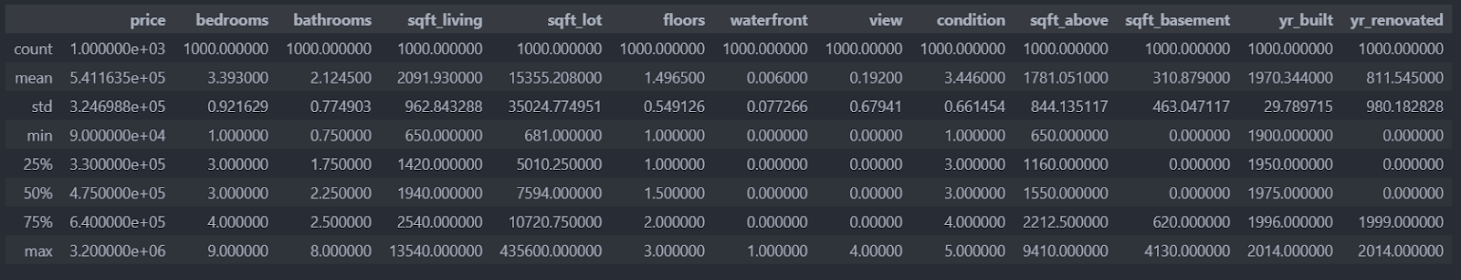
Kết quả:



Hình 2: Kết quả đọc dữ liệu từ file Excel

* Mô tả dữ liệu:

data.describe()

****

Hình 3: Kết quả mô tả dữ liệu

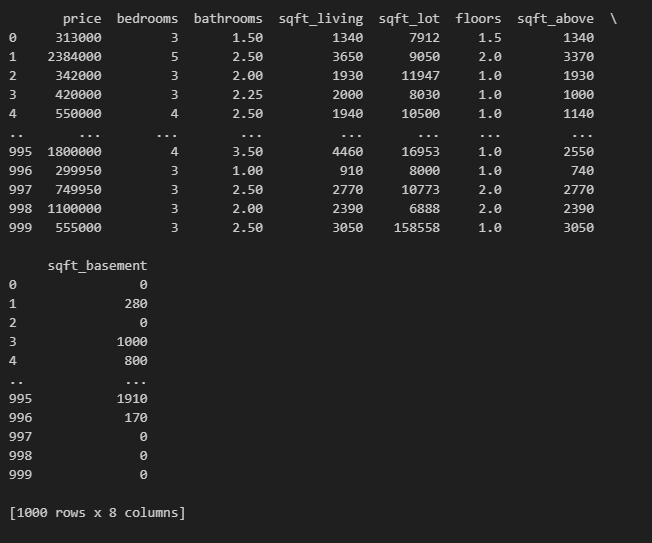
### Tiền xử lý dữ liệu

* **Loại bỏ dữ liệu không cần thiết:**

# Loại bỏ các cột không cần thiết

data\_drop = data.drop(['date','waterfront','view','condition','yr\_built','yr\_renovated','street','city','statezip','country'], axis=1)

print (data\_drop)



Hình 4: Tiền xử lý - Loại bỏ dữ liệu thừa

* **Loại bỏ outliers**

**Đồ thị phân phối dữ liệu trước khi loại bỏ ouliers**

import warnings

warnings.filterwarnings('ignore')

def dist\_custom(dataset, columns\_list, rows, cols, suptitle):

     #Tạo lượt đồ thị con kích thước 16x20(đơn vị)

    fig, axs = plt.subplots(rows, cols, figsize=(16,20))

    #Thêm tiêu đề cho các biểu đồ có kích thước là 16 đơn vị

    fig.suptitle(suptitle, size=16)

    #Làm phẳng mản 2D của đồ thị con thành mảng 1D

    axs=axs.flatten()

    for i, data in enumerate(columns\_list):

        #Vẽ đồ thị phân phối sử dụng Seaborn

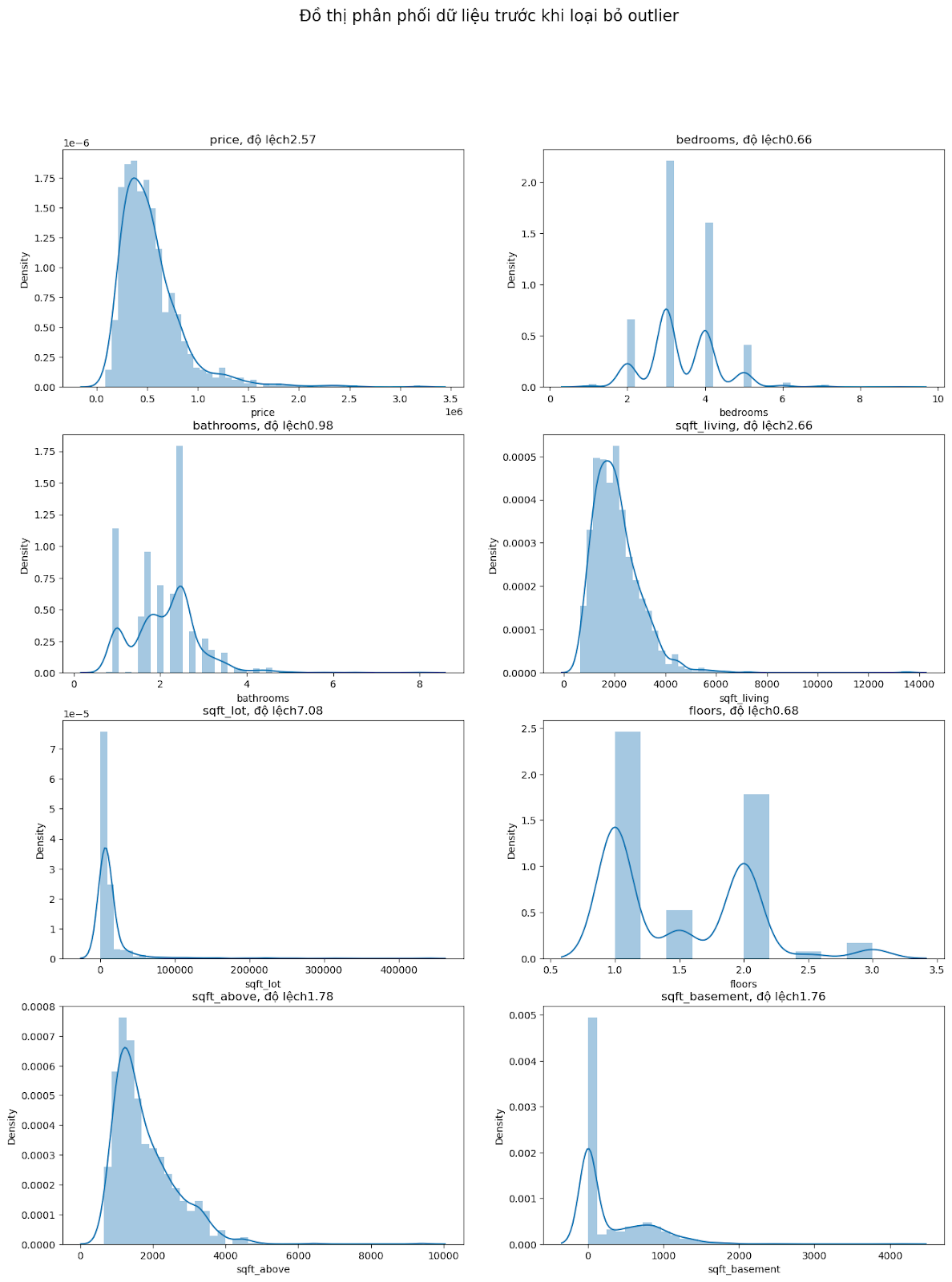
        sns.distplot(dataset[data], ax=axs[i])

         #Đặt tiêu đề cho mỗi đồ thị

        axs[i].set\_title(data+', độ lệch' + str(round(dataset[data].skew(axis=0, skipna=True),2)))

dist\_custom(dataset=data\_drop, columns\_list=data\_drop.columns, rows=4, cols=2, suptitle='Đồ thị phân phối dữ liệu trước khi loại bỏ outlier')

Kết quả:

****

Hình 5: Đồ thị phân phối dữ liệu trước khi loại bỏ outliers

Từ đồ thị phân phối dữ liệu trên, ta có thể thấy biểu đồ có độ lệch cao và lệch về phía trái cho thấy dữ liệu có sự biến động lớn và không phân bố đều. Điều này có thể là do có sự xuất hiện của các outlier( yếu tố ngoại lai) gây ra. Nó sẽ ảnh hưởng đến quá trình phân tích và làm mất tính chính xác của mô hình, dẫn đến dự đoán cũng sẽ sai lệch.

**Đồ thị boxplot trước khi loại bỏ outliers**

# Xây dựng 1 hàm để vẽ đồ thị boxplot cho dữ liệu

def boxplots\_custom(dataset, columns\_list, rows, cols, suptitle):

    # Tạo lưới đồ thị con

    fig, axs = plt.subplots(rows, cols, sharey=True, figsize=(16,12))

    fig.suptitle(suptitle, size=16) # Tạo tiêu đề chung

    axs = axs.flatten() # Làm phẳng mang 2D thành mảng 1D

    # chạy vòng lặp for để tiến hành vẽ các đồ thị con

    for i, data in enumerate(columns\_list):

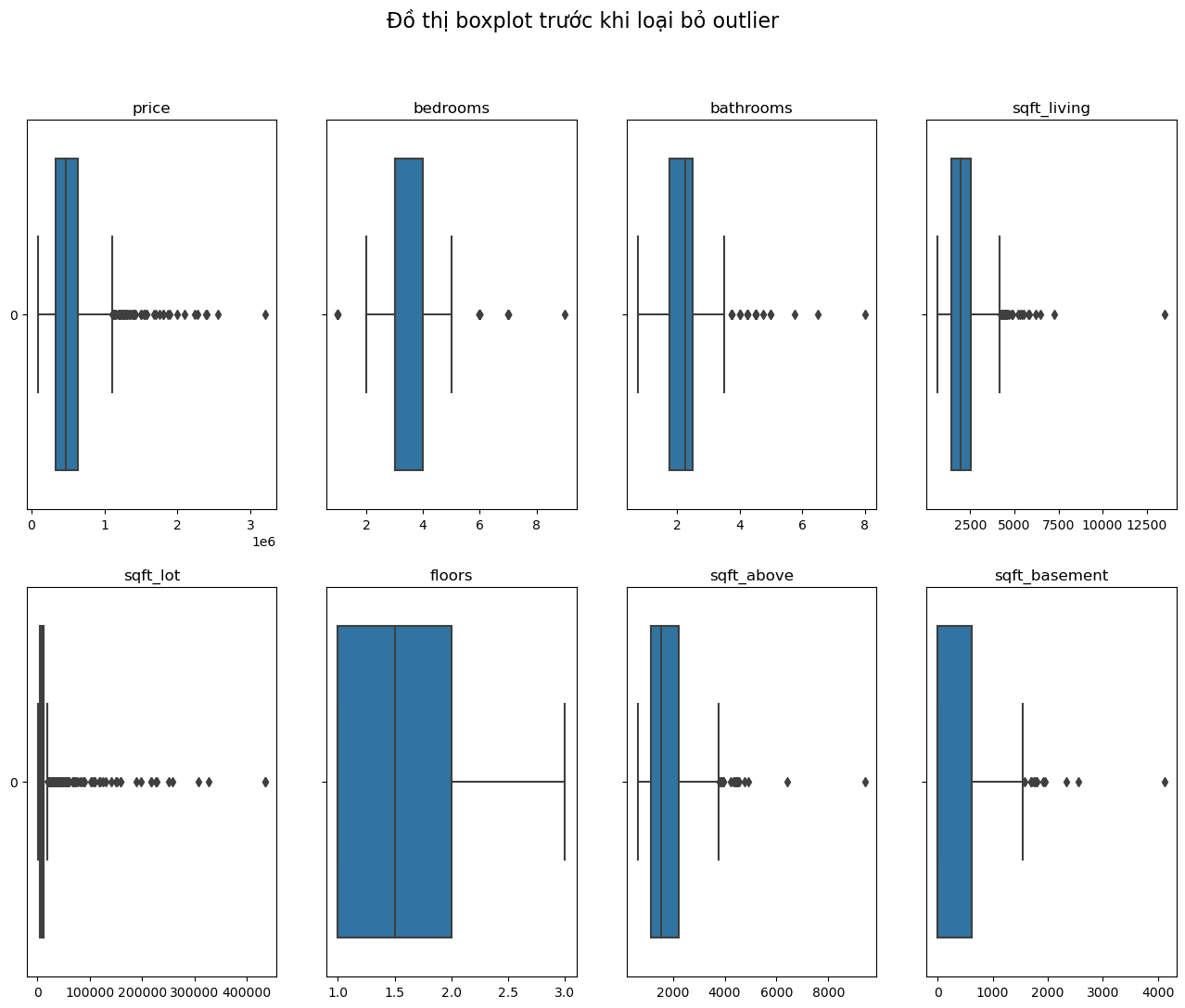
        sns.boxplot(data=dataset[data], orient='h', ax=axs[i])

        axs[i].set\_title(data) # đặt tiêu đề cho từng đồ thị

# Tiến hành gọi hàm để vẽ đồ thị phân phối dữ liệu trước khi loại bỏ outliers

boxplots\_custom(dataset=data\_drop, columns\_list= data\_drop.columns, rows=2, cols=4, suptitle='Đồ thị boxplot trước khi loại bỏ outlier')

Kết quả:



Hình 6: Đồ thị boxplot trước khi loại bỏ outliers

Đồ thị boxlpot trên cho ta thấy trong tập dữ liệu có chứa dữ liệu ngoại lai. Vì vậy, chúng ta cần loại bỏ outlier trước khi tiến hành xây dựng mô hình để tránh sai lệch, ảnh hưởng đến kết quả dự đoán của mô hình

**Loại bỏ outliers bằng IQR**

# Xác định Q1, Q3

Q1 = data\_drop.quantile(0.25)

Q3 = data\_drop.quantile(0.75)

IQR = Q3 - Q1

print('IQR: ', IQR)

# Tính giới hạn trên, dưới

lower = Q1-1.5\*IQR

upper = Q3+1.5\*IQR

# Xác định các outlier

outliers = ((data\_drop < lower) | (data\_drop > upper)).any(axis=1)

# số lượng outlier của dữ liệu

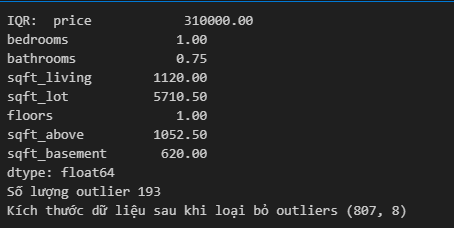
print('Số lượng outlier', outliers.sum())

#  Loại bỏ outlier

clean\_data = data\_drop[~outliers]

# Kích thước dữ liệu sau khi loại bỏ outlier

print('Kích thước dữ liệu sau khi loại bỏ outliers', clean\_data.shape)

****

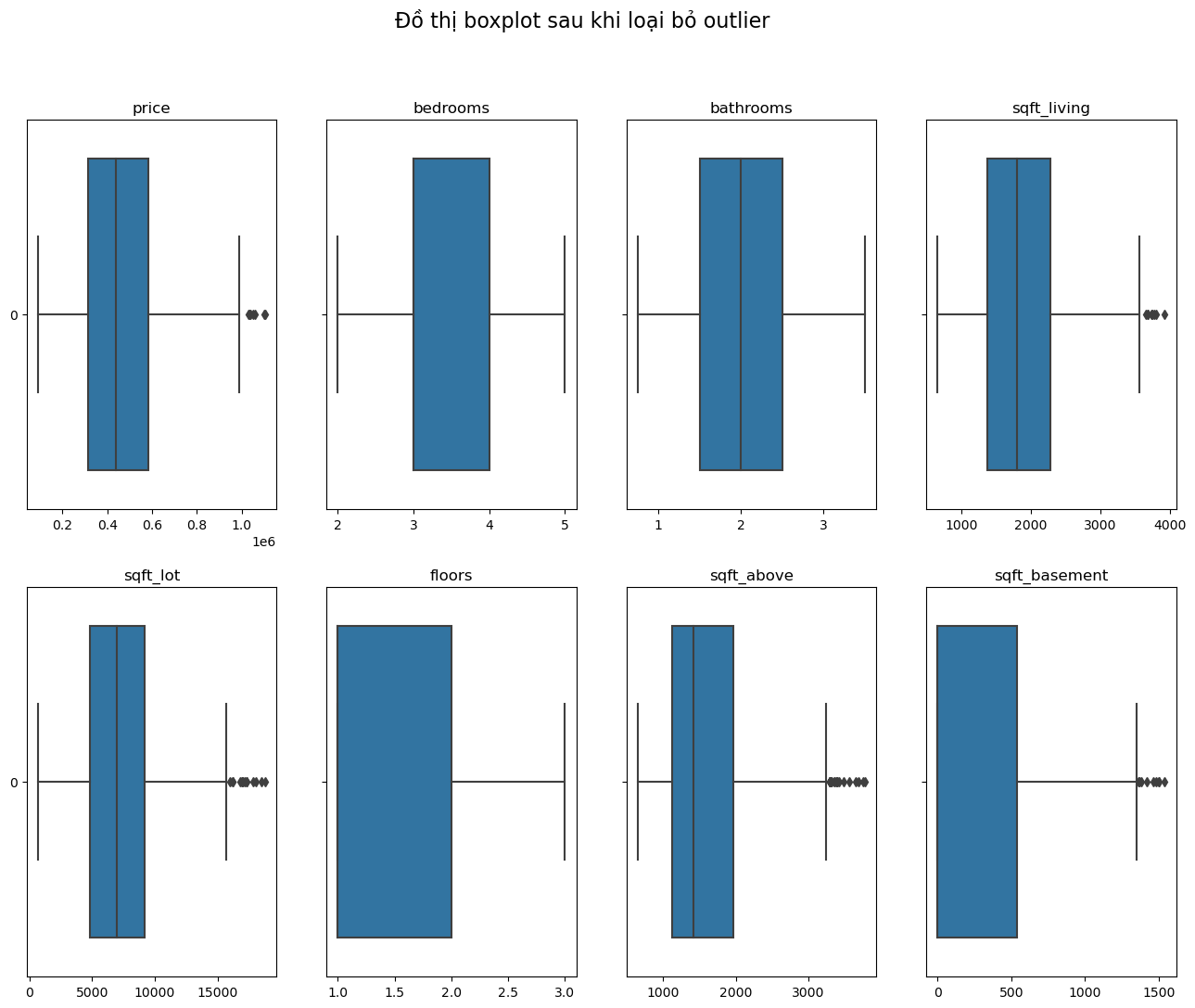
Hình 7: Kết quả loại bỏ outliers bằng IQR

Trong bước tiền xử lý này, chúng ta sẽ sử dụng phương pháp IQR để loại bỏ các outlier trong tập dữ liệu, số lượng outlier trong tập dữ liệu là 193, sau khi loại bỏ các outlier chúng ta sẽ còn lại 807 dòng dữ liệu

**Đồ thị boxplot sau khi loại bỏ outliers**

boxplots\_custom(dataset=clean\_data, columns\_list= clean\_data.columns, rows=2, cols=4, suptitle='Đồ thị boxplot sau khi loại bỏ outlier')

Kết quả:



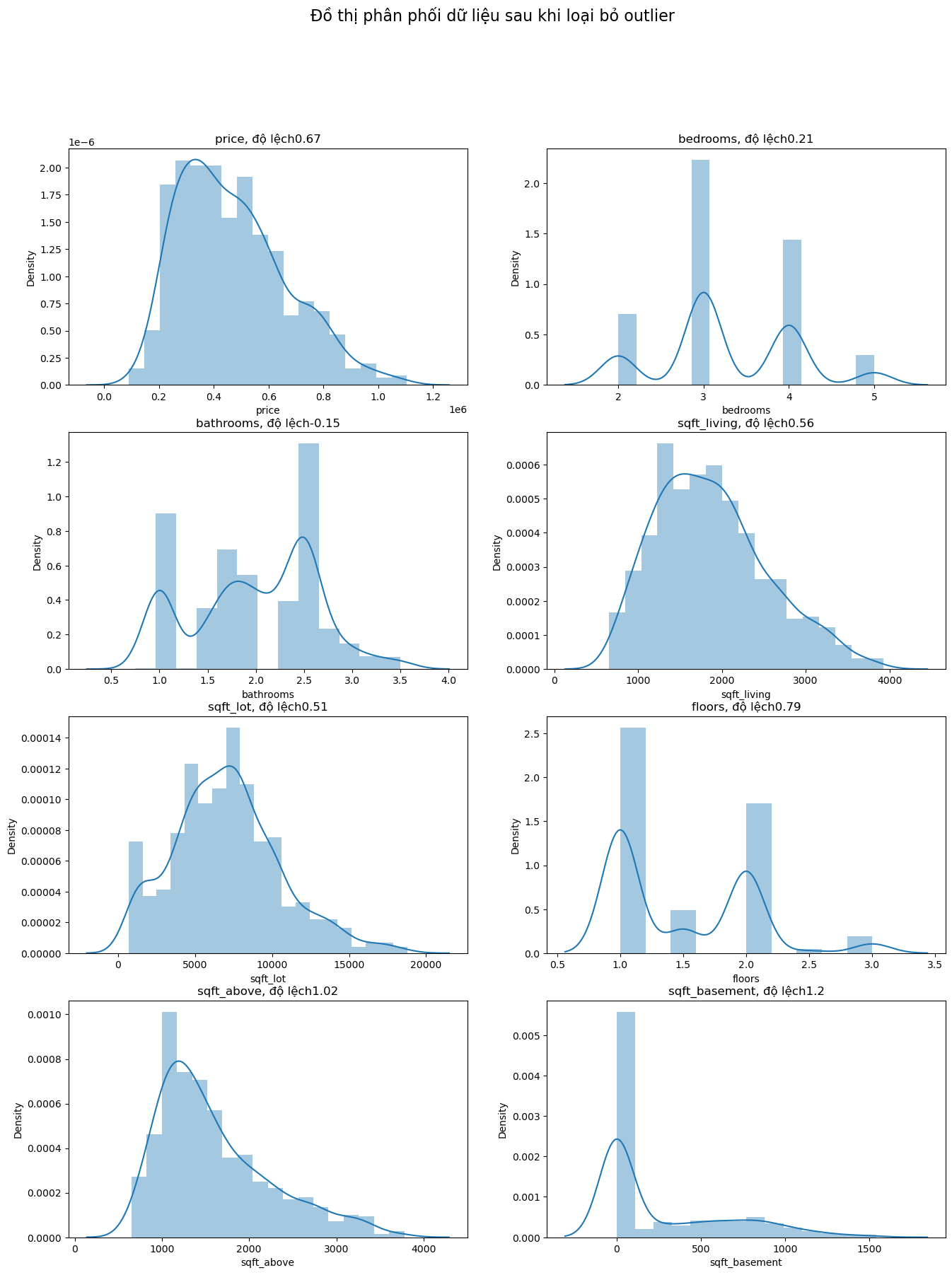
Hình 8: Đồ thị boxplot sau khi loại bỏ outliers

Đồ thị boxplot sau khi loại bỏ outliers cho thấy lượng dữ liệu ngoại lai trong tập dữ liệu được loại bỏ đáng kể, tuy không loại bỏ được 100% nhưng điều này đã giúp cho việc xây dựng mô hình dự đoán trở nên chính xác hơn

**Đồ thị phân phối dữ liệu sau khi loại bỏ ouliers**

dist\_custom(dataset=clean\_data, columns\_list=clean\_data.columns, rows=4, cols=2, suptitle='Đồ thị phân phối dữ liệu sau khi loại bỏ outlier')

Kết quả:



Hình 9: Đồ thị phân phối dữ liệu sau khi loại bỏ outliers

Sau khi loại bỏ các outlier và xây dựng lại đồ thị phân phối thì ta thấy độ lệch chuẩn của các đồ thị đã giảm xuống, đỉnh của mô hình đã tròn hơn dần lệch về phía phải và có dạng phân phối gần với một phân phối chuẩn. Điều này chỉ ra rằng, sự biến động trong giữ liệu giảm xuống các giá trị dữ liệu giờ đây phân bố gần hơn với giá trị trung bình hoặc trung vị của dữ liệu, từ đó có thể xây dựng mô hình và dự đoán kết quả một cách tương đối và chính xác hơn.

**Xây dựng ma trận tương quan**

# Tạo ma trận tương quan để biết nhưng biến độc lập được đưa vào có ý nghĩa nhiều không

corr\_matrix = clean\_data.corr()

# Vẽ heatmap

plt.figure(figsize=(10, 8))

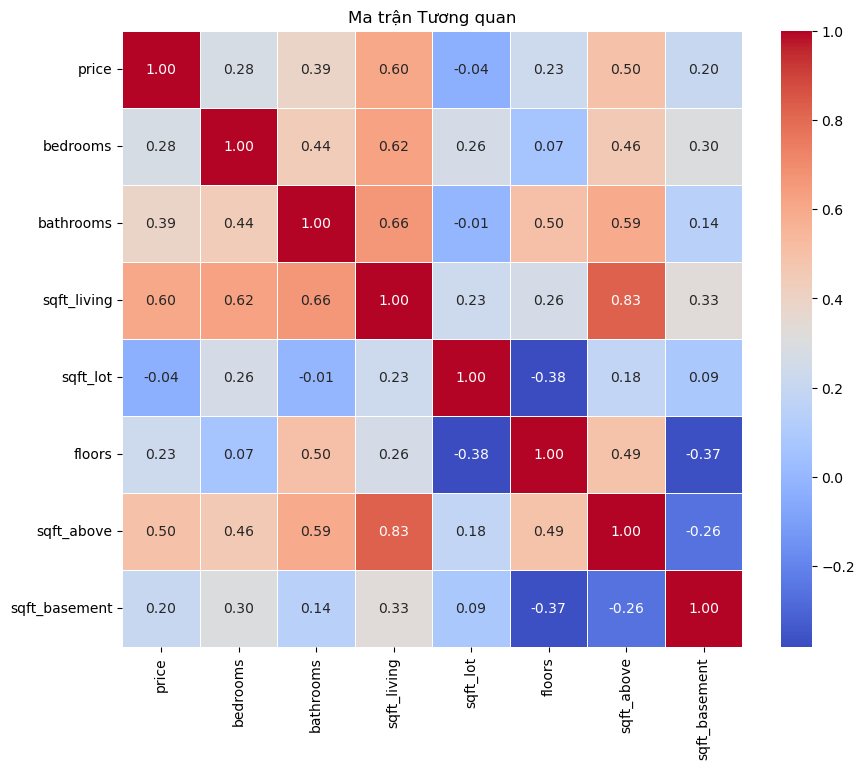
sns.heatmap(corr\_matrix, annot=True, cmap='coolwarm', fmt=".2f", linewidths=0.5)

plt.title('Ma trận Tương quan')

plt.show()

Tại đây, chúng ta sử dụng hàm corr() trên tập dữ liệu clean\_data để vẽ ma trận tương quan.

Kết quả:



Hình 10: Ma trận tương quan

Ma trận trên cho biết mối tương quan giữa tất cả các cột trong toàn bộ tập dữ liệu, heatmap trên đã trực quan hóa mối tương quan bằng cách sử dụng màu sắc để thể hiện mức độ tương quan mạnh yếu giữa tất cả các biến. Từ ma trận này có thể chọn ra các biến với độ tương quan phù hợp để xây dựng và phân tích mô hình

### Xây dựng mô hình hồi quy tuyến tính đa biến

**Lựa chọn biến độc lập và biến phụ thuộc**

Input Code:

**# Chọn các biến độc lập x và biến phụ thuộc y**

**x = data\_drop[['bedrooms', 'bathrooms', 'sqft\_living','sqft\_lot','floors','sqft\_above','sqft\_basement']]**

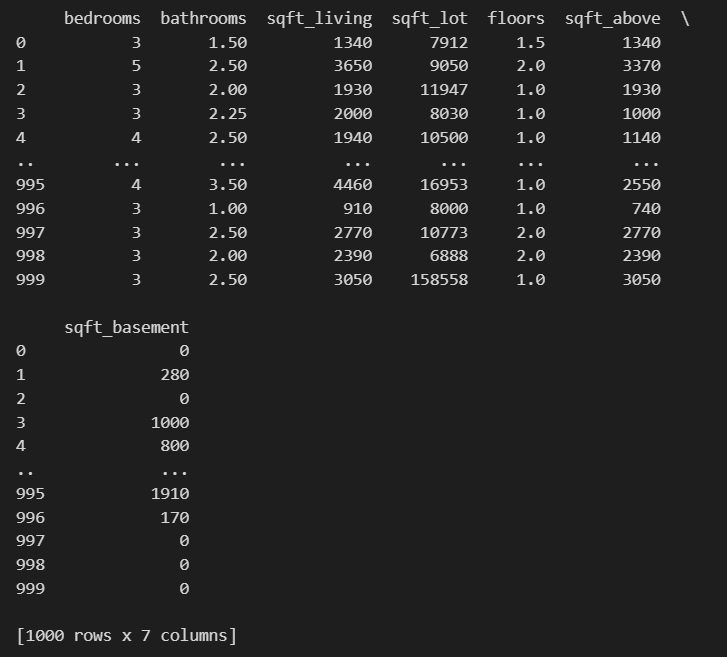
**y = data\_drop['price']**

**# In ra các biến độc lập x và biến phụ thuộc y**

**print(x)**

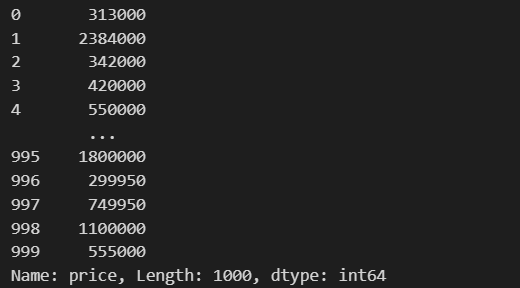
Output: Hình ảnh minh họa

* Biến độc lập x

****

Hình 11: Xây dựng mô hình - biến độc lập

* Biến phụ thuộc y

****

Hình 12: Xây dựng mô hình - Biến phụ thuộc price

**Chia dữ liệu thành 2 phần**:

* Data\_train: dữ liệu dùng để huấn luyện mô hình hồi quy
* Data\_test: dữ liệu dùng để kiểm tra lại mô hình đã được huấn luyện

Input Code:

**# 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)**

**# Kết hợp x\_train và y\_train thành một DataFrame**

**data\_train = pd.concat([x\_train, y\_train], axis=1)**

**# Kết hợp x\_test và y\_test thành một DataFrame**

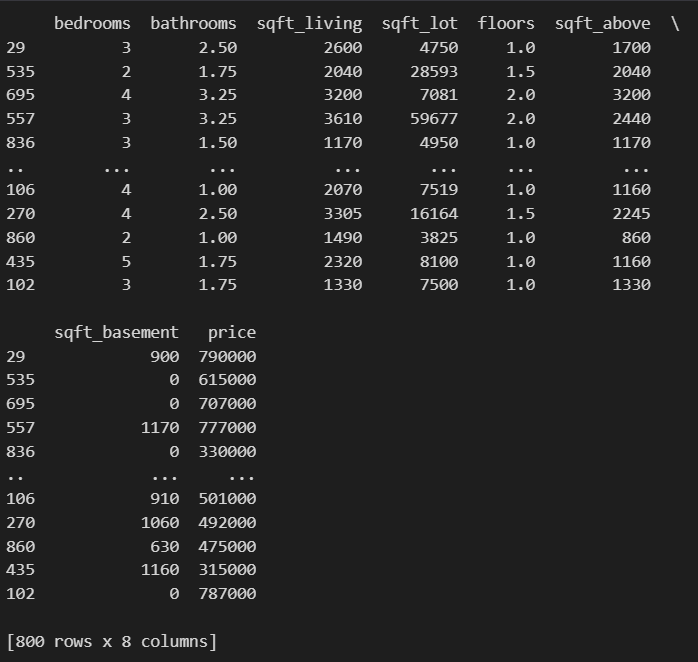
**data\_test = pd.concat([x\_test,y\_test],axis = 1)**

**# In ra dữ liệu train và dữ liệu test được chọn**

**print (data\_train)**

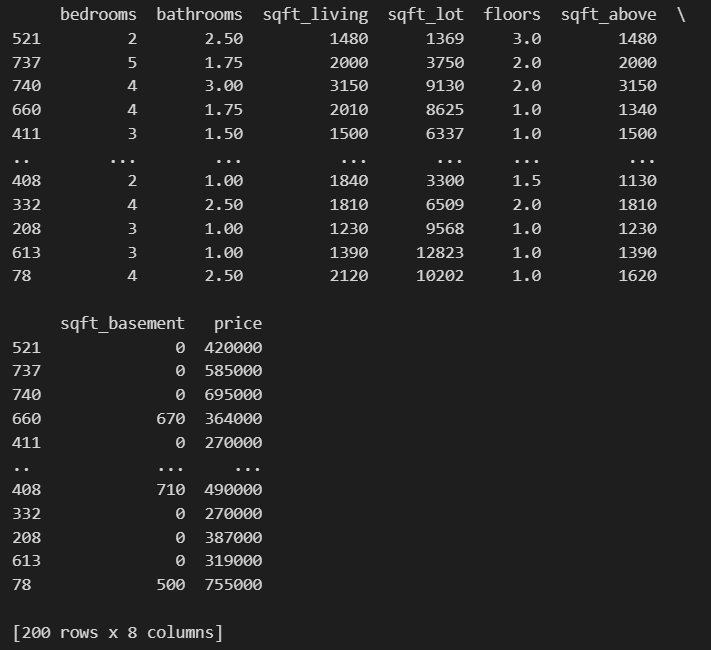
Output: Hình ảnh minh họa

* Tập dữ liệu huấn luyện (data\_train) được random trong 1 lần chạy

****

Hình 13: Xây dựng tập dữ liệu huấn luyện

* Tập dữ liệu kiểm tra (data\_test) được random trong 1 lần chạy

****

Hình 14: Xây dựng tập dữ liệu kiểm tra

**Xây dựng mô hình hồi quy tuyến tính trên tập dữ liệu huấn luyện**

Input Code:

# Tạo một đối tượng mô hình hồi quy tuyến tính với thư viện scikit-learn đã khai báo

lm = LinearRegression()

# Huấn luyện mô hình bằng dữ liệu huấn luyện

lm.fit(x\_train, y\_train)

Output: Hình ảnh minh họa

* Xây dựng mô hình thành công



Hình 15: Xây dựng mô hình hồi quy

**Ước lượng tham số cho mô hình**

Input Code:

# In ra hệ số chặn của mô hình

print('Hệ số chặn (intercept):', lm.intercept\_)

# tạo 1 dataframe để xem các hệ số hồi quy của các biến độc lập pd.DataFrame(lm.coef\_, x\_train.columns)

Output: Hình ảnh minh họa

* Hệ số chặn

**https://lh7-us.googleusercontent.com/W-53Rc6MBfSZjB0X951HvzQbY6pW3FRPQ6bgJUkOn6_XZEv42A546T-r848ZnHYHc9ww4NVY-3UmmWw9vlz3dDyl18FY7BpYuIBbIP8DCEwWyeTtLGzDyXXAr6iTyfy5Ov6BCqYjMgYt**

Hình 16: Hệ số chặn

* Hệ số hồi quy của từng biến độc lập

****

Hình 17: Hệ số hồi quy của từng biến độc lập

**Đọc ý nghĩa các tham số**

* **Hệ số chặn = 92559,794299978**

Hệ số chặn trên cho thấy giá trị của biến phụ thuộc (price) khi không bị ảnh hưởng bởi các biến độc lập ( bedrooms, bathrooms, sqft\_living, sqft\_lot, floors, sqft\_above, sqft\_basement)  là 92559,794299978.

* **Hệ số hồi quy của từng biến độc lập**
* bedrooms = - 24397,662463
  + Mối quan hệ với biến phụ thuộc: nghịch
  + Mức độ ảnh hưởng đến biến phụ thuộc: cao
  + Ý nghĩa: Số lượng phòng ngủ sẽ có ảnh hưởng  lớn và ngược chiều với giá nhà. Mỗi phòng ngủ thêm vào ngôi nhà có thể làm giảm giá bán của ngôi nhà lên đến 24397,662463$.
* bathrooms = - 39868,203975
  + Mối quan hệ với biến phụ thuộc: nghịch
  + Mức độ ảnh hưởng đến biến phụ thuộc: cao
  + Ý nghĩa: Số lượng phòng tắm sẽ có ảnh hưởng lớn và ngược chiều với giá nhà. Mỗi phòng tắm tăng thêm có thể làm giá nhà giảm nhiều đến 39868,203975$.
* sqft\_living = 186,644751
  + Mối quan hệ với biến phụ thuộc: thuận
  + Mức độ ảnh hưởng đến biến phụ thuộc: trung bình
  + Ý nghĩa: Diện tích của ngôi nhà sẽ có ảnh hưởng tích cực đến giá nhà nhưng ảnh hưởng của nó không quá nhiều. Mỗi mét vuông diện tích tăng thêm cũng sẽ làm cho giá bán của ngôi nhà tăng lên 186,644751$.
* sqft\_lot = -1, 061603
  + Mối quan hệ với biến phụ thuộc: nghịch
  + Mức độ ảnh hưởng đến biến phụ thuộc: thấp
  + Ý nghĩa: Diện tích của khu đất của ngôi nhà có ảnh hưởng ngược chiều đối với giá nhà nhưng ảnh hưởng này rất nhỏ. Mỗi mét vuông diện tích của ngôi nhà tăng lên chỉ khiến cho giá nhà giảm  -1, 061603$
* floors = 34173,210806
  + Mối quan hệ với biến phụ thuộc: thuận
  + Mức độ ảnh hưởng đến biến phụ thuộc: cao
  + Ý nghĩa: Số tầng của ngôi nhà sẽ có ảnh hưởng cùng chiều và to lớn đến ngồi nhà. Khi nhà có thêm 1 tầng thì giá nhà sẽ tăng lên 34173,210806$.
* sqft\_above = 90,7089
  + Mối quan hệ với biến phụ thuộc: thuận
  + Mức độ ảnh hưởng đến biến phụ thuộc: trung bình
  + Ý nghĩa: Diện tích tầng trên của ngôi nhà có ảnh hưởng thuận với nhà nhưng tác động không lớn. Diện tích tầng trên tăng thêm 1 mét vuông thì giá nhà sẽ tăng thêm 90.7089$.
* sqft\_basement = 95,936451
  + Mối quan hệ với biến phụ thuộc: thuận
  + Mức độ ảnh hưởng đến biến phụ thuộc: trung bình
  + Ý nghĩa: Diện tích tầng hầm của ngôi nhà có ảnh hưởng thuận với nhà nhưng tác động không lớn. Diện tích tầng trên tăng thêm 1 mét vuông thì giá nhà sẽ tăng thêm 95,936451.

**Nhận xét chung:**

* bedrooms, bathrooms, floors (số lượng phòng ngủ, số lượng phòng tắm, số tầng) là những yếu tố quan trọng nhất ảnh hưởng đến price (giá nhà) .
* bedrooms, bathrooms, sqft\_lot ( số lượng phòng ngủ, số lượng phòng tắm, diện tích khu đất) là những yếu tố có ảnh hưởng không tốt đối với price (giá nhà). Đặc biệt là bedrooms và bathrooms, còn sqft\_lot thì ảnh hưởng rất nhỏ, không đáng kể.
* sqft\_living, floors, sqft\_above, sqft\_basement (diện tích phòng khách, số tầng, diện tích tầng trên, diện tích tầng hầm) là những yếu tố có ảnh hưởng tích cực đối với price (giá nhà). Tuy nhiên chỉ floors là mang lại ảnh hưởng lớn, sqft\_living ở mức trung bình, còn sqft\_above và sqft\_basement thì có ảnh hưởng khá thấp.

### Dự báo giá nhà bằng mô hình đã xây dựng

# Dự báo giá dựa vào tập test

y\_pred = lm.predict(x\_test)

# Tạo DataFrame từ dự đoán

y\_pred\_df = pd.DataFrame(y\_pred, columns=['Price\_predict'], index=x\_test.index)

# Kết hợp các DataFrame và so sánh kết quả

result\_compare = pd.concat([x\_test, y\_test, y\_pred\_df], axis=1)

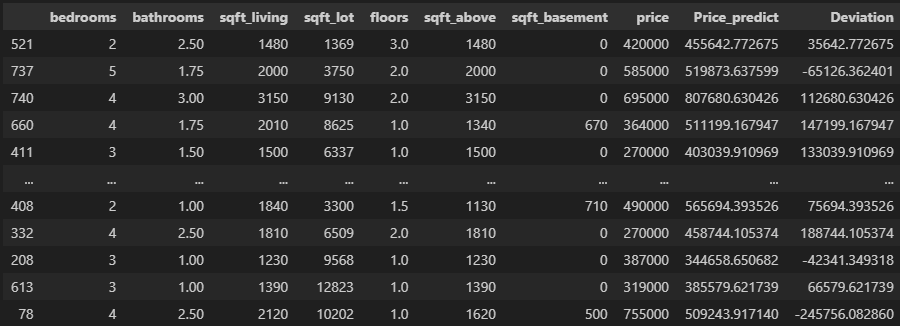
result\_compare['Deviation'] = result\_compare['Price\_predict'] - result\_compare['price']

# Hiển thị kết quả

result\_compare

Sau khi tiến hành bước tiền xử  lý và xây dựng mô hình hồi quy tuyến tính đa biến hoàn thành, chúng ta tiến hành dự báo giá dựa vào tập test

Đầu tiên là sử dụng mô hình tuyến tính lm.predict  đã được huấn luyện để dự đoán giá trên tập test x\_test đã được chia từ trước. Kết quả dự đoán được lưu vào biến y\_pred. Sau đó tạo một Data Frame y\_pred\_df từ các dự đoán y\_pred. Cột của Data Frame này được đặt tên là 'Price\_predict', và các chỉ số của Data Frame này được thiết lập để phù hợp với chỉ số của x\_test. Sử dụng hàm pd.concat() để kết hợp các Data Frame x\_test, y\_test (chứa giá thực tế), và y\_pred\_df (chứa giá dự đoán) thành một Data Frame mới là result\_compare. Sau đó, tính sai số giữa giá dự đoán và giá thực tế và lưu vào cột 'Deviation'.



Hình 18: Dự đoán giá nhà bằng mô hình hồi quy

Kết quả thu được cho thấy giá phòng của từng loại phòng hiện tại (**price**) và giá phòng sau khi chạy mô hình dự đoán (**Price\_predict**) . Đây là kết quả dự đoán giá dựa vào tập test và sau đó so sánh kết quả dự đoán với giá thực tế, từ đó đánh giá hiệu suất của mô hình trên dữ liệu. Cột **Deviation** cho kết quả chênh lệch cao hơn giá thực tế có nghĩa là mô hình dự đoán giá phòng cao hơn giá phòng ban đầu. Và ngược lại, kết quả của cột Deviation thấp  hơn giá phòng ban đầu, điều này có nghĩa là mô hình dự đoán giá thấp hơn so với giá phòng thực tế.

### Đánh giá mô hình

# Đánh giá mô hình sau khi test

# Tính toán MSE

mse\_test = mean\_squared\_error(y\_test, y\_pred)

# Tính toán R^2

r2\_test = r2\_score(y\_test, y\_pred)

# Hiển thị kết quả

print("Mean Squared Error trên tập kiểm tra:", mse\_test)

print("R-squared trên tập kiểm tra:", r2\_test)

Sử dụng MSE để đo lường trung bình của bình phương sai số giữa giá trị dự đoán và giá trị thực tế trên tập kiểm tra. Sau đó tiến hành tính toán R^2 để đánh giá khả năng giải thích của mô hình trên cùng tập dữ liệu này.



Hình 19: Đánh giá mô hình bằng MSE, R square

Giá trị MSE là khoảng 34,281,366,193.47, ở kết quả này giá trị MSE rất cao, cho thấy có sự chênh lệch lớn giữa giá trị dự đoán và giá trị thực tế trên tập test. Cụ thể, sai số bình phương trung bình giữa các dự đoán và giá trị thực tế là rất lớn, điều này có thể làm giảm hiệu suất của mô hình và cho thấy mô hình không dự đoán chính xác giá trị giá.

Kết quả của R^2 nằm trong khoảng từ 0 đến 1, giá trị càng gần 1 thì mô hình giải thích tốt hơn sự biến thiên dữ liệu. Nhưng giá trị R^2 của tập dữ liệu là khoảng 0.582, tức là mô hình có khả năng giải thích được khoảng 58.2% sự biến thiên của dữ liệu trên tập test. R^2 cung cấp một đánh giá về khả năng giải thích của mô hình, và trong trường hợp này, giá trị R^2 không đạt mức cao nhất, cho thấy mô hình không thể giải thích hết được biến thiên của dữ liệu.

**Kiểm định t-test**

# Kiểm định t-test cho từng biến độc lập

for column in x\_test.columns:

    t\_stat, p\_val = stats.ttest\_ind(result\_compare[column], result\_compare['Deviation'], equal\_var=False)

    print(f'Biến độc lập "{column}":')

    print(f'  - Giá trị t-score: {t\_stat}')

    print(f'  - Giá trị p-value: {p\_val}')

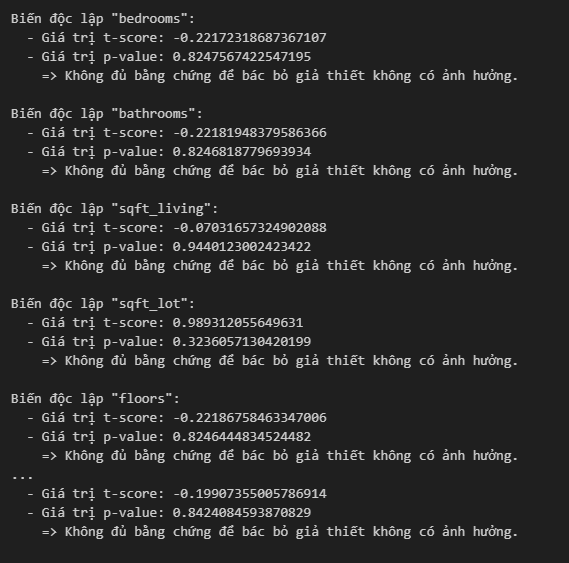
    if p\_val < 0.05:

        print('    => Có đủ bằng chứng để bác bỏ giả thiết không có ảnh hưởng.')

    else:

        print('    => Không đủ bằng chứng để bác bỏ giả thiết không có ảnh hưởng.')

    print()



Hình 20: Kết quả kiểm định t - test cho mô hình

Từ kết quả đánh giá t-test trên, ta thấy tất cả các biến độc lập đều có mức ảnh hưởng nhất định đến biến phụ thuộc.

**kiểm định f-test**

from scipy.stats import f

# Số lượng biến độc lập

k = x\_train.shape[1]

# Số lượng quan sát

n = len(y\_train)

# Độ tự do của mô hình

df\_model = k - 1

# Độ tự do của sai số

df\_residual = n - k

# Giá trị của F-statistic

f\_statistic = (lm.score(x\_train, y\_train) / df\_model) / ((1 - lm.score(x\_train, y\_train)) / df\_residual)

# Giá trị p-value

p\_value = f.cdf(f\_statistic, df\_model, df\_residual)

print(f'Giá trị F-statistic: {f\_statistic}')

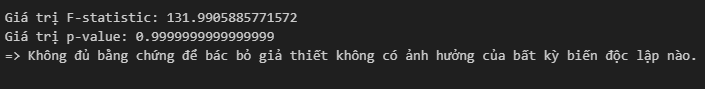
print(f'Giá trị p-value: {p\_value}')

if p\_value < 0.05:

    print('=> Có đủ bằng chứng để bác bỏ giả thiết không có ảnh hưởng của bất kỳ biến độc lập nào.')

else:

    print('=> Không đủ bằng chứng để bác bỏ giả thiết không có ảnh hưởng của bất kỳ biến độc lập nào.')



Hình 21: Kết quả kiểm định F - test

### Biểu đồ Scatter plot

**Biểu đồ giữa giá trị thực tế và giá trị dự đoán**

# Biểu đồ scatter plot giữa giá trị thực tế và giá trị dự đoán

plt.figure(figsize=(10, 6))

plt.scatter(result\_compare['price'], result\_compare['Price\_predict'], color='blue', label='Dự đoán')

plt.plot(result\_compare['price'], result\_compare['price'], color='red', linestyle='--', label='Giá trị thực tế')

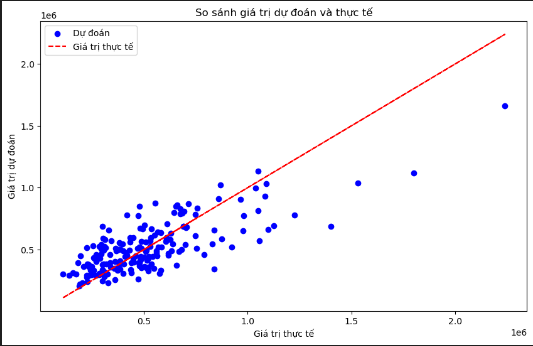
plt.xlabel('Giá trị thực tế')

plt.ylabel('Giá trị dự đoán')

plt.title('So sánh giá trị dự đoán và thực tế')

plt.legend()

plt.show()



Hình 22: Biểu đồ so sánh giá trị dự đoán và thực tế

* Trục x Biểu diễn giá trị thực tế của các mẫu trong tập test. Mỗi điểm trên trục x biểu thị giá trị thực tế của một mẫu.
* Trục y Biểu diễn giá trị dự đoán tương ứng với các mẫu trong tập test. Mỗi điểm trên trục y biểu thị giá trị được mô hình dự đoán cho một mẫu.
* Đường đứng đỏ (Red dashed line): Biểu diễn giá trị zero trên trục y. Nó giúp chúng ta dễ dàng nhận biết các điểm dữ liệu có sai số âm và dương.
* Các điểm dữ liệu được biểu diễn bằng các chấm màu xanh trên biểu đồ. Mỗi điểm biểu thị một mẫu trong tập test và sai số tương ứng của nó

Từ kết quả biểu đồ có thể thấy các điểm dữ liệu được biểu diễn bằng các chấm màu xanh. Đường đỏ nét đứt biểu diễn xu hướng của giá trị thực tế. Hầu hết các điểm dữ liệu tập trung gần đường đỏ, cho thấy giá trị dự đoán khá chính xác so với giá trị thực tế. Tuy nhiên, cũng có một số điểm dữ liệu nằm xa đường đỏ, cho thấy giá trị dự đoán không chính xác cho những điểm dữ liệu đó.

**Biểu đồ dự đoán và sai số**

# Biểu đồ dự đoán và sai số

plt.figure(figsize=(10, 6))

plt.scatter(result\_compare['price'], result\_compare['Deviation'], color='green')

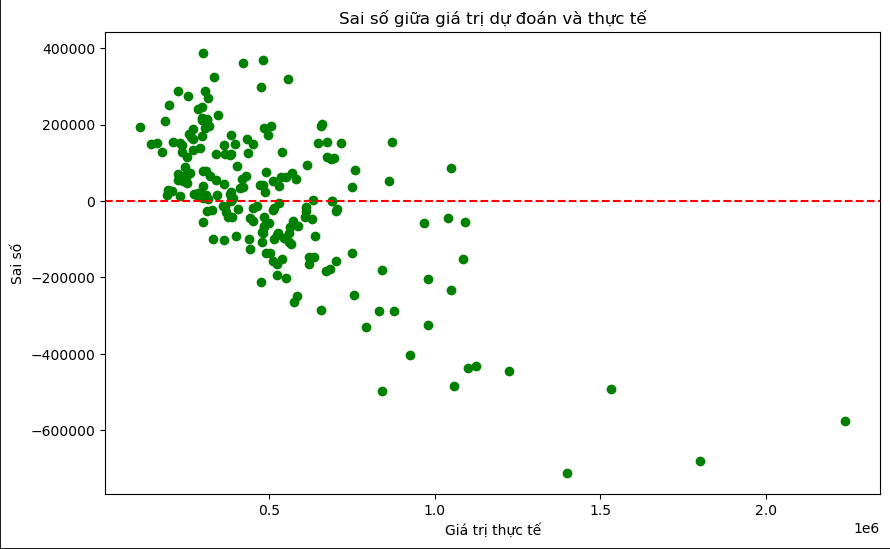
plt.axhline(y=0, color='red', linestyle='--')

plt.xlabel('Giá trị thực tế')

plt.ylabel('Sai số')

plt.title('Sai số giữa giá trị dự đoán và thực tế')

plt.show()



Hình 23: Biểu đồ dự đoán và sai số

* Trục x : Biểu diễn giá trị thực tế của các mẫu trong tập test. Mỗi điểm trên trục x biểu thị giá trị thực tế của một mẫu.
* Trục y : Biểu diễn sai số giữa giá trị dự đoán và giá trị thực tế. Mỗi điểm trên trục y biểu thị sai số tương ứng của một mẫu.
* Đường đỏ nét đứt ngang tại y=0: Đường này biểu diễn các điểm không có sự sai lệch giữa giá trị dự đoán và giá trị thực tế, nơi mà sai số bằng 0. Điều này có nghĩa là các điểm dữ liệu nằm trên đường này có dự đoán chính xác và không có sai số.
* Các điểm dữ liệu (Data points) được biểu diễn bằng các chấm màu xanh trên biểu đồ. Mỗi điểm biểu thị một mẫu trong tập test và sai số tương ứng của nó

Từ kết quả biểu đồ có thể thấy các điểm dữ liệu được biểu diễn bằng các chấm xanh lá cây tập trung xung quanh đường đỏ điều này cho thấy mô hình có khả năng dự đoán khá chính xác với một số mẫu trong tập test. Tuy nhiên cũng có khá nhiều chấm xanh bị phân tán ra dần càng xa đường đỏ, đặc biệt là khi giá trị thực tế tăng. Điều này cho thấy mô hình có thể gặp khó khăn trong việc dự đoán chính xác với các mẩu có giá trị cao hơn.

# Kết luận

Nhìn chung mô hình cho ra kết quả có độ chính xác ở **mức trung bình** trên toàn bộ phạm vi giá trị thực tế. Có những mẫu mà mô hình dự đoán gần như chính xác, trong khi số ít những mẫu khác lại đưa ra kết quả sai số lớn. Các sai số lớn đối với một số mẫu có thể chỉ ra rằng mô hình cần được cải thiện hoặc điều chỉnh thêm để dự đoán chính xác hơn.

# Tài liệu tham khảo

<https://maths.uel.edu.vn/Resources/Docs/SubDomain/maths/TaiLieuHocTap/ToanUngDung/m_hnh_hi_quy_n.html>

Công cụ hỗ trợ: <https://chat.openai.com/>

Video báo cáo: <https://drive.google.com/file/d/1sE83mTZNbTCOM4f80H_GZ0v_bEcVe498/view?usp=sharing>