**TRƯỜNG ĐẠI HỌC KỸ THUẬT CÔNG NGHIỆP**

**KHOA ĐIỆN TỬ**



**BÁO CÁO TIỂU LUẬN**

**Môn học: TRÍ TUỆ NHÂN TẠO**

**Đề tài: Dự đoán giá nhà bằng mô hình hồi quy tuyến tính (Linear Regression)**

**Giáo viên hướng dẫn: TS. Nguyễn Tuấn Linh**

**Sinh viên thực hiện: Trần Thị Duyên**

**Mã số sinh viên: K185480106004**

**Lớp: K54-KMT.01**

Thái Nguyên – 06/2022

**MỤC LỤC**

[CHƯƠNG 1. GIỚI THIỆU ĐỀ TÀI 1](#_Toc106002477)

[1.1. Đặt vấn đề 1](#_Toc106002478)

[1.2. Dữ liệu sử dụng để huấn luyện mô hình 1](#_Toc106002479)

[1.3. Lựa chọn thuật toán 1](#_Toc106002480)

[CHƯƠNG 2. PHÂN TÍCH VÀ TIỀN XỬ LÝ DỮ LIỆU 2](#_Toc106002481)

[CHƯƠNG 3. THUẬT TOÁN 6](#_Toc106002482)

[CHƯƠNG 4. CÀI ĐẶT VÀ KIỂM THỬ 8](#_Toc106002483)

[4.1. Xây dựng mô hình dự đoán bằng thư viện Scikit-Learn 8](#_Toc106002484)

[4.2. Dự đoán và đánh giá mô hình 9](#_Toc106002485)

[CHƯƠNG 5. KẾT LUẬN 18](#_Toc106002486)

[5.1. Kết quả đạt được 18](#_Toc106002487)

[5.2. Hướng phát triển 18](#_Toc106002488)

[Tài liệu tham khảo 19](#_Toc106002489)

[Phụ lục 19](#_Toc106002490)

# LỜI MỞ ĐẦU

Ngày nay, với khả năng tính toán và tốc độ vượt trội, AI (trí tuệ nhân tạo) đang âm thầm thay đổi ngành bất động sản, đem lại lợi ích cho các công ty môi giới, người bán nhà lẫn người mua nhà.

Theo CNBC, những công ty như Compass, Zillow, LoanSnap hiện dùng AI để tìm cho khách hàng những khoản thế chấp hay ngôi nhà vừa ý họ. Đối với các đại lý bất động sản, AI chính là yếu tố thay đổi cuộc chơi.

Hầu hết dữ liệu bất động sản đều được công khai, từ hồ sơ đất đai cho đến giấy tờ chủ quyền, giá cả và tiền thế chấp. Khó khăn duy nhất là chúng ta phải đến các văn phòng địa phương để lấy thông tin. Giờ đây, thuật toán máy tính có thể duyệt qua hàng triệu tài liệu trong vòng vài giây, xem giá trị tài sản, mức nợ, các dự án tu sửa nhà, thậm chí cả thông tin cá nhân của chủ nhà.

Chính vì vậy, nhằm áp dụng những kiến thức đã học của môn Trí tuệ nhân tạo và nâng cao khả năng phân tích vào bài toán thực tế. Trong bài tiểu luận này, chúng ta sẽ cùng tiến hành xây dựng và đào tạo mô hình “Dự đoán giá nhà bằng mô hình hổi quy tuyến tính(Linear Regression)”. Nội dung chính chúng ta sẽ tìm hiểu gồm 5 chương:

* Chương 1: Giới thiệu đề tài
* Chương 2: Phân tích và tiền xử lý dữ liệu
* Chương 3: Thuật toán
* Chương 4: Cài đặt và kiểm thử
* Chương 5: Kết luận

Em xin trân thành cảm ơn thầy Nguyễn Tuấn Linh đã giúp đỡ, truyền đạt cho em những kiến thức thú vị về AI; giúp em hoàn thành bài tiểu luận. Do kiến thức và thời gian giới hạn nên em còn nhiều thiếu sót, em rất mong sẽ nhận được góp ý của thầy (cô) để có thể hoàn thiện hơn.

|  |
| --- |
| Sinh viên thực hiện  *Duyên*  *Trần Thị Duyên* |

# CHƯƠNG 1. GIỚI THIỆU ĐỀ TÀI

* 1. **Đặt vấn đề**

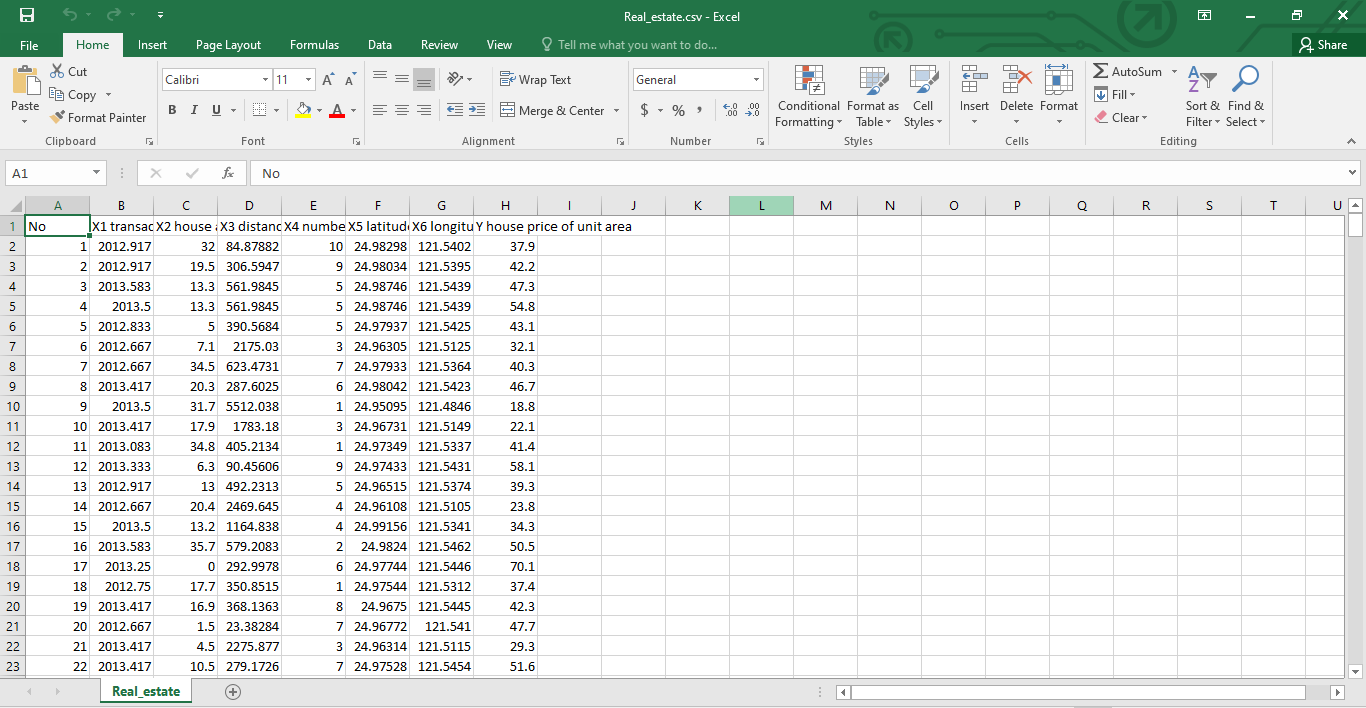
Giả sử có một tập dữ liệu gồm các thuộc tính của bất động sản và giá của chúng. Các thuộc tính như: số phòng ngủ, số phòng tắm, vị trí địa lý, năm xây dựng...và giá bán

Bài toán đặt ra: Tìm một công thức nào đó để từ một bất động sản bất kì, cho biết một vài thông tin như số phòng ngủ, vị trí địa lý... có thể tính toán ra được giá bán của căn nhà.

* 1. **Dữ liệu sử dụng để huấn luyện mô hình**

Tập dữ liệu bất động sản được lấy từ nguồn dữ liệu của Kangle tại link: <https://www.kaggle.com/datasets/quantbruce/real-estate-price-prediction?resource=download>

Tập dữ liệu có tên: Real estate.csv



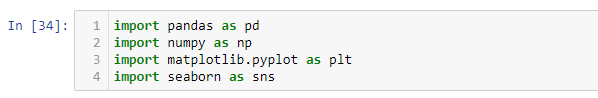
Hình 1. 1. Tập dữ liệu Real estate.csv

* 1. **Lựa chọn thuật toán**

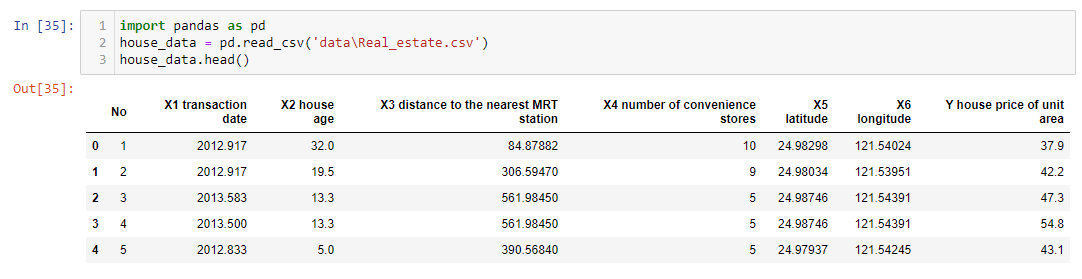
Chúng ta sẽ tiến hành xây dựng mô hình dự đoán giá nhà thuật toán hồi quy tuyến tính (Linear Regression)

# CHƯƠNG 2. PHÂN TÍCH VÀ TIỀN XỬ LÝ DỮ LIỆU

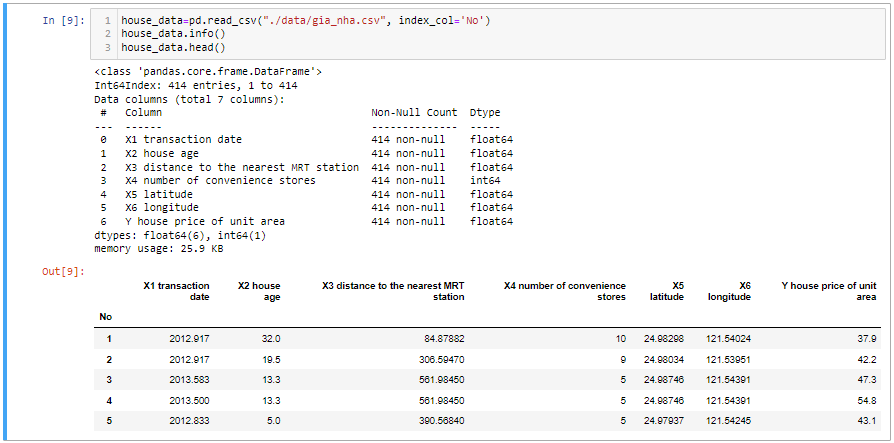
Để tạo ra một mô hình dự đoán tốt, việc rất quan trọng cần làm đó là phân tích và tiền xử lý dữ liệu, ta phải tìm hiểu thông tin về dataset đó, ý nghĩa nội dung các cột, dòng. Thường là các thông tin này được đính kèm bổ sung khi ta được cung cấp dữ liệu từ 1 công ty, doanh nghiệp hoặc các nguồn cung cấp khác như kaggle. Đầu tiên, chúng ta import thư viện cần dùng:



Trên đây là các thư viện thường xuyên sử dụng để xử lý, phân tích và trực quan hóa dữ liệu. Tiếp đến, tiến hành đọc file dataset đã lưu từ trang <https://www.kaggle.com/datasets/quantbruce/real-estate-price-prediction?resource=download> và quan sát dữ liệu:



Nhận thấy, dữ liệu gốc chúng ta tải được bao gồm 8 cột. Tuy nhiên, cột *No* và cột *Index column* của Dataframe tương đương nhau nên ta đặt luôn cột *Index column* bằng cột *No:*

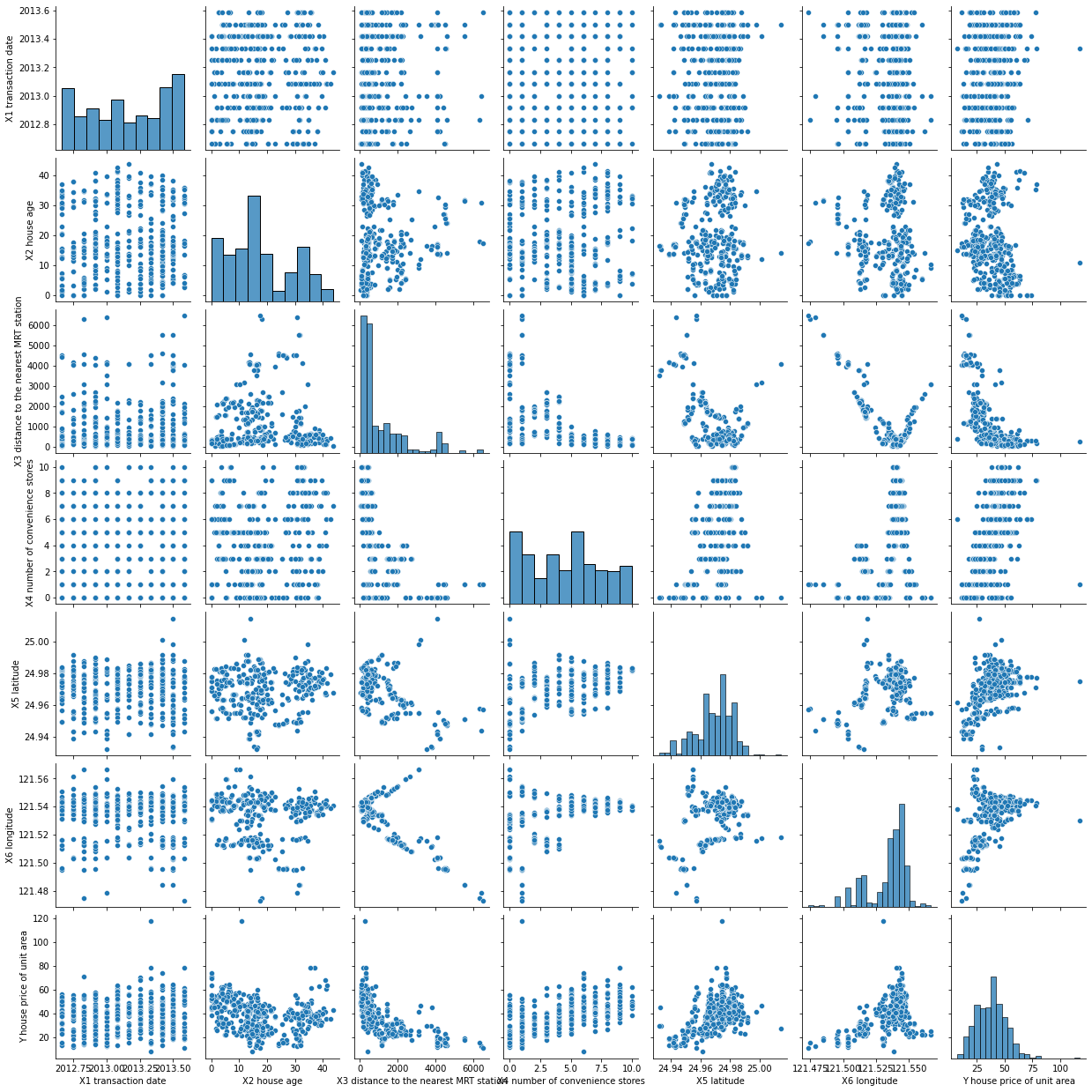


Như vậy, dữ liệu dữ liệu của chúng ta sử dụng sẽ có 414 dòng tương ứng với dữ liệu 414 ngôi nhà được bán và 7 cột. Ý nghĩa từng cột:

* *X1 transaction date:* Ngày giao dịch
* *X2 house age:* Tuổi của căn nhà
* *X3 distance to the nearest MRT station:* Khoảng cách đến ga tàu điện ngầm gần nhất
* *X4 number of convenience stores:* Số cửa hàng tiện lợi gần đó
* *X5 latitude:* Vĩ độ
* *X6 longitude:* Kinh độ
* *Y house price of unit area:* Giá bán của căn nhà

Sau đây, chúng ta sẽ tiến hành trực quan hóa dữ liệu phân bố bằng Seaborn. Để nhìn tổng quan dữ liệu chúng ta sẽ sử dụng biểu đồ pair plots:

*sns.pairplot(house\_data)*

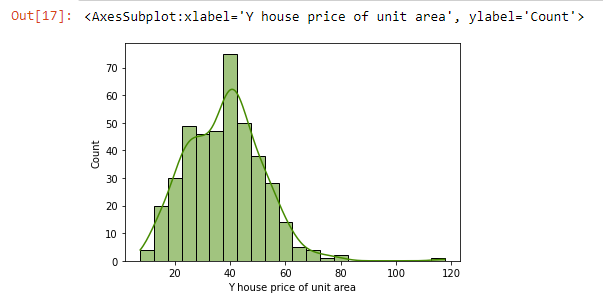
****

Hình 2. 1. Biểu đồ phân bố Pairplot-Seaborn

Về tổng quan các cột, ta thấy cột Price có kiểu phân tán theo mô hình tuyến tính, dựa trên thông tin này, ta xây dựng mô hình máy học hồi quy tuyến tính để dự đoán nó dựa trên giá trị các cột khác.

Tiếp theo, chúng ta cùng xem biểu đồ phân phối giá nhà:

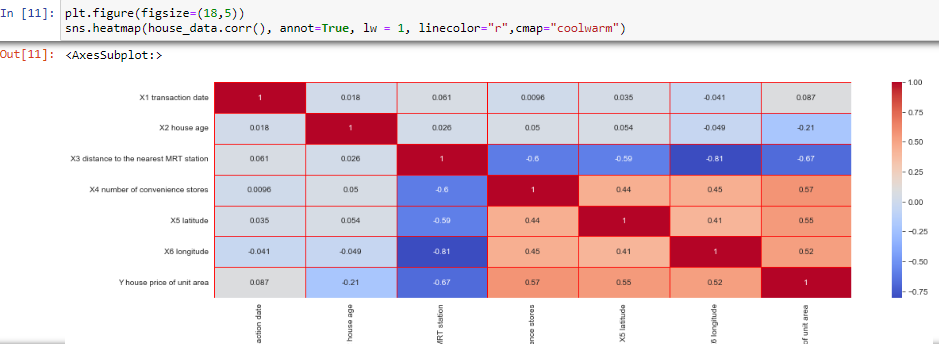




Hình 2. 2. Biểu đồ phân phối giá nhà

Dễ thấy các căn nhà đã bán thường tập trung ở mức giá $20 đến $60 trên một đơn vị diện tích và nhiều nhất dao động trong khoảng $40.

Kiểm tra độ tương quan giữa các cột bằng bản đồ nhiệt:



Hình 2. 3. Bản đồ nhiệt

Qua đó, ta phân tích được các cột có giá trị tương quan như như thế nào với nhau. Về cơ bản, cột giá (price) có chút tương quan với các cột còn lại nhiều nhất, chứng tỏ các yếu tố đó có tác động ít nhiều lên giá nhà.

Đến đây, chúng ta đã phân tích sơ qua về bảng dữ liệu, đây là một dataset tương đối đơn giản, các hàng và cột không có giá trị null nên ta không cần xử lý.

# CHƯƠNG 3. THUẬT TOÁN

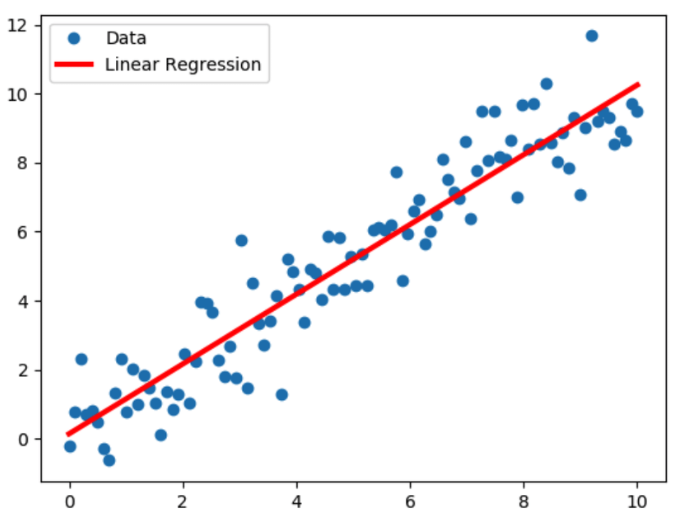
Như đã nói ở chương 1, thuật toán được sử dụng để đào tạo model là hồi quy tuyến tính (Linear Regression). Thuật toán này thuộc nhóm Supervised learning ( Học có giám sát ). Tuy là một phương pháp rất đơn giản nhưng Hồi quy tuyến tính đã được chứng minh được tính hữu ích cho một số lượng lớn các tình huống. Nó có thể được sử dụng cho các trường hợp chúng ta muốn dự đoán một số lượng liên tục. Ví dụ, dự đoán giao thông ở một cửa hàng bán lẻ, dự đoán thời gian người dùng dừng lại một trang nào đó hoặc số trang đã truy cập vào một website nào đó v.v... Hồi quy tuyến tính là thuật toán tìm ra phương trình tuyến tính dựa trên tập dữ liệu quan hệ giữa X (dữ liệu đầu vào) và Y (dữ liệu đầu ra). X là biến giải thích và Y là biến phụ thuộc.

Hồi quy tuyến tính được chia làm 2 loại: hồi quy tuyến tính đơn giản và hồi quy tuyến tính đa biến. Về cơ bản không có sự khác biệt giữa hồi quy tuyến tính 'giản đơn' và 'đa biến'. Cả hai đều làm việc tuân theo nguyên tắc OLS và thuật toán để có được đường hồi quy tối ưu nhất cũng tương tự.

Cụ thể với tập dữ liệu Real estate.csv sẽ sử dụng hồi quy tuyến tính đa biến. phương trình hồi quy sẽ có một hình dạng như sau:

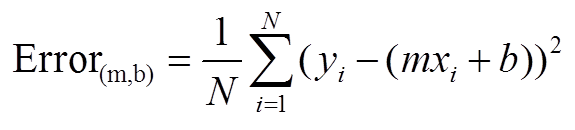
**(PT 1)**

Chúng ta đã phân tích và trực quan hóa mối quan hệ giữa các biến. Từ đó tạo ra mô hình quan hệ giữa các biến đầu vào X là ngày giao dịch, tuổi căn nhà, khoảng cách đến ga tàu, kinh độ, vĩ độ với dữ liệu đầu ra Y là giá nhà bằng mô hình hồi quy tuyến tính.



Hình 3. 1. Ảnh minh họa hồi quy tuyến tính

Về cơ bản, việc huấn luyện theo mô hình tuyến tính bản chất là đi tìm các giá trị m và b sao cho cực tiểu hóa hàm lỗi sau:



Ở trên, chúng ta đã biết rằng hồi quy tuyến tính là một kỹ thuật phổ biến và ai cũng có thể thấy các phương trình toán học của hồi quy tuyến tính. Nhưng làm thế nào để thực hiện một hồi quy tuyến tính trong Python? Có một số cách để có thể làm điều đó, ta có thể thực hiện hồi quy tuyến tính bằng cách sử dụng các mô hình thống kê, numpy, scipy và sckit learn. Nhưng trong bài này chúng ta sẽ sử dụng sckit-learn để thực hiện hồi quy tuyến tính.

Scikit-learn là một module Python mạnh mẽ cho việc học máy. Nó chứa hàm cho hồi quy, phân loại, phân cụm, lựa chọn mô hình và giảm kích chiều. Chúng ta sẽ khám phá module sklearn.linear\_model có chứa "các method để thực hiện hồi quy, trong đó giá trị mục tiêu sẽ là sự kết hợp tuyến tính của các biến đầu vào".Chúng ta sử dụng gói thư viện Scikit-learn của Python để làm việc này rất đơn giản như sau:

def linearRegressionModel(X\_train, Y\_train, X\_test, Y\_test):

linear = linear\_model.LinearRegression()

# Training process

linear.fit(X\_train, Y\_train)

# Evaluating the model

score\_trained = linear.score(X\_test, Y\_test)

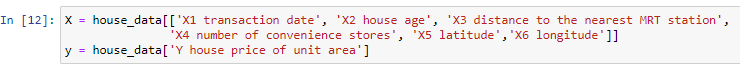
return score\_trained

Hàm trên chạy mô hình hồi quy tuyến tính trên tập dữ liệu huấn luyện gồm X\_train đại diện cho tập các thuộc tính của bất động sản và Y\_train đại diện cho giá của nhà. Hàm trả về một giá trị đánh giá điểm của mô hình khi chạy trên tập kiểm tra. Về cơ bản, chúng ta có thể sử dụng điểm này để so sánh giữa các phương pháp hồi quy khác sẽ trình bày ở phần tiếp theo. Có nghĩa là điểm càng tiến gần đến 1 thì mô hình của chúng ta càng tốt.

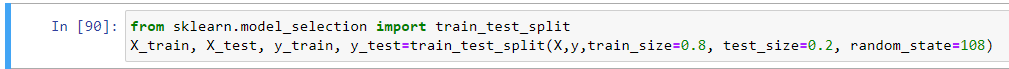
# CHƯƠNG 4. CÀI ĐẶT VÀ KIỂM THỬ

## 4.1. Xây dựng mô hình dự đoán bằng thư viện Scikit-Learn

Bây giờ chúng ta hãy bắt đầu đào tạo mô hình hồi quy. Trước tiên, chúng ta sẽ cần tách dữ liệu của mình thành tập dữ liệu X chứa các tính năng cần đào tạo (các biến độc lập) và tập y với biến mục tiêu (biến phụ thuộc), trong trường hợp này là cột *Y house price of unit area* *(Giá nhà)*.

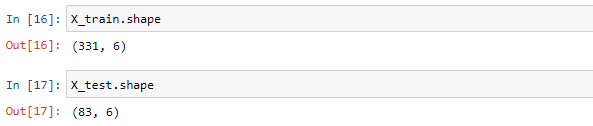


Tiếp theo, ta cần tách các biến thành giá trị train và test, hai giá trị này chúng ta sẽ luôn gặp và sử dụng trong quá trình xây dựng mô hình máy học. Đầu tiên, từ thư viện Scikit – Learn model\_selection ta import train\_test\_split, phương thức này giúp ta tạo ra dữ liệu dùng để train và test mô hình. Tiếp đến chia X,y thành 4 tập dữ liệu gồm X\_train, y\_train và X\_test, y\_test

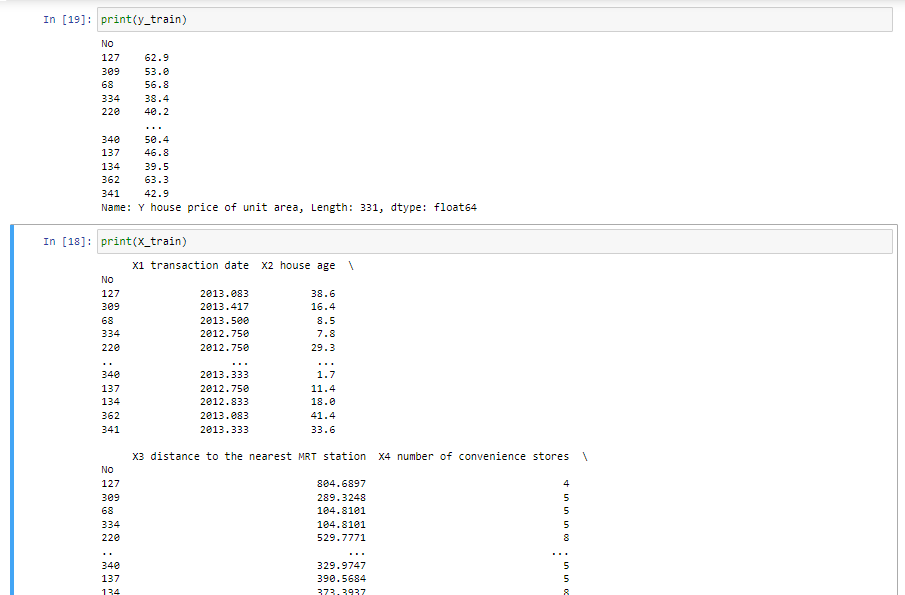


Với đối số truyền vào là giá trị X, y ta đã lấy từ dữ liệu bên trên, test\_size trả về cho ta phần trăm dữ liệu được chia, ở đây ta lấy 0.2 tương ứng với dữ liệu được chia thành 20% giá trị là test, còn lại là dữ liệu train. random\_state bằng một số tương ứng nào đó để đảm bảo mỗi lần ta chạy lại mô hình, giá trị phân tách ngẫu nhiên nhận được là giống nhau.

Sau đây là dữ liệu mà chúng ta vừa chia được:



Với bộ dữ liệu tổng là 414 dòng, chúng ta lấy 20% dữ liệu vào bộ test nên ở đây X\_test có 83 dòng, còn lại X\_train có 331 dòng.



Bộ dữ liệu X đã chứa dữ liệu các cột từ X1 đến X6 còn bộ y chứ dữ liệu Y giá nhà.

Bước tiếp theo chúng ta sẽ tạo model và training Linear Regression. Từ thư viện Scikit – Learn , linear\_model import module LinearRegression



Tiến hành train dữ liệu bằng phương thức fit():

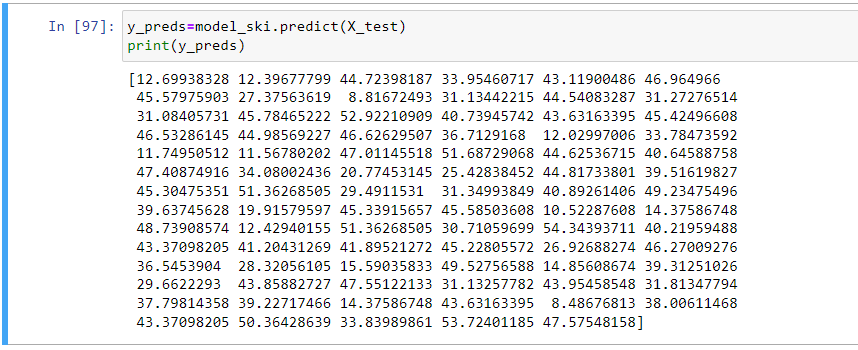


Kết quả trả về là một hàm LinearRegression(), chứng tỏ mô hình đã train xong.

## 4.2. Dự đoán và đánh giá mô hình

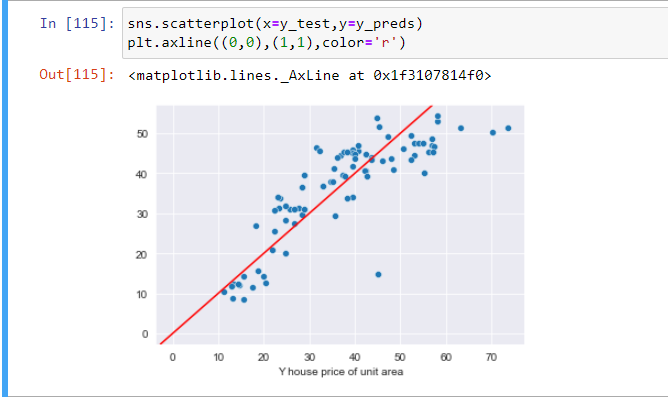
Để dự đoán và kiểm tra mô hình, ta sử dụng dữ liệu test bên trên mà ta đã tách ra. Trong đó, X\_test là các tính năng mà mô hình chưa biết, y\_test là kết quả biết trước để ta so sánh với kết quả dự đoán từ X\_test.

Chúng ta sử dụng phương thức predict() truyền đối số X\_test để lấy kết quả dự đoán rồi lưu vào *y\_preds*



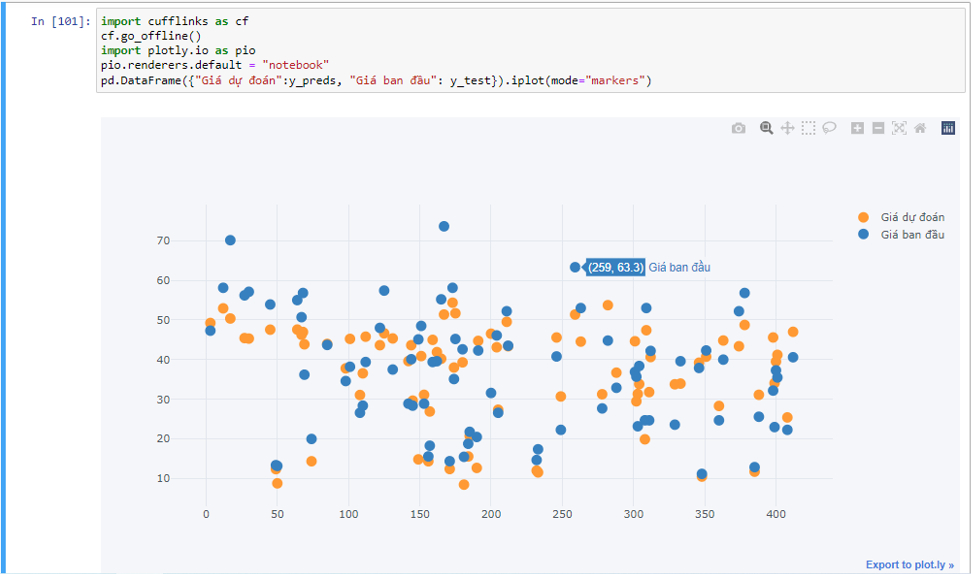
Kết quả dự đoán bên trên là một mảng trong numpy chứa kết quả dự đoán từ giá trị X\_test, để kiểm tra kết quả dự đoán (predictions) và kết quả ban đầu (y\_test) xem mô hình ta như thế nào? Ta có thể trực quan hóa quan sát bằng biểu đồ.

Sử dụng biểu đồ scatterplot của thư viện matplotlib truyền vào 2 giá trị trên vào để quan sát độ chênh lệch giữa giá nhà thực tế và giá nhà dự đoán



Hình 4. 1. Biểu đồ thể hiện chênh lệch giá bán dự đoán và thực tế

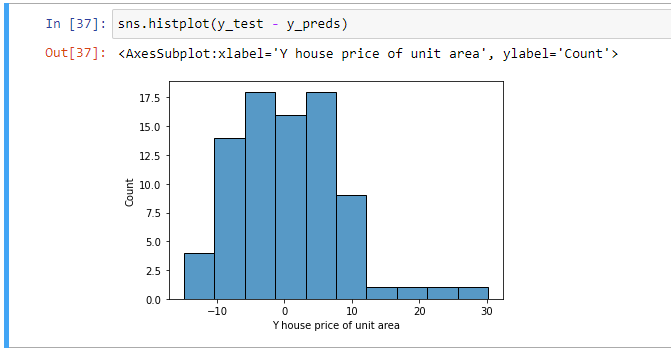
Từ biểu đồ chúng ta dễ thấy đặc trưng của hồi quy tuyến tính, giá bán dự đoán là đường chéo màu đỏ, các chấm xanh xung quanh là giá nhà bán thực tế. Có thể thấy, dữ liệu có tồn tại một vài điểm nhiễu. Tuy nhiên, để quan sát cụ thể giá bán và giá dự đoán hơn, chúng ta sử dụng thêm biểu đồ động cufflinks để trực quan:



Hình 4. 2. Biểu đồ động cufflinks

Nhìn trực quan bên trên ta thấy giữa giá dự đoán (màu cam) và giá ban đầu (màu xanh dương) có 1 sự chênh lệnh. Tùy theo độ chính xác mô hình, nếu độ chính xác mô hình càng cao thì độ chênh lệch các điểm trên biểu đồ càng ít lại.

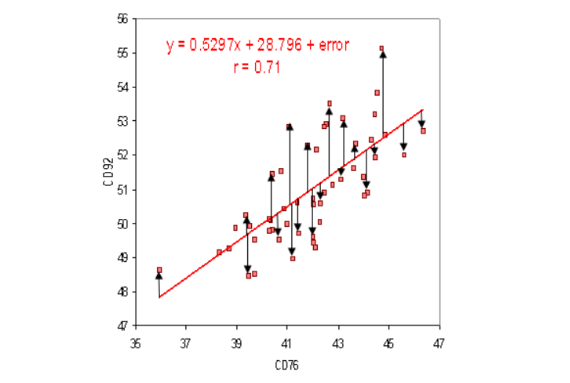
Tiếp theo, ta cùng sử dụng đồ thị distplot() trong Seaborn để tìm hiểu sự phân phối của độ chênh lệch này.



Hình 4. 3. Biểu đồ phân phối mức giá chênh lệch

Nhìn vào biểu đồ trên, ta thấy giá trị chênh lệch giữa giá dự đoán (predictions) và giá trị thực tế ban đầu (y\_test), phân bố tập trung ở $0 và trong khoảng $10.000, chứng tỏ mô hình của chúng ta có độ chính xác tương đối cao và hợp lý.

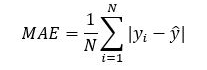
Nhưng nhìn vào biểu đồ trên, nó không cho ta biết được các giá trị cụ thể mà chỉ dựa trên phán đoán trực quan tổng thể. Bây giờ, ta hãy tìm số liệu cụ thể để có cái nhìn chính xác hơn dựa trên số liệu phân tích. Ta dựa vào các chỉ số đánh giá hồi quy.



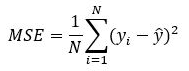
Hình 4. 4. Ảnh minh họa sai số của mô hình hồi quy

Chúng ta có các chỉ số đánh giá hồi quy để có số liệu chính xác:

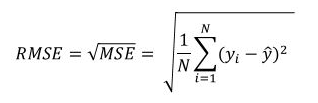
* Mean Absolute Error (MAE): MAE là một phương pháp đo lường sự khác biệt (độ chênh lệch giá trị) giữa hai biến liên tục. Giả sử rằng X và Y là hai biến liên tục thể hiện kết quả dự đoán của mô hình và kết quả thực tế, đây là chỉ số dễ hiểu nhất, vì đó là giá trị chênh lệch trung bình tuyệt đối và được xác định bằng công thức:



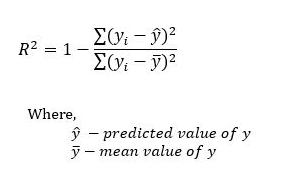
* Mean Squared Error (MSE): là giá trị trung bình của bình phương sai số (Hàm mất mát), là sự khác biệt giữa các giá trị được mô hình dự đoán và gía trị thực. MSE cũng được gọi là một hàm rủi ro, tương ứng với giá trị kỳ vọng của sự mất mát sai số bình phương hoặc mất mát bậc hai chỉ số này phổ biến hơn chỉ số MAE bên trên, được xác định bằng công thức:



* Root Mean Squared Error (RMSE): là căn bậc hai của giá trị trung bình của các sai số bình phương (MSE). Thông thường, ta thường dùng chỉ số này để xác định giá trị chênh lệch trung bình giữa giá dự đoán và giá trị test ban đầu, được xác định bằng công thức:



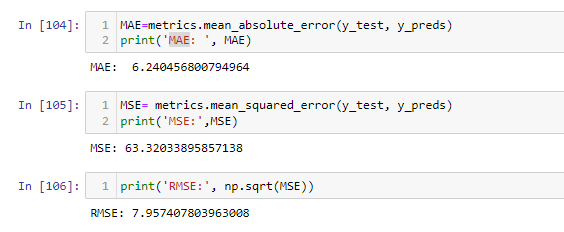
* R bình phương (Hệ số xác định) đại diện cho hệ số về mức độ phù hợp của các giá trị so với các giá trị ban đầu. Giá trị từ 0 đến 1 được hiểu là tỷ lệ phần trăm. Giá trị càng cao thì mô hình càng tốt.



Qua các công thức bên trên ta thấy rườm rà khó hiểu, nhưng mọi thứ đơn giản hơn khi ta sử dụng thư viện Scikit – Learn để tìm ra các chỉ số này. Để bắt đầu, chúng ta import metrics từ sklearn:

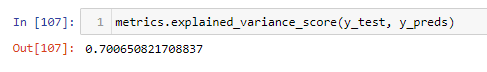


Tính các chỉ số đnáh giá hồi quy và in ra để quan sát



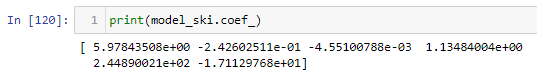
Từ chỉ số RMSE, cho thấy giá trị chênh lệch trung bình của giá dự đoán từ mô hình và giá trị thực tế xấp xỉ 8 nghìn USD.

Giờ ta kiểm tra độ chính xác mô hình trên dựa trên phương sai



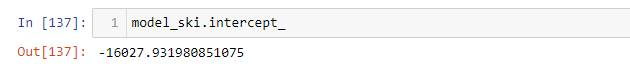
Giá trị trả về tốt nhất cho mô hình là 1, vậy với mô hình trên, giá trị trả về đạt 0.7, tương ứng với 70% hiệu quả của mô hình đào tạo.

Để đánh giá sực tác động của các tính năng ( các biến độc lập) lên kết quả đầu ra (biến phụ thuộc), ta sử dụng hệ số Coeff. Hệ số này cho ta biết khi giá trị biến độc lập thay đổi 1 đơn vị, thì giá trị đầu ra sẽ thay đổi như thế nào.



Những giá trị trên chính là W1,W2..W6 mà trong phương trình hồi quy

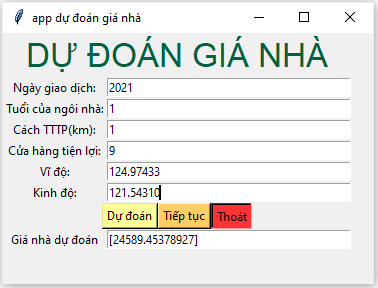
[**(PT 1)**](#_Toc105864384) chúng ta cần tìm. Để xem W0 chúng ta làm như sau:



Suy ra, phương trình chúng ta cần tìm để dự đoán nhà sẽ là:

**(PT 2)**

Như vậy, chúng ta đã cơ bản xong phần xây dựng và đánh giá mô hình máy học machine learning sử dụng thuật toán linear regression. Để kết thúc chương cài đặt và kiểm thử này chúng ta cùng xây dựng một app nhỏ dự đoán giá nhà có dạng như sau:



Hình 4. 5. Giao diện App dự đoán giá nhà

*Source code:*

from tkinter import \*

#Hàm xóa các ô nhập về rỗng

def tiepAction():

strNgay.set("")

strTuoi.set("")

strTTTP.set("")

strSoCuaHang.set("")

strViDo.set("")

strKinhDo.set("")

stringKQ.set("")

#Hàm dự đoán

def dudoanAction():

ngay=float(strNgay.get())

tuoi=float(strTuoi.get())

tttp=float(strTTTP.get())

so\_cua\_hang=int(strSoCuaHang.get())

vi\_do=float(strViDo.get())

kinh\_do=float(strKinhDo.get())

kq=model\_ski.predict([[ngay,tuoi,tttp,so\_cua\_hang,vi\_do,kinh\_do]])

stringKQ.set(kq)

root=Tk()

strNgay=StringVar()

strTuoi=StringVar()

strTTTP=StringVar()

strSoCuaHang=StringVar()

strViDo=StringVar()

strKinhDo=StringVar()

stringKQ=StringVar()

root.title("app dự đoán giá nhà")

root.minsize(height=250,width=350)

root.resizable(height=True,width=True)

Label(root,text="DỰ ĐOÁN GIÁ NHÀ", fg="#006241",anchor='center',font=("tohama",25)).grid(row=0,columnspan=2)

Label(root,text="Ngày giao dịch:",anchor="n").grid(row=1,column=0)

Entry(root,width=40,textvariable=strNgay).grid(row=1,column=1)

Label(root,text="Tuổi của ngôi nhà:").grid(row=2,column=0)

Entry(root,width=40,textvariable=strTuoi).grid(row=2,column=1)

Label(root,text="Cách TTTP(km):",anchor="w").grid(row=3,column=0)

Entry(root,width=40,textvariable=strTTTP).grid(row=3,column=1)

Label(root,text="Cửa hàng tiện lợi:").grid(row=4,column=0)

Entry(root,width=40,textvariable=strSoCuaHang).grid(row=4,column=1)

Label(root,text="Vĩ độ:").grid(row=5,column=0)

Entry(root,width=40,textvariable=strViDo).grid(row=5,column=1)

Label(root,text="Kinh độ:").grid(row=6,column=0)

Entry(root,width=40,textvariable=strKinhDo).grid(row=6,column=1)

frameButton=Frame()

Button(frameButton,text="Dự đoán",bg="#FFFF99",command=dudoanAction).pack(side=LEFT)

Button(frameButton,text="Tiếp tục",bg="#FFCC66",command=tiepAction).pack(side=LEFT)

Button(frameButton,text="Thoát",relief ='sunken',bg="#FF3333",command=root.quit).pack(side=LEFT)

frameButton.grid(row=7,columnspan=2)

Label(root,text="Giá nhà dự đoán").grid(row=8,column=0)

Entry(root,width=40,textvariable=stringKQ).grid(row=8,column=1)

root.mainloop()

# CHƯƠNG 5. KẾT LUẬN

## 5.1. Kết quả đạt được

Về ưu điểm, xây dựng thành công mô hình dự đoán giá bán của một căn nhà từ các dữ liệu đầu vào nhất định và kết hợp xây dựng App dự đoán giá nhà. Trực quan hóa có sử dụng kết hợp cả biểu đồ tĩnh và động để dễ dàng quan sát dữ liệu.

Bên cạnh những ưu điểm kể trên, mô hình còn tồn tại một số hạn chế. Do tập dữ liệu chưa đủ lớn và chứa khá nhiều điểm nhiễu nên kết quả huấn luyện mới đạt 70% về độ chính xác, giao diện dự đoán giá nhà còn sơ sài.

## 5.2. Hướng phát triển

Mô hình dự đoán giá nhà sẽ có thể nâng cao độ chính xác bằng việc sưu tập thêm dữ liệu. Từ đó xây dựng App dự đoán giá nhà hiệu quả và có nhiều chức năng, giao diện bắt mắt hơn.

# Tài liệu tham khảo

[1] Website: <https://www.kaggle.com/datasets/quantbruce/real-estate-price-prediction/code>

[2] Website: <https://www.researchgate.net/figure/Linear-Regression-model-sample-illustration_fig3_340271573>

[3] Website: <https://www.researchgate.net/figure/Linear-Regression-model-sample-illustration_fig3_340271573>

[4] Website: <https://blog.vietnamlab.vn/thu-dung-plotly-de-data-visualization-voi-python/>

[5] Website: <https://www.datatechnotes.com/2019/02/regression-model-accuracy-mae-mse-rmse.html>

# Phụ lục

[Hình 1. 1. Tập dữ liệu Real estate.csv 1](#_Toc105862696)

[Hình 2. 1. Biểu đồ phân bố Pairplot-Seaborn 4](#_Toc105862703)

[Hình 2. 2. Biểu đồ phân phối giá nhà 5](#_Toc105862704)

[Hình 2. 3. Bản đồ nhiệt 5](#_Toc105862705)

[Hình 3. 1. Ảnh minh họa hồi quy tuyến tính 7](#_Toc105862709)

[Hình 4. 1. Biểu đồ thể hiện chênh lệch giá bán dự đoán và thực tế 11](#_Toc105862713)

[Hình 4. 2. Biểu đồ động cufflinks 11](#_Toc105862714)

[Hình 4. 3. Biểu đồ phân phối mức giá chênh lệch 12](#_Toc105862715)

[Hình 4. 4. Ảnh minh họa sai số của mô hình hồi quy 13](#_Toc105862716)

[Hình 4. 5. Giao diện App dự đoán giá nhà 15](#_Toc105862717)