**TRƯỜNG ĐẠI HỌC CÔNG NGHIỆP HÀ NỘI**

**KHOA CÔNG NGHỆ THÔNG TIN**

---

# BÁO CÁO BÀI TẬP LỚN

## HỌC PHẦN: Trí tuệ nhân tạo

## ĐỀ TÀI: DỰ ĐOÁN GIÁ NHÀ BẰNG MÔ HÌNH HỒI QUY TUYẾN TÍNH

**Giảng viên hướng dẫn: Ts. Trần Thanh Huân**

**Lớp: 20241IT6094005**

**Nhóm: 14**


| | | |
|---|---|---|
| **Thành viên:** | **Đinh Xuân Thảo** | **MSV: 2022604942** |
| | **Nguyễn Hùng Thanh** | **MSV: 2022601002** |
| | **Phạm Chí Thành** | **MSV: 2022605231** |
| | **Ngô Tiến Thành** | **MSV: 2022607498** |


**Hà Nội – Năm 2024**

BỘ CÔNG THƯƠNG
TRƯỜNG ĐẠI HỌC CÔNG NGHIỆP HÀ NỘI

CỘNG HÒA XÃ HỘI CHỦ NGHĨA VIỆT NAM
Độc lập – tự do – hạnh phúc
-------o0o-------

# PHIẾU GIAO ĐỀ TÀI
# BÁO CÁO THỰC NGHIỆM/THÍ NGHIỆM

Nhóm thực hiện: 14 1) Phạm Chí Thành
2) Nguyễn Hùng Thanh
3) Đinh Xuân Thảo
4) Ngô Tiến Thành

Lớp học phần: 20241IT6094005

Khóa: 17

Khoa Công nghệ thông tin

Ngành học: Công nghệ thông tin

Tên đề tài: Dự đoán giá nhà bằng mô hình hồi quy tuyến tính

Mục đích: Xây dựng được mô hình dự đoán giá nhà dựa trên các tài liệu, file đã thu nhập được.

Yêu cầu:

- Hoàn thành các nhiệm vụ của đề tài

- Sử dụng kỹ năng làm việc nhóm trong quá trình thực hiện báo cáo

- Trình bầy báo cáo đúng yêu cầu, sử dụng kỹ năng "Viết báo cáo" khi thực hiện đề tài

- Sử dụng kỹ năng viết tài liệu kỹ thuật và phi kỹ thuật trong phần mở đầu của báo cáo

Kết quả thu được: Bản báo cáo đề tài (bản cứng và bản mềm); sản phẩm đề tài

Ngày giao đề tai: 20/11/2024

Ngày hoàn thành: 20/12/2024

Giảng viên hướng dẫn: Trần Thanh Huân

Hà nội, Ngày 06 Tháng 11 Năm 2024

GIẢNG VIÊN

Trần Thanh Huân

# PHIẾU PHÂN CÔNG NHIỆM VỤ

Nhóm 14, gồm 4 thành viên

1) Phạm Chí Thành
2) Nguyễn Hùng Thanh
3) Đinh Xuân Thảo
4) Ngô Tiến Thành

| TT | Người thực hiện | Công việc | Kết quả đạt đươc | Nhận xét của GV |
|---|---|---|---|---|
| **Tuần 1** | | | | |
| **1** | Đinh Xuân Thảo | 1) Tìm hiểu yêu cầu của chương trình<br>2) Lập kế hoạch tổng thể cho bài tập lớn<br>Chuẩn bị tài liệu hướng dẫn và quản lý nhóm | Đã phân tích được yêu cầu cơ bản của chương trình.<br>2) Lập được kế hoạch tổng thể và chuẩn bị được tài liệu hướng dẫn | |
| **2** | Phạm Chí Thành | 1) Nghiên cứu về phương pháp hồi quy tuyến tính<br>Thu thập tài liệu liên quan | 1) Nắm được kiến thức về các phương pháp<br>Tìm được các tài liệu tham khảo từ nhiều nguồn khác nhau | |
| **3** | Nguyễn Hùng Thanh | 1) Nghiên cứu về các công cụ và thư viện<br>2) Đề xuất công cụ và thư viện phù hợp | 1) Đề xuất sử dụng GG - Colab cho việc phát triển, GitHub để quản lý mã nguồn, sử dụng thư viện numpy, pandas, matplotlib, warning, sklearn,XgBoost,… | |
| **4** | Ngô Tiến Thành | 1) Xây dựng cấu trúc cơ bản cho báo cáo | 1) Xác định được khung báo cáo gồm mục tiêu, phân công nhiệm vụ, giới thiệu, cơ sở lý thuyết, triển khai | |
| **Tuần 2** | | | | |

| | | | | |
|---|---|---|---|---|
| **1** | Đinh Xuân Thảo | 1) Thiết kế kiến trúc hệ thống và phân chia các module.<br>2) Hỗ trợ các thành viên khác | 1) Hoàn thành việc phân chia chương trình thành các thành phần nhỏ để dễ quản lý | |
| **2** | Phạm Chí Thành | 1)Bắt đầu lập trình logic AI cho chương trình | 1)Bước đầu triển khai các phân tích toán học, và các phương pháp hồi quy cần thiết | |
| **3** | Nguyễn Hùng Thanh | 1)Triển khai giao diện cơ bản, phối hợp với các thành viên tích hợp giao diện với logic | 1)Hoàn thành việc xây dựng và tích hợp giao diện cơ bản cho chương trình | |
| **4** | Ngô Tiến Thành | 1) Viết tài liệu về quy trình thiết kế giao diện và các công cụ sử dụng | 1) Hoàn thành phần mở đầu cho tài liệu | |
| **Tuần3** | | | | |
| **1** | Đinh Xuân Thảo | 1)Tích hợp toàn bộ giao diện với logic chương trình, sửa lỗi nếu có | 1)Tích hợp thành công giao diện vào chương trình | |
| **2** | Phạm Chí Thành | 1) Hoàn thiện và tối ưu thuật toán AI | 1) Hoàn thành mô hình bài toán , và dự trên các dữ liệu có sẵn để dự đoán được giá nhà | |
| **3** | Nguyễn Hùng Thanh | 1) Kiểm tra và sửa lỗi hiển thị trên mô hình | 1) Rà soát các lỗi còn tồn tại trên mô hình | |
| **4** | Ngô Tiến Thành | 1) Viết phần mô tả chương trình và cách sử dụng trong báo cáo | 1) Hoàn thiện chương cơ sở lý thuyết và chương triển khai chương trình trong báo cáo | |
| **Tuần 4** | | | | |
| **1** | Đinh Xuân Thảo | 1)Tổng hợp các phần báo cáo về mặt kỹ thuật | 1)Thêm các hình ảnh code trong chương trình để minh họa cho báo cáo | |

| | | | | |
|---|---|---|---|---|
| **2** | Phạm Chí Thành | 1) Thử nghiệm lần cuối cùng với toàn bộ chương trình<br>Hỗ trợ tổng hợp các phần báo | 1)Thông báo kết quả kiểm thử chương trình cho các thành viên trong nhóm | |
| **3** | Nguyễn Hùng Thanh | 1)Hoàn thiện báo cáo phần giao diện và bàn giao cho nhóm trưởng để tổng hợp | 1)Bàn giao phần giao diện trong báo cáo cho nhóm trưởng | |
| **4** | Ngô Tiến Thành | 1)Chuẩn bị slide để trình bày sản phẩm | 1)Hoàn thành việc tạo slide báo cáo sản phẩm cho nhóm | |

# LỜI NÓI ĐẦU

Ngày nay, công nghệ thông tin được ứng dụng trong hầu hết các lĩnh vực của đời sống. Bên cạnh những cách làm truyền thống cũng đã xuất hiện những kỹ thuật mới được áp dụng và đem lại hiệu quả đáng kể. Với lượng thông tin lớn, những bài toán có độ phức tạp cấp hàm mũ, vấn đề đặt ra là làm thế nào để phát hiện tri thức, đưa ra lời giải mà thời gian thực hiện có thể chấp nhận được. Một trong số các kỹ thuật được sử dụng đó chính là hồi quy tuyến tính. Kỹ thuật này đã được thực hiện và ứng dụng rất nhiều trong đời sống qua nhiều lĩnh vực như: kinh tế, giáo dục, y khoa, nông nghiệp, công nghiệp, Và cũng đã có rất nhiều sản phẩm nổi bật như robot thông minh, nhà thông minh, …

Và với đề tài "Dự đoán giá nhà bằng mô hình hồi quy tuyến tính" chúng ta sẽ đi tìm hiểu rõ hơn về một ứng dụng của học máy đó chính là dự đoán mô hình. Trong đề tài này nhóm em sẽ đi tìm hiểu về mô hình tuyến tính và cách thức để có thể áp dụng nó vào dự đoán giá nhà. Chúng em xin chân thành cảm ơn thầy Trần Thanh Huân đã tận tình hướng dẫn chúng em trong quá trình hoàn thiện đề tài này. Trong quá trình làm bài chúng em đã cố gắng để hoàn thiện bài làm một cách tốt nhất có thể, rất mong được nhận sự góp ý từ cô và các bạn để bài làm hoàn thiện hơn nữa. Chúng em xin chân thành cảm ơn!

# MỤC LỤC

# DANH MỤC HÌNH ẢNH

# CHƯƠNG 1: TỔNG QUAN VỀ ĐỀ TÀI

## 1.1. Tính cấp thiết của đề tài

Trong bối cảnh thị trường bất động sản không ngừng biến động, việc dự đoán giá nhà trở thành một nhiệm vụ quan trọng đối với các nhà đầu tư, phát triển dự án và các bên liên quan. Giá nhà không chỉ phản ánh tình trạng cung cầu của thị trường mà còn chịu ảnh hưởng của nhiều yếu tố khác như tình hình kinh tế, lãi suất, và các chính sách nhà nước. Để hiểu rõ hơn về cách mà các yếu tố này ảnh hưởng đến giá nhà, việc sử dụng các mô hình dự đoán trở nên vô cùng cần thiết.

Mô hình hồi quy tuyến tính là một trong những phương pháp phổ biến nhất được áp dụng để phân tích mối quan hệ giữa giá nhà và các biến độc lập như diện tích, vị trí, số lượng phòng ngủ, và các yếu tố khác. Bằng cách phân tích dữ liệu quá khứ và tìm ra mối liên hệ giữa các yếu tố này với giá nhà, mô hình hồi quy tuyến tính giúp dự đoán giá nhà trong tương lai với độ chính xác cao.

## 1.2. Khảo sát hiện tượng giá nhà hiện nay

Thị trường bất động sản nhà ở Việt Nam được phân khúc theo loại hình (Biệt thự và Nhà liền thổ và Chung cư và Căn hộ) và theo Thành phố (Thành phố Hồ Chí Minh, Hà Nội, Đà Nẵng, Quảng Ninh và phần còn lại của Việt Nam). Báo cáo đưa ra quy mô thị trường và dự báo cho thị trường bất động sản nhà ở Việt Nam về giá trị (tỷ USD) cho tất cả các phân khúc trên.

### *1.2.1. Quy mô thị trường nhà ở Việt Nam*


*Hình 1.1: Quy mô thị trường nhà ở Việt Nam*

### *1.2.2. Phân tích thị trường bất động sản nhà ở Việt Nam*

- Thị trường bất động sản nhà ở tại Việt Nam đang trải qua giai đoạn phát triển mạnh mẽ. Quy mô thị trường được dự báo sẽ tăng trưởng nhanh chóng, từ 22,44 tỷ USD vào năm 2023 lên 40,53 tỷ USD vào năm 2028, với tốc độ tăng trưởng kép hàng năm (CAGR) ước tính khoảng 12,55% trong giai đoạn này.
- Việt Nam đang chứng kiến những thay đổi đáng kể về mặt nhân khẩu học và xã hội. Dân số cả nước, ước tính đạt 100 triệu người vào năm 2022, dự kiến sẽ tăng lên 120 triệu người vào năm 2050. Những thay đổi này dự kiến sẽ thúc đẩy nhu cầu mạnh mẽ trong thị trường bất động sản nhà ở trong những năm tới.
- Trong bối cảnh khu vực Đông Nam Á, Việt Nam đang nổi lên như một nền kinh tế đầy tiềm năng, đặc biệt trong lĩnh vực xây dựng và bất động sản.

Triển vọng thị trường nhà ở tại Việt Nam được đánh giá là sôi động, chủ yếu nhờ vào tốc độ tăng trưởng kinh tế mạnh mẽ, đô thị hóa nhanh chóng và sự xuất hiện của nhiều dự án lớn tại các thành phố trọng điểm như Hồ Chí Minh và Hà Nội.

- Sự phát triển của internet và sự gia tăng thu nhập của tầng lớp trung lưu trẻ tại Việt Nam đã thúc đẩy nhu cầu mua bán bất động sản qua các kênh trực tuyến. Tài sản cá nhân tăng nhanh đã khiến bất động sản trở nên dễ tiếp cận hơn với nhiều người dân, góp phần vào sự phát triển mới của thị trường và tăng giá trị bất động sản.
- Trên thị trường, xu hướng đã dịch chuyển từ phân khúc cao cấp sang phân khúc trung bình, khu đô thị hóa tiếp tục tạo ra nhu cầu ổn định về nhà ở tại các khu vực đô thị lớn. Bên cạnh đó, Việt Nam hiện cũng là điểm nóng cho các dự án bất động sản cao cấp, nhờ vào sự phát triển kinh tế ổn định và luật pháp thuận lợi cho người nước ngoài mua bất động sản.
- Thủ tướng Chính phủ Phạm Minh Chính đã đặt mục tiêu đến năm 2030 sẽ xây dựng ít nhất một triệu đơn vị nhà ở xã hội - những căn hộ giá rẻ được nhà nước hỗ trợ.
- Tăng cường các sáng kiến của chính phủ và chính sách phát triển nhà ở xã hội
- Chính phủ Việt Nam đã ban hành nhiều chính sách nhằm hỗ trợ và thúc đẩy việc phát triển nhà ở xã hội, nhắm đến các đối tượng thu nhập thấp tại khu vực đô thị và nông thôn. Các sáng kiến này bao gồm luật nhà ở và các chỉ thị chính thức khác, nhằm cung cấp sự hỗ trợ và khuyến khích cho việc phát

triển và quản lý nhà ở xã hội. Thị trường cũng đang mở rộng để thu hút đầu tư từ các cá nhân và tổ chức nước ngoài vào lĩnh vực này.

- Tại TP. Hồ Chí Minh, nhiều dự án đã được triển khai nhằm tăng nguồn cung đất sạch cho khu vực đô thị và vùng ven. Điều này có thể mở rộng quy mô phát triển nhà ở xã hội, đặc biệt là ở những khu vực ngoài trung tâm.
- Theo Bộ Xây dựng, hầu hết các dự án nhà ở xã hội sẽ được triển khai tại các khu vực có khu công nghiệp lớn như Long An, Bắc Giang, Bắc Ninh và Bình Dương. Tại các đô thị lớn như Hà Nội, Thành phố Hồ Chí Minh, Hải Phòng và Đà Nẵng, các dự án nhà ở xã hội cũng được ưu tiên. Khoảng 700.000 căn hộ dự kiến sẽ được hoàn thành trong giai đoạn 2021-2025, và con số này sẽ tăng lên 1,1 triệu căn hộ vào năm 2025-2030.
- Đô thị hóa nhanh chóng đang thúc đẩy nhu cầu về nhà ở
- Quá trình đô thị hóa sẽ là động lực quan trọng cho sự phát triển kinh tế - xã hội nhanh chóng và bền vững trong tương lai. Các khu vực đô thị được dự đoán sẽ chiếm phần lớn trong nền kinh tế của Việt Nam trong những năm tới. Do đó, các cấp chính quyền và đảng viên được yêu cầu tập trung vào việc quy hoạch, xây dựng và quản lý bền vững các đô thị vào năm 2030.
- Các mục tiêu cụ thể đã được đặt ra, bao gồm việc đạt tốc độ đô thị hóa ít nhất 5% vào năm 2025 và hơn 50% vào năm 2030. Tỷ trọng của nền kinh tế đô thị trong GDP cũng được dự đoán sẽ tăng lên khoảng 75% vào năm 2025 và 85% vào năm 2030. Với sự phát triển này, đô thị hóa không chỉ là động lực thúc đẩy thị trường bất động sản mà còn là yếu tố quan trọng trong sự phát triển toàn diện của đất nước.

### *1.2.3. Bất động sản nhà ở tại Việt Nam - Đô thị hóa tại Việt Nam*

Thị trường bất động sản nhà ở Việt Nam bị phân mảnh do sự hiện diện của nhiều người chơi trong nước và toàn cầu. Bất động sản nhà ở Việt Nam bao gồm:

1. Các công ty địa phương thuần Việt.
2. Quỹ đầu tư nước ngoài từ các công ty nước ngoài.
3. Các công ty liên doanh.

Nhiều công ty khởi nghiệp proptech và các công ty bất động sản truyền thống nhằm mục đích tận dụng công nghệ để cải thiện hoạt động và lợi thế cạnh tranh của họ bằng cách cung cấp các giải pháp thiết thực. Nó nâng cao trải nghiệm mua, bán, thuê và sống nhà tại Việt Nam.

Các công ty chủ chốt trong thị trường bất động sản nhà ở bao gồm Tập đoàn Novaland, Tập đoàn Đất Xanh, Tập đoàn FLC, Tổng công ty Đầu tư Kinh doanh Bất động sản Hưng Thịnh và Tổng công ty Đầu tư Nam Long.

### *1.2.4. Tin tức thị trường bất động sản nhà ở Việt Nam*

Tháng 8/2022: Công ty thiết kế trải nghiệm FORREC công bố quan hệ đối tác chiến lược lâu dài với SUN Group. Với điều này, công ty sẽ tạo ra một trải nghiệm mới tại thị trường Việt Nam cho nhà phát triển du lịch Sun Group.

Tháng 6 năm 2022: Thành phố Hồ Chí Minh - Thành phố lớn nhất phía Nam đặt mục tiêu xây dựng 107,5 triệu mét vuông nhà trong thập kỷ tới. Đây là một trong những điểm nổi bật trong dự thảo chương trình phát triển nhà ở

của Thành phố Hồ Chí Minh giai đoạn 2021-2030. Dân số thành phố sẽ tăng lên khoảng 10,25 triệu người từ năm 2021 đến năm 2025. Năm 2025, cần phát triển khoảng 50 triệu m2 không gian sống, tương đương khoảng 367.000 căn nhà.

## 1.3. Phát biểu bài toán

Bài toán dự đoán giá nhà là một bài toán dự báo giá trị tài sản bất động sản. Mục tiêu của bài toán là tìm ra một mô hình dự đoán chính xác giá trị của một căn nhà dựa trên các thông tin và đặc trưng liên quan.

Bài toán dự đoán giá nhà có thể được tiếp cận từ nhiều góc độ khác nhau. Một trong những phương pháp phổ biến là sử dụng các mô hình học máy. Thông thường, dữ liệu về các yếu tố có thể ảnh hưởng đến giá nhà được thu thập như diện tích, số phòng ngủ, vị trí, tiện ích xung quanh và nhiều yếu tố khác. Sau đó, các thuật toán học máy có thể được áp dụng để xây dựng mô hình từ dữ liệu này.

Các phương pháp dự đoán giá nhà có thể bao gồm các thuật toán như hồi quy tuyến tính, hồi quy logistic, máy vector hỗ trợ (SVM), mạng nơ-ron và cây quyết định. Quá trình đào tạo mô hình thường bao gồm việc chia dữ liệu thành tập huấn luyện và tập kiểm tra, và sau đó tiến hành đào tạo mô hình trên tập huấn luyện để tìm ra các tham số tối ưu. Sau khi hoàn thành quá trình đào tạo, mô hình có thể được sử dụng để dự đoán giá trị mới của nhà dựa trên các thông tin đầu vào.

Ngoài ra, bài toán dự đoán giá nhà cũng có thể được tiếp cận từ các phương pháp khác như phân tích dữ liệu không gian, dự báo thời tiết, dự báo xu hướng thị trường và các yếu tố khác. Một yếu tố quan trọng trong việc dự đoán giá nhà là

đánh giá độ chính xác của mô hình dự đoán, từ đó có thể cải thiện và tối ưu hóa các mô hình dự đoán trong tương lai.

Bài toán dự đoán giá nhà bằng hồi quy tuyến tính là một phương pháp dự đoán giá trị của một căn nhà dựa trên các biến độc lập. Ý tưởng chính của phương pháp này là tạo ra một mô hình tuyến tính dựa trên các biến độc lập như diện tích, số phòng ngủ, vị trí, tiện ích xung quanh và các yếu tố khác.

Quá trình thực hiện bài toán bằng hồi quy tuyến tính bao gồm các bước sau:

1. Thu thập dữ liệu: Thu thập các dữ liệu về giá nhà và các biến độc lập liên quan, như diện tích, số phòng ngủ, vị trí, tiện ích xung quanh.
2. Tiền xử lý dữ liệu: Tiền xử lý dữ liệu để chuẩn hóa và loại bỏ các giá trị nhiễu, cùng với việc điền các giá trị còn thiếu.
3. Xây dựng mô hình: Xây dựng mô hình hồi quy tuyến tính bằng cách tìm kiếm các hệ số tối ưu để tối thiểu hóa sai số dự đoán.
4. Đánh giá mô hình: Đánh giá mô hình dựa trên các chỉ số đánh giá như sai số trung bình, hệ số xác định và giá trị F-statistic.
5. Sử dụng mô hình: Sử dụng mô hình đã xây dựng để dự đoán giá nhà cho các mẫu mới, dựa trên các giá trị biến độc lập.

Phương pháp hồi quy tuyến tính là một phương pháp đơn giản và phổ biến để dự đoán giá nhà, tuy nhiên nó có thể có nhược điểm khi mô hình chỉ xét đến các biến độc lập tuyến tính và không thể mô hình hóa mối quan hệ phi tuyến giữa các biến. Do đó, có thể cần sử dụng các phương pháp khác như hồi quy phi tuyến tính, mạng nơ-ron, hoặc cây quyết định để cải thiện khả năng dự đoán giá nhà.

# CHƯƠNG 2: MỘT SỐ KỸ THUẬT GIẢI QUYẾT BÀI TOÁN

## 2.1. Chi tiết về phương pháp hồi quy tuyến tính

### *2.1.1. Giới thiệu*

"Hồi quy tuyến tính" là một phương pháp thống kê để hồi quy dữ liệu với biến phụ thuộc có giá trị liên tục trong khi các biến độc lập có thể có một trong hai giá trị liên tục hoặc là giá trị phân loại. Nói cách khác "Hồi quy tuyến tính" là một phương pháp để dự đoán biến phụ thuộc (Y) dựa trên giá trị của biến độc lập (X). 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...

Giả sử căn nhà rộng $x_1$ $m^2$, có $x_2$ $m^2$ phòng ngủ và cách trung tâm thành phố $x_3$ km có giá là bao nhiêu. Giả sử chúng ta đã có số liệu thống kê từ 1000 căn nhà trong thành phố đó, liệu rằng khi có một căn nhà mới với các thông số về diện tích, số phòng ngủ và khoảng cách tới trung tâm, chúng ta có thể dự đoán được giá của căn nhà đó không? Nếu có thì hàm dự đoán $y = f(x)$ sẽ có dạng như thế nào. Ở đây $x = [x_1, x_2, x_3]$ là một vector hàng chứa thông tin input, y là một số vô hướng (scalar) biểu diễn output (tức giá của căn nhà trong ví dụ này).

Một cách đơn giản nhất, chúng ta có thể thấy rằng: diện tích nhà càng lớn thì giá nhà càng cao; số lượng phòng ngủ càng lớn thì giá nhà càng cao; càng xa trung tâm thì giá nhà càng giảm. Một hàm số đơn giản nhất có thể

mô tả mối quan hệ giữa giá nhà và 3 đại lượng đầu vào là: $w_1$, $w_2$, $w_3$ với $w_0$ là các hằng số, $w_0$ còn được gọi là bias. Mối quan hệ $y \approx f(x)$ bên trên là một mối quan hệ tuyến tính (linear). Bài toán chúng ta đang làm là một bài toán thuộc loại regression. Bài toán đi tìm các hệ số tối ưu $\{w_1, w_2, w_3, w_0\}$ chính vì vậy được gọi là bài toán Linear Regression.

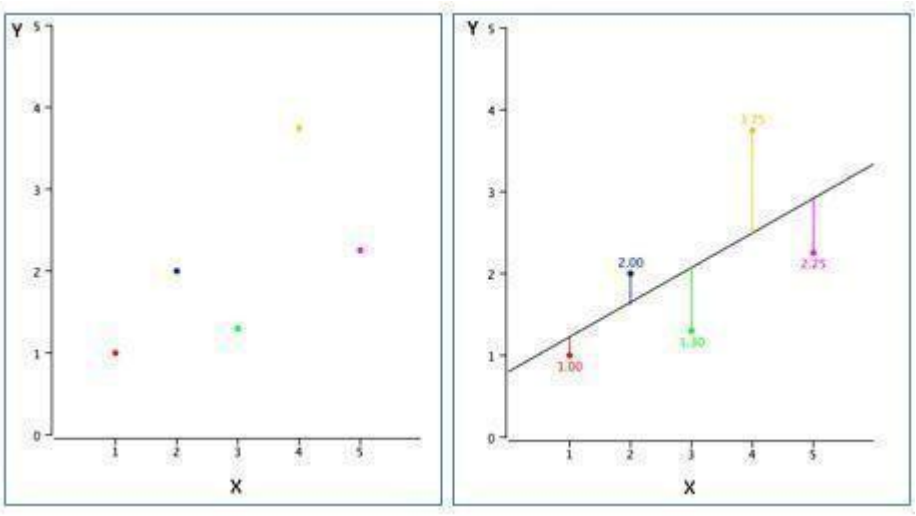
### *2.1.2. Phân tích toán học*

Trong phương trình 1 nếu chúng đặt $w = [w_0,w_1,w_2,w_3]$; $T = w = [w_0,w_1,w_2,w_3]$ T là vector (cột) hệ số cần phải tối ưu và $\bar{x} = [1,x_1,x_2,x_3]$

$\bar{x}=[1,x_1,x_2,x_3]$ (đọc là x bar trong tiếng Anh) là vector (hàng) dữ liệu đầu vào

mở rộng. Số 11 ở đầu được thêm vào để phép tính đơn giản hơn và thuận tiện cho việc tính toán. Khi đó, phương trình (1) có thể được viết lại dưới dạng:

Trong khi sử dụng hồi quy tuyến tính, mục tiêu của chúng ta là để làm sao một đường thẳng có thể tạo được sự phân bố gần nhất với hầu hết các điểm. Do đó làm giảm khoảng cách (sai số) của các điểm dữ liệu cho đến đường đó.




*Hình 2.1: Ảnh minh họa hồi quy tuyến tính*

## 2.2. Hồi quy ridge

### *2.2.1. Giới thiệu*

Hồi quy Ridge là một phương pháp hồi quy trong lĩnh vực học máy, được sử dụng để xây dựng mô hình dự đoán và giải thích mối quan hệ giữa các biến độc lập và biến phụ thuộc. Nó là một dạng của Regularized Linear Regression (Hồi quy tuyến tính với điều chuẩn) và là một trong những phương pháp phổ biến để giảm overfitting trong các mô hình hồi quy.

Trong hồi quy Ridge, mục tiêu là tìm ra một hàm số tuyến tính mà có thể dự đoán giá trị của biến phụ thuộc dựa trên các biến độc lập. Tuy nhiên, để tránh overfitting và đảm bảo tính ổn định của mô hình, chúng ta áp dụng một điều chuẩn (regularization) vào quá trình huấn luyện.

Phương pháp điều chuẩn trong hồi quy Ridge được thực hiện bằng cách thêm một thành phần điều chuẩn vào hàm mất mát của mô hình hồi quy tuyến tính. Thành phần điều chuẩn này là tổng bình phương của các hệ số hồi quy

(weights) nhân với một tham số điều chuẩn là lambda (λ). Công thức của hàm mất mát trong hồi quy Ridge là:

**L2 Loss = RSS + λ * (sum of squared weights)**

Trong công thức trên, RSS (Residual Sum of Squares) là tổng bình phương sai số giữa giá trị dự đoán và giá trị thực tế của biến phụ thuộc. Thành phần thứ hai là phần điều chuẩn, giúp giảm thiểu giá trị của các hệ số hồi quy để tránh overfitting.

Tham số điều chuẩn lambda (λ) có vai trò quyết định mức độ điều chuẩn được áp dụng trong mô hình. Khi λ càng lớn, giá trị của các hệ số hồi quy càng nhỏ, dẫn đến mô hình có tính ổn định hơn và giảm thiểu overfitting.

Hồi quy Ridge có nhiều ưu điểm, bao gồm khả năng xử lý các tình huống trong đó có sự tương quan mạnh giữa các biến độc lập, giảm thiểu overfitting và cung cấp một mô hình ổn định hơn. Tuy nhiên, cần lựa chọn tham số điều chuẩn λ một cách hợp lý để đạt được hiệu suất tốt nhất của mô hình.

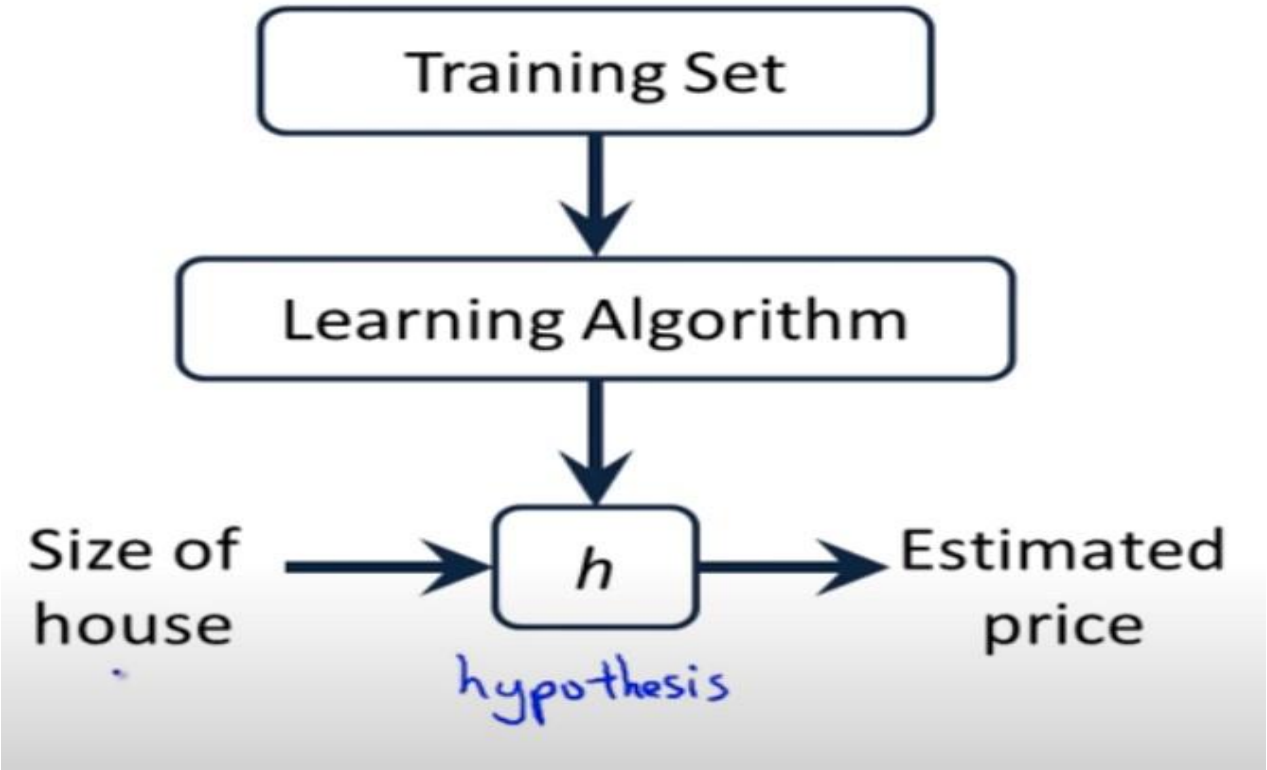
### *2.2.2. Phân tích toán học*

Giả định dữ liệu đầu vào bao gồm N quan sát là những cặp các biến đầu vào và biến mục tiêu ($X_1$, $Y_1$), ($X_2$, $Y_2$),…,($X_n$, $Y_n$). Quá trình hồi qui mô hình sẽ tìm kiếm một vectơ hệ số ước lượng W = [$w_0$, $w_1$,…,$w_n$]sao cho tối thiểu hoá *hàm mất mát* dạng MSE:

$$\mathcal{L}(\mathbf{w}) = \frac{1}{N}\sum_{i=1}^{N}(y_i - \mathbf{w}^\intercal \mathbf{x}_i) = \frac{1}{N}||\bar{\mathbf{X}}\mathbf{w} - \mathbf{y}||_2^2$$

*Hình 2.2: Công thức tổng quát Ridge*

Nhắc lại một chút về khái niệm hàm mất mát. Trong các mô hình học có giám sát của machine learning, từ dữ liệu đầu vào, thông qua phương pháp học tập (learning algorithm), chúng ta sẽ đặt ra một hàm giả thuyết *h* (hypothesis function) mô tả mối quan hệ dữ liệu giữa biến đầu vào và biến mục tiêu.




*Hình 2.2: Sơ đồ hồi quy ridge*

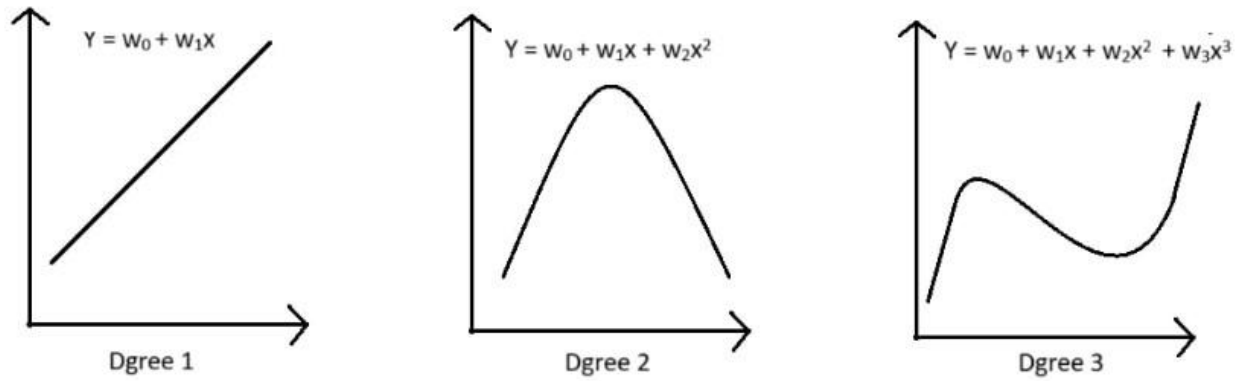
Nguồn: Andrew Ng - Linear Regression With One Variable. Từ một quan sát đầu vào X*i*, sau khi đưa vào hàm giả thuyết ℎ chúng ta thu được giá trị dự báo y^ ở đầu ra. Chữ ℎ của tên hàm thể hiện cho từ *hypothesis* có nghĩa

là *giả thuyết*, đây là một khái niệm đã tồn tại lâu năm trong thống kê. Để mô hình càng chuẩn xác thì sai số giữa giá trị dự báo y^ và ground truth y càng phải nhỏ. Vậy làm thế nào để đo lường được mức độ nhỏ của sai số giữa y^ và y. Các thuật toán học có giám sát trong machine learning sẽ sử dụng hàm mất mát để lượng hoá sai số này.

Hàm mất mát cũng chính là mục tiêu tối ưu khi huấn luyện mô hình. Dữ liệu đầu vào X và y được xem như là cố định và biến số của bài toán tối ưu chính là các giá trị trong vectơ w.

Giá trị hàm mất mát *MSE* chính là trung bình của tổng bình phương phần dư. Phần dư chính là chênh lệch giữa giá trị thực tế và giá trị dự báo. Tối thiểu hoá hàm mất mát nhằm mục đích làm cho giá trị dự báo ít chênh lệch so với giá trị thực tế, giá trị thực tế còn được gọi là ground truth. Trước khi huấn luyện mô hình chúng ta chưa thực sự biết vectơ hệ số W là gì. Chúng ta chỉ có thể đặt ra một giả thuyết về dạng hàm dự báo (trong trường hợp này là phương trình dạng tuyến tính) và các hệ số hồi quy tương ứng. Chính vì vậy mục đích của tối thiểu hoá hàm mất mát là để tìm ra tham số W phù hợp nhất mô tả một cách khái quát quan hệ dữ liệu giữa biến đầu vào X với biến mục tiêu Y trên tập huấn luyện.

Tuy nhiên mối quan hệ này nhiều khi không mô tả được quy luật khái quát của dữ liệu nên dẫn tới hiện tượng *quá khớp*. Một trong những nguyên nhân dẫn tới sự không khái quát của mô hình đó là do mô hình quá phức tạp. Mức độ phức tạp càng cao khi độ lớn của các hệ số trong mô hình hồi quy ở những bậc cao có xu hướng lớn như phân tích trong hình bên dưới:




*Hình 2.3: Đồ thị bài toán*

Hình thể hiện mức độ phức tạp của mô hình theo sự thay đổi của bậc. Phương trình có phức tạp lớn nhất 3: $y = w_0+w_1*x+w_2*x^2+w_3*x^3$. Trong chương trình THPT chúng ta biết rằng phương trình bậc 3 thông thường sẽ có 2 điểm uốn và độ phức tạp lớn hơn bậc hai chỉ có 1 điểm uốn. Khi $w_3 \rightarrow 0$ thì phương trình bậc 3 hội tụ về phương trình bậc 2: $y = w_0+w_1*x+w_2*x^2$, lúc này phương trình là một đường cong dạng parabol và có độ phức tạp giảm. Tiếp tục kiểm soát độ lớn để $w_2 \rightarrow 0$ trong phương trình bậc 2 ta sẽ thu được một đường thẳng tuyến tính dạng $y = w_0+w_1*x$ có độ phức tạp thấp nhất.

## 2.3. Hồi quy lasso

### *2.3.1. Giới thiệu*

Tuyến tính Lasso (Least Absolute Shrinkage and Selection Operator) là một phương pháp hồi quy trong lĩnh vực học máy, cũng thuộc vào nhóm Regularized Linear Regression (Hồi quy tuyến tính với điều chuẩn). Giống như hồi quy Ridge,

tuyến tính Lasso được sử dụng để xây dựng mô hình hồi quy và giải thích mối quan hệ giữa các biến độc lập và biến phụ thuộc. Tuy nhiên, Lasso có một cách tiếp cận khác để điều chuẩn và lựa chọn biến.

Trong tuyến tính Lasso, chúng ta cũng tìm kiếm một hàm số tuyến tính để dự đoán giá trị của biến phụ thuộc, những điểm khác biệt chính là phương pháp điều chuẩn. Thay vì sử dụng tổng bình phương các hệ số hồi quy như hồi quy Ridge, tuyến tính Lasso sử dụng tổng giá trị tuyệt đối của các hệ số hồi quy.

Công thức của hàm mất mát trong tuyến tính Lasso là:

**L1 Loss = RSS + λ * (sum of absolute weights)**

Trong công thức trên, RSS (Residual Sum of Squares) là tổng bình phương sai số giữa giá trị dự đoán và giá trị thực tế của biến phụ thuộc. Thành phần thứ hai là phần điều chuẩn, được tính bằng tổng giá trị tuyệt đối của các hệ số hồi quy nhân với tham số điều chuẩn lambda (λ). Điều này khiến một số hệ số trở thành 0, đồng nghĩa với việc loại bỏ các biến không quan trọng khỏi mô hình.

Tham số điều chuẩn lambda (λ) cũng có vai trò quyết định mức độ điều chuẩn trong mô hình. Khi λ càng lớn, mô hình sẽ có xu hướng loại bỏ nhiều biến không quan trọng hơn.

Tuyến tính Lasso có ưu điểm là giúp giải quyết vấn đề lựa chọn biến, loại bỏ các biến không quan trọng và giảm kích thước của mô hình. Điều này có thể giúp tăng tính diễn giải và hiệu suất của mô hình. Tuy nhiên, cần lựa chọn tham số điều chuẩn λ phù hợp để đạt được hiệu suất tốt nhất và tránh overfitting.

### *2.3.2. Phân tích toán học*

Trong hồi quy Lasso, thay vì sử dụng *thành phần điều chuẩn* là norm chuẩn bậc hai thì chúng ta sử dụng norm chuẩn bậc 1.

$$\mathcal{L}(\mathbf{w}) = \frac{1}{N}\|\bar{\mathbf{X}}\mathbf{w} - \mathbf{y}\|_2^2 + \underbrace{\alpha\|\mathbf{w}\|_1}_{\text{regularization term}}$$

Nếu bạn chưa biết về norm chuẩn bậc 1 thì có thể xem lại khái niệm norm chuẩn.

Khi tiến hành hồi qui mô hình *Lasso* trên một bộ dữ liệu mà có các biến đầu vào *đa cộng tuyến* (*multicollinear*) thì mô hình hồi quy Lasso sẽ có xu hướng lựa chọn ra một biến trong nhóm các biến đa cộng tuyến và bỏ qua những biến còn lại. Trong khi ở mô hình hồi quy tuyến tính thông thường và hồi quy Ridge thì có xu hướng sử dụng tất cả các biến đầu vào.

Bài toán tối ưu đối với *hàm hồi quy Lasso* tương đương với bài toán tối ưu với điều kiện ràng buộc về độ lớn của hàm mục tiêu:

$$\begin{aligned}\mathcal{L}(\mathbf{w}) &= \frac{1}{N}\|\bar{\mathbf{X}}\mathbf{w} - \mathbf{y}\|_2^2 + \alpha\|\mathbf{w}\|_1 \\ \text{subject} &: \|\mathbf{w}\|_1 < C, C > 0\end{aligned}$$

## 2.4. Hồi quy Elastic net

### *2.4.1. Giới thiệu*

Tuyến tính Elastic Net là một phương pháp hồi quy kết hợp cả điều chuẩn L1 (Lasso) và điều chuẩn L2 (Ridge) trong lĩnh vực học máy. Nó được

sử dụng để xây dựng mô hình hồi quy và giải thích mối quan hệ giữa các biến độc lập và biến phụ thuộc. Tuyến tính Elastic Net kết hợp những ưu điểm của cả Lasso và Ridge, giúp giải quyết vấn đề lựa chọn biến và ổn định hồi quy.

Trong tuyến tính Elastic Net, phương pháp điều chuẩn được áp dụng bằng cách kết hợp cả thành phần L1 và L2 vào hàm mất mát của mô hình. Công thức của hàm mất mát trong tuyến tính Elastic Net là:

**Elastic Net Loss = RSS + $\lambda_1$ * (sum of absolute weights) + $\lambda_2$ * (sum of squared weights)**

Trong công thức trên, RSS (Residual Sum of Squares) là tổng bình phương sai số giữa giá trị dự đoán và giá trị thực tế của biến phụ thuộc. Thành phần thứ hai là phần điều chuẩn L1, được tính bằng tổng giá trị tuyệt đối của các hệ số hồi quy nhân với tham số điều chuẩn lambda1 ($\lambda_1$). Thành phần thứ ba là phần điều chuẩn L2, được tính bằng tổng bình phương của các hệ số hồi quy nhân với tham số điều chuẩn lambda2 ($\lambda_2$).

Tham số điều chuẩn lambda1 ($\lambda_1$) và lambda2 ($\lambda_2$) là hai tham số quyết định mức độ điều chuẩn và tỉ lệ giữa L1 và L2 regularization. Khi lambda 1 = 0 và lambda 2 > 0, tuyến tính Elastic Net giống với hồi quy Ridge. Khi lambda 1 > 0 và lambda 2 = 0, nó tương đương với hồi quy Lasso. Khi cả lambda 1 và lambda 2 đều lớn hơn 0, cả hai phương pháp điều chuẩn đều được áp dụng.

Tuyến tính Elastic Net giúp giải quyết vấn đề lựa chọn biến và ổn định hồi quy. Nó thích hợp cho các tình huống trong đó có sự tương quan mạnh giữa các biến độc lập và khi mô hình cần lựa chọn một tập biến quan trọng từ

một tập biến lớn. Tuy nhiên, nhưng cần lựa chọn tham số điều chuẩn lambda 1 và lambda 2 phù hợp để đạt được hiệu suất tốt nhất và tránh overfitting.

### *2.4.2. Phân tích bài toán*

Hồi quy *Elastic Net* là một mô hình hồi quy cho phép chúng ta kết hợp đồng thời cả hai thành phần điều chuẩn là norm chuẩn bậc 1 và norm chuẩn bậc 2 theo một kết hợp tuyến tính lồi.

$$\mathcal{L}(\mathbf{w}) = \frac{1}{N}\|\bar{\mathbf{X}}\mathbf{w} - \mathbf{y}\|_2^2 + \alpha\left[\lambda\|\mathbf{w}\|_1 + \frac{(1-\lambda)}{2}\|\mathbf{w}\|_2^2\right]$$

# CHƯƠNG 3: THỰC NGHIỆM

## 3.1. Các thư viện được sử dụng

### *3.1.1. Thư viện numpy*

Thư viện numpy là một thư viện phổ biến trong ngôn ngữ lập trình Python, đặc biệt được sử dụng rộng rãi trong lĩnh vực tính toán khoa học và phân tích dữ liệu. Numpy cung cấp cho chúng ta các cấu trúc dữ liệu và hàm số mạnh mẽ để làm việc với mảng đa chiều và các phép toán số học trên chúng.

Một trong những tính năng quan trọng của numpy là khả năng xử lý dữ liệu nhanh và hiệu quả. Numpy sử dụng các khối dữ liệu gốc và các phép toán được thực hiện trực tiếp trên các mảng nhiều chiều, loại bỏ đi các vòng lặp và quá trình chuyển đổi dữ liệu khỏi chế độ dịch chuyển của các ngôn ngữ khác nhau. Điều này giúp numpy trở thành một công cụ mạnh mẽ để thao tác với dữ liệu lớn và phức tạp.

Numpy cung cấp nhiều hàm số và phương pháp để thực hiện các phép toán số học và thống kê trên các mảng. Ngoài ra, nó cũng cho phép ta thực hiện các phép gán, cắt, lọc và biến đổi các mảng một cách dễ dàng.

Thư viện numpy cũng kết hợp tốt với các thư viện khác trong ngữ cảnh tính toán khoa học, chẳng hạn như scipy, pandas và matplotlib. Nó cung cấp một cầu nối để chuyển đổi dữ liệu giữa các thư viện này một cách dễ dàng và hiệu quả.

Với những tính năng mạnh mẽ và hiệu suất cao, numpy là một công cụ quan trọng cho bất kỳ ai làm việc trong lĩnh vực tính toán khoa học và phân tích dữ liệu bằng Python.

- Các thao tác với Numpy

Cài đặt thư viện Numpy: Mở Command Prompt và gõ lệnh: pip install numpy

### *3.1.2. Thư viện pandas*

Thư viện pandas cũng là một trong những thư viện quan trọng trong ngôn ngữ lập trình Python, được sử dụng phổ biến trong lĩnh vực xử lý và phân tích dữ liệu.

Pandas cung cấp cấu trúc dữ liệu và công cụ mạnh mẽ để làm việc với dữ liệu số liệu hàng loạt và dữ liệu có cấu trúc, chẳng hạn như bảng dữ liệu hoặc tỷ lệ dữ liệu. Hai đối tượng cơ bản trong pandas là Series (dữ liệu 1 chiều) và DataFrame (dữ liệu 2 chiều), cho phép lưu trữ và xử lý dữ liệu một cách linh hoạt.

Pandas cho phép ta thực hiện các thao tác như truy cập, lọc, sắp xếp, thay đổi dữ liệu, thống kê, và tính toán trên các bộ dữ liệu. Nó cung cấp các hàm mạnh mẽ để xử lý dữ liệu bị thiếu, dữ liệu trùng lặp, và bộ dữ liệu không chuẩn. Ngoài ra, pandas cũng hỗ trợ các phép toán trên cột và hàng, và cho phép thực hiện các phép biến đổi dữ liệu phức tạp và merge dữ liệu từ nhiều nguồn.

Thư viện pandas cũng tích hợp tốt với các thư viện khác trong ngữ cảnh phân tích dữ liệu, chẳng hạn như numpy, matplotlib và scikit-learn. Nó cung

cấp các chức năng để chuyển đổi dữ liệu giữa các định dạng khác nhau và liên kết dữ liệu từ nhiều nguồn.

Với tính linh hoạt và hiệu suất cao trong việc làm việc với dữ liệu số liệu hàng loạt và có cấu trúc, thư viện pandas là một công cụ cần thiết cho các nhà phân tích dữ liệu và nhà khoa học dữ liệu sử dụng Python.

### *3.1.3. Thư viện matplotlib*

Thư viện matplotlib là một thư viện phổ biến trong ngôn ngữ lập trình Python, được sử dụng rộng rãi để tạo ra các biểu đồ và trực quan hóa dữ liệu. Với matplotlib, bạn có thể tạo ra nhiều loại biểu đồ như đường, cột, hình vuông, bản đồ, và biểu đồ phân loại.

Matplotlib cung cấp một API dễ sử dụng và mạnh mẽ để tạo ra các biểu đồ chất lượng cao và tùy chỉnh chúng theo ý muốn. Nó hỗ trợ các phép tính toán trên dữ liệu, nhãn trục, tiêu đề, chú thích, màu sắc, font chữ và nhiều thuộc tính khác để tạo ra các biểu đồ đẹp và dễ đọc.

Matplotlib cũng là một công cụ mạnh để thực hiện trực quan hóa dữ liệu phức tạp. Bạn có thể tạo ra các biểu đồ đa lớp, biểu đồ phổ, biểu đồ thống kê, biểu đồ tương quan, và nhiều loại biểu đồ khác để khám phá và hiểu dữ liệu của bạn. Nó cũng cung cấp các chức năng để lưu biểu đồ thành file ảnh hoặc PDF, và chia sẻ biểu đồ trong các ứng dụng khác.

Matplotlib có thể được tích hợp với các thư viện khác như numpy và pandas để trực quan hóa dữ liệu từ các cấu trúc dữ liệu này một cách dễ dàng. Nó cũng kết hợp tốt với các công cụ phân tích dữ liệu và máy học khác trong ngữ cảnh phân tích dữ liệu.

Với tính linh hoạt và khả năng tạo ra các biểu đồ đẹp và tương tác từ dữ liệu, matplotlib là một thư viện quan trọng cho việc trực quan hóa dữ liệu trong Python.

### *3.1.4. Thư viện warning*

Thư viện warnings trong Python là một module tích hợp sẵn để hiển thị cảnh báo (warnings) trong quá trình thực thi chương trình. Mục đích chính của module warnings là giúp người lập trình nhận biết những vấn đề có thể gây lỗi hoặc có thể cần kiểm tra, bổ sung, hoặc thay đổi.

Khi một cảnh báo được tạo ra trong quá trình thực thi, thông điệp cảnh báo sẽ được in ra. Mặc định, cảnh báo được in ra dưới dạng một thông báo cảnh báo đơn giản, nhưng bạn có thể tùy chỉnh cách hiển thị cảnh báo bằng cách sử dụng các chức năng trong thư viện warnings.

Các cảnh báo có thể được sử dụng để cảnh báo về những vấn đề như tham số không hợp lệ, sử dụng giá trị không mong muốn, hoặc những ưu tiên không tối ưu trong mã lập trình. Bằng cách sử dụng module warnings, người lập trình có thể kiểm soát việc sử dụng cảnh báo và đảm bảo rằng các vấn đề tiềm ẩn đã được nhận diện và xử lý một cách thích hợp.

### *3.1.5. Thư viện sklearn*

Thư viện sklearn (hay còn gọi là scikit-learn) là một trong những thư viện phổ biến và mạnh mẽ trong ngôn ngữ lập trình Python cho việc phân tích dữ liệu và học máy. Nó cung cấp một bộ công cụ rộng rãi để thực hiện các nhiệm vụ như phân loại, hồi quy, gom cụm, sao chép, rừng ngẫu nhiên, và nhiều nhiệm vụ khác liên quan đến học máy và phân tích dữ liệu.

Sklearn giúp bạn xây dựng mô hình học máy và thực hiện các công việc quan trọng như xử lý dữ liệu, chọn đặc trưng, phân loại, và đánh giá mô hình. Nó hỗ trợ nhiều thuật toán học máy phổ biến như hồi quy tuyến tính, hồi quy logistic, máy vector hỗ trợ, cây quyết định, và nhiều thuật toán khác. Ngoài ra, sklearn cũng cung cấp các công cụ để xử lý dữ liệu bị thiếu, chuẩn hóa dữ liệu và thực hiện cross-validation.

Sklearn cũng tích hợp tốt với các thư viện khác trong ngữ cảnh phân tích dữ liệu và học máy. Nó có thể được kết hợp với pandas để xử lý dữ liệu và numpy để thực hiện tính toán. Nó cũng kết hợp với thư viện matplotlib để trực quan hóa dữ liệu và kết quả của mô hình học máy.

Với tính dễ sử dụng, tính linh hoạt và hiệu suất cao, sklearn là một công cụ quan trọng cho người phân tích dữ liệu và nhà khoa học dữ liệu sử dụng Python để thực hiện các tác vụ học máy và phân tích dữ liệu.

### *3.1.6. Thư viện XGBoost*

Thư viện XGBoost là một thư viện thông dụng và mạnh mẽ được sử dụng cho việc xây dựng mô hình học máy dựa trên cây quyết định tăng cường (boosted decision trees). XGBoost là viết tắt của "eXtreme Gradient Boosting".

XGBoost cung cấp một giải thuật tối ưu hóa hiệu suất cao và có khả năng xử lý các vấn đề dự đoán và phân loại dữ liệu phức tạp. Nó có thể được sử dụng cho cả bài toán phân loại và hồi quy, và hỗ trợ các đặc trưng liên tục và rời rạc.

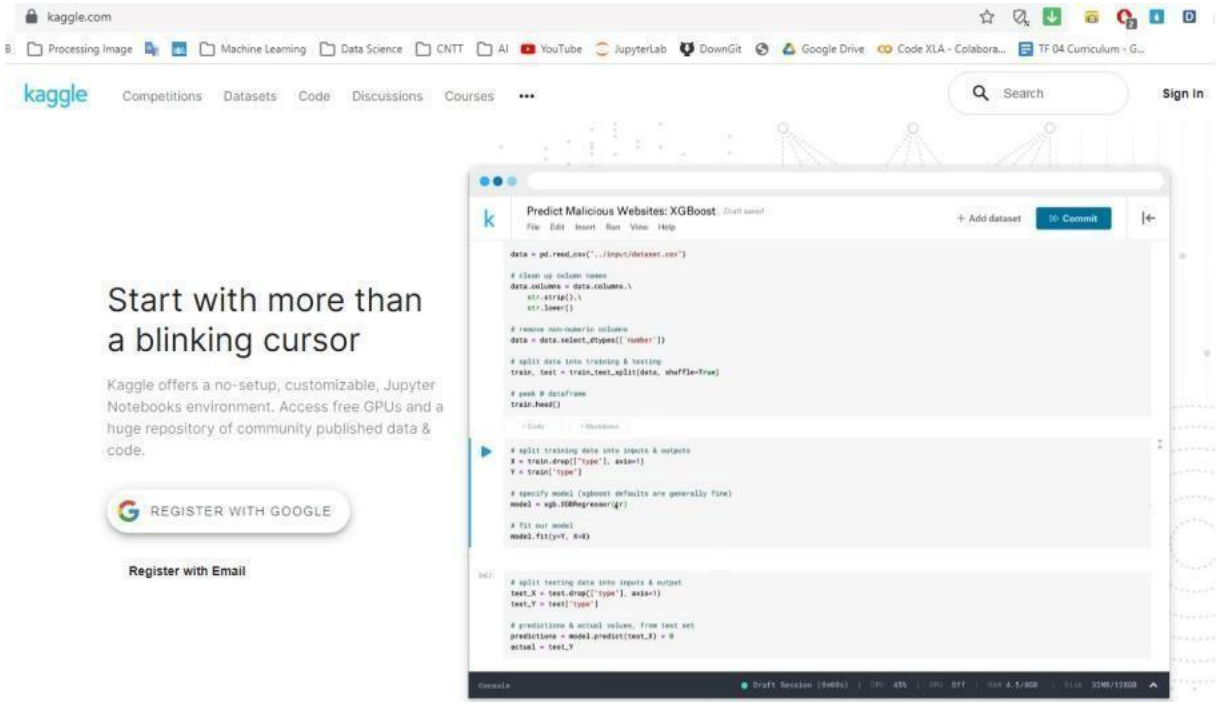
Một trong những đặc điểm nổi bật của XGBoost là khả năng xử lý lượng lớn dữ liệu và cải thiện hiệu suất mô hình thông qua tối ưu hóa đa luồng và quá trình tính toán song song. Nó cũng hỗ trợ các tiện ích như xử lý các giá trị thiếu và xử lý điểm ảnh hơn các phương pháp truyền thống khác.

XGBoost cung cấp các thông số tùy chỉnh để điều chỉnh mô hình, bao gồm số lượng cây, độ sâu cây, tỷ lệ học và chi phí học. Nó cũng hỗ trợ việc đánh giá và kiểm tra chéo mô hình để đánh giá độ chính xác và xử lý overfitting.

Với hiệu suất và tính ổn định cao, XGBoost là một thư viện quan trọng trong lĩnh vực học máy và đã được sử dụng trong nhiều ứng dụng thực tế và cuộc thi khoa học dữ liệu.

## 3.2. Bộ dữ liệu

Dữ liệu được lấy trên trang Kaggle: là một trong những trang tổ chức các cuộc thi nhiều nhất trên thế giới về AI.




*Hình 3.1: Giới thiệu về trang Kaggle*

Bộ dữ liệu gồm 80 thuộc tính, trong đó có 79 biến phụ thuộc và 1 biến độc lập:

- SalePrice - giá bán của bất động sản tính bằng đô la. Đây là biến mục tiêu mà bạn đang cố gắng dự đoán
- MSSubClass: Lớp xây dựng
- MSZoning: Phân loại phân vùng chung
- LotFrontage: Bộ tuyến tính của đường phố kết nối với tài sản
- LotArea: Kích thước lô tính bằng feet vuông
- Street: Loại đường vào
- Alley: Loại lối vào hẻm
- LotShape: Hình dạng chung của tài sản
- LandContour: Độ phẳng của tài sản
- Utilities: Loại tiện ích có sẵn
- LotConfig: Cấu hình lô
- LandSlope: Độ dốc của tài sản
- Neighborhood: Vị trí thực tế trong giới hạn thành phố Ames

- Condition1: Gần đường chính hoặc đường sắt
- Condition2: Gần đường chính hoặc đường sắt (nếu có một giây)
- BldgType: Loại nhà
- HouseStyle: Phong cách ở
- OverallQual: Chất lượng tổng thể và chất lượng hoàn thiện
- OverallCond: Đánh giá tình trạng tổng thể
- YearBuilt: Ngày xây dựng ban đầu
- YearRemodAdd: Ngày sửa lại
- RoofStyle: Loại mái
- RoofMatl: Vật liệu mái nhà
- Exterior1st: Lớp phủ bên ngoài ngôi nhà
- Exterior2nd: Lớp phủ bên ngoài ngôi nhà (nếu nhiều vật liệu)
- MasVnrType: Loại veneer xây
- MasVnrArea: Diện tích ván xây tính bằng feet vuông
- ExterQual: Chất lượng vật liệu ngoại thất
- ExterCond: Tình trạng hiện tại của vật liệu bên ngoài
- Foundation: Loại móng
- BsmtQual: Chiều cao của tầng hầm
- BsmtCond: Tình trạng chung của tầng hầm
- BsmtExposure: Lối đi hoặc tường tầng hầm sân vườn
- BsmtFinType1: Chất lượng hoàn thiện khu vực tầng hầm
- BsmtFinSF1: Loại 1 bộ vuông thành phẩm
- BsmtFinType2: Chất lượng của khu vực hoàn thiện thứ hai (nếu có)
- BsmtFinSF2: Loại 2 feet vuông thành phẩm
- BsmtUnfSF: Diện tích tầng hầm chưa hoàn thành bộ vuông
- TotalBsmtSF: Tổng diện tích tầng hầm
- Heating: Loại sưởi ấm
- HeatingQC: Chất lượng và tình trạng sưởi ấm
- CentralAir: Điều hòa trung tâm
- Electrical: Hệ thống điện
- 1stFlrSF: Bộ vuông tầng một
- 2ndFlrSF: Tầng hai bộ vuông
- LowQualFinSF: Bộ vuông hoàn thiện chất lượng thấp (tất cả các tầng)
- GrLivArea: Diện tích sống trên cao (mặt đất) bộ vuông
- BsmtFullBath: Phòng tắm đầy đủ tầng hầm

- BsmtHalfBath: Phòng tắm nửa tầng hầm
- FullBath: Phòng tắm đầy đủ trên lớp
- HalfBath: Nửa bồn tắm trên lớp
- Bedroom: Số phòng ngủ trên tầng hầm
- Kitchen: Số nhà bếp
- KitchenQual: Chất lượng bếp
- TotRmsAbvGrd: Tổng số phòng trên hạng (không bao gồm phòng tắm)
- Functional: Đánh giá chức năng nhà
- Fireplaces: Số lượng lò sưởi
- FireplaceQu: Chất lượng lò sưởi
- GarageType: Vị trí nhà để xe
- GarageYrBlt: Nhà để xe năm được xây dựng
- GarageFinish: Hoàn thiện nội thất của nhà để xe
- GarageCars: Kích thước nhà để xe có sức chứa ô tô
- GarageArea: Kích thước của nhà để xe tính bằng feet vuông
- GarageQual: Chất lượng nhà để xe
- GarageCond: Tình trạng nhà để xe
- PavedDrive: Đường lái xe trải nhựa
- WoodDeckSF: Diện tích sàn gỗ tính bằng mét vuông
- OpenPorchSF: Diện tích hiên mở tính bằng feet vuông
- EnclosedPorch: Diện tích hiên khép kín tính bằng feet vuông
- 3SsnPorch: Khu vực hiên ba mùa tính bằng mét vuông
- ScreenPorch: Màn hình hiện diện tích tính bằng feet vuông
- PoolArea: Diện tích hồ bơi tính bằng mét vuông
- PoolQC: Chất lượng hồ bơi
- Fence: Chất lượng hàng rào
- MiscFeature: Tính năng khác không được đề cập trong các danh mục khác
- MiscVal: $ Giá trị của tính năng linh tinh
- MoSold: Tháng đã bán
- YrSold: Năm đã bán
- SaleType: Hình thức bán hàng
- SaleCondition: Điều kiện bán hàng

## 3.3. Mô hình bài toán

### *3.3.1. Lấy dữ liệu*

- Dữ liệu gồm 1 ma trận có 1460 hàng và 81 cột

```
# Hiển thị số lượng ví dụ và đặc trưng trong tập train và test
print(df_train.shape)
print(df_test.shape)
```

```
(1460, 81)
(1459, 80)
```

*Hình 3.2: Hiển thị dữ liệu*

- Hiển thị 10 giá trị đầu tiên của tập train

```
54] # Hiển thị 10 giá trị đầu tiên của tập train
    df_train.head(10)
```

| | Id | MSSubClass | MSZoning | LotFrontage | LotArea | Street | Alley | LotShape | LandContour | Utilities | ... | PoolArea | PoolQC | Fence | MiscFeature | MiscVal | MoSold |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 1 | 60 | RL | 65.0 | 8450 | Pave | NaN | Reg | Lvl | AllPub | ... | 0 | NaN | NaN | NaN | 0 | 2 |
| 1 | 2 | 20 | RL | 80.0 | 9600 | Pave | NaN | Reg | Lvl | AllPub | ... | 0 | NaN | NaN | NaN | 0 | 5 |
| 2 | 3 | 60 | RL | 68.0 | 11250 | Pave | NaN | IR1 | Lvl | AllPub | ... | 0 | NaN | NaN | NaN | 0 | 9 |
| 3 | 4 | 70 | RL | 60.0 | 9550 | Pave | NaN | IR1 | Lvl | AllPub | ... | 0 | NaN | NaN | NaN | 0 | 2 |
| 4 | 5 | 60 | RL | 84.0 | 14260 | Pave | NaN | IR1 | Lvl | AllPub | ... | 0 | NaN | NaN | NaN | 0 | 12 |
| 5 | 6 | 50 | RL | 85.0 | 14115 | Pave | NaN | IR1 | Lvl | AllPub | ... | 0 | NaN | MnPrv | Shed | 700 | 10 |
| 6 | 7 | 20 | RL | 75.0 | 10084 | Pave | NaN | Reg | Lvl | AllPub | ... | 0 | NaN | NaN | NaN | 0 | 8 |
| 7 | 8 | 60 | RL | NaN | 10382 | Pave | NaN | IR1 | Lvl | AllPub | ... | 0 | NaN | NaN | Shed | 350 | 11 |
| 8 | 9 | 50 | RM | 51.0 | 6120 | Pave | NaN | Reg | Lvl | AllPub | ... | 0 | NaN | NaN | NaN | 0 | 4 |
| 9 | 10 | 190 | RL | 50.0 | 7420 | Pave | NaN | Reg | Lvl | AllPub | ... | 0 | NaN | NaN | NaN | 0 | 1 |

10 rows × 81 columns

*Hình 3.3: Hiển thị 10 giá trị đầu tiên trong bộ*

Xem một số chi tiết thống kê cơ bản như độ lệch chuẩn, trung bình ,của data thông qua describe(), những thông số này giúp ta hiểu được tập giá trị có đồng đều không..

```
df_train.describe()
```

| | MSSubClass | MSZoning | LotFrontage | LotArea | Street | LotShape | LandContour | LotConfig | LandSlope | M |
|---|---|---|---|---|---|---|---|---|---|---|
| **count** | 1455.000000 | 1455.000000 | 1455.000000 | 1455.000000 | 1455.000000 | 1455.000000 | 1455.000000 | 1455.000000 | 1455.000000 | |
| **mean** | 56.876289 | 3.028866 | 69.498282 | 10438.495533 | 0.995876 | 1.945704 | 2.780756 | 3.026804 | 0.062543 | |
| **std** | 42.370263 | 0.633101 | 20.730699 | 9856.332592 | 0.064106 | 1.408539 | 0.701277 | 1.617467 | 0.276683 | |
| **min** | 20.000000 | 0.000000 | 21.000000 | 1300.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | |
| **25%** | 20.000000 | 3.000000 | 60.000000 | 7537.500000 | 1.000000 | 0.000000 | 3.000000 | 2.000000 | 0.000000 | |
| **50%** | 50.000000 | 3.000000 | 69.000000 | 9464.000000 | 1.000000 | 3.000000 | 3.000000 | 4.000000 | 0.000000 | |
| **75%** | 70.000000 | 3.000000 | 79.000000 | 11568.500000 | 1.000000 | 3.000000 | 3.000000 | 4.000000 | 0.000000 | |
| **max** | 190.000000 | 4.000000 | 313.000000 | 215245.000000 | 1.000000 | 3.000000 | 3.000000 | 4.000000 | 2.000000 | |

8 rows × 81 columns

*Hình 3.4: Xuất ra giá trị về độ lệch chuẩn của dữ liệu*

Xem thông tin về tập dữ liệu gồm có tên cột số lượng phần tử rỗng và kiểu dữ liệu của các cột

```
# hiển thị thông tin của các cột dữ liệu train
df_train.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1460 entries, 0 to 1459
Data columns (total 81 columns):
 #   Column         Non-Null Count  Dtype
---  ------         --------------  -----
 0   Id             1460 non-null   int64
 1   MSSubClass     1460 non-null   int64
 2   MSZoning       1460 non-null   object
 3   LotFrontage    1201 non-null   float64
 4   LotArea        1460 non-null   int64
 5   Street         1460 non-null   object
 6   Alley          91 non-null     object
 7   LotShape       1460 non-null   object
 8   LandContour    1460 non-null   object
 9   Utilities      1460 non-null   object
 10  LotConfig      1460 non-null   object
 11  LandSlope      1460 non-null   object
 12  Neighborhood   1460 non-null   object
 13  Condition1     1460 non-null   object
 14  Condition2     1460 non-null   object
 15  BldgType       1460 non-null   object
 16  HouseStyle     1460 non-null   object
 17  OverallQual    1460 non-null   int64
 18  OverallCond    1460 non-null   int64
 19  YearBuilt      1460 non-null   int64
 20  YearRemodAdd   1460 non-null   int64
```

*Hình 3.5: Hiển thị thông tin của các cột train*

- Hiển thị nhãn dự đoán

```
df_train['SalePrice'].describe()
```

```
count      1460.000000
mean     180921.195890
std       79442.502883
min       34900.000000
25%      129975.000000
50%      163000.000000
75%      214000.000000
max      755000.000000
Name: SalePrice, dtype: float64
```

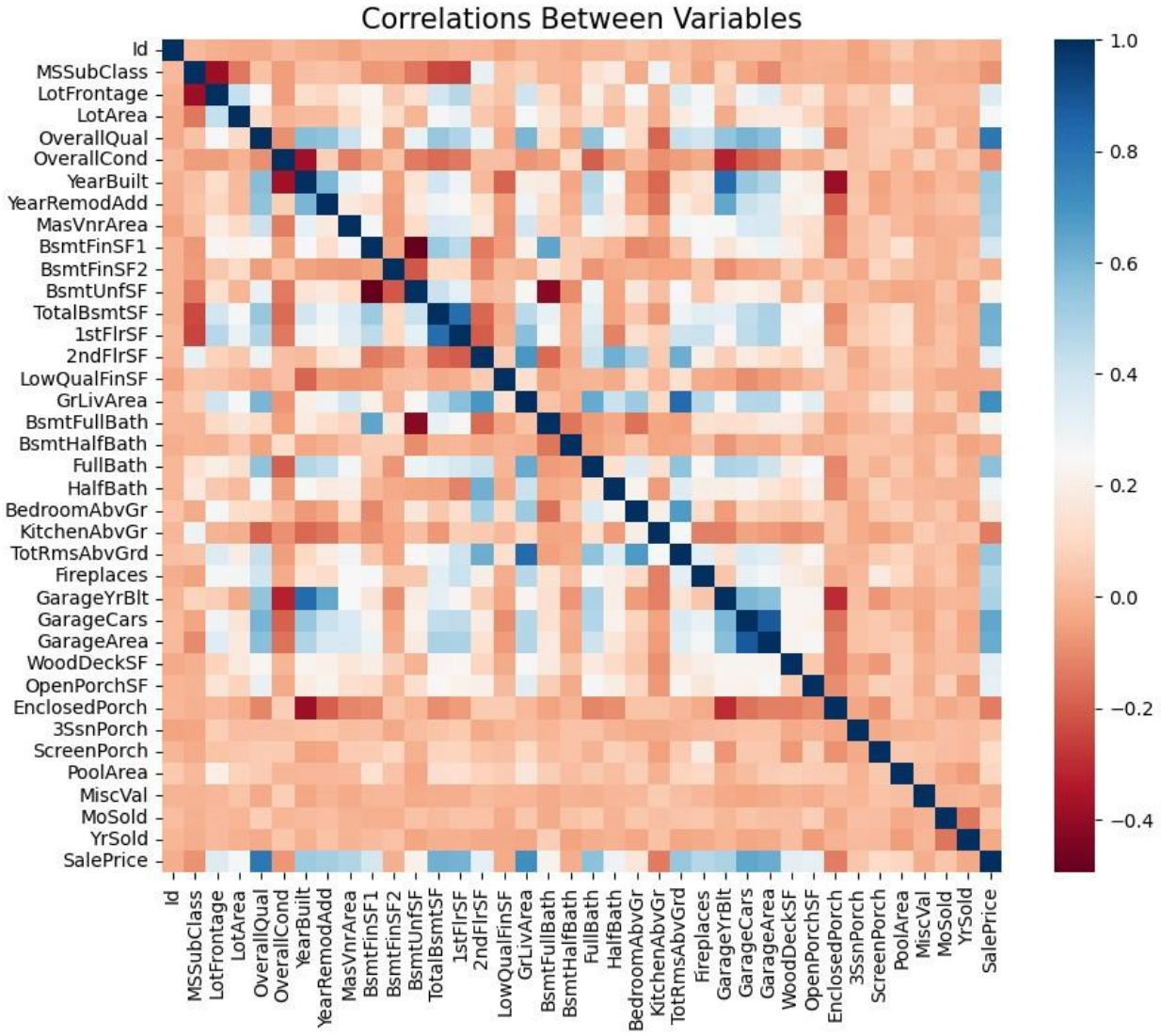
- Giá bán trung bình của một căn nhà là 180.921
- Giá bán tối đa của một căn nhà là 755.000 và Tối thiểu là 34.900

*Hình 3.6: Hiển thị nhãn dự đoán*

Tính độ tương quan giữa các dữ liệu với nhau và hiển thị dưới dạng biểu đồ nhiệt

```
# Hiển thị ma trận tương quan của tập train dưới dạng biểu đồ nhiệt
corrmat = df_train.corr()
# Cài đặt kcih thước hiển thị với chiều dài là 12, chiều rộng là 9
plt.figure(figsize=(50,30))
sns.heatmap(corrmat,annot=True)
```

*Hình 3.7: Độ tương quan giữa các dữ liệu với nhau và hiển thị dưới dạng biểu đồ nhiệt*

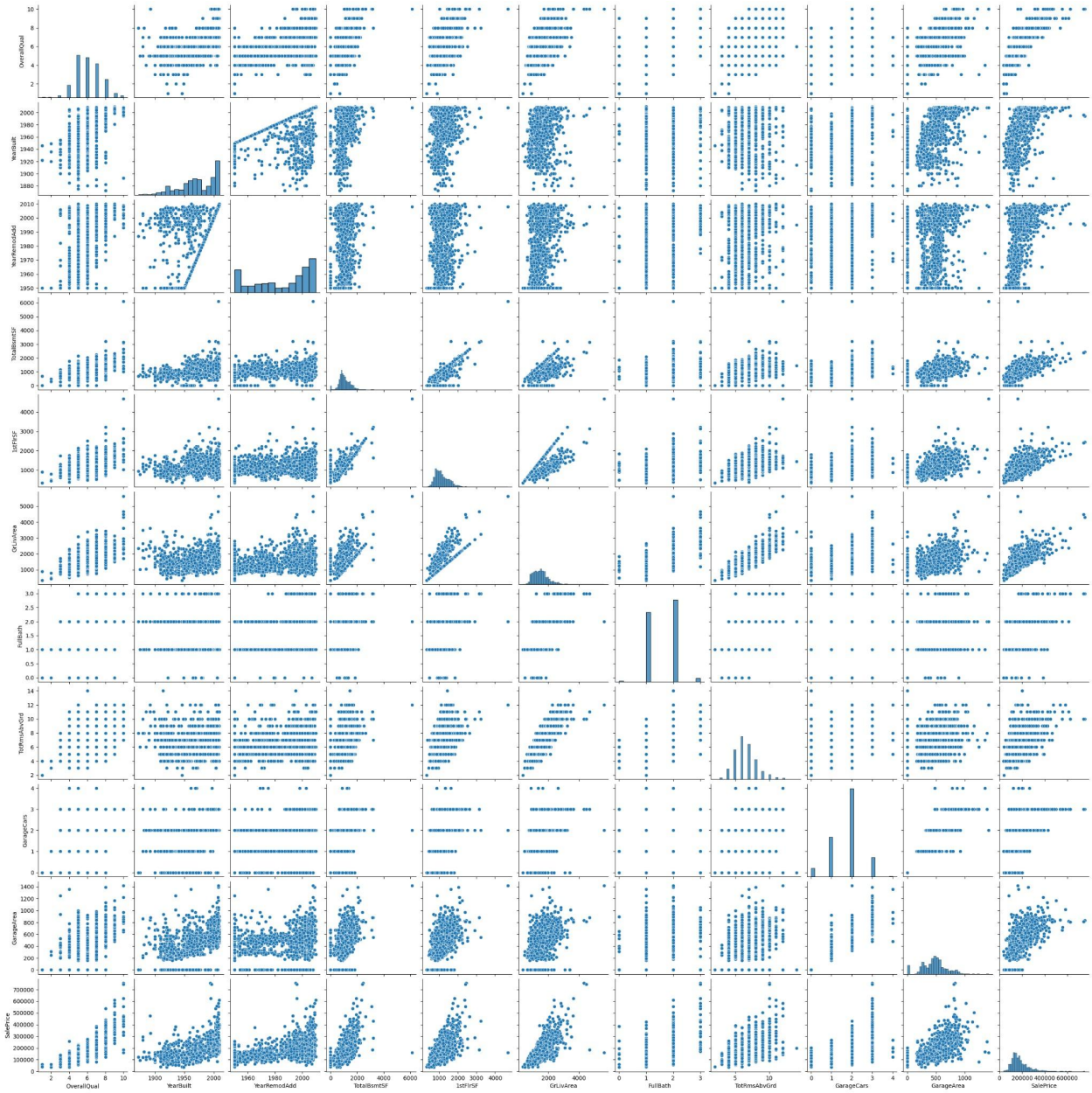



*Hình 3.8: Hiển thị kết quán sau khi tính độ tương quan*

- Hiển thị biểu đồ phần tán giữa 9 tính năng

```
rcParams['figure.figsize'] = 5,5
cols = ['SalePrice', 'OverallQual', 'GrLivArea', 'GarageCars','GarageArea', 'Total
sns_plot = sns.pairplot(df_train[cols])

plt.suptitle('Biểu đồ phân tán giữa 9 tính năng tốt nhất', y=1.04, size=25)
plt.tight_layout()
plt.show()
```

*Hình 3.9: Tính độ phân tán 9 tính năng*



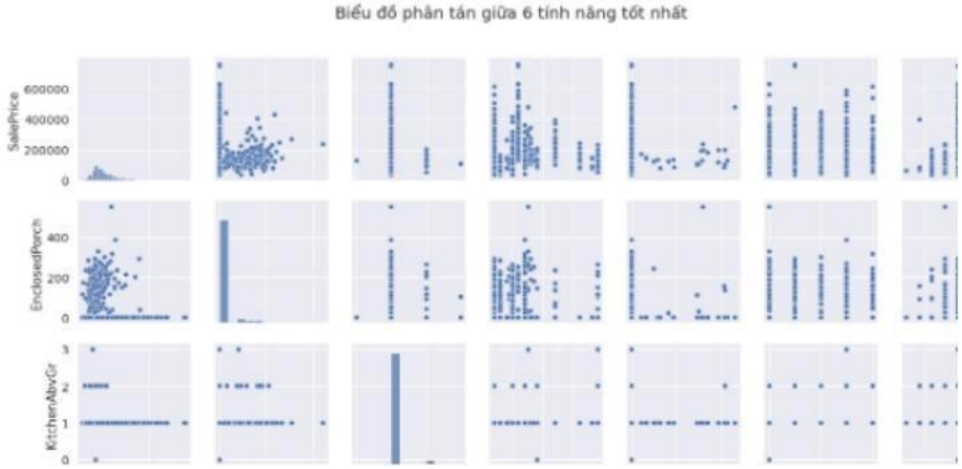

*Hình 3.10: Biểu đồ phân tán giữa 9 tính năng*

Biểu đồ phân tán giữa 6 tính năng tốt nhất

```
rcParams['figure.figsize'] = 5,5
cols = ['SalePrice','EnclosedPorch', 'KitchenAbvGr', 'MSSubClass', 'LowQualFinSF',
sns_plot = sns.pairplot(df_train[cols])

plt.suptitle('Biểu đồ phân tán giữa 6 tính năng tốt nhất', y=1.04, size=20)
plt.tight_layout()
plt.show()
```

Thêm ô văn bản

*Hình 3.11: Tính độ phân tán 6 tính năng tốt nhất*




*Hình 3.12: Biểu đồ phân tán giữa 6 tính năng tốt nhất*

- Xóa cột ID column

```
# xóa id vì nó không bắt buộc để đào tạo hoặc dự đoán
train_ID = df_train['Id']
test_ID = df_test['Id']

df_train.drop(['Id'], axis=1, inplace=True)
df_test.drop(['Id'], axis=1, inplace=True)

df_train.shape, df_test.shape
```

```
((1460, 80), (1459, 79))
```

*Hình 3.13: Xóa id ko bắt buộc*

### *3.3.2. Mã hóa dữ liệu*

Vì dữ liệu còn ở dạng chữ không thuận tiện cho việc tính toán nên ta phải chuẩn hóa các dữ liệu đã có về dạng số , sử dụng thư viện sklearn để chuẩn hóa.

```
from sklearn.preprocessing import LabelEncoder
lab=LabelEncoder()
label = list(df_train.select_dtypes(['object']).columns)

for i in label:
    df_train[i] = lab.fit_transform(df_train[i])
    # print(df_train[i])
```

*Hình 3.14: Chương trình mã hóa dữ liệu*

### *3.3.3. Xử lý ngoại lệ*

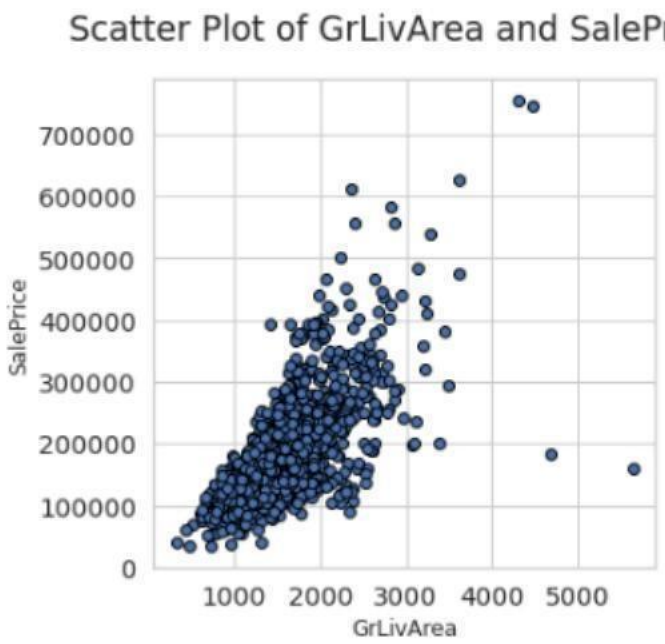
- Kiểm tra các ngoại lệ

```
sns.set_style('whitegrid')
edgecolor = 'black'

fig = plt.figure(figsize=(12,12))

#Hàm để hiển thị giữa 1 đặc trưng với cột Sale Price
def scatter_plot(a):
    fig, ax = plt.subplots()
    ax.scatter(x = df_train[a], y = df_train['SalePrice'], edgecolor=edgecolor)
    plt.ylabel('SalePrice', fontsize=12)
    plt.xlabel(a, fontsize=12)
    plt.suptitle("Scatter Plot of "+ a + " and SalePrice")
    plt.show()
```

*Hình 3.15: Kiểm tra ngoại lệ*

Hiển thị cột GrLiveArea và cột Sale Price

```
scatter_plot('GrLivArea')
```



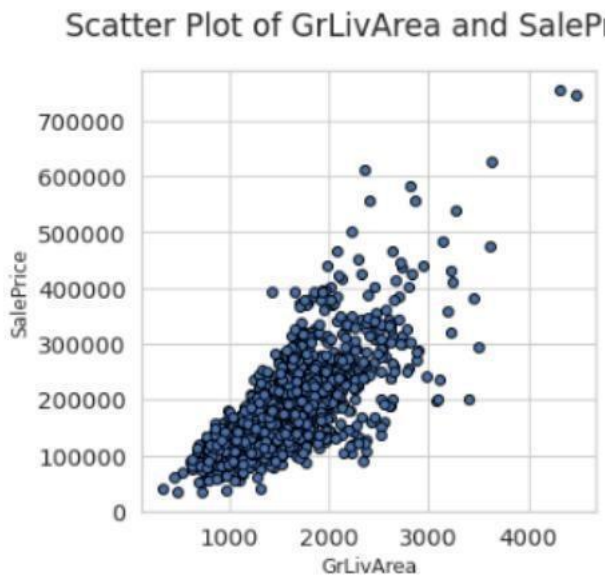

*Hình 3.16: Hiển thị cột GrLiveArea và cột Sale price*

Có thể nhận thấy rằng có những ngoại lệ lớn có thể ảnh hưởng tiêu cực đến dự đoán giá bán cao

- Vì vậy, các ngoại lệ cần được xóa

- Xóa các ngoại lệ

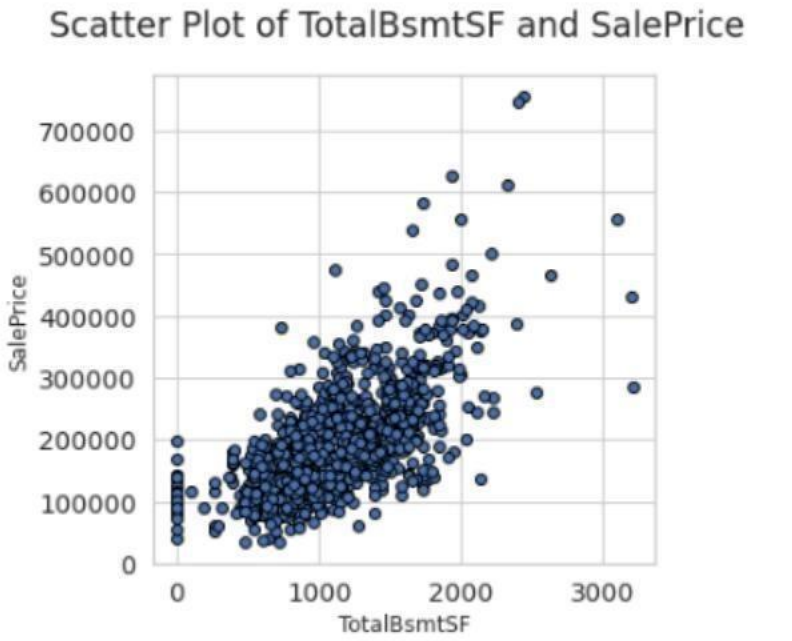
```
# Xóa các ngoại lệ
df_train =  df_train.drop( df_train[( df_train['GrLivArea'] > 4000) & ( df_train

# Kiểm tra lại hình ảnh
scatter_plot('GrLivArea')
```




*Hình 3.17: Xóa các ngoại lệ*

- Hiển thị cột TotalBsmtSF và SalePrice

```
scatter_plot('TotalBsmtSF')
```

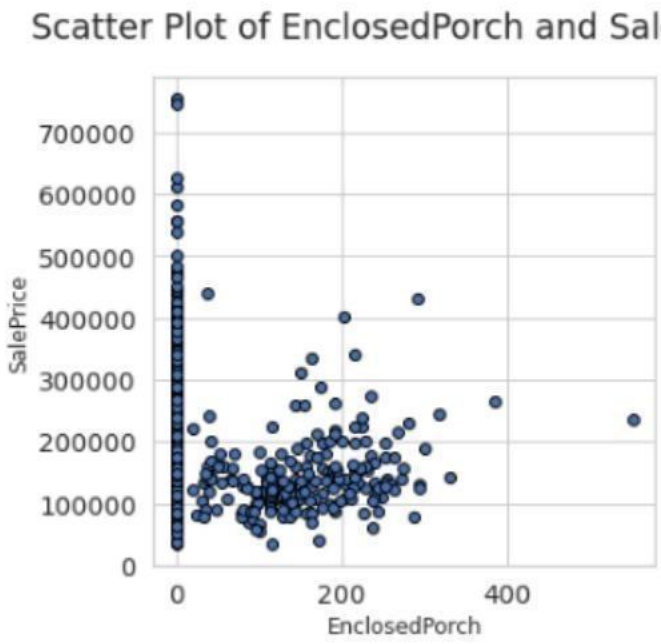



*Hình 3.18: Hiển thị cột TotalBsmtSF và SalePrice*

Không có ngoại lệ quá lớn, chúng tôi không cần xóa bất kỳ điểm nào

Hiển thị cột Enclosed Porch

```
scatter_plot('EnclosedPorch')
```




*Hình 3.19: Hiển thị cột Enclosed Porch*

- Có một số ngoại lệ cần được xóa để nó không ảnh hưởng nhiều đến dự đoán của chúng tôi

- Xóa ngoại lệ

```
# Xóa các ngoại lệ
df_train =  df_train.drop( df_train[( df_train['EnclosedPorch']>400)].index)

# Xóa các ngoại lệ
df_train =  df_train.drop( df_train[( df_train['SalePrice']>700000)].index)

# Kiểm tra lại hình ảnh
scatter_plot('EnclosedPorch')
```

*Hình 3.20: Xóa các ngoại lệ còn lại*

Xử lý dữ liệu bị thiếu

Xử lý dữ liệu bị thiếu

```
# Hàm hiển thị những data bị thiếu
def missing_data(df,n):
    total = df.isnull().sum().sort_values(ascending=False)          # Total No of
    percentage = (df.isnull().sum() / df.isnull().count()).sort_values(ascending=F
    No_unique_val = df.nunique()                                    # No of unique
    missing_data = pd.concat([total, percentage, No_unique_val], axis=1,
                             keys=['Total No of missing val', '% of Missing val','

    print(missing_data.head(n))
```

*Hình 3.21: Xử lý dữ liệu bị thiếu*

- Thêm vào tập training

```
#training data
missing_data(df_train,20)
```

| | Total No of missing val | % of Missing val | No of unique val |
|---|---|---|---|
| PoolQC | 1451 | 99.725086 | 2 |
| MiscFeature | 1401 | 96.288660 | 4 |
| Alley | 1364 | 93.745704 | 2 |
| Fence | 1176 | 80.824742 | 4 |
| FireplaceQu | 690 | 47.422680 | 5 |
| LotFrontage | 259 | 17.800687 | 109 |
| GarageYrBlt | 81 | 5.567010 | 97 |
| GarageCond | 81 | 5.567010 | 5 |
| GarageType | 81 | 5.567010 | 6 |
| GarageFinish | 81 | 5.567010 | 3 |
| GarageQual | 81 | 5.567010 | 5 |
| BsmtExposure | 38 | 2.611684 | 4 |
| BsmtFinType2 | 38 | 2.611684 | 6 |
| BsmtCond | 37 | 2.542955 | 4 |
| BsmtQual | 37 | 2.542955 | 4 |
| BsmtFinType1 | 37 | 2.542955 | 6 |
| MasVnrArea | 8 | 0.549828 | 324 |
| MasVnrType | 8 | 0.549828 | 4 |
| Electrical | 1 | 0.068729 | 5 |
| MSSubClass | 0 | 0.000000 | 15 |

*Hình 3.22: Thêm vào tập training*

### *3.3.4. Thiết lập mô hình*

Chia tập dữ liệu train và test . Ta chia dữ liệu train là 80% của bộ dữ liệu, còn lại là của test, sau đó ta trộn dữ liệu lên random_state = **42.**

```
X = df_train.drop(['SalePrice'], axis=1)
Y = df_train['SalePrice']

# chia tập data thành 2 bộ train, validation . train chiếm 70%
X_train, X_test , Y_train, Y_test = train_test_split(X,Y,train_size= 0.7, random_state=42, shuffle=True)

print("X_train shape: {}".format(X_train.shape))
print("y_train shape: {}".format(Y_train.shape))
print()
print("X_valid shape: {}".format(X_test.shape))
print("y_valid shape: {}".format(Y_test.shape))
print()

X_train shape: (1018, 80)
y_train shape: (1018,)

X_valid shape: (437, 80)
y_valid shape: (437,)
```

*Hình 3.23: Chia dữ liệu train và test*

- Training với thuật toán LinearRegression

```
lin_reg = LinearRegression()
lin_reg.fit(X_train, y_train)
predictions = lin_reg.predict(X_test)

mae, mse, rmse, r_squared = evaluation(y_test, predictions)
print("MAE:", mae)
print("MSE:", mse)
print("RMSE:", rmse)
print("R2 Score:", r_squared)
print("-"*30)
rmse_cross_val = rmse_cv(lin_reg)
print("RMSE Cross-Validation:", rmse_cross_val)
```

*Hình 3.24: Mô hình Linear Regression*

- Training với thuật toán Lasso

```
lasso = Lasso()
lasso.fit(X_train, y_train)
predictions = lasso.predict(X_test)

mae, mse, rmse, r_squared = evaluation(y_test, predictions)
print("MAE:", mae)
print("MSE:", mse)
print("RMSE:", rmse)
print("R2 Score:", r_squared)
print("-"*30)
rmse_cross_val = rmse_cv(lasso)
print("RMSE Cross-Validation:", rmse_cross_val)
```

*Hình 3.25: Mô hình hồi quy Lasso*

- Training với thuật toán Ridge

```
ridge = Ridge()
ridge.fit(X_train, y_train)
predictions = ridge.predict(X_test)

mae, mse, rmse, r_squared = evaluation(y_test, predictions)
print("MAE:", mae)
print("MSE:", mse)
print("RMSE:", rmse)
print("R2 Score:", r_squared)
print("-"*30)
rmse_cross_val = rmse_cv(ridge)
print("RMSE Cross-Validation:", rmse_cross_val)
```

*Hình 3.26: Mô hình hồi quy Ridge*

- Training với thuật toán Elastic Net

```
elastic_net = ElasticNet()
elastic_net.fit(X_train, y_train)
predictions = elastic_net.predict(X_test)

mae, mse, rmse, r_squared = evaluation(y_test, predictions)
print("MAE:", mae)
print("MSE:", mse)
print("RMSE:", rmse)
print("R2 Score:", r_squared)
print("-"*30)
rmse_cross_val = rmse_cv(elastic_net)
print("RMSE Cross-Validation:", rmse_cross_val)
```

*Hình 3.27: Mô hình hồi quy Elastic Net*

# CHƯƠNG 4: KẾT QUẢ THỰC NGHIỆM VÀ KIỂM THỬ

## 4.1. Khái quát về kiểm thử

Mục tiêu kiểm thử của chúng em nhằm:

- Kiểm tra xem sản phẩm đã đáp ứng tất cả yêu cầu đã mô tả chưa hoặc xác nhận xem phần mềm đã hoàn thiện và hoạt động đúng như mong đợi của người dùng và các bên liên quan khác chưa.
- Kiểm thử để ngăn ngừa lỗi bằng cách review tài liệu mô tả yêu cầu hay thiết kế hệ thống, bao gồm mã nguồn (source code) để phát hiện lỗi sớm.
- Kiểm thử nhằm nâng cao chất lượng sản phẩm, tăng sự tin tưởng của khách hàng đối với phần mềm thông qua việc phát hiện lỗi và sửa lỗi (và dĩ nhiên là phải kiểm thử lại sau khi được sửa).
- Cung cấp thông tin cho các bên liên quan (bao gồm quản lý dự án hay khách hàng tùy dự án) để họ có thể đưa ra quyết định về việc phát hành (release) một phần mềm nào đó hay không
- Giảm thiểu rủi ro do lỗi của phần mềm gây ra trong quá trình sử dụng thực tế.

### *4.1.1. Các mức kiểm thử*

Các mức kiểm thử là tập hợp các hoạt động test được thực hiện ở một giai đoạn phát triển cụ thể của sản phẩm, ví dụ từ lúc vài chức năng nhỏ

được lập trình, cho đến lúc sản phẩm hoàn thiện, và có thể tích hợp với các hệ thống khác. Dưới đây là một số mức kiểm thử cơ bản:

- *Kiểm thử (mức) đơn vị:*

Đây là mức kiểm thử thấp nhất. Ở mức kiểm thử này, các thành phần nhỏ nhất của một hệ thống phần mềm như class, function, hay module sẽ được kiểm thử riêng lẻ. Phần lớn test case (các trường hợp kiểm thử) ở mức này sẽ gọi trực tiếp đối tượng được kiểm thử (như class, function… nào đó) trong code (mã nguồn). Ở mức này nhóm đã kiểm thử, và chạy lại các dòng code để xem xét các vấn đề có xảy ra với chương trình của mình không.

- Kiểm thử tích hợp:

Ở giai đoạn này, các chức năng (hoặc class, function,…) đã được phát triển và kiểm thử xong, đã đến lúc tích hợp chúng lại với nhau. Và phải kiểm thử sự tương tác qua lại giữa các thành phần này. Tương tự như vậy, chúng em cũng cần kiểm thử sự tích hợp giữa các hệ thống khác nhau. Chúng em đã kiểm thử tích hợp giữa các hệ thống độc lập có liên quan đến nhau, để đảm bảo rằng chúng đáp ứng các luồng nghiệp vụ mong muốn.

- Kiểm thử hệ thống:

Một khi hệ thống đã được phát triển (lập trình) hoàn thiện, chúng ta cần kiểm thử các luồng xử lý chức năng, và một số khía cạnh phi chức năng như hiệu năng (trong đó có load testing – kiểm thử tải) và đánh giá tính khả dụng của hệ thống (usability – xem dễ sử dụng dễ hiểu cho mọi đối tượng người dùng hay không).

Thường sẽ dựa vào tài liệu mô tả chức năng và áp dụng các kỹ thuật kiểm thử hộp đen để viết test case (là end-to-end test case) cho mức kiểm thử này ○ Kiểm thử chấp nhận:

Nhóm đã đánh giá độ hài lòng của hệ thống bằng cách gửi các mẫu google front đến các khách hàng để họ đánh giá về sự phát triển của hệ thống. Qua đó khắc phục và sửa lại để phục vụ khách hàng.

#### *4.1.2. Kiểm thử chức năng*

Các hoạt động kiểm thử chức năng sẽ tập trung vào việc kiểm tra xem hệ thống có hoạt động theo đúng theo yêu cầu nghiệp vụ đã mô tả hay chưa. Tập trung đánh giá tính đúng đắn, mức độ chính xác, và tính phù hợp của hệ thống với người dùng, và một số khía cạnh khác. Kiểm thử chức năng có thể được thực hiện ở tất cả các mức kiểm thử.

### 4.2. Kiểm thử mô hình học máy

Để kiểm thử dự đoán giá nhà bằng mô hình học máy, bạn cần có một mô hình đã được huấn luyện trên dữ liệu giá nhà và sẵn sàng để dự đoán. Để kiểm thử thuật toán một cách hiệu quả cần chuẩn bị dữ liệu để kiểm thử. Trong bài toán trên 80% dữ liệu để huấn luyện mô hình và 20% để kiểm tra độ chính xác mô hình.

Đánh giá mô hình bằng tập kiểm tra bằng cách sử dụng các độ đo như độ lỗi(mean squared error), độ chính xác(accuracy) hoặc các độ đo khác để đánh giá kết quả mô hình.

Để kiểm tra mô hình cho bài toán dự đoán giá nhà có thể sử dụng các độ đo MAE (Mean Absolute Error), MSE (Mean Squared Error), RMSE (Root Mean

Squared Error), và R2 Score (R-squared Score). Đây là các độ đo để đánh giá hiệu suất của mô hình dự đoán trong bài toán hồi quy.

### *4.2.1. MAE(Mean Absolute Error)*

MAE đo lường trung bình giá trị tuyệt đối của sự chênh lệch giữa các giá trị dự đoán và giá trị thực tế. Nó được tính bằng cách lấy trung bình của các giá trị tuyệt đối của các sai số. MAE cung cấp thông tin về độ lớn trung bình của sai số dự đoán.

### *4.2.2. MSE (Mean Squared Error)*

MSE tính bình phương trung bình của sai số giữa các giá trị dự đoán và giá trị thực tế. Điều này đồng nghĩa với việc nó tính toán trung bình của bình phương sai số. MSE thường nhạy cảm với các giá trị lỗi lớn.

$$MSE = \frac{1}{n}\Sigma\underbrace{\left(y - \widehat{y}\right)}_{\text{The square of the difference between actual and predicted}}^{2}$$

$$\mathbf{MAE} = \frac{\mathbf{1}}{\boldsymbol{n}}\sum_{\boldsymbol{i=1}}^{\boldsymbol{n}}|\boldsymbol{x}_{\boldsymbol{i}} - \boldsymbol{x}|$$

### *4.2.3. RMSE (Root Mean Squared Error)*

RMSE là căn bậc hai của MSE. Nó được sử dụng để đo lường độ lớn trung bình của sai số dự đoán, và cùng đơn vị với giá trị thực tế. RMSE thường được ưa chuộng hơn MSE vì nó có cùng đơn vị với biến phụ thuộc và dễ hiểu hơn.

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

### *4.2.4. R2 Score (R-squared Score)*

R2 Score đo lường mức độ giải thích của mô hình đối với biến phụ thuộc. Giá trị R2 Score nằm trong khoảng từ 0 đến 1, trong đó giá trị 1 đại diện cho một mô hình hoàn hảo, trong khi giá trị 0 chỉ ra rằng mô hình không giải thích được biến phụ thuộc. R2 Score càng gần 1, mô hình càng tốt.

$$R^2 = 1 - \frac{SS_{RES}}{SS_{TOT}} = 1 - \frac{\sum_i (y_i - \hat{y}_i)^2}{\sum_i (y_i - \bar{y})^2}$$

Các độ đo này được sử dụng để so sánh hiệu suất giữa các mô hình khác nhau hoặc để đánh giá hiệu suất của một mô hình so với giá trị thực tế. Tuy nhiên, việc lựa chọn độ đo phù hợp phụ thuộc vào bài toán cụ thể và yêu cầu của quá trình dự đoán. Sử dụng cả bốn độ đo sẽ giúp chúng ta có cái nhìn tổng quan về hiệu suất của mô hình và hiểu rõ hơn về các sai số của mô hình.

### *4.2.5. Phương pháp Cross Validation*

Để có kết quả đánh giá mô hình một cách chính xác ta sẽ sử dụng kỹ thuật Cross-validation.

Cross-validation là một kỹ thuật quan trọng trong machine learning để đánh giá hiệu suất của mô hình dự đoán trên dữ liệu mới mà mô hình chưa từng thấy trước đó. Nó giúp ước lượng khả năng tổng quát hóa (generalization) của mô hình.

Trong quá trình cross-validation, tập dữ liệu được chia thành các tập con (folds) có kích thước tương đương. Mô hình được huấn luyện trên một số folds (thường gọi là training set) và được đánh giá trên fold còn lại (thường gọi là validation set hoặc test set). Quá trình này được lặp lại nhiều lần, với mỗi lần sử dụng các folds khác nhau để huấn luyện và đánh giá mô hình.

Có một số phương pháp cross-validation phổ biến:

1. K-fold Cross-Validation: K-fold cross-validation là một trong những kỹ thuật cross-validation phổ biến nhất. Trong K-fold cross-validation, tập dữ liệu được chia thành K folds có kích thước bằng nhau. Mô hình được huấn luyện trên K-1 folds và được đánh giá trên fold còn lại. Quá trình này được lặp lại K lần, trong mỗi lần sử dụng một fold khác nhau để đánh giá. Kết quả đánh giá từ K lần lặp lại được kết hợp để đưa ra ước lượng tổng quát về hiệu suất của mô hình.
2. Stratified K-fold Cross-Validation: Stratified K-fold cross-validation là một biến thể của K-fold cross-validation, đặc biệt hữu ích khi mục tiêu là phân loại (classification). Stratified K-fold cross-validation đảm bảo rằng tỷ lệ các lớp trong mỗi fold được duy trì tương tự như tỷ lệ trong toàn bộ tập dữ liệu. Điều này giúp đảm bảo rằng mỗi fold đại

diện cho một phân phối lớp tốt hơn và ước lượng hiệu suất của mô hình trên các lớp là công bằng.

3. Leave-One-Out Cross-Validation (LOOCV): Leave-One-Out Cross-Validation là một phương pháp cross-validation đặc biệt, trong đó mỗi mẫu trong tập dữ liệu được sử dụng lần lượt làm validation set, và các mẫu còn lại được sử dụng làm training set. Điều này có nghĩa là chúng ta huấn luyện mô hình K lần (với K bằng số lượng mẫu trong tập dữ liệu), và đánh giá mô hình trên từng mẫu một. LOOCV rất tốn kém về mặt tính toán, nhưng nó có thể cung cấp một ước lượng chính xác về hiệu suất của mô hình.

Cross-validation giúp đánh giá mô hình một cách công bằng và đáng tin cậy, bởi vì nó đảm bảo rằng mô hình được đánh giá trên các dữ liệu độc lập mà nó chưa từng thấy. Điều này giúp tránh việc mô hình chỉ tốt trên dữ liệu huấn luyện nhưng không tổng quát hóa tốt trên dữ liệu mới.

## 4.3. Các kết quả kiểm thử

Trước tiên, để kiểm tra tính ổn định cũng như tốc độ hội tụ của mô hình hồi quy tuyến tính được đề xuất ở trên. Tôi tiến hành kiểm thử với bộ dữ liệu“test.csv”. Để kiểm tra chênh lệch giữa giá nhà thực tế và giá nhà dự đoán chúng em sử dụng các độ đo MAE, MSE, RMSE và R2 Score. Để có thể kiểm tra cho tất cả các mô hình: Hồi quy tuyến tính, Hồi quy Lasso, Hồi quy Ridge. Hồi quy đa thức. Chúng em xây dựng một hàm để tính lần lượt các độ đo và tổng hợp kết quả đánh giá ở bảng dưới.

```python
def rmse_cv(model):
    rmse = np.sqrt(-cross_val_score(model, X, y, scoring="neg_mean_squared_error", cv=5)).mean()
    return rmse


def evaluation(y, predictions):
    mae = mean_absolute_error(y, predictions)
    mse = mean_squared_error(y, predictions)
    rmse = np.sqrt(mean_squared_error(y, predictions))
    r_squared = r2_score(y, predictions)
    return mae, mse, rmse, r_squared
```

*Hình 4.1: Các kết quả kiểm thử*

Hai hàm ở trên được sử dụng để tính các số đo với từng model. Hàm rmse_cv tính độ đo RMSE bằng phương pháp cross-validation với k = 5. Hàm evaluation để tính các độ đo MAE, MSE, RMSE, R2 SCORE trên bộ dữ liệu test.csv. Chi tiết số liệu được thể hiện ở bảng dưới đây.

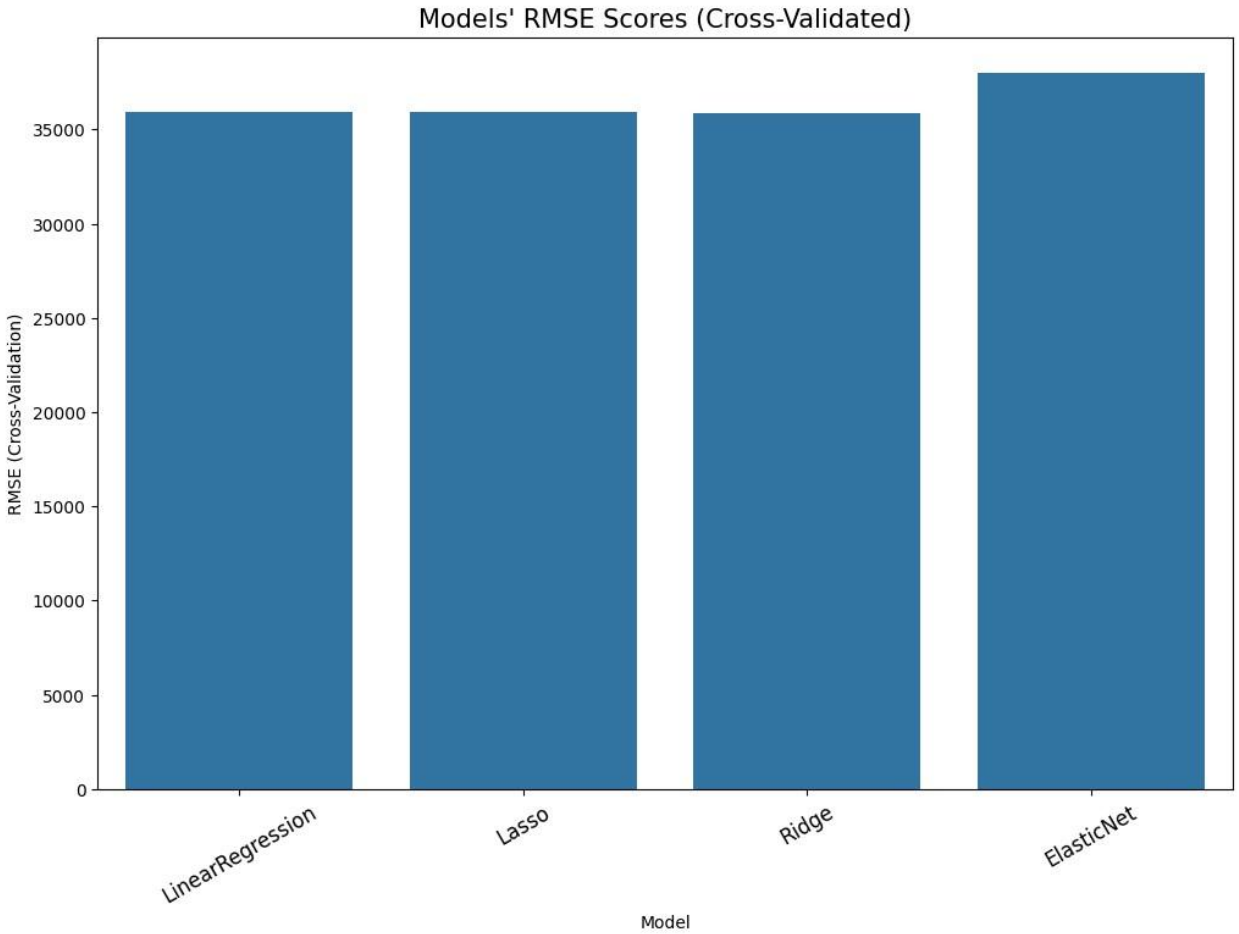
| Model | MAE | MSE | RMSE | R2 Score | RMSE (Cross-Validation) |
|---|---|---|---|---|---|
| Linear Regression | 23567.89057 | 1414931405 | 37615.57396 | 0.815531782 | 35933.12898 |
| Lasso | 23560.45808 | 1414337629 | 37607.68045 | 0.815609194 | 35923.05279 |
| Ridge | 23435.50371 | 1404264217 | 37473.51354 | 0.816922491 | 35891.49964 |
| ElasticNet | 23792.74378 | 1718445790 | 41454.14081 | 0.775961837 | 37997.25671 |

Kết quả đánh giá cho mô hình hồi quy tuyến tính cho thấy rằng mô hình có độ chính xác tương đối cao. Giá trị RMSE Cross Validation là 36236 cho thấy sai số giữa giá trị dự đoán và giá trị thực tế là khá thấp. Mô hình có khả năng giải thích dữ liệu tốt với R2 Score là 0.8155. Dựa vào kết quả trên cho thấy mô hình có kết quả tốt trên tập dữ liệu kiểm tra.

Mô hình Elastic Net có độ đo RMSE Cross Validation lớn nhất là 38449 và R2 Score bằng 0.7759 cho thấy mô hình có kết quả dự đoán kém nhất trong bốn mô hình hồi quy điều này có thể do số lượng biến đầu vào quá lớn so với lượng dữ liệu mẫu khiến cho mô hình có kết quả không tốt trên tập kiểm tra.

Mô hình hồi quy Lasso và hồi quy Ridge thể hiện kết quả tốt hơn với các mô hình hồi quy còn lại với độ đo RMSE thấp hơn và R2 Score cao hơn mô hình hồi quy tuyến tính và Elastic Net. Hồi quy Ridge thể hiện kết quả tốt nhất trong các mô hình.

## 4.4. Kết quả thực nghiệm



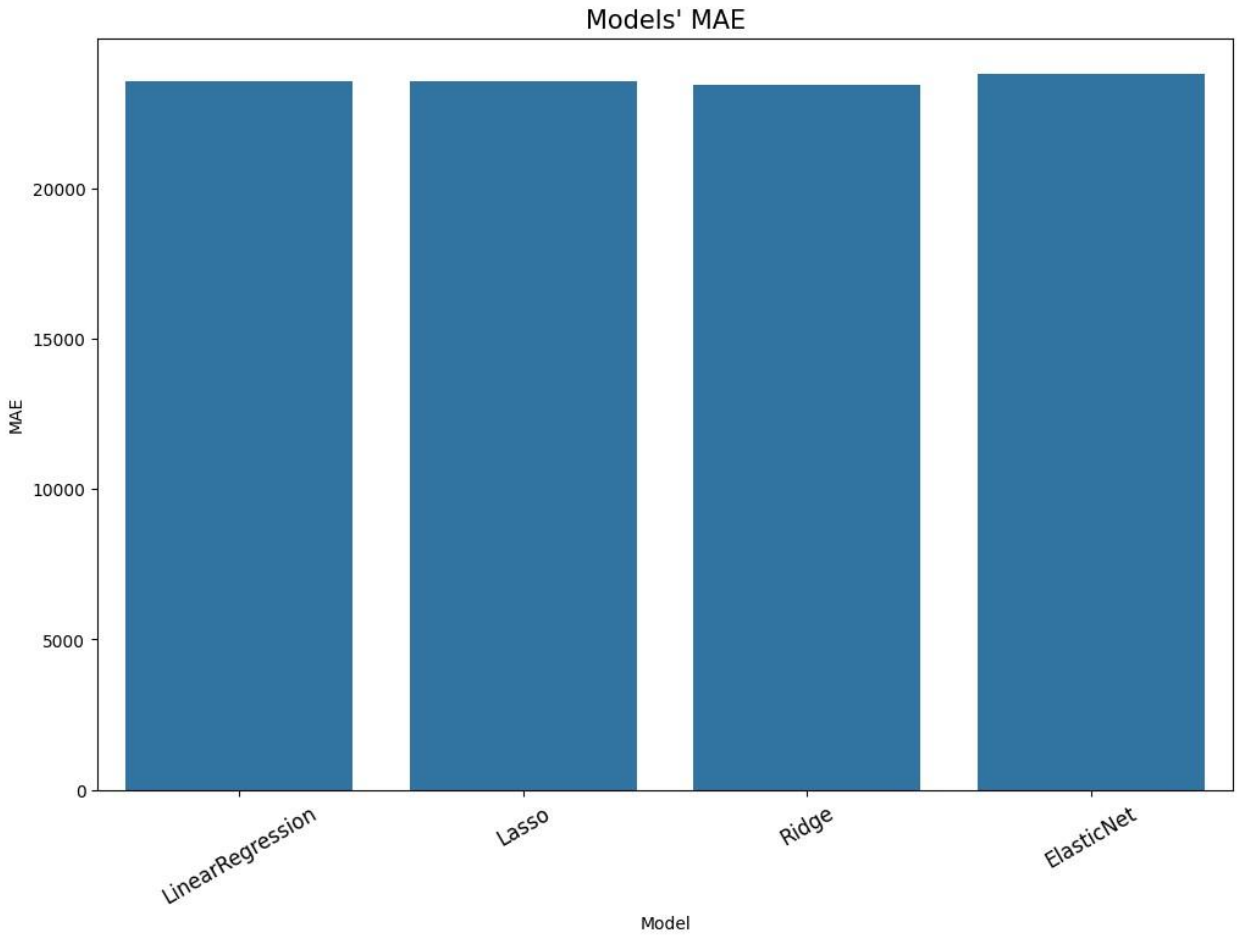
*Hình 4.2: Biểu thị kết quả chênh lệch của kết quả thực tế và kết quả của các mô hình hồi quy dựa trên độ đo RMSE (Cross-Validated)*

Trong bài toán dự đoán giá nhà, với RMSE (Cross-Validated) càng nhỏ thì mô hình có khả năng dự đoán chính xác trên dữ liệu mới chính xác hơn. Mô hình hồi quy Ridge được đề xuất đảm bảo được kết quả tốt khi có kết quả tốt nhất với bộ dữ liệu train và bộ dữ liệu test.

Qua kết quả trên có thể thấy rằng việc huấn luyện của 3 mô hình hồi quy tuyến tính, hồi quy Ridge và hồi quy Lasso tốt hơn mô hình Elastic Net với độ chênh lệch trong khoảng [35800, 36300]. Bằng cách giảm giá trị của các hệ số hồi quy sẽ khiến mô hình hồi quy Ridge và hồi quy Lasso phức tạp hơn mô hình hồi quy tuyến tính nhưng sẽ cho kết quả tốt hơn trên tập dữ liệu kiểm tra.

Mô hình hồi quy Elastic Net có kết quả không tốt như các mô hình còn lại là do đặc điểm của dữ liệu đầu vào. Dữ liệu đầu vào không có sự khác biệt rõ ràng giữa các thuộc quan trọng sẽ khiến việc kết hợp hồi quy Ridge và hồi quy Lasso trở nên không hiệu quả như việc sử dụng riêng biệt từng mô hình.

Không chỉ đánh giá trên mỗi RMSE (Cross-Validated), nhóm em cũng đã tiến hành trên nhiều độ đo khác nhau. Các kết quả trong Hình 4.2 được thực hiện trên bộ dữ liệu kiểm tra (test set) của các mô hình hồi quy nêu trên.




*Hình 4.3: Đo độ chênh lệch trung bình giữa giá trị dự đoán và giá trị thực tế bằng MAE*

Trong hình 4.3, chúng em sử dụng độ đo MAE để kiểm tra độ chênh lệch trung bình giữa giá trị dự đoán và giá trị thực tế của các mô hình hồi quy tuyến tính, hồi quy Ridge, hồi quy Lasso và hồi quy Elastic Net. Khi sử dụng độ đo trên có thể thấy rằng giá trị chênh lệch trung bình giữa bốn mô hình hồi quy là hồi quy tuyến tính, hồi quy Lasso, hồi quy Elastic Net và hồi quy Ridge có kết quả xấp xỉ nhau với độ chênh lệch không quá 2%.

Các kết quả thử nghiệm trên các độ đo trên cho thấy: mô hình hồi quy Ridge đề xuất thể hiện tốt trên cả bộ dữ liệu train và test. Mô hình hồi quy Ridge có kết

quả tốt nhất trên tất cả các độ đo. Mô hình hồi quy tuyến tính có kết quả gần như xếp xỉ mô hình hồi quy Lasso và hồi quy Elastic Net, tuy nhiên có độ chênh lệch lớn hơn 5% khi so với mô hình hồi quy Ridge.

# KẾT LUẬN

Đến đây cũng đã kết thúc bài báo cáo của nhóm về đề tài nghiên cứu **" Dự đoán giá nhà bằng mô hình hồi quy tuyến tính"** với sự đóng góp của các thành viên trong nhóm thực hiện. Nhìn lại quá trình thực hiện bài báo cáo này, nhóm thấy được những điểm mạnh và điểm yếu còn vướng mắc về kiến thức cũng như kỹ năng trình bày và tìm hiểu các tài liệu liên quan. Để rút kinh nghiệm cho những lần thực hiện sau và cải thiện chất lượng của bài báo cáo sau sửa đổi, nhóm đi đến vạch ra những ưu nhược điểm của cá nhân cũng như tập thể nhóm thực hiện đề tài về những gì nhóm làm, những gì nhóm đã đạt được, những gì còn thiếu sót và định hướng sẽ thực hiện trong tương lai.

Với nhận thức về vấn đề xã hội ngày càng phổ biến với những kết nối giữa tri thức và con người, giữa thế giới của trí tuệ nhân tạo với kinh nghiệm thực tiễn của chúng ta, áp dụng khoa học kỹ thuật là điều tất yếu. Xây dựng một đề tài để mang ứng dụng của nó vào thực tiễn nhằm cải thiện một lĩnh vực nào đó trong đời sống là mong muốn của thế hệ sinh viên nói chung và sinh viên ngành Công nghệ thông tin. Chính vì vậy, nhóm đã làm việc dưới sự chỉ đạo, cố vấn của giảng viên hướng dẫn để thực hiện đề tài **" Dự đoán giá nhà bằng mô hình hồi quy tuyến tính "**.

Từ những phân công trong nhóm, cá nhân các thành viên đã làm việc độc lập trong sự thống nhất để đi đến hoàn thành những nhiệm vụ đặt ra khi thực hiện đề tài.

- Nhóm đã tìm hiểu và xác định được đề tài nghiên cứu của mình thông qua việc khảo sát thị trường giá nhà bất động sản ở Việt Nam. Từ đó, xây dựng bài toán cụ thể về dự đoán giá nhà với tổng quan đã nêu ở ***Chương 1 (mục***

***I).***

- Đồng thời phân tích kỹ thuật sử dụng trong đề tài, các kiến trúc mạng sử dụng và các đặc trưng của kiến trúc đã nêu ở ***Chương 2 (mục I, mục II)***
- Nhóm cũng đã nêu ra được những dữ liệu sẽ sử dụng để học từ những nguồn tham khảo và nêu ra các thuật toán sử dụng trong hệ thống nhằm xử lý và phân tích dữ liệu, đi đến phân loại sau máy học đã nêu ở ***Chương 3.***
- Từ những dữ liệu mà nhóm đã tiến hành kiểm thử và đưa ra được những kết quả và debug nêu ở ***Chương 4.***

Tuy có những đóng góp tích cực nhằm hoàn thiện báo cáo, tuy nhiên không thể tránh khỏi những thiếu sót, hạn chế. Cụ thể vấn đề nằm ở dữ liệu huấn luyện sử dụng cho máy học còn hạn chế, điều này khiến cho model trở nên overfitting dẫn đến khả năng áp dụng thực tiễn có mức độ khả thi không cao.

Nhằm khắc phục những hạn chế kể trên, nhóm sẽ cần tìm hiểu và thêm nhiều dữ liệu và đảm bảo độ chính xác dữ liệu cao hơn để nâng cao tính khả thi khi áp dụng vào thực tiễn.

Cuối cùng với những đóng góp, cố gắng để hoàn thiện báo cáo, nhóm rất mong được sự góp ý chân thành, quý báu từ quý thầy cô và các bạn đọc. Nhóm xin chân thành cảm ơn!

***Nhóm thực hiện đề tài!***

# TÀI LIỆU THAM KHẢO

1. Tác phẩm sách:

[1] Hoàng Xuân Huấn. Giáo Trình Học Máy (NXB Đại Học Quốc Gia 2015).

[2] Nguyễn Phương Nga, Trần Hùng Cường. Giáo Trình Trí Tuệ Nhân Tạo Đại Học Công Nghiệp (NXB Thống Kê 2021).

[3] M. Emre Celebi và Bogdan Smolka. (2016). Convolutional Neural Networks in Visual Computing: A Concise Guide.

[4] Adrian Rosebrock. (2016). Practical Python and OpenCV.

[5] Kevin P. Murphy (2012). Machine Learning: A Probabilistic Perspective.

2. Tài liệu học thuật trực tuyến (ebook, học liệu trực tuyến): [1] Nguyễn Thanh Tuấn. (2020). Deep Learning Cơ Bản. https://nttuan8.com/sach-deep-learning-co-ban

[2] Vũ Hữu Tiệp. (2020). Machine Learning cơ bản. https://machinelearningcoban.com/ebook/

3. Website:

[1] https://machinelearningcoban.com/2016/12/28/linearregression/

[2] Linear Regression - Wikipedia