 BỘ GIÁO DỤC VÀ ĐÀO TẠO

TRƯỜNG ĐẠI HỌC VĂN HIẾN

**KHOA KỸ THUẬT – CÔNG NGHỆ**

**Logo

Description automatically generated**

**HỌC MÁY**

**ĐỀ TÀI :** **22 Leaen How to Build a Linear**

**Regression Model in Pytorch**

**GVGD: Ths. Hồ Nhựt Minh**

**SVTH: Cao Hồng Sơn - 221A010224**

**Nguyễn Văn Quy - 221A020003**

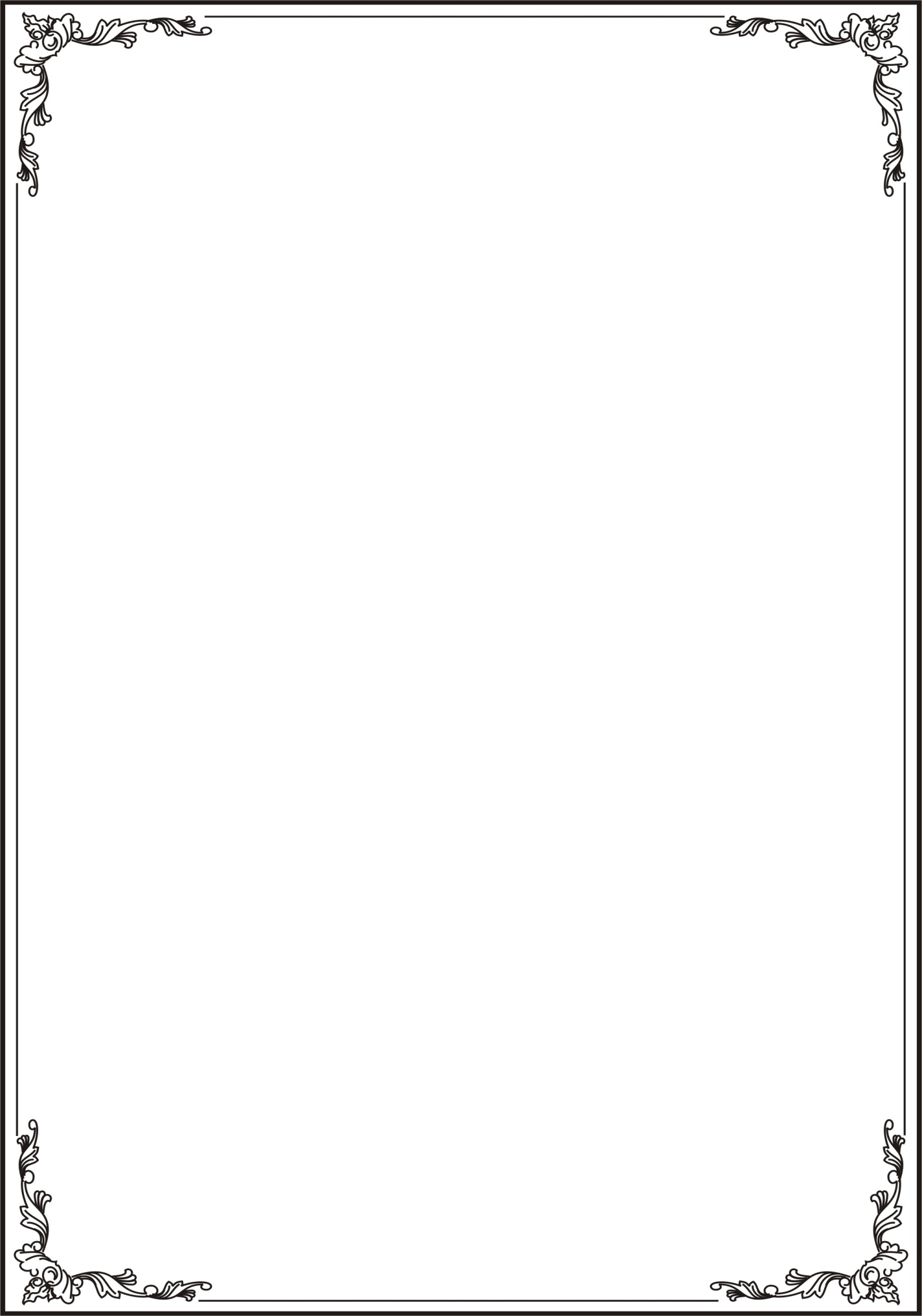
**Phạm Phúc Lộc - 221A240047**

**Lê Minh Thông - 221A300054**

**Dương Thi Sỹ - 221A020001**

**Đỗ Quốc Hưng - 221A020027**

**TP. HỒ CHÍ MINH - 2024**

 BỘ GIÁO DỤC VÀ ĐÀO TẠO

TRƯỜNG ĐẠI HỌC VĂN HIẾN

**KHOA KỸ THUẬT – CÔNG NGHỆ**

**Logo

Description automatically generated**

**HỌC MÁY**

**ĐỀ TÀI : 22 Leaen How to Build a Linear**

**Regression Model in Pytorch**

|  |  |  |  |
| --- | --- | --- | --- |
| **STT** | **HỌ VÀ TÊN SINH VIÊN** | **MSSV** | **CHỮ KÝ** |
| **1** | **Cao Hồng Sơn** | **221A010224** |  |
| **2** | **Nguyễn Văn Quy** | **221A020003** |  |
| **3** | **Dương Thi Sỹ** | **221A020001** |  |
| **4** | **Phạm Phúc Lộc** | **221A240047** |  |
| **5** | **Lê Minh Thông** | **221A300054** |  |
| **6** | **Đỗ Quốc Hưng** | **221A020027** |  |

**TP. HỒ CHÍ MINH - 2024**

**MỤC LỤC**

[LỜI CẢM ƠN 4](#_Toc180372574)

[NHẬN XÉT CỦA GIẢNG VIÊN 5](#_Toc180372575)

[TỔNG QUAN 6](#_Toc180372576)

[A. Mục Tiêu 6](#_Toc180372577)

[B. Mô Tả Dữ Liệu 6](#_Toc180372578)

[C. Công Nghệ Sử Dụng 6](#_Toc180372579)

[D. Cách Tiếp Cận 6](#_Toc180372580)

[E. Tổng Quan Mã Modular 7](#_Toc180372581)

[KẾT LUẬN 8](#_Toc180372582)

[1. Tensor là gì? 8](#_Toc180372583)

[2. Sự khác biệt giữa Tensor PyTorch và mảng Numpy 8](#_Toc180372584)

[2.1. Tính năng 8](#_Toc180372585)

[2.2. Hiệu suất 9](#_Toc180372586)

[2.3. Cú pháp và API 9](#_Toc180372587)

[2.4. Khả năng tương thích 10](#_Toc180372588)

[2.5. Tính linh hoạt 10](#_Toc180372589)

[3. Tiền xử lý dữ liệu là gì? 10](#_Toc180372590)

[3.1. Làm sạch dữ liệu 10](#_Toc180372591)

[3.2. Chuyển đổi dữ liệu 11](#_Toc180372592)

[3.3. Mã hóa dữ liệu 11](#_Toc180372593)

[3.4. Tách dữ liệu 11](#_Toc180372594)

[3.5. Giảm chiều dữ liệu 11](#_Toc180372595)

[3.6. Tạo đặc trưng (Feature Engineering) 11](#_Toc180372596)

[3.7. Kiểm tra và đánh giá 12](#_Toc180372597)

[4. Các đặc điểm phân loại và số 12](#_Toc180372598)

[4.1. Đặc điểm phân loại (Categorical Variables) 12](#_Toc180372599)

[4.2. Đặc điểm số (Numeric Variables) 13](#_Toc180372600)

[4.3. Ứng dụng trong phân tích dữ liệu 13](#_Toc180372601)

[5. Tại sao chúng ta cần tiền xử lý dữ liệu? 13](#_Toc180372602)

[5.1. Cải thiện chất lượng dữ liệu 13](#_Toc180372603)

[5.2. Tăng cường độ chính xác của mô hình 14](#_Toc180372604)

[5.3. Tiết kiệm thời gian và tài nguyên 14](#_Toc180372605)

[5.4. Tăng cường khả năng phân tích 14](#_Toc180372606)

[5.5. Phù hợp với các thuật toán học máy 14](#_Toc180372607)

[5.6. Đảm bảo tính pháp lý và đạo đức 14](#_Toc180372608)

[6. Cách thực hiện mã hóa nhãn cho các đặc điểm phân loại? 15](#_Toc180372609)

[7. Cách chuẩn hóa các đặc điểm số? 16](#_Toc180372610)

[7.1. Chuẩn hóa Min-Max (Min-Max Normalization) 16](#_Toc180372611)

[7.2. Chuẩn hóa Z-Score (Z-Score Normalization) 16](#_Toc180372612)

[7.3. Chuẩn hóa Robust (Robust Scaling) 16](#_Toc180372613)

[7.4. Log Transformation 16](#_Toc180372614)

[7.5. Power Transformation 17](#_Toc180372615)

[8. Hồi quy tuyến tính là gì? 17](#_Toc180372616)

[8.1. Ưu điểm của hồi quy tuyến tính 18](#_Toc180372617)

[8.2. Nhược điểm của hồi quy tuyến tính 18](#_Toc180372618)

[9.Các loại hồi quy tuyến tính. 18](#_Toc180372619)

[9.1. Hồi quy tuyến tính đơn giản (Simple Linear Regression) 18](#_Toc180372620)

[9.2. Hồi quy tuyến tính đa biến (Multiple Linear Regression) 19](#_Toc180372621)

[9.3. Hồi quy tuyến tính phi tuyến (Polynomial Regression) 19](#_Toc180372622)

[9.4. Hồi quy Ridge (Ridge Regression) 19](#_Toc180372623)

[9.5. Hồi quy Lasso (Lasso Regression) 19](#_Toc180372624)

[9.6. Hồi quy Elastic Net 20](#_Toc180372625)

[10. Hàm mất mát là gì? 20](#_Toc180372626)

[10.1.Hàm mất mát bình phương (Mean Squared Error - MSE) 21](#_Toc180372627)

[10.2. Hàm mất mát tuyệt đối (Mean Absolute Error - MAE) 21](#_Toc180372628)

[10.3. Hàm mất mát logistic (Binary Cross-Entropy Loss) 21](#_Toc180372629)

[10.4. Hàm mất mát đa lớp (Categorical Cross-Entropy Loss) 22](#_Toc180372630)

[10.5. Hàm mất mát Hinge 22](#_Toc180372631)

[11. Sai số bình phương trung bình là gì và tại sao chúng ta sử dụng nó? 23](#_Toc180372632)

[11.1. Sai số bình phương trung bình. 23](#_Toc180372633)

[11.2. Tại sao sử dụng MSE? 23](#_Toc180372634)

[12. R-squared là gì? 24](#_Toc180372635)

[13. Tối ưu hóa mô hình là gì? 25](#_Toc180372636)

[14. Thuật toán gradient descent ngẫu nhiên là gì? 26](#_Toc180372637)

[14.1. Cách thức hoạt động của SGD 26](#_Toc180372638)

[14.2. Quy trình của SGD 27](#_Toc180372639)

# **LỜI CẢM ƠN**

Lời nói đầu tiên em xin phép được gửi lời cảm ơn chân thành và sâu sắc nhất đến thầy Hồ Nhựt Minh vì những bài giảng vô cùng bổ ích và tâm huyết mà thầy đã truyền đạt trong suốt thời gian qua. Qua mỗi buổi học, em cảm nhận được sự nhiệt tình và lòng đam mê của thầy trong việc giảng dạy, cũng như sự kiên nhẫn và tận tâm trong việc giải đáp những thắc mắc của chúng em. Những phương pháp giảng dạy sinh động và gần gũi của thầy đã biến những bài học khô khan trở nên thú vị và dễ tiếp thu hơn.

Em hiểu rằng, để có được những bài giảng chất lượng và bổ ích như vậy, thầy đã phải đầu tư rất nhiều công sức và thời gian trong việc nghiên cứu và chuẩn bị. Em vô cùng biết ơn những nỗ lực và sự cống hiến hết mình của thầy. Nhờ có sự hướng dẫn của thầy, em đã vượt qua được nhiều khó khăn và tự tin hơn trong quá trình học tập. Em xin hứa sẽ tiếp tục cố gắng học tập, rèn luyện và áp dụng những kiến thức đã học được từ thầy vào thực tế một cách hiệu quả nhất. Em cũng mong rằng trong tương lai sẽ có thêm nhiều cơ hội được học hỏi và nhận được sự chỉ dẫn từ thầy.

Một lần nữa, em xin chân thành cảm ơn thầy và chúc cô luôn mạnh khỏe, hạnh phúc và thành công trong sự nghiệp giảng dạy. Mong rằng thầy sẽ tiếp tục truyền lửa và cảm hứng học tập cho nhiều thế hệ sinh viên sau.

# **NHẬN XÉT CỦA GIẢNG VIÊN**

TP. Hồ Chí Minh, ngày… tháng …năm…

**Giảng viên hướng dẫn**

(Ký tên và ghi rõ họ tên)

# **TỔNG QUAN**

Hồi quy tuyến tính giả định có mối quan hệ tuyến tính giữa biến dựđoán và biến mục tiêu. Cụ thể hơn, biến đầu ra có thể được tính toán bằng tổ hợp tuyến tính của các biến đầu vào. Hồi quy tuyến tính thường sử dụng phương pháp bình phương tối thiểu thông hường để tính toán hệ số của biến đầu vào. Mô hình hồi quy tuyến tính có thể được sử dụng để giải quyết nhiều vấn đề kinh doanh như dự đoán số lượng bán hàng, dự đoán huyết áp, dự đoán giá nhà, và nhiều vấn đề khác. Trong dự án này, một mô hình hồi quy tuyến tính để dự đoán số ngày đăng ký đã được triển khai từ đầu trong PyTorch. Nếu bạn chưa xem, đây là dự án trước của chuỗi "Học Cách Xây Dựng Mạng Nơ-ron PyTorch Từ Đầu".

## A. Mục Tiêu

- Hiểu hồi quy tuyến tính và hàm mất mát.

- Xây dựng mô hình hồi quy tuyến tính từ đầu trong PyTorch để dự đoán số ngày đăng ký.

## B. Mô Tả Dữ Liệu

Tập dữ liệu được sử dụng trong dự án này có thông tin về số ngày đăng ký dựa trên các đặc điểm khác nhau.

## C. Công Nghệ Sử Dụng

→ Ngôn ngữ: Python

→ Thư viện: pandas, pytorch, matplotlib, scikit-learn, numpy, torchvision, seaborn

## D. Cách Tiếp Cận

1. Dọn Dẹp Dữ Liệu

2. Tiền Xử Lý Dữ Liệu

3. Xây Dựng Mô Hình Hồi Quy Tuyến Tính

4. Đào Tạo Mô Hình

## E. Tổng Quan Mã Modular

- Input: data.csv

- ML Pipeline: Linear Regression.py, Preprocessing.py, TrainModel.py

- Notebook: Linear\_Regression.ipynb

- Output: y\_pred.png, Ly\_test.png

- Engine: Engine.py

- Readme: Readme.md

- yêu cầu: requirements.txt

Khi bạn giải nén tệp `pytorch\_linear\_regression.zip`, bạn sẽ tìm thấy các thư mục sau bên trong:

1. Input: Chứa dữ liệu mà chúng ta có để phân tích. Trong trường hợp của chúng ta, nó chứa `data.csv`.

2. Notebook: Chứa tệp notebook jupyter của dự án.

3. Output: Chứa biểu đồ của biến mục tiêu được dự đoán và thực

tế.

4. ML Pipeline: Chứa tất cả các hàm được đặt vào các tệp Python khác nhau, được đặt tên hợp lý. Các hàm Python này sau đó được gọi trong tệp `Engine.py`.

5. Requirements.txt: Chứa tất cả các thư viện cần thiết cùng với phiên bản tương ứng. Vui lòng cài đặt bằng lệnh `pip install -r requirements.txt`.

6. Readme.md: Tất cả các hướng dẫn để chạy mã được trình bày trong tệp này.

# **KẾT LUẬN**

## 1. Tensor là gì?

Tensor là một đối tượng học toán dùng để biểu diễn dữ liệu trong nhiều chiều. Chúng là phần mở rộng của các khái niệm cơ bản như vô hướng, cường và ma trận, cho phép mô tả dữ liệu dưới dạng phức tạp hơn. Một số đặc điểm chính của tensor bao gồm cấp (xếp hạng) và kích thước (dạng hình). Bậc của tensor cho biết số chiều của nó, ví dụ: bậc 0 là số vô hướng, bậc 1 là sảng, bậc 2 là ma trận, và bậc n là các mảng nhiều chiều phức tạp. Kích thước của phần tử tensor được xác định rõ ràng trong mỗi chiều, giúp hiển thị cấu trúc dữ liệu. Tensors có thể thực hiện nhiều phép toán như cộng đồng, trừ, và nhân, giúp xử lý dữ liệu trong hệ thống phức tạp.

Tensor có ứng dụng rộng rãi trong nhiều lĩnh vực khoa học và kỹ thuật. Trong vật lý, tensor mô tả các hiện tượng như lực, ứng dụng và trường điện từ, đặc biệt trong các nghiên cứu về thuyết tương tranh. Trong máy học, tensor là nền tảng của chiều sâu mô hình học, giúp xử lý hình ảnh dữ liệu, âm thanh và văn bản một cách hiệu quả. Ngoài ra, trong quá trình xử lý ảnh và video, tensor đại diện cho các pixel dưới dạng mảng nhiều chiều, cho phép thực hiện các phép biến đổi và phân tích dữ liệu một cách nhanh chóng. Hỗ trợ tính linh hoạt và khả năng biểu diễn dữ liệu đa chiều, tensor đã trở thành công cụ quan trọng trong nhiều lĩnh vực khoa học và công nghệ hiện đại.

## 2. Sự khác biệt giữa Tensor PyTorch và mảng Numpy

Dưới đây là sự khác biệt chi tiết giữa Tensor trong PyTorch và mảng NumPy:

### **2.1. Tính năng**

* PyTorch Tensor:

Tính toán GPU: Hỗ trợ tính toán trên GPU, cho phép tăng tốc độxử lý đáng kể cho các tác vụ tính toán nặng, như huấn luyện mô hình học sâu.

Theo dõi gradient: Cho phép theo dõi gradient tự động, rất quan trọng trong tối ưu hóa mô hình học sâu.

Hàm toán học phong phú: Cung cấp nhiều hàm toán học và các phương pháp tối ưu hóa đặc biệt cho học sâu.

* NumPy Array:

Tính toán CPU: Chỉ hỗ trợ tính toán trên CPU, không có tính năng tăng tốc GPU.

Không theo dõi gradient: Không hỗ trợ theo dõi gradient, do đó không phù hợp cho các tác vụ tối ưu hóa trong học máy.

Hàm toán học cơ bản: Cung cấp các hàm toán học cho xử lý sốliệu, nhưng không phong phú như PyTorch.

### **2.2. Hiệu suất**

PyTorch Tensor: Tối ưu cho các phép toán phức tạp, đặc biệt là trong môi trường học sâu. Có thể sử dụng các thư viện tối ưu hóa cho hiệu suất cao.

NumPy Array: Tối ưu cho các phép toán số học và xử lý dữ liệu nhẹ. Hiệu suất tốt cho các tác vụ số liệu cơ bản.

### **2.3. Cú pháp và API**

PyTorch Tensor: Cú pháp tương tự NumPy nhưng có thêm nhiều hàm và phương thức cho việc xây dựng mô hình, như `torch.nn` cho mạng nơ-ron.(Ví dụ: Thao tác trên tensors dễ dàng với các phương thức như `backward()` để tính gradient.)

NumPy Array: Cú pháp đơn giản và trực quan, dễ dàng sử dụng cho các phép toán cơ bản và thống kê.(Ví dụ: Sử dụng các hàm như `numpy.mean()` hay `numpy.dot()` cho các phép toán ma trận.)

### **2.4. Khả năng tương thích**

PyTorch Tensor: Dễ dàng tích hợp vào các dự án học sâu và có thể chuyển đổi qua lại với NumPy. Khi cần chuyển đổi, có thể sử dụng `tensor.numpy()` và `torch.from\_numpy()`.

NumPy Array: Thích hợp cho các ứng dụng khoa học và kỹ thuật chung, là lựa chọn đầu tiên cho xử lý số liệu trước khi đưa vào các mô hình học máy.

### **2.5. Tính linh hoạt**

PyTorch Tensor: Linh hoạt hơn trong việc xử lý các mô hình phức tạp và cho phép mở rộng dễ dàng khi cần thay đổi kiến trúc mạng.

NumPy Array: Thích hợp cho các tác vụ đơn giản hơn và không có khả năng mở rộng cho các mô hình học sâu phức tạp.

## 3. Tiền xử lý dữ liệu là gì?

Tiền xử lý dữ liệu là quá trình chuẩn bị dữ liệu trước khi phân tích hoặc xây dựng mô hình, nhằm nâng cao chất lượng và tính chính xác của dữ liệu. Dưới đây là các bước và chi tiết cụ thể trong quá trình này:

### **3.1. Làm sạch dữ liệu**

Xử lý giá trị thiếu (Missing Values):

Xóa: Loại bỏ các bản ghi có giá trị thiếu.

Thay thế: Sử dụng các phương pháp như trung bình, trung vị, hoặc các thuật toán dự đoán để thay thế giá trị thiếu.

Xử lý giá trị ngoại lệ (Outliers):

Phát hiện: Sử dụng các kỹ thuật như boxplot, Z-score hoặc IQR (Interquartile Range) để xác định giá trị ngoại lệ.

Xóa hoặc điều chỉnh: Quyết định xem có nên loại bỏ hoặc điều chỉnh các giá trị này.

### **3.2. Chuyển đổi dữ liệu**

Chuyển đổi kiểu dữ liệu: Chuyển đổi các biến về dạng phù hợp, ví dụ từ chuỗi sang số.(numeric) hoặc ngược lại.

Chuẩn hóa (Normalization): Đưa các giá trị về thang đo từ 0 đến 1, giúp cải thiện hiệu suất cho một số thuật toán học máy.

Chuẩn hóa (Standardization): Đưa các giá trị về trung bình 0 và độ lệch chuẩn 1, thường dùng cho dữ liệu có phân phối chuẩn.

### **3.3. Mã hóa dữ liệu**

Mã hóa biến phân loại (Categorical Variables): One-hot encoding: Tạo biến nhị phân cho mỗi giá trị của biến phân loại.

Label encoding: Chuyển đổi các giá trị phân loại thành các số nguyên.

### **3.4. Tách dữ liệu**

Chia dữ liệu thành các tập:

Tập huấn luyện (Training Set): Dùng để huấn luyện mô hình.

Tập kiểm tra (Testing Set): Dùng để đánh giá hiệu suất của mô hình.

Tập xác thực (Validation Set): Dùng để điều chỉnh tham số mô hình (nếu cần).

### **3.5. Giảm chiều dữ liệu**

Kỹ thuật giảm chiều:

PCA (Phân tích thành phần chính): Giúp giảm số lượng biến trong dữ liệu nhưng vẫn giữ lại phần lớn thông tin.

LDA (Phân tích phân biệt tuyến tính): Thích hợp cho các bài toán phân loại.

### **3.6. Tạo đặc trưng (Feature Engineering)**

Tạo đặc trưng mới: Kết hợp hoặc biến đổi các đặc trưng hiện có để tạo ra thông tin mới có thể cải thiện hiệu suất mô hình.

### **3.7. Kiểm tra và đánh giá**

Đánh giá chất lượng dữ liệu: Sử dụng các chỉ số như độ đầy đủ (completeness), độ chính xác (accuracy), và tính nhất quán (consistency) để đảm bảo dữ liệu đã được xử lý đúng cách.

**Tại sao tiền xử lý dữ liệu quan trọng?**

Cải thiện độ chính xác mô hình: Dữ liệu sạch và chính xác giúp tăng cường khả năng tổng quát của mô hình.

Tối ưu hóa hiệu suất: Giảm thiểu thời gian và tài nguyên tính toán cần thiết cho việc xử lý dữ liệu lớn.

Phát hiện sớm các vấn đề: Tiền xử lý giúp nhận diện và giải quyết các vấn đề trong dữ liệu từ đầu, tránh việc tạo ra các kết quả sai lệch trong phân tích.

## 4. Các đặc điểm phân loại và số

### **4.1. Đặc điểm phân loại (Categorical Variables)**

Đặc điểm phân loại là các biến mà giá trị của chúng được phân loại thành các nhóm hoặc loại khác nhau. Chúng không có thứ tự tự nhiên và thường được mô tả bằng các nhãn hoặc tên.

Các loại đặc điểm phân loại:

Biến nhị phân (Binary): Chỉ có hai giá trị (ví dụ: Có/Không, Có/Không có.)

Biến phân loại (Nominal): Có nhiều giá trị mà không có thứ tự (ví dụ: Màu sắc (đỏ, xanh, vàng), loại động vật (chó, mèo, chim).

Biến thứ tự (Ordinal): Có nhiều giá trị với thứ tự rõ ràng, nhưng khoảng cách giữa các giá trị không nhất quán, (ví dụ: Đánh giá (thấp, trung bình, cao), cấp bậc (đầu, giữa, cuối).

### **4.2. Đặc điểm số (Numeric Variables)**

Đặc điểm số là các biến có giá trị số thực, có thể thực hiện các phép toán số học. Chúng có thứ tự và có thể đo được khoảng cách giữa các giá trị.

Các loại đặc điểm số:

Biến liên tục (Continuous): Có thể nhận bất kỳ giá trị nào trong một khoảng nhất định.

Biến rời rạc (Discrete): Chỉ nhận các giá trị cụ thể, thường là số nguyên.

### **4.3. Ứng dụng trong phân tích dữ liệu**

Phân loại:

Các biến phân loại thường được sử dụng trong phân tích thống kê, mô hình hóa, và học máy để phân chia dữ liệu thành các nhóm.

Các thuật toán như hồi quy logistic, cây quyết định, và mạng nơ-ron thường sử dụng biến phân loại làm đầu vào hoặc đầu ra.

Số: Các biến số thường được sử dụng trong các phép toán thống kê như tính trung bình, phương sai, và hồi quy tuyến tính.

Các thuật toán học máy như hồi quy tuyến tính và hồi quy đa thức yêu cầu biến số làm đầu vào.

## 5. Tại sao chúng ta cần tiền xử lý dữ liệu?

Tiền xử lý dữ liệu là một bước thiết yếu trong quá trình phân tích dữ liệu và xây dựng mô hình học, nhằm đảm bảo rằng dữ liệu đầu tiên có chất lượng tốt nhất và phù hợp để phân tích. Việc này không chỉ giúp cải thiện hiệu suất của mô hình mà còn đảm bảo kết quả chính xác hơn. Dưới đây là những lý do giải thích chính tại sao tiền xử lý dữ liệu.

### **5.1. Cải thiện chất lượng dữ liệu**

Xóa bỏ dữ liệu sai lệch: Dữ liệu có thể chứa lỗi, giá trị thiếu hoặc ngoại lệ. Tiền xử lý giúp làm sạch dữ liệu, từ đó nâng cao độ chính xác.

Đảm bảo tính nhất quán: Dữ liệu từ nhiều nguồn khác nhau có thểkhông đồng nhất. Tiền xử lý giúp chuẩn hóa và đồng bộ hóa dữ liệu.

### **5.2. Tăng cường độ chính xác của mô hình**

Giảm thiểu sai số: Mô hình học máy hoạt động tốt hơn với dữ liệu sạch và chính xác, giúp giảm thiểu sai số và tăng cường khả năng tổng quát.

Phát hiện các mối quan hệ: Tiền xử lý giúp làm nổi bật các đặc điểm quan trọng và mối quan hệ giữa chúng, hỗ trợ trong việc xây dựng mô hình hiệu quả.

### **5.3. Tiết kiệm thời gian và tài nguyên**

Xử lý sớm các vấn đề: Phát hiện và xử lý các vấn đề trong dữ liệu từđầu giúp tiết kiệm thời gian và công sức trong các bước phân tích tiếp theo.

Tối ưu hóa quy trình làm việc: Dữ liệu đã được tiền xử lý giúp giảm thiểu công sức cho các bước phân tích và xây dựng mô hình sau này.

### **5.4. Tăng cường khả năng phân tích**

Cải thiện khả năng giải thích: Dữ liệu sạch và rõ ràng giúp dễ dàng hơn trong việc phân tích và đưa ra quyết định dựa trên dữ liệu.

Hỗ trợ phát hiện mẫu: Dữ liệu được chuẩn bị tốt giúp dễ dàng phát hiện các mẫu và xu hướng trong dữ liệu.

### **5.5. Phù hợp với các thuật toán học máy**

Yêu cầu về định dạng: Nhiều thuật toán học máy yêu cầu dữ liệu phải ở định dạng cụ thể. Tiền xử lý giúp chuyển đổi và mã hóa dữ liệu theo yêu cầu.

Tăng cường hiệu suất: Việc chuẩn hóa và giảm chiều dữ liệu giúp cải thiện hiệu suất của mô hình và rút ngắn thời gian huấn luyện.

### **5.6. Đảm bảo tính pháp lý và đạo đức**

Ngoài việc đảm bảo chất lượng và hiệu quả của dữ liệu trong quá trình phân tích và xây dựng mô hình, một yếu tố quan trọng không thể bỏ qua là đảm bảo tính pháp lý và đức tin trong quá trình xử lý dữ liệu. Việc thu thập, xử lý và sử dụng dữ liệu Yêu cầu phải góp thủ các quy định pháp lý và tiêu chuẩn đạo đức nhằm bảo vệ quyền lợi của cá nhân và đảm bảo tính minh bạch trong các quyết định dựa trên dữ liệu.

## 6. Cách thực hiện mã hóa nhãn cho các đặc điểm phân loại?

Mã hóa nhãn (Mã hóa nhãn) là một phương pháp quan trọng trong tiền xử lý dữ liệu, giúp chuyển đổi loại phân giá giá trị (giá trị phân loại) thành số giá trị tương ứng. Điều này cho phép các máy tính toán thuật toán có thể xử lý và hiểu được loại phân loại dữ liệu một cách hiệu quả hơn.

Mã hóa Nhãn là phương pháp đơn giản nhất để mã hóa các loại phân nhãn. Mỗi loại giá trị được chuyển thành một số duy nhất. Ví dụ: nếu chúng tôi có một tệp dữ liệu với các nhãn "A", "B" và "C", thì quá trình.

* "MỘT"
* "B"
* "C" → 2

Mặc dù phương pháp này được thực hiện rất dễ dàng, nhưng nó có thể gây ra vấn đề trong các thuật toán học có tính thứ tự (thứ tự), như hồi phục tính tuyến tính, vì các số giá trị được hiểu là có thứ tự. Điều này có thể dẫn đến mô hình công việc sẽ giả định rằng "B" (1) có mức độ ưu tiên cao hơn "A" (0) và thấp hơn "C" (2), mặc dù trên thực tế, các giá giá trị này không có mối quan hệ.

One-Hot Encoding giúp loại bỏ giả định về thứ tự giữa các lớp và cho phép máy tính toán thuật toán làm việc với loại phân tích dữ liệu một cách hiệu quả. Tuy nhiên, phương pháp này có thể làm tăng kích thước đáng kể của dữ liệu.

huẩn hóa các thông số (tính năng số) là một bước quan trọng trong tiền xử lý dữ liệu, giúp đưa ra tất cả các thông số về một thang đo chung, từ đó cải thiện hiệu suất của các máy học. Dưới đây là một số phương pháp phổ biến để chuẩn hóa các đặc điểm

## 7. Cách chuẩn hóa các đặc điểm số?

Chuẩn hóa các đặc điểm số là một bước quan trọng trong việc tiền xử lý dữ liệu, đặc biệt trong các mô hình học máy. Dưới đây là một số phương pháp phổ biến để chuẩn hóa các đặc điểm số:

### **7.1. Chuẩn hóa Min-Max (Min-Max Normalization)**

Công thức:

Kết quả: Giá trị sẽ nằm trong khoảng [0, 1]. Phương pháp này thích hợp khi bạn muốn giữ lại cấu trúc tương đối giữa các giá trị.

### **7.2. Chuẩn hóa Z-Score (Z-Score Normalization)**

Công thức:

Trong đó, μ\muμ là giá trị trung bình và σ\sigmaσ là độ lệch chuẩn của đặc điểm. Kết quả sẽ có trung bình là 0 và độ lệch chuẩn là 1. Phương pháp này thích hợp khi dữ liệu có phân phối gần với chuẩn.

### **7.3. Chuẩn hóa Robust (Robust Scaling)**

Công thức:

Trong đó, Q1 và Q3là các phần tư (quartiles) thứ nhất và thứ ba. Phương pháp này ít nhạy cảm với các giá trị ngoại lệ.

### **7.4. Log Transformation**

Sử dụng khi dữ liệu có sự phân phối lệch mạnh.

Công thức: **X′=log(X+c)**

Trong đó, ccc là một hằng số để tránh việc tính log của 0.

### **7.5. Power Transformation**

Bao gồm các phương pháp như Box-Cox hoặc Yeo-Johnson, giúp làm giảm sự lệch của dữ liệu bằng cách áp dụng các hàm lũy thừa.

## 8. Hồi quy tuyến tính là gì?

Hồi quy tuyến tính là một phương pháp thống kê và học máy dùng đểmô hình hóa mối quan hệ giữa một biến phụ thuộc (đầu ra) và một hoặc nhiều biến độc lập (đầu vào). Mục tiêu của hồi quy tuyến tính là tìm ra một hàm tuyến tính tốt nhất để dự đoán giá trị của biến phụthuộc dựa trên giá trị của các biến độc lập.

Các thành phần chính của hồi quy tuyến tính:

Biến phụ thuộc: Là biến mà bạn muốn dự đoán hoặc giải thích (ví dụ: giá nhà, doanh thu).

Biến độc lập: Là các biến dùng để dự đoán biến phụ thuộc (ví dụ: diện tích, số phòng, vị trí).

Hàm hồi quy: Trong hồi quy tuyến tính đơn giản, hàm có dạng: **Y=β0​+β1​x+ϵ**

Trong đó:

* y là biến phụ thuộc.
* x là biến độc lập.
* β0 là hệ số chệch (intercept).
* β1​ là hệ số góc (slope).
* ϵ là sai số ngẫu nhiên.

Hồi quy tuyến tính đơn giản: Chỉ có một biến độc lập.

Hồi quy tuyến tính đa biến: Có nhiều biến độc lập.

Quy trình thực hiện hồi quy tuyến tính:

Thu thập dữ liệu: Tập hợp dữ liệu với các biến cần thiết.

Tiền xử lý dữ liệu: Làm sạch và chuẩn bị dữ liệu cho mô hình.

Phân tích dữ liệu: Khám phá và hiểu dữ liệu trước khi xây dựng mô hình.

Xây dựng mô hình: Sử dụng các thuật toán hồi quy tuyến tính để tìm hệ số tốt nhất.

Đánh giá mô hình: Sử dụng các chỉ số như R-squared, RMSE (Root Mean Square Error) để đánh giá hiệu suất của mô hình.

Dự đoán: Sử dụng mô hình để dự đoán giá trị mới.

### **8.1. Ưu điểm của hồi quy tuyến tính**

Dễ hiểu: Mô hình hồi quy tuyến tính dễ giải thích và trực quan.

Nhanh chóng: Thời gian huấn luyện và dự đoán nhanh.

Có thể áp dụng trong nhiều lĩnh vực: Từ kinh tế đến khoa học xã hội.

### **8.2. Nhược điểm của hồi quy tuyến tính**

Giả định tuyến tính: Giả định rằng mối quan hệ giữa các biến là tuyến tính, điều này không luôn đúng.

Nhạy cảm với ngoại lệ: Dữ liệu ngoại lệ có thể ảnh hưởng mạnh đến mô hình.

Không xử lý tốt đa thức: Hồi quy tuyến tính không phù hợp cho các mối quan hệ phi tuyến.

## 9.Các loại hồi quy tuyến tính.

### **9.1. Hồi quy tuyến tính đơn giản (Simple Linear Regression)**

Đặc điểm: Chỉ có một biến độc lập và một biến phụ thuộc.

Công thức: **Y=β0​+β1​x+ϵ**

Sử dụng: Thích hợp cho các mối quan hệ tuyến tính giữa hai biến.

### **9.2. Hồi quy tuyến tính đa biến (Multiple Linear Regression)**

Đặc điểm: Có nhiều hơn một biến độc lập.

Công thức: **y=β0​+β1​x1​+β2​x2​+…+βn​xn​+ϵ**

Sử dụng: Dùng để phân tích các mối quan hệ phức tạp giữa một biến phụ thuộc và nhiều biến độc lập.

### **9.3. Hồi quy tuyến tính phi tuyến (Polynomial Regression)**

Đặc điểm: Mặc dù vẫn gọi là hồi quy tuyến tính, nhưng sử dụng các biến độc lập theo dạng đa thức (polynomial).

Công thức: **y=β0​+β1​x+β2​x2+ϵ**

Sử dụng: Thích hợp cho các mối quan hệ phi tuyến giữa biến phụ thuộc và biến độc lập.

### **9.4. Hồi quy Ridge (Ridge Regression)**

Đặc điểm: Một loại hồi quy tuyến tính đa biến có thêm điều kiện L2 để giảm thiểu hiện tượng overfitting.

Công thức: **Minimize ∥y−Xβ​**

Trong đó:

* y: Vector các giá trị mục tiêu (biến phụ thuộc).
* X: Ma trận các đặc trưng (biến độc lập).
* β: Vector các hệ số hồi quy.

Sử dụng: Giúp loại bỏ các biến không quan trọng và cải thiện khả năng giải thích của mô hình.

### **9.5. Hồi quy Lasso (Lasso Regression)**

Đặc điểm: Tương tự như hồi quy Ridge, nhưng sử dụng điều kiện L1 để chọn lọc các biến.

Công thức: **Minimize ∥y−Xβ∥**

Trong đó:

* y: Vector các giá trị mục tiêu (biến phụ thuộc).
* X: Ma trận các đặc trưng (biến độc lập).
* β: Vector các hệ số hồi quy.

Sử dụng: Giúp loại bỏ các biến không quan trọng và cải thiện khả năng giải thích của mô hình.

### **9.6. Hồi quy Elastic Net**

Đặc điểm: Kết hợp giữa hồi quy Ridge và Lasso, sử dụng cả điều kiện L1 và L2.

Công thức: **Minimize**

Trong đó:

* y: Vector các giá trị mục tiêu (biến phụ thuộc).
* X: Ma trận các đặc trưng (biến độc lập).
* β: Vector các hệ số hồi quy cần ước lượng.
* n: Số lượng quan sát.
* λ: Tham số điều chỉnh mức độ phạt (regularization parameter).
* α: Tham số cân bằng giữa L1 và L2. α∈[0,1]\alpha \in [0, 1]α∈[0,1].

Sử dụng: Khi dữ liệu có nhiều biến độc lập và có sự đa cộng tuyến.

## 10. Hàm mất mát là gì?

Hàm mất mát (loss function) là một công cụ quan trọng trong học máy và tối ưu hóa mô hình. Nó được sử dụng để đo lường sự khác biệt giữa giá trị dự đoán của mô hình và giá trị thực tế. Hàm mất mát giúp đánh giá hiệu suất của mô hình và hướng dẫn quá trình huấn luyện bằng cách tối ưu hóa các tham số của mô hình.Các loại hàm mất mát phổ biến.

### **10.1.Hàm mất mát bình phương (Mean Squared Error - MSE)**

Định nghĩa: Tính trung bình của bình phương các sai số giữa giá trị thực tế và giá trị dự đoán.

Công thức:

* yi​ là giá trị thực tế.
* y là giá trị dự đoán.
* N là số lượng mẫu.

Sử dụng: Thường dùng trong hồi quy tuyến tính.

### **10.2. Hàm mất mát tuyệt đối (Mean Absolute Error - MAE)**

Định nghĩa: Tính trung bình của giá trị tuyệt đối của các sai số.

Công thức:

Sử dụng: Thích hợp khi bạn muốn giảm thiểu ảnh hưởng của ngoại lệ (outliers).

### **10.3. Hàm mất mát logistic (Binary Cross-Entropy Loss)**

Định nghĩa: Sử dụng cho các bài toán phân loại nhị phân, đo lường sự khác biệt giữa phân phối thực tế và phân phối dự đoán.

Công thức: **Binary Cross-Entropy**Loss =

* yi là giá trị thực tế (0 hoặc 1).
* y ​ là xác suất dự đoán.

Sử dụng: Dùng cho các mô hình phân loại nhị phân như hồi quy logistic.

### **10.4. Hàm mất mát đa lớp (Categorical Cross-Entropy Loss)**

Định nghĩa: Dùng cho các bài toán phân loại đa lớp, đo lường sự khác biệt giữa phân phối xác suất thực và phân phối dự đoán.

Công thức:

Sử dụng: Thích hợp cho các mô hình phân loại đa lớp.

### **10.5. Hàm mất mát Hinge**

Định nghĩa: Thường dùng trong các bài toán phân loại với SVM (Support Vector Machines).

Công thức: **Cross-Entropy Loss =**

* C là số lớp (classes).
* yi​ là giá trị thực tế của lớp c (thường là 1 nếu đúng lớp, 0 nếu không).
* y​ là xác suất dự đoán của lớp c.

Sử dụng: Giúp tìm ra ranh giới phân loại tốt nhất.

Tại sao hàm mất mát quan trọng?

Hướng dẫn quá trình huấn luyện: Hàm mất mát cho biết mô hình hoạt động tốt hay xấu, từ đó điều chỉnh các tham số để cải thiện.

Đánh giá hiệu suất: Giúp đánh giá khả năng của mô hình trên tập huấn luyện và tập kiểm tra.

Tối ưu hóa: Sử dụng các thuật toán tối ưu hóa (như Gradient Descent) để giảm thiểu hàm mất mát và cải thiện độ chính xác của mô hình.

## 11. Sai số bình phương trung bình là gì và tại sao chúng ta sử dụng nó?

### **11.1. Sai số bình phương trung bình.**

Sai số bình phương trung bình (Mean Squared Error - MSE) là một chỉ số thống kê dùng để đo lường độ chính xác của một mô hình dựđoán. Nó tính toán mức độ khác biệt giữa giá trị thực tế và giá trị dựđoán, bằng cách lấy trung bình của bình phương các sai số.

Công thức tính MSE như sau:

### **11.2. Tại sao sử dụng MSE?**

Đo lường độ chính xác: MSE giúp đánh giá khả năng dự đoán của mô hình. Một giá trị MSE thấp cho thấy mô hình có độ chính xác cao hơn, tức là giá trị dự đoán gần với giá trị thực tế.Phân tích và tối ưu hóa: MSE là một hàm mất mát liên tục, giúp cho quá trình tối ưu hóa dễ dàng hơn. Các thuật toán như Gradient Descent có thể sử dụng MSE để điều chỉnh các tham số của mô hình.

Ưu điểm của MSE: Dễ dàng tính toán và có thể áp dụng cho nhiều loại mô hình hồi quy khác nhau.Nhấn mạnh vào sai số lớn: Bởi vì sai số được bình phương, MSE làm nổi bật những sai số lớn hơn, điều này có thể có ích trong một sốtình huống mà bạn muốn giảm thiểu ảnh hưởng của các sai số lớn.

Nhược điểm của MSE: Lỗi bình phương trung bình (MSE) là một biến phổ phổ biến trong việc đánh giá độ chính xác của các mô hình hồi phục. Tuy nhiên, MSE cũng có một số nhược điểm đáng chú ý cần được xem xét khi áp dụng trong thực tế. Đầu tiên, MSE rất nhạy cảm với các ngoại lệ giá trị (ngoại lệ). Những sai số lớn làm cho các dữ liệu không điển hình gây phiền toái có thể làm tăng giá trị MSE một cách đáng kể. Điều này có thể dẫn đến sự hiểu biết về khả năng mong đợi của mô hình, khi thực tế mô hình có thể hoạt động tốt trên phần lớn dữ liệu. Việc một số ngoại lệ dữ liệu có thể mở rộng MSE giá trị cao để đánh giá hiệu suất của mô hình.

Ngoài ra, MSE còn có một nhược điểm khác là không có đơn vị dễ hiểu. Giá trị của MSE có đơn vị là phương pháp của đơn vị đầu vào của biến. Ví dụ, nếu biến đầu vào là giá cả, thì MSE sẽ có đơn vị là đô la bình phương. Điều này gây khó khăn trong việc giải quyết kết quả thích hợp và so sánh trực tiếp với giá trị thực tế mà mô hình dự kiến. Khi sử dụng MSE, người dùng có thể không rõ ý nghĩa của dữ liệu, do đó nó không thể xác định sai số thực tế mà mô hình phải giải quyết.

## 12. R-squared là gì?

R-squared (R²), hay còn gọi là hệ số xác định, là một chỉ số thống kê dùng để đo lường mức độ phù hợp của mô hình hồi quy tuyến tính với dữ liệu. Nó cho biết phần trăm biến động của biến phụ thuộc (biến mà bạn muốn dự đoán) được giải thích bởi các biến độc lập trong mô hình.

Công thức tính R-squared

R-squared được tính bằng công thức:

Trong đó :

SS res (Residual Sum of Squares): Tổng bình phương sai số, tức là tổng của các bình phương sai số giữa giá trị thực tế và giá trị dự đoán.SS tot (Total Sum of Squares): Tổng bình phương tổng thể, tức là tổng của các bình phương sai số giữa giá trị thực tế và trung bình của giá trị thực tế.

Giá trị của R-squared:

0 ≤ R² ≤ 1: Giá trị R-squared nằm trong khoảng từ 0 đến 1.

R² = 0: Mô hình không giải thích được bất kỳ biến động nào của biến phụ thuộc (mô hình không tốt).

R² = 1: Mô hình giải thích hoàn toàn biến động của biến phụ thuộc (mô hình rất tốt).

**Tại sao R-squared quan trọng?**

Đánh giá hiệu suất: R-squared giúp đánh giá khả năng giải thích của mô hình, cho thấy mức độ mà các biến độc lập ảnh hưởng đến biến phụ thuộc.

So sánh các mô hình: Có thể sử dụng R-squared để so sánh giữa các mô hình khác nhau. Mô hình với R-squared cao hơn thường được xem là tốt hơn trong việc giải thích biến động.

**Lưu ý khi sử dụng R-squared**

Không phải là chỉ số hoàn hảo: R-squared không thể cho biết liệu mối quan hệ giữa các biến là tuyến tính hay phi tuyến, và nó không đánh giá được độ chính xác của dự đoán.

Có thể bị ảnh hưởng bởi số lượng biến: Khi thêm nhiều biến độc lập vào mô hình, R-squared có thể tăng ngay cả khi các biến này không thực sự có ý nghĩa. Do đó, có thể cần sử dụng R-squared điều chỉnh (Adjusted R-squared) để điều chỉnh cho số lượng biến trong mô hình.

## 13. Tối ưu hóa mô hình là gì?

Tối ưu hóa mô hình là quá trình điều chỉnh các tham số và cấu trúc của một mô hình học máy nhằm cải thiện hiệu suất của nó trên dữliệu. Mục tiêu của tối ưu hóa là tìm ra các tham số tốt nhất để mô hình có thể dự đoán chính xác và hiệu quả hơn.

Các bước trong quá trình tối ưu hóa mô hình:

Chọn mô hình: Xác định loại mô hình phù hợp với bài toán, như hồi quy tuyến tính, cây quyết định, mạng nơ-ron, v.v.

Tiền xử lý dữ liệu: Làm sạch và chuẩn bị dữ liệu để phù hợp với mô hình. Điều này có thể bao gồm mã hóa biến phân loại, chuẩn hóa dữliệu và chia tập dữ liệu thành tập huấn luyện và tập kiểm tra.

Chọn hàm mất mát: Xác định hàm mất mát phù hợp để đánh giá hiệu suất của mô hình. Hàm mất mát này sẽ được tối ưu hóa trong quá trình huấn luyện.

Tối ưu hóa tham số: Sử dụng các thuật toán tối ưu hóa (như Gradient Descent, Adam, RMSprop) để điều chỉnh các tham số của mô hình nhằm giảm thiểu hàm mất mát.

Điều chỉnh siêu tham số (Hyperparameter Tuning): Các siêu tham số là các tham số không được học trực tiếp từ dữ liệu, như số lượng cây trong rừng ngẫu nhiên hay độ sâu của cây quyết định. Việc điều chỉnh siêu tham số có thể được thực hiện qua các phương pháp như Grid Search, Random Search hoặc Bayesian Optimization.

Đánh giá mô hình: Sử dụng các chỉ số như R-squared, MSE, accuracy, precision, recall để đánh giá hiệu suất của mô hình trên tập kiểm tra.

Lặp lại: Quá trình tối ưu hóa có thể cần nhiều vòng lặp để điều chỉnh các tham số và cải thiện hiệu suất.

**Tại sao tối ưu hóa mô hình quan trọng?**

Cải thiện độ chính xác: Tối ưu hóa giúp mô hình dự đoán chính xác hơn, từ đó cải thiện chất lượng của dự đoán.

Giảm thiểu overfitting và underfitting: Tối ưu hóa giúp tìm ra được mô hình cân bằng, không quá phức tạp cũng như không quá đơn giản để phù hợp với dữ liệu.

Tăng cường khả năng tổng quát: Một mô hình được tối ưu hóa tốt có khả năng tổng quát tốt hơn trên dữ liệu chưa thấy, giúp tăng cường độ tin cậy của các dự đoán.

## 14. Thuật toán gradient descent ngẫu nhiên là gì?

Thuật toán Gradient Descent ngẫu nhiên (Stochastic Gradient Descent - SGD) là một phương pháp tối ưu hóa được sử dụng để tìm kiếm các tham số tối ưu trong các mô hình học máy, đặc biệt là trong hồi quy và mạng nơ-ron. SGD là một biến thể của phương pháp Gradient Descent truyền thống.

### **14.1. Cách thức hoạt động của SGD**

Gradient Descent truyền thống: Trong phương pháp Gradient Descent thông thường, gradient (đạo hàm) của hàm mất mát được tính toán trên toàn bộ tập dữ liệu. Điều này có thể tốn nhiều thời gian và tài nguyên, đặc biệt với các tập dữ liệu lớn.Gradient Descent ngẫu nhiên: Với SGD, thay vì tính gradient trên toàn bộ tập dữ liệu, phương pháp này chỉ sử dụng một mẫu ngẫu nhiên (hoặc một vài mẫu) để cập nhật tham số mô hình tại mỗi bước.

Điều này giúp giảm thiểu thời gian tính toán và cho phép cập nhật tham số nhanh hơn.

### **14.2. Quy trình của SGD**

Khởi tạo tham số: Bắt đầu với một giá trị khởi tạo cho các tham số của mô hình.

Chọn mẫu ngẫu nhiên: Chọn một mẫu ngẫu nhiên (hoặc một batch nhỏ) từ tập dữ liệu.

Tính toán gradient: Tính gradient của hàm mất mát dựa trên mẫu đã chọn.

Cập nhật tham số: Cập nhật các tham số theo công thức:

Lặp lại: Lặp lại các bước trên cho nhiều mẫu và nhiều vòng lặp (epochs) cho đến khi đạt được độ hội tụ mong muốn.

#### **14.2.1 Ưu điểm của SGD**

Tốc độ nhanh: SGD thường nhanh hơn Gradient Descent truyền thống, đặc biệt với các tập dữ liệu lớn.

Cập nhật liên tục: Các tham số được cập nhật thường xuyên, giúp mô hình có thể học linh hoạt hơn.

Khả năng thoát khỏi cực tiểu cục bộ: Bởi vì SGD sử dụng mẫu ngẫu nhiên, nó có thể giúp thoát khỏi các cực tiểu cục bộ trong hàm mất mát, điều này có thể hữu ích trong các bàitoán phi tuyến phức tạp.

#### **14.2.2. Nhược điểm của SGD**

Biến động cao: Việc sử dụng mẫu ngẫu nhiên có thể dẫn đến các cập nhật không ổn định và có nhiều biến động trong quá trình huấn luyện.

Tốc độ học cần điều chỉnh: Cần phải điều chỉnh tốc độ học cẩn thận, nếu quá cao có thể khiến mô hình không hội tụ.