HCMUT EE MACHINE LEARNING & IOT LAB

Khóa hè 2025: Python và Machine Learning
Day 7

LOGISTIC REGRESSION
SOFTMAX REGRESSION

Presentation By: Trần Huy

Ho Chi Minh City – 2025

Mục tiêu buổi học:

1. Hiểu được bản chất của Logistic Regression (nhị phân)
2. Mở rộng từ Logistic Regression sang Softmax Regression (đa lớp)
3. Biết cách sử dụng hàm Softmax và hàm mất mát Cross-Entropy
4. Thấy rõ mối liên hệ giữa phân loại nhị phân và đa lớp trong học máy

Table of Content

I. MEAN SQUARED ERROR

I. Mean Squared Error

MSE là gì?

Mean Squared Error (MSE) hay Sai số Bình phương Trung bình là một trong những **hàm mất mát (loss function)** quan trọng nhất trong học máy, đặc biệt là trong các bài toán hồi quy. ***MSE đo lường mức độ sai lệch giữa các giá trị dự đoán và giá trị thực tế bằng cách tính trung bình của bình phương các sai số.***

Cho tập dữ liệu gồm n điểm dữ liệu $\{(x_i, y_i)\}_{i=1}^{n}$, trong đó:

- x_i là vector đặc trưng của mẫu thứ i
- y_i là giá trị thực tế của mẫu thứ i
- $\hat{y}_i$ là giá trị dự đoán của mẫu thứ i

MSE được định nghĩa như sau:

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

I. Mean Squared Error

MSE trong bài toán Regression

Đối với mô hình hồi quy tuyến tính đơn giản:

$$\hat{y}_i = \beta_0 + \beta_1 x_i \tag{2}$$

MSE trở thành:

$$MSE(\beta_0, \beta_1) = \frac{1}{n}\sum_{i=1}^{n}(y_i - \beta_0 - \beta_1 x_i)^2 \tag{3}$$

Đối với hồi quy tuyến tính đa biến:

$$\hat{y}_i = \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \ldots + \beta_p x_{ip} = \mathbf{x}_i^T \boldsymbol{\beta} \tag{4}$$

MSE được viết dưới dạng vector:

$$\boxed{MSE(\boldsymbol{\beta}) = \frac{1}{n}\|\mathbf{y} - \mathbf{X}\boldsymbol{\beta}\|^2} \tag{5}$$

trong đó:

- $\mathbf{y} = [y_1, y_2, \ldots, y_n]^T$ là vector giá trị thực tế
- $\mathbf{X}$ là ma trận thiết kế có kích thước $n \times (p+1)$
- $\boldsymbol{\beta} = [\beta_0, \beta_1, \ldots, \beta_p]^T$ là vector tham số

I. Mean Squared Error

Tính chất

1. **Không âm**: $MSE \geq 0$, với $MSE = 0$ khi và chỉ khi $\hat{y}_i = y_i$ với mọi i
2. **Đối xứng**: MSE không phân biệt giữa over-prediction và under-prediction
3. **Đơn vị**: MSE có đơn vị là bình phương của đơn vị của biến phụ thuộc
4. **Nhạy cảm với outliers**: Do có bình phương, MSE rất nhạy cảm với các giá trị ngoại lệ

MSE có thể được phân tích thành:

$$\boxed{MSE = \text{Bias}^2 + \text{Variance} + \text{Irreducible Error}}$$

trong đó:

$$\begin{aligned} \text{Bias}^2 &= \mathbb{E}[(\mathbb{E}[\hat{y}] - y)^2] \\ \text{Variance} &= \mathbb{E}[(\hat{y} - \mathbb{E}[\hat{y}])^2] \\ \text{Irreducible Error} &= \sigma^2 \end{aligned}$$

I. Mean Squared Error

Tối ưu hóa

Để tìm tham số tối ưu, ta cần giải bài toán: $$\boldsymbol{\beta}^* = \arg\min_{\boldsymbol{\beta}} MSE(\boldsymbol{\beta})$$

Lấy đạo hàm theo $\boldsymbol{\beta}$:

$$\frac{\partial MSE}{\partial \boldsymbol{\beta}} = \frac{2}{n}\mathbf{X}^T(\mathbf{X}\boldsymbol{\beta} - \mathbf{y})$$

Đặt đạo hàm bằng 0:

$$\mathbf{X}^T(\mathbf{X}\boldsymbol{\beta} - \mathbf{y}) = 0$$

Giải phương trình normal:

$$\boxed{\boldsymbol{\beta}^* = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}}$$

Nghiệm duy nhất tồn tại khi và chỉ khi:

- Ma trận $\mathbf{X}^T\mathbf{X}$ khả nghịch
- $\text{rank}(\mathbf{X}) = p + 1$ (không có multicollinearity)
- $n > p$ (số mẫu lớn hơn số tham số)

I. Mean Squared Error

GD cho MSE

Khi không thể tính nghiệm giải tích, ta sử dụng gradient descent:

$$\boldsymbol{\beta}^{(t+1)} = \boldsymbol{\beta}^{(t)} - \alpha \nabla MSE(\boldsymbol{\beta}^{(t)})$$

với gradient:

$$\nabla MSE(\boldsymbol{\beta}) = \frac{2}{n}\mathbf{X}^T(\mathbf{X}\boldsymbol{\beta} - \mathbf{y})$$

Bài tập:

x	y
1	2
2	3
3	5

Dự đoán: $\hat{y} = \beta_0 + \beta_1 x$

II. Bài toán Classification

II. Bài toán classification

Dự đoán và Phân loại

Tiêu chí	Hồi quy (Regression)	Phân loại (Classification)
Dạng đầu ra	Liên tục (real number)	Rời rạc (discrete class label)
Mục tiêu	Dự đoán giá trị số lượng	Phân loại dữ liệu vào nhóm cụ thể
Ví dụ kết quả	Giá nhà = 2.3 tỷ đồng	Loại nhà = {Cao cấp, Trung cấp, Bình dân}
Hàm mất mát thường dùng	Mean Squared Error (MSE), MAE, Huber, ...	Cross-Entropy, Log Loss
Thuật toán phổ biến	Linear Regression, SVR, XGBoost (reg), ...	Logistic Regression, Random Forest, SVM, CNN, ...
Đánh giá mô hình	RMSE, MAE, R^2 score	Accuracy, Precision, Recall, F1-score, AUC
Đầu ra có thể là	1.5, 23.7, 100.8 (giá trị liên tục)	{0, 1} hoặc {Chó, Mèo, Cá}

II. Bài toán classification

Tại sao không dùng Linear Regression?

III. Hàm Sigmoid

III. Hàm Sigmoid

Mô hình tăng trưởng Logistic Growth

Logistic Growth (tăng trưởng logistic) mô tả **sự tăng trưởng giới hạn**, thường thấy trong tự nhiên — ví dụ:

- **Tăng trưởng dân số** trong môi trường có giới hạn tài nguyên
- **Sự phát triển của vi khuẩn** trong đĩa petri
- **Tăng trưởng thị phần** của một sản phẩm mới (S-curve adoption)

Mô hình được mô tả bởi **phương trình vi phân**:

$$\frac{dP}{dt} = rP\left(1 - \frac{P}{K}\right)$$

Trong đó:

- $P(t)$: quần thể tại thời điểm t
- r: tốc độ tăng trưởng nội tại
- K: **sức chứa môi trường (carrying capacity)** – mức tối đa mà môi trường chịu được

III. Hàm Sigmoid

Mô hình tăng trưởng Logistic Growth

Khi giải phương trình trên, ta được:

$$P(t) = \frac{K}{1 + Ae^{-rt}}$$

Với $A = \frac{K - P_0}{P_0}$, P_0 là giá trị ban đầu.

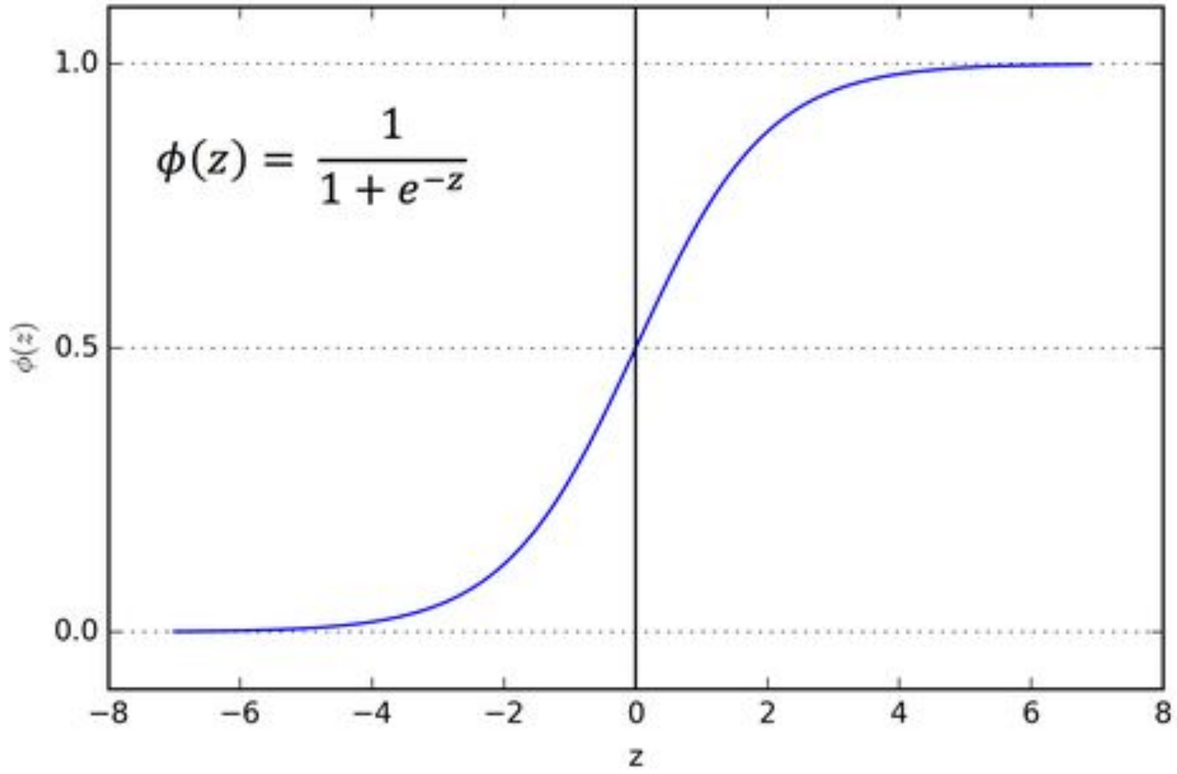
Đây chính là một **hàm sigmoid tổng quát**, vì nó có dạng:

$$P(t) = \frac{K}{1 + e^{-r(t - t_0)}}$$

III. Hàm Sigmoid

Hàm sigmoid là gì?

Hàm **sigmoid** là một hàm toán học thường được dùng trong **học sâu (deep learning)** và **học máy (machine learning)**, đặc biệt trong các **mạng nơ-ron**. Nó có dạng cong hình chữ "S", và được định nghĩa như sau:



III. Hàm Sigmoid

Tính chất

Tính chất	Chi tiết
Miền xác định	$(-\infty,+\infty)(-\infty, +\infty)$
Giá trị đầu ra	$(0,1)(0, 1)$ → rất thích hợp để mô hình hóa xác suất
Đạo hàm	$\sigma'(x)=\sigma(x)(1-\sigma(x))$ → đơn giản và thường dùng trong backpropagation
Đối xứng	Hàm sigmoid là **đối xứng tâm tại điểm (0, 0.5)**

IV. Logistic Regression

IV. Logistic Regression

Bài toán đặt ra

Logistic Regression là một mô hình phân loại nhị phân (binary classification), mục tiêu là dự đoán xác suất một đối tượng thuộc về lớp 1 (positive class) hay không (lớp 0 – negative class).

Cho tập dữ liệu huấn luyện:

$$\{(x^{(i)}, y^{(i)})\}_{i=1}^{N}$$

- $x^{(i)} \in \mathbb{R}^d$: vector đặc trưng (feature vector)
- $y^{(i)} \in \{0, 1\}$: nhãn lớp (label)

Hồi quy tuyến tính không phù hợp

$$h(x) = w^T x + b \tag{1}$$

- Output có thể là bất kỳ số thực, không thể diễn giải như xác suất.

Logistic Regression giải quyết bằng cách

- Dùng **hàm sigmoid** để "nén"output về khoảng [0, 1]:

$$\hat{y} = \sigma(w^T x + b) = \frac{1}{1 + e^{-(w^T x + b)}} \tag{2}$$

- Ý nghĩa: $\hat{y} = P(y = 1 \mid x)$ – xác suất mẫu x thuộc lớp 1.

IV. Logistic Regression

Bài toán đặt ra

Bối cảnh: Bạn là Data Scientist tại một công ty thương mại điện tử lớn. Công ty đang phát triển một hệ thống đề xuất sản phẩm thông minh và cần giải quyết nhiều vấn đề phân tích khác nhau.

Dữ liệu có sẵn:

- Thông tin khách hàng: tuổi, giới tính, thu nhập, địa điểm
- Lịch sử mua hàng: số lần mua, giá trị đơn hàng, danh mục sản phẩm
- Hành vi browsing: thời gian trên site, số trang xem, bounce rate
- Dữ liệu sản phẩm: giá, đánh giá, số lượng bán
- Dữ liệu marketing: email campaigns, ads clicked, promotions used

Các bài toán cần giải quyết:

A. Dự đoán tổng giá trị đơn hàng khách hàng sẽ mua trong 30 ngày tới để phân bổ ngân sách marketing

B. Dự đoán khả năng khách hàng abandon giỏ hàng để kích hoạt email reminder kịp thời

C. Dự đoán số lượng sản phẩm khách hàng sẽ mua trong một phiên shopping để tối ưu hóa giao diện

D. Dự đoán khả năng khách hàng trở thành premium member để targeting campaign đặc biệt

E. Dự đoán thời gian khách hàng sẽ dành trên website trong lần visit tiếp theo để tối ưu content loading

Câu hỏi: Phân loại các bài toán trên theo phương pháp phù hợp nhất:

1. A, C, E → Linear Regression; B, D → Logistic Regression

2. A, B, C → Linear Regression; D, E → Logistic Regression

3. A, C, D → Linear Regression; B, E → Logistic Regression

4. A, E → Linear Regression; B, C, D → Logistic Regression

IV. Logistic Regression

Odds và Log-odds

Odds là tỷ lệ giữa xác suất xảy ra một sự kiện (P) và xác suất không xảy ra sự kiện đó (1 - P). Trong ngữ cảnh hồi quy logistic, odds biểu thị khả năng một sự kiện thuộc lớp dương (1) so với lớp âm (0).

$$\text{odds} = \frac{p}{1-p}$$

Trong đó:

- P : Xác suất xảy ra sự kiện (ví dụ: xác suất một bệnh nhân bị bệnh).
- 1−P : Xác suất không xảy ra sự kiện (ví dụ: xác suất bệnh nhân không bị bệnh).

Ý nghĩa

- Nếu P=0.8, thì odds là: odds=0.8/(1-0.8)= 4
- Điều này có nghĩa là khả năng xảy ra sự kiện gấp 4 lần khả năng không xảy ra.
- Odds không bị giới hạn trong khoảng [0, 1] như xác suất, mà có thể nhận giá trị từ 0 đến vô cực:
  - Odds < 1: Sự kiện ít có khả năng xảy ra.
  - Odds = 1: Xác suất xảy ra và không xảy ra bằng nhau (P=0.5).
  - Odds > 1: Sự kiện có khả năng xảy ra cao hơn.

IV. Logistic Regression

Odds và Log-odds

Log-odds là logarit tự nhiên (ln) của odds. Nó chuyển odds từ một tỷ lệ (có thể từ 0 đến vô cực) sang một giá trị nằm trên toàn bộ dải số thực (từ −∞ đến +∞).

$$\text{Log-Odds} = \ln\left(\frac{P}{1-P}\right)$$

Ý nghĩa

- Log-odds là giá trị trung gian quan trọng trong hồi quy logistic, vì nó cho phép mô hình hóa mối quan hệ tuyến tính giữa các biến độc lập (features) và xác suất của sự kiện.
- Log-odds có tính chất tuyến tính, giúp hồi quy logistic sử dụng một hàm tuyến tính để dự đoán.

Trong hồi quy logistic, log-odds được biểu diễn như một hàm tuyến tính của các biến độc lập:

$$\text{Log-Odds} = \ln\left(\frac{P}{1-P}\right) = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_n x_n$$

Trong đó:

- β_0: Hằng số (intercept).
- $\beta_1, \beta_2, \ldots, \beta_n$: Hệ số của các biến độc lập $x_1, x_2, \ldots, x_n$.

IV. Logistic Regression

Ý nghĩa của hệ số Bi trong Logistic Regression

Hệ số β_i trong hồi quy logistic có ý nghĩa liên quan đến log-odds:

- Một đơn vị tăng trong biến x_i sẽ làm tăng (hoặc giảm, nếu $\beta_i < 0$) log-odds của sự kiện thêm β_i.
- Để diễn giải theo odds, ta lấy hàm mũ của hệ số:

$$e^{\beta_i}$$

  - Nếu $e^{\beta_i} > 1$: Một đơn vị tăng trong x_i làm tăng odds của sự kiện lên e^{β_i} lần.
  - Nếu $e^{\beta_i} < 1$: Một đơn vị tăng trong x_i làm giảm odds của sự kiện.

Giả sử mô hình hồi quy logistic cho kết quả:

$$\text{Log-Odds} = -2 + 0.5 \cdot \text{Age}$$

- Nếu tuổi (Age) tăng 1 đơn vị, log-odds tăng thêm 0.5.
- Odds sẽ tăng:

$$e^{0.5} \approx 1.648$$

Tức là odds tăng khoảng 64.8% khi tuổi tăng 1 đơn vị.

V. Cross - Entropy Loss Function

V. Cross-Entropy

Công thức Loss Function

Hàm mất mát cross-entropy (còn gọi là log-loss hoặc negative log-likelihood) được sử dụng để đánh giá sự khác biệt giữa phân phối xác suất dự đoán của mô hình và phân phối thực tế của dữ liệu. Trong hồi quy logistic, mục tiêu là dự đoán xác suất một quan sát thuộc vào lớp 1 ($P(Y=1|P)$), và cross-entropy loss đo lường mức độ sai lệch của xác suất dự đoán so với nhãn thực tế.

Ý nghĩa

- Cross-entropy loss phạt nặng hơn khi dự đoán của mô hình sai lệch lớn so với nhãn thực tế.
- Hàm này có tính chất lồi (convex), đảm bảo rằng quá trình tối ưu hóa (như gradient descent) sẽ hội tụ đến nghiệm tối ưu toàn cục.

V. Cross-Entropy

Công thức Loss Function

Hàm mất mát cho một quan sát

Cho một quan sát với:

- y: Nhãn thực tế ($y = 1$ nếu thuộc lớp dương, $y = 0$ nếu thuộc lớp âm).
- $\hat{p}$: Xác suất dự đoán rằng quan sát thuộc lớp 1, được tính bởi hàm sigmoid:

$$\hat{p} = P(Y = 1|X) = \frac{1}{1 + e^{-z}}, \quad z = \beta_0 + \beta_1 x_1 + \cdots + \beta_n x_n$$

Hàm mất mát cross-entropy cho một quan sát được định nghĩa là:

$$L(y, \hat{p}) = -\left[y \ln(\hat{p}) + (1 - y) \ln(1 - \hat{p})\right]$$

- Nếu $y = 1$: Hàm mất mát trở thành $-\ln(\hat{p})$. Mô hình bị phạt nặng nếu $\hat{p}$ nhỏ (dự đoán sai).
- Nếu $y = 0$: Hàm mất mát trở thành $-\ln(1 - \hat{p})$. Mô hình bị phạt nặng nếu $\hat{p}$ lớn (dự đoán sai).

V. Cross-Entropy

Công thức Loss Function

Hàm mất mát cho toàn bộ tập dữ liệu

Với tập dữ liệu có n quan sát, hàm mất mát trung bình (average cross-entropy loss) được tính như sau:

$$J(\beta) = -\frac{1}{n}\sum_{i=1}^{n}\left[y_i \ln(\hat{p}_i) + (1-y_i)\ln(1-\hat{p}_i)\right]$$

Trong đó:

- y_i: Nhãn thực tế của quan sát thứ i.
- $\hat{p}_i$: Xác suất dự đoán cho quan sát thứ i.
- β: Vector tham số $[\beta_0, \beta_1, \ldots, \beta_n]$.

Mục tiêu là **tối thiểu hóa** $J(\beta)$ bằng cách điều chỉnh các tham số β.

Hàm mất mát cross-entropy trong hồi quy logistic thực chất là negative log-likelihood, được suy ra từ **phương pháp ước lượng hợp lý tối đa (Maximum Likelihood Estimation - MLE).**

V. Cross-Entropy

Gradient Descent cho Loss Function

Quy trình tối ưu hóa hàm mất mác

1. **Khởi tạo**: Gán giá trị ban đầu cho β (thường là 0 hoặc ngẫu nhiên).
2. **Tính gradient**: Tính $\frac{\partial J(\beta)}{\partial \beta_j}$ cho tất cả các tham số.
3. **Cập nhật tham số**: Áp dụng công thức gradient descent để cập nhật β.
4. **Lặp lại**: Tiếp tục cho đến khi hàm mất mát hội tụ (thay đổi nhỏ hơn một ngưỡng) hoặc đạt số vòng lặp tối đa.

VI. Một số vấn đề trong Logistic Regression

VI. Một số vấn đề trong Lo.R

Các vấn đề trọng yếu

1. Giả định tuyến tính trong log-odds
2. Đa cộng tuyến (Multicollinearity)
3. Dữ liệu không cân bằng (Imbalanced Data)
4. Ngoại lệ (Outliers)
5. Quá khớp (Overfitting)
6. Kích thước mẫu nhỏ
7. Dữ liệu bị thiếu (Missing Data)
8. Không phù hợp của mô hình (Model Misspecification)
9. Phụ thuộc của quan sát
10. Diễn giải sai lệch của xác suất
11. Tính toán phức tạp với dữ liệu lớn
12. Tương quan giữa biến độc lập và biến phụ thuộc không đủ mạnh

VI. Một số vấn đề trong Lo.R

Giải pháp

STT	Vấn đề trọng yếu	Giải pháp chính
1	Giả định tuyến tính trong log-odds	Thêm biến tương tác, biến bậc cao; dùng mô hình phi tuyến như SVM, cây quyết định, mạng nơ-ron.
2	Đa cộng tuyến (Multicollinearity)	Kiểm tra VIF, loại/ghép biến tương quan cao, dùng PCA hoặc regularization (L1/L2).
3	Dữ liệu không cân bằng (Imbalanced Data)	Dùng resampling (SMOTE, undersampling), class weights, đánh giá bằng AUC-ROC hoặc PR-AUC.
4	Ngoại lệ (Outliers)	Phát hiện bằng IQR hoặc Z-score, loại bỏ hoặc dùng mô hình robust, thêm regularization.
5	Quá khớp (Overfitting)	Sử dụng regularization (L1, L2), cross-validation, giảm số đặc trưng.
6	Kích thước mẫu nhỏ	Thu thập thêm dữ liệu, dùng Bayesian estimation hoặc giảm số biến đầu vào.
7	Dữ liệu bị thiếu (Missing Data)	Loại bỏ, điền khuyết (trung bình, mô hình dự đoán), dùng EM.
8	Mô hình sai đặc tả (Model Misspecification)	Kiểm định giả định bằng đồ thị hoặc thống kê, chuyển sang mô hình phù hợp hơn.
9	Phụ thuộc giữa các quan sát	Dùng mô hình logistic hỗn hợp (mixed-effects), mô hình có hiệu ứng ngẫu nhiên.
10	Diễn giải sai lệch xác suất	Hiệu chỉnh xác suất (Platt Scaling), điều chỉnh ngưỡng phân loại.
11	Tính toán phức tạp với dữ liệu lớn	Dùng SGD, L-BFGS, huấn luyện theo mini-batch.
12	Tương quan yếu giữa đặc trưng và nhãn	Kiểm định thống kê (Wald test, LR test), thêm đặc trưng mạnh hơn.

VII. Softmax Regression

VII. Softmax Regression

Softmax Regression là gì?

Softmax Regression (hay còn gọi là **Multinomial Logistic Regression**) là một **phát triển tổng quát của Logistic Regression** dùng cho **bài toán phân loại đa lớp** (multi-class classification), trong khi Logistic Regression chỉ áp dụng cho bài toán phân loại **nhị phân** (binary classification).

- **Số lớp**: Giả sử có K lớp ($k = 1, 2, \ldots, K$).
- **Đầu ra tuyến tính**: Đối với mỗi lớp k, ta tính một giá trị tuyến tính (score) dựa trên các biến độc lập X :

$$z_k = \beta_{k0} + \beta_{k1}x_1 + \cdots + \beta_{kn}x_n$$

  - Ở đây, $\beta_k = [\beta_{k0}, \beta_{k1}, \ldots, \beta_{kn}]$ là vector hệ số cho lớp k.
  - Có K vector β_k, mỗi vector tương ứng với một lớp.
- **Hàm softmax**: Để chuyển các score z_k thành xác suất, sử dụng hàm softmax:

$$P(Y = k|X) = \frac{e^{z_k}}{\sum_{j=1}^{K} e^{z_j}}$$

  - e^{z_k}: Hàm mũ của score lớp k, biểu thị "độ mạnh" của lớp đó.
  - $\sum_{j=1}^{K} e^{z_j}$: Tổng các giá trị mũ của tất cả các lớp, dùng để chuẩn hóa.
  - Kết quả: $P(Y = k|X)$ là xác suất dự đoán cho lớp k, và tổng $\sum_{k=1}^{K} P(Y = k|X) = 1$.

VII. Softmax Regression

Loss Function và Tối ưu

- **Cross-entropy loss cho softmax**: Tương tự hồi quy logistic, softmax sử dụng cross-entropy loss để tối ưu hóa. Với nhãn y_i (giá trị từ 1 đến K) và xác suất dự đoán $\hat{p}_{ik} = P(Y = k|X_i)$, hàm mất mát là:

$$J(\beta) = -\frac{1}{n}\sum_{i=1}^{n}\sum_{k=1}^{K} y_{ik}\ln(\hat{p}_{ik})$$

  - $y_{ik} = 1$ nếu quan sát i thuộc lớp k, và 0 cho các lớp khác (one-hot encoding).
  - Mục tiêu là tối thiểu hóa $J(\beta)$, tương đương với tối đa hóa log-likelihood.

- **Gradient**: Đạo hàm của loss theo β_{jk} (hệ số của biến x_j cho lớp k) được tính bằng:

$$\frac{\partial J}{\partial \beta_{jk}} = \frac{1}{n}\sum_{i=1}^{n}(\hat{p}_{ik} - y_{ik})x_{ij}$$

  - Công thức này tương tự gradient của hồi quy logistic, nhưng áp dụng cho từng lớp k.

- **Phương pháp**: Sử dụng gradient descent, Newton-Raphson, hoặc các thuật toán tối ưu hóa khác.

VII. Regularization

VII. Regularization

Regularization là gì?

Mục tiêu chính của Regularization

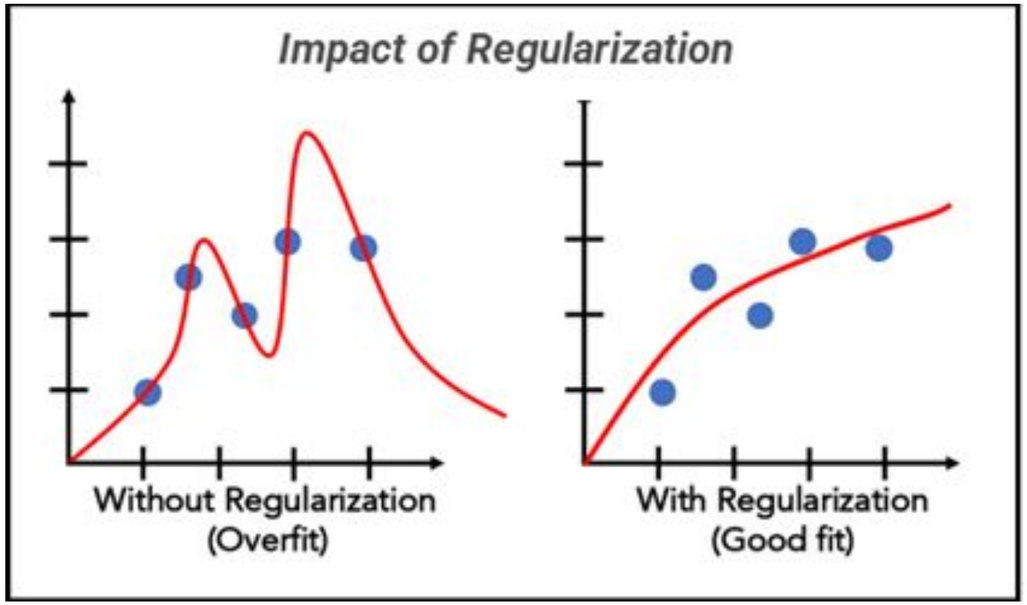
Cân bằng giữa độ khớp với dữ liệu và độ đơn giản của mô hình.

- Mô hình quá phức tạp (quá nhiều tham số, quá nhiều bậc tự do) → dễ **overfit**
- Regularization thêm một **ràng buộc (penalty)** vào hàm mất mát để **ưu tiên mô hình đơn giản hơn**.

Giả sử hàm mất mát ban đầu là $\mathcal{L}_{data}(\theta)$, regularization sẽ sửa thành:

$$\mathcal{L}_{reg}(\theta) = \mathcal{L}_{data}(\theta) + \lambda \cdot \mathcal{R}(\theta)$$

- $\mathcal{L}_{data}$: lỗi dữ liệu (như MSE, cross-entropy)
- $\mathcal{R}(\theta)$: hàm regularization
- λ: hệ số điều chỉnh mức độ regularization



VII. Regularization

L2 Regularization (Ridge)

- **Định nghĩa**: Hình phạt dựa trên bình phương của các tham số β, không bao gồm β_0 (hằng số) để tránh dịch chuyển toàn cục.
- **Công thức**:

$$R(\beta) = \frac{1}{2}\sum_{j=1}^{n}\beta_j^2$$

  - $\frac{1}{2}$ được thêm để đơn giản hóa đạo hàm (hệ số 2 sẽ triệt tiêu sau).
- **Hàm mất mát**:

$$J_{\text{Ridge}}(\beta) = -\frac{1}{n}\sum_{i=1}^{n}\sum_{k=1}^{K} y_{ik}\ln(\hat{p}_{ik}) + \frac{\lambda}{2}\sum_{j=1}^{n}\beta_j^2$$

VII. Regularization

L1 Regularization (Lasso)

- **Định nghĩa**: Hình phạt dựa trên giá trị tuyệt đối của các tham số β.
- **Công thức**:

$$R(\beta) = \sum_{j=1}^{n} |\beta_j|$$

- **Hàm mất mát**:

$$J_{\text{Lasso}}(\beta) = -\frac{1}{n} \sum_{i=1}^{n} \sum_{k=1}^{K} y_{ik} \ln(\hat{p}_{ik}) + \lambda \sum_{j=1}^{n} |\beta_j|$$

VII. Regularization

Tối ưu hóa

- **Cập nhật tham số:**
  - Với L2:

$$\beta_j \leftarrow \beta_j - \alpha \left(\frac{\partial J}{\partial \beta_j} + \lambda \beta_j \right)$$

  - Với L1:

$$\beta_j \leftarrow \beta_j - \alpha \left(\frac{\partial J}{\partial \beta_j} + \lambda \cdot \text{sign}(\beta_j) \right)$$

  - α: Tốc độ học.
- **Hành vi:**
  - L2: Thu nhỏ dần β_j theo hướng gradient cộng với lực kéo về 0.
  - L1: Tạo ngưỡng, khiến β_j có thể nhảy qua 0 nếu gradient đủ nhỏ.

ML IoT
HCMUT EE MACHINE LEARNING & IOT LAB

THANK YOU

CONTACT US

403.1 H6, BKHCM Campus 2

mlandiotlab@gmail.com

mliotlab.github.io

facebook.com/hcmut.ml.iot.lab

youtube.com/@mliotlab