TS. TRẦN QUANG QUÝ (Chủ biên), TS. NGÔ HỮU HUY (Đồng chủ biên),
TS. NGUYỄN ĐỨC BÌNH, TS. NGUYỄN TUẤN ANH

GIÁO TRÌNH

HỌC MÁY

NHÀ XUẤT BẢN ĐẠI HỌC THÁI NGUYÊN
NĂM 2026

MÃ SỐ: $\frac{\text{06 - 85}}{\text{ĐHTN - 2026}}$

Mục lục

Lời nói đầu 9

Chương 1. Các khái niệm cơ bản 11

1.1 Khái niệm học máy 11

1.2 Dữ liệu 13

1.3 Các bài toán cơ bản trong học máy 14

1.4 Phân nhóm các thuật toán học máy 21

1.5 Tham số mô hình và quá trình tối ưu hàm mất mát 28

1.6 Những thách thức chính của học máy 30

1.7 Sự đánh đổi giữa thiên lệch và phương sai 33

1.8 Quy trình thực hiện một dự án học máy 36

Chương 2. Các kỹ thuật xây dựng đặc trưng 40

2.1 Giới thiệu 40

2.2 Mô hình tổng quát cho các bài toán học máy 41

2.3 Một số kỹ thuật trích chọn đặc trưng 44

2.4 Chuẩn hóa vector đặc trưng 54

2.5 Kết luận 58

2.6 Câu hỏi ôn tập 59

Chương 3. Tối ưu hóa bằng hạ dốc 61
3.1 Giới thiệu 61
3.2 Gradient Descent cho hàm một biến 62
3.3 Gradient Descent cho hàm nhiều biến 67
3.4 Các biến thể của Gradient Descent 71
3.5 Kết luận 73
3.6 Câu hỏi ôn tập 73

Chương 4. Hồi quy tuyến tính 75
4.1 Phát biểu bài toán 75
4.2 Xây dựng và tối ưu hàm mất mát 76
4.3 Ví dụ trên Python 80
4.4 Thảo luận 83
4.5 Hồi quy Ridge và Lasso 85
4.6 Kết luận 88
4.7 Câu hỏi ôn tập 88

Chương 5. *K* láng giềng gần nhất 90
5.1 Giới thiệu 90
5.2 Phân tích toán học 92
5.3 Thực hành phân loại hoa IRIS 96
5.4 Ưu điểm và hạn chế của thuật toán KNN 99
5.5 Kết luận 101
5.6 Câu hỏi ôn tập 102

Chương 6. Bộ phân loại Naïve Bayes 104
6.1 Giới thiệu 104
6.2 Bài toán phân loại trong Naïve Bayes 107
6.3 Thực hành 114
6.4 Kết luận 117
6.5 Câu hỏi ôn tập 117

Chương 7. Mô hình nơ-ron Perceptron 120
7.1 Giới thiệu 120
7.2 Thuật toán học perceptron 121
7.3 Ví dụ và minh hoạ trên Python 125
7.4 Mô hình mạng neuron đầu tiên 126
7.5 Kết luận 129
7.6 Câu hỏi ôn tập 130

Chương 8. Cây quyết định 132
8.1 Khái niệm 133
8.2 Khái niệm về độ vẩn đục và độ tinh khiết 134
8.3 Các thuật toán xây dựng cây quyết định 138
8.4 Ví dụ xây dựng cây quyết định 142
8.5 Thực hành: Xây dựng mô hình cây quyết định bằng Python 146
8.6 Kết luận 151
8.7 Câu hỏi ôn tập 152

Chương 9. Hồi quy logistic 154
9.1 Giới thiệu 154
9.2 Hàm mất mát và phương pháp tối ưu 157
9.3 Triển khai thuật toán trên Python 159
9.4 Tính chất của hồi quy logistic 162
9.5 Bài toán phân biệt hai chữ số viết tay 164
9.6 Thảo luận 166
9.7 Kết luận 168
9.8 Câu hỏi ôn tập 168

Chương 10. Máy vector hỗ trợ - SVM 170
10.1 Mô hình máy vector hỗ trợ 170
10.2 Thực hành: Phân loại dữ liệu với SVM 178
10.3 Kết luận 180
10.4 Câu hỏi ôn tập 181

Chương 11. Phân cụm *K*-means 182
11.1 Giới thiệu 182
11.2 Phân tích toán học 184
11.3 Lựa chọn số lượng cụm tối ưu 186
11.4 Thực hành với Python 189
11.5 Thực hành phân vùng ảnh bằng thuật toán K-means 194

Chương 12. Phân cụm phân cấp 199
12.1 Khái niệm 199
12.2 Các chiến lược phân cụm phân cấp 203

12.3 Các tiêu chí liên kết trong phân cụm phân cấp 206
12.4 Thực hành trên Python 208
12.5 Kết luận 214
12.6 Câu hỏi ôn tập 214

Chương 13. Mạng Nơ ron nhân tạo 216
13.1 Khái niệm về mạng nơ-ron nhân tạo 217
13.2 Đơn vị cơ bản của mạng nơ-ron 217
13.3 Kiến trúc của mạng nơ-ron 221
13.4 Lan truyền tiến (Forward Propagation) 222
13.5 Hàm mất mát trong mạng nơ-ron 224
13.6 Thuật toán lan truyền ngược (Backpropagation) 226
13.7 Quá trình huấn luyện mạng Nơ-ron 228
13.8 Vanishing Gradient 230
13.9 Exploding Gradient 231
13.10 Các phương pháp khắc phục 234
13.11 Bài tập thực hành với Pytorch 236
13.12 Bài thực hành nhận dạng chữ số viết tay 240
13.13 Kết luận 244
13.14 Câu hỏi ôn tập 245

Chương 14. Một số dự án học máy tiêu biểu 248
14.1 Dự án 1: Dự đoán động đất bằng mô hình Random Forest 248
14.2 Dự án 2: Phân cụm khách hàng sử dụng K-means 255

Tài liệu tham khảo 274

Bảng 1: BẢNG CÁC KÝ HIỆU SỬ DỤNG TRONG GIÁO TRÌNH

Ký hiệu	Ý nghĩa
x, y, N, k	in nghiêng, thường hoặc hoa, là các số vô hướng
$\mathbf{x}, \mathbf{y}$	in đậm, chữ thường, là các vector
$\mathbf{X}, \mathbf{Y}$	in đậm, chữ hoa, là các ma trận
$\mathbb{R}$	tập hợp các số thực
$\mathbb{N}$	tập hợp các số tự nhiên
$\mathbb{C}$	tập hợp các số phức
$\mathbb{R}^m$	tập hợp các vector thực có m phần tử
$\mathbb{R}^{m\times n}$	tập hợp các ma trận thực có m hàng, n cột
$\mathbb{S}^n$	tập hợp các ma trận vuông đối xứng bậc n
$\mathbb{S}^n_+$	tập hợp các ma trận nửa xác định dương bậc n
$\mathbb{S}^n_{++}$	tập hợp các ma trận xác định dương bậc n
$\in$	phần tử thuộc tập hợp
$\exists$	tồn tại
$\forall$	mọi
$\triangleq$	ký hiệu là/bởi. Ví dụ $a \triangleq f(x)$ nghĩa là "ký hiệu $f(x)$ bởi a".
x_i	phần tử thứ i (tính từ 1) của vector $\mathbf{x}$
$\text{sgn}(x)$	hàm xác định dấu. Bằng 1 nếu $x \geq 0$, bằng -1 nếu $x < 0$.
$\exp(x)$	e^x
$\log(x)$	logarit *tự nhiên* của số thực dương x
$\underset{x}{\text{argmin}}\, f(x)$	giá trị của x để hàm $f(x)$ đạt giá trị nhỏ nhất
$\underset{x}{\text{argmax}}\, f(x)$	giá trị của x để hàm $f(x)$ đạt giá trị lớn nhất
a_{ij}	phần tử hàng thứ i, cột thứ j của ma trận $\mathbf{A}$
$\mathbf{A}^T$	chuyển vị của ma trận $\mathbf{A}$
$\mathbf{A}^H$	chuyển vị liên hợp (Hermitian) của ma trận phức $\mathbf{A}$
$\mathbf{A}^{-1}$	nghịch đảo của ma trận vuông $\mathbf{A}$, nếu tồn tại
$\mathbf{A}^\dagger$	giả nghịch đảo của ma trận không nhất thiết vuông $\mathbf{A}$
$\mathbf{A}^{-T}$	chuyển vị của nghịch đảo của ma trận $\mathbf{A}$, nếu tồn tại
$\|\mathbf{x}\|_p$	ℓ_p norm của vector $\mathbf{x}$
$\|\mathbf{A}\|_F$	Frobenius norm của ma trận $\mathbf{A}$
$\text{diag}(\mathbf{A})$	đường chéo chính của ma trận $\mathbf{A}$
$\text{trace}(\mathbf{A})$	trace của ma trận $\mathbf{A}$
$\det(\mathbf{A})$	định thức của ma trận vuông $\mathbf{A}$
$\text{rank}(\mathbf{A})$	hạng của ma trận $\mathbf{A}$
o.w	*otherwise* – trong các trường hợp còn lại
$\frac{\partial f}{\partial x}$	đạo hàm của hàm số f theo $x \in \mathbb{R}$
$\nabla_{\mathbf{x}} f$	gradient của hàm số f theo $\mathbf{x}$ ($\mathbf{x}$ là vector hoặc ma trận)
$\nabla^2_{\mathbf{x}} f$	gradient bậc hai của hàm số f theo $\mathbf{x}$, còn được gọi là *Hesse*
$\odot$	Hadamard product (elemenwise product). Phép nhân từng phần tử của hai vector hoặc ma trận cùng kích thước.
$\propto$	tỉ lệ với
———	đường nét liền
- - - - -	đường nét đứt
.....	đường nét chấm (đường chấm chấm)
—·—·—	đường chấm gạch
	nền chấm
	nền sọc chéo

Lời nói đầu

Trong bối cảnh chuyển đổi số và sự phát triển mạnh mẽ của Trí tuệ nhân tạo (AI), Học máy (Machine Learning) đã trở thành một trong những lĩnh vực cốt lõi, giữ vai trò nền tảng cho nhiều hướng nghiên cứu và ứng dụng hiện đại. Từ học sâu (Deep Learning), xử lý ngôn ngữ tự nhiên (Natural Language Processing), thị giác máy tính (Computer Vision) cho đến trí tuệ nhân tạo tạo sinh (Generative AI), tất cả đều được xây dựng dựa trên các nguyên lý và mô hình cơ bản của học máy. Vì vậy, việc nắm vững kiến thức nền tảng về học máy không chỉ có ý nghĩa học thuật mà còn mang tính định hướng lâu dài cho quá trình nghiên cứu và phát triển công nghệ.

Trong kỷ nguyên Dữ liệu lớn (Big Data), khối lượng dữ liệu được tạo ra mỗi ngày tăng với tốc độ rất lớn, đòi hỏi những phương pháp phân tích thông minh và có khả năng thích nghi cao. Học máy cung cấp cơ chế để hệ thống có thể tự động học từ dữ liệu, phát hiện các quy luật tiềm ẩn, xây dựng mô hình dự đoán và tối ưu hóa quyết định mà không cần lập trình tường minh cho từng tình huống cụ thể. Nhờ đặc tính này, học máy đã vượt ra khỏi phạm vi nghiên cứu thuần túy và trở thành công cụ thiết yếu trong nhiều lĩnh vực như công nghiệp, tài chính, y tế, giáo dục và quản trị.

Không chỉ dừng lại ở khả năng dự đoán, học máy còn góp phần hình thành tư duy phân tích dựa trên dữ liệu (data-driven approach), giúp người học hiểu rõ bản chất của quá trình mô hình hóa, cách đo lường sai số thông qua hàm mất mát (loss function), kiểm soát độ phức tạp của mô hình và đảm bảo khả năng tổng quát hóa (generalization) trên dữ liệu mới. Do đó, học máy không chỉ là một môn học kỹ thuật, mà còn là nền tảng phương pháp luận quan trọng cho việc phát triển các hệ thống AI hiện đại.

Cuốn giáo trình này được biên soạn với mục tiêu cung cấp cho người học một nền tảng lý thuyết vững chắc và có hệ thống về học máy. Nội dung giáo trình tập trung trình bày các mô hình học máy cơ bản như hồi quy tuyến tính, hồi quy logistic, phân loại và phân cụm; đồng thời làm rõ các khái niệm trọng tâm như hàm mất mát, tối ưu hóa, đánh giá mô hình, hiện tượng quá khớp (overfitting)

và thiếu khớp (underfitting). Các nội dung được trình bày theo lộ trình từ trực quan đến biểu diễn toán học, giúp người học hiểu sâu bản chất thay vì chỉ dừng lại ở thao tác công cụ.

Bên cạnh mục tiêu trang bị kiến thức chuyên môn về học máy, giáo trình còn được biên soạn theo định hướng đào tạo gắn với chuyển đổi số, nghiên cứu khoa học và đổi mới sáng tạo của Trường Đại học Công nghệ Thông tin và Truyền thông (ICTU) – Đại học Thái Nguyên. Nội dung giáo trình được xây dựng trên cơ sở kinh nghiệm giảng dạy, nghiên cứu và triển khai các ứng dụng trí tuệ nhân tạo trong thực tiễn, đồng thời chú trọng phát triển năng lực tư duy phân tích dữ liệu, năng lực giải quyết vấn đề và khả năng tự học công nghệ mới cho người học.

Thông qua các ví dụ minh họa, bài thực hành và dự án ứng dụng, giáo trình khuyến khích sinh viên vận dụng kiến thức học máy để giải quyết các bài toán thực tiễn trong giáo dục, quản trị, kinh doanh, nông nghiệp thông minh, y tế và chuyển đổi số. Đây cũng là những lĩnh vực đang được ICTU quan tâm phát triển nhằm đáp ứng nhu cầu nguồn nhân lực chất lượng cao trong thời đại kinh tế số và xã hội số.

Nội dung giáo trình được hình thành từ sự kết hợp giữa kinh nghiệm giảng dạy, nghiên cứu khoa học và quá trình cập nhật liên tục các tri thức mới trong lĩnh vực học máy trong nhiều năm qua. Bên cạnh việc kế thừa các nền tảng lý thuyết đã được kiểm chứng, giáo trình cũng tiếp cận những xu hướng và phương pháp hiện đại đang được sử dụng rộng rãi trong cộng đồng học thuật và công nghiệp. Cách tiếp cận này giúp bảo đảm tính khoa học, tính thực tiễn và khả năng kết nối giữa kiến thức cơ sở với các ứng dụng trí tuệ nhân tạo hiện nay.

Giáo trình được xây dựng hướng tới sinh viên ngành Công nghệ Thông tin, Khoa học Máy tính và các lĩnh vực liên quan, đồng thời cũng phù hợp với những độc giả mong muốn tiếp cận học máy một cách bài bản và có hệ thống. Nhóm biên soạn đã nỗ lực trình bày nội dung với tinh thần khoa học, chính xác và cập nhật; tuy nhiên khó tránh khỏi những hạn chế nhất định. Chúng tôi rất mong nhận được các ý kiến đóng góp từ quý độc giả và đồng nghiệp để giáo trình ngày càng được hoàn thiện hơn trong các lần tái bản tiếp theo.

Thái Nguyên, tháng 6 năm 2026
Nhóm biên soạn

Chương 1

Các khái niệm cơ bản

1.1. Khái niệm học máy

Học máy là một lĩnh vực khoa học nghiên cứu việc thiết kế và phát triển các hệ thống máy tính có khả năng học từ dữ liệu. Thay vì phải lập trình tường minh toàn bộ các quy tắc xử lý, cách tiếp cận này cho phép hệ thống tự động suy luận và hình thành các mô hình cũng như quy luật dựa trên việc phân tích dữ liệu quan sát được.

Hay theo một số định nghĩa mang tính kỹ thuật hơn:

Khái niệm 1

Học máy là một lĩnh vực nghiên cứu hướng tới việc trang bị cho máy tính khả năng học từ dữ liệu mà không cần phải được lập trình tường minh cho từng nhiệm vụ cụ thể.

Khái niệm 2

Học máy là quá trình xây dựng các hệ thống có khả năng tự động cải thiện hiệu năng của mình thông qua việc tích lũy và khai thác kinh nghiệm từ dữ liệu theo thời gian.

Khái niệm 3

Một chương trình máy tính được gọi là "học tập" từ kinh nghiệm E để hoàn thành nhiệm vụ T với hiệu quả được đo bằng phép đánh giá P, nếu hiệu quả của nó khi thực hiện nhiệm vụ T, khi được đánh giá bởi P, cải thiện theo kinh nghiệm E.

Định nghĩa trên làm rõ ba thành phần cơ bản của một hệ thống học máy: *nhiệm vụ* T mà hệ thống cần thực hiện; *phép đánh giá* P dùng để đo lường mức độ hoàn thành nhiệm vụ; và *kinh nghiệm* E, chính là dữ liệu mà hệ thống sử dụng để học. Việc xác định đúng ba yếu tố này có ý nghĩa quyết định trong quá trình thiết kế và xây dựng mô hình.

Nói một cách tổng quát, thuật toán học máy là những thuật toán có khả năng rút ra tri thức từ dữ liệu. Để xây dựng một hệ thống có khả năng học, cần xác định rõ ba thành phần cốt lõi gồm: nhiệm vụ cần thực hiện, tiêu chí đánh giá hiệu quả và đặc biệt là kinh nghiệm E, tức nguồn dữ liệu được sử dụng trong quá trình huấn luyện. Nếu thiếu dữ liệu hoặc không có cơ chế học phù hợp, việc xây dựng một hệ thống học máy là không khả thi.

Về mặt bản chất, học máy có thể được xem là quá trình tìm kiếm một mô hình hoặc một hàm ánh xạ từ không gian đầu vào $\mathcal{X}$ sang không gian đầu ra $\mathcal{Y}$ sao cho sai số dự đoán được giảm thiểu.

Giả sử ta có tập dữ liệu huấn luyện:

$$\mathcal{D} = \{(x_i, y_i)\}_{i=1}^{n}, \tag{1.1}$$

trong đó x_i là dữ liệu đầu vào và y_i là giá trị thực tế tương ứng. Ký hiệu mô hình cần học là $f_\theta(x)$ với θ là tập tham số. Khi đó, bài toán học máy có thể được phát biểu dưới dạng:

$$\theta^* = \arg\min_{\theta} \frac{1}{n} \sum_{i=1}^{n} \mathcal{L}\big(y_i, f_\theta(x_i)\big), \tag{1.2}$$

trong đó $\mathcal{L}$ là hàm mất mát đo lường mức độ sai lệch giữa giá trị dự đoán và giá trị thực tế. Công thức (1.2) cho thấy học máy về bản chất là một bài toán tối ưu hóa tham số mô hình.

Ví dụ minh họa trong thực tế: Giả sử ta muốn xây dựng một hệ thống dự đoán doanh thu bán hàng của một cửa hàng dựa trên dữ liệu lịch sử như số lượng khách mỗi ngày và chi phí quảng cáo. Mỗi ngày quan sát được tạo thành một cặp dữ liệu (x_i, y_i), trong đó x_i chứa thông tin đầu vào (khách, quảng cáo) và y_i là doanh thu thực tế.

Ban đầu, mô hình dự đoán có thể sai lệch đáng kể so với doanh thu thật. Thông qua việc điều chỉnh các tham số θ nhằm làm giảm giá trị trung bình của hàm mất mát trong công thức (1.2), sai số dự đoán dần được thu hẹp. Khi giá trị này đạt mức nhỏ nhất, ta thu được mô hình có khả năng dự đoán phù hợp nhất với dữ liệu đã quan sát.

1.2. Dữ liệu

Các *nhiệm vụ* trong học máy được mô tả thông qua cách một hệ thống xử lý một điểm dữ liệu đầu vào để tạo ra đầu ra mong muốn. Về mặt toán học, một mô hình học máy có thể được xem như một hàm

$$f_\theta : \mathcal{X} \to \mathcal{Y}, \tag{1.3}$$

trong đó $\mathcal{X}$ là không gian dữ liệu đầu vào, $\mathcal{Y}$ là không gian đầu ra, và θ là tập các tham số của mô hình. Đầu vào của hàm là dữ liệu quan sát được, và đầu ra là thông tin hoặc giá trị mà ta mong muốn suy ra từ dữ liệu đó.

Một điểm dữ liệu có thể tồn tại dưới nhiều dạng khác nhau như hình ảnh, tín hiệu âm thanh, văn bản, hoặc chuỗi hành vi của người dùng trên Internet. Tuy nhiên, để hệ thống máy tính có thể xử lý và học hiệu quả, các dữ liệu này thường được chuyển đổi thành dạng số. Khi đó, mỗi giá trị số đại diện cho một thuộc tính riêng biệt và được gọi là một *đặc trưng*.

Do đó, về mặt tính toán, mọi loại dữ liệu đều có thể được biểu diễn dưới dạng *vector đặc trưng*, ma trận hoặc tensor. Vector đặc trưng của một điểm dữ liệu thường được ký hiệu là

$$\mathbf{x} = (x_1, x_2, \ldots, x_d)^\top \in \mathbb{R}^d, \tag{1.4}$$

trong đó d là số lượng đặc trưng. Các mảng nhiều chiều về bản chất có thể được xem là đã được *vector hoá* thành mảng một chiều để thuận tiện cho việc biểu diễn và xử lý bằng các phép toán đại số tuyến tính. Kỹ thuật xây dựng vector đặc trưng cho dữ liệu sẽ được trình bày cụ thể hơn trong Chương 2.

Kinh nghiệm trong học máy là bộ dữ liệu được sử dụng để xây dựng mô hình. Giả sử ta có tập dữ liệu tổng thể

$$\mathcal{D} = \{(\mathbf{x}_i, y_i)\}_{i=1}^n, \tag{1.5}$$

trong đó $\mathbf{x}_i$ là vector đặc trưng và y_i là đầu ra tương ứng. Trong thực tế, để đánh giá và điều chỉnh mô hình một cách khách quan, bộ dữ liệu thường được chia thành ba tập con không giao nhau: tập huấn luyện $\mathcal{D}_{\text{train}}$, tập xác thực $\mathcal{D}_{\text{val}}$ và tập kiểm tra $\mathcal{D}_{\text{test}}$, sao cho:

$$\mathcal{D} = \mathcal{D}_{\text{train}} \cup \mathcal{D}_{\text{val}} \cup \mathcal{D}_{\text{test}}, \quad \mathcal{D}_{\text{train}} \cap \mathcal{D}_{\text{val}} \cap \mathcal{D}_{\text{test}} = \varnothing. \tag{1.6}$$

Tập huấn luyện được sử dụng để ước lượng các tham số θ của mô hình, tập xác thực được dùng để điều chỉnh các siêu tham số và lựa chọn mô hình phù hợp, trong khi tập kiểm tra đóng vai trò đánh giá cuối cùng nhằm ước lượng khả năng tổng quát hóa của mô hình trên dữ liệu chưa từng quan sát.

Tập huấn luyện (training set) bao gồm các điểm dữ liệu được sử dụng trực tiếp trong việc xây dựng mô hình, tức là để ước lượng các tham số θ thông qua việc tối thiểu hóa hàm mất mát trên tập dữ liệu này.

Tập kiểm tra (test set) được sử dụng để đánh giá hiệu quả cuối cùng của mô hình. Để đảm bảo tính khách quan và khả năng tổng quát hóa, tập dữ liệu này không được tham gia vào quá trình huấn luyện hay điều chỉnh mô hình. Một mô hình được xem là tốt khi đạt kết quả cao trên cả tập huấn luyện và tập kiểm tra, đồng thời sự chênh lệch giữa hai kết quả không đáng kể. Tập kiểm tra phản ánh các dữ liệu chưa từng được quan sát, tương ứng với các tình huống thực tế khi triển khai mô hình.

Để tăng hiệu quả của mô hình trên dữ liệu kiểm tra, người ta thường sử dụng thêm một tập dữ liệu trung gian được gọi là *tập xác thực* (validation set). Tập xác thực được dùng trong quá trình lựa chọn mô hình và điều chỉnh các siêu tham số, chẳng hạn như tốc độ học, độ phức tạp của mô hình hoặc các tham số điều chuẩn.

Quy trình xây dựng mô hình khi đó có thể được mô tả như sau: trước hết, mô hình được huấn luyện trên tập huấn luyện; tiếp theo, hiệu năng của mô hình được đánh giá trên tập xác thực để lựa chọn cấu hình tối ưu; cuối cùng, mô hình đã được lựa chọn sẽ được đánh giá một lần duy nhất trên tập kiểm tra để ước lượng khả năng tổng quát hóa thực sự.

Việc phân chia dữ liệu thành ba tập riêng biệt giúp đảm bảo rằng quá trình xây dựng mô hình không vô tình khai thác thông tin từ dữ liệu kiểm tra, qua đó duy trì tính khách quan và độ tin cậy của kết quả đánh giá.

1.3. Các bài toán cơ bản trong học máy

Nhiều bài toán phức tạp trong thực tiễn có thể được giải quyết hiệu quả bằng các phương pháp học máy. Những bài toán này thường có đặc điểm chung là dữ liệu đầu vào có kích thước lớn, cấu trúc phức tạp hoặc mối quan hệ giữa các biến không thể mô tả chính xác bằng các quy tắc lập trình tường minh.

Thay vì xây dựng các tập luật cố định, học máy cho phép mô hình tự động học ra quy luật từ dữ liệu quan sát được, từ đó đưa ra dự đoán, phân loại hoặc hỗ trợ ra quyết định. Chính nhờ khả năng thích nghi với dữ liệu và tự cải thiện hiệu năng theo thời gian, học máy đặc biệt phù hợp với các bài toán mà lời giải phụ thuộc mạnh vào mẫu dữ liệu thực tế.

Dưới đây là một số dạng bài toán phổ biến trong học máy:

1.3.1. Bài toán phân lớp

Phân lớp (classification) là một trong những bài toán cơ bản và quan trọng trong học máy. Nhiệm vụ của bài toán là xác định *nhãn* của một điểm dữ liệu từ tập gồm C nhãn có thể xảy ra.

Mỗi điểm dữ liệu được biểu diễn dưới dạng cặp $(\mathbf{x}, \mathbf{y})$, trong đó:

$$\mathbf{x} \in \mathbb{R}^d \tag{1.7}$$

là vector đặc trưng và

$$\mathbf{y} \in \mathcal{Y} \tag{1.8}$$

là nhãn tương ứng với:

$$\mathcal{Y} = \{1, 2, \ldots, C\}. \tag{1.9}$$

Việc xây dựng mô hình phân lớp tương đương với việc tìm một hàm ánh xạ:

$$\mathbf{f} : \mathbb{R}^d \to \mathcal{Y} \tag{1.10}$$

sao cho:

$$\mathbf{y} = \mathbf{f}(\mathbf{x}). \tag{1.11}$$

Hàm $\mathbf{f}$ được học từ dữ liệu huấn luyện và sau đó được áp dụng lên các dữ liệu mới nhằm xác định nhãn với sai số chấp nhận được.

Ví dụ 1: Phát hiện email rác

Bài toán xác định email rác là một ví dụ của phân lớp nhị phân với:

$$C = 2. \tag{1.12}$$

Trong đó:

- Nhiệm vụ T là xác định một email mới có phải là "rác" hay không.
- Phép đánh giá P có thể là tỉ lệ dự đoán đúng trên tập kiểm tra.
- Kinh nghiệm E là tập các cặp (email, nhãn) đã được gán nhãn trước đó để huấn luyện mô hình và xây dựng hàm $\mathbf{f}(\mathbf{x})$.

Phân lớp xác suất

Trong nhiều trường hợp, mô hình không xuất trực tiếp một nhãn rời rạc mà xuất ra một vector xác suất như sau:

$$\mathbf{p}(\mathbf{x}) = (p_1, p_2, \ldots, p_C). \tag{1.13}$$

Trong đó:

$$p_c = \mathbb{P}(\mathbf{y} = c \mid \mathbf{x}) \tag{1.14}$$

và:

$$\sum_{c=1}^{C} p_c = 1. \tag{1.15}$$

Nhãn dự đoán được xác định như sau:

$$\hat{\mathbf{y}} = \arg\max_{c \in \mathcal{Y}} p_c. \tag{1.16}$$

Biểu diễn xác suất cho phép đánh giá mức độ tin cậy của mô hình. Nếu xác suất lớn nhất vượt trội so với các xác suất còn lại, mô hình được xem là có độ chắc chắn cao. Ngược lại các xác suất gần nhau, khả năng phân loại nhầm sẽ cao hơn.

1.3.2. Hồi quy

Nếu tập đích $\mathcal{Y}$ bao gồm các giá trị thực, bài toán tương ứng được gọi là *hồi quy*. Mục tiêu khi đó là xây dựng một hàm ánh xạ từ không gian đặc trưng tới miền giá trị thực.

Cụ thể, ta cần tìm một hàm:

$$\mathbf{f} : \mathbb{R}^d \to \mathbb{R} \tag{1.17}$$

sao cho:

$$\mathbf{y} = \mathbf{f}(\mathbf{x}), \tag{1.18}$$

trong đó $\mathbf{x} \in \mathbb{R}^d$ là vector đặc trưng và $\mathbf{y} \in \mathbb{R}$ là giá trị đầu ra muốn dự đoán.

Khác với bài toán phân lớp, đầu ra của hồi quy là một đại lượng liên tục. Do đó, mô hình không đưa ra một nhãn rời rạc mà dự đoán một giá trị số phản ánh mức độ hoặc cường độ của một đại lượng quan tâm.

Ví dụ: Xét bài toán ước lượng giá của một căn nhà dựa trên các đặc trưng như diện tích x (m^2), số phòng ngủ y và khoảng cách đến trung tâm thành phố z (km). Khi đó, mỗi điểm dữ liệu có thể được biểu diễn dưới dạng:

$$\mathbf{x} = (x, y, z)^\top \in \mathbb{R}^3. \tag{1.19}$$

Ta mong muốn xây dựng một hàm dự đoán như sau:

$$\mathbf{f} : \mathbb{R}^3 \to \mathbb{R} \tag{1.20}$$

sao cho:

$$\text{Giá} = \mathbf{f}(x, y, z). \tag{1.21}$$

Bài toán hồi quy có thể được mở rộng sang trường hợp dự đoán đồng thời nhiều đại lượng đầu ra. Khi đó, hàm cần tìm có dạng:

$$\mathbf{f} : \mathbb{R}^d \to \mathbb{R}^m, \tag{1.22}$$

trong đó m là số lượng biến đầu ra.

Ví dụ, trong bài toán tạo ảnh độ phân giải cao từ một ảnh có độ phân giải thấp, mô hình cần dự đoán giá trị cường độ của nhiều điểm ảnh cùng một lúc. Khi đó, đầu ra là một vector hoặc một tensor nhiều chiều, và bài toán được xem là hồi quy nhiều đầu ra.

1.3.3. Máy dịch

Trong bài toán *máy dịch*, chương trình máy tính được yêu cầu dịch một đoạn văn từ một ngôn ngữ nguồn sang một ngôn ngữ đích. Về bản chất, đây là một bài toán học ánh xạ giữa hai không gian biểu diễn ngôn ngữ.

Ta có thể mô hình hóa bài toán như sau:

$$\mathbf{f} : \mathcal{X} \to \mathcal{Y}, \tag{1.23}$$

trong đó:

$$\mathbf{y} = \mathbf{f}(\mathbf{x}), \tag{1.24}$$

với $\mathbf{x}$ là câu hoặc đoạn văn trong ngôn ngữ nguồn và $\mathbf{y}$ là bản dịch tương ứng trong ngôn ngữ đích.

Dữ liệu huấn luyện là tập các cặp văn bản song ngữ:

$$\mathcal{D} = \{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1}^{n}, \tag{1.25}$$

trong đó mỗi cặp $(\mathbf{x}_i, \mathbf{y}_i)$ gồm một câu gốc và bản dịch tương ứng. Các văn bản này có thể chỉ gồm hai ngôn ngữ đang xét hoặc có thêm các ngôn ngữ trung gian nhằm cải thiện chất lượng dịch.

Ví dụ 1: Hệ thống dịch thông báo tại Trường Đại học Công nghệ Thông tin và Truyền thông Thái Nguyên (ICTU)

Giả sử tại ICTU muốn xây dựng một hệ thống tự động dịch các thông báo từ tiếng Việt sang tiếng Anh để phục vụ sinh viên quốc tế.

Một mẫu dữ liệu huấn luyện có thể là:

$$\mathbf{x} = \text{“Sinh viên ICTU nộp học phí trước ngày 15/09.”} \tag{1.26}$$

và:

$$\mathbf{y} = \text{“ICTU students must pay tuition before September 15.”} \tag{1.27}$$

Sau khi huấn luyện, mô hình học được hàm:

$$\mathbf{f} : \mathcal{X} \to \mathcal{Y} \tag{1.28}$$

sao cho:

$$\mathbf{y} = \mathbf{f}(\mathbf{x}). \tag{1.29}$$

Nếu ICTU đăng một thông báo mới như:

$$\mathbf{x}_{\text{new}} = \text{“Lịch thi học kỳ sẽ được công bố vào tuần tới.”} \tag{1.30}$$

Hệ thống có thể dự đoán bản dịch:

$$\hat{\mathbf{y}} = \text{“The semester exam schedule will be announced next week.”} \tag{1.31}$$

Ví dụ này cho thấy hệ thống máy dịch phải học được cấu trúc câu, ngữ cảnh và các thuật ngữ chuyên ngành để tạo ra bản dịch tự nhiên và chính xác.

Ví dụ 2: Dịch trong ứng dụng thực tế

Trong các ứng dụng như Google Translate hoặc các ứng dụng học ngoại ngữ, người dùng nhập một đoạn văn tiếng Việt:

$$\mathbf{x} = \text{“Hôm nay trời rất đẹp.”} \tag{1.32}$$

Hệ thống sẽ xuất ra bản dịch tiếng Anh:

$$\mathbf{y} = \text{“The weather is very nice today.”} \tag{1.33}$$

Ở đây, mô hình không chỉ dịch từng từ riêng lẻ mà còn phải hiểu ngữ cảnh và cấu trúc câu để tạo ra bản dịch tự nhiên và đúng ngữ pháp. Điều này cho thấy bài toán máy dịch là một bài toán phức tạp, trong đó đầu ra là một chuỗi ký tự có độ dài biến đổi và phụ thuộc mạnh vào ngữ cảnh của đầu vào.

Ví dụ Google translate là một ứng dụng (tất nhiên có nhiều kỹ thuật học sâu khác được áp dụng) học máy trong lĩnh vực máy dịch.

1.3.4. Phân cụm

Phân cụm (clustering) là bài toán chia tập dữ liệu $\mathcal{X}$ thành các nhóm nhỏ hơn dựa trên mức độ tương đồng giữa các điểm dữ liệu. Khác với bài toán phân lớp, trong phân cụm dữ liệu huấn luyện không có nhãn. Mô hình phải tự động phát hiện cấu trúc tiềm ẩn và chia dữ liệu thành các cụm sao cho các điểm trong cùng một cụm có mức độ tương đồng cao hơn so với các điểm thuộc cụm khác.

Giả sử tập dữ liệu gồm n điểm:

$$\mathcal{X} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\}, \tag{1.34}$$

mục tiêu của phân cụm là tìm một phép gán nhãn cụm:

$$\mathbf{g} : \mathcal{X} \to \{1, 2, \ldots, K\}, \tag{1.35}$$

trong đó K là số lượng cụm cần chia.

Ví dụ: Phân cụm khách hàng trong thương mại điện tử

Xét một trang web thương mại điện tử theo dõi hành vi của người dùng như số lần xem sản phẩm, loại sản phẩm quan tâm, giá trị đơn hàng trung bình hoặc tần suất mua sắm. Mỗi người dùng được biểu diễn bởi một vector đặc trưng $\mathbf{x}$ phản ánh hành vi mua hàng của họ.

Sau khi áp dụng thuật toán phân cụm, hệ thống có thể chia người dùng thành các nhóm như:

- Nhóm khách hàng thường xuyên mua sản phẩm công nghệ.
- Nhóm khách hàng quan tâm đến thời trang và mỹ phẩm.
- Nhóm khách hàng chỉ mua khi có khuyến mãi.

Việc phân cụm này giúp doanh nghiệp hiểu rõ hơn về cấu trúc tập khách hàng mà không cần nhãn có sẵn. Từ đó, hệ thống có thể đề xuất sản phẩm hoặc quảng cáo phù hợp với từng nhóm người dùng dựa trên hành vi chung của các thành viên trong cùng cụm.

Bài toán phân cụm vì vậy đóng vai trò quan trọng trong khai phá dữ liệu, phân tích hành vi người dùng và hỗ trợ ra quyết định trong nhiều lĩnh vực thực tế.

1.3.5. Hoàn thiện dữ liệu

Hoàn thiện dữ liệu là bài toán nhằm dự đoán các giá trị còn thiếu trong tập dữ liệu. Mục tiêu của bài toán là tận dụng mối quan hệ giữa các điểm dữ liệu để ước lượng những giá trị chưa được quan sát.

Giả sử ta có một tập dữ liệu được biểu diễn dưới dạng ma trận:

$$\mathbf{R} \in \mathbb{R}^{m \times n}, \tag{1.36}$$

trong đó phần tử R_{ij} biểu thị mức độ tương tác giữa đối tượng thứ i và mục thứ j (ví dụ: số sao đánh giá). Một số phần tử của $\mathbf{R}$ chưa được biết và cần dự đoán.

Bài toán hoàn thiện dữ liệu có thể được mô tả như sau:

$$\hat{R}_{ij} = \mathbf{f}(i, j), \tag{1.37}$$

trong đó $\hat{R}_{ij}$ là giá trị ước lượng thay cho phần tử còn thiếu.

Ví dụ: Hệ thống khuyến nghị

Xét một hệ thống đánh giá phim với 3 người dùng và 3 bộ phim. Ma trận đánh giá có dạng:

$$\mathbf{R} = \begin{pmatrix} 5 & 4 & ? \\ 4 & ? & 2 \\ ? & 3 & 1 \end{pmatrix}. \tag{1.38}$$

Dấu "?" biểu thị các giá trị bị thiếu trong dữ liệu. Hệ thống cần ước lượng những giá trị này dựa trên mối tương quan giữa người dùng hoặc giữa các bộ phim.

Ví dụ, nếu hai người dùng có xu hướng đánh giá giống nhau ở nhiều bộ phim, hệ thống có thể ước lượng rằng họ cũng sẽ có đánh giá tương tự cho bộ phim chưa xem. Giá trị dự đoán được xác định theo công thức:

$$\hat{R}_{ij} = \frac{\sum_{k \in \mathcal{N}(i)} w_{ik} R_{kj}}{\sum_{k \in \mathcal{N}(i)} w_{ik}}, \tag{1.39}$$

trong đó $\mathcal{N}(i)$ là tập các người dùng có mức độ tương đồng với người dùng i, và w_{ik} biểu diễn hệ số đo lường độ tương tự giữa hai người dùng.

Các ứng dụng khác

Ngoài hệ thống khuyến nghị, hoàn thiện dữ liệu còn xuất hiện trong:

- Khôi phục ảnh cũ hoặc ảnh bị hỏng bằng cách dự đoán các điểm ảnh bị thiếu.
- Tô màu ảnh đen trắng dựa trên học từ các ảnh màu tương tự.

Bên cạnh đó, học máy còn bao gồm nhiều dạng bài toán khác như: *xếp hạng*, *giảm chiều dữ liệu*, *phát hiện bất thường*, tùy thuộc vào yêu cầu đầu ra, đặc điểm dữ liệu và khả năng giải quyết của thuật toán.

1.4. Phân nhóm các thuật toán học máy

Phân nhóm dựa theo phương thức học có các loại thuật toán học máy sau:

1.4.1. Học có giám sát

Một thuật toán học máy được gọi là *học có giám sát* nếu việc xây dựng mô hình được thực hiện dựa trên các cặp dữ liệu đầu vào và đầu ra đã biết trước. Nói cách khác, trong tập huấn luyện, mỗi điểm dữ liệu đều có nhãn tương ứng.

Giả sử tập huấn luyện gồm các cặp:

$$\mathcal{D} = \{(\mathbf{x}_1, \mathbf{y}_1), \ldots, (\mathbf{x}_N, \mathbf{y}_N)\}, \tag{1.40}$$

trong đó $\mathbf{x}_i$ là dữ liệu đầu vào và $\mathbf{y}_i$ là đầu ra mong muốn.

Mục tiêu của học có giám sát là xây dựng một hàm dự đoán:

$$\mathbf{y} = \mathbf{f}(\mathbf{x}), \tag{1.41}$$

sao cho với các dữ liệu trong tập huấn luyện, giá trị $\mathbf{f}(\mathbf{x}_i)$ gần với $\mathbf{y}_i$ nhất có thể. Sau khi huấn luyện, mô hình được sử dụng để dự đoán đầu ra cho các dữ liệu mới chưa từng xuất hiện trong tập huấn luyện.

Bản chất của quá trình huấn luyện là điều chỉnh tham số của mô hình nhằm làm giảm sai số giữa giá trị dự đoán và giá trị thực tế. Nếu mô hình học tốt, nó không chỉ khớp với dữ liệu huấn luyện mà còn dự đoán chính xác trên dữ liệu mới.

Ví dụ 1: Dạy em bé nhận biết chữ cái

Học có giám sát có thể được hiểu tương tự như quá trình dạy một em bé học bảng chữ cái.

Giả sử ta đưa cho em bé xem một hình ảnh chữ cái và nói rõ đó là chữ gì. Ví dụ:

- Hình ảnh chữ "A" $\rightarrow$ nói với bé: "Đây là chữ A".
- Hình ảnh chữ "B" $\rightarrow$ nói với bé: "Đây là chữ B".
- Hình ảnh chữ "C" $\rightarrow$ nói với bé: "Đây là chữ C".

Trong học máy, mỗi hình ảnh được xem như một đầu vào $\mathbf{x}$ và chữ cái tương ứng là nhãn $\mathbf{y}$.

Tập huấn luyện khi đó có dạng:

$$\mathcal{D} = \{(\mathbf{x}_1, \mathbf{y}_1), \ldots, (\mathbf{x}_N, \mathbf{y}_N)\}, \tag{1.42}$$

trong đó mỗi $\mathbf{x}_i$ là một ảnh chữ cái và $\mathbf{y}_i \in \{A, B, C, \dots\}$ là nhãn đúng đã biết.

Quá trình học diễn ra như sau: em bé nhìn chữ, đoán tên chữ; nếu đoán sai, người lớn sẽ sửa lại. Việc sửa sai này chính là cơ chế "giám sát".

Tương tự, trong học máy, mô hình đưa ra dự đoán:

$$\hat{\mathbf{y}} = \mathbf{f}(\mathbf{x}), \tag{1.43}$$

sau đó so sánh với nhãn đúng $\mathbf{y}$ để điều chỉnh tham số bên trong.

Sau khi đã được dạy nhiều lần với các ví dụ có nhãn, khi gặp một hình ảnh chữ mới mà chưa từng thấy, em bé có thể nhận ra đúng chữ đó. Đây chính là khả năng tổng quát hóa của học có giám sát.

Ví dụ 2 về học máy có giám sát:

Giả sử ta muốn xây dựng một hệ thống dự đoán kết quả "Đạt" hay "Không đạt" của sinh viên ICTU trong một học phần dựa trên hai thông tin:

- x_1: số buổi đi học,
- x_2: điểm giữa kỳ.

Trong quá trình huấn luyện, ta đã có dữ liệu của các sinh viên khóa trước, với mỗi sinh viên đều biết rõ kết quả cuối cùng. Tập dữ liệu có dạng:

$$\mathcal{D} = \{(\mathbf{x}_1, \mathbf{y}_1), \dots, (\mathbf{x}_N, \mathbf{y}_N)\}, \tag{1.44}$$

trong đó:

- $\mathbf{x}_i = (x_{1i}, x_{2i})$ là thông tin đầu vào,
- $\mathbf{y}_i \in \{\text{Đạt}, \text{Không đạt}\}$ là kết quả thực tế đã biết.

Vì mỗi điểm dữ liệu đều đã có nhãn $\mathbf{y}_i$, mô hình có thể so sánh giữa dự đoán và kết quả thật để điều chỉnh tham số của mình.

Sau khi huấn luyện, với một sinh viên mới:

$$\mathbf{x}_{\text{new}} = (x_1, x_2), \tag{1.45}$$

mô hình sẽ dự đoán:

$$\hat{\mathbf{y}} = \mathbf{f}(\mathbf{x}_{\text{new}}). \tag{1.46}$$

Điểm mấu chốt ở đây là: trong quá trình học, mô hình luôn được "giám sát" bởi các nhãn đúng đã biết trước. Nếu dự đoán sai so với nhãn thực tế, mô hình sẽ điều chỉnh tham số để giảm sai số. Chính sự tồn tại của nhãn đầu ra trong tập huấn luyện làm nên bản chất của học có giám sát.

1.4.2. Học không giám sát

Một thuật toán học máy được gọi là *học không giám sát* nếu dữ liệu huấn luyện không chứa nhãn đầu ra. Nói cách khác, mô hình chỉ được cung cấp các điểm dữ liệu đầu vào và phải tự khám phá cấu trúc hoặc quy luật tiềm ẩn bên trong dữ liệu đó.

Giả sử tập dữ liệu có dạng:

$$\mathcal{X} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N\}, \tag{1.47}$$

trong đó không tồn tại các nhãn $\mathbf{y}_i$ tương ứng.

Mục tiêu của học không giám sát là tìm ra một cấu trúc ẩn, một cách biểu diễn mới, hoặc một cách phân nhóm dữ liệu sao cho các điểm có tính chất tương đồng được đặt gần nhau hơn trong cùng một nhóm.

Ví dụ: Phân nhóm sinh viên ICTU theo hành vi học tập

Giả sử ta thu thập dữ liệu về sinh viên ICTU gồm các thông tin:

- Số giờ tự học mỗi tuần,
- Số buổi đi học,
- Số lần truy cập hệ thống học trực tuyến.

Mỗi sinh viên được biểu diễn bởi một vector:

$$\mathbf{x} = (x_1, x_2, x_3). \tag{1.48}$$

Trong trường hợp này, ta không biết trước sinh viên thuộc nhóm nào (ví dụ: chăm học, học trung bình hay ít tham gia). Thuật toán học không giám sát sẽ tự động phân chia sinh viên thành các nhóm dựa trên mức độ tương đồng của các đặc trưng.

Kết quả có thể cho thấy:

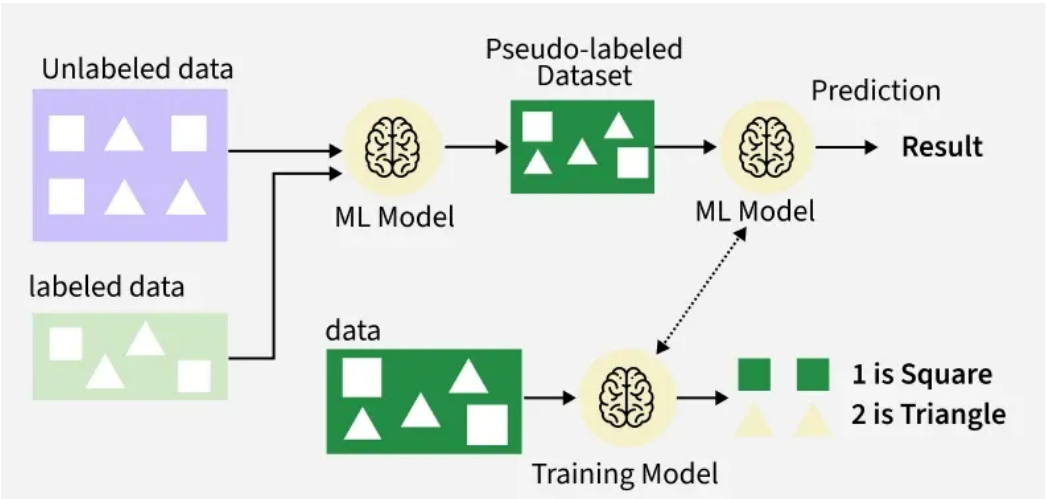
- Một nhóm sinh viên có số giờ tự học cao và tham gia đầy đủ,
- Một nhóm có mức độ tham gia trung bình,
- Một nhóm ít tham gia học tập.

Ở đây, không có nhãn đúng được cung cấp trước. Mô hình tự tìm ra cấu trúc dữ liệu mà không có sự "giám sát" từ nhãn đầu ra.

1.4.3. Học bán giám sát

Học bán giám sát là một hướng tiếp cận trong học máy, trong đó tập huấn luyện bao gồm cả dữ liệu có nhãn và dữ liệu không có nhãn. Cụ thể, mô hình được cung cấp một số mẫu đã biết đầu ra để học, đồng thời khai thác thêm lượng lớn dữ liệu chưa được gán nhãn nhằm nâng cao khả năng dự đoán.

Trong thực tế, phương pháp này ngày càng được quan tâm do quá trình thu thập và gán nhãn dữ liệu thường đòi hỏi nhiều thời gian, chi phí và nguồn lực chuyên môn, trong khi dữ liệu thô lại dồi dào và dễ tiếp cận. Việc tận dụng hiệu quả dữ liệu chưa gán nhãn giúp cải thiện hiệu năng mô hình mà không làm tăng đáng kể chi phí gán nhãn.



Hình 1.1. Minh họa mô hình học bán giám sát: kết hợp dữ liệu có nhãn và không có nhãn trong quá trình huấn luyện.

Hình minh họa cho thấy quy trình học bán giám sát thông qua việc kết hợp dữ liệu có nhãn và dữ liệu không có nhãn trong cùng một hệ thống.

Ban đầu, ta có hai nguồn dữ liệu:

- *Labeled data*: tập dữ liệu đã có nhãn, ví dụ các hình vuông và tam giác đã biết rõ thuộc lớp nào.
- *Unlabeled data*: tập dữ liệu chưa có nhãn, tức chỉ có hình dạng nhưng chưa biết đó là hình vuông hay tam giác.

Bước đầu tiên, mô hình học máy được huấn luyện trên phần dữ liệu có nhãn. Sau khi học được một mức độ nhất định, mô hình này được sử dụng để dự đoán

nhãn cho phần dữ liệu chưa có nhãn. Kết quả của bước này tạo ra một tập dữ liệu mới gọi là *pseudo-labeled dataset*, tức là dữ liệu được gán nhãn tạm thời dựa trên dự đoán của mô hình.

Tiếp theo, tập dữ liệu gán nhãn tạm thời này được kết hợp với tập dữ liệu có nhãn ban đầu để huấn luyện lại mô hình. Quá trình này giúp mô hình tận dụng thêm thông tin từ dữ liệu chưa có nhãn, từ đó cải thiện khả năng phân loại.

Trong ví dụ minh họa ở hình, mô hình cuối có thể đưa ra kết quả dự đoán:

- Nhãn 1 tương ứng với hình vuông.
- Nhãn 2 tương ứng với hình tam giác.

Điểm quan trọng của học bán giám sát thể hiện rõ qua hình: mặc dù ban đầu chỉ có một số ít dữ liệu được gán nhãn, mô hình vẫn có thể khai thác cấu trúc của toàn bộ tập dữ liệu để nâng cao độ chính xác. Việc sử dụng dữ liệu không có nhãn làm tăng lượng thông tin sẵn có mà không cần tốn chi phí gán nhãn.

Ví dụ: Ảnh y học trong chẩn đoán bệnh

Giả sử một bệnh viện thu thập được hàng nghìn ảnh chụp X-quang phổi. Tuy nhiên, chỉ một phần nhỏ trong số đó được bác sĩ chuyên khoa gán nhãn là "bình thường" hoặc "có dấu hiệu bệnh". Việc gán nhãn đòi hỏi chuyên môn cao và nhiều thời gian, nên không thể thực hiện cho toàn bộ dữ liệu.

Trong trường hợp này, mô hình học bán giám sát sẽ:

- Sử dụng các ảnh đã có nhãn để học mối quan hệ giữa đặc trưng hình ảnh và tình trạng bệnh.
- Đồng thời khai thác cấu trúc của tập ảnh chưa có nhãn để nhận diện các mẫu tương đồng, từ đó cải thiện khả năng phân loại.

Nhờ kết hợp cả dữ liệu có nhãn và không có nhãn, mô hình có thể đạt độ chính xác cao hơn so với việc chỉ sử dụng lượng nhỏ dữ liệu đã được gán nhãn.

Học bán giám sát vì vậy đóng vai trò quan trọng trong các lĩnh vực mà dữ liệu dồi dào nhưng nhãn lại khan hiếm, như y tế, xử lý ảnh, nhận dạng tiếng nói và phân tích hành vi người dùng.

1.4.4. Học tăng cường

Bên cạnh học có giám sát và học không giám sát, tồn tại một nhóm thuật toán học máy khác không yêu cầu tập dữ liệu huấn luyện cố định. Thay vào đó, mô hình

học cách ra quyết định thông qua quá trình tương tác trực tiếp với môi trường. Nhóm phương pháp này được gọi là *học củng cố* (reinforcement learning).

Trong học củng cố, tại mỗi thời điểm, hệ thống (gọi là tác nhân) quan sát trạng thái hiện tại của môi trường và thực hiện một hành động. Sau hành động đó, môi trường phản hồi lại bằng một giá trị đánh giá gọi là *điểm thưởng* và chuyển sang trạng thái mới. Quá trình này lặp lại nhiều lần, và mô hình dần điều chỉnh chiến lược hành động để đạt kết quả tốt hơn.

Mục tiêu của học củng cố là tìm một chính sách ra quyết định sao cho tổng điểm thưởng tích lũy là lớn nhất. Nếu ký hiệu r_t là điểm thưởng tại thời điểm t, thì tổng điểm thưởng trong một quá trình có thể được biểu diễn như sau:

$$G = \sum_{t=0}^{T} r_t, \tag{1.49}$$

trong đó T là số bước tương tác.

Khi đó, mục tiêu học tập có thể được diễn đạt như sau:

$$\max_{\pi} \ \mathbb{E}[G], \tag{1.50}$$

trong đó π là chính sách ra quyết định và $\mathbb{E}[G]$ là kỳ vọng của tổng điểm thưởng.

Khác với học có giám sát, học củng cố không có nhãn đúng cho từng hành động mà chỉ nhận phản hồi qua điểm thưởng để tự học chiến lược tối ưu bằng quá trình thử và sai.

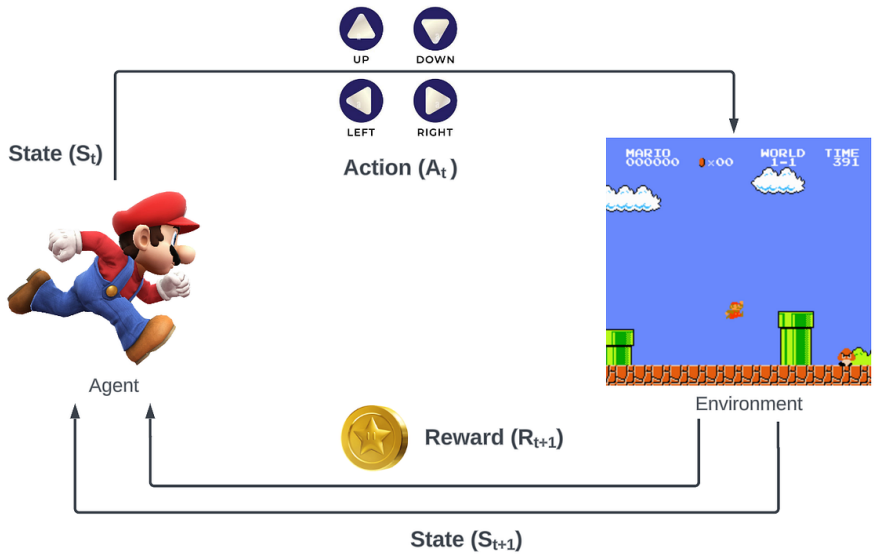
Ví dụ minh họa

Ví dụ 1: Chơi cờ hoặc trò chơi điện tử

Một chương trình chơi cờ không được cung cấp trước nước đi đúng cho từng tình huống. Thay vào đó, nó thử nhiều nước đi khác nhau, và sau mỗi ván cờ sẽ nhận được điểm thưởng (thắng, hòa hoặc thua). Qua nhiều lần chơi, hệ thống học được chiến lược giúp tăng xác suất chiến thắng.

Ví dụ 2: Xe tự lái

Trong bài toán xe tự lái, tác nhân phải quyết định tăng tốc, phanh hay rẽ hướng dựa trên trạng thái hiện tại của môi trường. Nếu xe di chuyển an toàn và đúng lộ trình, hệ thống nhận điểm thưởng dương; nếu va chạm hoặc đi sai đường, điểm thưởng sẽ thấp hoặc âm. Qua quá trình tương tác liên tục, mô hình học được chính sách lái xe tối ưu.



Hình 1.2. Minh họa học tăng cường thông qua trò chơi Mario: tác nhân tương tác với môi trường và nhận điểm thưởng.

Hình 1.2 minh họa cơ chế học tăng cường thông qua trò chơi Mario. Trong ví dụ này, Mario đóng vai trò là *tác nhân* (agent), còn thế giới trò chơi là *môi trường* (environment).

Tại mỗi thời điểm, Mario quan sát *trạng thái* hiện tại (vị trí của mình, vị trí ống nước, kẻ địch, hố sâu phía trước) và phải lựa chọn một *hành động*: nhảy, đi trái, đi phải hoặc đứng yên.

Sau khi thực hiện hành động, môi trường phản hồi lại bằng:

- Điểm thưởng (reward): ví dụ nhảy qua hố an toàn được cộng điểm.
- Trạng thái mới: Mario tiến gần hơn đến cột cờ hoặc... rơi xuống hố.

Nếu Mario nhảy đúng lúc và vượt qua chướng ngại vật, mô hình nhận điểm thưởng dương. Nếu Mario va phải kẻ địch hoặc rơi xuống hố, điểm thưởng có thể rất thấp hoặc âm (và người chơi thì hơi "đau tim").

Điều quan trọng là: không ai nói cho Mario biết "bây giờ phải nhảy". Không tồn tại nhãn đúng cho từng hành động cụ thể. Mario chỉ biết hành động nào tốt hay xấu thông qua điểm thưởng nhận được sau đó.

Qua hàng nghìn, thậm chí hàng triệu lần chơi, mô hình học được chiến lược tối ưu: khi thấy hố thì nhảy, khi thấy nấm thì ăn, khi thấy kẻ địch thì tránh hoặc dẫm lên.

Đây chính là bản chất của học tăng cường: học thông qua thử và sai, và dần dần tối đa hóa tổng điểm thưởng thay vì học từ các cặp dữ liệu đã gán nhãn trước.

Các cách phân loại khác

Ngoài cách phân loại theo mức độ giám sát, các thuật toán học máy còn có thể được phân loại theo nhiều tiêu chí khác:

- *Học dựa trên mẫu* và *học dựa trên mô hình*: phân biệt giữa phương pháp lưu trữ trực tiếp dữ liệu huấn luyện và phương pháp xây dựng mô hình tổng quát.
- *Học theo batch* và *học trực tuyến*: phân biệt giữa huấn luyện trên toàn bộ dữ liệu cùng một lúc và cập nhật mô hình dần dần khi dữ liệu mới xuất hiện.

1.5. Tham số mô hình và quá trình tối ưu hàm mất mát

Mỗi mô hình học máy được mô tả thông qua một tập các đại lượng gọi là *tham số mô hình* (model parameters), quyết định cách thức mô hình ánh xạ từ dữ liệu đầu vào sang đầu ra dự đoán. Chẳng hạn, trong hồi quy tuyến tính, các hệ số của biến đầu vào đóng vai trò là tham số; còn trong mạng nơ-ron nhân tạo, các trọng số và độ lệch của các lớp mạng là những tham số cần được học.

Mục tiêu của thuật toán học máy là tìm ra bộ tham số tối ưu sao cho mô hình đạt khả năng dự đoán tốt nhất trên dữ liệu, tức là các giá trị dự đoán càng gần với giá trị thực tế càng tốt.

Để đánh giá mức độ phù hợp của mô hình, người ta sử dụng *hàm mất mát* (loss function), là một hàm số đo lường sai lệch giữa giá trị dự đoán và giá trị thực. Giá trị của hàm này càng nhỏ thì hiệu năng của mô hình càng cao.

Trong bài toán phân loại, hiệu quả của mô hình thường được thể hiện qua tỷ lệ dự đoán chính xác, trong khi đối với bài toán hồi quy, tiêu chí đánh giá dựa trên mức độ sai khác giữa giá trị dự đoán và giá trị thực của biến đầu ra.

Giả sử tập tham số của mô hình được ký hiệu là θ, khi đó hàm mất mát tương ứng có thể được biểu diễn dưới dạng $\mathcal{L}(\theta)$ hoặc $J(\theta)$. Khi đó, quá trình huấn luyện mô hình có thể được diễn đạt như một bài toán tối ưu hóa:

$$\theta^* = \underset{\theta}{\operatorname{argmin}}\, \mathcal{L}(\theta). \tag{1.51}$$

Ký hiệu $\operatorname{argmin}_\theta$ biểu thị giá trị của θ làm cho hàm số $\mathcal{L}(\theta)$ đạt giá trị nhỏ nhất. Nói cách khác, θ^* là bộ tham số tối ưu của mô hình.

Như vậy, quá trình xây dựng một mô hình học máy thực chất là quá trình giải một bài toán tối ưu. Trong quá trình này, thuật toán liên tục điều chỉnh các tham số của mô hình nhằm giảm dần giá trị của hàm mất mát. Quá trình đó chính là quá trình "học" của mô hình.

Ví dụ về hàm mất mát

Xét một bài toán hồi quy đơn giản trong đó ta muốn dự đoán giá trị đầu ra y dựa trên một biến đầu vào x. Mô hình hồi quy tuyến tính có dạng:

$$\hat{y} = wx + b \tag{1.52}$$

trong đó:

- w là hệ số của biến đầu vào,
- b là hệ số chệch (bias),
- $\hat{y}$ là giá trị dự đoán của mô hình.

Hai đại lượng w và b chính là các tham số của mô hình. Để đánh giá chất lượng của mô hình, ta có thể sử dụng hàm mất mát *sai số trung bình bình phương* (Mean Squared Error – MSE):

$$\mathcal{L}(w, b) = \frac{1}{2N} \sum_{i=1}^{N} (y_i - \hat{y}_i)^2 \tag{1.53}$$

trong đó:

- N là số lượng mẫu dữ liệu,
- y_i là giá trị thực của mẫu thứ i,
- $\hat{y}_i$ là giá trị dự đoán của mô hình.

Hệ số $\frac{1}{2}$ thường được thêm vào để thuận tiện cho việc tính đạo hàm trong các thuật toán tối ưu như Gradient Descent.

Mục tiêu của quá trình huấn luyện là tìm các giá trị w và b sao cho hàm mất mát $\mathcal{L}(w, b)$ đạt giá trị nhỏ nhất. Khi đó, đường thẳng hồi quy sẽ xấp xỉ tốt nhất mối quan hệ giữa biến đầu vào x và biến đầu ra y.

Sai số huấn luyện

Sai số huấn luyện (training error) phản ánh mức độ sai khác giữa giá trị dự đoán của mô hình và giá trị thực trên *tập dữ liệu huấn luyện*. Trong nhiều trường hợp, sai số huấn luyện chính là giá trị của hàm mất mát được tính trên toàn bộ tập dữ liệu huấn luyện:

$$\text{Training Error} = \frac{1}{2N_{\text{train}}} \sum_{i=1}^{N_{\text{train}}} (y_i - \hat{y}_i)^2 \tag{1.54}$$

Giá trị này cho biết mô hình khớp với dữ liệu huấn luyện tốt đến mức nào.

Sai số kiểm tra

Sau khi quá trình huấn luyện hoàn tất, mô hình cần được đánh giá trên một tập dữ liệu độc lập gọi là *tập kiểm tra* (test set). Tập dữ liệu này không được sử dụng trong quá trình huấn luyện nhằm đảm bảo việc đánh giá là khách quan.

Sai số kiểm tra được xác định bởi:

$$\text{Test Error} = \frac{1}{2N_{\text{test}}} \sum_{i=1}^{N_{\text{test}}} (y_i - \hat{y}_i)^2 \tag{1.55}$$

Việc sử dụng giá trị trung bình là cần thiết do kích thước của tập huấn luyện và tập kiểm tra có thể không đồng nhất. Sai số trên tập kiểm tra phản ánh khả năng *tổng quát hóa* (generalization) của mô hình đối với các dữ liệu chưa từng xuất hiện trong quá trình huấn luyện.

1.6. Những thách thức chính của học máy

Mặc dù học máy đã đạt được nhiều thành tựu đáng kể trong các lĩnh vực như thị giác máy tính, xử lý ngôn ngữ tự nhiên và phân tích dữ liệu lớn, việc xây dựng một hệ thống học máy hiệu quả vẫn đối mặt với nhiều thách thức. Các thách thức này thường xuất phát từ hai nguồn chính: *dữ liệu* và *thuật toán*.

Trong thực tế, chất lượng của một hệ thống học máy phụ thuộc rất lớn vào dữ liệu huấn luyện và khả năng của thuật toán trong việc học được các quy luật từ dữ liệu đó. Nếu dữ liệu không phù hợp hoặc thuật toán không đủ khả năng biểu diễn các mối quan hệ trong dữ liệu, mô hình sẽ khó đạt được hiệu quả cao và có thể dẫn đến các dự đoán sai lệch.

1.6.1. Các thách thức liên quan đến dữ liệu

Thiếu dữ liệu huấn luyện

Nhiều thuật toán học máy, đặc biệt là các mô hình học sâu, yêu cầu một lượng lớn dữ liệu để có thể học được các mẫu và cấu trúc tiềm ẩn trong dữ liệu. Nếu số lượng dữ liệu quá ít, mô hình sẽ không đủ thông tin để học được quy luật tổng quát của bài toán.

Ví dụ, trong các hệ thống nhận diện hình ảnh, các mô hình hiện đại thường cần hàng trăm nghìn hoặc hàng triệu ảnh để đạt được độ chính xác cao. Khi dữ liệu huấn luyện hạn chế, mô hình dễ rơi vào tình trạng học không đầy đủ và cho kết quả dự đoán kém chính xác.

Dữ liệu không mang tính đại diện

Một tập dữ liệu huấn luyện cần phản ánh được các đặc điểm của dữ liệu mà hệ thống sẽ gặp trong thực tế. Nếu dữ liệu huấn luyện chỉ bao gồm một số trường hợp đặc biệt hoặc bị thiên lệch, mô hình có thể học được những quy luật sai lệch và hoạt động kém khi gặp dữ liệu mới.

Ví dụ, nếu một hệ thống nhận diện khuôn mặt được huấn luyện chủ yếu trên dữ liệu của một nhóm người nhất định, hệ thống có thể hoạt động kém đối với các nhóm khác. Hiện tượng này thường được gọi là *thiên lệch dữ liệu* (data bias).

Dữ liệu nhiễu và sai lệch

Trong nhiều ứng dụng thực tế, dữ liệu thường chứa lỗi đo lường, giá trị ngoại lai hoặc các thông tin không chính xác. Những yếu tố này có thể làm sai lệch quá trình học của mô hình và khiến mô hình khó phát hiện được các cấu trúc thực sự tồn tại trong dữ liệu.

Do đó, một bước quan trọng trong nhiều dự án học máy là *tiền xử lý dữ liệu* (data preprocessing), bao gồm các công việc như phát hiện ngoại lai, xử lý giá trị thiếu và chuẩn hóa dữ liệu.

Các đặc trưng không phù hợp

Chất lượng dữ liệu không chỉ phụ thuộc vào số lượng mẫu mà còn phụ thuộc vào các đặc trưng (features) được sử dụng để mô tả dữ liệu. Nếu các đặc trưng không liên quan đến bài toán, mô hình sẽ khó có thể học được các quy luật có ý nghĩa.

Vì vậy, việc xây dựng các đặc trưng phù hợp, thường được gọi là *thiết kế đặc trưng* (feature engineering), đóng vai trò quan trọng trong quá trình xây dựng mô hình học máy. Quá trình này có thể bao gồm:

- Lựa chọn các đặc trưng quan trọng từ tập dữ liệu ban đầu.
- Kết hợp các đặc trưng hiện có để tạo ra các đặc trưng mới có ý nghĩa hơn.
- Thu thập thêm dữ liệu để bổ sung các đặc trưng cần thiết.

1.6.2. Các thách thức liên quan đến mô hình và thuật toán

Hiện tượng quá khớp (Overfitting)

Quá khớp (overfitting) xảy ra khi mô hình học quá sát các đặc điểm của dữ liệu huấn luyện, bao gồm cả nhiễu và những biến động ngẫu nhiên. Khi đó, mô hình thường đạt kết quả rất cao trên tập huấn luyện nhưng lại suy giảm hiệu năng khi áp dụng trên dữ liệu mới.

Hiện tượng này thường xuất hiện khi độ phức tạp của mô hình không phù hợp với quy mô dữ liệu huấn luyện. Để hạn chế quá khớp, có thể áp dụng một số giải pháp sau:

- Đơn giản hóa mô hình.
- Bổ sung thêm dữ liệu huấn luyện.
- Sử dụng các kỹ thuật điều chuẩn (regularization).

Hiện tượng dưới khớp (Underfitting)

Ngược lại với quá khớp, dưới khớp xảy ra khi mô hình quá đơn giản và không đủ khả năng học được cấu trúc của dữ liệu. Khi đó, mô hình sẽ cho kết quả kém cả trên tập huấn luyện và tập kiểm tra.

Để khắc phục hiện tượng này, ta có thể:

- Sử dụng mô hình phức tạp hơn.
- Cung cấp thêm các đặc trưng có ý nghĩa cho mô hình.
- Giảm mức độ điều chuẩn của mô hình.

Lựa chọn và điều chỉnh siêu tham số

Bên cạnh các tham số được học trong quá trình huấn luyện, nhiều thuật toán còn sử dụng *siêu tham số* (hyperparameters). Đây là những giá trị được xác định trước khi bắt đầu huấn luyện, có ảnh hưởng đáng kể đến hiệu năng của mô hình.

Chẳng hạn, trong thuật toán Gradient Descent, tốc độ học (learning rate) là một siêu tham số quan trọng. Nếu giá trị này quá lớn, quá trình tối ưu có thể không hội tụ; ngược lại, nếu quá nhỏ, tốc độ học sẽ trở nên rất chậm.

Việc lựa chọn các siêu tham số phù hợp thường cần thông qua nhiều lần thử nghiệm và đóng vai trò quan trọng trong quá trình xây dựng mô hình học máy.

1.6.3. Ảnh hưởng của các thách thức trong học máy

Những thách thức nêu trên không chỉ ảnh hưởng đến độ chính xác của mô hình mà còn có thể gây ra các hệ quả đáng kể trong các ứng dụng thực tế. Nếu dữ liệu huấn luyện bị thiên lệch hoặc mô hình không được huấn luyện đúng cách, hệ thống có thể đưa ra các dự đoán sai lệch hoặc thiếu công bằng.

Trong các lĩnh vực quan trọng như y tế, tài chính hoặc giao thông tự động, những sai lệch này có thể dẫn đến các quyết định không chính xác và gây ra những hậu quả nghiêm trọng. Vì vậy, việc hiểu rõ các thách thức trong học máy và áp dụng các phương pháp phù hợp để khắc phục chúng là một bước quan trọng trong quá trình phát triển các hệ thống trí tuệ nhân tạo đáng tin cậy.

1.7. Sự đánh đổi giữa thiên lệch và phương sai

Một trong những vấn đề trung tâm trong việc xây dựng mô hình học máy là tìm được sự cân bằng hợp lý giữa khả năng học của mô hình trên dữ liệu huấn luyện và khả năng khái quát hóa trên dữ liệu mới. Hai khái niệm quan trọng giúp giải thích hiện tượng này là *thiên lệch* (bias) và *phương sai* (variance).

1.7.1. Thiên lệch (Bias)

Thiên lệch (bias) biểu thị mức độ sai lệch giữa giá trị dự đoán trung bình của mô hình và mối quan hệ thực sự trong dữ liệu. Mô hình có thiên lệch cao thường quá đơn giản, không đủ khả năng nắm bắt các cấu trúc phức tạp.

Ví dụ, khi dữ liệu có quan hệ phi tuyến nhưng mô hình sử dụng là tuyến tính, mô hình sẽ không thể biểu diễn chính xác mối quan hệ giữa các biến. Khi đó, sai số sẽ lớn ngay cả trên tập huấn luyện, dẫn đến hiện tượng *dưới khớp* (underfitting).

Các đặc điểm thường gặp của mô hình có thiên lệch cao:

- Sai số trên tập huấn luyện lớn.
- Sai số trên tập kiểm tra cũng lớn.
- Mô hình quá đơn giản so với độ phức tạp của dữ liệu.

1.7.2. Phương sai (Variance)

Phương sai (variance) thể hiện mức độ nhạy của mô hình trước sự thay đổi của dữ liệu huấn luyện. Mô hình có phương sai cao thường ghi nhớ quá chi tiết các đặc trưng của dữ liệu, bao gồm cả nhiễu và các dao động ngẫu nhiên.

Khi đó, mô hình có thể đạt sai số rất thấp trên tập huấn luyện nhưng lại suy giảm hiệu năng trên dữ liệu mới, dẫn đến hiện tượng *quá khớp* (overfitting).

Các đặc điểm điển hình của mô hình có phương sai cao:

- Sai số huấn luyện rất thấp.
- Sai số trên tập kiểm tra cao.
- Mô hình có độ phức tạp lớn so với quy mô dữ liệu huấn luyện.

1.7.3. Sự đánh đổi giữa thiên lệch và phương sai

Trong thực tế, tồn tại sự đánh đổi giữa thiên lệch và phương sai: việc giảm một đại lượng thường dẫn đến sự gia tăng của đại lượng còn lại. Khi sử dụng mô hình phức tạp hơn, khả năng biểu diễn được cải thiện giúp giảm thiên lệch, nhưng đồng thời mô hình trở nên nhạy hơn với nhiễu, làm tăng phương sai.

Ngược lại, mô hình đơn giản thường có phương sai thấp hơn nhưng lại gặp hạn chế trong việc nắm bắt cấu trúc dữ liệu, dẫn đến thiên lệch cao.

Vì vậy, mục tiêu trong xây dựng mô hình học máy là đạt được sự cân bằng hợp lý giữa thiên lệch và phương sai, nhằm đảm bảo mô hình vừa học được các quy luật chính của dữ liệu, vừa duy trì khả năng tổng quát hóa tốt trên dữ liệu chưa từng quan sát.

1.7.4. Ví dụ minh họa

Xét một tập dữ liệu gồm các điểm quan sát trong đó đầu ra phụ thuộc phi tuyến vào biến đầu vào. Nếu ta sử dụng một mô hình tuyến tính để xấp xỉ dữ liệu, mô hình sẽ không thể biểu diễn được cấu trúc phi tuyến và dẫn đến sai số lớn. Đây là trường hợp mô hình có *thiên lệch cao*.

Ngược lại, nếu ta sử dụng một đa thức bậc rất cao để khớp với dữ liệu huấn luyện, mô hình có thể đi qua gần như tất cả các điểm dữ liệu. Tuy nhiên, mô hình này có thể dao động mạnh giữa các điểm dữ liệu và cho dự đoán kém trên dữ liệu mới. Đây là trường hợp mô hình có *phương sai cao*.

Hình 1.3 minh họa mối quan hệ giữa độ phức tạp của mô hình và sai số dự đoán. Trục hoành biểu diễn độ phức tạp của mô hình (model complexity), trong khi

trục tung biểu diễn sai số dự đoán (prediction error). Khi độ phức tạp của mô hình thay đổi, các thành phần của sai số dự đoán cũng thay đổi theo những xu hướng khác nhau.

Giả sử dữ liệu được sinh ra theo mô hình

$$y = f(x) + \epsilon \tag{1.56}$$

trong đó $f(x)$ là hàm thực sự mô tả mối quan hệ giữa biến đầu vào và đầu ra, còn ϵ là nhiễu ngẫu nhiên với $\mathbb{E}[\epsilon] = 0$ và $\mathrm{Var}(\epsilon) = \sigma^2$.

Một mô hình học máy sau khi được huấn luyện sẽ xây dựng một hàm dự đoán $\hat{f}(x)$. Sai số dự đoán trung bình của mô hình tại một điểm x có thể được phân rã thành ba thành phần chính theo công thức

$$\mathbb{E}\big[(y - \hat{f}(x))^2\big] = \underbrace{(\mathbb{E}[\hat{f}(x)] - f(x))^2}_{\text{Bias}^2} + \underbrace{\mathbb{E}\big[(\hat{f}(x) - \mathbb{E}[\hat{f}(x)])^2\big]}_{\text{Variance}} + \sigma^2. \tag{1.57}$$

Trong đó, thành phần thứ nhất $(\mathbb{E}[\hat{f}(x)] - f(x))^2$ được gọi là *bình phương thiên lệch* (Bias^2), phản ánh mức độ sai lệch giữa mô hình trung bình và hàm thực sự của dữ liệu. Thành phần thứ hai là *phương sai* (Variance), thể hiện mức độ nhạy cảm của mô hình đối với sự thay đổi của dữ liệu huấn luyện. Thành phần σ^2 là nhiễu không thể tránh khỏi của dữ liệu.

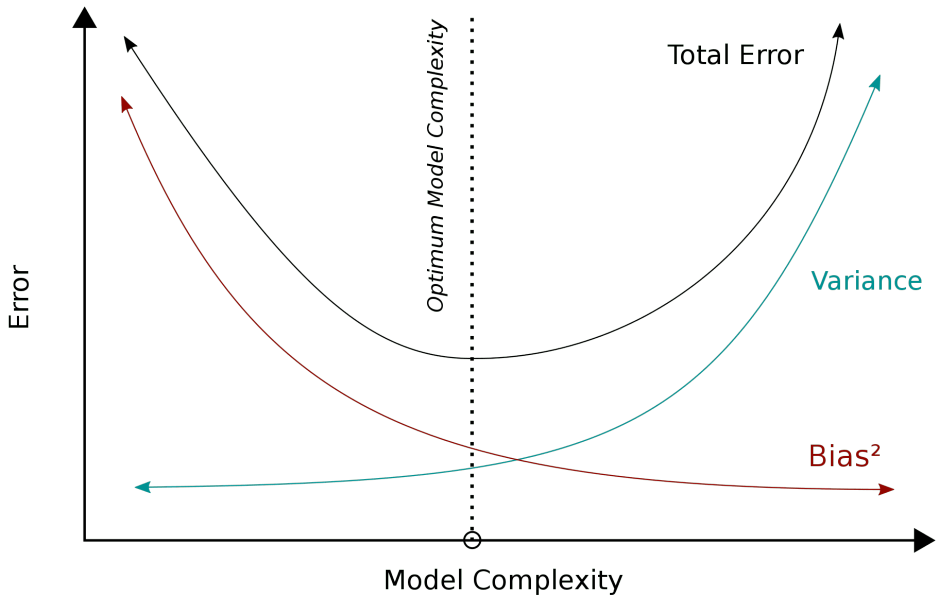
Trong hình, đường màu đỏ biểu diễn sự thay đổi của Bias^2. Khi mô hình có độ phức tạp thấp, Bias^2 thường lớn vì mô hình quá đơn giản để mô tả cấu trúc thực của dữ liệu. Khi độ phức tạp của mô hình tăng lên, Bias^2 giảm dần do mô hình có khả năng biểu diễn dữ liệu tốt hơn.

Ngược lại, đường màu xanh biểu diễn phương sai của mô hình. Khi mô hình đơn giản, phương sai thường nhỏ vì mô hình ít phụ thuộc vào từng điểm dữ liệu cụ thể. Tuy nhiên, khi độ phức tạp của mô hình tăng lên, mô hình có xu hướng học cả các nhiễu trong dữ liệu huấn luyện, làm cho phương sai tăng nhanh.

Sai số tổng thể của mô hình, được biểu diễn bởi đường cong màu đen, có dạng chữ U vì nó là tổng của hai thành phần Bias^2 và Variance (ngoài ra còn có thành phần nhiễu không đổi). Khi mô hình quá đơn giản, sai số lớn do Bias^2 cao; khi mô hình quá phức tạp, sai số lại tăng do Variance lớn. Giữa hai vùng này tồn tại một điểm tối ưu, tại đó sai số tổng thể đạt giá trị nhỏ nhất. Điểm này được biểu diễn trong hình bởi đường thẳng đứng nét đứt và được gọi là *độ phức tạp tối ưu của mô hình* (optimal model complexity).

Việc lựa chọn mô hình trong học máy thực chất là tìm ra mức độ phức tạp phù hợp để đạt được sự cân bằng giữa thiên lệch và phương sai, từ đó giúp mô hình

vừa học được các quy luật của dữ liệu huấn luyện vừa duy trì khả năng khái quát hóa tốt đối với dữ liệu mới.



Hình 1.3. Minh họa sự đánh đổi giữa thiên lệch và phương sai

1.7.5. Ý nghĩa trong thực tế

Hiểu được sự đánh đổi giữa thiên lệch và phương sai giúp chúng ta lựa chọn mô hình phù hợp và thiết kế các chiến lược huấn luyện hiệu quả hơn. Nhiều kỹ thuật trong học máy hiện đại, chẳng hạn như điều chuẩn (regularization), cắt tỉa cây quyết định, hay phương pháp ensemble, đều nhằm mục tiêu kiểm soát sự cân bằng giữa hai yếu tố này.

Trong các chương tiếp theo, nhiều thuật toán tối ưu như Gradient Descent cùng với các kỹ thuật điều chuẩn sẽ được trình bày nhằm giảm sai số tổng thể của mô hình thông qua việc kiểm soát sự cân bằng giữa thiên lệch và phương sai.

1.8. Quy trình thực hiện một dự án học máy

Một dự án học máy thường bao gồm nhiều bước khác nhau, từ việc xác định bài toán đến triển khai hệ thống trong thực tế. Mặc dù quy trình có thể thay đổi tùy thuộc vào từng ứng dụng, phần lớn các dự án học máy đều trải qua các bước cơ bản sau:

1. Xác định bài toán và mục tiêu của hệ thống

2. Thu thập và hiểu dữ liệu

3. Khám phá và phân tích dữ liệu
4. Chuẩn bị dữ liệu cho quá trình huấn luyện
5. Huấn luyện và lựa chọn mô hình học máy phù hợp
6. Tinh chỉnh mô hình và đánh giá hiệu năng
7. Triển khai và giám sát hệ thống trong thực tế

Các bước trên không nhất thiết phải tuân theo một trình tự cố định. Trong thực tế, sau khi đánh giá mô hình, ta thường quay lại các bước trước đó để cải thiện dữ liệu hoặc lựa chọn mô hình phù hợp hơn.

Để làm rõ quy trình này, ta xét một ví dụ minh họa về bài toán dự đoán giá nhà.

Ví dụ: Bài toán dự đoán giá nhà

Giả sử một công ty bất động sản muốn xây dựng một hệ thống có khả năng ước lượng giá bán của một căn nhà dựa trên các đặc trưng của căn nhà đó.

1. Xác định bài toán

   Bài toán được mô hình hóa như một bài toán hồi quy có giám sát. Đầu vào mô hình là các đặc trưng của căn nhà, còn đầu ra là giá bán của căn nhà đó.

2. Thu thập dữ liệu

   Dữ liệu có thể được thu thập từ các giao dịch bất động sản trước đó. Mỗi mẫu dữ liệu có thể bao gồm các thuộc tính như:

   - Diện tích căn nhà
   - Số phòng ngủ
   - Khoảng cách tới trung tâm thành phố
   - Năm xây dựng
   - Giá bán thực tế

3. Khám phá dữ liệu

   Ở bước này ta tiến hành phân tích dữ liệu để hiểu rõ đặc điểm của tập dữ liệu, ví dụ như phân phối của giá nhà, các điểm dữ liệu ngoại lai hoặc mối tương quan giữa các thuộc tính.

4. Chuẩn bị dữ liệu

   Dữ liệu cần được xử lý trước khi đưa vào mô hình, chẳng hạn như:

   - Xử lý các giá trị bị thiếu
   - Chuẩn hóa các đặc trưng
   - Chuyển các biến dạng hạng mục thành dạng số

   Sau đó dữ liệu được chia thành tập huấn luyện và tập kiểm tra.

5. Huấn luyện mô hình

   Giả sử mỗi căn nhà được biểu diễn bởi một vector đặc trưng

$$\mathbf{x} = (x_1, x_2, x_3) \tag{1.58}$$

   trong đó

   - x_1 là diện tích căn nhà
   - x_2 là số phòng ngủ
   - x_3 là khoảng cách tới trung tâm thành phố

   Một mô hình hồi quy tuyến tính đơn giản có thể được sử dụng để dự đoán giá nhà:

$$\hat{y} = w_1 x_1 + w_2 x_2 + w_3 x_3 + b \tag{1.59}$$

   trong đó $\hat{y}$ là giá nhà dự đoán, w_1, w_2, w_3 là các tham số của mô hình và b là hệ số chệch.

   Các tham số của mô hình được học từ dữ liệu huấn luyện bằng cách tối thiểu hóa hàm mất mát, ví dụ như sai số trung bình bình phương:

$$\mathcal{L}(w, b) = \frac{1}{N} \sum_{i=1}^{N} (y_i - \hat{y}_i)^2 \tag{1.60}$$

6. Tinh chỉnh mô hình

   Sau khi huấn luyện, các siêu tham số của mô hình có thể được điều chỉnh để cải thiện hiệu năng. Ngoài ra, ta cũng có thể thử nhiều mô hình khác nhau như cây quyết định, rừng ngẫu nhiên hoặc mạng nơ-ron để tìm ra mô hình phù hợp nhất.

7. Triển khai hệ thống

   Khi mô hình đạt chất lượng mong muốn, hệ thống có thể được triển khai trong ứng dụng thực tế. Ví dụ, một trang web bất động sản có thể sử dụng mô hình để ước lượng giá của một căn nhà khi người dùng nhập các thông tin như diện tích, số phòng và vị trí của căn nhà đó.