

**ĐẠI HỌC QUỐC GIA TP HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC BÁCH KHOA**

Quản Thành Thơ (Chủ biên)
Lê Đình Thuận, Bùi Thị Huyền Trang
Lê Thị Xinh, Nguyễn Quang Đức

CÁC HỆ THỐNG THÔNG MINH

NHÀ XUẤT BẢN ĐẠI HỌC QUỐC GIA
TP. HỒ CHÍ MINH - 2025

To laoshi, Bao Bao and Dong Dong

Mục lục

Phần 1 Các kỹ thuật học máy phổ biến	19
Chương 1: Học có giám sát và không giám sát	21
1.1 Học có giám sát	21
1.1.1 Khái niệm	21
1.1.2 Nguyên lý hoạt động	21
1.1.3 Phân loại	22
1.1.3.1 Phân loại theo mục tiêu dự đoán	22
1.1.3.2 Phân loại theo mô hình	22
1.1.4 Sự khác biệt giữa các mô hình	23
1.1.4.1 Hàm măt măt	23
1.1.4.2 Hàm giá trị	24
1.1.4.3 Hàm hợp lý	24
1.1.5 Một số giải thuật học có giám sát	25
1.1.6 Ứng dụng	25
1.2 Học không giám sát	26
1.2.1 Đặt vấn đề	26
1.2.2 Khái niệm	27
1.2.3 Nguyên lý hoạt động	27
1.2.4 Phân loại	28
1.2.5 Một số giải thuật học không giám sát	28
1.2.6 Ứng dụng	29
Chương 2: Thuật toán k-NN	31
2.1 Bài toán mở đầu	31
2.2 Khái niệm	31
2.3 Khoảng cách giữa các điểm dữ liệu	32
2.3.1 Khoảng cách Manhattan	32
2.3.2 Khoảng cách Euclid	34
2.3.3 Khoảng cách Hamming	35
2.3.4 Khoảng cách Minkowski	36
2.3.5 Mối quan hệ với Định chuẩn L_p	38
2.3.6 Khoảng cách cosine	39
2.3.7 Khoảng cách góc	40
2.4 Thuật toán	41
2.4.1 Các bước tiến hành thuật toán	41
2.4.2 Lựa chọn tham số k	43
2.4.3 Trọng số trong k-NN	44
2.5 Ưu điểm và nhược điểm của k-NN	48

2.5.1	Ưu điểm	48
2.5.2	Nhược điểm	48
2.6	Bài tập	49
Chương 3:	Cây quyết định và Rừng ngẫu nhiên	51
3.1	Cây quyết định	51
3.1.1	Bài toán mở đầu	51
3.1.2	Khái niệm	53
3.1.3	Giải thuật ID3	54
3.1.3.1	Ý tưởng	54
3.1.3.2	Hàm Entropy và Information Gain	55
3.1.4	Điều kiện dừng	56
3.1.5	Giải thuật	57
3.2	Rừng ngẫu nhiên	63
3.2.1	Bài toán mở đầu	63
3.2.2	Khái niệm	64
3.2.3	Nhược điểm của cây quyết định và sự cải tiến của Rừng ngẫu nhiên	65
3.2.4	Giải thuật	66
3.3	Bài tập	66
Chương 4:	Thuật toán Naïve Bayes	71
4.1	Bài toán mở đầu	71
4.2	Khái niệm	72
4.3	Các định lý xác suất	72
4.4	Nguyên lý phân loại dựa vào xác suất	73
4.5	Xác suất 0 - Zero Probability	77
4.5.1	Nguyên nhân của xác suất bằng 0	78
4.5.2	Phương pháp làm mượt Laplace	78
4.5.3	Ý nghĩa	79
4.6	Dánh giá mô hình	79
4.7	Ma trận confusion	80
4.7.1	Khái niệm	80
4.7.2	Các chỉ số hiệu suất	81
4.7.3	Vấn đề mất cân bằng lớp	82
4.8	Ưu và nhược điểm của Naïve Bayes	83
4.8.1	Ưu điểm	83
4.8.2	Nhược điểm	84
4.8.3	Tại sao Naïve Bayes phù hợp với lọc thư rác?	84
4.9	Bài tập	85
Chương 5:	Thuật toán Support Vector Machine	87
5.1	Bài toán mở đầu	87
5.2	Khái niệm	88
5.3	Ý tưởng tìm siêu phẳng phân chia	89
5.4	Trực quan về khoảng cách biên	92
5.5	Thuật toán	93
5.5.1	Kí hiệu	93
5.5.2	Hàm khoảng cách	93
5.5.3	Khoảng cách hình học	93
5.6	Bài toán tối ưu trong SVM	95
5.6.1	Mô hình SVM cơ bản	95

5.6.2	Mục tiêu tối ưu	95
5.6.3	Bài toán tối ưu hóa trong SVM	96
5.6.4	Ý nghĩa của ràng buộc	96
5.6.5	Chuyển sang dạng đối ngẫu	97
5.7	Bài toán đối ngẫu Lagrange	98
5.7.1	Phương pháp nhân tử Lagrange với ràng buộc đẳng thức	98
5.7.2	Bài toán đối ngẫu với ràng buộc bất đẳng thức	99
5.7.3	Áp dụng vào SVM	100
5.7.4	Lợi ích và ứng dụng trong SVM	102
5.8	Hàm kernel trong SVM	102
5.8.1	Khái niệm hàm kernel và ánh xạ đặc trưng	102
5.8.2	Đa thức kernel	104
5.8.3	Gaussian kernel	104
5.8.4	Tính hợp lệ của hàm kernel	105
5.9	Ưu điểm	107
5.10	Bài tập	108
Chương 6: Giải thuật k-means		111
6.1	Bài toán mở đầu	111
6.2	Khái niệm	112
6.3	Giải thuật	112
6.3.1	Hàm mắt mèo	113
6.3.2	Giải thuật tối ưu	114
6.4	Một số phương pháp hỗ trợ k-means	117
6.4.1	Phương pháp khuỷu tay	117
6.4.2	Điểm số Silhouette	118
6.4.3	So sánh và ứng dụng	119
6.5	Hạn chế	119
6.6	Bài tập	120
Chương 7: Giải thuật di truyền		123
7.1	Bài toán mở đầu	123
7.2	Khái quát về di truyền và thuyết tiến hóa của Darwin	124
7.2.1	Di truyền và quy luật Mendel	124
7.2.2	Thuyết tiến hóa của Darwin	125
7.3	Giải thuật di truyền	125
7.3.1	Quy trình cơ bản của giải thuật di truyền	126
7.3.2	Khởi tạo quần thể ban đầu	128
7.3.3	Hàm mục tiêu	129
7.3.4	Các toán tử di truyền	130
7.3.4.1	Toán tử chọn lọc	130
7.3.4.2	Toán tử lai ghép	130
7.3.4.3	Toán tử đột biến	131
7.4	Ứng dụng của Giải thuật di truyền	131
7.4.1	Bài toán One-max	131
7.4.1.1	Giới thiệu bài toán One-max	131
7.4.1.2	Sử dụng giải thuật di truyền để giải bài toán One-max	131
7.4.2	Bài toán sắp xếp thời khoá biểu	136
7.4.2.1	Giới thiệu bài toán xếp thời khoá biểu	136
7.4.2.2	Sử dụng giải thuật di truyền để giải bài toán xếp thời khoá biểu	137

7.4.3	Một số bài toán khác	141
7.4.3.1	Bài toán người du lịch	141
7.4.3.2	Bài toán lập lịch	142
7.4.3.3	Phân hoạch đối tượng và đồ thị	142
7.4.3.4	Vạch đường cho robot di chuyển	142
7.5	Bài tập	143
Phần 2	Ứng dụng minh họa	147
Chương 8:	Hệ thống Quản lý Phản hồi Khách hàng Thông minh	149
8.1	Dề bài	149
8.2	Tài liệu phân tích yêu cầu	149
8.2.1	Giới thiệu	149
8.2.1.1	Mục đích	149
8.2.1.2	Phạm vi	150
8.2.1.3	Định nghĩa và viết tắt	150
8.2.2	Mô tả tổng quan	150
8.2.2.1	Phân quyền người dùng	150
8.2.2.2	Lược đồ Use-case	151
8.2.2.3	Yêu cầu chức năng của hệ thống	151
8.2.2.4	Yêu cầu phi chức năng của hệ thống	151
8.2.2.5	Kịch bản Use Case	153
8.2.2.6	Lược đồ trạng thái của hệ thống	166
8.3	Tài liệu thiết kế hệ thống	168
8.3.1	Mục đích	168
8.3.2	Kiến trúc hệ thống	168
Phần 3	Một số kỹ thuật học máy khác	171
Chương 9:	Linear Regression	173
9.1	Cơ sở lý thuyết	173
9.1.1	Giới thiệu về Linear Regression	173
9.1.2	Dạng của Linear Regression	173
9.1.3	Sai số dự đoán	174
9.1.4	Hàm mất mát	174
9.1.5	Nghiệm của bài toán Linear Regression	175
9.2	Chuẩn bị dữ liệu để thực hiện hồi quy tuyến tính	175
9.3	Tranning model sử dụng Linear Regression với dữ liệu thực tế	177
9.3.1	Mẫu dữ liệu sử dụng	177
9.3.2	Ví dụ về chỉ số BMI	178
9.3.3	Huấn luyện mô hình	179
9.3.4	Huấn luyện mô hình với Sklearn	179
9.4	Ứng dụng cho điểm và đưa ra lời khuyên sức khỏe	180
9.4.1	Ý tưởng	180
9.4.2	Mô tả ứng dụng	181
9.4.3	Kiến trúc phần mềm	181
9.4.4	Sequence diagram	181
9.5	Bài tập	181
Chương 10:	Logistic Regression	185
10.1	Bài toán mở đầu	185
10.2	Hàm sigmoid	186
10.3	Thiết lập bài toán	187

10.3.1 Mô hình	187
10.3.2 Hàm mất mát	188
10.3.3 Quy tắc dây chuyền (chain rule)	189
10.3.4 Gradient descent	189
10.3.5 Quan hệ giữa xác suất và phương trình tuyến tính	191
10.4 Đánh giá hiệu suất	191
10.5 Ví dụ	192
10.5.1 Mô tả	192
10.5.2 Thiết lập bài toán	193
10.5.3 Hiện thực bài toán bằng ngôn ngữ lập trình Python	194
10.5.4 Kiểm tra kết quả	196
10.6 Ứng dụng	196
10.6.1 Ứng dụng hồi quy Logistic vào dự án phần mềm	197
10.6.2 Mô tả	197
10.6.3 Phân tích yêu cầu (requirement) của ứng dụng	197
10.6.4 Thiết kế mô hình phân loại nội dung	197
10.6.5 Kiến trúc phần mềm	199
10.6.6 Sequence diagram	200
10.7 Bài tập	200
Chương 11: Giải thuật DBSCAN	203
11.1 Bài toán đặt ra	203
11.2 Sơ lược về phân cụm dữ liệu	203
11.2.1 Khái niệm phân cụm dữ liệu	203
11.2.2 Các giải thuật gom cụm	205
11.3 Giải thuật DBSCAN	205
11.3.1 Khái niệm	205
11.3.2 Định nghĩa sử dụng trong giải thuật DBSCAN	206
11.3.3 Mã giả giải thuật DBSCAN	207
11.3.4 Xác định thông số Eps and MinPts	208
11.3.5 Độ phức tạp	209
11.3.6 Bài toán ví dụ về Thuật Toán DBSCAN	209
11.3.7 Ưu điểm và hạn chế	210
11.3.7.1 Ưu điểm	210
11.3.7.2 Hạn chế	211
11.4 Ứng dụng vào dự án thực tế	211
11.4.1 Mô tả	211
11.4.2 Thiết kế	212
11.4.3 Kiến trúc phần mềm	214
11.4.4 Sequence Diagram	214
11.5 Bài tập	215
Chương 12: Giải thuật Association Rule Mining	217
12.1 Đặt ra vấn đề	217
12.1.1 Luật kết hợp	217
12.2 Các tính chất của một luật kết hợp	218
12.2.1 Độ hỗ trợ (support)	218
12.2.2 Độ tin cậy (confidence)	218
12.2.3 Tính hợp lệ của luật kết hợp	218
12.2.4 Các mục tiêu cần đạt trong việc khai thác luật kết hợp	219

12.3	Thuật toán	219
12.3.1	Thuật toán Apriori	219
12.4	Hiện thực	219
12.4.1	Giới thiệu phần hiện thực: Phiên bản C# Console	219
12.4.2	Hiện thực giải thuật: Phiên bản C# Console	220
12.4.2.1	Class ItemSet	220
12.4.2.2	Class AssociationRule	220
12.4.2.3	Class Service	220
12.4.2.4	Class Apriori	221
12.4.2.5	Class Program: Class chứa hàm Main	227
12.4.3	Kết quả	228
12.5	Ứng dụng vào dự án phần mềm	230
12.6	Bài tập	233
Chương 13:	Matrix Factorization for Collaborative filtering	235
13.1	Tổng quan về Recommendation System	235
13.1.1	Giới thiệu	235
13.1.2	Utility matrix	236
13.2	Content-based System	236
13.2.1	Giới thiệu	236
13.2.2	Xây dựng hàm mất mát	237
13.2.3	Ví dụ về hàm mất mát cho User	238
13.3	Content-based System	238
13.3.1	Giới thiệu	238
13.3.2	Xây dựng hàm mất mát	239
13.3.3	Ví dụ về hàm mất mát cho User	239
13.4	Matrix Factorization for Collaborative filtering	240
13.4.1	Giới thiệu	240
13.4.2	Xây dựng và tối ưu hàm mất mát	241
13.4.2.1	Hàm mất mát	241
13.4.2.2	Tối ưu hàm mất mát	241
13.5	Áp dụng MFCF vào hệ thống khuyến nghị phim ảnh trên Website xem phim trực tuyến	242
13.5.1	Dataset cho quá trình huấn luyện	242
13.5.2	Kiến trúc của hệ thống khuyến nghị	243
13.5.3	Sequence Diagram quá trình hoạt động của hệ thống khuyến nghị	243
13.6	Bài tập	244
Chương 14:	Giải thuật ARIMA	245
14.1	Giới thiệu	245
14.1.1	Bài toán mở đầu	245
14.1.2	Giới thiệu về chuỗi thời gian	245
14.1.3	Giới thiệu về autocorrelation function	246
14.1.4	ARIMA	247
14.2	Lý thuyết mô hình ARIMA	247
14.2.1	Giải quyết tính dừng	248
14.2.2	Auto Regressive và Moving Average	248
14.3	Xây dựng mô hình ARIMA trong Python	249
14.3.1	Tìm bậc của sai phân	249
14.3.2	Tìm bậc của tham số MA (q)	250
14.3.3	Tìm bậc của tham số AR (p)	251

14.3.4 Xây dựng mô hình ARIMA	251
14.3.5 Phương pháp Auto ARIMA	253
14.3.6 Dự báo	255
14.4 Ứng dụng ARIMA vào dự án phần mềm	255
14.5 Bài tập	256
Chương 15: Topic Modeling	259
15.1 Mạng thần kinh tích chập (Convolutional neural network)	259
15.1.1 Giới thiệu	259
15.1.2 Lớp tích chập	260
15.1.2.1 Ma trận tích chập	261
15.1.2.2 Stride	262
15.1.2.3 Padding	262
15.1.3 Lớp phi tuyến tính	264
15.1.4 Lớp tổng hợp	265
15.1.5 Lớp kết nối đầy đủ	266
15.2 Ứng dụng mạng thần kinh tích chập vào thực tế	267
15.2.1 Đặt vấn đề	267
15.2.2 Phân tích yêu cầu	267
15.2.2.1 Use case diagram của hệ thống	267
15.2.2.2 Đặc tả use case	268
15.2.2.3 Yêu cầu phi chức năng của hệ thống	272
15.3 Thiết kế hệ thống	273
15.3.1 Xây dựng mạng thần kinh tích chập	273
15.3.2 Activity diagram của hệ thống	273
15.3.3 Sequence diagram của hệ thống	276
15.3.4 Class diagram của hệ thống	277
15.4 Demo ứng dụng	277

Danh sách hình vẽ

1.1	Minh họa học có giám sát	21
1.2	Minh họa về học không giám sát	27
2.1	Khoảng cách Manhattan trong không gian hai chiều.	33
2.2	Khoảng cách Euclid giữa hai điểm trong không gian hai chiều.	35
2.3	Mô phỏng khoảng cách Hamming giữa hai chuỗi nhị phân 010 và 101.	36
2.4	Minh họa khoảng cách cosine giữa hai vector.	40
2.5	Ảnh hưởng của phân phối nhãn khi chọn tham số k trong k-NN	43
3.1	Cây quyết định theo thuộc tính <i>Thời tiết</i>	60
3.2	Cây quyết định sau khi chia theo <i>Thời tiết</i> = Tốt	61
3.3	Cây quyết định hoàn chỉnh	62
3.4	Sơ đồ quy trình chọn bộ phim hay nhất dựa trên ý kiến tập thể	63
5.1	Minh họa ba đường thẳng phân loại các điểm dữ liệu.	89
5.2	Minh họa các siêu phẳng phân chia với khoảng cách biên khác nhau.	90
5.3	Minh họa việc xác định khoảng cách biên và vector hỗ trợ.	90
5.4	Minh họa các cặp dữ liệu huấn luyện và ranh giới quyết định trong không gian hai chiều.	91
5.5	Minh họa khoảng cách hình học từ điểm dữ liệu đến siêu phẳng.	94
5.6	Minh họa ranh giới quyết định của siêu phẳng trong không gian hai chiều.	95
5.7	Tập dữ liệu không thể phân tách tuyến tính	103
5.8	Tập dữ liệu sau khi ánh xạ, trở thành phân tách tuyến tính	103
5.9	Xác định đường biên tốt nhất bằng trực quan	109
5.10	Biểu diễn tập dữ liệu không phân tách tuyến tính	109
7.1	Minh họa cấu trúc của nhiễm sắc thể, <i>gen</i> , và quần thể trong giải thuật di truyền.	126
7.2	Lưu đồ quy trình giải thuật di truyền	127
7.3	Khởi tạo quần thể	132
7.4	Dánh giá độ thích nghi cá thể	132
7.5	Ví dụ về chọn lọc	133
7.6	Quần thể mới từ sự chọn lọc	133
7.7	Lai ghép hai cá thể	134
7.8	Cá thể đột biến	134
7.9	Lịch học ở dạng chuỗi số nhị phân	138
7.10	Hình minh họa bước xác suất lai ghép lịch học giữa 2 sinh viên bạn B và bạn D	140
7.11	Hình minh họa bước lai ghép lịch học giữa 2 sinh viên bạn B và bạn D	141

DANH SÁCH HÌNH VẼ

8.1 Lược đồ use case tổng quát	152
8.2 Lược đồ trạng thái của bảng khảo sát	167
8.3 Lược đồ trạng thái của một dự án học máy	169
8.4 Kiến trúc hệ thống	170
9.1 Sự thay đổi của hàm hồi quy khi có các giá trị nhiễu	176
9.2 Đồ thị tương quan giữa chỉ số BMI và tuổi thọ	178
9.3 Kiến trúc phần mềm	182
9.4 Sequence Diagram khi train Sample data	182
9.5 Sequence Diagram khi người dùng sử dụng	183
10.1 Kết quả thi ứng với điểm trung bình (trên thang 100) các môn Văn và Toán ở năm học cuối cấp	185
10.2 Đồ thị giữa điểm Toán, Văn và kết quả thi	186
10.3 Phác thảo đồ thị giá trị của J sau mỗi lần thực hiện gradient descent	195
10.4 Đường phân cách các điểm dữ liệu đậu và rớt	196
10.5 Tập dữ liệu huấn luyện	198
10.6 Dữ liệu sau khi chuẩn hóa	199
10.7 Kiến trúc chương trình phân loại thư điện tử	200
10.8 Sequence diagram cho quá trình phân loại thư điện tử	201
11.1 Minh họa phân cụm dữ liệu.	204
11.2 Quá trình phân cụm dữ liệu.	204
11.3 Dạng cụm dữ liệu được khám phá bởi giải thuật DBSCAN.	206
11.4 Đồ thị sorted 4-dist	208
11.5 Dữ liệu GPS của hệ thống xe bus tại thành phố Chicago, Mỹ	212
11.6 Kết quả phân vùng những điểm ùn tắc giao thông	212
11.7 Minh họa trực quan những điểm ùn tắc giao thông trong thành phố, những điểm màu đen là điểm nhiễu. Người dùng có thể di chuyển, phóng to, thu nhỏ để xem chi tiết	213
11.8 Kiến trúc của chương trình	214
11.9 Sequence Diagram khi người dùng coi tra cứu điểm ùn tắc giao thông	214
11.10 Xác định trực quan	215
12.1 Tập dữ liệu.	228
12.2 Kết quả cho tập L1.	229
12.3 Kết quả cho tập L2.	229
12.4 Kết quả cho tập L3.	230
12.5 Tập dữ liệu.	230
12.6 Giao diện ban đầu của ứng dụng	231
12.7 Giải thuật với Support = 2	231
12.8 Kiến trúc phần mềm	232
12.9 Sequence Diagram chức năng hiển thị các luật tương ứng khi chạy giải thuật Association Rule Mining với tập dữ liệu có sẵn	232
13.1 Bài toán Recommendation tương đương với việc hoàn thiện ma trận Utility	236
13.2 Kiến trúc của hệ thống khuyến nghị trong một ứng dụng xem phim trực tuyến	243
13.3 Sequence Diagram quá trình hoạt động của hệ thống khuyến nghị	243
14.1 Một chuỗi thời gian (time series) về doanh số của một cửa hàng trong 36 tháng	246

DANH SÁCH HÌNH VẼ

14.2 Đồ thị của dữ liệu sau khi lấy sai phân bậc 1	250
14.3 Đồ thị của dữ liệu sai phân bậc 1 (trái) và đồ thị ACF tương ứng (phải)	251
14.4 Đồ thị của dữ liệu sai phân bậc 1 (trái) và đồ thị PACF tương ứng (phải)	252
14.5 Đồ thị doanh số bán hàng và kết quả dự đoán của 6 tháng tiếp theo	255
14.6 Kiến trúc phần mềm của hệ thống SFCS tích hợp ARIMA	257
14.7 Biểu đồ tuần tự của chức năng dự đoán doanh số bán hàng	258
15.1 Các lớp trong mạng CNN trích xuất các đặc tính từ dữ liệu đầu vào	260
15.2 Dữ liệu đầu vào 3 chiều	261
15.3 Sử dụng ma trận tích chập để kết nối các neuron	262
15.4 Các hiệu quả khác nhau của ma trận tích chập	263
15.5 Sử dụng stride=1 với ma trận tích chập	263
15.6 Kỹ thuật zero padding	264
15.7 Các hàm phi tuyến tính phổ biến	265
15.8 Các lớp tổng hợp phổ biến	266
15.9 Lớp kết nối đầy đủ	266
15.10 Use case diagram của hệ thống	268
15.11 Sự thay đổi của hàm chi phí trong quá trình huấn luyện	274
15.12 Sự thay đổi của độ chính xác trong quá trình huấn luyện	274
15.13 Activity diagram của hệ thống	275
15.14 Sequence diagram của hệ thống	276
15.15 Class diagram của hệ thống	277
15.16 Chức năng chụp (chọn) hình ảnh	278
15.17 Chức năng dự đoán từ hình ảnh đầu vào	280
15.18 Chức năng tìm kiếm thông tin	281

Danh sách bảng

1.1	So sánh mô hình phân biệt (<i>discriminative model</i>) và mô hình sinh (<i>generative model</i>)	23
1.2	Một số hàm mất mát phổ biến	24
2.1	Dữ liệu khách hàng: <i>chiều cao (height)</i> , <i>cân nặng (weight)</i> , và <i>kích cỡ áo (shirt size)</i>	45
2.2	Tập dữ liệu huấn luyện sau khi chuẩn hóa (<i>normalization</i>).	46
2.3	Khoảng cách từ $P_{\text{norm}} = (0.3684, 0.6522)$ đến các khách hàng trong tập dữ liệu chuẩn hóa.	47
2.4	Dữ liệu khách hàng	50
2.5	Bộ dữ liệu điểm bất thường với ba thuộc tính số.	50
3.1	Dữ liệu sở thích	51
3.2	Dữ liệu huấn luyện về việc đi đánh cầu	58
3.3	Tập dữ liệu ứng với <i>Thời tiết = Tốt</i>	60
3.4	Tập dữ liệu ứng với <i>Nhiệt độ = Thấp</i>	61
3.5	Tập dữ liệu ứng với <i>Nhiệt độ = Cao</i>	61
3.6	Bảng dữ liệu khách hàng với các đặc trưng và quyết định mua hàng.	66
3.7	Bảng dữ liệu sản phẩm với các đặc trưng và phân loại chất lượng.	67
3.8	Dữ liệu phê duyệt khoản vay với các thuộc tính đầu vào và kết quả.	68
4.1	Dữ liệu huấn luyện về bệnh tiểu đường	75
4.2	Ma trận confusion cho bài toán phân loại nhị phân	80
4.3	Ma trận confusion cho bài toán phân loại bệnh tiểu đường	81
4.4	Tập dữ liệu bệnh nhân tiểu đường.	85
6.1	Khoảng cách Euclidean đến các tâm cụm ban đầu	116
6.2	Khoảng cách Euclidean đến các tâm cụm mới	116
6.3	Khoảng cách Euclidean đến các tâm cụm mới	117
6.4	So sánh giữa phương pháp khuỷu tay và điểm số Silhouette	119
6.5	Tọa độ các điểm dữ liệu	121
7.1	Dữ liệu thời gian mong muốn của các sinh viên	138
7.2	Lịch học của các sinh viên	139
7.3	Bảng sau khi lai tạo	141
7.4	Thế hệ cha	144
7.5	Bảng dữ liệu Code và Fitness cho thế hệ con	144
7.6	Bảng kết quả với các số ngẫu nhiên	145

DANH SÁCH BẢNG

8.1	Định nghĩa các thuật ngữ phổ biến	150
8.2	Kịch bản Thu thập khảo sát	154
8.3	Kịch bản Hiển thị danh sách khảo sát	154
8.4	Kịch bản Tạo khảo sát	155
8.5	Kịch bản Liệt kê các dự án học máy	157
8.6	Kịch bản Tạo dự án học máy	157
8.7	Kịch bản Chi tiết dự án học máy	158
8.8	Kịch bản Cấu hình Liên kết nguồn dữ liệu cho dự án học máy	159
8.9	Kịch bản Fetch nguồn dữ liệu cho hệ thống	161
8.10	Kịch bản Huấn luyện mô hình	162
8.11	Kịch bản Hiển thị kết quả huấn luyện	163
8.12	Kịch bản Đăng nhập	163
8.13	Kịch bản Admin dashboard	165
8.14	Kịch bản Liệt kê danh sách user	165
9.1	Bảng giá trị dữ liệu của hàm số $f(x)$	176
9.2	Kết quả sau khi train	179
9.3	Kết quả sau khi train với thư viện SKLearn	180
9.4	Bảng dữ liệu giá thành sản phẩm của Hùng	183
10.1	Ma trận phân loại	192
10.2	Bảng dữ liệu mẫu để thử tính toán công thức bằng tay	193
10.3	So sánh xác suất tính được với kết quả thực tế	196
10.4	Kết quả dự đoán và thực tế của tập dữ liệu sau khi áp dụng mô hình	200
12.1	Bảng cơ sở dữ liệu chứa các giao dịch	217
15.1	Bảng mô tả các chức năng	279

Phần 1

Các kỹ thuật học máy phổ biến

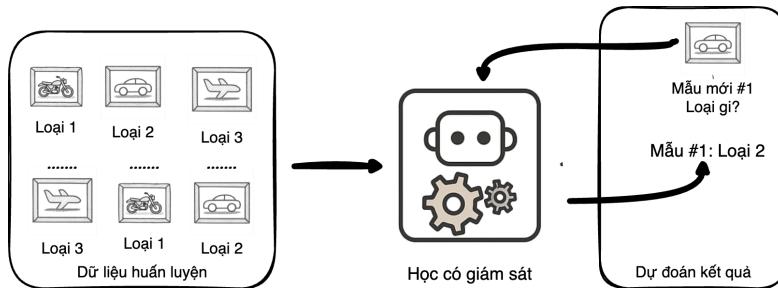
Chương 1

Học có giám sát và không giám sát

1.1 Học có giám sát

1.1.1 Khái niệm

Học có giám sát (*supervised learning*) [1] là một phương pháp cốt lõi trong lĩnh vực học máy (*machine learning*). Phương pháp này huấn luyện các thuật toán dựa trên tập dữ liệu chứa các cặp đầu vào (*input*) và nhãn (*label*), với mỗi cặp gồm một giá trị đầu vào x và nhãn đầu ra tương ứng y . Mục tiêu của học có giám sát là xây dựng mô hình có khả năng dự đoán nhãn cho dữ liệu mới dựa trên mối quan hệ được học từ dữ liệu huấn luyện, từ đó hỗ trợ giải quyết các bài toán thực tiễn trong nhiều lĩnh vực. Phương pháp học máy có giám sát được minh họa trong Hình 1.1.



Hình 1.1: Minh họa học có giám sát

1.1.2 Nguyên lý hoạt động

Học có giám sát hoạt động dựa trên tập dữ liệu huấn luyện gồm các cặp $(x_1, y_1), (x_2, y_2), \dots, (x_N, y_N)$, trong đó $x_1, x_2, \dots, x_N \in \mathcal{X}$ là các giá trị đầu vào và $y_1, y_2, \dots, y_N \in \mathcal{Y}$ là các nhãn tương ứng. Mục đích là tìm một hàm ánh xạ $g : \mathcal{X} \rightarrow \mathcal{Y}$, sao cho $g(x)$ dự đoán giá trị nhãn \hat{y} cho đầu vào $x \in \mathcal{X}$.

1.1. HỌC CÓ GIÁM SÁT

Quá trình huấn luyện trong học có giám sát nhằm mục tiêu tối ưu hoá hàm ánh xạ g cho kết quả của dự đoán $\hat{y} = g(x)$ gần nhất với nhãn thực tế y . Để đạt được điều này, các thuật toán sử dụng một hàm mất mát (*loss function*) để đo lường sai số giữa kết quả dự đoán và giá trị thực tế.

Tuỳ vào bản chất của bài toán, mô hình học giám sát được xây dựng để thực hiện nhiệm vụ hồi quy (*regression*) hoặc phân loại (*classification*), tương ứng với bài toán có miền \mathcal{Y} là liên tục hay rời rạc. Mục tiêu cuối cùng là xây dựng một mô hình có khả năng tổng quát hoá tốt - dự đoán chính xác nhãn cho các dữ liệu chưa từng gặp, đảm bảo hiệu quả khi áp dụng thực tế như nhận dạng mẫu, phân tích hành vi, dự báo xu hướng...

1.1.3 Phân loại

Trong học có giám sát, các thuật toán được phân loại dựa trên những tiêu chí khác nhau, phản ánh đặc điểm bài toán cũng như cấu trúc của mô hình. Hai tiêu chí phân loại phổ biến là: phân loại theo mục tiêu dự đoán và phân loại theo kiểu mô hình học.

1.1.3.1 Phân loại theo mục tiêu dự đoán

Các bài toán trong học có giám sát có thể được phân chia dựa trên kiểu đầu ra cần dự đoán, bao gồm hai nhóm chính: bài toán phân loại và bài toán hồi quy, mỗi nhóm có những đặc trưng và ứng dụng riêng biệt.

- **Bài toán phân loại** là bài toán dự đoán với một nhãn kết quả thuộc lớp rời rạc. Mỗi dữ liệu đầu vào được gán nhãn vào một lớp cụ thể, với mục tiêu xác định nhãn đầu ra dựa trên các lớp đã định nghĩa. Bài toán phân loại bao gồm phân loại nhị phân (hai lớp) và phân loại đa lớp (nhiều lớp). Ví dụ, ứng dụng phân loại thư rác hoặc thư thường thuộc bài toán phân loại nhị phân; ứng dụng đánh giá nguy cơ bệnh tật dựa trên dữ liệu lâm sàng, và nhận diện hình ảnh, như phân loại thành chó, mèo, người, hoặc xe thuộc bài toán phân loại đa lớp.
- **Bài toán hồi quy** là loại bài toán trong đó nhãn đầu ra cần dự đoán là một giá trị liên tục, trong miền số thực hoặc tập con của nó. Khác với bài toán phân loại - nơi mô hình gán mỗi đầu vào vào một lớp cụ thể - bài toán hồi quy không phân nhóm dữ liệu mà tập trung vào việc ước lượng chính xác một đại lượng định lượng. Ví dụ điển hình của bài toán hồi quy là dự đoán giá trị bất động sản dựa trên các đặc trưng như số phòng, diện tích, hoặc vị trí địa lý. Tương tự, thay vì phân loại một bệnh nhân là “mắc bệnh” hay “không mắc bệnh” như trong phân loại, hồi quy có thể được dùng để dự đoán chính xác chỉ số đường huyết hoặc huyết áp của bệnh nhân đó.

1.1.3.2 Phân loại theo mô hình

Các mô hình học có giám sát có thể được phân loại dựa trên cơ chế học và phương pháp ước lượng xác suất, bao gồm hai nhóm chính: mô hình phân biệt (*discriminative model*) và mô hình sinh (*generative model*). Mỗi nhóm đại diện cho một cách tiếp cận riêng trong việc xây dựng mô hình, với các ưu điểm và hạn chế nhất định, phù hợp cho những mục tiêu học máy khác nhau như phân loại, sinh dữ liệu, hoặc học trong điều kiện thông tin thiếu. Bảng 1.1 trình bày so sánh các yếu tố về mục tiêu, cách thức học, ứng dụng, ưu điểm và nhược điểm của hai mô hình này.

- **Mô hình phân biệt** là các mô hình tập trung vào việc ước lượng xác suất có điều kiện $P(y|x)$, tức là xác định xác suất nhãn y tương ứng với đầu vào x . Các mô hình này hướng

đến việc phân tách ranh giới giữa các lớp một cách trực tiếp mà không cần mô hình hóa phân phối dữ liệu đầu vào. Ưu điểm của các mô hình này thường đạt hiệu quả cao trong các bài toán phân loại, đặc biệt khi dữ liệu huấn luyện đầy đủ và rõ ràng, nhờ khả năng tối ưu hóa trực tiếp ranh giới quyết định. Ngược lại, Nhược điểm chính là khả năng tổng quát hóa bị hạn chế trong môi trường thiếu dữ liệu hoặc khi phân phối dữ liệu phức tạp.

- **Mô hình sinh** là các mô hình học phân phối xác suất $P(x|y)$ cho mỗi lớp và sử dụng Định lý Bayes [2] để suy ra $P(y|x)$. Các mô hình này không chỉ dùng để phân loại mà còn có khả năng sinh dữ liệu mới từ phân phối đã học. Ưu điểm của mô hình sinh là khả năng xử lý tốt khi dữ liệu không đầy đủ, hỗ trợ học trong điều kiện thiếu nhãn và phù hợp cho các bài toán tổng quát hóa hoặc sinh mẫu; tuy nhiên, chúng thường yêu cầu giả định rõ về phân phối dữ liệu và chi phí tính toán cao hơn, đồng thời hiệu suất có thể suy giảm nếu các giả định mô hình không phù hợp với dữ liệu thực tế.

Bảng 1.1: So sánh mô hình phân biệt (*discriminative model*) và mô hình sinh (*generative model*)

Yếu tố	Mô hình phân biệt	Mô hình sinh
Mục tiêu	Ước lượng $P(y x)$	Ước lượng $P(x y)$, suy ra $P(y x)$ qua Định lý Bayes
Cách thức học	Phân tách các lớp dữ liệu	Mô phỏng quá trình sinh dữ liệu
Ứng dụng	Tối ưu cho phân loại chính xác	Sinh dữ liệu, xử lý dữ liệu thiếu
Ưu điểm	Hiệu quả với dữ liệu đầy đủ	Tốt cho tổng quát hóa, sinh dữ liệu
Nhược điểm	Yêu cầu dữ liệu lớn	Phức tạp, tính toán nặng

1.1.4 Sự khác biệt giữa các mô hình

Mô hình phân biệt, như Hồi quy Logistic hay Support Vector Machine, tập trung vào việc ước lượng xác suất có điều kiện $P(y|x)$, tức là chúng chỉ quan tâm đến việc phân biệt giữa các lớp dữ liệu khác nhau. Ngược lại, mô hình sinh như Naive Bayes hay Mô hình hỗn hợp Gaussian, ước lượng phân phối xác suất của dữ liệu $P(x|y)$ và sử dụng các thông tin này để suy luận ra xác suất $P(y|x)$.

Điều này dẫn đến sự khác biệt trong cách các mô hình học từ dữ liệu. Mô hình phân biệt học trực tiếp biên quyết định giữa các lớp, trong khi mô hình sinh lại học về phân phối xác suất của dữ liệu. Do đó, mô hình phân biệt thường cho kết quả chính xác hơn trong các bài toán phân loại, nhưng mô hình sinh lại có khả năng mô phỏng dữ liệu và tạo ra dữ liệu giả lập, điều này có thể hữu ích trong một số ứng dụng như tạo ra dữ liệu mô phỏng hoặc phân tích dữ liệu chưa biết.

1.1.4.1 Hàm mất mát

Trong học có giám sát, quá trình huấn luyện mô hình thường được tiếp cận dưới góc nhìn của một bài toán tối ưu: tìm hàm ánh xạ sao cho mô hình đưa ra dự đoán gần nhất với nhãn thực tế. Để thực hiện điều này, cần một đại lượng định lượng mức độ sai lệch giữa đầu ra của mô hình và nhãn đúng — đó chính là *hàm mất mát* (loss function). Hàm mất mát, ký hiệu $L : (z, y) \in \mathbb{R} \times \mathcal{Y} \rightarrow \mathbb{R}$, là một hàm số ánh xạ từ giá trị dự đoán z và nhãn thực y sang một số thực biểu thị độ sai lệch giữa hai giá trị này.

Việc lựa chọn hàm mất mát có ảnh hưởng trực tiếp đến quá trình tối ưu và kết quả học. Tùy theo tính chất của bài toán (phân loại, hồi quy, nhị phân, đa lớp...) cũng như đặc điểm của mô

1.1. HỌC CÓ GIÁM SÁT

hình và dữ liệu, các hàm mất mát khác nhau sẽ thể hiện hiệu quả khác nhau trong việc dẫn dắt mô hình học đúng mục tiêu. Chẳng hạn, trong bài toán hồi quy, việc sử dụng sai số bình phương giúp nhấn mạnh các lỗi lớn, trong khi ở bài toán phân loại, hàm mất mát như cross-entropy phản ánh tốt xác suất dự đoán sai lầm. Do đó, sự đa dạng của các hàm mất mát không chỉ là kết quả của nhu cầu ứng dụng, mà còn bắt nguồn từ lý thuyết tối ưu hóa — nơi hàm mục tiêu cần phản ánh chính xác đặc trưng và mục đích học của mô hình.

Một số hàm mất mát phổ biến thường được sử dụng trong học máy được trình bày trong Bảng 1.2.

Bảng 1.2: Một số hàm mất mát phổ biến

Tên hàm mất mát	Biểu thức
Least Squared Error (Lỗi bình phương)	$\frac{1}{2}(y - z)^2$
Logistic Loss (Mất mát logistic)	$\ln(1 + e^{-yz})$
Hinge Loss (Mất mát biên)	$\max(0, 1 - yz)$
Cross-Entropy Loss (Mất mát entropy chéo)	$-[y \ln z + (1 - y) \ln(1 - z)]$

1.1.4.2 Hàm giá trị

Hàm giá trị (*cost function*), ký hiệu $J(\theta)$, là đại lượng tổng hợp toàn bộ sai số dự đoán của mô hình trên tập dữ liệu huấn luyện. Cụ thể, hàm giá trị được định nghĩa dựa trên hàm mất mát $L(\cdot, \cdot)$ áp dụng cho từng cặp dữ liệu:

$$J(\theta) = \sum_{i=1}^m L\left(h_\theta(x^{(i)}), y^{(i)}\right) \quad (1.1)$$

Trong đó, $h_\theta(x^{(i)})$ là đầu ra dự đoán của mô hình với tham số θ ứng với đầu vào $x^{(i)}$, và $y^{(i)}$ là nhãn thực tương ứng. Mục tiêu chính của quá trình huấn luyện là tìm tham số θ tối ưu sao cho giá trị $J(\theta)$ là nhỏ nhất, tức là tổng sai số trên toàn bộ tập dữ liệu được giảm thiểu. Việc tối ưu hóa hàm giá trị là trung tâm của hầu hết các thuật toán học máy, và là cơ sở cho các phương pháp như gradient descent.

1.1.4.3 Hàm hợp lý

Hàm hợp lý (*likelihood function*), ký hiệu $L(\theta)$, là một công cụ quan trọng trong việc ước lượng tham số của mô hình học máy. Hàm này biểu diễn xác suất mà mô hình với tham số θ tạo ra được tập dữ liệu huấn luyện đã quan sát. Trong quá trình huấn luyện, việc tối đa hóa hàm hợp lý cho phép tìm ra giá trị tham số tối ưu, tức là tham số giúp mô hình khớp tốt nhất với dữ liệu:

$$\theta^{\text{opt}} = \arg \max_{\theta} L(\theta) \quad (1.2)$$

Trên thực tế, để đơn giản hóa việc tính toán và tăng độ ổn định số, người ta thường sử dụng logarit của hàm hợp lý, gọi là hàm hợp lý-log (log-likelihood), ký hiệu $\ell(\theta) = \log L(\theta)$. Việc tối đa hóa hàm hợp lý-log là hoàn toàn tương đương với tối đa hóa hàm hợp lý, nhưng thuận tiện hơn trong thực hành, đặc biệt khi xử lý dữ liệu độc lập dẫn đến biểu thức tổng đơn giản hơn so với tích xác suất. Cách tiếp cận này là cơ sở cho nhiều thuật toán học máy dựa trên mô hình xác suất, chẳng hạn như hồi quy logistic, Naive Bayes và các mô hình đồ thị xác suất.

1.1.5 Một số giải thuật học có giám sát

Các thuật toán học có giám sát là những công cụ cốt lõi trong lĩnh vực học máy, được thiết kế để giải quyết hiệu quả các bài toán phân loại và hồi quy. Nhờ vào khả năng học từ dữ liệu có gán nhãn, các thuật toán này không chỉ đạt được độ chính xác cao mà còn thể hiện tính linh hoạt trong nhiều ứng dụng thực tiễn khác nhau. Dưới đây là một số phương pháp tiêu biểu thường được sử dụng trong các hệ thống học có giám sát hiện đại.

- **k-Nearest Neighbors (k-NN):** Là một thuật toán phân loại đơn giản nhưng hiệu quả, k-NN hoạt động dựa trên nguyên lý "bỏ phiếu đa số" từ k điểm lân cận gần nhất trong không gian đặc trưng. Mỗi điểm dữ liệu mới được gán nhãn dựa trên nhãn phổ biến nhất trong số các điểm lân cận này. Thuật toán không cần giai đoạn huấn luyện rõ ràng và thường được ứng dụng trong các bài toán nhận diện mẫu, phân tích hình ảnh, và phân nhóm hành vi khách hàng.
- **Cây quyết định (Decision Tree) và Rừng ngẫu nhiên (Random Forest):** Cây quyết định xây dựng một mô hình phân nhánh tuân tự bằng cách chia không gian đặc trưng dựa trên giá trị của các thuộc tính đầu vào. Đây là phương pháp trực quan, có thể sử dụng cho cả phân loại và hồi quy. Rừng ngẫu nhiên là một kỹ thuật tổ hợp gồm nhiều cây quyết định huấn luyện độc lập, sau đó tổng hợp kết quả để cải thiện độ chính xác và khả năng tổng quát hóa, đồng thời giảm hiện tượng quá khớp (*overfitting*). Hai mô hình này được ứng dụng rộng rãi trong phân tích tài chính, y học dự đoán, và hệ thống khuyến nghị.
- **Naïve Bayes:** Là một mô hình xác suất dựa trên Định lý Bayes, với giả định mạnh mẽ rằng các đặc trưng đầu vào độc lập có điều kiện cho nhãn. Dù giả định này không hoàn toàn chính xác trong thực tế, Naïve Bayes vẫn mang lại hiệu quả cao trong các bài toán phân loại văn bản, phát hiện thư rác, và phân tích cảm xúc, nhờ tính đơn giản, tốc độ xử lý nhanh và khả năng mở rộng tốt cho dữ liệu lớn.
- **Support Vector Machine (SVM):** là một mô hình học có giám sát dùng để phân loại và hồi quy, hoạt động bằng cách tìm một siêu phẳng tối ưu phân tách các lớp với biên lớn nhất. Với các bài toán không tuyến tính, SVM có thể kết hợp với các hàm nhân (*kernel*) để ánh xạ dữ liệu vào không gian đặc trưng cao hơn, nơi dữ liệu có thể phân tách tuyến tính. Mô hình này được đánh giá cao trong các bài toán phân loại có độ chính xác cao, như nhận diện chữ viết tay, phân loại ảnh hoặc phân tích gene.
- **Mạng nơ-ron nhân tạo (Artificial Neural Networks) (ANN):** Mạng nơ-ron là một mô hình tính toán lấy cảm hứng từ cấu trúc và cơ chế hoạt động của não bộ sinh học. Thông qua các lớp kết nối gồm nhiều nút (nơ-ron), mô hình học được các biểu diễn đặc trưng phức tạp từ dữ liệu. Mạng nơ-ron hiện đại có khả năng xử lý các bài toán phức tạp trong nhận diện hình ảnh, nhận dạng tiếng nói và xử lý ngôn ngữ tự nhiên, đóng vai trò then chốt trong nhiều hệ thống trí tuệ nhân tạo hiện nay.

1.1.6 Ứng dụng

Nhờ khả năng tổng quát hóa và dự đoán chính xác cho các trường hợp mới, các thuật toán học có giám sát được ứng dụng rộng rãi trong nhiều lĩnh vực. Một số ví dụ tiêu biểu, nhưng không hạn chế như:

1.2. HỌC KHÔNG GIÁM SÁT

- Trong lĩnh vực an ninh thông tin, một trong những ứng dụng phổ biến nhất của học có giám sát là trong hệ thống lọc thư rác. Các thuật toán như *Naive Bayes* hoặc *Support Vector Machine* được huấn luyện trên tập dữ liệu thư điện tử đã được gán nhãn là “thư rác” hoặc “hợp lệ”. Mô hình học cách nhận diện các đặc trưng quan trọng như từ khóa, tần suất xuất hiện, tiêu đề hoặc địa chỉ người gửi. Từ đó, mô hình có thể phân loại chính xác các thư điện tử mới. Việc này giúp tăng hiệu quả bảo mật và giảm thời gian xử lý thông tin rác cho cả người dùng, giảm khói lượng phải lưu trữ trên hệ thống máy chủ.
- Trong lĩnh vực y tế, học có giám sát được ứng dụng mạnh mẽ trong hỗ trợ chẩn đoán bệnh và ra quyết định điều trị. Ví dụ, mô hình được huấn luyện trên dữ liệu hồ sơ bệnh án đã gán nhãn (như tình trạng mắc bệnh, mức độ nguy cơ, kết quả xét nghiệm) để dự đoán khả năng mắc các bệnh như tim mạch, ung thư hoặc tiểu đường. Các thuật toán như Hồi quy Logistic, Rừng ngẫu nhiên hoặc mạng nơ-ron được sử dụng để phân loại bệnh nhân theo các mức độ nguy cơ (thấp, trung bình, cao), từ đó hỗ trợ bác sĩ đưa ra quyết định điều trị sớm và hiệu quả hơn.
- Trong lĩnh vực xử lý ngôn ngữ tự nhiên (*Natural Language Processing*), học có giám sát được sử dụng rộng rãi trong các tác vụ như phân loại văn bản (ví dụ: tin tức thể thao, kinh tế, giải trí) hoặc phân tích cảm xúc (tích cực, tiêu cực, trung lập). Tập dữ liệu huấn luyện bao gồm các văn bản đã được gán nhãn rõ ràng, ví dụ mỗi đoạn văn được đánh dấu là “tích cực” hoặc “tiêu cực”. Mô hình học từ các biểu diễn đặc trưng của văn bản (như TF-IDF [3] hoặc word embeddings [4]) để đưa ra dự đoán cho các văn bản mới. Một số thuật toán điển hình bao gồm *Logistic Regression* [5], *Support Vector Machine* [6], *LSTM* [7] và các mô hình dựa trên *Transformer* [8].
- Trong lĩnh vực thị giác máy tính (*Computer Vision*), học có giám sát là nền tảng của nhiều ứng dụng quan trọng như nhận diện khuôn mặt, phân loại hình ảnh, nhận dạng chữ viết tay, phát hiện vật thể, hoặc hỗ trợ xe tự hành. Các mô hình được huấn luyện trên tập dữ liệu hình ảnh có nhãn (ví dụ: hình ảnh kèm nhãn “ô tô”, “người”, “biển báo”) để học cách phân biệt các đối tượng dựa trên đặc trưng thị giác như cạnh, màu sắc, hoặc hình dạng. Kiến trúc phổ biến được sử dụng là mạng tích chập (*Convolutional Neural Networks* [9] - *CNN*), với ưu điểm mang lại hiệu quả cao trong trích xuất đặc trưng và phân loại chính xác.

Tổng thể, học có giám sát không chỉ đóng vai trò như một công cụ phân loại hoặc dự đoán, mà còn là nền tảng giúp các hệ thống trí tuệ nhân tạo hiểu và tương tác với thế giới một cách hiệu quả, chính xác và thông minh hơn.

1.2 Học không giám sát

1.2.1 Đặt vấn đề

Trong phần 1.1, ta đã trình bày về học có giám sát, mô hình máy học được huấn luyện trên một tập dữ liệu đã được gán nhãn (*labeled data*), tức là mỗi mẫu dữ liệu đều đi kèm với giá trị đầu ra chính xác. Quá trình huấn luyện này tương tự như việc học dưới sự hướng dẫn của một giáo viên, với mục tiêu xây dựng một hàm ánh xạ tối ưu để dự đoán biến đầu ra \mathcal{Y} từ dữ liệu đầu vào mới \mathcal{X} . Tuy nhiên, trong thực tế, phần lớn dữ liệu không có sẵn nhãn — được gọi là dữ liệu chưa được gán nhãn (*unlabeled data*) — và chiếm tỷ lệ đáng kể trong nhiều bài toán ứng dụng.

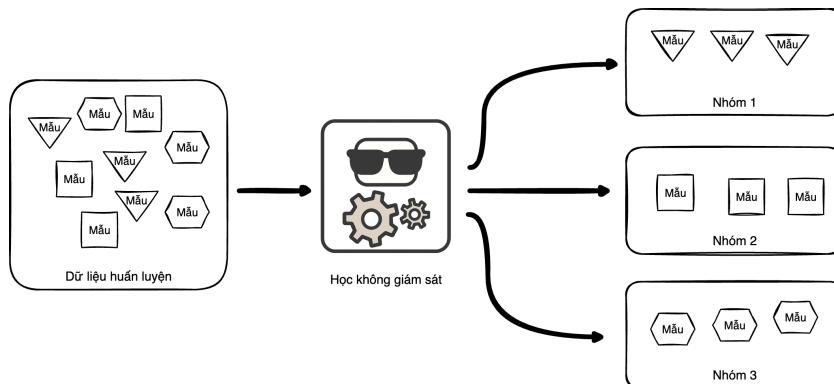
Chẳng hạn, đối với dữ liệu hành vi khách hàng (bao gồm các đặc trưng như số tiền chi tiêu, thời gian truy cập, v.v.), ta có thể phân nhóm khách hàng theo đặc điểm hành vi, ví dụ: nhóm

CHƯƠNG 1. HỌC CÓ GIÁM SÁT VÀ KHÔNG GIÁM SÁT

có mức chi tiêu cao nhưng thời gian ra quyết định ngắn, hoặc nhóm có mức chi tiêu thấp nhưng thời gian truy cập dài. Kết quả phân nhóm này giúp doanh nghiệp xây dựng các chiến lược kinh doanh phù hợp cho từng đối tượng khách hàng. Trong học máy, các phương pháp xử lý dữ liệu chưa được gán nhãn được gọi là học không giám sát (*unsupervised learning*).

1.2.2 Khái niệm

Học không có giám sát (*unsupervised learning*) [10] là một nhánh quan trọng trong học máy, trong đó các thuật toán tự động phân tích và khám phá các cấu trúc hoặc mẫu ẩn trong dữ liệu mà không cần đến nhãn dữ liệu có sẵn. Khác với học có giám sát, học không giám sát chỉ sử dụng dữ liệu đầu vào \mathcal{X} mà không có thông tin về biến đầu ra \mathcal{Y} tương ứng. Mục tiêu của phương pháp này là để mô hình tự động nhận diện các đặc trưng tiềm ẩn và mối quan hệ nội tại trong dữ liệu. Phương pháp học không giám sát được minh họa trong Hình 1.2.



Hình 1.2: Minh họa về học không giám sát

Một khái niệm then chốt trong học không giám sát là tự tổ chức (*self-organizing*). Trong quá trình này, mô hình có khả năng tự động phân loại và tổ chức dữ liệu mà không cần sự can thiệp trực tiếp từ con người. Các phương pháp học không giám sát không chỉ dừng lại ở việc phân nhóm dữ liệu (*clustering*), mà còn bao gồm việc xây dựng các mô hình thống kê hoặc mô hình xác suất để biểu diễn cấu trúc dữ liệu.

Học không giám sát là một trong ba nhánh chính của học máy, bên cạnh học có giám sát và học củng cố (*reinforcement learning* [11]). Bên cạnh đó, một biến thể quan trọng là học bán giám sát (*semi-supervised learning* [12]), kết hợp giữa dữ liệu có nhãn và không nhãn trong quá trình huấn luyện. Phương pháp này đặc biệt hữu ích trong các tình huống mà việc gán nhãn dữ liệu tốn nhiều chi phí hoặc khó thực hiện — chẳng hạn như trong bài toán nhận dạng hình ảnh hoặc phân tích các tập văn bản quy mô lớn — và giúp cải thiện hiệu suất của mô hình.

Ngày nay, học không giám sát đóng vai trò ngày càng quan trọng trong khai thác dữ liệu, cho phép các tổ chức và doanh nghiệp tận dụng hiệu quả nguồn dữ liệu chưa được gán nhãn. Phương pháp này không chỉ hỗ trợ phát hiện các mẫu tiềm ẩn mà còn góp phần nâng cao trải nghiệm người dùng thông qua các giải pháp thông minh và mang tính cá nhân hóa.

1.2.3 Nguyên lý hoạt động

Khác với học có giám sát, trong học không giám sát, các thuật toán được triển khai mà không có sự hỗ trợ của nhãn dữ liệu đi kèm. Thay vào đó, chúng xử lý trực tiếp tập dữ liệu đầu vào

1.2. HỌC KHÔNG GIÁM SÁT

thô, vốn chỉ bao gồm các đặc trưng \mathcal{X} , nhằm tự động khám phá các mẫu hình hoặc cấu trúc tiềm ẩn trong dữ liệu. Mục tiêu chính của học không giám sát là xác định các mối quan hệ, mức độ tương đồng, hoặc phân nhóm các điểm dữ liệu sao cho những điểm có đặc điểm giống nhau được nhóm lại, trong khi các điểm khác biệt được phân tách rõ ràng.

Cụ thể, các thuật toán thuộc lĩnh vực này thực hiện phân tích các đặc trưng của dữ liệu và tự động tổ chức chúng thành các nhóm, hoặc khám phá các quy luật ẩn chưa được biết trước. Quá trình này dựa trên việc đo lường sự tương đồng hoặc khác biệt giữa các điểm dữ liệu mà không cần đến thông tin về nhãn hoặc phân loại đã được xác định từ trước. Mục đích là phát hiện các cấu trúc tiềm ẩn trong không gian đặc trưng, chẳng hạn như việc nhóm các đối tượng có tính chất tương tự hoặc xác định các mối quan hệ giữa các đặc trưng của dữ liệu.

Một số phương pháp phổ biến trong học không giám sát bao gồm phân cụm (*clustering*) và giảm chiều dữ liệu (*dimensionality reduction*). Phân cụm nhằm mục đích nhóm các điểm dữ liệu tương tự vào cùng một cụm, và giảm chiều dữ liệu tập trung vào việc trích xuất các đặc trưng quan trọng nhất từ một tập dữ liệu có kích thước lớn, qua đó giảm thiểu độ phức tạp mà vẫn bảo toàn được thông tin cốt lõi.

Tóm lại, học không giám sát hướng đến việc khám phá và làm sáng tỏ các cấu trúc tiềm ẩn trong dữ liệu mà không cần đến thông tin nhãn hoặc kết quả đầu ra \mathcal{Y} . Phương pháp này đặc biệt hữu ích trong các bài toán như phân tích hành vi người dùng hoặc xử lý các tập dữ liệu lớn chưa được gán nhãn, từ đó hỗ trợ việc khai thác hiệu quả các thông tin có giá trị từ dữ liệu thô.

1.2.4 Phân loại

Trong lĩnh vực công nghệ thông tin, đặc biệt là học máy, học không giám sát thường được ứng dụng để phân chia dữ liệu thành các nhóm hoặc khám phá các mối quan hệ tiềm ẩn giữa các yếu tố trong dữ liệu. Các phương pháp chính bao gồm phân cụm và kết hợp, được trình bày chi tiết như sau:

- Phân cụm (*clustering*) là quá trình chia tập dữ liệu thành các nhóm sao cho các điểm dữ liệu trong cùng một nhóm có mức độ tương đồng cao, đồng thời khác biệt rõ rệt so với các nhóm khác. Phương pháp này cho phép nhận diện các nhóm dữ liệu có đặc điểm chung mà không cần thông tin nhãn trước. Ví dụ, trong phân tích hành vi khách hàng, phân cụm có thể được sử dụng để nhóm khách hàng dựa trên thói quen mua sắm, từ đó hỗ trợ xây dựng các chiến lược tiếp thị phù hợp và hiệu quả.
- Kết hợp (*association*) là phương pháp tập trung vào việc khám phá các quy luật hoặc mối quan hệ giữa các yếu tố trong dữ liệu. Một ứng dụng phổ biến của kết hợp là trong các hệ thống gợi ý, nơi các mẫu đồng xuất hiện của các yếu tố được phát hiện. Chẳng hạn, trong lĩnh vực bán lẻ, dữ liệu có thể cho thấy rằng khách hàng mua áo sơ mi thường có xu hướng mua thêm đồng hồ hoặc thắt lưng. Tương tự, trong ngành giải trí, những người xem phim về Người Nhện thường có xu hướng xem thêm phim về Người Dơi¹. Các quy luật này hỗ trợ việc xây dựng các hệ thống gợi ý thông minh, nhằm khuyến khích khách hàng mua sắm hoặc khám phá thêm các nội dung liên quan.

1.2.5 Một số giải thuật học không giám sát

Các giải thuật học không giám sát được thiết kế để giải quyết các bài toán phân cụm hoặc kết hợp, với một số phương pháp phổ biến bao gồm:

¹Người Nhện và Người Dơi là hai loạt phim điện ảnh siêu anh hùng nổi tiếng của Hoa Kỳ, dựa trên các nhân vật truyện tranh cùng tên.

- Phân cụm K-means (*K-means Clustering* [13]) là một trong những giải thuật phân cụm phổ biến nhất, chia dữ liệu thành k nhóm sao cho các điểm trong cùng nhóm có mức độ tương đồng cao. Giải thuật khởi tạo ngẫu nhiên k trung tâm cụm (*centroids*), sau đó lặp lại quá trình cập nhật vị trí các trung tâm và phân bổ dữ liệu để tối ưu hóa việc phân nhóm.
- Phân cụm phân cấp (*Hierarchical Clustering* [14]) xây dựng một cấu trúc phân cấp dựa trên sơ đồ cây (*dendrogram*), cho phép phân tích dữ liệu ở nhiều mức độ chi tiết. Có hai cách tiếp cận chính: gộp cụm từ dưới lên (*agglomerative*), trong đó các điểm dữ liệu được hợp nhất dần thành các cụm lớn hơn; và phân cụm từ trên xuống (*divisive*), trong đó toàn bộ tập dữ liệu được chia tách dần thành các cụm nhỏ hơn.
- Mô hình hỗn hợp Gaussian (*Gaussian Mixture Models - GMM* [15]) là một mô hình thống kê sử dụng tổ hợp các phân phối Gaussian để mô tả cấu trúc cụm trong dữ liệu. Mô hình giả định rằng dữ liệu được tạo ra từ nhiều phân phối Gaussian khác nhau, và sử dụng thuật toán kỳ vọng-tối đa (*Expectation-Maximization - EM*) để ước lượng các tham số. So với K-means, GMM linh hoạt hơn trong việc xử lý các cụm có hình dạng và phương sai khác nhau.
- Phân cụm dựa trên mật độ có nhiễu (*Density-Based Spatial Clustering of Applications with Noise - DBSCAN* [16]) xác định các cụm dựa trên mật độ điểm dữ liệu trong không gian, cho phép nhận diện các cụm có hình dạng bất kỳ mà không cần xác định trước số lượng cụm k . Ngoài ra, DBSCAN có khả năng xử lý tốt các điểm nhiễu, giúp phân biệt giữa các cụm chính và các điểm dữ liệu bất thường.

1.2.6 Ứng dụng

Học không giám sát là một kỹ thuật cốt lõi trong học máy, được ứng dụng rộng rãi nhờ khả năng khám phá các cấu trúc tiềm ẩn trong dữ liệu mà không cần đến nhãn. Trong lĩnh vực tiếp thị và bán hàng, các doanh nghiệp tận dụng học không giám sát để phân tích hành vi khách hàng thông qua phân cụm. Phương pháp này cho phép phân nhóm khách hàng dựa trên các đặc trưng như tần suất mua sắm, sở thích sản phẩm hoặc mức chi tiêu, từ đó hỗ trợ xây dựng các chiến dịch tiếp thị cá nhân hóa, nâng cao hiệu quả quảng bá và cải thiện trải nghiệm người dùng.

- Trong ngành tài chính, học không giám sát đóng vai trò quan trọng trong phát hiện bất thường (*anomaly detection*), giúp nhận diện các giao dịch có dấu hiệu khác thường so với hành vi thông thường. Chẳng hạn, các thuật toán như phân cụm dựa trên mật độ có nhiễu (*DBSCAN*) được sử dụng để phát hiện gian lận tài chính, góp phần tăng cường tính an toàn và bảo mật cho hệ thống.
- Trong lĩnh vực an ninh công đồng, học không giám sát hỗ trợ phân tích dữ liệu từ hệ thống camera giám sát nhằm phát hiện các hành vi nguy hiểm hoặc bất thường. Bằng cách phân nhóm các mẫu hành vi không nhãn, hệ thống có thể đưa ra cảnh báo sớm, hỗ trợ công tác giám sát và phản ứng kịp thời.
- Trong chăm sóc sức khỏe, học không giám sát được áp dụng để phân tích các tập dữ liệu bệnh nhân quy mô lớn, giúp phát hiện các mẫu bệnh lý tiềm ẩn. Các thuật toán phân cụm có thể nhận diện những nhóm bệnh nhân có đặc điểm lâm sàng tương đồng, từ đó hỗ trợ chẩn đoán chính xác hơn và xây dựng phác đồ điều trị phù hợp.

1.2. HỌC KHÔNG GIÁM SÁT

- Trong các ngành công nghiệp như dầu khí và vận tải, học không giám sát được ứng dụng trong giám sát tình trạng thiết bị thông qua phát hiện bất thường. Phương pháp này giúp nhận diện sớm các dấu hiệu có thể dẫn đến hỏng hóc hoặc gián đoạn vận hành, từ đó nâng cao hiệu quả hoạt động và giảm thiểu rủi ro kỹ thuật.

Chương 2

Thuật toán k-NN

2.1 Bài toán mở đầu

Hai cá nhân, Phương và Phát lần đầu tiên đến Đà Lạt và muốn chọn quán cà phê phù hợp với sở thích: yên tĩnh, có quang cảnh đẹp và giá cả hợp lý. Tuy nhiên, họ không có nhiều thông tin và quyết định sử dụng một ứng dụng đánh giá địa điểm. Ứng dụng này cho phép người dùng tìm các quán cà phê tương tự dựa trên những địa điểm mà họ đã thích trước đó.

Phương và Phát đánh dấu ba quán từng ghé ở Hà Nội: quán A có không gian yên tĩnh, quang cảnh hồ; quán B có nội thất đẹp, giá hợp lý; và quán C hơi ồn nhưng có đồ uống rất ngon. Ứng dụng sử dụng thông tin từ ba quán này để tìm các quán cà phê tại Đà Lạt có đặc điểm tương đồng nhất, cụ thể là những quán gần nhất trong không gian đặc trưng về “giá cả”, “độ ồn”, “quang cảnh”, và “chất lượng đồ uống”.

Từ các quán ở Đà Lạt, hệ thống chọn ra 3 quán “gần nhất” với sở thích của Phương và Phát - tức là có đặc điểm gần giống nhất trong không gian đặc trưng. Sau đó, hệ thống tổng hợp thông tin của 3 quán này và đề xuất quán X là lựa chọn phù hợp. Thực tế trải nghiệm cho thấy Phương và Phát rất hài lòng với quán này.

Ví dụ trên minh họa cách hoạt động của thuật toán k-Nearest Neighbors: khi cần phân loại hoặc dự đoán một điểm dữ liệu mới, hệ thống sẽ tìm k điểm giống nhất về đặc điểm trong tập dữ liệu đã biết. Từ đó, hệ thống sử dụng thông tin của các điểm tương đồng này để đưa ra dự đoán. Cụm từ “gần nhất” không nói về vị trí địa lý, mà là khoảng cách nhỏ nhất trong không gian đặc trưng - nghĩa là mức độ tương đồng cao về các yếu tố như giá cả, độ ồn, quang cảnh, chất lượng đồ uống.

Thuật toán k-Nearest Neighbors là một phương pháp học không giám sát đơn giản nhưng hiệu quả, đặc biệt khi các đặc trưng được biểu diễn rõ ràng và nhất quán. Chất lượng dự đoán phụ thuộc nhiều vào việc lựa chọn đặc trưng phù hợp và cách đo khoảng cách chính xác, chứ không phụ thuộc vào “độ tin cậy” chủ quan của dữ liệu.

2.2 Khái niệm

Thuật toán k-Nearest Neighbors (k-NN), được giới thiệu bởi Thomas Cover và Peter Hart [17], là một trong những phương pháp đơn giản và hiệu quả nhất thuộc nhóm học có giám sát. Thuật toán này thuộc lớp học dựa trên mẫu, nghĩa là thay vì xây dựng một mô hình tổng quát từ dữ liệu huấn luyện, k-NN lưu trữ toàn bộ tập dữ liệu huấn luyện và sử dụng nó để đưa ra dự đoán

2.3. KHOẢNG CÁCH GIỮA CÁC ĐIỂM DỮ LIỆU

cho các điểm dữ liệu mới. Do đó, k-NN không có giai đoạn huấn luyện thực sự, mà các phép tính được thực hiện trong giai đoạn dự đoán, khi thuật toán tìm kiếm k điểm dữ liệu gần nhất với điểm dữ liệu mới trong không gian đặc trưng.

Trong bài toán phân loại, thuật toán k-NN xác định k điểm dữ liệu gần nhất với một điểm mới dựa trên độ đo khoảng cách đã chọn. Nhãn của điểm mới được gán theo nguyên tắc bỏ phiếu đa số — tức là nhãn phổ biến nhất trong số k lân cận. Phương pháp này tỏ ra hiệu quả trong nhiều ứng dụng thực tế, chẳng hạn như phân loại thư điện tử thành thư rác hoặc thư hợp lệ, dựa trên sự tương đồng với các mẫu đã được gán nhãn trước đó.

Trong bài toán hồi quy, k-NN dự đoán giá trị liên tục bằng cách tính giá trị trung bình của các giá trị tương ứng từ k điểm lân cận. Tổng thể, k-NN là một công cụ linh hoạt, dễ triển khai trong học có giám sát, với hiệu quả phụ thuộc vào việc chọn giá trị k và độ đo khoảng cách phù hợp. Tương tự như việc đưa ra quyết định dựa trên các nguồn thông tin đáng tin cậy trong thực tiễn, k-NN sử dụng các điểm lân cận để đưa ra dự đoán chính xác, minh họa rõ mối liên hệ giữa học máy và các tình huống thực tế.

2.3 Khoảng cách giữa các điểm dữ liệu

Trong học máy, khoảng cách giữa các điểm dữ liệu là một giá trị số học thể hiện mức độ gần hoặc xa giữa hai điểm trong không gian nào đó. Hàm khoảng cách từ điểm x đến điểm y , ký hiệu $d(x, y)$, được gọi là *metric* nếu thỏa mãn các tính chất cơ bản sau:

- **Tính không âm:** $d(x, y) \geq 0$ với mọi x, y .
- **Tính duy nhất:** $d(x, y) = 0$ khi và chỉ khi $x = y$.
- **Tính đối xứng:** $d(x, y) = d(y, x)$ với mọi x, y .
- **Bất đẳng thức tam giác:** $d(x, y) \leq d(x, z) + d(z, y)$ với mọi x, y, z .

Các tính chất này đảm bảo rằng hàm khoảng cách đo lường sự tương đồng hoặc khác biệt giữa các điểm dữ liệu một cách nhất quán [18].

Trong các bài toán học máy, điểm dữ liệu thường được biểu diễn dưới dạng vector trong không gian p -chiều, với mỗi điểm $x_i = (x_{i1}, x_{i2}, \dots, x_{ip})$ đại diện cho một tập hợp p đặc trưng, và $i = 1, 2, \dots, n$, trong đó n là tổng số điểm dữ liệu. Tùy thuộc vào bài toán cụ thể, nhiều hàm khoảng cách khác nhau có thể được sử dụng, mỗi loại có ưu điểm và ứng dụng riêng. Các hàm khoảng cách phổ biến bao gồm *khoảng cách Euclid*, *khoảng cách Manhattan*, *khoảng cách Hamming* [19], *khoảng cách Minkowski* [18], và *độ tương tự Cosine* [20].

2.3.1 Khoảng cách Manhattan

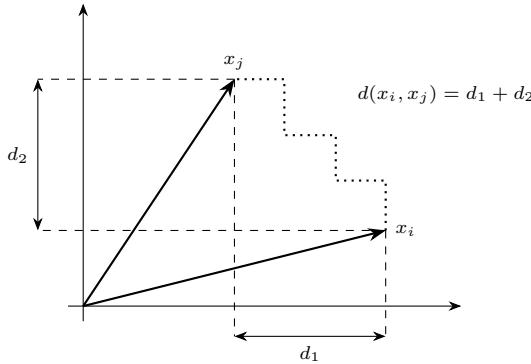
Khoảng cách Manhattan (*Manhattan distance*), còn được gọi là khoảng cách trong thành phố (*city block distance*), là một thước đo khoảng cách trong không gian Euclid, được sử dụng để xác định độ xa giữa hai điểm dữ liệu dựa trên tổng giá trị tuyệt đối của hiệu tọa độ trên từng chiều. Khác với khoảng cách Euclid, vốn tính theo đường thẳng, khoảng cách Manhattan giả định rằng việc di chuyển giữa hai điểm chỉ có thể thực hiện song song với các trục tọa độ. Công thức 2.1 biểu diễn khoảng cách Manhattan.

$$d(x_i, x_j) = \sum_{k=1}^p |x_{ik} - x_{jk}| \quad (2.1)$$

Trong đó:

- x_{ik} và x_{jk} lần lượt là giá trị của đặc trưng thứ k của các điểm dữ liệu x_i và x_j .
- p là số chiều của không gian đặc trưng.

Về mặt hình học, khoảng cách Manhattan biểu thị quãng đường ngắn nhất giữa hai điểm trong không gian p -chiều khi chỉ di chuyển dọc theo các trục tọa độ, tương tự như cách di chuyển trong một thành phố có hệ thống đường phố vuông góc theo mô hình lưới (*grid-based city layout*). Đặc điểm này làm cho khoảng cách Manhattan trở thành một công cụ quan trọng trong các lĩnh vực học máy, phân tích dữ liệu không gian, và tối ưu hóa lô trình [21]. Hình 2.1 minh họa khoảng cách Manhattan trong không gian hai chiều, với các điểm dữ liệu được kết nối bằng các đoạn thẳng song song với trục tọa độ.



Hình 2.1: Khoảng cách Manhattan trong không gian hai chiều.

Ngoài công thức cơ bản, khoảng cách Manhattan còn được mở rộng thông qua các biến thể, mỗi biến thể được thiết kế để phù hợp với các ứng dụng cụ thể trong các lĩnh vực khác nhau. Dưới đây là hai biến thể phổ biến:

- **Khoảng cách Canberra** (*Canberra distance*) là một biến thể của khoảng cách Manhattan, đặc biệt nhạy với các thay đổi nhỏ gần giá trị 0, do đó thường được sử dụng trong các bài toán phân tích dữ liệu đa chiều. Công thức tính khoảng cách Canberra được định nghĩa như sau:

$$d(x_i, x_j) = \sum_{k=1}^p \frac{|x_{ik} - x_{jk}|}{|x_{ik}| + |x_{jk}|} \quad (2.2)$$

Công thức 2.2 chuẩn hóa khoảng cách bằng cách chia hiệu tuyệt đối cho tổng giá trị tuyệt đối của các đặc trưng, giúp giảm thiểu ảnh hưởng của các giá trị lớn trong tập dữ liệu [22].

- **Khoảng cách Sorensen** (*Sørensen distance*): Biến thể này được ứng dụng rộng rãi trong các nghiên cứu sinh thái học và phân tích sự tương đồng giữa các cộng đồng hoặc quần thể sinh vật. Công thức của khoảng cách Sorensen được biểu diễn như sau:

$$d(x_i, x_j) = 1 - \frac{2 \sum_{k=1}^p \min(x_{ik}, x_{jk})}{\sum_{k=1}^p (x_{ik} + x_{jk})} \quad (2.3)$$

2.3. KHOẢNG CÁCH GIỮA CÁC ĐIỂM DỮ LIỆU

Trong đó:

- p là số chiều (hoặc số đặc trưng) của vector dữ liệu.
- x_i, x_j là hai vector dữ liệu (có p chiều).
- x_{ik}, x_{jk} là giá trị tại chiều thứ k của vector x_i và x_j .

So với khoảng cách Canberra, khoảng cách Sorensen tập trung vào sự tương đồng giữa các điểm dữ liệu, thường được sử dụng cho dữ liệu không âm [23].

Một trường hợp đặc biệt của khoảng cách Manhattan là khoảng cách Hamming (*Hamming distance*), được sử dụng phổ biến trong các bài toán liên quan đến dữ liệu nhị phân hoặc dữ liệu rời rạc. Khoảng cách Hamming đo lường số lượng vị trí khác biệt giữa hai chuỗi có cùng độ dài, thường được áp dụng trong mã hóa, phân tích dữ liệu phân loại nhị phân, và xử lý chuỗi [24].

Khoảng cách Manhattan và các biến thể của nó đóng vai trò quan trọng trong nhiều lĩnh vực, từ các thuật toán học máy k-NN đến các ứng dụng thực tiễn như tối ưu hóa lộ trình và phân tích dữ liệu không gian. Việc lựa chọn loại khoảng cách phù hợp phụ thuộc vào đặc điểm của dữ liệu và yêu cầu cụ thể của bài toán.

2.3.2 Khoảng cách Euclid

Khoảng cách Euclid (*Euclidean distance*) là một trong những độ đo khoảng cách phổ biến nhất trong học máy, được sử dụng rộng rãi trong các bài toán như phân cụm, k-Nearest Neighbors [17], [25]. Độ đo này đặc biệt phù hợp cho các tập dữ liệu liên tục, nơi các điểm dữ liệu được biểu diễn dưới dạng vector trong không gian đặc trưng p -chiều. Khoảng cách Euclid giữa hai điểm x_i và x_j được định nghĩa bằng Công thức 2.4.

$$d(x_i, x_j) = \sqrt{\sum_{k=1}^p (x_{ik} - x_{jk})^2} \quad (2.4)$$

Trong đó:

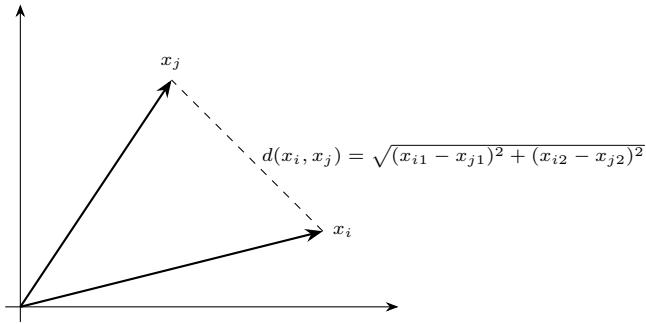
- x_{ik} và x_{jk} là giá trị của đặc trưng thứ k của các điểm x_i và x_j , tương ứng.
- p là số lượng đặc trưng (số chiều) của mỗi điểm dữ liệu.

Về mặt hình học, khoảng cách Euclid biểu thị độ dài của đoạn thẳng nối hai điểm trong không gian p -chiều. Trong trường hợp hai chiều ($p = 2$), khoảng cách này tương ứng với độ dài cạnh huyền của một tam giác vuông, như Hình 2.2. Độ đo này trực quan và dễ tính toán, khiến nó trở thành lựa chọn mặc định trong nhiều thuật toán học máy.

Khoảng cách Euclid có một số ưu điểm nổi bật, bao gồm tính đơn giản, khả năng biểu diễn trực quan, và phù hợp với các bài toán có dữ liệu phân bố đồng đều. Tuy nhiên, nó cũng có hạn chế, đặc biệt khi các đặc trưng có đơn vị hoặc phạm vi khác nhau. Trong trường hợp này, dữ liệu cần được *chuẩn hóa* trước khi áp dụng khoảng cách Euclid để tránh thiên lệch do sự khác biệt về tỷ lệ [25]. Ví dụ, trong bài toán k-NN phân loại hình ảnh, nếu các đặc trưng như cường độ pixel không được chuẩn hóa, các đặc trưng có giá trị lớn hơn sẽ chi phối khoảng cách, dẫn đến kết quả không chính xác.

Một biến thể quan trọng của khoảng cách Euclid là khoảng cách Euclid bình phương, được định nghĩa trong công thức 2.5:

$$d(x_i, x_j) = \sqrt{\sum_{k=1}^p (x_{ik} - x_{jk})^2} \quad (2.5)$$



Hình 2.2: Khoảng cách Euclid giữa hai điểm trong không gian hai chiều.

Công thức 2.5 cho phép loại bỏ phép tính căn bậc hai, giúp giảm chi phí tính toán trong các thuật toán tối ưu hóa. Mặc dù khoảng cách Euclid bình phương không thỏa mãn tính chất *tính duy nhất* của một hàm khoảng cách (vì nó không đảm bảo $d(x, y) = 0$ chỉ khi $x = y$), nó vẫn được sử dụng rộng rãi do tính chất bảo toàn thứ tự khoảng cách.

2.3.3 Khoảng cách Hamming

Khoảng cách Hamming (*Hamming Distance*), được đặt theo tên của nhà toán học Richard W. Hamming, người đã giới thiệu nó vào năm 1950 trong công trình nghiên cứu về mã hóa và phát hiện lỗi [24], là một độ đo được sử dụng để đánh giá mức độ khác biệt giữa hai vector, đặc biệt phổ biến trong các bài toán liên quan đến chuỗi nhị phân, phân tích bit-wise, và phát hiện lỗi trong lý thuyết mã hóa. Độ đo này được định nghĩa là số lượng vị trí mà tại đó các phần tử tương ứng của hai vector khác nhau. Công thức toán học của khoảng cách Hamming giữa hai điểm dữ liệu x_i và x_j được biểu diễn như sau:

$$d_H(x_i, x_j) = \sum_{k=1}^d \mathbb{I}\{x_{ik} \neq x_{jk}\} \quad (2.6)$$

Trong đó:

- $x_i, x_j \in \mathcal{X}^d$: các vector đặc trưng, thường thuộc không gian rời rạc.
- x_{ik}, x_{jk} : giá trị của đặc trưng thứ k của x_i và x_j .
- $\mathbb{I}\{x_{ik} \neq x_{jk}\}$: hàm chỉ thị, trả về 1 nếu $x_{ik} \neq x_{jk}$, và 0 nếu $x_{ik} = x_{jk}$.
- d : số lượng đặc trưng của các vector.

Khoảng cách Hamming có ý nghĩa thực tiễn trong việc đo lường số bước tối thiểu cần thiết để chuyển đổi một chuỗi thành chuỗi khác thông qua thay đổi từng vị trí (*bit flipping*). Độ đo này được ứng dụng rộng rãi trong nhiều lĩnh vực, bao gồm:

- Mã hóa và sửa lỗi: Được sử dụng trong các mã sửa lỗi như mã Hamming để phát hiện và sửa lỗi trong truyền thông số.
- Phân tích dữ liệu phân loại: So sánh các đặc trưng phân loại trong học máy.
- Xử lý chuỗi: Dánh giá sự khác biệt giữa hai chuỗi ký tự, chẳng hạn trong xử lý ngôn ngữ tự nhiên.

2.3. KHOẢNG CÁCH GIỮA CÁC ĐIỂM DỮ LIỆU

Một tính chất đặc đáo và hữu ích của khoảng cách Hamming là khả năng biểu diễn thông qua phép toán hoặc loại trừ (*exclusive OR*, viết tắt là *XOR*) đối với các chuỗi nhị phân. Cụ thể, với hai chuỗi nhị phân a và b có cùng độ dài, khoảng cách Hamming được tính như sau:

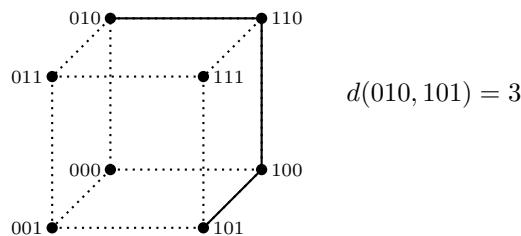
$$\text{Khoảng cách Hamming}(a, b) = \text{Số lượng bit 1 trong } (a \oplus b) \quad (2.7)$$

Trong đó \oplus biểu thị phép toán XOR, trả về 1 tại các vị trí mà các bit tương ứng của a và b khác nhau, và 0 nếu chúng giống nhau. Số lượng bit 1 trong kết quả XOR chính là khoảng cách Hamming giữa hai chuỗi.

Ví dụ 2.1. Xét hai chuỗi nhị phân $a = 010$ và $b = 101$. Ta thực hiện phép toán XOR như sau:

$$a \oplus b = 010 \oplus 101 = 111$$

Kết quả cho thấy chuỗi XOR có 3 bit 1, do đó khoảng cách Hamming giữa a và b là 3, tương ứng với 3 vị trí bit khác nhau giữa hai chuỗi. Quá trình này được minh họa trực quan trong Hình 2.3.



Hình 2.3: Mô phỏng khoảng cách Hamming giữa hai chuỗi nhị phân 010 và 101.

Khoảng cách Hamming thỏa mãn các tính chất của một hàm khoảng cách, bao gồm:

- Tính không âm: $d(x_i, x_j) \geq 0$, và $d(x_i, x_j) = 0$ nếu và chỉ nếu $x_i = x_j$.
- Tính đối xứng: $d(x_i, x_j) = d(x_j, x_i)$.
- Bất đẳng thức tam giác: $d(x_i, x_j) \leq d(x_i, x_k) + d(x_k, x_j)$.

Những tính chất này làm cho khoảng cách Hamming trở thành một công cụ mạnh mẽ trong các thuật toán k-NN, nơi nó được sử dụng để đo lường sự khác biệt giữa các mẫu dữ liệu danh mục hoặc nhị phân [17].

Mặc dù khoảng cách Hamming rất hiệu quả với các chuỗi nhị phân hoặc danh mục, nó không phù hợp với dữ liệu số liên tục, vì nó không xem xét đến độ lớn của sự khác biệt giữa các giá trị. Để khắc phục, các độ đo khác như khoảng cách Euclid hoặc khoảng cách Minkowski thường được sử dụng. Ngoài ra, các phương pháp học khoảng cách, chẳng hạn như *Large Margin Nearest Neighbor* (viết tắt là *LMNN*), có thể được áp dụng để tối ưu hóa khoảng cách Hamming trong các bài toán phân loại [26].

2.3.4 Khoảng cách Minkowski

Khoảng cách Minkowski là một độ đo tổng quát trong không gian vector chuẩn hóa, được giới thiệu bởi nhà toán học Hermann Minkowski trong [27] xuất bản năm 1896. Độ đo này bao hàm nhiều loại khoảng cách phổ biến như *khoảng cách Manhattan*, *khoảng cách Euclid*, và *khoảng*

cách Chebyshev [18]. Độ đo này được định nghĩa dựa trên *định chuẩn* L_p (L_p -norm), cho phép linh hoạt điều chỉnh thông qua tham số n . Phương trình 2.8 biểu diễn khoảng cách Minkowski giữa hai điểm dữ liệu x_i và x_j .

$$d(x_i, x_j) = \left(\sum_{k=1}^p |x_{ik} - x_{jk}|^n \right)^{\frac{1}{n}} \quad (2.8)$$

Trong đó:

- x_{ik} và x_{jk} lần lượt là giá trị của đặc trưng thứ k của hai điểm dữ liệu x_i và x_j .
- n là tham số điều chỉnh, được gọi là thứ tự của định chuẩn (*order of the norm*), quyết định tính chất của khoảng cách.

Khoảng cách Minkowski là một công cụ mạnh mẽ trong học máy, khai phá dữ liệu và phân tích dữ liệu, nhờ khả năng tổng quát hóa các dạng khoảng cách khác nhau dựa trên giá trị của n . Các trường hợp cụ thể của n bao gồm:

- Khi $n = 1$: Khoảng cách Minkowski trở thành khoảng cách Manhattan, được tính bằng tổng các giá trị tuyệt đối của hiệu các đặc trưng:

$$d(x_i, x_j) = \sum_{k=1}^p |x_{ik} - x_{jk}| \quad (2.9)$$

- Khi $n = 2$: Khoảng cách Minkowski trở thành khoảng cách Euclid, biểu diễn khoảng cách hình học trong không gian Euclid:

$$d(x_i, x_j) = \sqrt{\sum_{k=1}^p (x_{ik} - x_{jk})^2} \quad (2.10)$$

- Khi $n \rightarrow \infty$: Khoảng cách Minkowski hội tụ về khoảng cách Chebyshev, được định nghĩa trong Phương trình 2.11.

$$d(x_i, x_j) = \lim_{n \rightarrow \infty} \left(\sum_{k=1}^p |x_{ik} - x_{jk}|^n \right)^{\frac{1}{n}} = \max_{k=1}^p |x_{ik} - x_{jk}| \quad (2.11)$$

Khoảng cách Chebyshev đo lường độ khác biệt lớn nhất giữa các đặc trưng của hai điểm dữ liệu, thường được sử dụng trong các bài toán tối ưu hóa.

- Khi $n \rightarrow -\infty$: Khoảng cách Minkowski hội tụ về giá trị nhỏ nhất của sự khác biệt giữa các đặc trưng:

$$d(x_i, x_j) = \lim_{n \rightarrow -\infty} \left(\sum_{k=1}^p |x_{ik} - x_{jk}|^n \right)^{\frac{1}{n}} = \min_{k=1}^p |x_{ik} - x_{jk}| \quad (2.12)$$

Trường hợp này ít được sử dụng trong thực tế nhưng có ý nghĩa lý thuyết trong việc phân tích hành vi của *định chuẩn* L_p ở các giá trị cực biên.

2.3. KHOẢNG CÁCH GIỮA CÁC ĐIỂM DỨ LIỆU

Khi $n \geq 1$, khoảng cách Minkowski thỏa mãn các tính chất của một hàm khoảng cách, nhờ vào bất đẳng thức Minkowski. Bất đẳng thức này được biểu diễn như sau:

$$\left(\sum_{i=1}^p |x_i + y_i|^n \right)^{\frac{1}{n}} \leq \left(\sum_{i=1}^n |x_i|^n \right)^{\frac{1}{n}} + \left(\sum_{i=1}^n |y_i|^n \right)^{\frac{1}{n}}, \quad \forall n \geq 1 \quad (2.13)$$

Bất đẳng thức 2.13 đảm bảo rằng khoảng cách Minkowski thỏa mãn các tính chất sau:

- Tính không âm: $d(x_i, x_j) \geq 0$, và $d(x_i, x_j) = 0$ nếu và chỉ nếu $x_i = x_j$.
- Tính đối xứng: $d(x_i, x_j) = d(x_j, x_i)$.
- Bất đẳng thức tam giác: $d(x_i, x_j) \leq d(x_i, x_k) + d(x_k, x_j)$.

Tuy nhiên, khi $n < 1$, khoảng cách Minkowski không còn thỏa mãn bất đẳng thức tam giác và do đó không phải là một hàm khoảng cách. Chẳng hạn, với $n = 0.5$, xét ba điểm $x = (0, 0)$, $y = (1, 0)$, và $z = (1, 1)$; khi đó, dễ dàng kiểm tra rằng bất đẳng thức tam giác không được thỏa mãn, tức $d(x, z) > d(x, y) + d(y, z)$.

2.3.5 Mối quan hệ với Định chuẩn L_p

Khoảng cách Minkowski thuộc về họ *định chuẩn L_p* (L_p -norm), trong đó tham số $p = n$. Các trường hợp đặc biệt bao gồm:

- $p = 1$: *Định chuẩn L_1* (L_1 -norm), tương ứng với khoảng cách Manhattan.
- $p = 2$: *Định chuẩn L_2* (L_2 -norm), tương ứng với khoảng cách Euclid.
- $p \rightarrow \infty$: *Định chuẩn L_∞* (L_∞ -norm), tương ứng với khoảng cách Chebyshev.

Sự linh hoạt của khoảng cách Minkowski cho phép nó được sử dụng trong nhiều bài toán khác nhau, từ phân loại và phân cụm trong học máy đến tối ưu hóa trong trí tuệ nhân tạo. Việc lựa chọn giá trị n phụ thuộc vào đặc điểm của bài toán và loại dữ liệu, chẳng hạn dữ liệu có phân phối đều hay tập trung vào các giá trị cực đại.

Tuy nhiên, khoảng cách Minkowski có một số hạn chế. Khi n tăng, khoảng cách trở nên nhạy cảm hơn với các giá trị khác biệt lớn, có thể dẫn đến sự thiên lệch trong các bài toán có dữ liệu nhiễu. Ngoài ra, việc chọn giá trị n tối ưu thường đòi hỏi thử nghiệm hoặc kiến thức chuyên môn về bài toán [25].

Ví dụ 2.2. Tính khoảng cách Minkowski với $n = 1$ và $n = 2$ giữa hai điểm $x = (3, 2, 5)$ và $y = (4, 7, 9)$.

- Với $n = 1$ (khoảng cách Manhattan):

$$d(x, y) = |3 - 4| + |2 - 7| + |5 - 9| = 1 + 5 + 4 = 10$$

- Với $n = 2$ (khoảng cách Euclid):

$$d(x, y) = \sqrt{(3 - 4)^2 + (2 - 7)^2 + (5 - 9)^2} = \sqrt{1 + 25 + 16} = \sqrt{42} \approx 6.48$$

Khoảng cách Minkowski là một độ đo linh hoạt, tổng quát hóa nhiều dạng khoảng cách phổ biến, cho phép điều chỉnh thông qua tham số n . Nhờ tính chất này, nó được ứng dụng rộng rãi trong học máy, khai phá dữ liệu, và tối ưu hóa. Tuy nhiên, việc lựa chọn giá trị n phù hợp và hiểu rõ các hạn chế của độ đo là yếu tố quan trọng để đảm bảo hiệu quả trong các ứng dụng thực tế.

2.3.6 Khoảng cách cosine

Trong các bài toán mà sự khác biệt giữa các điểm dữ liệu chủ yếu nằm ở hướng của vector thay vì độ dài của vector, độ tương tự cosine (*cosine similarity*) là một độ đo phổ biến và hiệu quả. Độ tương tự cosine được định nghĩa là cosine của góc giữa hai vector x_i và x_j trong không gian đặc trưng, được tính bằng công thức 2.14.

$$\text{sim}_{\cos}(x_i, x_j) = \frac{x_i \cdot x_j}{\sqrt{\sum_{k=1}^p x_{ik}^2} \cdot \sqrt{\sum_{k=1}^p x_{jk}^2}} \quad (2.14)$$

Trong đó:

- $x_i \cdot x_j$ là tích vô hướng giữa hai vector x_i và x_j , được định nghĩa bằng Công thức 2.15.

$$x_i \cdot x_j = \sum_{k=1}^p x_{ik} \cdot x_{jk} \quad (2.15)$$

- $\sqrt{\sum_{k=1}^p x_{ik}^2}$ và $\sqrt{\sum_{k=1}^p x_{jk}^2}$ là định chuẩn L_2 (L_2 -norm) của các vector x_i và x_j , tương ứng.

Độ tương tự cosine nhận giá trị trong khoảng $[-1, 1]$, với:

- Giá trị 1 biểu thị hai vector cùng hướng,
- Giá trị -1 biểu thị hai vector ngược hướng,
- Giá trị 0 tương ứng với hai vector vuông góc (không tương quan).

Đây là một độ đo đặc biệt hữu ích trong các bài toán học máy, nơi *hướng của vector* quan trọng hơn *độ lớn tuyệt đối* của chúng. Độ tương tự cosine thường được áp dụng trong các mô hình biểu diễn văn bản như mô hình túi từ (*bag-of-words*) và vector nhúng từ (*word embeddings*) [28].

Tuy nhiên, độ tương tự cosine không được sử dụng trực tiếp như một khoảng cách. Thay vào đó, khoảng cách cosine được định nghĩa như sau để chuyển đổi độ tương tự cosine thành một độ đo khoảng cách:

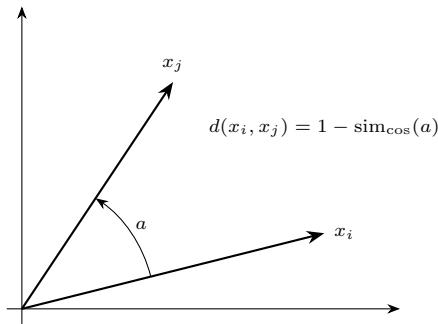
$$d(x_i, x_j) = 1 - \text{sim}_{\cos}(x_i, x_j) \quad (2.16)$$

Khoảng cách cosine có giá trị nằm trong khoảng $[0, 2]$, với giá trị 0 khi hai vector cùng hướng và giá trị 2 khi hai vector ngược hướng hoàn toàn. Hình 2.4 minh họa trực quan khoảng cách cosine giữa hai vector.

Mặc dù khoảng cách cosine được sử dụng rộng rãi trong thực tiễn, nó không phải là một hàm khoảng cách đúng nghĩa, do không thỏa mãn *bất đẳng thức tam giác*. Điều này có thể gây ra hạn chế trong các bài toán yêu cầu tính chất hàm khoảng cách nghiêm ngặt, chẳng hạn như trong một số thuật toán phân cụm hoặc tối ưu hóa [18].

Ngoài ra, khoảng cách cosine giả định rằng các vector đầu vào đã được chuẩn hóa, và kết quả có thể bị sai lệch nếu các vector có độ dài chênh lệch đáng kể. Để khắc phục điều này, người ta thường sử dụng một biến thể gọi là khoảng cách góc (*angular distance*), có khả năng phản ánh tốt hơn mối quan hệ về phương pháp giữa các vector.

2.3. KHOẢNG CÁCH GIỮA CÁC ĐIỂM DỮ LIỆU



Hình 2.4: Minh họa khoảng cách cosine giữa hai vector.

2.3.7 Khoảng cách góc

Để khắc phục các hạn chế của khoảng cách cosine, khoảng cách góc (*angular distance*) được đề xuất như một độ đo thay thế, đảm bảo tính chất hàm khoảng cách trong một số trường hợp cụ thể. Khoảng cách góc được định nghĩa dựa trên góc giữa hai vector, sử dụng hàm ngược của cosine như sau:

$$d(x_i, x_j) = \frac{c \cdot \cos^{-1}(\text{sim}_{\cos}(x_i, x_j))}{\pi} \quad (2.17)$$

Trong đó:

- $\text{sim}_{\cos}(x_i, x_j)$ là độ tương tự cosine, được định nghĩa trong Phương trình 2.14.
- \cos^{-1} là hàm arccosine, trả về góc (tính bằng radian) giữa hai vector.
- c là hằng số điều chỉnh, nhận giá trị:
 - $c = 1$ nếu một trong hai vector có phần tử âm.
 - $c = 2$ nếu cả hai vector đều có các phần tử dương.

Hàng số c được sử dụng để chuẩn hóa khoảng cách góc trong khoảng $[0, 1]$ hoặc $[0, 2]$, tùy thuộc vào tính chất của vector. Khoảng cách góc khắc phục một số hạn chế của khoảng cách cosine bằng cách trực tiếp sử dụng góc giữa hai vector, thay vì chỉ dựa vào giá trị cosine. Điều này làm cho khoảng cách góc phù hợp hơn trong các bài toán yêu cầu độ đo khoảng cách thỏa mãn bất đẳng thức tam giác trong các trường hợp cụ thể [29].

Khoảng cách góc đặc biệt hữu ích trong các ứng dụng liên quan đến dữ liệu hướng, chẳng hạn trong thống kê định hướng hoặc xử lý tín hiệu. Tuy nhiên, việc tính toán hàm arccosine có thể tốn kém về mặt tính toán so với khoảng cách cosine, đặc biệt với các tập dữ liệu lớn.

Ví dụ 2.3. Xét hai vector trong không gian ba chiều: $x = (3, 2, 5)$ và $y = (4, 7, 9)$. Hãy tính khoảng cách cosine và khoảng cách góc giữa hai vector này.

Dầu tiên, ta tính độ tương tự cosine theo công thức 2.14:

$$\begin{aligned}\text{sim}_{\cos}(x, y) &= \frac{x \cdot y}{\sqrt{\sum_{k=1}^p x_k^2} \cdot \sqrt{\sum_{k=1}^p y_k^2}} = \frac{3 \cdot 4 + 2 \cdot 7 + 5 \cdot 9}{\sqrt{3^2 + 2^2 + 5^2} \cdot \sqrt{4^2 + 7^2 + 9^2}} \\ &= \frac{71}{\sqrt{38} \cdot \sqrt{146}} = \frac{71}{\sqrt{5548}} \approx 0.953\end{aligned}$$

Ta tính khoảng cách cosine theo công thức 2.16:

$$d(x, y) = 1 - \text{sim}_{\cos}(x, y) = 1 - 0.953 \approx 0.047$$

Ta tính khoảng cách góc theo công thức 2.17:

$$d(x, y) = \frac{c \cdot \cos^{-1}(\text{sim}_{\cos}(x, y))}{\pi} = \frac{c \cdot \cos^{-1}(0.953)}{\pi} \approx 0.098 \cdot c$$

Trong đó, c là hằng số điều chỉnh.

2.4 Thuật toán

2.4.1 Các bước tiến hành thuật toán

Thuật toán k-NN là một phương pháp học máy không tham số, được sử dụng rộng rãi trong các bài toán phân loại và hồi quy nhờ tính đơn giản và hiệu quả của nó. Thuật toán hoạt động dựa trên nguyên lý rằng các điểm dữ liệu tương tự nhau về mặt đặc trưng thường thuộc cùng một lớp hoặc có giá trị gần nhau. Quy trình thực hiện thuật toán k-NN bao gồm các bước sau:

- Chuẩn bị và tiền xử lý dữ liệu:** Sử dụng một tập dữ liệu huấn luyện đã được gán nhãn, ký hiệu là $D = \{(x_i, y_i)\}_{i=1}^n$, trong đó $x_i \in \mathbb{R}^p$ là vector đặc trưng của điểm dữ liệu thứ i , và y_i là nhãn lớp (trong bài toán phân loại) hoặc giá trị thực (trong bài toán hồi quy). Điểm dữ liệu mới cần được dự đoán được ký hiệu là A .

Để đảm bảo tính công bằng trong việc tính toán khoảng cách, các đặc trưng cần được chuẩn hóa (*normalized*) nhằm đưa các giá trị về cùng một thang đo, tránh hiện tượng các đặc trưng có giá trị lớn chi phối kết quả. Hai phương pháp phổ biến là:

- Chuẩn hóa Min-Max (*Min-Max normalization*): chuyển các giá trị về đoạn $[0, 1]$ theo công thức

$$x' = \frac{x - \min(x)}{\max(x) - \min(x)}. \quad (2.18)$$

Trong đó

- x : giá trị gốc (giá trị ban đầu) của thuộc tính cần được chuẩn hóa.
- $\min(x)$: giá trị nhỏ nhất của thuộc tính trong toàn bộ tập dữ liệu.
- $\max(x)$: giá trị lớn nhất của thuộc tính trong tập dữ liệu.

2.4. THUẬT TOÁN

- Chuẩn hóa Z-score (*Z-score normalization*): đưa dữ liệu về phân phối chuẩn với trung bình bằng 0 và độ lệch chuẩn bằng 1, theo công thức

$$x' = \frac{x - \mu}{\sigma} \quad (2.19)$$

trong đó μ là giá trị trung bình và σ là độ lệch chuẩn.

Việc chuẩn hóa này đặc biệt quan trọng khi các đặc trưng có đơn vị đo hoặc thang đo khác nhau [30].

2. **Lựa chọn tham số k :** Xác định số lượng láng giềng gần nhất k , là một số nguyên dương, thường được chọn dựa trên đặc điểm của tập dữ liệu và yêu cầu bài toán. Giá trị k ảnh hưởng lớn đến hiệu suất của thuật toán: một giá trị k quá nhỏ có thể dẫn đến hiện tượng quá khớp (*overfitting*), trong khi k quá lớn có thể làm mờ ranh giới giữa các lớp, gây ra dưới khớp (*underfitting*) [25]. Trong thực tế, k thường được chọn thông qua các phương pháp như kiểm định chéo để tối ưu hóa hiệu suất trên tập kiểm tra. Một cách tiếp cận phổ biến là thử các giá trị k lẻ (ví dụ: 3, 5, 7) để tránh trường hợp hòa phiếu trong bài toán phân loại.
3. **Tính toán khoảng cách:** Đo lường khoảng cách từ điểm dữ liệu mới A đến tất cả các điểm x_i trong tập dữ liệu huấn luyện D , sử dụng một độ đo khoảng cách phù hợp. Các độ đo phổ biến bao gồm khoảng cách Euclid, được định nghĩa là $d(x_i, A) = \sqrt{\sum_{j=1}^p (x_{ij} - A_j)^2}$, phù hợp cho dữ liệu liên tục, và khoảng cách Manhattan, được định nghĩa là $d(x_i, A) = \sum_{j=1}^p |x_{ij} - A_j|$, phù hợp cho dữ liệu có đặc trưng rời rạc hoặc nhạy cảm với giá trị ngoại lai [18]. Ngoài ra, các độ đo khác như khoảng cách Hamming có thể được sử dụng cho dữ liệu nhị phân hoặc danh mục. Sau khi tính toán, chọn k điểm dữ liệu có khoảng cách nhỏ nhất đến A , được gọi là tập k-NN.

4. Dự đoán nhãn:

- Trong bài toán phân loại, nhãn của điểm dữ liệu A được xác định thông qua phương pháp bỏ phiếu đa số, tức là chọn nhãn xuất hiện nhiều nhất trong số k láng giềng gần nhất. Trong một số trường hợp, có thể sử dụng bỏ phiếu có trọng số, trong đó mỗi láng giềng được gán trọng số dựa trên khoảng cách (ví dụ: trọng số nghịch đảo khoảng cách, $w_i = 1/d(x_i, A)$), để ưu tiên các láng giềng gần hơn [31].
- Trong bài toán hồi quy, giá trị dự đoán cho A được tính bằng giá trị trung bình (hoặc đôi khi trung vị) của các nhãn y_i của k láng giềng gần nhất, theo công thức $\hat{y} = \frac{1}{k} \sum_{i \in N_k} y_i$, trong đó N_k là tập các chỉ số của k láng giềng gần nhất. Tương tự, có thể áp dụng trọng số dựa trên khoảng cách để cải thiện độ chính xác.

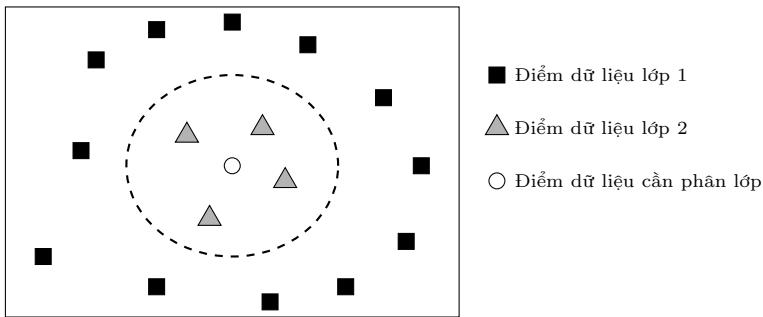
5. **Gán nhãn hoặc giá trị dự đoán:** Gán nhãn (trong bài toán phân loại) hoặc giá trị số (trong bài toán hồi quy) thu được từ bước trước cho điểm dữ liệu mới A . Kết quả này đại diện cho dự đoán cuối cùng của thuật toán k-NN.

Quy trình trên thể hiện tính linh hoạt và đơn giản của thuật toán k-NN, nhưng cũng đi kèm với một số hạn chế. Thuật toán yêu cầu lưu trữ toàn bộ tập dữ liệu huấn luyện, dẫn đến chi phí bộ nhớ và tính toán cao, đặc biệt với các tập dữ liệu lớn. Ngoài ra, việc lựa chọn độ đo khoảng cách và tham số k đòi hỏi sự cân nhắc kỹ lưỡng để đảm bảo hiệu suất tối ưu [25]. Tuy nhiên, nhờ tính không tham số và khả năng thích ứng với nhiều loại dữ liệu, k-NN vẫn là một trong những phương pháp học máy phổ biến và hiệu quả trong nhiều ứng dụng thực tế.

2.4.2 Lựa chọn tham số k

Việc lựa chọn tham số k trong thuật toán k-NN là một yếu tố quan trọng quyết định hiệu suất của mô hình, đặc biệt trong các bài toán phân loại. Tham số k , biểu thị số lượng láng giềng gần nhất được xem xét để dự đoán nhãn của một điểm dữ liệu mới, có ảnh hưởng trực tiếp đến độ chính xác và tính tổng quát của thuật toán [25]. Một trong những thách thức lớn của thuật toán k-NN là khi tập dữ liệu có sự mất cân bằng về số lượng phần tử giữa các lớp, dẫn đến hiện tượng thiên lệch trong dự đoán nhãn.

Để minh họa, giả sử ta có một tập dữ liệu với sự phân phối nhãn không đồng đều, như được thể hiện trong Hình 2.5.



Hình 2.5: Ánh hưởng của phân phối nhãn khi chọn tham số k trong k-NN

Trong trường hợp này, điểm dữ liệu chưa được gán nhãn nằm trong khu vực chủ yếu bao quanh bởi các điểm thuộc lớp 2. Tuy nhiên, do số lượng điểm dữ liệu thuộc lớp 1 chiếm ưu thế trong tập dữ liệu, việc chọn một giá trị k quá lớn (chẳng hạn, $k > 8$) có thể khiến thuật toán ưu tiên nhãn của lớp chiếm đa số (lớp 1), dẫn đến việc điểm dữ liệu bị gán nhãn sai, bất kể vị trí địa lý thực sự của nó trong không gian đặc trưng. Hiện tượng này xảy ra vì khi k tăng, vùng láng giềng được xem xét trở nên rộng hơn, làm tăng khả năng bao gồm các điểm từ lớp chiếm ưu thế, ngay cả khi chúng ở xa điểm cần dự đoán [17].

Ánh hưởng của tham số k đến hiệu suất của thuật toán có thể được phân tích qua hai khía cạnh:

- k quá nhỏ:** Khi chọn một giá trị k nhỏ, ví dụ $k = 1$, thuật toán chỉ xem xét láng giềng gần nhất, dẫn đến nguy cơ bị ảnh hưởng bởi nhiều trong dữ liệu. Điều này có thể gây ra hiện tượng quá khớp (*overfitting*), khi mô hình quá nhạy với các biến động ngẫu nhiên trong tập dữ liệu huấn luyện, làm giảm độ chính xác của dự đoán trên các dữ liệu mới [25]. Ví dụ, nếu một điểm dữ liệu nhiễu thuộc lớp thiểu số nằm gần điểm cần dự đoán, việc chọn $k = 1$ sẽ dẫn đến dự đoán sai lệch.
- k quá lớn:** Ngược lại, khi k quá lớn, thuật toán có xu hướng ưu tiên nhãn của lớp chiếm đa số trong tập dữ liệu, đặc biệt trong các tập dữ liệu mất cân bằng. Điều này làm giảm khả năng phát hiện các mẫu thuộc lớp thiểu số, dẫn đến hiện tượng dưới khớp (*underfitting*), khi mô hình không nắm bắt được các đặc điểm phức tạp của dữ liệu [30]. Trong ví dụ ở Hình 2.5, một giá trị k lớn khiến điểm dữ liệu bị gán nhãn màu xanh dù nằm trong vùng chủ yếu của lớp màu đỏ.

Do đó, việc lựa chọn giá trị k tối ưu là một nhiệm vụ quan trọng và phụ thuộc vào đặc điểm cụ thể của tập dữ liệu. Một cách tiếp cận phổ biến để xác định k là sử dụng *kiểm định chéo*,

2.4. THUẬT TOÁN

chẳng hạn như *kiểm định chéo k-fold*, trong đó tập dữ liệu được chia thành các tập con để đánh giá hiệu suất của mô hình với các giá trị k khác nhau [32]. Độ chính xác hoặc các chỉ số hiệu suất như *F1-score* hoặc độ chính xác trung bình được tính toán để chọn giá trị k tối ưu. Ngoài ra, các giá trị k lẻ (ví dụ: 3, 5, 7) thường được ưu tiên trong bài toán *phân loại* để tránh trường hợp hòa phiếu trong phương pháp *bỏ phiếu đa số*. Một số nghiên cứu cũng đề xuất sử dụng *bỏ phiếu có trọng số*, trong đó các láng giềng gần hơn được gán trọng số cao hơn (ví dụ: $w_i = 1/d(x_i, A)$), để cải thiện độ chính xác khi các láng giềng có mức độ liên quan khác nhau [31].

Việc lựa chọn k không chỉ là một bài toán kỹ thuật mà còn đòi hỏi sự hiểu biết sâu sắc về cấu trúc dữ liệu và mục tiêu của bài toán, đảm bảo cân bằng giữa độ chính xác và tính tổng quát của mô hình.

2.4.3 Trọng số trong k-NN

Trong thuật toán k-NN, việc lựa chọn giá trị k tối ưu là yếu tố quan trọng để đảm bảo hiệu suất mô hình, nhưng một phương pháp khác để nâng cao độ chính xác là sử dụng trọng số (*weights*) cho các điểm dữ liệu trong tập k điểm gần nhất. Trong phiên bản cơ bản của thuật toán, mỗi điểm láng giềng gần nhất được xem xét với mức độ ảnh hưởng đồng đều, tức là tất cả các điểm đều có trọng số bằng nhau khi thực hiện bỏ phiếu đa số trong bài toán phân loại hoặc tính trung bình trong bài toán hồi quy. Tuy nhiên, cách tiếp cận này có thể không tối ưu, đặc biệt khi các điểm láng giềng có mức độ liên quan khác nhau đến điểm dữ liệu cần dự đoán. Việc gán trọng số dựa trên khoảng cách cho phép ưu tiên các điểm láng giềng gần hơn, từ đó cải thiện độ chính xác của dự đoán [31].

Một phương pháp phổ biến để tính trọng số là sử dụng nghịch đảo khoảng cách, trong đó trọng số của điểm láng giềng thứ i được xác định bởi công thức 2.20:

$$\omega_i = \frac{1}{d(x_i, A) + \epsilon} \quad (2.20)$$

Trong đó,

- ω_i là *trọng số* của điểm láng giềng x_i ,
- $d(x_i, A)$ là khoảng cách giữa điểm dữ liệu mới A và điểm láng giềng x_i , thường được tính bằng khoảng cách Euclid hoặc khoảng cách Manhattan,
- hằng số $\epsilon > 0$ (thường là một giá trị rất nhỏ, ví dụ $\epsilon = 10^{-6}$) được thêm vào mẫu số để tránh trường hợp chia cho 0 khi $d(x_i, A) = 0$

Phương pháp này đảm bảo rằng các điểm láng giềng gần hơn có ảnh hưởng lớn hơn đến dự đoán, vì trọng số giảm tỷ lệ nghịch với khoảng cách. Ví dụ, trong bài toán phân loại, nhãn của điểm A được xác định bằng cách tính tổng trọng số của các láng giềng thuộc cùng lớp và chọn lớp có tổng trọng số cao nhất [31].

Một cách tiếp cận khác là sử dụng hàm số mũ để tính trọng số, được định nghĩa trong công thức 2.21:

$$\omega_i = e^{-\frac{d(x_i, A)^2}{\sigma^2}} \quad (2.21)$$

Trong đó, σ là một hằng số dương điều chỉnh mức độ nhạy của trọng số đối với khoảng cách. Giá trị σ lớn sẽ làm cho trọng số giảm chậm hơn khi khoảng cách tăng, trong khi σ nhỏ khiến trọng số giảm nhanh, ưu tiên mạnh mẽ hơn cho các láng giềng rất gần. Phương pháp này có ưu điểm là tạo ra sự phân biệt mượt mà hơn giữa các láng giềng, tránh hiện tượng thay đổi đột ngột trong trọng số như ở phương pháp nghịch đảo khoảng cách [33]. Ví dụ, nếu $\sigma = 1$ và khoảng

cách $d(x_i, A) = 2$, thì $\omega_i = e^{-4} \approx 0.018$, cho thấy điểm láng giềng này có ảnh hưởng rất nhỏ so với các điểm gần hơn.

Cả hai phương pháp trên đều nhằm mục đích cải thiện độ chính xác bằng cách phản ánh mức độ liên quan của các láng giềng dựa trên khoảng cách. Tuy nhiên, chúng cũng có những ưu và nhược điểm riêng. Phương pháp nghịch đảo khoảng cách đơn giản và dễ triển khai, nhưng có thể nhạy cảm với các giá trị khoảng cách rất nhỏ, dẫn đến trọng số lớn bất thường cho các điểm quá gần. Trong khi đó, phương pháp hàm số mũ cho phép điều chỉnh linh hoạt thông qua tham số σ , nhưng việc chọn giá trị σ tối ưu lại đòi hỏi thử nghiệm hoặc kiểm định chéo [32]. Ngoài ra, các phương pháp trọng số khác, như trọng số tuyến tính hoặc trọng số dựa trên thứ hạng cũng có thể được sử dụng tùy thuộc vào đặc điểm của bài toán [33].

Việc áp dụng trọng số trong thuật toán k-NN đặc biệt hữu ích trong các tập dữ liệu có mật độ phân bố không đồng đều hoặc khi các láng giềng gần nhất có mức độ ảnh hưởng khác nhau đáng kể. Tuy nhiên, cần lưu ý rằng việc sử dụng trọng số có thể làm tăng chi phí tính toán, đặc biệt với các tập dữ liệu lớn, do yêu cầu tính toán và lưu trữ trọng số cho từng láng giềng [25]. Do đó, việc lựa chọn phương pháp trọng số cần được cân nhắc dựa trên đặc điểm của tập dữ liệu và yêu cầu cụ thể của bài toán.

Ví dụ 2.4. Một cửa hàng quần áo trên sàn thương mại điện chuyên bán các loại áo với ba kích cỡ cơ bản: *S* (*Small*), *M* (*Medium*), và *L* (*Large*). Qua thời gian hoạt động, cửa hàng đã thu thập thông tin của 10 khách hàng trước đó, bao gồm *chiều cao* (đơn vị: cm), *cân nặng* (đơn vị: kg) và *kích cỡ áo* mà mỗi khách hàng đã thử và lựa chọn thành công. Thông tin được cho trong Bảng 2.1.

Bảng 2.1: Dữ liệu khách hàng: *chiều cao* (*height*), *cân nặng* (*weight*), và *kích cỡ áo* (*shirt size*).

Khách hàng	Chiều cao (cm)	Cân nặng (kg)	Kích cỡ áo
A	160	52	M
B	150	45	S
C	170	58	M
D	177	65	L
E	173	58	L
F	144	57	M
G	166	61	M
H	150	52	S
I	155	54	S
J	182	68	L

Bây giờ, một khách hàng mới *P* đang đặt hàng online và cung cấp thông tin cá nhân như sau:

- Chiều cao: 158 cm
- Cân nặng: 60 kg

Tuy nhiên, khách hàng không chắc chắn nên chọn kích cỡ nào. Thay vì chọn ngẫu nhiên hoặc phải thử nhiều lần, nhân viên bán hàng muốn tận dụng dữ liệu từ các khách hàng cũ để tư vấn kích cỡ phù hợp một cách thông minh. Họ sử dụng thuật toán k-NN để tìm ra những khách hàng có chiều cao và cân nặng gần giống nhất, rồi dựa trên kích cỡ của họ để đưa ra dự đoán.

Trong ví dụ này, ta áp dụng thuật toán k-NN với $k = 3$ và sử dụng *khoảng cách Euclid* để xác định 3 hàng xóm gần nhất. Sau đó, kích cỡ được gọi ý sẽ là nhãn phổ biến nhất trong số các hàng xóm này.

2.4. THUẬT TOÁN

Bước 1: Chuẩn hóa dữ liệu

Do *chiều cao* (cm) và *cân nặng* (kg) có thang đo khác nhau, việc tính toán khoảng cách trực tiếp có thể gây sai lệch, bởi vì đặc trưng có giá trị lớn hơn (*chiều cao*) sẽ chi phối kết quả. Vì vậy, ta áp dụng chuẩn hóa Min-Max (đã trình bày trong công thức (2.18)) để đưa tất cả đặc trưng về cùng một thang đo trong khoảng [0, 1].

Với tập dữ liệu đang xét, giá trị tối thiểu và tối đa cho từng đặc trưng được xác định như sau:

- *Chiều cao*: min = 144, max = 182,
- *Cân nặng*: min = 45, max = 68.

Áp dụng công thức chuẩn hóa (2.18) vào từng đặc trưng, ta thu được biểu thức cụ thể cho bài toán:

- Đối với *chiều cao*:

$$x_{\text{norm}} = \frac{x - 144}{182 - 144} = \frac{x - 144}{38}$$

- Đối với *cân nặng*:

$$y_{\text{norm}} = \frac{y - 45}{68 - 45} = \frac{y - 45}{23}$$

Khách hàng mới $P = (158, 60)$ sẽ được chuyển thành một vector chuẩn hóa trước khi tính khoảng cách với các điểm dữ liệu huấn luyện.

$$\text{Chiều cao}_{\text{norm}} = \frac{158 - 144}{38} = \frac{14}{38} \approx 0.3684$$

$$\text{Cân nặng}_{\text{norm}} = \frac{60 - 45}{23} = \frac{15}{23} \approx 0.6522$$

Vậy, $P_{\text{norm}} = (0.3684, 0.6522)$.

Tập dữ liệu sau khi chuẩn hóa được trình bày trong Bảng 2.2.

Bảng 2.2: Tập dữ liệu huấn luyện sau khi chuẩn hóa (*normalization*).

Khách hàng	Tọa độ chuẩn hóa	Chiều cao	Cân nặng	Kích cỡ áo
A	(0.4211, 0.3043)	0.4211	0.3043	M
B	(0.1579, 0.0000)	0.1579	0.0000	S
C	(0.6842, 0.5652)	0.6842	0.5652	M
D	(0.8684, 0.8696)	0.8684	0.8696	L
E	(0.7632, 0.5652)	0.7632	0.5652	L
F	(0.0000, 0.5217)	0.0000	0.5217	M
G	(0.5789, 0.6957)	0.5789	0.6957	M
H	(0.1579, 0.3043)	0.1579	0.3043	S
I	(0.2895, 0.3913)	0.2895	0.3913	S
J	(1.0000, 1.0000)	1.0000	1.0000	L

Bước 2: Tính khoảng cách Euclid

Trong thuật toán k -NN, việc tính khoảng cách Euclid giữa điểm dữ liệu mới và các điểm trong tập huấn luyện là bước quan trọng để xác định k láng giềng gần nhất, từ đó dự đoán nhãn hoặc giá trị. Khoảng cách Euclid giữa hai điểm (x_1, y_1) và (x_2, y_2) trong không gian hai chiều được tính theo công thức (2.22).

$$d = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2} \quad (2.22)$$

Để dự đoán nhãn cho điểm dữ liệu chuẩn hóa $P_{\text{norm}} = (0.3684, 0.6522)$, ta tính khoảng cách Euclid từ P_{norm} đến từng điểm trong tập dữ liệu huấn luyện đã chuẩn hóa

- Khách hàng A: $(0.4211, 0.3043)$, nhãn M:

$$d = \sqrt{(0.3684 - 0.4211)^2 + (0.6522 - 0.3043)^2} = \sqrt{0.0028 + 0.1212} \approx 0.3528$$

Khoảng cách từ P_{norm} đến các khách hàng còn lại được tính tương tự. Kết quả được tổng hợp trong Bảng 2.3:

Bảng 2.3: Khoảng cách từ $P_{\text{norm}} = (0.3684, 0.6522)$ đến các khách hàng trong tập dữ liệu chuẩn hóa.

Khách hàng	Tọa độ chuẩn hóa	Nhân	Khoảng cách
A	$(0.4211, 0.3043)$	M	0.3528
B	$(0.1579, 0.0000)$	S	0.6855
C	$(0.6842, 0.5652)$	M	0.3276
D	$(0.8684, 0.8696)$	L	0.5455
E	$(0.7632, 0.5652)$	L	0.4045
F	$(0.0000, 0.5217)$	M	0.3905
G	$(0.5789, 0.6957)$	M	0.2152
H	$(0.1579, 0.3043)$	S	0.4068
I	$(0.2895, 0.3913)$	S	0.2726
J	$(1.0000, 1.0000)$	L	0.7210

Chọn $k = 3$ láng giềng gần nhất: khách hàng G (M), khách hàng I (S), và khách hàng C (M). Trong 3 láng giềng, có 2 nhãn M và 1 nhãn S. Do đó, nhãn được gán cho khách hàng P là M.

Bước 3: Sử dụng trọng số

Để cải thiện độ chính xác, ta có thể sử dụng trọng số để đánh giá độ quan trọng của các điểm láng giềng dựa trên nghịch đảo khoảng cách:

$$\omega_i = \frac{1}{d_i}$$

Tính trọng số cho 3 láng giềng gần nhất:

- Khách hàng G ($d = 0.2152$, nhãn M): $\omega_G = \frac{1}{0.2152} \approx 4.6468$

2.5. ƯU ĐIỂM VÀ NHƯỢC ĐIỂM CỦA K-NN

- Khách hàng I ($d = 0.2726$, nhãn S): $\omega_I = \frac{1}{0.2726} \approx 3.6684$
- Khách hàng C ($d = 0.3276$, nhãn M): $\omega_C = \frac{1}{0.3276} \approx 3.0525$

Tổng trọng số theo nhãn:

$$N_M = \omega_G + \omega_C = 4.6468 + 3.0525 = 7.6993$$

$$N_S = \omega_I = 3.6684$$

Vì $N_M > N_S$, nhãn được gán cho khách hàng P vẫn là **M**.

Sau khi chuẩn hóa dữ liệu và áp dụng thuật toán k-NN với $k = 3$, cả hai phương pháp: bỏ phiếu đa số và bỏ phiếu có trọng số theo khoảng cách đều dự đoán kích cỡ áo phù hợp cho khách hàng P (cao 158 cm, nặng 60 kg) là **M**.

Việc chuẩn hóa giúp đưa các đặc trưng như *chiều cao* và *cân nặng* về cùng một thang đo, từ đó đảm bảo rằng không đặc trưng nào chiếm ưu thế trong quá trình tính khoảng cách. Nhờ đó, mô hình đưa ra dự đoán công bằng và chính xác hơn, đặc biệt quan trọng trong những bài toán mà các đặc trưng có đơn vị hoặc phạm vi khác nhau.

2.5 Ưu điểm và nhược điểm của k-NN

Mặc dù k-NN là một thuật toán đơn giản và dễ triển khai, nhưng nó cũng đi kèm với một số hạn chế. Dưới đây là phân tích các ưu điểm và nhược điểm tiêu biểu của k-NN.

2.5.1 Ưu điểm

- **Tính đơn giản và dễ triển khai:** Thuật toán k-NN nổi bật nhờ sự đơn giản trong cả lý thuyết và triển khai. Không yêu cầu các bước tối ưu hóa phức tạp như trong các phương pháp học máy tham số, thuật toán k-NN chỉ cần thực hiện ba bước cơ bản: tính toán *khoảng cách* giữa các điểm dữ liệu, xác định k điểm láng giềng gần nhất, và dự đoán dựa trên *bỏ phiếu đa số* hoặc tính trung bình [25]. Sự đơn giản này khiến thuật toán k-NN trở thành lựa chọn lý tưởng cho các bài toán phân loại và hồi quy cơ bản, đặc biệt phù hợp với những người mới học về học máy hoặc khi cần nhanh chóng triển khai một mô hình dự đoán mà không đòi hỏi kiến thức chuyên sâu về tối ưu hóa.
- **Không yêu cầu huấn luyện trước:** Không giống như các thuật toán học máy khác đòi hỏi giai đoạn huấn luyện phức tạp để tối ưu hóa tham số, thuật toán k-NN là một phương pháp học lười (*lazy learning*), trong đó toàn bộ tập dữ liệu huấn luyện được lưu trữ và các phép tính chỉ được thực hiện khi dự đoán cho một điểm dữ liệu mới [34]. Điều này loại bỏ nhu cầu xây dựng mô hình trước, giúp tiết kiệm thời gian trong các ứng dụng mà tập dữ liệu không quá lớn hoặc khi cần triển khai nhanh chóng. Tuy nhiên, điều này cũng đồng nghĩa với việc toàn bộ gánh nặng tính toán được chuyển sang giai đoạn dự đoán.

2.5.2 Nhược điểm

- **Hiệu suất tính toán kém với tập dữ liệu lớn:** Một hạn chế lớn của thuật toán k-NN là chi phí tính toán cao khi xử lý các tập dữ liệu lớn. Vì thuật toán yêu cầu tính *khoảng cách* giữa điểm dữ liệu mới và tất cả các điểm trong tập huấn luyện, độ phức tạp tính toán của nó là $O(n \cdot p)$, trong đó n là số lượng điểm dữ liệu và p là số chiều đặc trưng [35]. Điều

này khiến thuật toán k-NN trở nên kém hiệu quả trong các bài toán với hàng triệu điểm dữ liệu, dẫn đến thời gian dự đoán lâu và yêu cầu tài nguyên bộ nhớ lớn để lưu trữ tập huấn luyện. Các kỹ thuật như *KD-tree* [35] hoặc *Ball-tree* [36] có thể được sử dụng để giảm độ phức tạp, nhưng hiệu quả của chúng giảm đáng kể khi số chiều tăng cao.

- **Không xem xét tầm quan trọng của các đặc trưng:** Thuật toán k-NN giả định rằng tất cả các đặc trưng trong tập dữ liệu có mức độ quan trọng như nhau khi tính *khoảng cách*, điều này có thể dẫn đến kết quả không chính xác trong các bài toán thực tế, nơi các đặc trưng thường có mức độ ảnh hưởng khác nhau [30]. Ví dụ, trong một bài toán chẩn đoán y tế, các đặc trưng như huyết áp hoặc nhịp tim có thể quan trọng hơn nhiều so với các đặc trưng như tuổi hoặc giới tính, nhưng thuật toán k-NN không có cơ chế tự động đánh giá và điều chỉnh trọng số của các đặc trưng. Sự thiếu sót này có thể làm giảm độ chính xác, đặc biệt khi tập dữ liệu chứa các đặc trưng không liên quan hoặc nhiễu. Để khắc phục, các kỹ thuật như *lựa chọn đặc trưng* hoặc *chuẩn hóa đặc trưng* cần được áp dụng trước khi sử dụng thuật toán [37].
- **Hiệu suất giảm trong không gian nhiều chiều:** Thuật toán k-NN gặp phải vấn đề “*lời nguyền của chiều*”, trong đó hiệu suất của thuật toán giảm đáng kể khi số chiều của dữ liệu tăng [38]. Khi số chiều tăng, *khoảng cách* giữa các điểm dữ liệu trở nên đồng đều hơn, làm cho việc phân biệt giữa các láng giềng gần và xa trở nên khó khăn. Điều này dẫn đến việc các dự đoán trở nên kém chính xác, đặc biệt trong các bài toán với hàng chục hoặc hàng trăm đặc trưng, như trong xử lý hình ảnh hoặc văn bản. Để giải quyết vấn đề này, các phương pháp *giảm chiều* như *Phân tích thành phần chính*, thường được sử dụng để giảm số chiều trước khi áp dụng thuật toán [32].
- **Thách thức trong việc lựa chọn giá trị k :** Việc chọn giá trị k tối ưu là một thách thức lớn trong thuật toán k-NN. Một giá trị k quá nhỏ (ví dụ, $k = 1$) có thể khiến mô hình nhạy cảm với nhiễu, dẫn đến quá khớp (*overfitting*), trong khi một giá trị k quá lớn có thể làm mờ ranh giới giữa các lớp, gây ra dưới khớp (*underfitting*) [25]. Ví dụ, trong một tập dữ liệu mất cân bằng, một giá trị k lớn có thể ưu tiên lớp chiếm đa số, làm giảm khả năng phát hiện các mẫu thuộc lớp thiểu số. Việc xác định k tối ưu thường yêu cầu sử dụng *kiểm định chéo*, chẳng hạn như kiểm định chéo k-fold, để đánh giá hiệu suất với các giá trị k khác nhau [32]. Điều này làm tăng chi phí tính toán và đòi hỏi sự cân nhắc kỹ lưỡng về đặc điểm của tập dữ liệu.

2.6 Bài tập

Bài tập 2.1. Tính khoảng cách Euclid, khoảng cách Manhattan và khoảng cách cosine cho các điểm sau:

- $x = (3, 4, 7, 1)$ và $y = (2, 5, 1, 8)$
- $x = (2, -6, -8, 1)$ và $y = (0, -3, -9, 0)$

Bài tập 2.2. Một ngân hàng đang cho vay đối với một số đối tượng. Tuy nhiên, ngân hàng cần xét xem hồ sơ nào có đủ khả năng chi trả và ra quyết định cho vay. Bảng 2.4 mô tả số liệu về độ tuổi, thu nhập và khả năng chi trả.

Yêu cầu:

- Sử dụng thuật toán k-NN với $k = 3$ và khoảng cách Euclid, hãy dự đoán xem một người 24 tuổi, thu nhập 8 triệu đồng, vay 250 triệu đồng có khả năng chi trả không?

2.6. BÀI TẬP

Bảng 2.4: Dữ liệu khách hàng

Khách hàng	Độ tuổi	Thu nhập (triệu đồng)	Số tiền vay (trăm triệu đồng)	Khả năng chi trả
1	18	0	1.0	Không
2	21	5	2.0	Không
3	22	10	1.5	Có
4	20	7	0.7	Không
5	28	20	2.0	Có
6	24	4	1.0	Không
7	26	9	1.2	Có
8	23	17	4.0	Có
9	22	18	5.0	Có
10	30	28	7.0	Có

- b) Có thể áp dụng thuật toán k-NN nếu bảng số liệu có thêm thuộc tính giới tính không? Nếu có, áp dụng thế nào? Nếu không, giải thích lý do.
- c) Vì sao không lấy đơn vị của số tiền vay là triệu đồng? Có cách nào giữ nguyên đơn vị độ tuổi và giới tính mà vẫn tăng độ chính xác không?

Bài tập 2.3. Bộ dữ liệu được miêu tả trong Bảng 2.5. Yêu cầu đề xuất một phương pháp sử dụng thuật toán k-NN để phát hiện điểm bất thường và áp dụng phương pháp này cho bộ dữ liệu.

Bảng 2.5: Bộ dữ liệu điểm bất thường với ba thuộc tính số.

Điểm	Thuộc tính 1	Thuộc tính 2	Thuộc tính 3
A	4.1	0.8	4.2
B	4.0	4.8	6.2
C	3.0	5.0	5.2
D	3.3	6.2	6.3
E	1.1	2.4	2.2
F	2.0	2.5	3.2
G	1.2	3.5	2.6

Chương 3

Cây quyết định và Rừng ngẫu nhiên

3.1 Cây quyết định

3.1.1 Bài toán mở đầu

Một công ty xây dựng phần mềm dự đoán giới tính (Nam hoặc Nữ) của khách hàng dựa trên sở thích cá nhân. Dữ liệu thu thập bao gồm hai đặc điểm: khách hàng có xem *trình diễn ca nhạc* và *trao cúp trong giải bóng đá*, cùng với nhãn *giới tính*. Mục tiêu là xác định thuộc tính nào mang lại khả năng phân loại giới tính tốt nhất, hướng tới ứng dụng thuật toán cây quyết định.

Hãy xem xét tập dữ liệu với 4 người, như được trình bày trong Bảng 3.1.

Bảng 3.1: Dữ liệu sở thích

Người	Trình diễn ca nhạc	Trao cúp bóng đá	Giới tính
1	Có	Có	Nam
2	Không	Có	Nam
3	Có	Không	Nữ
4	Có	Không	Nữ

Để đánh giá khả năng phân loại của từng thuộc tính, dữ liệu được chia thành các nhóm con dựa trên các giá trị của thuộc tính đó và được đánh giá theo mức độ thuần chủng (*purity*). Độ thuần chủng phản ánh mức độ mà tất cả các điểm dữ liệu trong một nhóm thuộc cùng một lớp. Một nhóm hoàn toàn thuần chủng chỉ chứa các điểm dữ liệu của một lớp duy nhất, cho phép dự đoán chính xác mà không cần phân chia thêm. Ngược lại, một nhóm có độ bất thuần chủng (*impurity*) cao khi chứa nhiều lớp với tỷ lệ gần bằng nhau, gây khó khăn trong việc phân lớp do sự pha trộn giữa các lớp. Do đó, các thuộc tính giúp tạo ra các nhóm con có độ thuần chủng cao sẽ có giá trị phân loại tốt hơn và được ưu tiên lựa chọn trong quá trình xây dựng cây quyết định.

Dựa trên tập dữ liệu trong Bảng 3.1, ta tiến hành phân tích hai thuộc tính là *trình diễn ca nhạc* và *trao cúp trong giải bóng đá*, nhằm đánh giá thuộc tính nào mang lại khả năng phân loại giới tính cao hơn.

3.1. CÂY QUYẾT ĐỊNH

- Xét thuộc tính *trình diễn ca nhạc*

- Nhóm “Có”: Gồm 3 người (1 Nam, 2 Nữ) \Rightarrow độ thuần chủng: 66.7%.
- Nhóm “Không”: Gồm 1 người (1 Nam) \Rightarrow độ thuần chủng: 100%.

Kết luận: Việc chia theo thuộc tính này tạo ra một nhóm “Không” thuần chủng và một nhóm nhỏ, không hiệu quả để phân loại.

- Xét thuộc tính *trao cúp trong giải bóng đá*

- Nhóm “Có”: Gồm 2 người (2 Nam) \Rightarrow độ thuần chủng: 100%.
- Nhóm “Không”: Gồm 2 người (2 Nữ) \Rightarrow độ thuần chủng: 100%.

Kết luận: Đây là thuộc tính phân loại tối ưu vì chia thành các nhóm hoàn toàn thuần chủng.

Vì vậy, câu hỏi liên quan đến thuộc tính *chương trình trao cúp bóng đá* mang lại giá trị thông tin cao hơn, từ đó giúp cải thiện khả năng phân loại chính xác nhờ tạo ra các nhóm con có độ thuần chủng cao hơn.

Dể đánh giá mức độ thuần chủng (*purity*) và xác định thuộc tính tối ưu, cần sử dụng một thước đo độ bất thuần chủng (*impurity*), trong đó *entropy* là một chỉ số phổ biến. Entropy phản ánh mức độ hỗn loạn trong một tập dữ liệu.

Dộ thuần chủng thể hiện mức độ mà các điểm dữ liệu trong một nhóm thuộc cùng một lớp. Một nhóm hoàn toàn thuần chủng chỉ chứa các điểm dữ liệu thuộc về một lớp duy nhất, cho phép mô hình dự đoán chính xác mà không cần phân chia thêm. Ngược lại, một nhóm có độ bất thuần chủng cao chứa nhiều lớp với tỷ lệ gần bằng nhau, gây khó khăn cho việc dự đoán.

Dể đánh giá độ thuần chủng một cách tổng quát, thuật toán cây quyết định bắt đầu bằng việc xác định mức độ bất thuần chủng của tập dữ liệu ban đầu. Sau đó, dữ liệu được chia thành các nhóm con dựa trên các giá trị của từng thuộc tính (như *Có* hoặc *Không* trong ví dụ). Mỗi nhóm con được kiểm tra để đánh giá mức độ bất thuần, tức là mức độ phân bố của các lớp trong nhóm đó.

Chẳng hạn, nếu một nhóm con chỉ chứa một lớp (như nhóm *Có* của thuộc tính trao cúp bóng đá chỉ gồm Nam), nhóm đó hoàn toàn thuần chủng. Ngược lại, nếu nhóm con chứa cả Nam và Nữ với tỷ lệ gần bằng nhau (như nhóm *Có* của chương trình trình diễn ca nhạc với 1 Nam và 2 Nữ), thì nhóm đó có độ bất thuần chủng cao hơn.

Thuật toán tiếp tục tính trung bình mức độ bất thuần của các nhóm con, với trọng số dựa trên tỷ lệ số điểm dữ liệu trong mỗi nhóm so với tổng số điểm. Thuộc tính làm giảm độ bất thuần nhiều nhất, tức là tạo ra các nhóm con có độ thuần chủng cao hơn, sẽ được chọn để phân chia, vì nó tối ưu hóa khả năng dự đoán chính xác.

Do đó, việc phân chia theo thuộc tính *chương trình trao cúp bóng đá* dẫn đến độ bất thuần chủng sau phân chia rất thấp, đồng thời độ lợi thông tin (*information gain*) đạt giá trị cao. Đây là cơ sở để thuật toán cây quyết định ưu tiên chọn thuộc tính này làm nút gốc.

Thuật toán cây quyết định sử dụng entropy để đánh giá chất lượng của mỗi phân chia tại từng nút. Cụ thể, thuộc tính nào làm giảm độ bất thuần chủng nhiều nhất, tức là tăng độ thuần chủng của các nhóm con, sẽ được chọn để phân tách dữ liệu. Quá trình phân chia được thực hiện đệ quy, tiếp tục trên từng nhánh con cho đến khi tất cả các nút lá đạt trạng thái thuần chủng (chỉ chứa một lớp) hoặc thỏa mãn điều kiện dừng. Nhờ đó, cây quyết định có thể dự đoán giới tính một cách hiệu quả dựa trên các đặc điểm sở thích cá nhân.

3.1.2 Khái niệm

Cây quyết định (*decision tree*) là một mô hình học có giám sát được phát triển để giải quyết các bài toán phân lớp và hồi quy. Trong đó, ứng dụng trong phân lớp được đánh giá cao hơn nhờ khả năng biểu diễn trực quan các quy tắc ra quyết định và tính dễ diễn giải, phù hợp với nhiều lĩnh vực như y học, tài chính, và khoa học dữ liệu [39], [40]. Tài liệu này tập trung phân tích ứng dụng của cây quyết định trong bài toán phân loại, với mục tiêu gán nhãn cho các điểm dữ liệu dựa trên tập hợp các thuộc tính quan sát được.

Cây quyết định được mô tả dưới dạng một cấu trúc đồ thị hình cây, bao gồm các thành phần cơ bản sau, được nghiên cứu chi tiết trong các tài liệu học máy [41], [42]:

- **Nút lá (*leaf node*):** Là các nút không có nút con, thường được biểu diễn bằng hình tròn trong sơ đồ cây. Nút lá chứa nhãn phân lớp cuối cùng, được xác định thông qua quá trình phân tích đệ quy các thuộc tính từ nút gốc đến nút lá. Nhãn này phản ánh kết quả dự đoán, chẳng hạn như “Nam” hoặc “Nữ”, và thường được quyết định dựa trên chiến lược đa số phiếu hoặc xác suất tối đa của lớp tại nút lá, tùy thuộc vào phương pháp huấn luyện [43].
- **Nút nội bộ (*internal node*):** Là các nút có ít nhất hai nút con, thường được biểu diễn bằng hình chữ nhật. Nút nội bộ đại diện cho một thuộc tính được chọn làm tiêu chí phân chia, đóng vai trò như một điều kiện hoặc ngưỡng để tách tập dữ liệu thành các nhóm con. Thuộc tính này có thể mang dạng rời rạc (ví dụ: màu sắc, trạng thái đẹp/xấu) hoặc liên tục (ví dụ: chiều dài, nhiệt độ), và giá trị của nó định hình cách phân nhánh tiếp theo, ảnh hưởng trực tiếp đến độ thuần chủng của các nhóm [40].
- **Nút gốc (*root node*):** Là nút duy nhất không có nút cha, đóng vai trò khởi đầu của cây quyết định. Nút gốc được chọn dựa trên thuộc tính cung cấp thông tin phân loại hiệu quả nhất, thường được đánh giá thông qua các thước đo định lượng như độ lợi thông tin (*information gain*), chỉ số Gini (*Gini index*), hoặc độ giảm phương sai (*variance reduction*) tùy thuộc vào thuật toán được áp dụng [42].
- **Cành (*branch*):** Là các đoạn thẳng nối các nút, biểu thị giá trị cụ thể của thuộc tính tại nút nội bộ hoặc nút gốc. Các đường dẫn (*path*) từ nút gốc qua các nút nội bộ và cành đến các nút lá tạo thành các quy tắc phân lớp, thể hiện chuỗi các điều kiện logic dẫn đến kết quả dự đoán cuối cùng. Mỗi đường dẫn đại diện cho một kịch bản phân loại dựa trên tập hợp các thuộc tính [43].

Quá trình xây dựng cây quyết định từ một tập dữ liệu huấn luyện là một quá trình đệ quy nhằm xác định các tiêu chí phân tách (câu hỏi) và thứ tự áp dụng chúng so cho tối ưu. Mỗi tiêu chí được xây dựng dựa trên từng thuộc tính đơn lẻ hoặc tổ hợp tuyến tính của các thuộc tính. Tuy nhiên, trong thực tế, việc sử dụng từng thuộc tính riêng lẻ thường được ưu tiên hơn do tính đơn giản trong biểu diễn, hiệu quả tính toán, và khả năng diễn giải rõ ràng đối với con người [44].

- Đối với thuộc tính dạng rời rạc, câu hỏi thường mang hình thức: *Nó thuộc loại nào?*, trong khi với thuộc tính dạng liên tục, câu hỏi thường là: *Nó nằm trong ngưỡng nào?*. Các ngưỡng cho thuộc tính liên tục thường được xác định tự động thông qua các phương pháp như tìm kiếm nhị phân hoặc tối ưu hóa độ thuần chủng [45].
- Thứ tự áp dụng các câu hỏi được sắp xếp dựa trên việc tối ưu hóa các thước đo thông tin, bao gồm entropy - đo độ bất thuần chủng (*impurity*) của một tập dữ liệu, hoặc chỉ số Gini - đo mức độ không đồng nhất giữa các lớp. Thuật toán ID3, do Quinlan phát triển

3.1. CÂY QUYẾT ĐỊNH

vào năm 1986, sử dụng *độ lợi thông tin* (*information gain*) để chọn thuộc tính, trong khi C4.5, phiên bản nâng cấp, hỗ trợ xử lý thuộc tính liên tục, dữ liệu bị thiếu, và áp dụng kỹ thuật cắt tỉa (*pruning*) để giảm nguy cơ quá khớp [40], [42].

- Quá trình xây dựng tiếp tục đệ quy cho đến khi các nút lá đạt độ thuần chủng cao (tức là tất cả mẫu trong nút lá thuộc cùng một lớp) hoặc thỏa mãn các điều kiện dừng, chẳng hạn như độ sâu tối đa của cây, số lượng mẫu tối thiểu tại mỗi nút, hoặc mức độ giảm thông tin dưới một ngưỡng nhất định [43]. Kỹ thuật cắt tỉa sau huấn luyện cũng được áp dụng để cải thiện khả năng tổng quát hóa trên dữ liệu kiểm tra [44].

Sự kết hợp giữa các thuật toán tối ưu hóa và điều kiện dừng giúp cây quyết định không chỉ đạt độ chính xác cao trên tập huấn luyện mà còn duy trì hiệu suất tốt trên dữ liệu chưa từng thấy. Tuy nhiên, cần cân nhắc cẩn thận để tránh hiện tượng quá khớp, đặc biệt trong các trường hợp dữ liệu có số chiều lớn hoặc chứa nhiều nhiễu [25].

3.1.3 Giải thuật ID3

3.1.3.1 Ý tưởng

Giải thuật Iterative Dichotomiser 3 (viết tắt là *ID3*) là một phương pháp học có giám sát thuộc nhóm thuật toán xây dựng cây quyết định (*decision tree learning algorithm*), được đề xuất bởi J. Ross Quinlan vào năm 1986 [42]. Ý tưởng cốt lõi của *ID3* là xây dựng cây phân loại bằng cách lặp đi lặp lại quá trình lựa chọn thuộc tính phân chia “tốt nhất” tại mỗi nút trong cây, nhằm làm giảm độ hỗn loạn (*impurity*) của dữ liệu.

Tại mỗi bước phân chia, *ID3* lựa chọn thuộc tính “tối ưu” nhằm tối đa hóa sự tăng mức độ *thuần chủng*, hoặc, tương đương, tối thiểu hóa mức độ *bất thuần chủng* (*impurity*) trong các tập con sau phân chia [40]. Một tập dữ liệu được coi là *thuần chủng* khi phần lớn các mẫu trong tập thuộc cùng một lớp, phản ánh mức độ đồng nhất cao và độ hỗn loạn thấp. Ngược lại, nếu tập dữ liệu chứa nhiều mẫu thuộc các lớp khác nhau, thì nó được coi là *bất thuần chủng*, phản ánh sự không đồng nhất và độ hỗn loạn cao hơn [43].

Để đo lường và định lượng mức độ *bất thuần chủng* tại mỗi nút, *ID3* sử dụng hàm *entropy*, một khái niệm được giới thiệu trong lý thuyết thông tin của Claude Shannon [46]. Entropy đo lường mức độ không chắc chắn hoặc hỗn loạn của một tập dữ liệu phân lớp. Giá trị của *entropy* càng cao thì sự hỗn loạn trong dữ liệu càng lớn, và ngược lại. Trong bối cảnh của *ID3*, hàm *entropy* đóng vai trò then chốt trong việc định lượng hiệu quả của mỗi thuộc tính. Thuật toán từ đó tính toán *độ lợi thông tin* (*information gain*) - tức là chênh lệch về *entropy* trước và sau khi thực hiện phân chia - để xác định thuộc tính tối ưu tại mỗi bước xây dựng cây quyết định [40], [42].

Quá trình lựa chọn thuộc tính phân chia tại mỗi nút được tiến hành bằng cách tính toán mức độ *lợi thông tin* (*information gain*) - tức là mức độ giảm của *entropy* sau khi phân chia theo thuộc tính đó. Thuộc tính mang lại mức tăng thông tin lớn nhất sẽ được chọn để tạo ra nút phân chia tiếp theo [40], [42]. Cách tiếp cận này đảm bảo rằng cây quyết định được xây dựng sẽ tối ưu hóa độ tinh khiết tại các nút con và tiến gần đến cấu trúc phân loại hiệu quả nhất.

Tổng quan, *ID3* xây dựng cây quyết định một cách đệ quy bằng cách chọn thuộc tính có khả năng phân biệt tốt nhất giữa các lớp của tập dữ liệu, với mục tiêu là giảm thiểu *entropy* và tối đa hóa tính *thuần nhất* trong các tập con.

3.1.3.2 Hàm Entropy và Information Gain

Hàm entropy là một khái niệm nền tảng trong lý thuyết thông tin do Claude Shannon phát triển, được sử dụng để định lượng mức độ hỗn loạn (*disorder*) hoặc bất thuần chủng (*impurity*) trong một tập dữ liệu huấn luyện (*training dataset*). Hàm entropy được biểu diễn thông qua công thức toán học như trong Công thức 3.1:

$$H(p) = - \sum_{i=1}^n p_i \log_2(p_i) \quad (3.1)$$

Trong đó, $p = (p_1, p_2, \dots, p_n)$ là phân phối xác suất của một biến rời rạc x có thể nhận n giá trị khác nhau x_1, x_2, \dots, x_n (tương ứng với n lớp trong bài toán phân lớp), với $p_i = p(x = x_i)$ biểu thị xác suất một điểm dữ liệu thuộc về lớp i . Các điều kiện áp dụng bao gồm $0 \leq p_i \leq 1$ và $\sum_{i=1}^n p_i = 1$, với quy ước $0 \log_2(0) = 0$ để đảm bảo tính toán hợp lệ trong các trường hợp giới hạn [46].

Giá trị của hàm *entropy* $H(p)$ phản ánh mức độ *hỗn loạn* trong tập dữ liệu một cách định lượng:

- $H(p)$ đạt giá trị nhỏ nhất, bằng 0, khi tồn tại một giá trị $p_i = 1$ (các giá trị còn lại bằng 0), tương ứng với trường hợp tập dữ liệu hoàn toàn *thuần chủng* (*purity*), tức là tất cả các điểm dữ liệu thuộc về cùng một lớp.
- $H(p)$ đạt giá trị tối đa khi tất cả các p_i bằng nhau và bằng $\frac{1}{n}$, phản ánh trạng thái *hỗn loạn* cao nhất, xảy ra khi các lớp được phân bố đồng đều trong tập dữ liệu [42].

Trong quá trình xây dựng cây quyết định với n lớp, tại mỗi nút nội bộ (*internal node*), tập dữ liệu S chứa các điểm dữ liệu được phân tích để xác định tính tối ưu cho việc phân chia. Đại lượng độ lợi thông tin (*information gain*) được sử dụng để đánh giá hiệu quả của một thuộc tính x (có tập giá trị $\{x_1, x_2, \dots, x_m\}$) và được định nghĩa theo Công thức 3.2:

$$G(x, S) = H(S) - \sum_{j=1}^m \frac{|S_j|}{|S|} H(S_j) \quad (3.2)$$

Trong đó:

- $H(S)$ là entropy của tập dữ liệu ban đầu S ,
- S_j là tập con của S chứa các điểm dữ liệu có giá trị $x = x_j$,
- $H(S_j)$ là entropy của tập con S_j ,
- $\frac{|S_j|}{|S|}$ đại diện cho tỷ lệ số lượng mẫu trong S_j so với tổng số lượng mẫu trong S .

Đại lượng độ lợi thông tin đo lường mức độ giảm *hỗn loạn* hoặc *bất thuần chủng* khi thực hiện phân chia tập S dựa trên thuộc tính x . Do đó, giá trị *độ lợi thông tin* càng lớn, thuộc tính x càng được đánh giá cao trong việc giảm *hỗn loạn*, từ đó nâng cao hiệu quả phân lớp. Thuộc tính “tối ưu” x^* được chọn để phân chia tại nút nội bộ được xác định thông qua quá trình tối ưu hóa như trong Công thức 3.3:

$$x^* = \arg \max_x G(x, S) \quad (3.3)$$

Quá trình này đảm bảo rằng tại mỗi bước, thuật toán ID3 lựa chọn thuộc tính mang lại sự phân chia hiệu quả nhất, từ đó xây dựng một cấu trúc cây quyết định tối ưu hóa khả năng phân

3.1. CÂY QUYẾT ĐỊNH

lớp. Tuy nhiên, phương pháp này phụ thuộc đáng kể vào chất lượng và tính đầy đủ của dữ liệu ban đầu, đồng thời gặp khó khăn khi xử lý dữ liệu bị thiếu hoặc thuộc tính liên tục, một hạn chế được khắc phục trong các thuật toán cải tiến như C4.5 [40], [44].

3.1.4 Điều kiện dừng

Một thách thức phổ biến trong các thuật toán cây quyết định là xu hướng tạo ra một cấu trúc cây đạt độ chính xác 100% trên tập dữ liệu huấn luyện nếu quá trình phân chia các nút chưa *thuần chủng* tiếp tục mà không áp dụng các giới hạn hợp lý. Tuy nhiên, điều này dẫn đến sự phát triển quá mức của cây, với số lượng nút (bao gồm cả nút lá) tăng lên đáng kể, thường bao gồm các nút lá chỉ chứa một lượng nhỏ điểm dữ liệu. Hậu quả trực tiếp của hiện tượng này là quá khớp, trong đó mô hình học quá chi tiết các đặc trưng đặc thù của tập huấn luyện, làm giảm khả năng tổng quát hóa trên dữ liệu mới hoặc tập kiểm tra [25].

Để kiểm soát và giảm thiểu nguy cơ quá khớp, các điều kiện dừng được đưa ra nhằm giới hạn quá trình phân chia nút một cách chiến lược. Các phương pháp này bao gồm:

- Entropy của nút đạt giá trị 0, tương ứng với trạng thái *thuần chủng* hoàn toàn, tức là tất cả các điểm dữ liệu trong nút thuộc về cùng một lớp, không cần phân chia thêm [42].
- Số lượng điểm dữ liệu trong nút nhỏ hơn một ngưỡng xác định trước, nhằm ngăn chặn sự phát triển quá sâu của cây với các nút chứa số lượng mẫu không đại diện [43].
- Khoảng cách từ nút hiện tại đến nút gốc (*root node*) nhỏ hơn một ngưỡng chiều sâu tối đa, giúp kiểm soát độ phức tạp tổng thể của cây [44].
- Giới hạn số lượng nút lá tối đa, một cách tiếp cận khác để duy trì tính đơn giản và tránh sự phức tạp không cần thiết trong cấu trúc cây [40].
- Các điều kiện tùy chỉnh dựa trên đặc thù của bài toán hoặc yêu cầu thực tiễn, chẳng hạn như ngưỡng lỗi hoặc độ tin cậy thống kê, tùy thuộc vào ngữ cảnh ứng dụng [25].

Trong số các chiến lược kiểm soát quá khớp, hai phương pháp được xem là hiệu quả và phổ biến nhất là kỹ thuật cắt tỉa (*pruning*) và thuật toán Rừng ngẫu nhiên (*Random Forest*) sẽ được đề cập trong phần 3.2 [47].

Kỹ thuật cắt tỉa (*pruning*) là một phương pháp hậu xử lý nhằm loại bỏ các nhánh và nút lá không mang tính đại diện hoặc không cần thiết trong cây quyết định, mà không làm suy giảm đáng kể độ chính xác trên tập dữ liệu huấn luyện. Mục tiêu chính của *cắt tỉa* là đơn giản hóa cấu trúc cây, từ đó cải thiện khả năng tổng quát hóa trên dữ liệu mới. Một phương pháp *cắt tỉa* phổ biến là *Reduced Error Pruning*, được thực hiện thông qua các bước sau:

- Chia tập dữ liệu huấn luyện ban đầu thành hai tập con: một tập huấn luyện nhỏ hơn dùng để xây dựng cây và một tập kiểm tra độc lập để đánh giá hiệu quả.
- Xây dựng cây quyết định trên tập huấn luyện cho đến khi tất cả các điểm dữ liệu trong tập này được phân lớp chính xác, dẫn đến một cây ban đầu có thể quá phức tạp.
- Thực hiện quá trình *cắt tỉa* bằng cách loại bỏ các nút lá và thay thế bằng các nút bối mẹ, với nhãn lớp được gán dựa trên lớp chiếm đa số trong các điểm dữ liệu của các nút con đã bị loại bỏ.
- Quá trình *cắt tỉa* tiếp tục lặp lại cho đến khi không còn cải thiện được độ chính xác trên tập kiểm tra, đánh dấu điểm dừng tối ưu hóa [48].

CHƯƠNG 3. CÂY QUYẾT ĐỊNH VÀ RỪNG NGẦU NHIÊN

Phương pháp *Reduced Error Pruning* đặc biệt hiệu quả trong việc cân bằng giữa độ chính xác trên tập huấn luyện và khả năng tổng quát hóa, mặc dù nó đòi hỏi một tập kiểm tra đủ lớn để đảm bảo kết quả đáng tin cậy. Ngoài ra, các kỹ thuật *cắt tia* khác, như *Cost-Complexity Pruning*, có thể được áp dụng để tối ưu hóa dựa trên một hàm chi phí kết hợp giữa độ phức tạp của cây và lỗi phân lớp [43].

3.1.5 Giải thuật

Giải thuật cây quyết định sử dụng thuật toán ID3 có mã giả như Thuật toán 3.1.

Algorithm 3.1: Thuật toán ID3

Input: Tập dữ liệu huấn luyện D , tập thuộc tính \mathcal{A}
Output: Cây quyết định T

- 1 **if** mọi ví dụ trong D thuộc cùng một lớp C **then**
- 2 **return** Nút lá gán nhãn C
- 3 **if** $\mathcal{A} = \emptyset$ **then**
- 4 **return** Nút lá gán nhãn lớp phổ biến nhất trong D
- 5 **if** $D = \emptyset$ **then**
- 6 **return** Nút lá với nhãn mặc định (lớp phổ biến nhất từ cha của D)
- 7 Chọn thuộc tính $A \in \mathcal{A}$ sao cho **Information Gain** là lớn nhất;
- 8 Tạo nút gốc T gán nhãn A ;
- 9 **for** mỗi giá trị v của A **do**
- 10 **let** $D_v \leftarrow$ tập con của D thoả $A = v$;
- 11 **let** $T_v \leftarrow \text{ID3}(D_v, \mathcal{A} \setminus \{A\})$;
- 12 Gắn T_v làm nhánh con của T tương ứng với giá trị v ;
- 13 **return** T

Ví dụ 3.1. Để khám phá cách xây dựng cây quyết định một cách chi tiết, chúng ta sẽ thực hiện từng bước để phát triển mô hình dựa trên tập dữ liệu huấn luyện được liệt kê trong Bảng 3.2. Bảng này trình bày mối quan hệ giữa ba yếu tố *Thời tiết*, *Nhiệt độ*, và *Số người tham gia* với quyết định đi đánh cầu. Nhiệm vụ của chúng ta là dự đoán kết quả ở cột cuối cùng (Di đánh cầu) dựa trên thông tin từ Bảng 3.2.

Ta nhận thấy có 3 thuộc tính ở dạng rời rạc:

- *Thời tiết* nhận 1 trong 2 giá trị: Tốt và Xấu.
- *Nhiệt độ* nhận 1 trong 2 giá trị: Cao và Thấp.
- *Số người tham gia* nhận 1 trong 2 giá trị: Nhiều (> 3 người) và Ít (≤ 3 người).

Mặc dù trên thực tế, *Nhiệt độ* là một thuộc tính liên tục, ta vẫn có thể áp dụng mô hình này bằng cách phân loại rời rạc theo ngưỡng định trước, ví dụ: Cao ($> 30^\circ\text{C}$), Thấp ($\leq 30^\circ\text{C}$). Từ tập dữ liệu huấn luyện với 9 mẫu, gồm 6 “Có” và 3 “Không”, ta sẽ xây dựng cây quyết định để dự đoán việc đi đánh cầu.

Nút gốc chứa tập dữ liệu huấn luyện ban đầu với 9 mẫu, gồm 6 mẫu thuộc lớp “Có” và 3 mẫu thuộc lớp “Không”. Từ Công thức 3.1, ta tính được entropy của nút gốc như sau:

3.1. CÂY QUYẾT ĐỊNH

Bảng 3.2: Dữ liệu huấn luyện về việc đi đánh cầu

Thời tiết	Nhiệt độ	Số người tham gia	Đi đánh cầu
Tốt	Cao	Nhiều	Có
Tốt	Cao	Ít	Không
Xấu	Cao	Nhiều	Không
Xấu	Cao	Ít	Không
Tốt	Thấp	Nhiều	Có
Tốt	Thấp	Ít	Có
Tốt	Thấp	Nhiều	Có
Tốt	Thấp	Ít	Có
Tốt	Cao	Nhiều	Có

$$H = -\frac{6}{9} \log_2 \frac{6}{9} - \frac{3}{9} \log_2 \frac{3}{9} \approx 0.9183 \quad (3.4)$$

Chia nút gốc:

- Nếu dùng thuộc tính *Thời tiết* (viết tắt là TT) để phân chia nút gốc, ta nhận được hai nút con với entropy như sau:

- Nút con ứng với *Thời tiết* = Tốt (6 mẫu: 5 “Có”, 1 “Không”):

$$H(TT = \text{Tốt}) = -\frac{5}{6} \log_2 \frac{5}{6} - \frac{1}{6} \log_2 \frac{1}{6} \approx 0.6500 \quad (3.5)$$

- Nút con ứng với *Thời tiết* = Xấu (3 mẫu: 0 “Có”, 3 “Không”):

$$H(TT = \text{Xấu}) = -\frac{0}{3} \log_2 \frac{0}{3} - \frac{3}{3} \log_2 \frac{3}{3} = 0.0000 \quad (3.6)$$

Information Gain khi phân chia nút gốc theo thuộc tính *Thời tiết*:

$$\begin{aligned} G(TT) &= 0.9183 - \left(\frac{6}{9} \cdot 0.6500 + \frac{3}{9} \cdot 0.0000 \right) \\ &\approx 0.9183 - (0.6667 \cdot 0.6500 + 0.3333 \cdot 0.0000) \\ &\approx 0.4850 \end{aligned} \quad (3.7)$$

- Nếu dùng thuộc tính *Nhiệt độ* (viết tắt là NT) để phân chia nút gốc, ta nhận được hai nút con với entropy như sau:

- Nút con ứng với *Nhiệt độ* = Cao (4 mẫu, 2 “Có”, 2 “Không”):

$$H(NT = \text{Cao}) = -\frac{2}{4} \log_2 \frac{2}{4} - \frac{2}{4} \log_2 \frac{2}{4} = 1.0000 \quad (3.8)$$

CHƯƠNG 3. CÂY QUYẾT ĐỊNH VÀ RỪNG NGÃU NHIÊN

- Nút con ứng với *Nhiệt độ* = Thấp (5 mẫu, 4 “Có”, 1 “Không”):

$$H(NT = \text{Thấp}) = -\frac{4}{5} \log_2 \frac{4}{5} - \frac{1}{5} \log_2 \frac{1}{5} \approx 0.7219 \quad (3.9)$$

Information Gain khi phân chia nút gốc theo thuộc tính *Nhiệt độ*:

$$G(NT) = 0.9183 - \left(\frac{4}{9} \cdot 1.0000 + \frac{5}{9} \cdot 0.7219 \right) \approx 0.9183 - (0.4444 + 0.4011) \approx 0.0728 \quad (3.10)$$

- Nếu dùng thuộc tính *Số người tham gia* (viết tắt là SPT) để phân chia nút gốc, ta nhận được hai nút con với entropy như sau:

- Nút con ứng với *Số người tham gia* = Nhiều (4 mẫu, 3 “Có”, 1 “Không”):

$$H(SPT = \text{Nhiều}) = -\frac{3}{4} \log_2 \frac{3}{4} - \frac{1}{4} \log_2 \frac{1}{4} \approx 0.8113 \quad (3.11)$$

- Nút con ứng với *Số người tham gia* = Ít (5 mẫu, 3 “Có”, 2 “Không”):

$$H(SPT = \text{Ít}) = -\frac{3}{5} \log_2 \frac{3}{5} - \frac{2}{5} \log_2 \frac{2}{5} \approx 0.9710 \quad (3.12)$$

Information Gain khi phân chia nút gốc theo thuộc tính *Số người tham gia*:

$$G(SPT) = 0.9183 - \left(\frac{4}{9} \cdot 0.8113 + \frac{5}{9} \cdot 0.9710 \right) \approx 0.9183 - (0.3606 + 0.5394) \approx 0.0183 \quad (3.13)$$

Nhận thấy, khi dùng thuộc tính *Thời tiết* để phân chia nút gốc sẽ có Information Gain cao nhất (0.4850), nên ta chọn thuộc tính *Thời tiết* để phân chia **nút gốc**.

Sau bước phân chia đầu tiên, ta nhận được hai nút con:

- Nút con ứng với *Thời tiết* = Tốt: Chứa 6 mẫu, cần phân chia tiếp vì chưa tinh khiết.
- Nút con ứng với *Thời tiết* = Xấu: Đã tinh khiết (entropy = 0, nhãn “Không”), nên không cần phân chia tiếp.

Cây quyết định thu được như trong Hình 3.1.

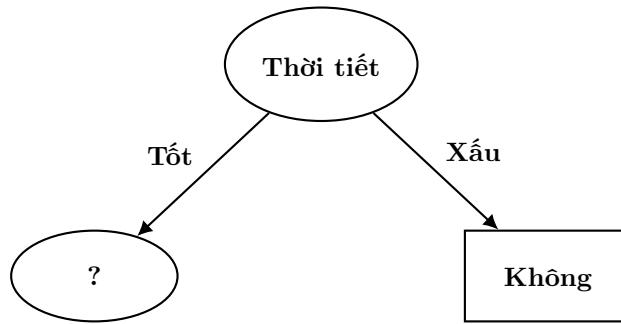
Chia nút con ứng với *Thời tiết* = Tốt:

Khi *Thời tiết* = Tốt, tập dữ liệu như trong Bảng 3.3, bao gồm có 6 mẫu: 5 “Có” và 1 “Không”. Entropy được tính như công thức 3.14

$$H = -\frac{5}{6} \log_2 \frac{5}{6} - \frac{1}{6} \log_2 \frac{1}{6} \approx 0.6500 \quad (3.14)$$

Tương tự như bước trước, ta xét các thuộc tính còn lại để tìm thuộc tính có Information Gain cao nhất.

3.1. CÂY QUYẾT ĐỊNH



Hình 3.1: Cây quyết định theo thuộc tính *Thời tiết*.

Bảng 3.3: Tập dữ liệu ứng với *Thời tiết* = Tốt

Nhiệt độ	Số người tham gia	Đi đánh cầu
Cao	Nhiều	Có
Cao	Ít	Không
Thấp	Nhiều	Có
Thấp	Ít	Có
Thấp	Nhiều	Có
Thấp	Ít	Có

- Xét thuộc tính *Nhiệt độ* (viết tắt là NT), có 2 giá trị: Cao và Thấp. Ta tính entropy cho từng giá trị:

$$H(NT = \text{Cao}) = -\frac{1}{2} \log_2 \frac{1}{2} - \frac{1}{2} \log_2 \frac{1}{2} = 1.0000$$

$$H(NT = \text{Thấp}) = -\frac{4}{4} \log_2 \frac{4}{4} - \frac{0}{4} \log_2 \frac{0}{4} = 0.0000$$

Information Gain của *Nhiệt độ*:

$$G(NT) = 0.6500 - \left(\frac{2}{6} \cdot 1.0000 + \frac{4}{6} \cdot 0.0000 \right) \approx 0.6500 - 0.3333 = 0.3167$$

- Xét thuộc tính *Số người tham gia* (viết tắt là SPT), có 2 giá trị: Nhiều và Ít. Ta tính entropy cho từng giá trị:

$$H(SPT = \text{Nhiều}) = -\frac{3}{3} \log_2 \frac{3}{3} - \frac{0}{3} \log_2 \frac{0}{3} = 0.0000$$

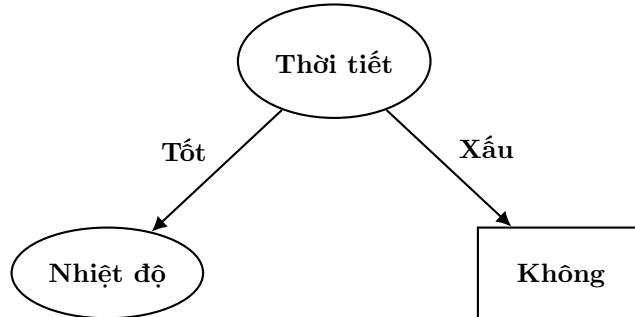
$$H(SPT = \text{Ít}) = -\frac{2}{3} \log_2 \frac{2}{3} - \frac{1}{3} \log_2 \frac{1}{3} \approx 0.9183$$

Information Gain của *Số người tham gia*:

CHƯƠNG 3. CÂY QUYẾT ĐỊNH VÀ RỪNG NGÃU NHIÊN

$$G(SPT) = 0.6500 - \left(\frac{3}{6} \cdot 0.0000 + \frac{3}{6} \cdot 0.9183 \right) \approx 0.6500 - 0.4592 = 0.1908$$

Vì Information Gain của thuộc tính *Nhiệt độ* (0.3167) cao hơn, ta chọn thuộc tính *Nhiệt độ* để phân chia nút con này, và thu được cây quyết định như trong Hình 3.2.



Hình 3.2: Cây quyết định sau khi chia theo *Thời tiết* = Tốt

Sau khi phân chia theo thuộc tính *Nhiệt độ*, ta tiếp tục nhận được hai nút con:

- Nút con ứng với *Nhiệt độ* = Thấp: Chứa 4 mẫu, toàn bộ đều thuộc lớp “Có”, nên đã tinh khiết (entropy = 0, nhãn “Có”). Bảng 3.4 trình bày tập dữ liệu này.

Bảng 3.4: Tập dữ liệu ứng với *Nhiệt độ* = Thấp

Số người tham gia	Đi đánh cầu
Nhiều	Có
Ít	Có
Nhiều	Có
Ít	Có

Do đó, nút này đã tinh khiết (entropy = 0, nhãn “Có”), nên không cần phân chia tiếp.

- Nút con ứng với *Nhiệt độ* = Cao: Chứa 2 mẫu, gồm 1 “Có” và 1 “Không”, thể hiện như trong Bảng 3.5.

Bảng 3.5: Tập dữ liệu ứng với *Nhiệt độ* = Cao

Số người tham gia	Đi đánh cầu
Nhiều	Có
Ít	Không

Ta xét tiếp trường hợp của nút con này như sau:

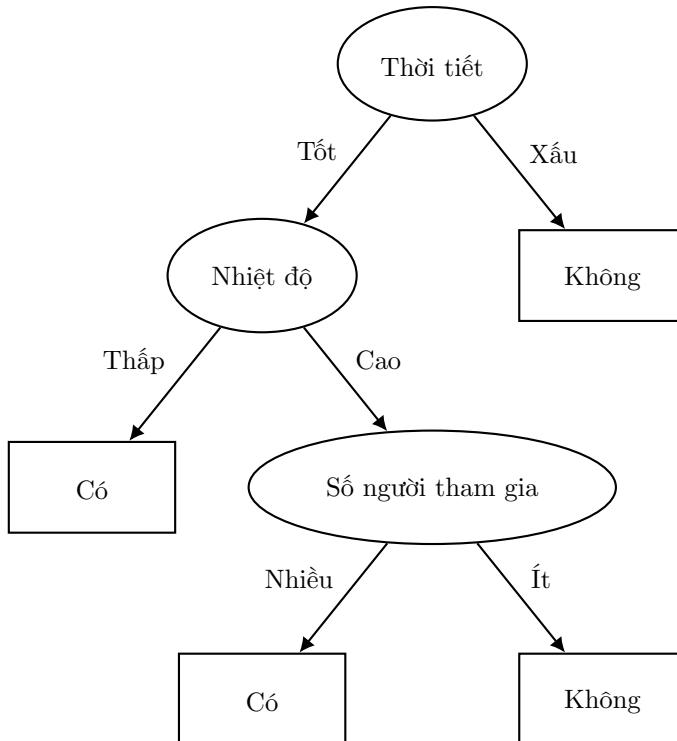
- Nút con ứng với *Số người tham gia* = Nhiều: Chứa 1 mẫu, thuộc lớp “Có”, nên đã tinh khiết (entropy = 0, nhãn “Có”).

3.2. RỪNG NGÂU NHIÊN

- Nút con ứng với $Số người tham gia = Ít$: Chứa 1 mẫu, thuộc lớp “Không”, nên đã tinh khiết (entropy = 0, nhãn “Không”).

Do cả hai nút con này đều đã tinh khiết, ta không cần phân chia tiếp.

Như vậy, cây quyết định hoàn chỉnh như Hình 3.3.



Hình 3.3: Cây quyết định hoàn chỉnh

Dự đoán ví dụ: Xét một trường hợp cụ thể với các thuộc tính: $Thời tiết = Tốt$, $Nhiệt độ = Cao$ (trên 30°C), và $Số người tham gia = Nhiều$, quá trình dự đoán được thực hiện theo các bước sau:

- Khởi đầu tại nút gốc với giá trị $Thời tiết = Tốt$ dẫn đến việc chuyển sang nút con $Nhiệt độ$.
- Tại nút con $Nhiệt độ$, với giá trị $Nhiệt độ = Cao$ dẫn đến việc chuyển sang nút con $Số người tham gia$.
- Tại nút con $Số người tham gia$, với giá trị $Số người tham gia = Nhiều$ dẫn đến nhãn “Có”.

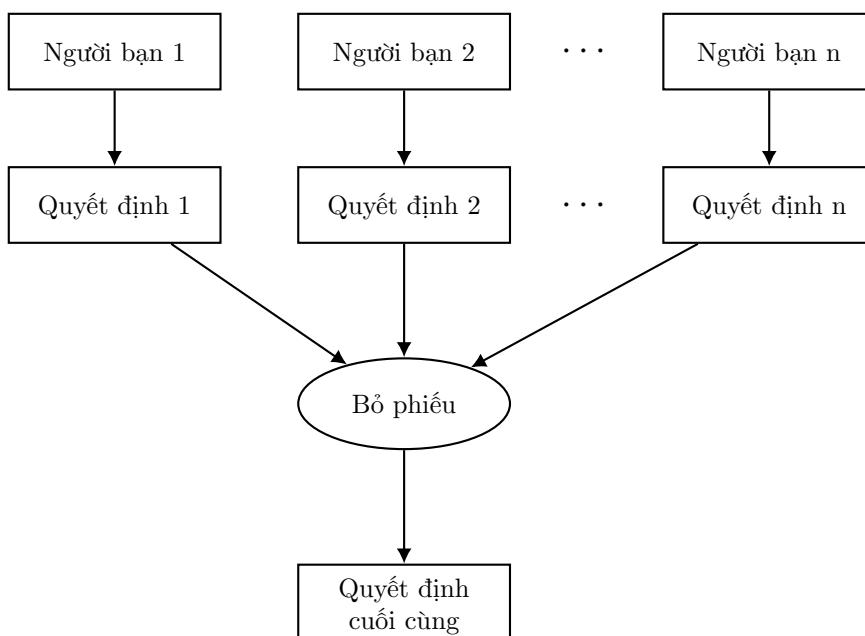
Do đó, ngày này được dự đoán sẽ diễn ra hoạt động đi đánh cầu.

3.2 Rừng ngẫu nhiên

3.2.1 Bài toán mở đầu

Trong nhiều tình huống ra quyết định thực tiễn, việc lựa chọn phương án tối ưu dựa trên thông tin tổng hợp từ tập thể thường cho kết quả đáng tin cậy hơn so với việc chỉ dựa vào một nguồn duy nhất. Chẳng hạn, xét bài toán lựa chọn một bộ phim để xem trong số các bộ phim đang được công chiếu. Một cá nhân có thể tham khảo ý kiến từ nhiều người bạn nhằm đưa ra quyết định cuối cùng.

Quy trình ra quyết định trong trường hợp này bao gồm ba bước chính: (i) lập danh sách các bộ phim khả thi, (ii) thu thập các đánh giá độc lập, và (iii) lựa chọn bộ phim nhận được nhiều phiếu ủng hộ nhất thông qua một cơ chế bỏ phiếu. Phương pháp này không chỉ phản ánh sự đồng thuận trong tập thể mà còn góp phần giảm thiểu ảnh hưởng của các đánh giá chủ quan hoặc sai lệch từ một cá nhân. Toàn bộ quy trình được minh họa trong Hình 3.4.



Hình 3.4: Sơ đồ quy trình chọn bộ phim hay nhất dựa trên ý kiến tập thể

Cách tiếp cận dựa trên sự đồng thuận này là một nguyên tắc phổ biến trong nhiều khía cạnh của đời sống, từ quản lý dự án đến phân tích dữ liệu, nhờ khả năng giảm thiểu sai số và tăng độ tin cậy [49]. Trong lĩnh vực học máy, nguyên tắc tương tự được thể hiện thông qua mô hình *rừng ngẫu nhiên*, một phương pháp học có giám sát được phát triển để nâng cao hiệu suất dự đoán bằng cách kết hợp nhiều cây quyết định đơn lẻ [50]. Khái niệm rừng ngẫu nhiên ra đời từ ý tưởng rằng sự đa dạng trong các mô hình riêng lẻ, khi được tổng hợp, dẫn đến kết quả tổng quát hóa tốt hơn so với bất kỳ mô hình đơn lẻ nào, một nguyên tắc được gọi là khả năng tổng hợp thông minh (*wisdom of crowds*) [51].

Rừng ngẫu nhiên hoạt động bằng cách xây dựng một tập hợp các cây quyết định độc lập, trong đó mỗi cây được huấn luyện trên một mẫu ngẫu nhiên của tập dữ liệu huấn luyện cùng với một tập con ngẫu nhiên của các thuộc tính. Các dự đoán từ từng cây thành phần sau đó

3.2. RỪNG NGẦU NHIÊN

được tổng hợp lại — thông qua cơ chế bỏ phiếu đa số trong bài toán phân lớp hoặc trung bình hóa trong bài toán hồi quy — để tạo ra kết quả cuối cùng. Cách tiếp cận này giúp cải thiện độ chính xác tổng thể của mô hình bằng cách giảm thiểu hiện tượng quá khớp và biến thiên [40], [50]. Khác với một cây quyết định đơn lẻ, rừng ngẫu nhiên đưa vào hai yếu tố ngẫu nhiên hóa trong quá trình huấn luyện (mẫu dữ liệu và tập thuộc tính), từ đó tăng cường khả năng tổng quát hóa và cải thiện độ ổn định của mô hình [51].

3.2.2 Khái niệm

Rừng ngẫu nhiên (*Random Forest*) là một mô hình học có giám sát được phát triển nhằm giải quyết các bài toán phân lớp và hồi quy với hiệu suất vượt trội so với các mô hình cây quyết định đơn lẻ [50], [52]. Tương tự như một khu rừng được hình thành từ nhiều cây, mô hình này bao gồm một tập hợp các cây quyết định được xây dựng độc lập, mỗi cây được huấn luyện trên một tập dữ liệu con được tạo ngẫu nhiên từ tập dữ liệu huấn luyện ban đầu. Kết quả dự đoán cuối cùng của rừng ngẫu nhiên được xác định thông qua cơ chế bỏ phiếu đa số (*majority voting*) cho bài toán phân lớp hoặc trung bình hóa cho bài toán hồi quy, dựa trên các dự đoán từ từng cây quyết định riêng lẻ [40], [53].

Cơ chế hoạt động của rừng ngẫu nhiên có thể được liên hệ với ví dụ về quá trình ra quyết định tập thể được minh họa trong Hình 3.4. Trong đó, mỗi cây quyết định đóng vai trò tương tự như một cá nhân với kinh nghiệm riêng, được huấn luyện trên một tập dữ liệu con khác nhau, trong khi kết quả cuối cùng của rừng ngẫu nhiên tương ứng với quyết định chung đạt được thông qua bỏ phiếu đa số. Phương pháp này không chỉ tăng cường độ chính xác dự đoán mà còn giảm thiểu hiện tượng quá khớp, một hạn chế phổ biến của cây quyết định đơn lẻ, nhờ vào sự đa dạng và độc lập giữa các cây trong rừng [50].

Dặc trưng “ngẫu nhiên” trong rừng ngẫu nhiên được thể hiện qua hai yếu tố chính, đóng vai trò cốt lõi trong việc tối ưu hóa hiệu suất mô hình:

- **Ngẫu nhiên hóa tập dữ liệu huấn luyện:** Quá trình này được thực hiện thông qua kỹ thuật *bootstrapping*, một phương pháp lấy mẫu có lặp lại từ tập dữ liệu huấn luyện ban đầu. Mỗi tập dữ liệu con được tạo ra ngẫu nhiên, cho phép một số điểm dữ liệu xuất hiện nhiều lần trong cùng một tập, trong khi các điểm khác có thể bị loại bỏ. Kỹ thuật này, thường được gọi là *bagging* (*bootstrap aggregating*), đảm bảo rằng mỗi cây quyết định trong rừng được huấn luyện trên một tập con khác nhau, từ đó tăng cường tính đa dạng và giảm thiểu biến thiên trong dự đoán [44], [50]. Sự lặp lại của các điểm dữ liệu trong quá trình lấy mẫu giúp mô hình thích nghi với các mẫu đặc trưng, đồng thời hạn chế tác động của nhiều dữ liệu, một yếu tố quan trọng trong việc cải thiện độ tổng quát hóa [25].
- **Ngẫu nhiên hóa tập con thuộc tính:** Trong quá trình phân chia các nút nội bộ (*internal node*) của mỗi cây quyết định, chỉ một tập con ngẫu nhiên của các thuộc tính được xem xét thay vì toàn bộ tập thuộc tính. Số lượng thuộc tính trong tập con thường được đặt bằng căn bậc hai của tổng số thuộc tính ban đầu, một quy tắc được đề xuất bởi Breiman để cân bằng giữa tính đa dạng và hiệu quả tính toán [50]. Chẳng hạn, nếu tập dữ liệu có 16 thuộc tính, chỉ khoảng 4 thuộc tính được chọn ngẫu nhiên tại mỗi nút để xác định ngưỡng phân chia. Phương pháp này không chỉ tăng cường tính độc lập giữa các cây mà còn giảm nguy cơ quá khớp bằng cách hạn chế sự phụ thuộc vào một tập thuộc tính cố định [25].

Sự kết hợp của hai yếu tố ngẫu nhiên hóa kể trên tạo nên một mô hình có khả năng học linh hoạt từ dữ liệu phức tạp, đồng thời duy trì độ ổn định và độ chính xác cao hơn so với một cây quyết định đơn lẻ. Tuy vậy, việc lựa chọn các tham số như số lượng cây trong rừng hoặc kích thước tập con thuộc tính cần được cân nhắc cẩn trọng, nhằm tránh làm gia tăng chi phí tính toán mà không mang lại cải thiện đáng kể về hiệu suất [44].

3.2.3 Nhược điểm của cây quyết định và sự cải tiến của Rừng ngẫu nhiên

Cây quyết định là một mô hình học có giám sát được sử dụng phổ biến trong lĩnh vực học máy nhờ tính dễ hiểu và khả năng biểu diễn trực quan các quy tắc phân loại. Tuy nhiên, mô hình này còn tồn tại một số hạn chế đáng kể, ảnh hưởng đến hiệu quả áp dụng trong các bài toán quy mô lớn hoặc đòi hỏi độ chính xác cao. Một trong những nhược điểm quan trọng nhất của cây quyết định là hiện tượng quá khớp, đặc biệt khi dữ liệu huấn luyện có nhiều biến động hoặc nhiễu. Cây quyết định thường có xu hướng ghi nhớ chi tiết các đặc điểm cụ thể của tập huấn luyện, và do đó có thể dẫn đến mô hình phù hợp quá mức với dữ liệu đã thấy, nhưng lại không tổng quát tốt với các dữ liệu mới, chưa thấy.

Ngoài ra, cây quyết định cũng có thể bị thiên lệch nếu dữ liệu huấn luyện không đủ đa dạng hoặc có sự phân bố không đồng đều của các lớp. Điều này có thể dẫn đến việc cây quyết định chỉ học được các quy tắc phân chia mà không phản ánh đúng được sự phân bố của dữ liệu tổng thể.

Cuối cùng, cây quyết định có thể gặp phải vấn đề không ổn định, tức là thay đổi nhỏ trong dữ liệu huấn luyện có thể dẫn đến sự thay đổi lớn trong cấu trúc của cây. Điều này làm cho cây quyết định có thể trở nên khá nhạy cảm với những thay đổi trong tập huấn luyện và gây khó khăn trong việc xây dựng mô hình đáng tin cậy.

Để khắc phục những hạn chế trên, rừng ngẫu nhiên được đề xuất như một phương pháp tiên tiến, tận dụng sự kết hợp của nhiều cây quyết định để nâng cao hiệu suất và khả năng thích nghi. Mô hình này xây dựng một tập hợp các cây quyết định song song, mỗi cây được huấn luyện trên một tập dữ liệu con ngẫu nhiên, và kết quả cuối cùng được tổng hợp thông qua cơ chế bỏ phiếu đa số (*majority voting*) cho phân lớp hoặc trung bình hóa cho hồi quy [50]. Sự cải tiến này không chỉ giải quyết các nhược điểm của cây quyết định mà còn mở ra tiềm năng ứng dụng trong các bài toán quy mô lớn. Rừng ngẫu nhiên có thể giải quyết các vấn đề của cây quyết định bằng cách:

- *Giảm hiện tượng quá khớp*: Việc huấn luyện nhiều cây quyết định trên các mẫu dữ liệu khác nhau giúp rừng ngẫu nhiên giảm thiểu hiện tượng quá khớp. Mỗi cây được xây dựng dựa trên một tập con khác biệt của dữ liệu (thông qua quá trình bootstrapping), do đó mô hình tổng thể ít bị ảnh hưởng bởi các điểm dữ liệu nhiễu hoặc ngoại lai.
- *Tăng tính đa dạng và độ ổn định*: Bằng cách sử dụng nhiều cây quyết định được huấn luyện trên các tập dữ liệu và tập thuộc tính khác nhau, rừng ngẫu nhiên có khả năng học từ các khía cạnh khác nhau của dữ liệu. Điều này giúp tạo ra một mô hình ổn định hơn và ít nhạy cảm với các biến động nhỏ trong tập huấn luyện.
- *Tối ưu hóa quá trình chọn thuộc tính*: Thay vì sử dụng toàn bộ tập thuộc tính tại mỗi nút phân chia, rừng ngẫu nhiên chỉ lựa chọn ngẫu nhiên một tập con các thuộc tính. Chiến lược này giúp giảm thiểu sự thiên lệch trong việc học và tăng khả năng tổng quát hóa của mô hình.
- *Hiệu suất thực nghiệm cao*: Trong nhiều bài toán thực tế, rừng ngẫu nhiên đạt được hiệu suất rất cao và thường khó bị vượt qua. Nhờ cơ chế bỏ phiếu, các cây quyết định có hiệu suất thấp hoặc bị quá khớp sẽ có ảnh hưởng hạn chế đến kết quả cuối cùng, trong khi các cây mạnh được ưu tiên. Việc kết hợp nhiều cây chất lượng cao giúp cải thiện độ chính xác phân loại so với việc sử dụng một cây quyết định đơn lẻ.

Trong trường hợp dữ liệu có ít điểm dữ liệu hoặc dữ liệu nhiễu, rừng ngẫu nhiên vẫn có thể cho ra kết quả phân loại tốt nhờ vào việc lựa chọn và kết hợp những cây quyết định có hiệu suất cao, giúp mô hình trở nên mạnh mẽ và chính xác hơn.

3.2.4 Giải thuật

Giải thuật huấn luyện rừng ngẫu nhiên và dự đoán với rừng ngẫu nhiên có mã giả như Algorithm 3.2 và Algorithm 3.3.

Algorithm 3.2: RandomForestTrain — Huấn luyện mô hình Random Forest

Input: Tập huấn luyện (X, Y) gồm N mẫu;
 Số lượng cây B ;
 Số đặc trưng ngẫu nhiên tại mỗi node: m
Output: Tập hợp các cây $\{f_1, f_2, \dots, f_B\}$

- 1 Khởi tạo rừng rỗng: $\mathcal{F} \leftarrow \emptyset$;
- 2 **for** $b = 1$ **to** B **do**
- 3 Lấy mẫu bootstrap (X_b, Y_b) từ (X, Y) bằng cách chọn N mẫu có hoàn lại;
- 4 Huấn luyện cây f_b trên (X_b, Y_b) với:
 - Tại mỗi node: chọn ngẫu nhiên m đặc trưng và chia theo đặc trưng tốt nhất
 - Thêm f_b vào rừng: $\mathcal{F} \leftarrow \mathcal{F} \cup \{f_b\}$;
- 5 **return** \mathcal{F}

Algorithm 3.3: RandomForestPredict — Dự đoán với mô hình Random Forest

Input: Mẫu mới x ;
 Tập huấn luyện (X, Y) ;
 Số cây B , số đặc trưng ngẫu nhiên m
Output: Dự đoán \hat{y}

- 1 Huấn luyện mô hình: $\mathcal{F} \leftarrow \text{RandomForestTrain}(X, Y, B, m)$;
- 2 Lấy dự đoán từ từng cây: $\hat{y}_b \leftarrow f_b(x), \forall f_b \in \mathcal{F}$;
- 3 **return** Biểu quyết đa số (classification) hoặc trung bình (regression) của $\{\hat{y}_1, \dots, \hat{y}_B\}$

3.3 Bài tập

Bài tập 3.1. Bạn là chuyên viên phân tích dữ liệu tại một chuỗi cửa hàng bán lẻ. Bạn được giao nhiệm vụ phân tích hành vi mua hàng của khách sau buổi tư vấn để dự đoán khả năng mua hàng (Purchase Decision), dựa trên các thông tin liên quan đến khách hàng và quá trình tư vấn.

Bảng 3.6: Bảng dữ liệu khách hàng với các đặc trưng và quyết định mua hàng.

Financial Status	Interest Level	Consultation Time	Previous Visit	Purchase Decision
Stable	High	Long	Yes	Yes
Limited	Low	Short	No	No
Stable	High	Long	Yes	Yes
Limited	Low	Short	No	No
Average	Medium	Medium	Yes	Yes
Stable	High	Long	No	Yes
Limited	Low	Short	No	No
Average	Medium	Medium	Yes	Yes
Stable	High	Medium	Yes	Yes
Limited	Low	Short	No	No

CHƯƠNG 3. CÂY QUYẾT ĐỊNH VÀ RỪNG NGÃU NHIÊN

Bảng dữ liệu trên mô tả các yếu tố ảnh hưởng đến việc khách hàng đưa ra quyết định mua hàng sau khi được tư vấn. Các cột gồm:

- Financial Status: Tình trạng tài chính của khách hàng (Stable, Average, Limited).
- Interest Level: Mức độ quan tâm đến sản phẩm (High, Medium, Low).
- Consultation Time: Thời gian tư vấn mà khách hàng nhận được (Short, Medium, Long).
- Previous Visit: Khách hàng đã từng đến cửa hàng trước đó hay chưa (Yes, No).
- Purchase Decision: Quyết định cuối cùng của khách hàng (Yes - Mua hàng, No - Không mua).

Hãy dùng Bảng 3.6 cho a và b.

a) Tính Entropy tại node đầu tiên (node gốc) và chỉ ra thuộc tính nào được chọn để chia cây quyết định.

b) Xây dựng một cây quyết định hoàn chỉnh cho bảng dữ liệu trên. Tìm kết quả dự đoán của các trường hợp sau:

- Financial Status: Average, Interest Level: Medium, Consultation Time: Medium, Previous Visit: Yes
- Financial Status: Limited, Interest Level: Low, Consultation Time: Short, Previous Visit: No
- Financial Status: Stable, Interest Level: High, Consultation Time: Long, Previous Visit: Yes

Bài tập 3.2. Bạn là nhân viên QC (Quality Control) được giao nhiệm vụ phân tích dữ liệu từ một nhà máy sản xuất để dự đoán lớp chất lượng sản phẩm (Quality Class), dựa trên các thông tin liên quan đến quá trình sản xuất.

Bảng 3.7: Bảng dữ liệu sản phẩm với các đặc trưng và phân loại chất lượng.

Defect Rate	Material Quality	Safety Rating	Inspection Score	Quality Class
0.05	High	High	90	High
0.10	Medium	Medium	70	Low
0.02	High	High	95	High
0.15	Low	Low	40	Low
0.07	Medium	Medium	60	Low
0.01	High	High	98	High
0.12	Low	Low	45	Low
0.05	Medium	Medium	65	Low
0.03	High	High	93	High
0.08	Medium	Medium	68	Low
0.04	High	High	96	High
0.09	Medium	Low	50	Low
0.06	Medium	Medium	75	High
0.02	High	High	85	High

Bảng dữ liệu trên mô tả các yếu tố ảnh hưởng đến chất lượng của từng sản phẩm trong quy trình sản xuất. Các cột gồm:

3.3. BÀI TẬP

- Defect Rate: Tỷ lệ lỗi phát sinh trong quá trình sản xuất (tính bằng phần trăm).
- Material Quality: Đánh giá chất lượng nguyên liệu đầu vào (High, Medium, Low).
- Safety Rating: Mức độ an toàn của quy trình sản xuất (High, Medium, Low).
- Inspection Score: Kết quả kiểm tra chất lượng của sản phẩm (chỉ số từ 10 đến 100).
- Quality Class: Chất lượng cuối cùng của sản phẩm, với hai giá trị: “High” (Chất lượng cao) hoặc “Low” (Chất lượng thấp).

Hãy dùng Bảng 3.7 cho a và b.

- a) Tính Entropy tại node đầu tiên (node gốc) và chỉ ra thuộc tính nào được chọn để chia cây quyết định.
b) Xây dựng một cây quyết định hoàn chỉnh cho bảng dữ liệu trên. Tìm kết quả dự đoán của các trường hợp sau:

- Defect Rate: 0.06, Material Quality: “Medium”, Safety Rating: “Medium”, Inspection Score: 75
- Defect Rate: 0.09, Material Quality: “Low”, Safety Rating: “Low”, Inspection Score: 50
- Defect Rate: 0.04, Material Quality: “Medium”, Safety Rating: “High”, Inspection Score: 92

Bài tập 3.3. Bạn là một nhân viên quản lý rủi ro tại một ngân hàng. Nhiệm vụ của bạn là phê duyệt khoản vay cho khách hàng dựa trên các thông tin mà họ cung cấp trong Bảng 3.8.

Bảng 3.8: Dữ liệu phê duyệt khoản vay với các thuộc tính đầu vào và kết quả.

Age	Income Loan Ratio	Credit Score	Education Level	Loan Approval Status
25	0.714	700	Bachelor	Approved
35	1.091	450	High School	Not Approved
45	—	800	Bachelor	Approved
55	1.067	750	Master	Approved
—	5.500	620	High School	Approved
32	2.000	680	Bachelor	Approved
40	—	720	Master	Not Approved
—	2.031	660	High School	Not Approved
60	0.878	690	Bachelor	Approved
30	0.879	640	Bachelor	Not Approved

Bảng dữ liệu 3.8 mô tả các yếu tố ảnh hưởng đến khả năng được phê duyệt khoản vay (Loan Approval Status) của một khách hàng. Các cột gồm:

- Age: Tuổi của khách hàng.
- Income Loan Ratio: Tỷ lệ giữa thu nhập hàng năm của khách hàng và số tiền khoản vay mà họ yêu cầu. Tỷ lệ này được tính bằng cách chia thu nhập hàng năm cho khoản vay. Tỷ lệ càng cao cho thấy khách hàng có khả năng chi trả khoản vay tốt hơn.
- Credit Score: Điểm tín dụng của khách hàng (tính theo thang điểm FICO Score từ 300 đến 850).

CHƯƠNG 3. CÂY QUYẾT ĐỊNH VÀ RỪNG NGẦU NHIÊN

- Education Level: Trình độ học vấn của khách hàng với ba giá trị có thể là “High School” (Trung học phổ thông), “Bachelor” (Cử nhân), hoặc “Master” (Thạc sĩ).
- Loan Approval Status: Tình trạng phê duyệt khoản vay đối với khách hàng với hai giá trị có thể là “Not Approved” (Không được phê duyệt) hoặc “Approved” (Được phê duyệt).

Hãy dùng dữ liệu Bảng 3.8 cho hai câu sau:

- a) Hãy trình bày các bước xử lý dữ liệu bị thiếu trong Bảng 3.8.
- b) Xây dựng một cây quyết định hoàn chỉnh cho Bảng 3.8. Tìm kết quả dự đoán của các trường hợp:
 - Khách hàng 1: Age: 28, Income Loan Ratio: 0.9, Credit Score: 720, Education Level: Bachelor.
 - Khách hàng 2: Age: 38, Income Loan Ratio: 0.8, Credit Score: 780, Education Level: Master.
 - Khách hàng 3: Age: 50, Income Loan Ratio: 1.1, Credit Score: 600, Education Level: High School.

Bài tập 3.4. Vì sao thuật toán Cây quyết định lại rất dễ bị gắp hiện tượng “quá khớp”? Vì sao Rừng ngẫu nhiên lại khắc phục được điều đó?

Chương 4

Thuật toán Naïve Bayes

4.1 Bài toán mở đầu

Trong bối cảnh kỷ nguyên số, việc quản lý khối lượng thông tin khổng lồ, chẳng hạn như thư điện tử, đã trở thành một thách thức lớn đối với người dùng cá nhân và tổ chức. Mỗi ngày, khi truy cập hộp thư đến trên các nền tảng như Gmail, người dùng phải đối mặt với hàng loạt thư điện tử từ nhiều nguồn khác nhau, bao gồm thư từ đồng nghiệp, đối tác, thông báo hệ thống, cũng như thư rác và thư quảng cáo không mong muốn [54]. Trong số đó, chỉ một phần nhỏ thư điện tử thực sự quan trọng, chẳng hạn như thư công việc, giao dịch tài chính, hoặc thông báo cá nhân, trong khi phần lớn còn lại thường gây mất thời gian và làm giảm hiệu suất làm việc nếu phải phân loại thủ công. Việc xác định nhanh chóng và chính xác các thư điện tử quan trọng giữa dòng chảy thông tin liên tục là một nhu cầu cấp thiết trong môi trường số hiện đại.

Để giải quyết vấn đề này, các hệ thống tự động phân loại thư điện tử (*automatic email classification*) đã được phát triển, cho phép phân biệt hiệu quả giữa thư hợp lệ và thư rác. Các hệ thống này tận dụng các thuật toán học máy để học từ hành vi của người dùng, chẳng hạn như khi người dùng đánh dấu một thư điện tử là “thư rác” hoặc “thư hợp lệ”. Thông qua các thao tác này, hệ thống không chỉ ghi nhận hành động của người dùng mà còn cập nhật mô hình, từ đó cải thiện khả năng phân loại chính xác các thư điện tử mới trong tương lai [55].

Những hệ thống này thậm chí có khả năng thực hiện phân loại trong thời gian thực, áp dụng ngay lập tức các quy tắc đã học được đối với thư điện tử vừa nhận, giúp giảm thiểu sự can thiệp thủ công và nâng cao trải nghiệm người dùng. Ví dụ, khi người dùng đánh dấu một thư điện tử chứa các từ như “khuyến mãi” hoặc “miễn phí” là thư rác, hệ thống sẽ học cách nhận diện các mẫu tương tự trong các thư điện tử tiếp theo, từ đó tự động chuyển chúng vào thư mục thích hợp.

Trong số các thuật toán học máy được sử dụng cho bài toán này, *thuật toán Naïve Bayes* nổi bật như một phương pháp mạnh mẽ và phổ biến, đặc biệt trong các ứng dụng phân loại văn bản như phân loại thư điện tử [56]. Thuật toán này hoạt động bằng cách phân tích các đặc trưng (chẳng hạn như các từ hoặc cụm từ) trong nội dung thư điện tử, học từ dữ liệu được người dùng cung cấp để dự đoán xác suất một thư điện tử thuộc vào một lớp cụ thể, chẳng hạn thư rác hoặc thư hợp lệ. Điểm mạnh của thuật toán Naïve Bayes nằm ở tính đơn giản, hiệu quả tính toán, và khả năng xử lý tốt các tập dữ liệu văn bản lớn với số lượng đặc trưng (từ vựng) lớn [57]. Hơn nữa, thuật toán hỗ trợ học tiệm tiến (*incremental learning*), cho phép mô hình cập nhật liên tục khi người dùng cung cấp thêm dữ liệu mới, chẳng hạn như đánh dấu thêm thư điện tử là thư rác hoặc thư hợp lệ.

4.2 Khái niệm

Thuật toán Naïve Bayes (*Naïve Bayes algorithm*) là một trong những phương pháp học máy đơn giản nhưng mạnh mẽ, thuộc nhóm các kỹ thuật phân loại dựa vào xác suất (*probabilistic classification*). Thuật toán này được xây dựng trên nền tảng của lý thuyết quyết định Bayes, kết hợp với giả định rằng các đặc trưng đầu vào là độc lập có điều kiện khi biết lớp của dữ liệu [58].

Mặc dù giả định về tính độc lập này hiếm khi đúng hoàn toàn trong thực tế — đặc biệt trong các bài toán xử lý văn bản, nơi các đặc trưng (từ ngữ) thường mang tính liên kết ngữ nghĩa chặt chẽ—nhưng nó cho phép đơn giản hóa đáng kể việc tính toán xác suất. Nhờ đó, mô hình có thể huấn luyện và suy diễn hiệu quả ngay cả khi dữ liệu đầu vào có số lượng đặc trưng rất lớn.

Với cấu trúc mô hình đơn giản, khả năng học nhanh và tiêu tốn ít tài nguyên tính toán, thuật toán Naïve Bayes đã trở thành một công cụ hữu ích trong thực hành học máy, đặc biệt trong lĩnh vực phân loại văn bản. Một trong những ứng dụng nổi bật của nó là trong phát hiện thư rác [54], [56], nơi cần xử lý lượng lớn thư điện tử và phân loại chúng một cách nhanh chóng và chính xác. Nhờ tốc độ huấn luyện cao và chi phí bộ nhớ thấp, thuật toán này phù hợp với các ứng dụng thời gian thực và các hệ thống có tài nguyên hạn chế.

4.3 Các định lý xác suất

Thuật toán Naïve Bayes là một phương pháp phân loại dựa trên lý thuyết xác suất, cụ thể là định lý Bayes, cùng với giả định rằng các đặc trưng đầu vào độc lập có điều kiện với nhau khi biết nhãn lớp. Cốt lõi của thuật toán là ước lượng xác suất để một mẫu dữ liệu thuộc về một lớp cụ thể, dựa trên các đặc trưng quan sát được [58].

- Xác suất tiên nghiệm (*Prior Probability*): Ký hiệu $P(c)$, xác suất tiên nghiệm biểu thị xác suất xảy ra của một lớp c trước khi có thông tin về các đặc trưng của dữ liệu. Trong các bài toán phân loại, xác suất tiên nghiệm được ước lượng từ phân bố của các lớp trong tập dữ liệu huấn luyện, phản ánh xác suất tự nhiên của mỗi lớp. Xác suất tiên nghiệm đóng vai trò nền tảng trong việc kết hợp với các đặc trưng quan sát để đưa ra dự đoán chính xác [57].
- Xác suất có điều kiện (*Conditional Probability*): Ký hiệu $P(x_i|c)$, xác suất có điều kiện biểu thị xác suất xuất hiện của một đặc trưng x_i khi lớp c đã được xác định. Trong bối cảnh phân loại văn bản, đặc trưng x_i thường là một từ hoặc cụm từ trong văn bản, và xác suất có điều kiện đo lường mức độ thường xuyên đặc trưng này xuất hiện trong các mẫu thuộc lớp c . Xác suất có điều kiện là thành phần cốt lõi để đánh giá mức độ liên quan của các đặc trưng đối với từng lớp trong mô hình thuật toán Naïve Bayes [54].
- Xác suất kết hợp (*Joint Probability*): Ký hiệu $P(x_1, x_2, \dots, x_n)$, biểu thị xác suất xảy ra đồng thời của các biến ngẫu nhiên hoặc đặc trưng x_1, x_2, \dots, x_n . Trong thuật toán Naïve Bayes, xác suất kết hợp có điều kiện $P(x_1, x_2, \dots, x_n|c)$ thể hiện xác suất đồng thời của các đặc trưng khi biết lớp c . Việc tính toán trực tiếp xác suất này thường rất phức tạp do số lượng tổ hợp đặc trưng tăng theo cấp số nhân, đặc biệt trong các bài toán như phân loại văn bản hoặc chẩn đoán y khoa, nơi có số lượng đặc trưng lớn [10]. Hơn nữa, sự phụ thuộc giữa các đặc trưng làm tăng thêm độ phức tạp, vì phải mô hình hóa các mối quan hệ này. Để giải quyết, thuật toán Naïve Bayes áp dụng giả định độc lập có điều kiện, tức $P(x_1, x_2, \dots, x_n|c) = P(x_1|c) \cdot P(x_2|c) \cdot \dots \cdot P(x_n|c)$, giúp đơn giản hóa tính toán nhưng có thể giảm độ chính xác khi các đặc trưng thực tế không độc lập.

- Giả định độc lập Naïve (*Naïve Independence Assumption*): Để đơn giản hóa việc tính toán Xác suất kết hợp, thuật toán Naïve Bayes giả định rằng các đặc trưng x_1, x_2, \dots, x_n là độc lập có điều kiện (*conditionally independent*) khi lớp c đã biết. Dựa trên giả định này, Xác suất kết hợp được biểu diễn dưới dạng tích các xác suất có điều kiện của từng đặc trưng:

$$P(x_1, x_2, \dots, x_n | c) = \prod_{i=1}^n P(x_i | c) \quad (4.1)$$

Giả định độc lập Naïve cho phép giảm đáng kể độ phức tạp tính toán, đặc biệt khi xử lý các tập dữ liệu có số lượng đặc trưng lớn, chẳng hạn như trong phân loại văn bản. Ngoài ra, nó còn cho phép thuật toán Naïve Bayes duy trì hiệu suất tốt trong các bài toán phân loại với dữ liệu cao chiều, chẳng hạn như phân loại thư rác [25], [58]. Mặc dù giả định này có thể không hoàn toàn phù hợp trong một số trường hợp thực tế, khi các đặc trưng có mối quan hệ phụ thuộc (chẳng hạn, các từ trong văn bản có liên kết ngữ nghĩa), thuật toán Naïve Bayes vẫn đạt hiệu quả cao trong nhiều ứng dụng nhờ khả năng cân bằng giữa các xác suất và tính đơn giản của mô hình [59].

- Xác suất hậu nghiệm (*Posterior Probability*): Ký hiệu $P(c|x_1, x_2, \dots, x_n)$, xác suất hậu nghiệm biểu thị xác suất một mẫu dữ liệu thuộc lớp c khi đã biết tập hợp các đặc trưng x_1, x_2, \dots, x_n . Xác suất hậu nghiệm được tính toán dựa trên định lý Bayes như sau:

$$P(C = c | X_1 = x_1, \dots, X_n = x_n) = \frac{P(X_1 = x_1, \dots, X_n = x_n | C = c) \cdot P(C = c)}{P(X_1 = x_1, \dots, X_n = x_n)} \quad (4.2)$$

Trong đó:

- $P(X_1 = x_1, \dots, X_n = x_n | C = c)$: Xác suất kết hợp có điều kiện, được đơn giản hóa trong Naïve Bayes nhờ giả định độc lập có điều kiện: $\prod_{i=1}^n P(X_i = x_i | C = c)$.
- $P(C = c)$: Xác suất tiên nghiệm của lớp c .
- $P(X_1 = x_1, \dots, X_n = x_n)$: Xác suất biên (*Marginal Probability*), hằng số chuẩn hóa, tính bằng: $\sum_c P(X_1 = x_1, \dots, X_n = x_n | C = c)P(C = c)$.

Trong quá trình phân loại, thuật toán Naïve Bayes so sánh xác suất hậu nghiệm của các lớp để chọn lớp có giá trị cao nhất làm dự đoán cuối cùng. Để đảm bảo độ chính xác và ổn định tính toán, các triển khai thực tế thường sử dụng logarithm để chuyển phép nhân các xác suất thành phép cộng, tránh các vấn đề liên quan đến số học khi xử lý nhiều giá trị xác suất nhỏ [54].

4.4 Nguyên lý phân loại dựa vào xác suất

Phân loại dựa vào xác suất là một phương pháp học máy sử dụng các nguyên tắc xác suất để gán nhãn lớp cho dữ liệu đầu vào dựa trên các đặc trưng quan sát được. Trong thuật toán Naïve Bayes, quá trình phân loại được thực hiện thông qua Maximum A Posteriori – phương pháp Tối đa Hậu nghiệm (MAP), nhằm chọn lớp có xác suất hậu nghiệm cao nhất [58]

Phương pháp MAP được áp dụng để xác định lớp tối ưu cho một mẫu dữ liệu đầu vào x , với $x = \{x_1, x_2, \dots, x_n\}$ biểu thị tập hợp các đặc trưng (chẳng hạn, các từ trong văn bản) và $\{c_1, c_2, \dots, c_L\}$ là tập hợp L lớp có thể có. Quy tắc này được định nghĩa như sau:

4.4. NGUYÊN LÝ PHÂN LOẠI DỰA VÀO XÁC SUẤT

$$c^* = \arg \max_{c_i} P(c_i|x), \quad \forall i = 1, 2, \dots, L \quad (4.3)$$

Trong đó, $P(c_i|x)$ là xác suất hậu nghiệm của lớp c_i khi biết dữ liệu x , và c^* là lớp có giá trị xác suất hậu nghiệm cao nhất. Phương pháp MAP đảm bảo rằng quyết định phân loại tối ưu hóa khả năng chọn đúng lớp dựa trên thông tin đặc trưng quan sát được [10].

Dể tính xác suất hậu nghiệm $P(c_i|x)$, thuật toán Naïve Bayes sử dụng định lý Bayes:

$$P(c_i|x) = \frac{P(x|c_i) \cdot P(c_i)}{P(x)}, \quad \forall i = 1, 2, \dots, L \quad (4.4)$$

Trong đó:

- $P(c_i|x)$: Xác suất hậu nghiệm của lớp c_i khi biết dữ liệu x .
- $P(x|c_i)$: Xác suất có điều kiện, hay likelihood biểu thị xác suất xuất hiện của dữ liệu x khi lớp c_i đã biết.
- $P(c_i)$: Xác suất tiên nghiệm biểu thị xác suất xảy ra của lớp c_i trước khi quan sát dữ liệu.
- $P(x)$: Xác suất chứng cứ là xác suất tổng quát của dữ liệu x , được tính bằng:

$$P(x) = \sum_{i=1}^L P(x|c_i) \cdot P(c_i) \quad (4.5)$$

Trong phương pháp MAP, vì xác suất chứng cứ $P(x)$ là một hằng số chuẩn hóa không đổi đối với tất cả các lớp c_i , nó có thể được bỏ qua khi so sánh các xác suất hậu nghiệm. Do đó, phương pháp MAP được đơn giản hóa thành:

$$c^* = \arg \max_{c_i} P(x|c_i) \cdot P(c_i) \quad (4.6)$$

Dể tính xác suất có điều kiện $P(x|c_i)$, thuật toán Naïve Bayes áp dụng giả định độc lập Naïve, trong đó các đặc trưng x_1, x_2, \dots, x_n được giả định là độc lập có điều kiện khi biết lớp c_i . Điều này cho phép biểu diễn:

$$P(x|c_i) = P(x_1, x_2, \dots, x_n|c_i) = \prod_{j=1}^n P(x_j|c_i) \quad (4.7)$$

Kết hợp với phương pháp MAP, lớp được chọn là:

$$c^* = \arg \max_{c_i} P(c_i) \cdot \prod_{j=1}^n P(x_j|c_i) \quad (4.8)$$

Ví dụ 4.1. Xét bài toán phân lớp cụ thể như sau:

Cho tập dữ liệu chứa thông tin về *Giới tính*, *Tiền sử gia đình mắc bệnh tiểu đường*, *Chế độ ăn uống*, *Hoạt động thể chất*, *Mắc bệnh tiểu đường* từ Cơ sở dữ liệu *Dữ liệu khám sức khỏe về bệnh tiểu đường*. Với một bộ giá trị đầu vào của khách hàng khám bệnh về tiểu đường tại bệnh viện, hãy dự đoán xem người đó có mắc bệnh tiểu đường hay không?

Dữ liệu của ban đầu thu thập được thể hiện trong Bảng 4.1.

$P(\text{Mắc bệnh} = \text{Có} | \text{Giới tính} = \text{Nữ}, \text{Tiền sử gia đình} = \text{Không}, \text{Chế độ ăn uống} = \text{Kém}, \text{Hoạt động thể chất} = \text{Bình thường})$

Bảng 4.1: Dữ liệu huấn luyện về bệnh tiểu đường

RID	Giới tính	Tiền sử gia đình	Chế độ ăn uống	Hoạt động thể chất	Mắc bệnh
1	Nam	Có	Kém	Ít vận động	Có
2	Nữ	Không	Tốt	Vận động nhiều	Không
3	Nam	Không	Kém	Vận động nhiều	Không
4	Nữ	Có	Tốt	Vận động nhiều	Không
5	Nam	Không	Kém	Ít vận động	Có
6	Nữ	Có	Kém	Ít vận động	Có
7	Nam	Không	Tốt	Vận động nhiều	Không
8	Nữ	Có	Tốt	Bình thường	Không
9	Nam	Có	Tốt	Ít vận động	Có
10	Nữ	Không	Tốt	Vận động nhiều	Không
11	Nam	Không	Tốt	Vận động nhiều	Không
12	Nữ	Có	Kém	Bình thường	Có

Bài toán nêu trên là bài toán phân lớp cho m lớp C_1, C_2, \dots, C_m . Trong trường hợp cụ thể này, $m = 2$ lớp. Giả sử có một điểm dữ liệu $X = (\text{Giới tính} = \text{Nữ}, \text{Tiền sử gia đình} = \text{Không}, \text{Chế độ ăn uống} = \text{Kém}, \text{Hoạt động thể chất} = \text{Bình thường})$. Ta cần tính xác suất để điểm dữ liệu này rơi vào class C_i .

Ta sẽ áp dụng luật MAP vào bài toán này, cụ thể ta cần tính $P(y = C_i|X)$ hay $P(C_i|X), i = 1..m$.

Thông qua công thức này, ta có thể xác định điểm dữ liệu X thuộc lớp nào bằng cách chọn lớp C_i có xác suất hậu nghiệm $P(C_i|X)$ cao nhất. Ta bắt đầu với:

$$c = \arg \max_{i=1..m} P(C_i|X), \quad (4.9)$$

$$= \arg \max_{i=1..m} \frac{P(X|C_i)P(C_i)}{P(X)}, \quad (4.10)$$

$$\propto \arg \max_{i=1..m} P(X|C_i)P(C_i). \quad (4.11)$$

Trong đó:

- $P(C_i)$: Xác suất tiên nghiệm của lớp C_i , biểu thị xác suất xảy ra của lớp C_i trước khi biết điểm dữ liệu X .
- $P(X | C_i)$: Xác suất có điều kiện, biểu thị khả năng xuất hiện điểm dữ liệu X nếu nó thuộc lớp C_i .
- $P(X)$: Xác suất bìen của điểm dữ liệu X , là một hằng số chung cho mọi lớp C_i trong quá trình so sánh xác suất hậu nghiệm.

Tại bước (4.11), ta bỏ qua $P(X)$ vì xác suất này không phụ thuộc vào lớp C_i , tức là nó giống nhau cho mọi lớp. Do đó, việc tối đa hóa $P(C_i|X)$ trở thành tối đa hóa tỷ lệ $P(X|C_i)P(C_i)$.

Tiếp theo, xác suất có điều kiện $P(X|C_i)$ thể hiện khả năng xuất hiện của X trong lớp C_i . Nếu điểm dữ liệu X bao gồm nhiều thuộc tính (ví dụ: $X = (x_1, x_2, \dots, x_n)$), thì ta cần tính:

4.4. NGUYÊN LÝ PHÂN LOẠI DỰA VÀO XÁC SUẤT

$$P(X|C_i) = P(x_1, x_2, \dots, x_n|C_i). \quad (4.12)$$

Tuy nhiên, việc tính toán trực tiếp $P(x_1, x_2, \dots, x_n|C_i)$ thường không khả thi vì các thuộc tính x_1, x_2, \dots, x_n có thể phụ thuộc lẫn nhau. Điều này làm tăng độ phức tạp tính toán.

Đối với bộ phân loại Naïve Bayes, ta sẽ giả sử các thuộc tính trong X độc lập xác suất có điều kiện lớp $C_i, i = 1..m$. Khi đó:

$$P(x_1, x_2, \dots, x_n|C) = P(x_1|x_2, \dots, x_n, C)P(x_2, \dots, x_n|C) \quad (4.13)$$

$$= P(x_1|C)P(x_2, \dots, x_n|C) \quad (4.14)$$

$$= P(x_1|C)P(x_2|C) \dots P(x_n|C) \quad (4.15)$$

$$= \prod_{i=1}^n P(x_i|C) \quad (4.16)$$

Vậy, với một điểm dữ liệu X , lớp của nó sẽ được xác định bởi:

$$c = \arg \max P(C_i) \prod_{j=1}^n P(x_j|C_i), i = 1..m \quad (4.17)$$

Giả thiết về sự độc lập xác suất có điều kiện của các thuộc tính được gọi là *Naïve Bayes*. Bộ phân lớp dựa trên giả thiết trên là *Naïve Bayes Classifier*.

Quay trở lại bài toán ban đầu, ta sẽ tiến hành xác định xem bệnh nhân có bị mắc bệnh tiểu đường hay không theo các bước sau:

Áp dụng công thức 4.17, ta có:

$P(\text{Mắc bệnh} = \text{Có} | \text{Giới tính} = \text{Nữ}, \text{Tiền sử gia đình} = \text{Không}, \text{Chế độ ăn uống} = \text{Kém}, \text{Hoạt động thể chất} = \text{Bình thường})$

$$\begin{aligned} P(X | \text{Mắc bệnh} = \text{Có}) \cdot P(\text{Mắc bệnh} = \text{Có}) &= P(\text{Mắc bệnh} = \text{Có}) \cdot P(\text{Giới tính} = \text{Nữ} | \text{Mắc bệnh} = \text{Có}) \\ &\quad \cdot P(\text{Tiền sử gia đình} = \text{Không} | \text{Mắc bệnh} = \text{Có}) \\ &\quad \cdot P(\text{Chế độ ăn uống} = \text{Kém} | \text{Mắc bệnh} = \text{Có}) \\ &\quad \cdot P(\text{Hoạt động thể chất} = \text{Bình thường} | \text{Mắc bệnh} = \text{Có}) \end{aligned}$$

Nhìn vào bảng dữ liệu, ta tính được:

$$P(\text{Mắc bệnh} = \text{Có}) = \frac{5}{12} \approx 0.4167,$$

$$P(\text{Giới tính} = \text{Nữ} | \text{Mắc bệnh} = \text{Có}) = \frac{2}{5} = 0.4 \quad (2/5 \text{ mẫu có bệnh là nữ}),$$

$$P(\text{Tiền sử gia đình} = \text{Không} | \text{Mắc bệnh} = \text{Có}) = \frac{2}{5} = 0.4 \quad (2/5 \text{ mẫu có bệnh không có tiền sử}),$$

$$P(\text{Chế độ ăn uống} = \text{Kém} | \text{Mắc bệnh} = \text{Có}) = \frac{4}{5} = 0.8 \quad (4/5 \text{ mẫu có bệnh ăn uống kém}),$$

$$P(\text{Hoạt động thể chất} = \text{Bình thường} | \text{Mắc bệnh} = \text{Có}) = \frac{1}{5} = 0.2 \quad (1/5 \text{ mẫu có bệnh hoạt động bình thường}).$$

Thay vào:

$$\begin{aligned} P(\text{Mắc bệnh} = \text{Có}|X) &\propto \frac{5}{12} \cdot \frac{2}{5} \cdot \frac{2}{5} \cdot \frac{4}{5} \cdot \frac{1}{5} \\ &= 0.4167 \cdot 0.4 \cdot 0.4 \cdot 0.8 \cdot 0.2 \approx 0.010667. \end{aligned}$$

Tương tự:

$P(\text{Mắc bệnh} = \text{Không} | \text{Giới tính} = \text{Nữ}, \text{Tiền sử gia đình} = \text{Không}, \text{Chế độ ăn uống} = \text{Kém}, \text{Hoạt động thể chất} = \text{Bình thường})$

$$\begin{aligned} P(\text{Mắc bệnh} = \text{Không}) &= \frac{7}{12} \approx 0.5833, \\ P(\text{Giới tính} = \text{Nữ} | \text{Không}) &= \frac{3}{7} \approx 0.4286 \quad (3/7 \text{ mẫu không bệnh là nữ}), \\ P(\text{Tiền sử gia đình} = \text{Không} | \text{Không}) &= \frac{5}{7} \approx 0.7143 \quad (5/7 \text{ mẫu không bệnh không có tiền sử}), \\ P(\text{Chế độ ăn uống} = \text{Kém} | \text{Không}) &= \frac{1}{7} \approx 0.1429 \quad (1/7 \text{ mẫu không bệnh ăn uống kém}), \\ P(\text{Hoạt động thể chất} = \text{Bình thường} | \text{Không}) &= \frac{1}{7} \approx 0.1429 \quad (1/7 \text{ mẫu không bệnh hoạt động bình thường}). \end{aligned}$$

Thay vào:

$$\begin{aligned} P(\text{Mắc bệnh} = \text{Không}|X) &\propto \frac{7}{12} \cdot \frac{3}{7} \cdot \frac{5}{7} \cdot \frac{1}{7} \cdot \frac{1}{7} \\ &= 0.5833 \cdot 0.4286 \cdot 0.7143 \cdot 0.1429 \cdot 0.1429 \approx 0.00364. \end{aligned}$$

Bình thường hóa:

$$\begin{aligned} P(\text{Mắc bệnh} = \text{Có}|X) &= \frac{0.010667}{0.010667 + 0.00364} \approx 0.7457, \\ P(\text{Mắc bệnh} = \text{Không}|X) &= \frac{0.00364}{0.010667 + 0.00364} \approx 0.2543. \end{aligned}$$

Sau các bước trên, ta tính xong xác suất và nhận thấy rằng $P(\text{Mắc bệnh} = \text{Có} | \text{Giới tính} = \text{Nữ}, \text{Tiền sử gia đình} = \text{Không}, \text{Chế độ ăn uống} = \text{Kém}, \text{Hoạt động thể chất} = \text{Bình thường}) > P(\text{Mắc bệnh} = \text{Không} | \text{Giới tính} = \text{Nữ}, \text{Tiền sử gia đình} = \text{Không}, \text{Chế độ ăn uống} = \text{Kém}, \text{Hoạt động thể chất} = \text{Bình thường})$, dự đoán của chúng ta là bệnh nhân nữ này có nguy cơ bị tiểu đường.

4.5 Xác suất 0 - Zero Probability

Mặc dù thuật toán Naïve Bayes nổi bật với tính hiệu quả và đơn giản trong các bài toán phân loại, công thức phân loại dựa trên quy tắc Tối đa Hậu nghiệm (*Maximum A Posteriori - MAP*), như được trình bày trong công thức (4.17), vẫn tồn tại một hạn chế đáng kể liên quan đến vấn đề xác suất bằng 0 (*zero probability*). Hạn chế này xảy ra khi một đặc trưng không xuất hiện trong tập huấn luyện cho một lớp cụ thể, dẫn đến xác suất có điều kiện bằng 0, làm cho toàn bộ tích xác suất trong công thức phân loại trở thành 0 [10], [58].

4.5.1 Nguyên nhân của xác suất bằng 0

Trong thuật toán Naïve Bayes, xác suất hậu nghiệm của lớp C_i cho một điểm dữ liệu $X = (x_1, x_2, \dots, x_n)$ được tính dựa trên công thức:

$$P(C_i|X) \propto P(C_i) \cdot \prod_{j=1}^n P(x_j|C_i)$$

Trong đó, $P(x_j|C_i)$ là xác suất có điều kiện của đặc trưng x_j khi biết lớp C_i , được ước lượng từ tập huấn luyện theo công thức:

$$P(x_j|C_i) = \frac{n_{c_j}}{n_i}$$

Trong đó:

- n_{c_j} : Số mẫu trong tập huấn luyện thuộc lớp C_i và có giá trị đặc trưng x_j .
- n_i : Tổng số mẫu trong tập huấn luyện thuộc lớp C_i .

Vấn đề xác suất bằng 0 xảy ra khi $n_{c_j} = 0$, tức là đặc trưng x_j không xuất hiện trong bất kỳ mẫu nào của lớp C_i trong tập huấn luyện. Khi đó, $P(x_j|C_i) = 0$, dẫn đến toàn bộ tích $\prod_{j=1}^n P(x_j|C_i) = 0$, khiến xác suất hậu nghiệm $P(C_i|X) = 0$, bất kể giá trị của xác suất tiên nghiệm $P(C_i)$ hoặc các xác suất có điều kiện khác. Điều này làm cho thuật toán Naïve Bayes không thể phân loại chính xác, vì tất cả các lớp có thể bị loại bỏ nếu một đặc trưng có xác suất bằng 0 [59].

Ví dụ, xét một tập dữ liệu y tế với đặc trưng *Hoạt động thể chất* (*Physical Activity*) có ba giá trị: *Ít vận động*, *Bình thường*, và *Vận động nhiều*. Giả sử trong tập huấn luyện với 1000 mẫu thuộc lớp *Mắc bệnh* = Có, có 750 mẫu với *Hoạt động thể chất* = Ít vận động, 250 mẫu với *Bình thường*, và 0 mẫu với *Vận động nhiều*. Khi đó:

$$P(\text{Hoạt động thể chất} = \text{Vận động nhiều} | \text{Có}) = \frac{0}{1000} = 0$$

Nếu một bệnh nhân mới có đặc trưng *Hoạt động thể chất* = Vận động nhiều, tích xác suất $\prod_{j=1}^n P(x_j|\text{Có})$ sẽ bằng 0, dẫn đến việc lớp *Mắc bệnh* = Có bị loại bỏ, ngay cả khi các đặc trưng khác hỗ trợ mạnh cho lớp này [57].

4.5.2 Phương pháp làm mượt Laplace

Để khắc phục vấn đề xác suất bằng 0, kỹ thuật làm mượt Laplace được sử dụng. Phương pháp này điều chỉnh ước lượng xác suất có điều kiện bằng cách thêm một hằng số dương nhỏ k (thường là 1) vào tử số và điều chỉnh mẫu số tương ứng để đảm bảo tổng xác suất bằng 1. Công thức làm mượt Laplace được biểu diễn như sau:

$$P(x_j|C_i) = \frac{n_{c_j} + k}{n_i + k \cdot |X_j|} \quad (4.18)$$

Trong đó:

- n_{c_j} : Số mẫu trong lớp C_i có giá trị đặc trưng x_j .
- n_i : Tổng số mẫu trong lớp C_i .

- $|X_j|$: Số giá trị có thể có của đặc trưng x_j .
- k : Hằng số làm mượt, thường chọn $k = 1$ (được gọi là làm mượt Laplace chuẩn).

Áp dụng vào ví dụ trên với $k = 1$ và $|X_j| = 3$ (do *Hoạt động thẻ chất* có ba giá trị: *Ít vận động*, *Bình thường*, *Vận động nhiều*):

$$P(\text{Hoạt động thẻ chất} = \text{Ít vận động} | \text{Có}) = \frac{750 + 1}{1000 + 1 \cdot 3} = \frac{751}{1003} \approx 0.7488$$

$$P(\text{Hoạt động thẻ chất} = \text{Bình thường} | \text{Có}) = \frac{250 + 1}{1000 + 1 \cdot 3} = \frac{251}{1003} \approx 0.2502$$

$$P(\text{Hoạt động thẻ chất} = \text{Vận động nhiều} | \text{Có}) = \frac{0 + 1}{1000 + 1 \cdot 3} = \frac{1}{1003} \approx 0.0010$$

Kỹ thuật làm mượt Laplace đảm bảo rằng không có xác suất nào bằng 0, cho phép thuật toán Naïve Bayes đưa ra dự đoán ngay cả khi một đặc trưng không xuất hiện trong tập huấn luyện. Điều này đặc biệt quan trọng trong các bài toán phân loại văn bản, nơi các đặc trưng mới có thể xuất hiện trong dữ liệu kiểm tra [54].

4.5.3 Ý nghĩa

Phương pháp làm mượt Laplace giải quyết vấn đề xác suất bằng 0 trong thuật toán Naïve Bayes giúp cải thiện tính ổn định khi xử lý các tập dữ liệu nhỏ hoặc thừa thớt. Bằng cách thêm hằng số k , kỹ thuật này giả định mỗi giá trị đặc trưng có ít nhất một lần xuất hiện “ảo”, đảm bảo mọi xác suất có điều kiện đều lớn hơn 0, từ đó nâng cao hiệu quả phân loại trong các bài toán [10]. Tuy nhiên, giá trị k cần được chọn cẩn thận, vì nếu quá lớn, có thể làm phẳng quá mức các xác suất, dẫn đến giảm độ chính xác của ước lượng [58].

4.6 Đánh giá mô hình

Sau khi hoàn thành việc xây dựng mô hình, bước quan trọng tiếp theo là đánh giá hiệu quả hoạt động của mô hình để đảm bảo nó đáp ứng được mục tiêu đặt ra. Trong bài toán phân lớp, việc đánh giá thường được thực hiện thông qua bộ dữ liệu kiểm thử. Đây là tập dữ liệu độc lập với bộ dữ liệu huấn luyện, được sử dụng để kiểm tra khả năng tổng quát hóa của mô hình khi xử lý các dữ liệu chưa từng gặp.

Quy trình đánh giá bắt đầu bằng việc đưa bộ dữ liệu kiểm thử vào mô hình. Mô hình sẽ dựa trên các đặc trưng đầu vào của bộ dữ liệu để đưa ra dự đoán về lớp mà mỗi mẫu dữ liệu thuộc về. Kết quả dự đoán này được gọi là tập lớp dự đoán. Đồng thời, bộ dữ liệu kiểm thử cũng đi kèm với tập lớp thực tế, tức là các nhãn đúng của từng mẫu dữ liệu, được sử dụng làm cơ sở so sánh.

Việc đánh giá hiệu quả của mô hình được thực hiện bằng cách so sánh tập lớp dự đoán với tập lớp thực tế. Dựa trên sự so sánh này, các chỉ số đánh giá hiệu suất như độ chính xác (accuracy), độ nhạy (recall), độ đặc hiệu (specificity), chỉ số F1 (F1-score), độ chính xác dự đoán (precision) và ma trận nhầm lẫn (confusion matrix) được tính toán để cung cấp cái nhìn chi tiết về cách mô hình hoạt động.

Chẳng hạn, độ chính xác cho biết tỷ lệ dự đoán đúng trên tổng số mẫu, trong khi độ nhạy đo lường khả năng mô hình nhận diện chính xác các mẫu thuộc một lớp cụ thể. Precision đo lường khả năng mô hình đưa ra dự đoán đúng cho các mẫu dương tính, trong khi F1-score kết

4.7. MA TRẬN CONFUSION

hợp giữa precision và recall để đánh giá hiệu quả chung của mô hình trong các bài toán không cân bằng dữ liệu. Ma trận nhầm lẫn cung cấp thông tin trực quan về các lỗi mà mô hình gặp phải khi phân loại.

Nhờ vào những chỉ số đánh giá này, ta có thể xác định điểm mạnh và điểm yếu của mô hình. Nếu mô hình đạt hiệu suất cao, nó có thể được triển khai vào thực tế. Ngược lại, nếu hiệu suất không đạt yêu cầu, ta cần phân tích sâu hơn để tìm ra nguyên nhân, từ đó thực hiện các điều chỉnh, chẳng hạn như cải tiến dữ liệu đầu vào, thay đổi thuật toán hoặc tinh chỉnh các siêu tham số của mô hình.

4.7 Ma trận confusion

Trong các bài toán phân loại, việc đánh giá hiệu suất của mô hình như thuật toán Naïve Bayes đòi hỏi sự hiểu biết chi tiết về cách các mẫu được phân loại vào từng lớp. Chỉ xét độ chính xác (*accuracy*) tổng quát không đủ để đánh giá hiệu quả, vì nó không cung cấp thông tin về các lỗi phân loại cụ thể hoặc hiệu suất trên từng lớp, đặc biệt trong các bài toán mất cân bằng lớp (*class imbalance*). Ma trận confusion (*confusion matrix*) là một công cụ quan trọng giúp trực quan hóa và phân tích hiệu suất mô hình bằng cách thể hiện số lượng mẫu được phân loại đúng và sai cho từng lớp. Phần này trình bày khái niệm Ma trận confusion, các chỉ số hiệu suất liên quan, và cách xử lý vấn đề mất cân bằng lớp, với trọng tâm là ứng dụng trong thuật toán Naïve Bayes [10].

4.7.1 Khái niệm

Ma trận nhầm lẫn (*confusion matrix*) là một bảng biểu diễn số lượng mẫu được phân loại đúng và sai theo từng lớp, thường được sử dụng trong các bài toán phân loại nhị phân hoặc đa lớp. Trong trường hợp phân loại nhị phân, ma trận confusion được xây dựng dựa trên hai lớp: lớp dương tính (*positive*) và lớp âm tính (*negative*). Bảng 4.2 thể hiện cấu trúc của ma trận confusion cho bài toán *phân loại nhị phân*:

Bảng 4.2: Ma trận confusion cho bài toán phân loại nhị phân

Lớp thật	dương tính (<i>Positive</i>)	âm tính (<i>Negative</i>)
dương tính (<i>Positive</i>)	<i>True Positive</i> (TP)	<i>False Negative</i> (FN)
âm tính (<i>Negative</i>)	<i>False Positive</i> (FP)	<i>True Negative</i> (TN)

Trong đó:

- Dương tính thật (*True Positive* - TP): Số mẫu thực sự thuộc lớp dương tính và được dự đoán đúng là dương tính.
- Âm tính giả (*False Negative* - FN): Số mẫu thực sự thuộc lớp dương tính nhưng bị dự đoán sai là âm tính.
- Dương tính giả (*False Positive* - FP): Số mẫu thực sự thuộc lớp âm tính nhưng bị dự đoán sai là dương tính.
- Âm tính thật (*True Negative* - TN): Số mẫu thực sự thuộc lớp âm tính và được dự đoán đúng là âm tính.

Đối với bài toán phân loại đa lớp (*multi-class classification*) với m lớp, ma trận confusion là một ma trận $m \times m$, trong đó phần tử $CM_{i,j}$ biểu thị số mẫu thuộc lớp thực tế i nhưng được dự đoán là lớp j . Ma trận confusion cung cấp cái nhìn chi tiết về hiệu suất của mô hình trên từng lớp, giúp xác định các lớp thường bị nhầm lẫn và các lỗi phân loại cụ thể [58].

Ví dụ 4.2. Để minh họa, xét bài toán *phân loại* bệnh tiểu đường (xem 4.1) với hai lớp: *Mắc bệnh* = Có (*Positive*) và *Mắc bệnh* = Không (*Negative*). Giả sử mô hình thuật toán Naïve Bayes được kiểm tra trên một *tập kiểm thử* (*test set*) gồm 5000 mẫu, kết quả được thể hiện trong Bảng 4.3.

Bảng 4.3: Ma trận confusion cho bài toán phân loại bệnh tiểu đường

Lớp thật	<i>Mắc bệnh</i> = Có	<i>Mắc bệnh</i> = Không	Tổng cộng
<i>Mắc bệnh</i> = Có	1200 (TP)	300 (FN)	1500
<i>Mắc bệnh</i> = Không	200 (FP)	3300 (TN)	3500
Tổng cộng	1400	3600	5000

Từ Bảng 4.3, ta thấy:

- 1200 mẫu thực sự thuộc lớp *Mắc bệnh* = Có được dự đoán đúng (TP).
- 300 mẫu thực sự thuộc lớp *Mắc bệnh* = Có bị dự đoán sai là *Không* (FN).
- 200 mẫu thực sự thuộc lớp *Mắc bệnh* = Không bị dự đoán sai là *Có* (FP).
- 3300 mẫu thực sự thuộc lớp *Mắc bệnh* = Không được dự đoán đúng (TN).

4.7.2 Các chỉ số hiệu suất

Dựa trên Ma trận confusion, các chỉ số hiệu suất sau được tính toán để đánh giá mô hình:

- **Độ chính xác tổng thể** (*Accuracy*): Tỷ lệ mẫu được phân loại đúng trên tổng số mẫu:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

Từ ví dụ: $Accuracy = \frac{1200+3300}{5000} = 0.9$ (90%).

- **Tỉ lệ lỗi** (*Error Rate*): Tỷ lệ mẫu bị phân loại sai:

$$Error Rate = 1 - Accuracy = \frac{FP + FN}{TP + TN + FP + FN}$$

Từ ví dụ: $Error Rate = \frac{200+300}{5000} = 0.1$ (10%).

- **Độ nhạy** (*Sensitivity*) hoặc **độ thu hồi** (*Recall*): Tỷ lệ mẫu dương tính được dự đoán đúng trên tổng số mẫu thực sự dương tính:

$$Sensitivity = Recall = \frac{TP}{TP + FN}$$

4.7. MA TRẬN CONFUSION

Từ ví dụ: $Sensitivity = \frac{1200}{1200+300} = 0.8$ (80%). Điều này cho thấy mô hình nhận diện đúng 80% bệnh nhân thực sự mắc bệnh tiểu đường, nhưng bỏ sót 20% (FN), có thể nghiêm trọng trong bối cảnh y tế [60].

- **Độ đặc hiệu (Specificity):** Tỷ lệ mẫu âm tính được dự đoán đúng trên tổng số mẫu thực sự âm tính:

$$Specificity = \frac{TN}{TN + FP}$$

Từ ví dụ: $Specificity = \frac{3300}{3300+200} = 0.943$ (94.3%). Điều này cho thấy mô hình hiệu quả trong việc xác định các bệnh nhân không mắc bệnh.

- **Độ chính xác dự đoán (Precision):** Tỷ lệ mẫu được dự đoán là dương tính thực sự thuộc lớp dương tính:

$$Precision = \frac{TP}{TP + FP}$$

Từ ví dụ: $Precision = \frac{1200}{1200+200} \approx 0.857$ (85.7%).

- **Chỉ số F1 (F1-score):** Trung bình điều hòa của độ chính xác dự đoán và độ thu hồi, được tính bằng:

$$F1\text{-score} = 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall}$$

Từ ví dụ: $F1\text{-score} = 2 \cdot \frac{0.857 \cdot 0.8}{0.857 + 0.8} \approx 0.828$ (82.8%).

- **Chỉ số F_β (F_β -score):** Phiên bản tổng quát của chỉ số F1, trong đó tham số β điều chỉnh trọng số giữa độ chính xác dự đoán và độ thu hồi:

$$F_\beta = (1 + \beta^2) \cdot \frac{Precision \cdot Recall}{\beta^2 \cdot Precision + Recall}$$

Trong đó, $\beta > 1$ ưu tiên độ thu hồi (hữu ích trong các bài toán y tế như phát hiện bệnh), trong khi $\beta < 1$ ưu tiên độ chính xác dự đoán (phù hợp với các bài toán như phát hiện thư rác) [61]. Ví dụ, với $\beta = 2$:

$$F_2 = (1 + 2^2) \cdot \frac{0.857 \cdot 0.8}{2^2 \cdot 0.857 + 0.8} \approx 0.811$$

4.7.3 Vấn đề mất cân bằng lớp

Trong nhiều bài toán thực tiễn, chẳng hạn như phát hiện thư rác, hoặc phát hiện gian lận, số mẫu thuộc lớp dương tính thường ít hơn đáng kể so với lớp âm tính, dẫn đến vấn đề mất cân bằng lớp [61]. Trong trường hợp này, độ chính xác có thể không phản ánh đúng hiệu suất, vì mô hình có xu hướng thiên về lớp chiếm ưu thế (lớp âm tính). Ví dụ, xét một tập dữ liệu với 10,000 giao dịch hợp lệ (âm tính) và 10 giao dịch gian lận (dương tính):

- Mô hình 1: Dự đoán sai 7/10 giao dịch gian lận (FN) và 10/10,000 giao dịch hợp lệ (FP).

$$\text{Accuracy} = \frac{10,000 - 7 - 10}{10,010} \approx 0.999.$$

- Mô hình 2: Dự đoán sai 2/10 giao dịch gian lận (FN) và 102/10,000 giao dịch hợp lệ (FP).

$$\text{Accuracy} = \frac{10,000 - 2 - 102}{10,010} \approx 0.990.$$

Dù mô hình 1 có độ chính xác cao hơn, nó bỏ sót nhiều giao dịch gian lận hơn (7 so với 2), là lỗi nghiêm trọng hơn trong bối cảnh phát hiện gian lận. Điều này cho thấy độ chính xác không đủ để đánh giá trong các bài toán mất cân bằng lớp. Thay vào đó, độ chính xác dự đoán, độ thu hồi, và chỉ số F1 là các chỉ số phù hợp hơn, vì chúng tập trung vào hiệu suất trên lớp dương tính [61].

Trong thuật toán Naïve Bayes, vấn đề mất cân bằng lớp có thể được giảm thiểu bằng các kỹ thuật như lấy mẫu lại (*resampling*), sử dụng trọng số lớp, hoặc điều chỉnh ngưỡng phân loại để ưu tiên lớp dương tính [10]. Ngoài ra, kỹ thuật làm mượt Laplace (xem 4.5) cũng giúp cải thiện ước lượng xác suất trong các lớp thiểu số, từ đó tăng cường hiệu quả của mô hình [57].

4.8 Ưu và nhược điểm của Naïve Bayes

4.8.1 Ưu điểm

Thuật toán Naïve Bayes sở hữu nhiều ưu điểm nổi bật, khiến nó trở thành lựa chọn phổ biến trong các bài toán phân loại:

- Tốc độ dự đoán nhanh: Nhờ giả định độc lập có điều kiện, Thuật toán Naïve Bayes đơn giản hóa việc tính toán các xác suất có điều kiện $p(\mathbf{a}_i|v)$, giảm độ phức tạp tính toán xuống mức tuyến tính [59]. Điều này cho phép thuật toán đưa ra dự đoán nhanh chóng, ngay cả với các tập dữ liệu lớn, khiến nó trở thành lựa chọn lý tưởng cho các ứng dụng yêu cầu thời gian thực như lọc thư rác hoặc phân loại văn bản [54].
- Dễ dàng cập nhật với dữ liệu mới: Thuật toán Naïve Bayes hỗ trợ học tiệm tiến (*incremental learning*), cho phép cập nhật các xác suất có điều kiện mà không cần huấn luyện lại toàn bộ mô hình khi có dữ liệu mới [60]. Điều này đặc biệt hữu ích trong các ứng dụng như lọc thư rác, nơi dữ liệu liên tục thay đổi. Ví dụ, khi người dùng gán nhãn lại một thư điện tử bị phân loại sai, mô hình có thể cập nhật các xác suất một cách nhanh chóng, đảm bảo hiệu suất được cải thiện theo thời gian [56].
- Tiết kiệm tài nguyên: So với các phương pháp học máy phức tạp như *rừng ngẫu nhiên* (*Random Forest*) hoặc *học sâu* (*Deep Learning*), Thuật toán Naïve Bayes yêu cầu ít tài nguyên tính toán hơn, cả về bộ nhớ và sức mạnh xử lý [10]. Điều này giúp giảm chi phí triển khai và bảo trì, đặc biệt trong các môi trường tính toán hạn chế hoặc các ứng dụng yêu cầu xử lý nhanh trên thiết bị có cấu hình thấp [57].
- Hiệu quả với dữ liệu lớn và cấu trúc đơn giản: Cấu trúc đơn giản của Thuật toán Naïve Bayes, dựa trên các xác suất có điều kiện, giúp nó xử lý hiệu quả các tập dữ liệu lớn mà không yêu cầu các phép tính phức tạp [25]. Sự đơn giản này cũng giúp việc triển khai và bảo trì mô hình trở nên dễ dàng, phù hợp cho cả các dự án nhỏ và các ứng dụng quy mô lớn.

4.8. ƯU VÀ NHƯỢC ĐIỂM CỦA NAÏVE BAYES

- Xử lý tốt dữ liệu rác: Thuật toán Naïve Bayes đặc biệt hiệu quả khi làm việc với các đặc trưng rác, chẳng hạn như các từ trong phân loại văn bản hoặc các giá trị nhị phân trong lọc thư rác [56]. Ví dụ, trong phân loại văn bản, thuật toán sử dụng tần suất xuất hiện của các từ để ước lượng xác suất, kết hợp với làm mượt Laplace (xem 4.5.2) để xử lý các từ hiếm gặp, đảm bảo hiệu suất ổn định [54].

4.8.2 Nhược điểm

Mặc dù có nhiều ưu điểm, Thuật toán Naïve Bayes cũng tồn tại một số hạn chế cần được xem xét:

- Giả định độc lập giữa các đặc trưng: Nhược điểm lớn nhất của thuật toán Naïve Bayes là giả định rằng các đặc trưng trong dữ liệu là độc lập với nhau khi biết lớp [59]. Trong thực tế, các đặc trưng thường có mối quan hệ phụ thuộc, chẳng hạn như *chế độ ăn uống và hoạt động thể chất* trong bài toán dự đoán bệnh tiểu đường (xem 4.1). Sự phụ thuộc này có thể làm giảm độ chính xác của mô hình, đặc biệt trong các bài toán phức tạp [60].
- Nhạy cảm với dữ liệu: Hiệu suất của thuật toán Naïve Bayes phụ thuộc nhiều vào chất lượng và tính cân bằng của dữ liệu huấn luyện. Nếu tập huấn luyện có ít mẫu hoặc bị mất cân bằng lớp, mô hình có thể thiên về lớp chiếm ưu thế, dẫn đến dự đoán sai cho các lớp thiểu số. Ví dụ, trong bài toán phát hiện gian lận, nếu số giao dịch gian lận quá ít so với giao dịch hợp lệ, mô hình có thể bỏ sót các trường hợp gian lận [61].
- Hạn chế với biến liên tục: Khi xử lý các đặc trưng liên tục, thuật toán Naïve Bayes thường giả định rằng dữ liệu tuân theo phân phối chuẩn. Nếu giả định này không đúng, độ chính xác của mô hình có thể bị ảnh hưởng [10]. Để khắc phục, các kỹ thuật như ước lượng mật độ Kernel (*Kernel Density Estimation*) hoặc rời rạc hóa (*discretization*) có thể được áp dụng, nhưng điều này làm tăng độ phức tạp của quá trình tiền xử lý [62].

4.8.3 Tại sao Naïve Bayes phù hợp với lọc thư rác?

Thuật toán Naïve Bayes đặc biệt phù hợp cho các ứng dụng như lọc thư rác nhờ vào các đặc tính ưu việt của nó trong việc xử lý dữ liệu văn bản và khả năng học tiềm tiến [54]

- Học tiềm tiến: Thuật toán Naïve Bayes cho phép cập nhật nhanh các xác suất có điều kiện khi có dữ liệu mới mà không cần huấn luyện lại toàn bộ mô hình [56]. Trong lọc thư rác, nếu một thư điện tử bị phân loại sai (ví dụ, một thư điện tử hợp lệ bị đánh dấu là thư rác), người dùng có thể sửa nhãn, và mô hình sẽ tự động cập nhật các xác suất dựa trên phản hồi này, cải thiện hiệu suất theo thời gian [57].
- Phản hồi ngầm để cập nhật: Hành động của người dùng, chẳng hạn như gắn nhãn “không phải thư rác” cho một thư điện tử bị phân loại sai, cung cấp phản hồi ngầm (*implicit feedback*) để cải thiện mô hình mà không cần can thiệp thủ công vào quá trình huấn luyện [54]. Điều này giúp Thuật toán Naïve Bayes thích ứng nhanh chóng với các mẫu thư rác mới, chẳng hạn như các chiến dịch thư rác sử dụng từ khóa khác biệt.
- Cải thiện liên tục: Thuật toán Naïve Bayes trở nên chính xác hơn khi nhận được nhiều dữ liệu hơn thông qua tương tác của người dùng [56]. Trong các hệ thống thư điện tử, khi người dùng liên tục gắn nhãn các thư điện tử mới, mô hình học được các đặc trưng mới của thư rác và thư hợp lệ, từ đó nâng cao hiệu quả phân loại theo thời gian.

- Hiệu quả với đặc trưng văn bản: Lọc thư rác thường liên quan đến dữ liệu văn bản với các đặc trưng rời rạc (như tần suất từ khóa). Thuật toán Naïve Bayes xử lý tốt các đặc trưng này, đặc biệt khi kết hợp với làm mượt Laplace để xử lý các từ hiếm gặp, đảm bảo hiệu suất ổn định ngay cả khi dữ liệu kiểm thử chứa các từ không có trong tập huấn luyện [57].

Tổng kết lại, so với các thuật toán phức tạp hơn như *hoc sâu* hoặc *rừng ngẫu nhiên*, thuật toán Naïve Bayes có lợi thế về tốc độ và tính đơn giản. Nhờ đó, thuật toán này đặc biệt phù hợp cho các ứng dụng đòi hỏi khả năng học gia tăng và phản hồi nhanh, điển hình là bài toán lọc thư rác [54].

4.9 Bài tập

Sử dụng dữ liệu bệnh nhân tiểu đường ở Bảng 4.4 cho hai câu dưới đây:

Bảng 4.4: Tập dữ liệu bệnh nhân tiểu đường.

Giới tính	Chế độ ăn uống	Hoạt động thể chất	Mắc bệnh
Nam	Kém	Ít vận động	Có
Nữ	Tốt	Vận động nhiều	Không
Nam	Kém	Vận động nhiều	Không
Nữ	Tốt	Vận động nhiều	Không
Nam	Kém	Ít vận động	Có
Nữ	Kém	Ít vận động	Có
Nam	Tốt	Vận động nhiều	Không
Nữ	Tốt	Bình thường	Không
Nam	Tốt	Ít vận động	Có
Nữ	Tốt	Vận động nhiều	Không
Nam	Tốt	Vận động nhiều	Không
Nữ	Kém	Bình thường	Có

Bài tập 4.1. Tính các xác suất sau mà không sử dụng kỹ thuật làm tròn:

- $P(\text{Giới tính} = \text{Nữ})$
- $P(\text{Chế độ ăn uống} = \text{Tốt})$
- $P(\text{Hoạt động thể chất} = \text{Bình thường})$
- $P(\text{Mắc bệnh} = \text{Có})$
- $P(\text{Mắc bệnh} = \text{Không} | \text{Giới tính} = \text{Nữ}, \text{Chế độ ăn uống} = \text{Tốt}, \text{Hoạt động thể chất} = \text{Bình thường})$

Bài tập 4.2. Tính lại các xác suất trên nếu sử dụng làm mượt Laplace với $k = 1$.

Bài tập 4.3. Bài toán: Phần mềm lọc thư rác.

Giả sử bạn có một phần mềm lọc thư rác được cài đặt để tự động ngăn chặn các thư điện tử thư rác trước khi chúng đến hộp thư của bạn. Thống kê cho thấy, khoảng 40% thư điện tử

4.9. BÀI TẬP

trong tổng số là thư rác. Phần mềm này có khả năng phát hiện chính xác 95% thư rác, nghĩa là nếu một thư điện tử là thư rác, phần mềm sẽ nhận diện đúng 95% trường hợp. Tuy nhiên, phần mềm cũng có thể nhận nhầm một số thư điện tử hợp lệ là thư rác. Xác suất để phần mềm nhận nhầm một thư điện tử hợp lệ là thư rác (hay còn gọi là dương tính giả) là 10%.

Các câu hỏi đặt ra:

- a) Nếu phần mềm báo rằng một thư điện tử là thư rác, thì xác suất thực sự là thư rác là bao nhiêu?
- b) Nếu phần mềm báo rằng một thư điện tử là thư rác, thì xác suất thư điện tử đó là hợp lệ (không phải thư rác) là bao nhiêu?
- c) Nếu một thư điện tử là thư rác nhưng phần mềm không phát hiện, thì xác suất điều này xảy ra là bao nhiêu?
- d) Tổng xác suất mà phần mềm có thể phân loại đúng cả thư rác và thư hợp lệ là bao nhiêu?
- e) Nếu bạn muốn cải thiện hiệu suất của phần mềm, bạn sẽ tập trung vào giảm tỷ lệ dương tính giả (false positives) hay âm tính giả (false negatives)? Hãy giải thích lý do.

Bài tập 4.4. Hãy nêu một cách xây dựng bộ phân lớp Naïve Bayes cho bài toán phân loại thư điện tử rác hay không phải thư rác: các đầu vào gồm những thuộc tính nào (ví dụ: các từ trong thư điện tử, số lượng từ của thư điện tử, ...), lí do tại sao lại chọn những đầu vào đó; đầu ra sẽ có bao nhiêu nhãn, gồm những nhãn nào?

Bài tập 4.5. Vì sao bộ phân lớp Naïve Bayes nếu không áp dụng kỹ thuật Laplace Smoothing sẽ rất dễ bị overfit? Vì sao khi áp dụng Laplace Smoothing sẽ khắc phục được điều đó? Lấy một ví dụ với bài toán ở bài tập 4.2.

Chương 5

Thuật toán Support Vector Machine

5.1 Bài toán mở đầu

Trong bối cảnh kỹ nguyên số, sự bùng nổ của internet đã mang lại vô số cơ hội kết nối và giao dịch trực tuyến, nhưng đồng thời cũng làm gia tăng các mối đe dọa an ninh mạng, đặc biệt là các trang web lừa đảo. Các trang web này được thiết kế tinh vi để giả mạo các trang web hợp lệ, chẳng hạn như trang ngân hàng, thương mại điện tử hoặc dịch vụ trực tuyến, nhằm đánh cắp thông tin cá nhân như tên đăng nhập, mật khẩu hoặc thông tin tài chính của người dùng [63]. Mỗi ngày, người dùng internet phải đối mặt với hàng loạt liên kết từ email, mạng xã hội hoặc các nguồn không rõ ràng, trong đó chỉ một số ít là an toàn, trong khi phần lớn có thể dẫn đến các trang web độc hại. Việc xác định nhanh chóng và chính xác các trang web lừa đảo giữa dòng chảy thông tin trực tuyến liên tục là một nhu cầu cấp thiết trong môi trường số hiện đại.

Để giải quyết vấn đề này, các hệ thống tự động phân loại trang web (*automatic website classification*) đã được phát triển, cho phép phân biệt hiệu quả giữa các trang web hợp lệ và lừa đảo. Các hệ thống này tận dụng các thuật toán học máy để học từ dữ liệu lịch sử, chẳng hạn như thông tin về các trang web đã được xác minh là an toàn hoặc độc hại. Thông qua việc phân tích các đặc trưng của trang web, chẳng hạn như độ dài URL, sự hiện diện của giao thức HTTPS, tỷ lệ liên kết nội bộ/ngoại bộ, hoặc tuổi của tên miền, hệ thống có thể dự đoán nhãn của một trang web mới với độ chính xác cao [64].

Những hệ thống này có khả năng thực hiện phân loại trong thời gian thực, áp dụng các quy tắc đã học được để cảnh báo người dùng ngay khi họ truy cập một trang web tiềm ẩn nguy cơ. Ví dụ, khi một trang web có URL dài bất thường hoặc chứa nhiều ký tự đặc biệt, hệ thống có thể nhận diện và gắn nhãn nó là lừa đảo, từ đó chuyển hướng người dùng hoặc hiển thị cảnh báo trước khi thông tin nhạy cảm bị xâm phạm.

Trong số các thuật toán học máy được sử dụng cho bài toán này, thuật toán Support Vector Machine (viết tắt là SVM) nổi bật như một phương pháp mạnh mẽ và hiệu quả, đặc biệt trong các bài toán phân loại nhị phân như phân biệt trang web lừa đảo và hợp lệ [65]. Thuật toán này hoạt động bằng cách tìm một siêu phẳng (*hyperplane*) trong không gian đặc trưng để tách biệt hai lớp với khoảng cách biên (*margin*) lớn nhất, đảm bảo khả năng tổng quát hóa tốt ngay cả khi dữ liệu mới có sự chồng chéo hoặc nhiễu. Điểm mạnh của SVM nằm ở khả năng xử lý các tập dữ liệu có chiều cao, chẳng hạn như các đặc trưng đa dạng của trang web, và khả năng sử dụng kỹ thuật kernel để xử lý các trường hợp dữ liệu không tách biệt tuyến tính [6]. Hơn nữa,

5.2. KHÁI NIỆM

SVM hỗ trợ khả năng điều chỉnh tham số, chẳng hạn như tham số C để cân bằng giữa lỗi phân loại và tối ưu hóa biên, giúp mô hình thích nghi tốt với các đặc trưng phức tạp của dữ liệu trang web.

5.2 Khái niệm

Thuật toán Support Vector Machine (SVM) là một phương pháp học máy có giám sát được sử dụng chủ yếu cho các bài toán phân loại nhị phân (*binary classification*), dù cũng có thể mở rộng cho phân loại đa lớp hoặc hồi quy [6]. SVM hoạt động bằng cách xây dựng một siêu phẳng (*hyperplane*), là một không gian con $(n - 1)$ chiều trong không gian đặc trưng n chiều, nhằm tách biệt hai lớp dữ liệu một cách tối ưu. Trong không gian hai chiều, siêu phẳng này là một đường thẳng phân chia mặt phẳng thành hai vùng, mỗi vùng đại diện cho một lớp dữ liệu [66].

Tính tối ưu của siêu phẳng được xác định thông qua việc tối đa hóa khoảng cách biên (*margin*), tức là khoảng cách từ siêu phẳng đến các *vector hỗ trợ* (*support vectors*) - các điểm dữ liệu gần nhất của mỗi lớp. Khoảng cách này thường được tính dựa trên độ đo Euclidean trong không gian đặc trưng, và khoảng cách biên lớn hơn giúp mô hình tổng quát hóa tốt hơn, giảm nguy cơ quá khớp khi dự đoán trên dữ liệu mới [67]. Trong quá trình huấn luyện, SVM sử dụng tập dữ liệu có nhãn để xác định siêu phẳng tối ưu bằng cách giải bài toán tối ưu hóa, đảm bảo sự phân tách rõ ràng giữa các lớp.

Khi dữ liệu không thể tách biệt tuyến tính (*linearly separable*) trong không gian đặc trưng ban đầu, SVM áp dụng kỹ thuật kernel để ánh xạ dữ liệu vào một không gian chiều cao hơn, nơi các lớp có thể được tách biệt. Tham số C trong SVM được sử dụng để điều chỉnh mức độ khoan dung đối với lỗi phân loại, cân bằng giữa tối ưu hóa khoảng cách biên và độ chính xác trên tập huấn luyện, giúp xử lý dữ liệu có nhiễu [68].

Sau khi huấn luyện, SVM phân loại dữ liệu mới bằng cách xác định vị trí của chúng trong không gian đặc trưng, dựa trên khoảng cách từ điểm dữ liệu đến siêu phẳng. Với khả năng xử lý các không gian đặc trưng cao chiều và dữ liệu phức tạp, SVM được ứng dụng rộng rãi trong các bài toán như phân loại văn bản, nhận diện hình ảnh, và phát hiện gian lận trong an ninh mạng [69].

Ví dụ 5.1. Cho ba điểm dữ liệu: $A(1, 2)$ thuộc nhãn -1 , $B(3, 4)$ thuộc nhãn $+1$, và $C(5, 2)$ thuộc nhãn $+1$. Xét ba đường thẳng dùng để phân loại các điểm này như sau:

- Đường thẳng $1.2x + y - 6 = 0$ (tương đương với $y_1 = -1.2x + 6$):
 - $A(1, 2)$: phân loại đúng (-1)
 - $B(3, 4)$: phân loại đúng ($+1$)
 - $C(5, 2)$: phân loại đúng ($+1$)

Dộ chính xác: 100%.

- Đường thẳng $-4x + y + 16 = 0$ (tương đương với $y_2 = 4x - 16$):
 - $A(1, 2)$: phân loại sai ($+1$ thay vì -1)
 - $B(3, 4)$: phân loại đúng ($+1$)
 - $C(5, 2)$: phân loại sai (-1 thay vì $+1$)

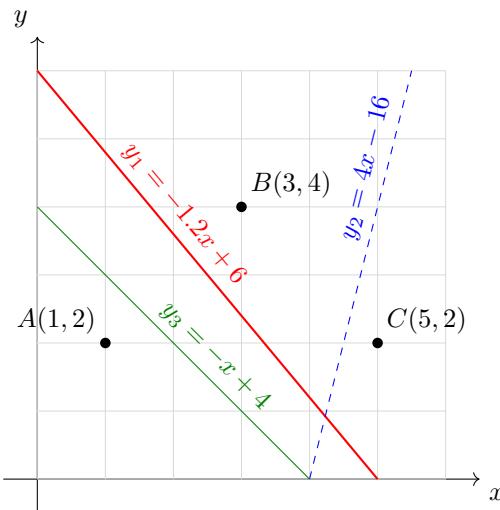
Dộ chính xác: 33.3%.

- Đường thẳng $x + y - 4 = 0$ (tương đương với $y_3 = -x + 4$):

- $A(1, 2)$: phân loại đúng (-1)
- $B(3, 4)$: phân loại đúng ($+1$)
- $C(5, 2)$: phân loại đúng ($+1$)

Dộ chính xác: 100%.

Hình 5.1 minh họa ba đường thẳng phân loại các điểm dữ liệu. Ta có thể thấy rằng, cả hai đường thẳng $y_1 = -1.2x + 6$ và $y_3 = -x + 4$ đều phân loại chính xác ba điểm A, B, C theo đúng nhau. Tuy nhiên, một câu hỏi quan trọng được đặt ra: trong số các đường phân cách thỏa mãn điều kiện phân loại đúng, đâu là đường ranh giới quyết định “tối ưu nhất”? Nói cách khác, làm thế nào để lựa chọn đường phân tách có khả năng “tổng quát hóa” tốt nhất trên dữ liệu chưa thấy?



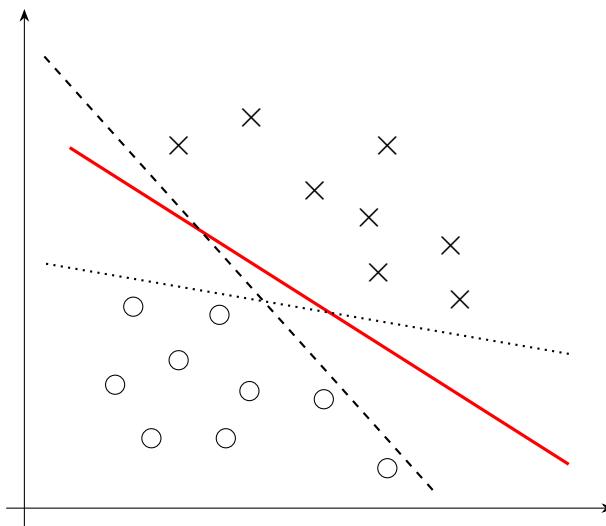
Hình 5.1: Minh họa ba đường thẳng phân loại các điểm dữ liệu.

5.3 Ý tưởng tìm siêu phẳng phân chia

Trong các bài toán phân loại nhị phân, thuật toán SVM tìm kiếm một siêu phẳng (*hyperplane*) trong không gian đặc trưng n chiều để tách biệt hai lớp dữ liệu một cách tối ưu [6]. Để minh họa ý tưởng, ta bắt đầu với trường hợp đơn giản trong không gian hai chiều, nơi siêu phẳng là một đường thẳng phân chia mặt phẳng thành hai vùng, mỗi vùng đại diện cho một lớp dữ liệu, chẵng hạn các điểm biểu diễn bằng hình tròn và hình vuông. Trong tập dữ liệu huấn luyện có nhau, có thể tồn tại nhiều siêu phẳng có khả năng phân tách đúng toàn bộ dữ liệu. Tuy nhiên, chất lượng của các siêu phẳng này không đồng đều, vì nó ảnh hưởng trực tiếp đến khả năng tổng quát hóa của mô hình, tức là khả năng dự đoán chính xác trên các điểm dữ liệu mới, chưa từng xuất hiện trong tập huấn luyện [67].

Hình 5.2 thể hiện ba siêu phẳng phân chia trong không gian hai chiều, tất cả đều phân tách đúng các điểm huấn luyện của hai lớp. Tuy nhiên, siêu phẳng nét liền được coi là tối ưu, vì nó có khoảng cách biên lớn nhất, tức là khoảng cách từ siêu phẳng đến các điểm dữ liệu gần nhất của mỗi lớp, được gọi là vector hỗ trợ (*support vectors*). Hai siêu phẳng khác (nét đứt và chấm)

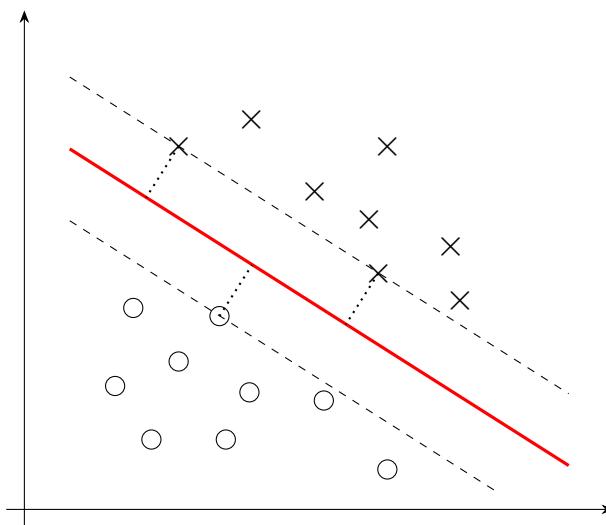
5.3. Ý TƯỞNG TÌM SIÊU PHẲNG PHÂN CHIA



Hình 5.2: Minh họa các siêu phẳng phân chia với khoảng cách biên khác nhau.

tuy phân tách đúng nhưng nằm quá gần các vector hỗ trợ, khiến mô hình dễ bị nhạy cảm với nhiễu hoặc các điểm dữ liệu mới có vị trí hơi lệch, dẫn đến nguy cơ phân loại sai [68].

Ý tưởng cốt lõi của SVM là tìm siêu phẳng tối ưu, được định nghĩa là siêu phẳng có khoảng cách biên lớn nhất, cách đều hai lớp và đảm bảo khoảng cách từ siêu phẳng đến các vector hỗ trợ của mỗi lớp là tối đa. Trong không gian hai chiều, điều này tương đương với việc xác định hai đường thẳng song song, mỗi đường đi qua các vector hỗ trợ gần nhất của một lớp, và siêu phẳng tối ưu là đường thẳng nằm chính giữa hai đường này. Khoảng cách từ siêu phẳng đến các vector hỗ trợ được tính toán dựa trên độ đo Euclidean trong không gian đặc trưng, và việc tối đa hóa khoảng cách biên giúp tăng cường khả năng tổng quát hóa của mô hình [70].

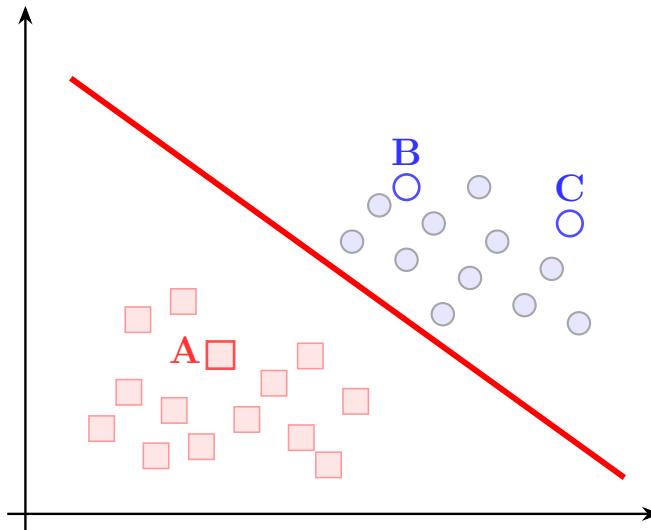


Hình 5.3: Minh họa việc xác định khoảng cách biên và vector hỗ trợ.

Quá trình xác định khoảng cách biên được minh họa trong Hình 5.3. Trong số các siêu phẳng phân tách, siêu phẳng tối ưu là đường có khoảng cách lớn nhất và cách đều các vector hỗ trợ của hai lớp. Khi một điểm dữ liệu mới xuất hiện, việc phân loại được thực hiện bằng cách xác định vị trí của nó so với siêu phẳng, dựa trên khoảng cách đến siêu phẳng trong không gian đặc trưng. Một siêu phẳng có khoảng cách biên lớn sẽ giảm xác suất phân loại sai, đặc biệt khi dữ liệu mới nằm gần ranh giới giữa hai lớp [66].

Trong thực tế, khi dữ liệu có nhiều chiều hoặc không tách biệt tuyến tính (*linearly separable*), việc tìm siêu phẳng bằng tay trở nên bất khả thi. SVM giải quyết vấn đề này bằng cách sử dụng các phương pháp tối ưu hóa lồi (*convex optimization*) để tự động xác định siêu phẳng có khoảng cách biên lớn nhất. Các vector hỗ trợ, là những điểm dữ liệu gần siêu phẳng nhất, đóng vai trò quyết định trong việc định hình siêu phẳng. Tên gọi “*Support Vector Machine*” xuất phát từ nguyên lý này, nhấn mạnh vai trò của các vector hỗ trợ trong quá trình huấn luyện [67].

Khi dữ liệu không tách biệt tuyến tính, SVM sử dụng kỹ thuật kernel để ánh xạ dữ liệu vào không gian chiều cao hơn, nơi một siêu phẳng tuyến tính có thể được xây dựng. Các hàm kernel như *radial basis function kernel* giúp tính toán khoảng cách trong không gian này mà không cần biểu diễn trực tiếp các vector đặc trưng, từ đó nâng cao hiệu quả tính toán [70]. Quá trình tìm siêu phẳng tối ưu được biểu diễn dưới dạng bài toán tối ưu hóa với các ràng buộc, thường sử dụng phương pháp nhân tử Lagrange để tìm nghiệm, như sẽ được trình bày chi tiết trong các phần tiếp theo.



Hình 5.4: Minh họa các cặp dữ liệu huấn luyện và ranh giới quyết định trong không gian hai chiều.

Giả sử chúng ta có một tập dữ liệu gồm ba điểm: điểm A (hình vuông), điểm B và điểm C (hình tròn), như trong Hình 5.4. Nhiệm vụ của bài toán phân loại là tìm ra một đường phân chia sao cho điểm A được phân loại vào một nhóm và điểm B, C được phân loại vào nhóm còn lại.

Mục tiêu của thuật toán SVM là tìm ra một đường thẳng phân chia tối ưu sao cho các điểm trong mỗi nhóm (ví dụ: nhóm hình vuông và nhóm hình tròn) được phân loại chính xác. Sau khi tìm được đường phân loại, SVM sẽ tự động gán nhãn cho các điểm trong tập huấn luyện: nhóm hình vuông có nhãn -1 và nhóm hình tròn có nhãn $+1$, hoặc ngược lại, tùy vào cách mà thuật toán tìm ra đường phân chia.

5.4. TRỰC QUAN VỀ KHOẢNG CÁCH BIÊN

Điều quan trọng ở đây là SVM không chỉ tìm đường phân chia sao cho phân loại đúng, mà còn tối ưu hóa khoảng cách biên giữa các nhóm. Margin được hiểu là khoảng cách giữa đường phân chia và điểm gần nhất của mỗi nhóm. SVM sẽ tìm ra đường phân chia sao cho margin giữa các nhóm là lớn nhất, điều này giúp thuật toán có khả năng tổng quát tốt hơn, tức là có thể phân loại chính xác những điểm chưa thấy trong dữ liệu huấn luyện.

Sau khi tìm ra đường phân chia tối ưu, SVM sẽ tự động gán nhãn cho từng điểm trong tập huấn luyện dựa trên vị trí của chúng đối với đường phân chia. Các điểm nằm về một phía của đường phân chia sẽ có nhãn +1 (hoặc -1 tùy thuộc vào cách thuật toán xác định), và các điểm còn lại sẽ có nhãn ngược lại. Điều này giúp phân loại chính xác các điểm trong không gian.

5.4 Trực quan về khoảng cách biên

Để hiểu rõ hơn về thuật toán SVM, việc nắm bắt ý nghĩa trực quan của khoảng cách biên (*margin*) và vai trò của nó trong việc đảm bảo độ tin cậy của dự đoán là một bước quan trọng. Phần này tập trung vào việc minh họa cách khoảng cách từ điểm dữ liệu đến ranh giới quyết định ảnh hưởng đến độ tin cậy của dự đoán, đồng thời so sánh với một phương pháp phân loại khác là hồi quy Logistic để làm nổi bật ưu điểm của SVM [67].

Trong hồi quy Logistic, xác suất một điểm dữ liệu x thuộc lớp dương ($y = 1$) được mô hình hóa bởi hàm sigmoid: $h_\theta(x) = g(\theta^T x) = \frac{1}{1 + e^{-\theta^T x}}$, với θ là vector tham số. Dự đoán $y = 1$ được đưa ra khi $h_\theta(x) \geq 0.5$, tương ứng với $\theta^T x \geq 0$. Độ tin cậy của dự đoán tỷ lệ thuận với giá trị $|\theta^T x|$: nếu $\theta^T x \gg 0$, xác suất $p(y = 1 | x; \theta)$ cao, dẫn đến độ tin cậy lớn cho nhãn $y = 1$; ngược lại, nếu $\theta^T x \ll 0$, nhãn $y = 0$ được dự đoán với độ tin cậy cao [71]. Tuy nhiên, hồi quy Logistic không trực tiếp tối ưu hóa khoảng cách từ điểm dữ liệu đến ranh giới quyết định, mà chủ yếu tập trung vào tối đa hóa xác suất nhãn đúng, điều này có thể dẫn đến ranh giới quyết định nhạy cảm với nhiễu trong dữ liệu.

Ngược lại, SVM ưu tiên tìm một siêu phẳng sao cho khoảng cách biên - khoảng cách từ siêu phẳng đến các điểm dữ liệu gần nhất của mỗi lớp - là lớn nhất. Khoảng cách biên lớn giúp tăng khả năng tổng quát hóa, giảm nguy cơ phân loại sai khi dữ liệu mới xuất hiện gần ranh giới quyết định [6]. Để minh họa, xét một tập dữ liệu huấn luyện trong không gian hai chiều, với các điểm thuộc lớp dương ($y = 1$) được biểu diễn bằng ký hiệu chữ X và lớp âm ($y = 0$) bằng ký hiệu chữ O, như trong Hình 5.4.

Hình 5.4 thể hiện ba điểm dữ liệu A, B và C thuộc lớp dương ($y = 1$). Điểm C nằm xa ranh giới quyết định, dẫn đến độ tin cậy cao khi dự đoán $y = 1$, vì khoảng cách lớn (tính bằng độ đo Euclidean) từ điểm C đến siêu phẳng làm giảm nguy cơ bị ảnh hưởng bởi nhiễu hoặc biến động nhỏ trong dữ liệu. Ngược lại, điểm A nằm rất gần ranh giới quyết định, khiến độ tin cậy của dự đoán $y = 1$ thấp hơn, do một thay đổi nhỏ có thể đẩy điểm A sang phía lớp âm ($y = 0$). Điểm B, nằm giữa A và C, có độ tin cậy trung gian. Như vậy, trong SVM, độ tin cậy của dự đoán tỷ lệ thuận với khoảng cách từ điểm dữ liệu đến siêu phẳng, và một khoảng cách biên lớn đảm bảo dự đoán chính xác hơn trên dữ liệu mới [68].

So với hồi quy Logistic, SVM có lợi thế trong việc tối ưu hóa khoảng cách biên, giúp ranh giới quyết định ít nhạy cảm hơn với nhiễu và phù hợp với các bài toán phân loại có dữ liệu phức tạp, chẳng hạn như phân loại văn bản hoặc phát hiện gian lận trong an ninh mạng [69]. Phần tiếp theo sẽ đi sâu vào cách SVM sử dụng các kỹ thuật tối ưu hóa để xác định siêu phẳng có khoảng cách biên lớn nhất, đảm bảo hiệu quả phân loại tối ưu.

5.5 Thuật toán

5.5.1 Kí hiệu

Trong bài toán phân loại nhị phân, tập dữ liệu huấn luyện gồm các cặp $(x^{(i)}, y^{(i)})$, với $x^{(i)} \in \mathbb{R}^n$ là vector đặc trưng và $y^{(i)} \in \{-1, 1\}$ là nhãn lớp. Việc sử dụng nhãn $\{-1, 1\}$ thay vì $\{0, 1\}$ giúp đơn giản hóa biểu diễn toán học của siêu phẳng $w^T x + b = 0$, là ranh giới quyết định phân tách hai lớp [67]. Bộ phân loại tuyến tính được tham số hóa bởi vector trọng số $w \in \mathbb{R}^n$ và tham số chặn $b \in \mathbb{R}$, với hàm phân loại:

$$h_{w,b}(x) = g(w^T x + b), \quad (5.1)$$

trong đó $g(z) = 1$ nếu $z \geq 0$ và $g(z) = -1$ nếu $z < 0$. Tham số b đại diện cho ngưỡng chặn (*bias*), xác định vị trí của siêu phẳng, trong khi w quyết định hướng của nó, với $\|w\|$ ảnh hưởng đến độ dốc của ranh giới quyết định. Hàm $h_{w,b}(x)$ trực tiếp dự đoán nhãn $\{-1, 1\}$ dựa trên dấu của $w^T x + b$, khác với hồi quy Logistic, vốn sử dụng hàm *sigmoid* để tính xác suất nhãn [71]. Ví dụ, với $x = [1, 2]^T$, $w = [1, 1]^T$, $b = -1$, nếu $w^T x + b = 2 > 0$, thì $h_{w,b}(x) = 1$, dự đoán nhãn dương.

5.5.2 Hàm khoảng cách

Hàm khoảng cách (*functional margin*) đo lường mức độ đáng tin cậy của dự đoán đối với một cặp dữ liệu $(x^{(i)}, y^{(i)})$. Đối với siêu phẳng $w^T x + b = 0$, hàm khoảng cách được định nghĩa:

$$\hat{\gamma}^{(i)} = y^{(i)}(w^T x^{(i)} + b). \quad (5.2)$$

Nếu $y^{(i)} = 1$, cần $w^T x^{(i)} + b > 0$ và giá trị này càng lớn thì dự đoán càng tự tin; tương tự, nếu $y^{(i)} = -1$, cần $w^T x^{(i)} + b < 0$ với giá trị âm lớn. Ví dụ, với $x^{(i)} = [1, 2]^T$, $w = [1, 1]^T$, $b = -1$, nếu $y^{(i)} = 1$, thì $\hat{\gamma}^{(i)} = 1 \cdot (1 \cdot 1 + 1 \cdot 2 - 1) = 2$, cho thấy dự đoán đúng và tự tin [68].

Tuy nhiên, hàm khoảng cách có hạn chế: nếu thay (w, b) bằng (kw, kb) với $k > 0$, thì $\hat{\gamma}^{(i)}$ tăng tỷ lệ k mà không thay đổi dấu của $w^T x + b$, dẫn đến giá trị hàm khoảng cách không nhất quán. Ví dụ, nếu $k = 2$, $\hat{\gamma}^{(i)}$ tăng lên 4, nhưng dự đoán không thay đổi. Để khắc phục, SVM thường áp dụng chuẩn hóa, chẳng hạn đặt $\|w\|_2 = 1$, tức là sử dụng $\left(\frac{w}{\|w\|_2}, \frac{b}{\|w\|_2}\right)$, để đảm bảo tính so sánh được giữa các siêu phẳng [70].

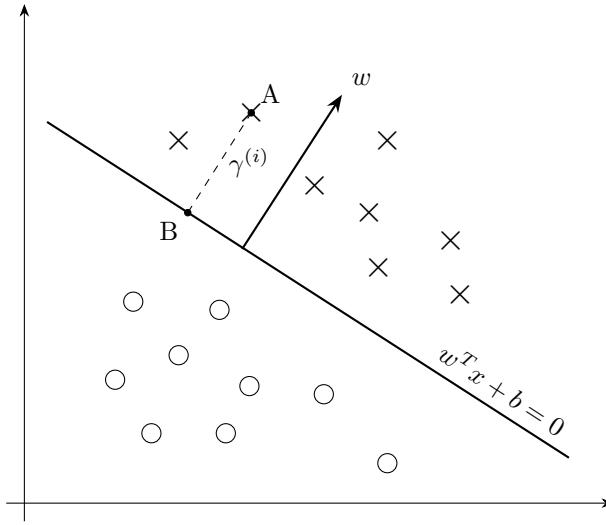
Đối với tập dữ liệu huấn luyện $S = \{(x^{(i)}, y^{(i)}); i = 1, 2, \dots, m\}$, hàm khoảng cách của siêu phẳng được định nghĩa là:

$$\hat{\gamma} = \min_{i=1, \dots, m} \hat{\gamma}^{(i)}. \quad (5.3)$$

Giá trị $\hat{\gamma}$ thể hiện mức độ tin cậy thấp nhất của siêu phẳng, và mục tiêu của SVM là tối đa hóa $\hat{\gamma}$ để đảm bảo phân tách tốt nhất [6].

5.5.3 Khoảng cách hình học

Khoảng cách hình học (*geometric margin*) đo lường khoảng cách Euclidean từ điểm dữ liệu đến siêu phẳng, cung cấp thước đo trực quan hơn về khoảng cách biên. Trong không gian hai chiều, như minh họa trong Hình 5.5, các điểm thuộc lớp dương ($y = 1$) được biểu diễn bằng chữ X, và lớp âm ($y = -1$) bằng chữ O. Siêu phẳng $w^T x + b = 0$ là ranh giới quyết định, với w là vector pháp tuyến vuông góc với siêu phẳng.



Hình 5.5: Minh họa khoảng cách hình học từ điểm dữ liệu đến siêu phẳng.

Xét điểm A với đầu vào $x^{(i)}$ và nhãn $y^{(i)} = 1$, khoảng cách hình học $\gamma^{(i)}$ là độ dài đoạn thẳng AB từ A đến siêu phẳng. Điểm B nằm trên siêu phẳng, có tọa độ $x^{(i)} - \gamma^{(i)} \cdot \frac{w}{\|w\|}$, với $\frac{w}{\|w\|}$ là vector đơn vị cùng hướng với w . Vì B thỏa mãn $w^T(x^{(i)} - \gamma^{(i)} \cdot \frac{w}{\|w\|}) + b = 0$, ta suy ra:

$$w^T x^{(i)} - \gamma^{(i)} \cdot \frac{w^T w}{\|w\|_2} + b = 0, \quad (5.4)$$

với $w^T w = \|w\|_2^2$, nên:

$$\gamma^{(i)} = \frac{w^T x^{(i)} + b}{\|w\|_2} = \left(\frac{w}{\|w\|_2} \right)^T x^{(i)} + \frac{b}{\|w\|_2}. \quad (5.5)$$

Tổng quát hơn, khoảng cách hình học được định nghĩa:

$$\gamma^{(i)} = y^{(i)} \left(\left(\frac{w}{\|w\|_2} \right)^T x^{(i)} + \frac{b}{\|w\|_2} \right). \quad (5.6)$$

Khoảng cách này đo lường khoảng cách từ điểm dữ liệu đến siêu phẳng, với giá trị dương khi dự đoán đúng nhãn và âm khi sai. Độ lớn của $\gamma^{(i)}$ tỷ lệ thuận với khoảng cách từ điểm đến siêu phẳng, và được chuẩn hóa theo độ dài của vector pháp tuyến w , đảm bảo tính nhất quán trong các phép biến đổi tỉ lệ của w và b .

Ví dụ, với $x^{(i)} = [1, 2]^T$, $w = [1, 1]^T$, $b = -1$, $y^{(i)} = 1$, ta có $\|w\| = \sqrt{1^2 + 1^2} = \sqrt{2}$, nên $\gamma^{(i)} = 1 \cdot \frac{1 \cdot 1 + 1 \cdot 2 - 1}{\sqrt{2}} = \frac{2}{\sqrt{2}} = \sqrt{2}$. Công thức này đảm bảo $\gamma^{(i)}$ dương khi dự đoán đúng nhãn, và không thay đổi khi thực hiện phép biến đổi tỉ lệ (scaling) w và b :

$$\frac{w^T x^{(i)} + b}{\|w\|_2} = \frac{k(w^T x^{(i)} + b)}{k\|w\|_2} = \frac{(kw)^T x^{(i)} + (kb)}{\|kw\|_2}. \quad (5.7)$$

Khi $\|w\|_2 = 1$, hàm khoảng cách và khoảng cách hình học bằng nhau, thể hiện mối liên hệ chặt chẽ giữa hai khái niệm [68]. Độ lớn $\|w\|_2$ ảnh hưởng đến khoảng cách, vì nó xác định độ dốc của siêu phẳng; một $\|w\|_2$ nhỏ hơn làm tăng $\gamma^{(i)}$, phù hợp với mục tiêu ưu hóa của SVM.

Đối với tập dữ liệu huấn luyện S , khoảng cách hÌnh học của siêú phảng là:

$$\gamma = \min_{i=1,\dots,m} \gamma^{(i)}. \quad (5.8)$$

Mục tiêu của SVM là tối đa hóa γ , thường thông qua ràng buộc chuẩn hóa $w^T x^{(i)} + b = \pm 1$ cho các vector hỗ trợ, là những điểm có $\gamma^{(i)} = \gamma$, để đơn giản hóa bài toán tối ưu hóa [70].

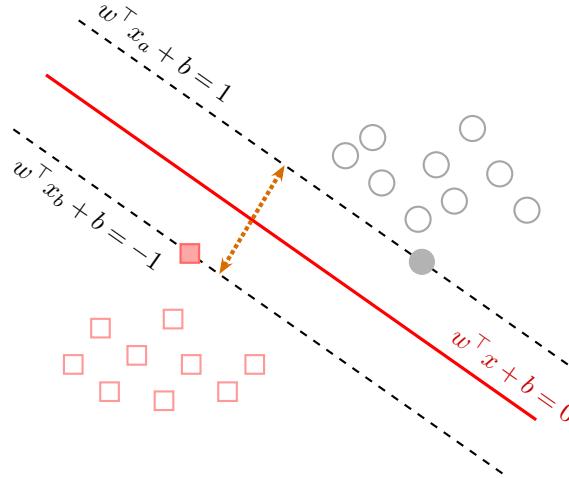
5.6 Bài toán tối ưu trong SVM

5.6.1 Mô hình SVM cơ bản

Cho tập dữ liệu huấn luyện $S = \{(x^{(i)}, y^{(i)}); i = 1, 2, \dots, m\}$, với $x^{(i)} \in \mathbb{R}^n$ là vector đặc trưng và $y^{(i)} \in \{-1, 1\}$ là nhãn lớp, mục tiêu là tìm siêú phảng $w^T x + b = 0$, trong đó $w \in \mathbb{R}^n$ là vector trọng số và $b \in \mathbb{R}$ là bias - hằng số điều chỉnh cho siêú phảng quyết định. Siêú phảng này phân tách không gian đặc trưng thành hai nửa: các điểm với $y^{(i)} = 1$ nằm ở phía $w^T x + b > 0$, và các điểm với $y^{(i)} = -1$ nằm ở phía $w^T x + b < 0$, như minh họa trong Hình 5.6 [67]. Hàm phân loại được định nghĩa:

$$h_{w,b}(x) = \text{sign}(w^T x + b), \quad (5.9)$$

trong đó $\text{sign}(z) = 1$ nếu $z \geq 0$ và -1 nếu $z < 0$.



Hình 5.6: Minh họa ranh giới quyết định của siêú phảng trong không gian hai chiều.

5.6.2 Mục tiêu tối ưu

Mục tiêu của SVM là tìm siêú phảng sao cho khoảng cách, tức khoảng cách từ siêú phảng đến các vector hỗ trợ, là lớn nhất. Các vector hỗ trợ là những điểm $x^{(i)}$ thỏa mãn:

$$y^{(i)}(w^T x^{(i)} + b) = 1, \quad (5.10)$$

nằm trên các siêu phẳng biên $w^T x + b = \pm 1$. Khoảng cách từ siêu phẳng đến một vector hỗ trợ là $\frac{1}{\|w\|}$, do đó khoảng cách biên tổng giữa hai lớp là:

$$\text{Margin} = \frac{2}{\|w\|}. \quad (5.11)$$

Tối đa hóa $\frac{2}{\|w\|}$ tương đương với tối thiểu hóa $\|w\|$. Để đảm bảo hàm mục tiêu lồi và dễ tính đạo hàm, ta sử dụng $\frac{1}{2}\|w\|^2$ thay vì $\|w\|$, vì $\frac{1}{2}\|w\|^2$ là một hàm bậc hai lồi [68]. Ví dụ, nếu $w = [1, 1]^T$, thì $\|w\| = \sqrt{2}$, và khoảng cách biên là $\frac{2}{\sqrt{2}} = \sqrt{2}$.

5.6.3 Bài toán tối ưu hóa trong SVM

Giả sử tập dữ liệu có thể phân tách tuyến tính, bài toán tối ưu hóa dạng gốc (*primal form*) của SVM được phát biểu:

$$\begin{aligned} \min_{w,b} \quad & \frac{1}{2}\|w\|^2 \\ \text{s.t.} \quad & y^{(i)}(w^T x^{(i)} + b) \geq 1, \quad i = 1, 2, \dots, m. \end{aligned} \quad (5.12)$$

Hàm mục tiêu $\frac{1}{2}\|w\|^2$ đảm bảo tính lồi, và hệ số $\frac{1}{2}$ giúp đơn giản hóa đạo hàm ($\frac{\partial}{\partial w} \frac{1}{2}\|w\|^2 = w$). Ràng buộc $y^{(i)}(w^T x^{(i)} + b) \geq 1$ đảm bảo mọi điểm dữ liệu được phân loại đúng và nằm ngoài hoặc trên lề (*margin*) [70].

Để suy diễn bài toán này, ta bắt đầu từ mục tiêu tối đa hóa khoảng cách hình học γ :

$$\begin{aligned} \max_{\gamma, w, b} \quad & \gamma \\ \text{s.t.} \quad & y^{(i)}(w^T x^{(i)} + b) \geq \gamma, \quad i = 1, 2, \dots, m, \\ & \|w\| = 1. \end{aligned} \quad (5.13)$$

Ràng buộc $\|w\| = 1$ đảm bảo γ là khoảng cách hình học, nhưng làm bài toán không lồi. Thay vào đó, ta sử dụng hàm khoảng cách $\hat{\gamma}^{(i)} = y^{(i)}(w^T x^{(i)} + b)$, và xét:

$$\begin{aligned} \max_{\hat{\gamma}, w, b} \quad & \frac{\hat{\gamma}}{\|w\|} \\ \text{s.t.} \quad & y^{(i)}(w^T x^{(i)} + b) \geq \hat{\gamma}, \quad i = 1, 2, \dots, m. \end{aligned} \quad (5.14)$$

Áp dụng ràng buộc chuẩn hóa $\hat{\gamma} = 1$, bài toán trở thành (4), với mục tiêu tối thiểu hóa $\frac{1}{2}\|w\|^2$ [67].

5.6.4 Ý nghĩa của ràng buộc

Ràng buộc $y^{(i)}(w^T x^{(i)} + b) \geq 1$ đảm bảo:

- Nếu $y^{(i)} = 1$, thì $w^T x^{(i)} + b \geq 1$, tức điểm nằm trên hoặc ngoài siêu phẳng $w^T x + b = 1$.
- Nếu $y^{(i)} = -1$, thì $w^T x^{(i)} + b \leq -1$, tức điểm nằm trên hoặc ngoài siêu phẳng $w^T x + b = -1$.
- Các điểm thỏa mãn đẳng thức $y^{(i)}(w^T x^{(i)} + b) = 1$ là vector hỗ trợ, nằm trên biên của lề và quyết định ranh giới quyết định [66].

Ví dụ, với $x^{(i)} = [1, 2]^T$, $y^{(i)} = 1$, $w = [2, 0]^T$, $b = -1$, ta có:

$$w^T x^{(i)} + b = 2 \cdot 1 + 0 \cdot 2 - 1 = 1,$$

thỏa mãn $y^{(i)}(w^T x^{(i)} + b) = 1 \cdot 1 = 1$, cho thấy điểm này là một vector hỗ trợ.

5.6.5 Chuyển sang dạng đối ngẫu

Bài toán gốc có thể được giải bằng quy hoạch bậc hai (*quadratic programming*), nhưng khi số chiều đặc trưng n lớn hoặc dữ liệu không phân tách tuyến tính, dạng đối ngẫu (*dual form*) hiệu quả hơn. Hàm Lagrange của bài toán là:

$$\mathcal{L}(w, b, \alpha) = \frac{1}{2} \|w\|^2 - \sum_{i=1}^m \alpha_i [y^{(i)}(w^T x^{(i)} + b) - 1], \quad (5.15)$$

với $\alpha_i \geq 0$ là nhân tử Lagrange (*Lagrange multipliers*). Lấy đạo hàm theo w và b :

$$\frac{\partial \mathcal{L}}{\partial w} = w - \sum_{i=1}^m \alpha_i y^{(i)} x^{(i)} = 0 \implies w = \sum_{i=1}^m \alpha_i y^{(i)} x^{(i)}, \quad (5.16)$$

$$\frac{\partial \mathcal{L}}{\partial b} = - \sum_{i=1}^m \alpha_i y^{(i)} = 0 \implies \sum_{i=1}^m \alpha_i y^{(i)} = 0. \quad (5.17)$$

Thay vào \mathcal{L} , bài toán đối ngẫu trở thành:

$$\max_{\alpha} \quad \sum_{i=1}^m \alpha_i - \frac{1}{2} \sum_{i,j=1}^m \alpha_i \alpha_j y^{(i)} y^{(j)} (x^{(i)})^T x^{(j)}, \quad (5.18)$$

dưới các ràng buộc:

$$\alpha_i \geq 0, \quad \sum_{i=1}^m \alpha_i y^{(i)} = 0. \quad (5.19)$$

Các vector hỗ trợ tương ứng với $\alpha_i > 0$. Dạng đối ngẫu chỉ phụ thuộc vào tích vô hướng $(x^{(i)})^T x^{(j)}$, cho phép áp dụng hàm kernel như đa thức kernel (*polynomial kernel*) $K(x, z) = (x^T z + c)^d$ hoặc Gaussian kernel $K(x, z) = \exp(-\gamma \|x - z\|^2)$ để xử lý dữ liệu phi tuyến [70]. Các điều kiện KKT đảm bảo tính tối ưu:

- $\alpha_i \geq 0$,
- $y^{(i)}(w^T x^{(i)} + b) - 1 \geq 0$,
- $\alpha_i [y^{(i)}(w^T x^{(i)} + b) - 1] = 0$.

Ví dụ 5.2. Xét bài toán với ba điểm: $A(1, 2)$ với nhãn $y_A = -1$, $B(3, 4)$ với nhãn $y_B = 1$, và $C(5, 2)$ với nhãn $y_C = 1$. Giả sử bài toán đối ngẫu cho các nhân tử Lagrange: $\alpha_A = 1$, $\alpha_B = 0$, $\alpha_C = 1$, cho thấy A và C là vector hỗ trợ.

Bước 1: Tính vector trọng số w

$$w = \sum_i \alpha_i y_i x_i = 1 \cdot (-1) \cdot \begin{bmatrix} 1 \\ 2 \end{bmatrix} + 0 \cdot 1 \cdot \begin{bmatrix} 3 \\ 4 \end{bmatrix} + 1 \cdot 1 \cdot \begin{bmatrix} 5 \\ 2 \end{bmatrix} = \begin{bmatrix} -1 \\ -2 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \end{bmatrix} + \begin{bmatrix} 5 \\ 2 \end{bmatrix} = \begin{bmatrix} 4 \\ 0 \end{bmatrix}.$$

Bước 2: Tính hằng số b

Chọn điểm C (một vector hỗ trợ), với $y_C = 1$:

$$w^T x_C = \begin{bmatrix} 4 \\ 0 \end{bmatrix} \cdot \begin{bmatrix} 5 \\ 2 \end{bmatrix} = 4 \cdot 5 + 0 \cdot 2 = 20,$$

$$y_C(w^T x_C + b) = 1 \implies 1 \cdot (20 + b) = 1 \implies b = -19.$$

Bước 3: Xác định ranh giới quyết định

$$w^T x + b = 0 \implies 4x_1 + 0x_2 - 19 = 0 \implies 4x_1 = 19.$$

Bước 4: Kiểm tra khoảng cách biên

$$\|w\| = \sqrt{4^2 + 0^2} = 4, \quad \text{Margin} = \frac{2}{\|w\|} = \frac{2}{4} = 0.5.$$

Ví dụ này minh họa cách vector hỗ trợ (A và C) xác định siêu phẳng tối ưu, với $\alpha_B = 0$ cho thấy B không ảnh hưởng đến ranh giới quyết định [66].

Dạng đối ngẫu không chỉ hiệu quả khi số vector hỗ trợ nhỏ, mà còn cho phép áp dụng hàm kernel để xử lý dữ liệu không phân tách tuyến tính. Ví dụ, trong phân loại văn bản, đa thức kernel có thể được sử dụng để ánh xạ các vector đặc trưng văn bản vào không gian nhiều chiều [69]. Các biến thể như SVM lè mềm (*Soft-margin SVM*), sử dụng biến trượt (*slack variables*) để xử lý nhiễu, và SVM kernel tăng cường khả năng phân tách phi tuyến, làm cho SVM trở thành công cụ mạnh mẽ trong các ứng dụng như nhận dạng chữ viết tay hoặc phát hiện gian lận [72].

5.7 Bài toán đối ngẫu Lagrange

Phương pháp nhân tử Lagrange (*Lagrange multipliers*) là một công cụ quan trọng để giải các bài toán tối ưu hóa có ràng buộc, đặc biệt trong thuật toán SVM. Phần này trình bày lý thuyết tổng quát về bài toán tối ưu hóa dạng gốc (*primal problem*) và đối ngẫu (*dual problem*), bao gồm cách xây dựng hàm Lagrange, suy ra bài toán đối ngẫu, và áp dụng các điều kiện Karush-Kuhn-Tucker (KKT). Sau đó, lý thuyết này được áp dụng vào bài toán tối ưu hóa của SVM, làm rõ cách dạng đối ngẫu giúp xác định vector hỗ trợ và hỗ trợ sử dụng kỹ thuật kernel (*kernel trick*) để xử lý dữ liệu không phân tách tuyến tính [73].

5.7.1 Phương pháp nhân tử Lagrange với ràng buộc đẳng thức

Xét bài toán tối ưu hóa với các ràng buộc đẳng thức:

$$\begin{aligned} & \min_w f(w) \\ & \text{s.t. } h_i(w) = 0, \quad i = 1, \dots, l. \end{aligned} \tag{5.20}$$

Để giải bài toán này, ta định nghĩa hàm Lagrange:

$$\mathcal{L}(w, \beta) = f(w) + \sum_{i=1}^l \beta_i h_i(w), \tag{5.21}$$

trong đó β_i là nhân tử Lagrange. Để tìm nghiệm, ta tính đạo hàm của \mathcal{L} theo w và β , rồi đặt bằng 0:

$$\frac{\partial \mathcal{L}}{\partial w_i} = 0, \quad i = 1, \dots, n; \quad \frac{\partial \mathcal{L}}{\partial \beta_i} = 0, \quad i = 1, \dots, l. \quad (5.22)$$

Giải hệ phương trình này, ta thu được w và β thỏa mãn bài toán gốc [73].

Ví dụ 5.3. Xét bài toán:

$$\min_{w_1, w_2} f(w) = w_1^2 + w_2^2, \quad \text{s.t.} \quad h_1(w) = w_1 + w_2 - 1 = 0.$$

Hàm Lagrange là:

$$\mathcal{L}(w, \beta_1) = w_1^2 + w_2^2 + \beta_1(w_1 + w_2 - 1).$$

Lấy đạo hàm:

$$\frac{\partial \mathcal{L}}{\partial w_1} = 2w_1 + \beta_1 = 0, \quad \frac{\partial \mathcal{L}}{\partial w_2} = 2w_2 + \beta_1 = 0, \quad \frac{\partial \mathcal{L}}{\partial \beta_1} = w_1 + w_2 - 1 = 0.$$

Giải hệ, ta được $w_1 = w_2 = \frac{1}{2}$, $\beta_1 = -1$, và giá trị tối ưu của $f(w) = \frac{1}{2}$. Ví dụ này minh họa cách nhân tử Lagrange giúp tìm nghiệm tối ưu dưới ràng buộc đẳng thức [74].

5.7.2 Bài toán đối ngẫu với ràng buộc bắt đẳng thức

Xét bài toán tối ưu hóa tổng quát hơn, bao gồm cả ràng buộc bắt đẳng thức (*primal problem*):

$$\begin{aligned} & \min_w f(w) \\ & \text{s.t. } g_i(w) \leq 0, \quad i = 1, \dots, k, \\ & \quad h_i(w) = 0, \quad i = 1, \dots, l. \end{aligned} \quad (5.23)$$

Hàm Lagrange tổng quát được định nghĩa:

$$\mathcal{L}(w, \alpha, \beta) = f(w) + \sum_{i=1}^k \alpha_i g_i(w) + \sum_{i=1}^l \beta_i h_i(w), \quad (5.24)$$

với $\alpha_i \geq 0$ và β_i là nhân tử Lagrange (*Lagrange multipliers*). Ta xét hàm:

$$\theta_{\mathcal{P}}(w) = \max_{\alpha, \beta: \alpha_i \geq 0} \mathcal{L}(w, \alpha, \beta). \quad (5.25)$$

Nếu w vi phạm ràng buộc ($g_i(w) > 0$ hoặc $h_i(w) \neq 0$), ta có thể chọn $\alpha_i \rightarrow \infty$ hoặc β_i phù hợp để:

$$\theta_{\mathcal{P}}(w) = f(w) + \sum_{i=1}^k \alpha_i g_i(w) + \sum_{i=1}^l \beta_i h_i(w) = \infty.$$

Ngược lại, nếu w thỏa mãn tất cả ràng buộc, thì $\alpha_i g_i(w) \leq 0$ và $h_i(w) = 0$, dẫn đến $\theta_{\mathcal{P}}(w) = f(w)$. Do đó:

$$\theta_{\mathcal{P}}(w) = \begin{cases} f(w) & \text{nếu } w \text{ thỏa mãn các ràng buộc,} \\ \infty & \text{ngược lại.} \end{cases} \quad (5.26)$$

Bài toán gốc tương đương với:

$$\min_w \theta_{\mathcal{P}}(w) = \min_w \max_{\alpha, \beta: \alpha_i \geq 0} \mathcal{L}(w, \alpha, \beta), \quad (5.27)$$

với giá trị tối ưu $p^* = \min_w \theta_{\mathcal{P}}(w)$, gọi là trị số của bài toán gốc (*primal value*).

Ta định nghĩa hàm đối ngẫu:

$$\theta_{\mathcal{D}}(\alpha, \beta) = \min_w \mathcal{L}(w, \alpha, \beta). \quad (5.28)$$

Bài toán đối ngẫu (*dual problem*):

$$\max_{\alpha, \beta: \alpha_i \geq 0} \theta_{\mathcal{D}}(\alpha, \beta) = \max_{\alpha, \beta: \alpha_i \geq 0} \min_w \mathcal{L}(w, \alpha, \beta). \quad (5.29)$$

Giá trị tối ưu của bài toán đối ngẫu là $d^* = \max_{\alpha, \beta: \alpha_i \geq 0} \theta_{\mathcal{D}}(\alpha, \beta)$. Theo nguyên lý đối ngẫu yếu (*weak duality*), ta luôn có:

$$d^* \leq p^*. \quad (5.30)$$

Nếu các điều kiện lồi được thỏa mãn (tức f và g_i là hàm lồi (*convex functions*), h_i là đa tạp tuyến tính (*affine*), ví dụ: $h_i(w) = a_i^T w + b_i$), và điều kiện Slater (*Slater's condition*) được đáp ứng (tồn tại w sao cho $g_i(w) < 0, h_i(w) = 0$), thì đối ngẫu mạnh (*strong duality*) đảm bảo $d^* = p^*$. Lúc này, nghiệm w^*, α^*, β^* thỏa mãn các điều kiện Karush-Kuhn-Tucker (KKT):

$$\frac{\partial \mathcal{L}}{\partial w_i}(w^*, \alpha^*, \beta^*) = 0, \quad i = 1, \dots, n, \quad (5.31)$$

$$\frac{\partial \mathcal{L}}{\partial \beta_i}(w^*, \alpha^*, \beta^*) = 0, \quad i = 1, \dots, l, \quad (5.32)$$

$$\alpha_i^* g_i(w^*) = 0, \quad i = 1, \dots, k, \quad (5.33)$$

$$g_i(w^*) \leq 0, \quad i = 1, \dots, k, \quad (5.34)$$

$$\alpha_i^* \geq 0, \quad i = 1, \dots, k. \quad (5.35)$$

Điều kiện KKT đối ngẫu bổ sung (*dual complementarity*) (5.33) cho thấy nếu $\alpha_i^* > 0$, thì $g_i(w^*) = 0$, nghĩa là ràng buộc bất đẳng thức trở thành đẳng thức. Điều này rất quan trọng trong SVM, như sẽ trình bày dưới đây [73].

5.7.3 Áp dụng vào SVM

Trong SVM, bài toán tối ưu hóa dạng gốc (*primal form*) cho tập dữ liệu huấn luyện $S = \{(x^{(i)}, y^{(i)}); i = 1, \dots, m\}$, với $x^{(i)} \in \mathbb{R}^n, y^{(i)} \in \{-1, 1\}$, là:

$$\begin{aligned} \min_{w, b} \quad & f(w, b) = \frac{1}{2} \|w\|^2 \\ \text{s.t.} \quad & g_i(w, b) = 1 - y^{(i)}(w^T x^{(i)} + b) \leq 0, \quad i = 1, \dots, m. \end{aligned} \quad (5.36)$$

Hàm Lagrange là:

$$\mathcal{L}(w, b, \alpha) = \frac{1}{2} \|w\|^2 + \sum_{i=1}^m \alpha_i \left[1 - y^{(i)}(w^T x^{(i)} + b) \right], \quad (5.37)$$

với $\alpha_i \geq 0$ là nhân tử Lagrange. Lấy đạo hàm:

$$\frac{\partial \mathcal{L}}{\partial w} = w - \sum_{i=1}^m \alpha_i y^{(i)} x^{(i)} = 0 \implies w = \sum_{i=1}^m \alpha_i y^{(i)} x^{(i)}, \quad (5.38)$$

$$\frac{\partial \mathcal{L}}{\partial b} = - \sum_{i=1}^m \alpha_i y^{(i)} = 0 \implies \sum_{i=1}^m \alpha_i y^{(i)} = 0. \quad (5.39)$$

Thay vào \mathcal{L} , ta được:

$$\mathcal{L} = \frac{1}{2} \left(\sum_{i=1}^m \alpha_i y^{(i)} x^{(i)} \right)^T \left(\sum_{j=1}^m \alpha_j y^{(j)} x^{(j)} \right) + \sum_{i=1}^m \alpha_i - \sum_{i=1}^m \alpha_i y^{(i)} \left(\left(\sum_{j=1}^m \alpha_j y^{(j)} x^{(j)} \right)^T x^{(i)} + b \right).$$

Rút gọn:

$$\mathcal{L} = \sum_{i=1}^m \alpha_i - \frac{1}{2} \sum_{i,j=1}^m \alpha_i \alpha_j y^{(i)} y^{(j)} (x^{(i)})^T x^{(j)}.$$

Bài toán đối ngẫu của SVM là:

$$\begin{aligned} \max_{\alpha} \quad & \sum_{i=1}^m \alpha_i - \frac{1}{2} \sum_{i,j=1}^m \alpha_i \alpha_j y^{(i)} y^{(j)} (x^{(i)})^T x^{(j)} \\ \text{s.t.} \quad & \alpha_i \geq 0, \quad i = 1, \dots, m, \\ & \sum_{i=1}^m \alpha_i y^{(i)} = 0. \end{aligned} \quad (5.40)$$

Điều kiện KKT đối ngẫu bổ sung (*dual complementarity*):

$$\alpha_i^* \left[y^{(i)} ((w^*)^T x^{(i)} + b^*) - 1 \right] = 0,$$

cho thấy $\alpha_i^* > 0$ chỉ khi $y^{(i)} ((w^*)^T x^{(i)} + b^*) = 1$, tức là $x^{(i)}$ là vector hỗ trợ. Các vector hỗ trợ nằm trên các siêu phẳng biên $w^T x + b = \pm 1$, quyết định ranh giới quyết định [70].

Hàm mục tiêu đối ngẫu chỉ phụ thuộc vào tích vô hướng $(x^{(i)})^T x^{(j)}$, cho phép sử dụng hàm kernel như đa thức kernel $K(x, z) = (x^T z + c)^d$ hoặc Gaussian kernel $K(x, z) = \exp(-\gamma \|x - z\|^2)$ để ánh xạ dữ liệu vào không gian đặc trưng nhiều chiều, xử lý các bài toán không phân tách tuyến tính [66].

Ví dụ 5.4. Xét tập dữ liệu với ba điểm: $A(1, 2)$ với $y_A = -1$, $B(3, 4)$ với $y_B = 1$, $C(5, 2)$ với $y_C = 1$. Giả sử bài toán đối ngẫu cho $\alpha_A = 1$, $\alpha_B = 0$, $\alpha_C = 1$. Tính vector trọng số w :

$$w = \sum_{i=1}^m \alpha_i y_i x_i = 1 \cdot (-1) \cdot \begin{bmatrix} 1 \\ 2 \end{bmatrix} + 0 \cdot 1 \cdot \begin{bmatrix} 3 \\ 4 \end{bmatrix} + 1 \cdot 1 \cdot \begin{bmatrix} 5 \\ 2 \end{bmatrix} = \begin{bmatrix} -1 \\ -2 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \end{bmatrix} + \begin{bmatrix} 5 \\ 2 \end{bmatrix} = \begin{bmatrix} 4 \\ 0 \end{bmatrix}.$$

Tính hằng số (*bias*) b , chọn C làm vector hỗ trợ:

$$w^T x_C + b = \begin{bmatrix} 4 \\ 0 \end{bmatrix} \cdot \begin{bmatrix} 5 \\ 2 \end{bmatrix} + b = 20 + b = 1 \implies b = -19.$$

5.8. HÀM KERNEL TRONG SVM

Siêu phẳng: $4x_1 = 19$. Kiểm tra khoảng cách biên:

$$\|w\| = \sqrt{4^2 + 0^2} = 4, \quad \text{Margin} = \frac{2}{\|w\|} = 0.5.$$

Ví dụ này minh họa cách các vector hỗ trợ (A, C) với $\alpha_i > 0$ xác định siêu phẳng tối ưu, trong khi B ($\alpha_B = 0$) không ảnh hưởng [6].

Để minh họa hàm kernel, xét bài toán không phân tách tuyến tính với đa thức kernel $K(x, z) = (x^T z + 1)^2$. Với $x^{(i)} = [x_1, x_2]^T$, $x^{(j)} = [z_1, z_2]^T$, ta có:

$$K(x^{(i)}, x^{(j)}) = (x_1 z_1 + x_2 z_2 + 1)^2,$$

ánh xạ dữ liệu vào không gian đặc trưng nhiều chiều, giúp phân tách các lớp phức tạp hơn [70].

5.7.4 Lợi ích và ứng dụng trong SVM

Dạng đối ngẫu mang lại nhiều lợi thế trong SVM:

- Tích vô hướng: Hàm mục tiêu chỉ phụ thuộc vào $(x^{(i)})^T x^{(j)}$, cho phép sử dụng hàm kernel để xử lý dữ liệu không phân tách tuyến tính mà không cần tính toán trực tiếp trong không gian nhiều chiều.
- Vector hỗ trợ: Số lượng vector hỗ trợ thường nhỏ, giảm độ phức tạp tính toán, đặc biệt với tập dữ liệu lớn.
- Điều kiện KKT: Điều kiện đối ngẫu bổ sung (5.33) đảm bảo chỉ các vector hỗ trợ có $\alpha_i > 0$, giúp xác định siêu phẳng tối ưu một cách hiệu quả.

5.8 Hàm kernel trong SVM

Trong các mục trước, bài toán đối ngẫu của SVM (xem mục **Bài toán đối ngẫu Lagrange** trong Phần 5.7) đã được trình bày, nhấn mạnh vai trò của vector hỗ trợ. Tuy nhiên, khi tập dữ liệu không phân tách tuyến tính như minh họa trong Hình 5.7, SVM cần ánh xạ dữ liệu vào không gian đặc trưng nhiều chiều, nơi dữ liệu có thể trở thành phân tách tuyến tính (Hình 5.8). Phần này giới thiệu hàm kernel, một công cụ mạnh mẽ giúp SVM xử lý dữ liệu phi tuyến mà không cần tính toán trực tiếp tọa độ trong không gian đặc trưng.

5.8.1 Khái niệm hàm kernel và ánh xạ đặc trưng

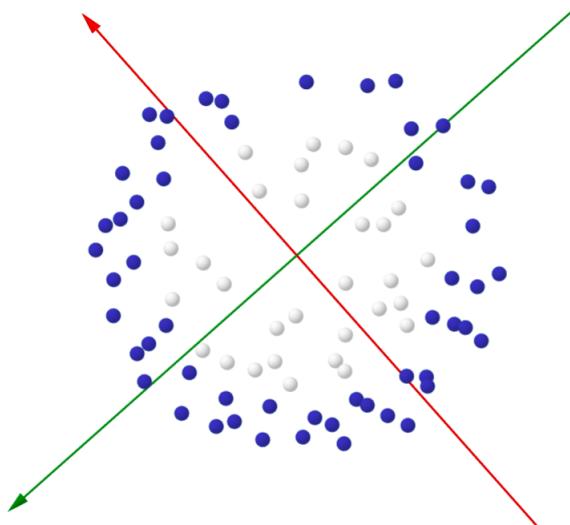
Ánh xạ đặc trưng (*feature mapping*) $\phi : \mathbb{R}^n \rightarrow \mathbb{R}^d$ chuyển các điểm dữ liệu x từ không gian gốc sang không gian đặc trưng nhiều chiều, nơi tồn tại siêu phẳng phân tách hai lớp. Thay vì áp dụng SVM trực tiếp trên x , ta sử dụng $\phi(x)$. Trong bài toán đối ngẫu của SVM, hàm mục tiêu chỉ chứa tích vô hướng $(x^{(i)})^T x^{(j)}$, được thay bằng:

$$K(x, z) = \phi(x)^T \phi(z), \tag{5.41}$$

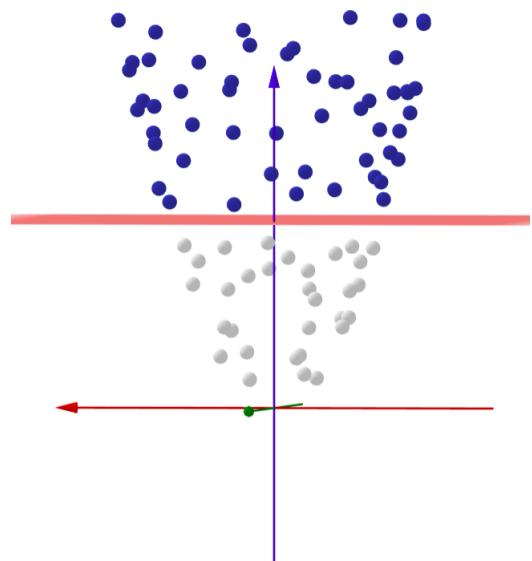
gọi là hàm kernel. Bằng cách thay $(x^{(i)})^T x^{(j)}$ bằng $K(x^{(i)}, x^{(j)})$ trong hàm mục tiêu đối ngẫu:

$$\max_{\alpha} \sum_{i=1}^m \alpha_i - \frac{1}{2} \sum_{i,j=1}^m \alpha_i \alpha_j y^{(i)} y^{(j)} K(x^{(i)}, x^{(j)}),$$

SVM có thể học trong không gian đặc trưng mà không cần tính $\phi(x)$ trực tiếp, giảm đáng kể độ phức tạp tính toán [72].



Hình 5.7: Tập dữ liệu không thể phân tách tuyến tính



Hình 5.8: Tập dữ liệu sau khi ánh xạ, trở thành phân tách tuyến tính

5.8.2 Đa thức kernel

Xét đa thức kernel $K(x, z) = (x^T z)^2$ với $x, z \in \mathbb{R}^n$:

$$K(x, z) = \left(\sum_{i=1}^n x_i z_i \right)^2 = \sum_{i,j=1}^n (x_i x_j)(z_i z_j). \quad (5.42)$$

Với $n = 3$, ánh xạ đặc trưng tương ứng là:

$$\phi(x) = \begin{bmatrix} x_1 x_1 \\ x_1 x_2 \\ x_1 x_3 \\ x_2 x_1 \\ x_2 x_2 \\ x_2 x_3 \\ x_3 x_1 \\ x_3 x_2 \\ x_3 x_3 \end{bmatrix}. \quad (5.43)$$

Tính $\phi(x)$ có độ phức tạp $O(n^2)$, nhưng tính $K(x, z)$ chỉ cần $O(n)$, minh họa hiệu quả của hàm kernel.

Tổng quát hơn, đa thức kernel $K(x, z) = (x^T z + c)^d$ ánh xạ vào không gian đặc trưng có số chiều $\binom{n+d}{d}$. Với $n = 3$, $d = 2$, $c = 1$, ta có:

$$K(x, z) = (x^T z + 1)^2 = \sum_{i,j=1}^3 (x_i x_j)(z_i z_j) + \sum_{i=1}^3 (\sqrt{2}x_i)(\sqrt{2}z_i) + 1, \quad (5.44)$$

với ánh xạ đặc trưng:

$$\phi(x) = \begin{bmatrix} x_1 x_1 \\ x_1 x_2 \\ x_1 x_3 \\ x_2 x_1 \\ x_2 x_2 \\ x_2 x_3 \\ x_3 x_1 \\ x_3 x_2 \\ x_3 x_3 \\ \sqrt{2}x_1 \\ \sqrt{2}x_2 \\ \sqrt{2}x_3 \\ 1 \end{bmatrix}. \quad (5.45)$$

Tham số c điều chỉnh số giữa các thành phần tuyến tính (x_i) và phi tuyến ($x_i x_j$), giúp kiểm soát độ phức tạp của siêu phẳng [66].

5.8.3 Gaussian kernel

Một hàm kernel phổ biến khác là Gaussian kernel, còn gọi là *kernel RBF (Radial Basis Function kernel)*:

$$K(x, z) = \exp \left(-\frac{\|x - z\|^2}{2\sigma^2} \right). \quad (5.46)$$

Hàm kernel này ánh xạ dữ liệu vào không gian vô hạn chiều, với σ điều chỉnh độ nhạy của kernel. Khi $x \approx z$, $K(x, z) \approx 1$; khi x và z xa nhau, $K(x, z) \approx 0$, phản ánh độ tương đồng giữa x và z . Tham số σ nhỏ làm kernel nhạy hơn, phù hợp với dữ liệu có biến động lớn; σ lớn làm kernel mượt hơn, phù hợp với dữ liệu đơn giản [70]. Ví dụ, với $\sigma = 0.5$, dữ liệu phức tạp hơn có thể được phân tách tốt hơn so với $\sigma = 2$.

Ví dụ 5.5. Xét tập dữ liệu: $x^{(1)} = (1, 1)$, $y^{(1)} = -1$; $x^{(2)} = (2, 2)$, $y^{(2)} = 1$; $x^{(3)} = (0, 0)$, $y^{(3)} = -1$. Trong không gian gốc, các điểm này không phân tách tuyến tính. Sử dụng đa thức kernel $K(x, z) = (x^T z + 1)^2$:

$$K(x^{(1)}, x^{(2)}) = \left(\begin{bmatrix} 1 \\ 1 \end{bmatrix}^T \begin{bmatrix} 2 \\ 2 \end{bmatrix} + 1 \right)^2 = (2 + 2 + 1)^2 = 25.$$

Ánh xạ đặc trưng ($n = 2$, $d = 2$, $c = 1$):

$$\phi(x^{(1)}) = \begin{bmatrix} 1 \\ 1 \\ 1 \\ \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \\ 1 \end{bmatrix}, \quad \phi(x^{(2)}) = \begin{bmatrix} 4 \\ 4 \\ 4 \\ 4 \\ 2\sqrt{2} \\ 2\sqrt{2} \\ 1 \end{bmatrix}.$$

Kiểm tra: $\phi(x^{(1)})^T \phi(x^{(2)}) = 1 \cdot 4 + 1 \cdot 4 + 1 \cdot 4 + 1 \cdot 4 + \sqrt{2} \cdot 2\sqrt{2} + \sqrt{2} \cdot 2\sqrt{2} + 1 \cdot 1 = 25$, khớp với $K(x^{(1)}, x^{(2)})$.

Sử dụng Gaussian kernel với $\sigma = 1$:

$$K(x^{(1)}, x^{(3)}) = \exp \left(-\frac{\|(1, 1) - (0, 0)\|^2}{2 \cdot 1^2} \right) = \exp \left(-\frac{\sqrt{2}^2}{2} \right) = \exp(-1) \approx 0.3679.$$

Các giá trị $K(x^{(i)}, x^{(j)})$ được sử dụng trong bài toán đối ngẫu để tìm vector hỗ trợ, minh họa cách hàm kernel giúp SVM xử lý dữ liệu phi tuyến [6].

5.8.4 Tính hợp lệ của hàm kernel

Một hàm kernel được gọi là hợp lệ nếu nó tương ứng với ánh xạ đặc trưng ϕ sao cho $K(x, z) = \phi(x)^T \phi(z)$. Điều này được đảm bảo bởi định lý Mercer.

Theo định lý Mercer, hàm $K : \mathbb{R}^n \times \mathbb{R}^n \rightarrow \mathbb{R}$ là hàm kernel hợp lệ nếu và chỉ nếu với mọi tập hữu hạn $\{x^{(1)}, \dots, x^{(m)}\}$, ma trận kernel \mathbf{K} với $K_{ij} = K(x^{(i)}, x^{(j)})$ là đối xứng và nửa xác định dương (*positive semi-definite*), tức là:

$$z^T \mathbf{K} z = \sum_{i,j=1}^m z_i K(x^{(i)}, x^{(j)}) z_j \geq 0, \quad \forall z \in \mathbb{R}^m. \quad (5.47)$$

Chứng minh: Với $K(x, z) = \phi(x)^T \phi(z)$, ma trận kernel \mathbf{K} có:

$$K_{ij} = \phi(x^{(i)})^T \phi(x^{(j)}) = K_{ji},$$

nên \mathbf{K} đối xứng. Hơn nữa:

$$z^T \mathbf{K} z = \sum_{i,j=1}^m z_i \phi(x^{(i)})^T \phi(x^{(j)}) z_j = \sum_k \left(\sum_{i=1}^m z_i \phi_k(x^{(i)}) \right)^2 \geq 0, \quad (5.48)$$

nên \mathbf{K} là nửa xác định dương. Định lý Mercer xác nhận rằng đa thức kernel và Gaussian kernel là hàm kernel hợp lệ [70].

Ví dụ, với tập $\{x^{(1)}, x^{(2)}\}$ và $K(x, z) = (x^T z)^2$, ma trận kernel:

$$\mathbf{K} = \begin{bmatrix} K(x^{(1)}, x^{(1)}) & K(x^{(1)}, x^{(2)}) \\ K(x^{(2)}, x^{(1)}) & K(x^{(2)}, x^{(2)}) \end{bmatrix},$$

là đối xứng và nửa xác định dương, thỏa mãn định lý Mercer.

Ví dụ 5.6. Xét ba điểm trong \mathbb{R}^2 :

$$A = (1, 2), y_A = -1; \quad B = (3, 4), y_B = +1; \quad C = (5, 2), y_C = +1.$$

(a) Với ánh xạ bậc 2 (tương ứng với $K_2(x, y) = (1 + x^T y)^2$)

$$\phi_2(x) = [1, \sqrt{2}x_1, \sqrt{2}x_2, x_1^2, \sqrt{2}x_1x_2, x_2^2]^\top,$$

hãy tính $A' = \phi_2(A), B' = \phi_2(B), C' = \phi_2(C)$ và các tích vô hướng $A'B', A'C', B'C'$ (dùng kernel $K_2(x_i, x_j) = (1 + x_i^T x_j)^2$ để kiểm tra).

(b) Với kernel bậc 3

$$K_3(x_i, x_j) = (1 + x_i^T x_j)^3,$$

hãy tính $K_3(A, B), K_3(A, C), K_3(B, C)$ và các giá trị cần thiết.

(c) Cho các hệ số $\alpha_A = 1, \alpha_B = 0, \alpha_C = 1$. Viết hàm quyết định $f(x)$ rồi phân loại điểm $D = (5, 5)$ trong trường hợp bỏ b (lấy $b = 0$).

Lời giải.

Bước chuẩn bị tích vô hướng trong \mathbb{R}^2 :

$$A \cdot B = 11, \quad A \cdot C = 9, \quad B \cdot C = 23,$$

$$A \cdot A = 5, \quad B \cdot B = 25, \quad C \cdot C = 29.$$

(a) Ánh xạ bậc 2 (dạng chuẩn):

$$A' = \phi_2(1, 2) = [1, \sqrt{2}, 2\sqrt{2}, 1, 2\sqrt{2}, 4],$$

$$B' = \phi_2(3, 4) = [1, 3\sqrt{2}, 4\sqrt{2}, 9, 12\sqrt{2}, 16],$$

$$C' = \phi_2(5, 2) = [1, 5\sqrt{2}, 2\sqrt{2}, 25, 10\sqrt{2}, 4].$$

Dùng kernel bậc 2 để kiểm tra:

$$A'B' = (1 + 11)^2 = 12^2 = 144, \quad A'C' = (1 + 9)^2 = 10^2 = 100, \quad B'C' = (1 + 23)^2 = 24^2 = 576.$$

(b) Kernel bậc 3:

$$K_3(A, B) = (1 + 11)^3 = 12^3 = 1728, \quad K_3(A, C) = (1 + 9)^3 = 10^3 = 1000,$$

$$K_3(B, C) = (1 + 23)^3 = 24^3 = 13824.$$

Một vài giá trị khác:

$$K_3(A, A) = (1+5)^3 = 6^3 = 216, K_3(B, B) = (1+25)^3 = 26^3 = 17576, K_3(C, C) = (1+29)^3 = 30^3 = 27000.$$

Với $D = (5, 5)$:

$$A \cdot D = 15 \Rightarrow K_3(A, D) = (1 + 15)^3 = 16^3 = 4096,$$

$$C \cdot D = 35 \Rightarrow K_3(C, D) = (1 + 35)^3 = 36^3 = 46656.$$

(c) Hàm quyết định theo α đã cho:

$$f(x) = \sum_{i \in \{A, B, C\}} \alpha_i y_i K_3(x_i, x) + b = -K_3(A, x) + K_3(C, x) + b.$$

Nếu lấy $b = 0$ thì

$$f(D) = K_3(C, D) - K_3(A, D) = 46656 - 4096 = 42560 > 0,$$

vậy

D được phân loại là +1.

5.9 Ưu điểm

SVM là một trong những thuật toán học máy có giám sát mạnh mẽ, đặc biệt hiệu quả trong các bài toán phân loại. Dưới đây là các ưu điểm nổi bật của SVM:

- Xử lý dữ liệu phi tuyến với kỹ thuật kernel: SVM ánh xạ dữ liệu không phân tách tuyến tính vào không gian đặc trưng chiều cao bằng hàm kernel, cho phép phân tách bằng siêu phẳng mà không cần tính trực tiếp ánh xạ đặc trưng. Điều này giảm chi phí tính toán và xử lý hiệu quả các bài toán phức tạp [70].
- Chỉ sử dụng vector hỗ trợ: SVM dựa trên một số ít điểm dữ liệu gần siêu phẳng, gọi là vector hỗ trợ, có $\alpha_i > 0$ trong bài toán đối ngẫu (xem mục 5.7). Như đã đề cập, đây là chìa khóa để SVM chỉ sử dụng số lượng nhỏ vector hỗ trợ, giúp tiết kiệm bộ nhớ và tăng tốc độ suy luận [66].
- Hiệu quả trong không gian nhiều chiều: SVM hoạt động tốt khi số chiều đặc trưng lớn hơn số mẫu, như trong phân loại văn bản hoặc dữ liệu gen, nhờ tối ưu hóa khoảng cách biên và chỉ dựa trên vector hỗ trợ, đảm bảo độ chính xác cao [69].
- Xử lý dữ liệu không cân bằng: SVM điều chỉnh trọng số lớp trong hàm mục tiêu, tập trung vào vector hỗ trợ gần siêu phẳng, giúp phân tách hiệu quả ngay cả khi số lượng mẫu giữa các lớp chênh lệch [75].
- Linh hoạt với hàm kernel: SVM hỗ trợ nhiều hàm kernel như kernel tuyến tính, đa thức kernel và Gaussian kernel, thích ứng với các dạng phân bố dữ liệu khác nhau [70].
- Tốc độ suy luận nhanh: Sau huấn luyện, SVM chỉ tính toán trên vector hỗ trợ, đảm bảo suy luận nhanh, phù hợp với các ứng dụng thời gian thực như nhận dạng hình ảnh hoặc phân tích văn bản [72].

5.10 Bài tập

Bài tập 5.1. Cho ba điểm dữ liệu:

- $A(1, 3)$ thuộc lớp -1 .
- $B(2, 3)$ thuộc lớp -1 .
- $C(6, 1)$ thuộc lớp $+1$.

Xét ba đường thẳng phân chia:

$$y_1 = 0.5x + 1, \quad y_2 = -x + 6, \quad y_3 = 2x - 5$$

- Tính độ chính xác của mỗi đường thẳng trong việc phân loại các điểm trên.
- Với độ chính xác của mỗi đường thẳng trong việc phân loại trong câu a). Hãy xác định lớp của điểm mới $D(3, 3)$?
- Giả sử $\alpha_A = 0, \alpha_B = 1, \alpha_C = 1$. Tìm siêu phẳng tối ưu. Sau đó, xác định lớp của điểm $E(6, 3)$.

Bài tập 5.2. Cho các điểm:

$$P(1, 2), \quad Q(3, 4), \quad R(5, 2), \quad S(0, 1)$$

với nhãn tương ứng:

$$y_P = +1, \quad y_Q = -1, \quad y_R = +1, \quad y_S = -1$$

- Với các hệ số Lagrange: $\alpha_P = 0.6, \alpha_Q = 0.6, \alpha_R = 0.4, \alpha_S = 0.4$, hãy phân loại điểm mới $T(3, 3)$ bằng SVM tuyến tính.
- Sử dụng kernel $K(x, y) = 1 + x^T y$ và hệ số:

$$\alpha_P = 0, \quad \alpha_Q = 1, \quad \alpha_R = 0, \quad \alpha_S = 1$$

Hãy phân loại lại điểm T .

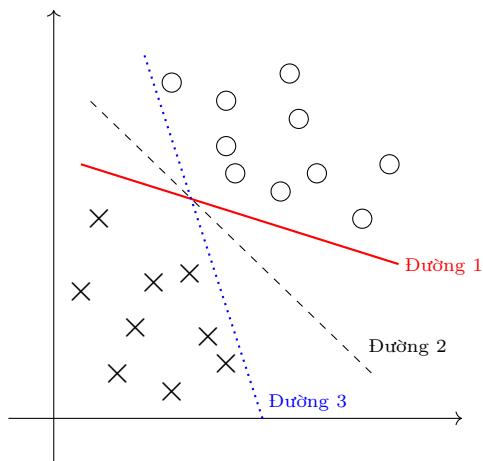
- Sử dụng kernel bậc ba $K(x, y) = (1 + x^T y)^3$ với hệ số:

$$\alpha_P = 1, \quad \alpha_Q = 1, \quad \alpha_R = 0, \quad \alpha_S = 0$$

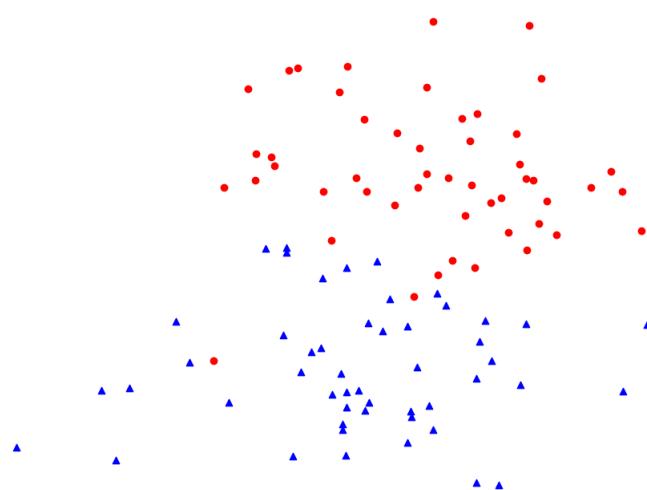
Phân loại lại điểm T .

Bài tập 5.3. Dựa vào Hình 5.9, với mẫu được ký hiệu bằng chữ X và O. Trong ba đường biên được biểu diễn, đường nào là đường phân chia tốt nhất? Giải thích dựa trên khái niệm *margin*.

Bài tập 5.4. Cho tập dữ liệu như Hình 5.10. Liệu SVM tuyến tính có thể tìm được lời giải cho tập dữ liệu này không? Giải thích tại sao. Nếu có, hãy vẽ đường biên phân chia; nếu không, hãy đề xuất cải tiến để giải quyết bài toán.



Hình 5.9: Xác định đường biên tốt nhất bằng trực quan



Hình 5.10: Biểu diễn tập dữ liệu không phân tách tuyến tính

Chương 6

Giải thuật k-means

6.1 Bài toán mở đầu

Trong bối cảnh ngành công nghiệp trò chơi điện tử phát triển nhanh chóng, các công ty phát triển game đang phải đổi mới với khối lượng dữ liệu khổng lồ về hành vi và sở thích của người chơi. Dữ liệu này bao gồm các thông tin như độ tuổi, thời gian chơi trung bình mỗi ngày, và thể loại game ưa thích, được thu thập từ hàng triệu người dùng trên các nền tảng trực tuyến. Việc phân tích dữ liệu này để xác định các nhóm người chơi có đặc điểm tương đồng là yếu tố then chốt để tối ưu hóa chiến lược phát triển sản phẩm, thiết kế tính năng game phù hợp, và triển khai các dịch vụ quảng cáo nhắm mục tiêu hiệu quả. Tuy nhiên, với sự đa dạng và phức tạp của dữ liệu người chơi, việc phân nhóm thủ công trở nên bất khả thi, đòi hỏi các phương pháp tự động hóa để khai thác thông tin một cách nhanh chóng và chính xác.

Để giải quyết vấn đề này, các hệ thống phân cụm tự động đã được phát triển, cho phép phân chia người chơi thành các nhóm dựa trên sự tương đồng của các đặc trưng. Các hệ thống này tận dụng các thuật toán học máy để phân tích dữ liệu lịch sử, chẳng hạn như thông tin về hành vi chơi game hoặc sở thích của người dùng. Thông qua việc phân tích các đặc trưng như độ tuổi, thời gian chơi, hoặc giá trị số hóa của thể loại game (được mã hóa thông qua các kỹ thuật như *one-hot encoding*), hệ thống có thể tự động xác định các nhóm người chơi có hành vi tương tự với độ chính xác cao.

Những hệ thống này có khả năng thực hiện phân cụm trong thời gian thực, giúp các công ty game nhanh chóng nhận diện các nhóm người chơi để đưa ra các quyết định chiến lược. Ví dụ, khi một nhóm người chơi được xác định là chủ yếu yêu thích các thể loại game nhập vai (*RPG*) và dành nhiều thời gian chơi mỗi ngày, công ty có thể tập trung phát triển các tính năng mới phù hợp với nhóm này hoặc thiết kế các dịch vụ quảng cáo nhắm đến sở thích của họ.

Trong số các thuật toán học máy được sử dụng cho bài toán này, thuật toán phân cụm k-means (*k-means Clustering*) nổi bật như một phương pháp mạnh mẽ và hiệu quả, đặc biệt trong các bài toán phân cụm không giám sát [76]. Thuật toán này hoạt động bằng cách phân chia dữ liệu thành k cụm (*clusters*), sao cho tổng bình phương khoảng cách từ các điểm dữ liệu đến tâm cụm (*centroid*) của chúng được tối thiểu hóa. Điểm mạnh của phân cụm k-means nằm ở khả năng xử lý các tập dữ liệu lớn với nhiều đặc trưng, chẳng hạn như dữ liệu người chơi với các đặc trưng đa chiều như độ tuổi, thời gian chơi, và sở thích thể loại game. Hơn nữa, thuật toán này hỗ trợ khả năng điều chỉnh tham số k , cho phép người dùng kiểm soát số lượng cụm phù hợp với mục tiêu phân tích.

6.2 Khái niệm

Thuật toán phân cụm k-means (*k-means clustering*) là một phương pháp nền tảng trong lĩnh vực học máy không giám sát, một nhánh của học máy tập trung vào việc khám phá các cấu trúc ẩn trong dữ liệu mà không cần thông tin nhãn (*unlabeled data*). Khác với học máy có giám sát, nơi dữ liệu được gán nhãn để hướng dẫn quá trình học, thì học máy không giám sát yêu cầu thuật toán tự động nhận diện các mẫu hoặc nhóm dựa trên sự tương đồng nội tại của dữ liệu [77]. Trong bối cảnh này, phân cụm k-means được thiết kế để giải quyết bài toán phân cụm (*clustering*), với mục tiêu chia tập dữ liệu thành các cụm sao cho các điểm dữ liệu trong cùng một cụm có mức độ tương đồng cao, trong khi các cụm khác nhau được phân tách rõ rệt dựa trên các đặc trưng số hóa.

Thuật ngữ phân cụm k-means (*k-means clustering*) được xây dựng từ ba thành phần chính, phản ánh cơ chế hoạt động của thuật toán:

- ***k***: Đại diện cho số lượng cụm được xác định trước. Giá trị *k* là một tham số quan trọng, ảnh hưởng trực tiếp đến chất lượng phân cụm.
- ***means***: Đề cập đến việc sử dụng giá trị trung bình của các điểm dữ liệu trong mỗi cụm, được gọi là tâm cụm (*centroid*), làm đại diện cho cụm đó. Tâm cụm được tính toán dựa trên giá trị trung bình của các đặc trưng số hóa (*numerical features*) của các điểm thuộc cụm, đảm bảo rằng nó phản ánh đặc điểm chung của cụm.
- ***clustering***: Quá trình phân chia dữ liệu thành các nhóm dựa trên sự tương đồng, thường được đo lường bằng cách Euclidean trong không gian đặc trưng. Sự tương đồng này đảm bảo rằng các điểm dữ liệu trong cùng một cụm có các đặc trưng gần giống nhau hơn so với các điểm ở các cụm khác.

Mục tiêu của phân cụm k-means là chia tập dữ liệu thành các cụm sao cho các điểm trong cùng cụm có mức độ tương đồng cao, thường được đo lường thông qua khoảng cách trong không gian đặc trưng. Thuật toán này nổi bật nhờ tính đơn giản, hiệu quả tính toán, và khả năng ứng dụng trong các lĩnh vực như phân khúc khách hàng, phân tích thị trường, và xử lý dữ liệu lớn [77].

6.3 Giải thuật

Cho một tập dữ liệu X gồm n điểm dữ liệu, trong đó mỗi điểm x_i là một vector đặc trưng trong không gian d -chiều, được biểu diễn dưới dạng $X = [x_1, x_2, \dots, x_n] \in \mathbb{R}^{d \times n}$, với $x_i \in \mathbb{R}^d$. Mục tiêu của thuật toán phân cụm k-means là phân chia tập dữ liệu này thành k cụm S_1, S_2, \dots, S_k , sao cho các điểm dữ liệu trong cùng một cụm có sự tương đồng cao, thường được đo bằng khoảng cách Euclidean trong không gian đặc trưng. Ý tưởng cốt lõi của thuật toán là gán mỗi điểm dữ liệu vào cụm có tâm cụm gần nhất, sau đó cập nhật các tâm cụm dựa trên giá trị trung bình của các điểm trong cụm, và lặp lại quá trình này để đạt được sự phân cụm tối ưu [76].

Mô tả của thuật toán k-means được mô tả trong Thuật toán 6.1, bao gồm các bước

1. **Khởi tạo:** Chọn k điểm dữ liệu làm tâm cụm ban đầu. Việc khởi tạo này có thể được thực hiện ngẫu nhiên, nhưng để cải thiện hiệu quả, các phương pháp như *k-means++* thường được sử dụng để chọn các tâm cụm ban đầu sao cho chúng phân bố đều hơn trong không gian dữ liệu [78].

Algorithm 6.1: Thuật toán phân cụm *k-means*

Input: Tập dữ liệu $X = \{x_1, \dots, x_n\}$, số cụm k , ngưỡng hội tụ ε , số lặp tối đa t_{\max}
Output: Các cụm S_1, \dots, S_k và tâm cụm c_1, \dots, c_k

```

// Khởi tạo
1 for  $i \leftarrow 1$  đến  $k$  do
2    $c_i^{(0)} \leftarrow x_{0i}$ , với  $x_{0i}$  được chọn ngẫu nhiên từ  $X$ ;
3    $t \leftarrow 0$ ;
4 repeat
5    $t \leftarrow t + 1$ ;
   // Bước gán cụm
6   foreach  $x_i \in X$  do
7      $j^* \leftarrow \arg \min_{1 \leq j \leq k} \|x_i - c_j^{(t-1)}\|_2$ ;
8      $S_{j^*}^{(t)} \leftarrow S_{j^*}^{(t)} \cup \{x_i\}$ ;
   // Bước cập nhật tâm cụm
9   for  $j \leftarrow 1$  đến  $k$  do
10     $c_j^{(t)} \leftarrow \frac{1}{|S_j^{(t)}|} \sum_{x_i \in S_j^{(t)}} x_i$ ;
11 until  $\max_j \|c_j^{(t)} - c_j^{(t-1)}\|_2 < \varepsilon$  hoặc  $t \geq t_{\max}$ ;
12 return  $S_1^{(t)}, \dots, S_k^{(t)}, c_1^{(t)}, \dots, c_k^{(t)}$ 

```

2. **Gán cụm:** Với mỗi điểm dữ liệu x_i , gán nó vào cụm S_j có tâm cụm c_j gần nhất, dựa trên khoảng cách Euclidean, tức là:

$$S_j = \{x_i \mid \|x_i - c_j\|_2 \leq \|x_i - c_l\|_2, \forall l \neq j\}.$$

3. **Cập nhật tâm cụm:** Tính lại tâm cụm c_i cho mỗi cụm S_i bằng cách lấy trung bình các tọa độ của tất cả các điểm dữ liệu trong cụm:

$$c_i = \frac{1}{|S_i|} \sum_{x_j \in S_i} x_j,$$

trong đó $|S_i|$ là số lượng điểm trong cụm S_i .

4. **Lặp lại:** Lặp lại các bước gán cụm và cập nhật tâm cụm cho đến khi các tâm cụm không thay đổi đáng kể (tức là đạt điểm hội tụ (*convergence*)) hoặc đạt số lần lặp tối đa t_{\max} được xác định trước.

Quá trình này đảm bảo rằng các điểm dữ liệu được phân chia vào các cụm sao cho tổng khoảng cách từ các điểm đến tâm cụm tương ứng được giảm thiểu, dẫn đến sự tương đồng tối đa trong mỗi cụm [76].

6.3.1 Hàm mất mát

Để định lượng mức độ tối ưu của việc phân cụm, thuật toán phân cụm k-means dựa trên một hàm mất mát (*loss function*) nhằm đo độ sai lệch giữa các điểm dữ liệu và tâm cụm tương ứng.

6.3. GIẢI THUẬT

Trong đó, hàm đo độ sai lệch phổ biến nhất là tổng bình phương khoảng cách Euclid, và thuật toán tìm cách cực tiểu hóa tổng bình phương khoảng cách này để đạt được phân cụm tối ưu. Cụ thể, với mỗi cụm S_i , mục tiêu là tìm tâm cụm c_i sao cho tổng bình phương khoảng cách Euclidean từ các điểm $x_j \in S_i$ đến c_i là nhỏ nhất, được biểu diễn bằng:

$$\sum_{x_j \in S_i} \|x_j - c_i\|_2^2. \quad (6.1)$$

Khi xét trên toàn bộ tập dữ liệu với k cụm, *hàm mất mát tổng quát*, thường được gọi là *tổng bình phương khoảng cách trong cụm* (*within-cluster sum of squares - WCSS*), được định nghĩa như sau:

$$\sum_{i=1}^k \sum_{x_j \in S_i} \|x_j - c_i\|_2^2. \quad (6.2)$$

Bài toán tối ưu hóa phân cụm k-means tương đương với việc tìm tập hợp các cụm $\{S_1, S_2, \dots, S_k\}$ và các tâm cụm $\{c_1, c_2, \dots, c_k\}$ sao cho *hàm mất mát* tại Công thức (6.2) đạt giá trị nhỏ nhất.

Tuy nhiên, việc tìm nghiệm tối ưu toàn cục cho *hàm mất mát* này là một bài toán phức tạp tính toán, cụ thể là thuộc lớp bài toán NP-khó khi k và d lớn [79]. Do đó, thuật toán phân cụm k-means sử dụng phương pháp lặp để tìm nghiệm gần đúng, thông qua quá trình gán cụm và cập nhật tâm cụm như đã mô tả. Mặc dù không đảm bảo đạt được nghiệm tối ưu toàn cục, thuật toán thường hội tụ nhanh chóng đến một nghiệm cục bộ tốt, đặc biệt khi sử dụng các kỹ thuật khởi tạo như *k-means++* [78].

6.3.2 Giải thuật tối ưu

Quá trình tối ưu hóa của thuật toán phân cụm k-means nhằm cực tiểu hóa *hàm mất mát* tại Công thức (6.2), được định nghĩa là tổng bình phương khoảng cách từ các điểm dữ liệu đến tâm cụm (*centroid*) tương ứng. Thuật toán thực hiện điều này thông qua một quy trình lặp xen kẽ giữa hai bước: gán nhãn cho các điểm dữ liệu dựa trên tâm cụm gần nhất và cập nhật các tâm cụm để giảm *hàm mất mát*. Mỗi bước được thiết kế để đảm bảo *hàm mất mát* giảm dần, tiến tới một nghiệm cục bộ tối ưu [76].

Trong bước gán nhãn, mỗi điểm dữ liệu x_j được phân bổ vào cụm S_i có tâm cụm c_i sao cho khoảng cách Euclidean $\|x_j - c_i\|_2$ là nhỏ nhất so với các tâm cụm khác. Điều này đảm bảo rằng, với tập hợp tâm cụm cố định, *hàm mất mát* tại Công thức (6.2) được tối ưu cục bộ cho từng điểm dữ liệu, vì việc chọn cụm gần nhất giảm thiểu đóng góp của x_j vào tổng bình phương khoảng cách [76].

Trong bước cập nhật tâm cụm, với mỗi cụm S_i , thuật toán tìm tâm cụm mới c_i^* sao cho tổng bình phương khoảng cách từ các điểm $x_j \in S_i$ đến c_i là nhỏ nhất:

$$c_i^* = \operatorname{argmin}_{c_i} \sum_{x_j \in S_i} \|x_j - c_i\|_2^2. \quad (6.3)$$

Gọi $f(c_i) = \sum_{x_j \in S_i} \|x_j - c_i\|_2^2$, ta tính đạo hàm của $f(c_i)$ theo c_i và đặt bằng 0:

$$\frac{\partial f(c_i)}{\partial c_i} = 2 \sum_{x_j \in S_i} (c_i - x_j) = 0. \quad (6.4)$$

Giải phương trình:

$$\begin{aligned} \sum_{x_j \in S_i} (c_i - x_j) &= 0, \\ |S_i|c_i &= \sum_{x_j \in S_i} x_j, \\ c_i^* &= \frac{1}{|S_i|} \sum_{x_j \in S_i} x_j, \end{aligned}$$

trong đó $|S_i|$ là số lượng điểm trong cụm S_i . Kết quả này cho thấy tâm cụm mới c_i^* là trung bình cộng của các điểm trong cụm, đảm bảo cực tiểu hóa hàm mất mát trong cụm [76].

Quá trình lặp xen kẽ giữa gán nhãn và cập nhật tâm cụm đảm bảo rằng hàm mất mát trong Công thức (6.2) giảm dần sau mỗi bước. Do hàm mất mát là không âm và bị chặn dưới, theo định lý Weierstrass, thuật toán phân cụm k-means luôn hội tụ (*converge*) đến một nghiệm cục bộ sau một số lần lặp hữu hạn [80]. Tuy nhiên, vì bài toán tối ưu hóa là NP-khó, nghiệm đạt được không nhất thiết là tối ưu toàn cục, và kết quả phụ thuộc vào việc khởi tạo tâm cụm [79]. Kỹ thuật *k-means++* được sử dụng để cải thiện khởi tạo, tăng khả năng đạt nghiệm tốt hơn [78].

Ví dụ 6.1. Xét một tập hợp gồm 6 điểm dữ liệu trong không gian 2 chiều như sau:

Điểm	Tọa độ
A1	(1, 4)
A2	(3, 4)
A3	(4, 5)
B1	(8, 5)
B2	(5, 6)
B3	(2, 5)

Hãy thực hiện Thuật toán phân cụm k-means với $k = 2$ cụm (*clusters*), sử dụng khoảng cách Euclidean để gán điểm và cập nhật tâm cụm (*centroid*). Công thức khoảng cách Euclidean giữa hai điểm (x_1, y_1) và (x_2, y_2) là:

$$d = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}.$$

Bước 1: Khởi tạo tâm cụm. Chọn ngẫu nhiên hai điểm làm tâm cụm ban đầu:

- Tâm cụm 1: $C_1 = (3, 4)$ (tọa độ của A2).
- Tâm cụm 2: $C_2 = (2, 5)$ (tọa độ của B3).

Bước 2: Tính khoảng cách Euclidean và gán cụm.

Tính khoảng cách từ mỗi điểm đến hai tâm cụm và gán điểm vào cụm có tâm cụm gần nhất, như trình bày trong Bảng 6.1.

Kết quả gán:

- Cụm 1: {A2, A3, B1, B2}.
- Cụm 2: {A1, B3}.

Bước 3: Cập nhật tâm cụm.

Tính trung bình tọa độ của các điểm trong mỗi cụm:

6.3. GIẢI THUẬT

Bảng 6.1: Khoảng cách Euclidean đến các tâm cụm ban đầu

Điểm	$d(C_1)$	$d(C_2)$	Cụm được gán
A1 (1, 4)	$\sqrt{(3-1)^2 + (4-4)^2} = 2$	$\sqrt{(2-1)^2 + (5-4)^2} \approx 1.41$	cụm 2
A2 (3, 4)	$\sqrt{(3-3)^2 + (4-4)^2} = 0$	$\sqrt{(2-3)^2 + (5-4)^2} \approx 1.41$	cụm 1
A3 (4, 5)	$\sqrt{(3-4)^2 + (4-5)^2} \approx 1.41$	$\sqrt{(2-4)^2 + (5-5)^2} = 2$	cụm 1
B1 (8, 5)	$\sqrt{(3-8)^2 + (4-5)^2} \approx 5.10$	$\sqrt{(2-8)^2 + (5-5)^2} = 6$	cụm 1
B2 (5, 6)	$\sqrt{(3-5)^2 + (4-6)^2} \approx 2.83$	$\sqrt{(2-5)^2 + (5-6)^2} \approx 3.16$	cụm 1
B3 (2, 5)	$\sqrt{(3-2)^2 + (4-5)^2} \approx 1.41$	$\sqrt{(2-2)^2 + (5-5)^2} = 0$	cụm 2

- Cụm 1: $C_1^{\text{mới}} = \left(\frac{3+4+8+5}{4}, \frac{4+5+5+6}{4} \right) = (5, 5).$
- Cụm 2: $C_2^{\text{mới}} = \left(\frac{1+2}{2}, \frac{4+5}{2} \right) = (1.5, 4.5).$

Bước 4: Lặp lại với tâm cụm mới.

Tính lại khoảng cách đến $C_1 = (5, 5)$ và $C_2 = (1.5, 4.5)$ như trình bày trong Bảng 6.2.

Bảng 6.2: Khoảng cách Euclidean đến các tâm cụm mới

Điểm	$d(C_1)$	$d(C_2)$	Cụm được gán
A1 (1, 4)	$\sqrt{(5-1)^2 + (5-4)^2} \approx 4.12$	$\sqrt{(1.5-1)^2 + (4.5-4)^2} \approx 0.71$	cụm 2
A2 (3, 4)	$\sqrt{(5-3)^2 + (5-4)^2} \approx 2.24$	$\sqrt{(1.5-3)^2 + (4.5-4)^2} \approx 1.58$	cụm 2
A3 (4, 5)	$\sqrt{(5-4)^2 + (5-5)^2} \approx 1.00$	$\sqrt{(1.5-4)^2 + (4.5-5)^2} \approx 2.55$	cụm 1
B1 (8, 5)	$\sqrt{(5-8)^2 + (5-5)^2} \approx 3.00$	$\sqrt{(1.5-8)^2 + (4.5-5)^2} \approx 6.50$	cụm 1
B2 (5, 6)	$\sqrt{(5-5)^2 + (5-6)^2} \approx 1.00$	$\sqrt{(1.5-5)^2 + (4.5-6)^2} \approx 3.81$	cụm 1
B3 (2, 5)	$\sqrt{(5-2)^2 + (5-5)^2} \approx 3.00$	$\sqrt{(1.5-2)^2 + (4.5-5)^2} \approx 0.71$	cụm 2

Kết quả gán:

- Cụm 1: {A3, B1, B2}.
- Cụm 2: {A1, A2, B3}.

Bước 5: Cập nhật tâm cụm mới. Tính trung bình tọa độ:

- Cụm 1: $C_1^{\text{mới}} = \left(\frac{4+8+5}{3}, \frac{5+5+6}{3} \right) = \left(\frac{17}{3}, \frac{16}{3} \right) \approx (5.67, 5.33).$
- Cụm 2: $C_2^{\text{mới}} = \left(\frac{1+3+2}{3}, \frac{4+4+5}{3} \right) = \left(2, \frac{13}{3} \right) \approx (2, 4.33).$

Bước 6: Kiểm tra hội tụ. Tính lại khoảng cách với $C_1 = (5.67, 5.33)$ và $C_2 = (2, 4.33)$ như trình bày trong Bảng 6.3.

Kết quả gán không thay đổi so với bước trước (cụm 1: {A3, B1, B2}, cụm 2: {A1, A2, B3}), cho thấy thuật toán đã hội tụ. Các tâm cụm cuối cùng là:

- Tâm cụm 1: $C_1 = (5.67, 5.33).$
- Tâm cụm 2: $C_2 = (2, 4.33).$

Bảng 6.3: Khoảng cách Euclidean đến các tâm cụm mới

Điểm	$d(C_1)$	$d(C_2)$	Cụm được gán
A1 (1, 4)	$\sqrt{(5.67 - 1)^2 + (5.33 - 4)^2} \approx 4.85$	$\sqrt{(2 - 1)^2 + (4.33 - 4)^2} \approx 1.05$	cụm 2
A2 (3, 4)	$\sqrt{(5.67 - 3)^2 + (5.33 - 4)^2} \approx 2.84$	$\sqrt{(2 - 3)^2 + (4.33 - 4)^2} \approx 1.05$	cụm 2
A3 (4, 5)	$\sqrt{(5.67 - 4)^2 + (5.33 - 5)^2} \approx 1.74$	$\sqrt{(2 - 4)^2 + (4.33 - 5)^2} \approx 2.11$	cụm 1
B1 (8, 5)	$\sqrt{(5.67 - 8)^2 + (5.33 - 5)^2} \approx 2.36$	$\sqrt{(2 - 8)^2 + (4.33 - 5)^2} \approx 6.04$	cụm 1
B2 (5, 6)	$\sqrt{(5.67 - 5)^2 + (5.33 - 6)^2} \approx 0.94$	$\sqrt{(2 - 5)^2 + (4.33 - 6)^2} \approx 3.43$	cụm 1
B3 (2, 5)	$\sqrt{(5.67 - 2)^2 + (5.33 - 5)^2} \approx 3.68$	$\sqrt{(2 - 2)^2 + (4.33 - 5)^2} \approx 0.67$	cụm 2

Kết luận: Thuật toán phân cụm k-means đã hoàn tất sau ba lần lặp, với các cụm ổn định và hàm mất mát đạt nghiệm cục bộ. Kết quả phân cụm phản ánh hai nhóm điểm có sự tương đồng cao trong không gian 2 chiều.

6.4 Một số phương pháp hỗ trợ k-means

Trong học máy không giám sát, khác với học máy có giám sát nơi dữ liệu đã được gán nhãn để đánh giá hiệu suất mô hình, các thuật toán phân cụm như *k-means* phải đối mặt với thách thức trong việc xác định số cụm tối ưu và đánh giá chất lượng phân cụm khi không có tiêu chuẩn đánh giá rõ ràng.

Thuật toán *k-means* yêu cầu người dùng chỉ định trước số cụm k , một tham số có ảnh hưởng lớn đến kết quả phân cụm. Nếu k được chọn không phù hợp, các cụm thu được có thể không phản ánh đúng cấu trúc tự nhiên của dữ liệu. Do đó, việc xác định giá trị k tối ưu là một bài toán quan trọng trong phân cụm. Các phương pháp phổ biến hỗ trợ quá trình này bao gồm phương pháp khuỷu tay (*Elbow Method*) [81] và chỉ số Silhouette (*Silhouette Score*) [82]. Những phương pháp này giúp đánh giá mức độ phù hợp của mô hình và hỗ trợ lựa chọn giá trị k đảm bảo cân bằng giữa khả năng mô tả chi tiết và tính tổng quát của mô hình.

Mục tiêu của phân cụm là chia tập dữ liệu thành các nhóm sao cho các điểm trong cùng cụm có mức độ tương đồng cao, trong khi các cụm khác biệt nhau rõ rệt - thường được đo bằng khoảng cách Euclidean trong không gian đặc trưng. Các tiêu chí hỗ trợ *k-means* tập trung vào việc định lượng chất lượng phân cụm và lựa chọn số cụm k phù hợp, dựa trên các thước đo như tổng bình phương khoảng cách trong cụm (Within-Cluster Sum of Squares - WCSS) hoặc độ tương đồng trung bình giữa các điểm dữ liệu và cụm của chúng.

6.4.1 Phương pháp khuỷu tay

Fương pháp khuỷu tay là một kỹ thuật heuristic phổ biến để ước lượng số cụm k tối ưu trong phân cụm. Phương pháp này dựa trên việc tính *tổng bình phương khoảng cách trong cụm* – *Within-Cluster Sum of Squares* (WCSS), được định nghĩa trong Công thức (6.5).

$$\text{WCSS}(k) = \sum_{i=1}^k \sum_{x_j \in S_i} \|x_j - c_i\|_2^2, \quad (6.5)$$

trong đó S_i là cụm thứ i , và c_i là tâm cụm của cụm đó.

Quá trình thực hiện bao gồm:

6.4. MỘT SỐ PHƯƠNG PHÁP HỖ TRỢ K-MEANS

1. Chạy thuật toán phân cụm k-means với các giá trị k khác nhau (thường từ 1 đến một số lượng lớn, ví dụ k_{\max}).
2. Vẽ đồ thị WCSS theo k , trong đó WCSS giảm khi k tăng do các cụm trở nên nhỏ hơn.
3. Xác định điểm “khuỷu tay” trên đồ thị, nơi WCSS giảm mạnh ban đầu nhưng bắt đầu chậm lại, hình thành một góc khuỷu. Giá trị k tại điểm này được xem là số cụm tối ưu.

Phương pháp khuỷu tay thể hiện sự đánh đổi giữa việc giảm WCSS và tránh sử dụng quá nhiều cụm, nhằm hạn chế hiện tượng quá khớp trong phân cụm. Tuy nhiên, phương pháp này vẫn tồn tại những hạn chế nhất định. Khi đồ thị không xuất hiện điểm khuỷu rõ ràng, việc lựa chọn số cụm k trở nên chủ quan và thiếu cơ sở định lượng vững chắc. Ngoài ra, WCSS chỉ phản ánh độ chật bên trong cụm (*intra-cluster compactness*) mà không xem xét độ tách biệt giữa các cụm (*inter-cluster separation*), khiến phương pháp có thể không nhận diện được các cấu trúc phân cụm phức tạp.

6.4.2 Điểm số Silhouette

Điểm số Silhouette là một phương pháp định lượng chất lượng phân cụm bằng cách đo lường mức độ tương đồng của mỗi điểm dữ liệu với cụm mà nó thuộc về, so với mức độ tương đồng với cụm gần nhất khác. Chỉ số này cung cấp một thước đo trực quan và định lượng, cho phép đánh giá mức độ tách biệt và đồng nhất của các cụm. So với phương pháp khuỷu tay, điểm số Silhouette thường mang lại kết quả đáng tin cậy hơn, đặc biệt trong các tập dữ liệu có cấu trúc phức tạp hoặc số cụm không rõ ràng.

Với mỗi điểm dữ liệu x_i trong cụm $C(i)$, ta tính các giá trị sau:

- *Khoảng cách trung bình trong cụm (a^i):* Trung bình khoảng cách Euclidean từ x_i đến tất cả các điểm khác trong cùng cụm $C(i)$, được tính bằng Công thức (6.6):

$$a^i = \frac{1}{|C(i)| - 1} \sum_{j \in C(i), j \neq i} d(i, j), \quad (6.6)$$

trong đó $d(i, j) = \|x_i - x_j\|_2$ là khoảng cách Euclidean, và $|C(i)|$ là số điểm trong cụm $C(i)$. Nếu $|C(i)| = 1$, đặt $a^i = 0$.

- *Khoảng cách trung bình đến cụm gần nhất (b^i):* Trung bình khoảng cách từ x_i đến tất cả các điểm trong cụm gần nhất khác $C(k)$, với $k \neq C(i)$, được tính bằng Công thức (6.7):

$$b^i = \min_{k \neq C(i)} \frac{1}{|C(k)|} \sum_{j \in C(k)} d(i, j). \quad (6.7)$$

- *Giá trị Silhouette (s^i):* Thước đo mức độ tương đồng của x_i với cụm của nó so với cụm gần nhất khác, được tính bằng Công thức (6.8):

$$s^i = \frac{b^i - a^i}{\max(a^i, b^i)}, \quad (6.8)$$

nếu $|C(i)| > 1$; nếu $|C(i)| = 1$, đặt $s^i = 0$.

Giá trị s^i nằm trong khoảng $[-1, 1]$:

- $s^i \approx 1$: Điểm x_i nằm trong cụm đúng, cách xa các cụm khác.

- $s^i \approx 0$: Điểm x_i nằm gần ranh giới giữa hai cụm.
- $s^i < 0$: Điểm x_i có thể đã được gán sai cụm.

Để chọn k tối ưu, thuật toán phân cụm k-means được thực thi với các giá trị k khác nhau, sau đó tính giá trị Silhouette trung bình trên toàn bộ tập dữ liệu như Công thức (6.9):

$$s(k) = \frac{1}{n} \sum_{i=1}^n s^i. \quad (6.9)$$

Giá trị k cho Silhouette trung bình cao nhất thường được chọn, vì điều này biểu thị các cụm có sự tương đồng trong cụm cao và tách biệt tốt giữa các cụm.

So với phương pháp khuỷu tay, chỉ số Silhouette có ưu điểm nổi bật là đồng thời xem xét mức độ đồng nhất bên trong cụm (*cohesion*) và mức độ tách biệt giữa các cụm (*separation*). Điều này giúp đánh giá chất lượng phân cụm một cách toàn diện và chính xác hơn, đặc biệt khi dữ liệu có cấu trúc phức tạp hoặc khi số cụm không dễ xác định. Tuy nhiên, điểm hạn chế của phương pháp này là chi phí tính toán cao, do cần phải tính toán khoảng cách giữa mỗi điểm và tất cả các điểm còn lại, khiến nó kém hiệu quả hơn khi áp dụng cho các tập dữ liệu lớn.

6.4.3 So sánh và ứng dụng

Cả phương pháp khuỷu tay và điểm số Silhouette đều là những công cụ phổ biến hỗ trợ thuật toán thuật toán *k-means* trong việc xác định số lượng cụm tối ưu. Tuy nhiên, hai phương pháp này có những đặc trưng và phạm vi ứng dụng khác nhau, được tóm tắt trong Bảng 6.4.

Bảng 6.4: So sánh giữa phương pháp khuỷu tay và điểm số Silhouette

Tiêu chí	Phương pháp khuỷu tay	Điểm số Silhouette
Nguyên lý	Dựa trên tốc độ giảm của WCSS	Dựa trên độ chát trong cụm và độ tách biệt giữa các cụm
Ưu điểm	Đơn giản, trực quan, dễ triển khai	Đánh giá toàn diện, phản ánh chất lượng phân cụm tốt hơn
Nhược điểm	Chủ quan nếu không có “điểm khuỷu” rõ ràng	Chi phí tính toán cao với tập dữ liệu lớn
Thích hợp cho	Dữ liệu nhỏ, cấu trúc cụm rõ ràng	Dữ liệu phức tạp, cần đánh giá khách quan hơn

Trong thực tiễn, hai phương pháp này thường được sử dụng kết hợp để tăng độ tin cậy của quá trình lựa chọn số cụm k . Cụ thể, phương pháp khuỷu tay được áp dụng trước để xác định phạm vi giá trị k hợp lý, sau đó chỉ số Silhouette được sử dụng để lựa chọn giá trị k tối ưu trong phạm vi đó. Cách tiếp cận này đã được ứng dụng rộng rãi trong các lĩnh vực như phân khúc khách hàng, phân tích hình ảnh và khai phá dữ liệu, nhằm nâng cao hiệu quả của thuật toán phân cụm *k-means* trong việc khám phá cấu trúc tiềm ẩn của dữ liệu.

6.5 Hạn chế

Thuật toán phân cụm k-means là một phương pháp đơn giản và hiệu quả, đặc biệt phù hợp với các tập dữ liệu có cấu trúc cụm gần dạng hình cầu. Tuy nhiên, k-means cũng tồn tại một số hạn chế quan trọng cần được xem xét khi áp dụng trong thực tế:

- **Yêu cầu xác định trước số cụm k :** Thuật toán đòi hỏi người dùng chỉ định trước số cụm k . Việc lựa chọn k không phù hợp có thể dẫn đến phân cụm sai lệch so với cấu trúc tự nhiên của dữ liệu. Các kỹ thuật như phương pháp khuỷu tay hoặc điểm số Silhouette thường được sử dụng để ước lượng giá trị k tối ưu [82].
- **Nhạy cảm với khởi tạo tâm cụm:** Kết quả phân cụm phụ thuộc mạnh vào vị trí tâm cụm ban đầu. Việc khởi tạo ngẫu nhiên có thể khiến thuật toán hội tụ đến nghiệm cục bộ không mong muốn, đặc biệt khi dữ liệu có cấu trúc phức tạp. Phương pháp $k\text{-means}++$ giúp giảm rủi ro này bằng cách lựa chọn tâm ban đầu phân bố đều hơn, cải thiện chất lượng nghiệm [78].
- **Giả định về kích thước và mật độ cụm:** Thuật toán k-means có xu hướng tạo ra các cụm có kích thước và mật độ tương đối đồng đều, do dựa vào khoảng cách Euclidean đến tâm cụm. Khi dữ liệu chứa các cụm chênh lệch lớn về kích thước hoặc mật độ, thuật toán có thể phân cụm không chính xác, chẳng hạn tách một cụm lớn thành nhiều phần để cân bằng với cụm nhỏ hơn [77].
- **Hạn chế về hình dạng cụm:** Do sử dụng khoảng cách Euclidean, k-means hoạt động tốt với các cụm có hình dạng đơn giản, khả năng mở rộng và hiệu quả tính toán. Khi kết hợp với các kỹ thuật cải tiến như $k\text{-means}++$ hoặc được áp dụng đúng bối cảnh dữ liệu, thuật toán tiếp tục đóng vai trò quan trọng trong nhiều ứng dụng như phân khúc khách hàng, phân tích thị trường và tiền xử lý dữ liệu trong học máy không giám sát.

6.6 Bài tập

Bài tập 6.1. Giải thích tại sao tâm cụm của một cụm sau khi cập nhật trong thuật toán phân cụm k-means được tính là trung bình cộng của các điểm dữ liệu thuộc cụm đó. Hãy sử dụng lý thuyết tối ưu hóa hàm mất mát (6.3) để làm rõ câu trả lời.

Bài tập 6.2. Trong thuật toán phân cụm k-means, khoảng cách Euclidean thường được sử dụng để đo sự tương đồng giữa các điểm dữ liệu và tâm cụm.

1. Có bắt buộc phải sử dụng khoảng cách Euclidean không? Hãy giải thích.
2. Có thể sử dụng các thước đo khoảng cách khác như khoảng cách Manhattan hoặc khoảng cách Minkowski không? Nếu có, hãy nêu ví dụ về một trường hợp cụ thể mà thước đo khác có thể phù hợp hơn.
3. Việc lựa chọn thước đo khoảng cách ảnh hưởng như thế nào đến kết quả phân cụm? Hãy phân tích tác động của thước đo khoảng cách đến hình dạng và chất lượng của các cụm.

Bài tập 6.3. Xét tập dữ liệu gồm 6 điểm trong không gian 2 chiều, với tọa độ được cho trong Bảng 6.5.

Thực hiện các nhiệm vụ sau để phân cụm và đánh giá chất lượng phân cụm:

1. Thực hiện thuật toán phân cụm k-means thủ công với $k = 2, 3, 4$.

Bảng 6.5: Tọa độ các điểm dữ liệu

Điểm	Tọa độ
A	(1, 2)
B	(1, 4)
C	(2, 2)
D	(4, 4)
E	(5, 5)
F	(6, 8)

- Chọn ngẫu nhiên các tâm cụm ban đầu (ví dụ, chọn các điểm trong tập dữ liệu).
 - Tính khoảng cách Euclidean từ mỗi điểm đến các tâm cụm, gán điểm vào cụm có tâm cụm gần nhất.
 - Cập nhật tâm cụm bằng cách tính trung bình cộng tọa độ của các điểm trong mỗi cụm.
 - Lặp lại quá trình cho đến khi các cụm ổn định (hội tụ - converge).
 - Với mỗi k , ghi lại các cụm cuối cùng, tâm cụm, và tính tổng bình phương khoảng cách (within-cluster sum of squares - WCSS).
2. Phương pháp khuỷu tay (Elbow Method): Sử dụng các giá trị WCSS thu được từ các lần chạy k-means với $k = 2, 3, 4$.
- Vẽ đồ thị WCSS theo k .
 - Xác định điểm “khuỷu tay” trên đồ thị, nơi WCSS giảm chậm lại, và đề xuất số cụm k tối ưu.
3. Điểm số Silhouette (Silhouette Score): Với $k = 2$ và $k = 3$, tính chỉ số Silhouette cho mỗi điểm dữ liệu dựa trên các công thức (6.6), (6.7), và (6.8).
- Tính khoảng cách trung bình trong cụm (a^i) và khoảng cách trung bình đến cụm gần nhất (b^i) cho mỗi điểm.
 - Tính giá trị Silhouette (s^i) và giá trị Silhouette trung bình ($s(k)$) cho mỗi k .
 - Dựa trên giá trị $s(k)$, đánh giá mức độ tương đồng của các cụm.
4. Thảo luận:
- Dựa trên kết quả từ Phương pháp khuỷu tay và Điểm số Silhouette, hãy đề xuất giá trị k tối ưu cho tập dữ liệu này và giải thích lý do.
 - Trong thực tế, bạn ưu tiên sử dụng Phương pháp khuỷu tay hay Điểm số Silhouette để chọn k ? Hãy phân tích ưu và nhược điểm của từng phương pháp dựa trên bài toán này.

Chương 7

Giải thuật di truyền

7.1 Bài toán mở đầu

Trong các nhà máy sản xuất hoặc công ty cung cấp dịch vụ, việc sắp xếp lịch làm việc cho nhân viên, máy móc hoặc ca kíp làm việc là một vấn đề quan trọng nhưng đầy thách thức. Mỗi nhiệm vụ sở hữu thời gian xử lý riêng biệt, trong khi nguồn lực bị giới hạn, đồng thời phải tuân thủ các ràng buộc nghiêm ngặt như tránh trùng lặp thời gian, đảm bảo hoàn thành đúng hạn định, hoặc ưu tiên các nhiệm vụ quan trọng hơn. Khi số lượng công việc và tài nguyên tăng lên, các phương pháp lập lịch thủ công hoặc dựa trên các thuật toán đơn giản nhanh chóng bộc lộ hạn chế, dẫn đến tình trạng chậm trễ trong tiến độ và lãng phí nguồn lực một cách đáng kể.

Bài toán lập lịch được xem là một trong những vấn đề tối ưu hóa tổ hợp kinh điển và phổ biến trong thực tiễn. Một ví dụ minh họa điển hình là bài toán lập lịch cho năm công việc cần thực hiện trên hai máy xử lý song song. Mỗi công việc mang thời gian xử lý khác nhau, và mỗi máy chỉ có khả năng xử lý một công việc tại một thời điểm. Mục tiêu cốt lõi là sắp xếp thứ tự thực hiện các công việc sao cho tổng thời gian hoàn thành toàn bộ, hay còn gọi là *makespan*, đạt giá trị nhỏ nhất có thể.

Không gian nghiệm của bài toán này tăng theo cấp số nhân, cụ thể tương ứng với giai thừa của số lượng công việc, khiến các phương pháp giải quyết truyền thống như phương pháp vét cạn (*brute force*) hoặc quy hoạch động trở nên không khả thi về mặt tính toán khi quy mô bài toán tăng lên. Trong bối cảnh đó, giải thuật di truyền (*Genetic Algorithm*) nổi lên như một công cụ hiệu quả để tìm kiếm nghiệm gần tối ưu, đặc biệt trong các bài toán lập lịch phức tạp.

Giải thuật di truyền hoạt động dựa nguyên tắc chọn lọc tự nhiên theo thuyết tiến hóa của Darwin, mô phỏng quá trình sinh sản, biến đổi và chọn lọc trong thế giới sinh học. Các nghiệm ban đầu, hay còn gọi là các cá thể trong quần thể, được khởi tạo một cách ngẫu nhiên. Sau đó, chúng trải qua các giai đoạn như lai ghép (*crossover*), đột biến (*mutation*), và đánh giá độ thích nghi (*fitness*) để dần dần tiến hóa hướng tới các nghiệm chất lượng cao hơn. Trong ngữ cảnh bài toán lập lịch, mỗi nghiệm thường được biểu diễn dưới dạng một chuỗi thứ tự các công việc, ví dụ: chuỗi (3,1,2,5,4) thể hiện thứ tự xử lý các công việc lần lượt là công việc 3, công việc 1, công việc 2, công việc 5, và công việc 4. Chuỗi này thể hiện thứ tự ưu tiên để phân bổ công việc vào các máy, với mỗi công việc được gán cho máy có thời gian hoàn thành sớm nhất tại thời điểm đó, theo quy tắc lập lịch theo danh sách. Độ thích nghi của mỗi nghiệm được đánh giá dựa trên giá trị *makespan*, với mục tiêu là giảm thiểu chỉ số này.

Chẳng hạn, xét một nhà máy với năm công việc có thời gian thực hiện tương ứng là 4, 3, 2, 6, và 5 đơn vị thời gian, được thực hiện trên hai máy hoạt động song song. Giải thuật di truyền có thể khám phá không gian nghiệm và tiến hoá để tìm ra một lịch trình tối ưu bằng cách phân

7.2. KHÁI QUÁT VỀ DI TRUYỀN VÀ THUYẾT TIẾN HÓA CỦA DARWIN

bổ công việc một cách cân bằng giữa hai máy, đảm bảo rằng thời gian hoàn thành của cả hai máy xấp xỉ nhau, từ đó giảm thiểu thời gian chờ đợi không cần thiết và nâng cao hiệu suất sử dụng thiết bị. Lý do việc phân bổ công việc cho hai máy đạt được tính tối ưu trong giải thuật di truyền nằm ở cơ chế mã hóa và giải mã nghiệm: chuỗi thứ tự đại diện cho thứ tự ưu tiên phân bổ công việc, và trong quá trình lập lịch, mỗi công việc được gán cho máy có thời gian hoàn thành sớm nhất tại thời điểm đó. Thông qua các thế hệ tiến hóa, thuật toán loại bỏ dần các nghiệm kém (với *makespan* lớn do tải không cân bằng) và ưu tiên các nghiệm tốt hơn, nơi tải công việc được phân bổ hài hòa, dẫn đến *makespan* tối thiểu có thể đạt được. Ví dụ, một lịch trình với thứ tự công việc 3, công việc 1, công việc 2, công việc 5, và công việc 4 có thể dẫn đến việc phân bổ như sau: công việc 3 và công việc 5 cho máy 1, công việc 1, công việc 2, và công việc 4 cho máy 2, sao cho tổng thời gian hoàn thành của cả hai máy là gần bằng nhau, tránh tình trạng một máy quá tải trong khi máy kia nhàn rỗi.

Nhờ vào khả năng xử lý các bài toán quy mô lớn với vô số ràng buộc phức tạp, giải thuật di truyền đã được ứng dụng rộng rãi trong lĩnh vực tối ưu hóa lập lịch, sắp xếp ca làm việc, lập lịch thi cử, cũng như nhiều bài toán quản lý nguồn lực khác trong môi trường doanh nghiệp.

7.2 Khái quát về di truyền và thuyết tiến hóa của Darwin

Di truyền và thuyết tiến hóa đại diện cho những trụ cột lõi của sinh học hiện đại, cung cấp khung lý thuyết vững chắc để giải thích sự đa dạng sinh học và khả năng thích nghi của các loài đối với môi trường thay đổi. Di truyền tập trung vào cơ chế truyền tải các đặc tính từ thế hệ này sang thế hệ khác thông qua các yếu tố di truyền, cụ thể là gen, trong khi thuyết tiến hóa mô tả quá trình biến đổi dần dần của các loài qua thời gian, chủ yếu thông qua cơ chế chọn lọc tự nhiên. Những khái niệm này không chỉ làm phong phú hóa hiểu biết về thế giới sinh học mà còn truyền cảm hứng cho các mô hình tính toán, chẳng hạn như giải thuật di truyền (*Genetic Algorithm*), một phương pháp tối ưu hóa mô phỏng quá trình tiến hóa sinh học để giải quyết các vấn đề phức tạp trong khoa học máy tính và kỹ thuật [83]. Bằng cách mô phỏng các quá trình như di truyền, biến dị và chọn lọc, giải thuật di truyền cho phép tìm kiếm nghiệm gần tối ưu trong không gian tìm kiếm rộng lớn, tương tự như cách mà tiến hóa tự nhiên thúc đẩy sự thích nghi của các loài qua hàng triệu năm [84].

7.2.1 Di truyền và quy luật Mendel

Khái niệm di truyền được đặt nền móng vững chắc bởi công trình tiên phong của Gregor Mendel vào năm 1865 [85], thông qua các thí nghiệm hệ thống trên cây đậu Hà Lan. Mendel đã phát hiện ra rằng các đặc tính sinh học được truyền qua các yếu tố di truyền rời rạc, ngày nay được biết đến là gen, và ông đã đề xuất hai quy luật cơ bản: quy luật phân ly (*law of segregation*) và quy luật phân ly độc lập (*law of independent assortment*). Theo quy luật phân ly, mỗi cá thể mang hai bản sao (allele) của một gen, và chúng phân ly độc lập trong quá trình hình thành giao tử, dẫn đến tỷ lệ di truyền đặc tính trội-lặn điển hình là 3:1 ở thế hệ con kế tiếp. Ví dụ, trong thí nghiệm lai chéo giữa cây đậu hạt vàng (trội) và hạt xanh (lặn), thế hệ F1 toàn bộ là hạt vàng, nhưng thế hệ F2 xuất hiện tỷ lệ 3 hạt vàng:1 hạt xanh, minh họa rõ nét quy luật này. Quy luật phân ly độc lập khẳng định rằng các gen nằm trên các nhiễm sắc thể khác nhau sẽ di truyền độc lập, không ảnh hưởng lẫn nhau, trừ trường hợp liên kết gen.

Những quy luật này không chỉ hình thành nền tảng của di truyền học Mendel mà còn mở rộng sang di truyền quần thể, nơi các mô hình toán học như phương trình Hardy-Weinberg mô tả sự cân bằng allele trong quần thể không chịu áp lực chọn lọc [86]. Trong bối cảnh tính toán, các nguyên tắc di truyền Mendel đã truyền cảm hứng cho giải thuật di truyền, nơi các nghiệm

được biểu diễn như các nhiễm sắc thể, và các toán tử như lai ghép và đột biến mô phỏng quá trình phân ly và biến đổi để tạo ra đa dạng trong quần thể nghiệm, từ đó thúc đẩy việc tìm kiếm giải pháp tối ưu [87].

7.2.2 Thuyết tiến hóa của Darwin

Trong cuốn sách *On the Origin of Species* xuất bản năm 1859 [88], Charles Darwin đã đề xuất thuyết tiến hóa qua chọn lọc tự nhiên, một lý thuyết cách mạng khẳng định rằng sự đa dạng của các loài phát sinh từ quá trình thích nghi dần dần với môi trường. Theo Darwin, các cá thể trong quần thể sinh vật tồn tại sự biến đổi (*variation*) tự nhiên, và những cá thể sở hữu đặc điểm mang lại lợi thế sinh tồn chẳng hạn như khả năng chống chịu hạn hán tốt hơn ở thực vật hoặc tốc độ chạy nhanh hơn ở động vật săn mồi - sẽ có xác suất sống sót và sinh sản cao hơn. Qua nhiều thế hệ, các đặc điểm có lợi này sẽ lan tỏa trong quần thể, dẫn đến sự thay đổi tiến hóa.

Darwin mô tả tiến hóa như một quá trình hậu duệ với sửa đổi (*descent with modification*), nơi các thế hệ sau kế thừa phần lớn đặc tính từ tổ tiên nhưng đồng thời tích lũy các biến đổi nhỏ do biến đổi ngẫu nhiên hoặc ảnh hưởng môi trường. Áp lực chọn lọc tự nhiên đóng vai trò như một "bộ lọc" loại bỏ các biến đổi bất lợi, trong khi bảo tồn và nhân rộng những biến đổi có lợi, từ đó hình thành nên các loài mới qua thời gian địa chất dài lâu. Ví dụ, sự tiến hóa của vây cá thành chân ở động vật lưỡng cư minh họa cách biến đổi dần dần đến thích nghi với môi trường mới [89].

Sự tổng hợp giữa thuyết tiến hóa của Darwin và các quy luật di truyền của Mendel đã hình thành nên thuyết tiến hóa hiện đại (*modern evolutionary synthesis*), kết hợp cơ chế di truyền phân tử với chọn lọc tự nhiên để giải thích đầy đủ hơn về sự đa dạng sinh học [90]. Lý thuyết này không chỉ ảnh hưởng sâu sắc đến sinh học mà còn cung cấp nền tảng cho các thuật toán mô phỏng như giải thuật di truyền, nơi quần thể nghiệm trải qua "chọn lọc" dựa trên độ thích nghi, mô phỏng quá trình Darwin để giải quyết các bài toán tối ưu hóa phức tạp trong kỹ thuật và khoa học dữ liệu [83].

7.3 Giải thuật di truyền

Giải thuật di truyền (*Genetic Algorithm*) là một phương pháp tìm kiếm tối ưu thuộc nhóm thuật toán tiến hóa, được phát triển dựa trên các nguyên lý của thuyết tiến hóa thông qua chọn lọc tự nhiên của Darwin và các quy luật di truyền của Mendel. Được giới thiệu lần đầu bởi John Holland vào những năm 1970, giải thuật di truyền đã chứng minh tính hiệu quả trong việc giải quyết nhiều bài toán tối ưu hóa tổ hợp phức tạp như bài toán xếp ba lô, lập lịch sản xuất, hoặc tối ưu hóa mạng lưới - những trường hợp mà các phương pháp tối ưu truyền thống như tối ưu lồi hay quy hoạch động thường gặp khó khăn do không gian tìm kiếm quá lớn hoặc tính phi tuyến mạnh của bài toán [73], [83].

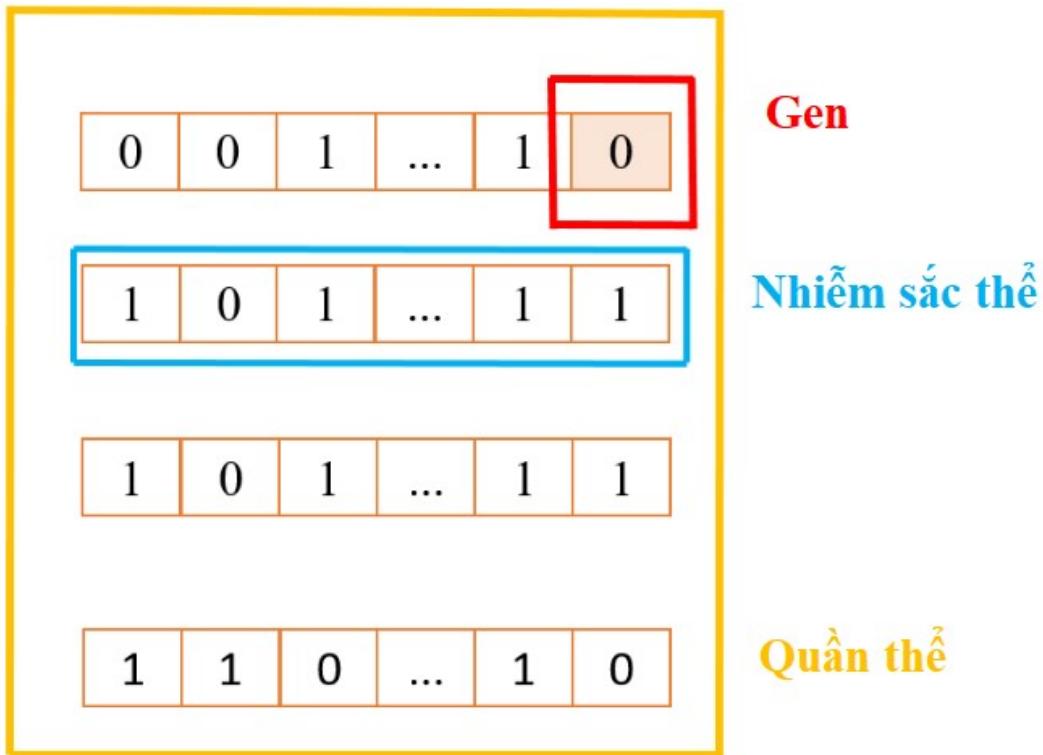
Khác với các thuật toán tìm kiếm cục bộ như thuật toán leo dồi [91], giải thuật di truyền là một kỹ thuật tìm kiếm ngẫu nhiên lấy cảm hứng từ quá trình tiến hóa sinh học, bao gồm các cơ chế như chọn lọc tự nhiên, lai ghép, và đột biến. Thuật toán này hoạt động trên một *quần thể* các cá thể - mỗi cá thể là một nghiệm tiềm năng trong không gian tìm kiếm, được mã hóa dưới dạng nhiễm sắc thể. Nhiễm sắc thể có thể là chuỗi nhị phân, vector số thực, hoặc các cấu trúc dữ liệu khác tùy theo bài toán cụ thể [84].

Quá trình tiến hóa của giải thuật di truyền mô phỏng sự thích nghi của các loài trong tự nhiên, nơi các cá thể có độ thích nghi cao hơn - tức là nghiệm tốt hơn theo hàm mục tiêu - sẽ có xác suất sinh sản và truyền lại đặc tính của mình cho thế hệ sau cao hơn. Qua nhiều thế hệ, thuật toán hướng tới việc khám phá các nghiệm tối ưu hoặc gần tối ưu thông qua sự cân bằng

7.3. GIẢI THUẬT DI TRUYỀN

giữa khai thác các nghiệm tốt hiện tại và khám phá các vùng mới trong không gian tìm kiếm. Tính chất này giúp giải thuật di truyền đặc biệt hiệu quả trong các bài toán có không gian tìm kiếm phức tạp, đa cục trị, hoặc không liên tục [87].

Để trực quan hóa cơ chế mã hóa của giải thuật di truyền, Hình 7.1 mô tả cách một cá thể (nghiệm) được biểu diễn dưới dạng nhiễm sắc thể gồm nhiều gen, và cách nhiều nhiễm sắc thể này tạo thành quần thể mà thuật toán vận hành trên đó.

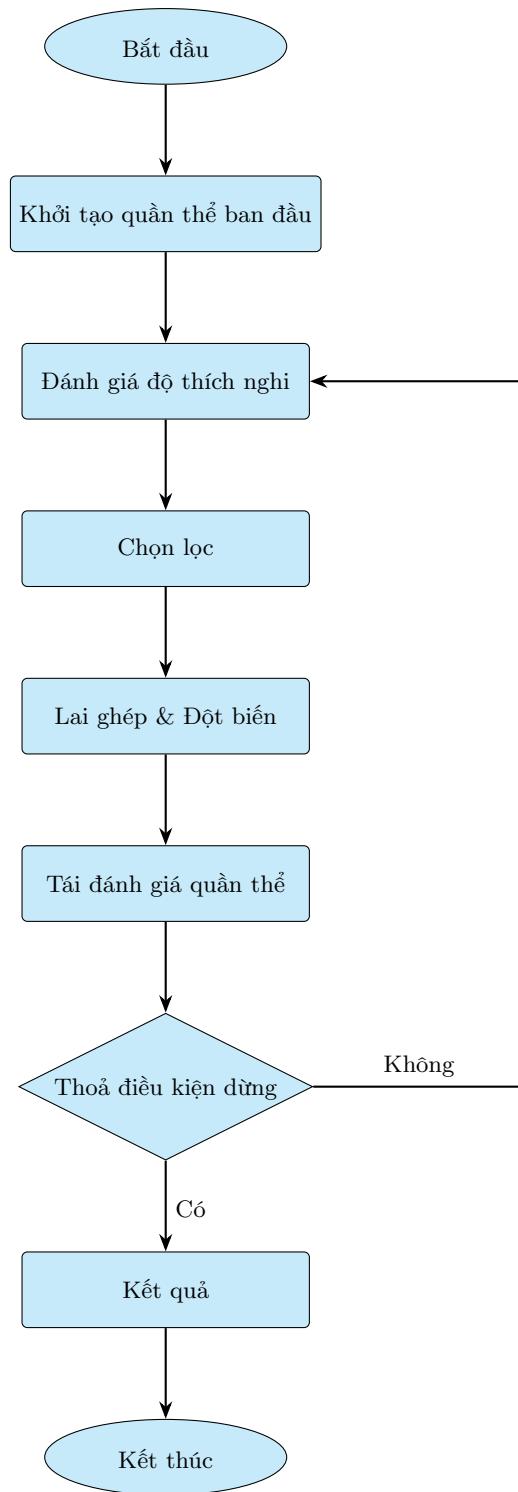


Hình 7.1: Minh họa cấu trúc của nhiễm sắc thể, *gen*, và quần thể trong giải thuật di truyền.

7.3.1 Quy trình cơ bản của giải thuật di truyền

Giải thuật di truyền mô phỏng các cơ chế tiến hóa sinh học thông qua một chuỗi các bước tuần tự, được thiết kế để cải thiện dần chất lượng của quần thể nghiệm qua các thế hệ. Hình 7.2 minh họa quy trình hoạt động của giải thuật di truyền, bao gồm các bước chính như sau:

- Khởi tạo quần thể ban đầu:** Một tập hợp các cá thể, mỗi cá thể đại diện cho một nghiệm khả thi, được tạo ra để khởi đầu thuật toán. Các cá thể này thường được khởi tạo ngẫu nhiên để đảm bảo sự đa dạng trong quần thể, nhưng trong một số trường hợp, các phương pháp heuristic hoặc tri thức miền có thể được sử dụng để tạo ra các nghiệm ban đầu có chất lượng cao hơn, giúp tăng tốc độ hội tụ [92]. Chi tiết về các chiến lược khởi tạo sẽ được trình bày tại Mục 7.3.2.
- Đánh giá độ thích nghi:** Mỗi cá thể trong quần thể được đánh giá dựa trên hàm mục tiêu của bài toán, thường được định nghĩa để đo lường chất lượng của nghiệm. Ví dụ, trong



Hình 7.2: Lưu đồ quy trình giải thuật di truyền

7.3. GIẢI THUẬT DI TRUYỀN

bài toán lập lịch, độ thích nghi có thể được tính dựa trên *makespan*, trong khi trong bài toán tối ưu hóa chi phí, nó có thể là tổng chi phí. Giai đoạn này cung cấp cơ sở để so sánh và lựa chọn các cá thể [84].

3. **Chọn lọc:** Các cá thể có độ thích nghi cao hơn được chọn với xác suất cao hơn để tham gia vào quá trình sinh sản, phản ánh nguyên lý “sống sót của kẻ thích nghi nhất” trong *thuyết tiến hóa*. Các phương pháp lựa chọn phổ biến bao gồm lựa chọn bánh xe roulette (*roulette wheel selection*), lựa chọn giải đấu (*tournament selection*), hoặc lựa chọn dựa trên xếp hạng (*rank-based selection*). Các phương pháp này đảm bảo rằng các cá thể tốt hơn có cơ hội đóng góp nhiều hơn vào thế hệ sau, nhưng vẫn duy trì một mức độ đa dạng để tránh hội tụ sớm [93].
4. **Lai ghép và đột biến:** Các cá thể được chọn trải qua quá trình lai ghép, trong đó hai cá thể cha mẹ trao đổi một phần nhiễm sắc thể để tạo ra các cá thể con, nhằm kết hợp các đặc điểm tốt từ cả hai. Ví dụ, trong lai ghép một điểm, một điểm cắt được chọn ngẫu nhiên trên nhiễm sắc thể, và các đoạn sau điểm cắt được hoán đổi giữa hai cha mẹ. Sau đó, đột biến được áp dụng với xác suất thấp để tạo ra các thay đổi ngẫu nhiên trong nhiễm sắc thể, giúp duy trì sự đa dạng và khám phá các vùng mới trong không gian tìm kiếm. Ví dụ, trong bài toán lập lịch, đột biến có thể là việc hoán đổi ngẫu nhiên vị trí của hai công việc trong chuỗi thứ tự [87].
5. **Tái đánh giá và cập nhật quần thể:** Quần thể mới được hình thành từ các cá thể con và một số cá thể cha mẹ (nếu áp dụng chiến lược thay thế không hoàn toàn). Sau đó, độ thích nghi của các cá thể trong quần thể mới được đánh giá lại để chuẩn bị cho vòng lặp tiếp theo. Các chiến lược thay thế, chẳng hạn như thay thế toàn bộ hoặc thay thế thế hệ với elitism (giữ lại một số cá thể tốt nhất), được sử dụng để cân bằng giữa sự cải thiện và đa dạng [94].
6. **Kiểm tra điều kiện dừng:** Thuật toán kết thúc khi thỏa mãn một trong các tiêu chí dừng, chẳng hạn như đạt đến số lượng thế hệ tối đa, không cải thiện độ thích nghi sau một số vòng lặp nhất định, hoặc đạt được giá trị hàm mục tiêu mong muốn. Khi đó, nghiệm tốt nhất trong quần thể được trả về như kết quả cuối cùng. Nếu chưa thỏa mãn, quy trình quay lại bước lựa chọn để tiếp tục tiến hóa [93].

7.3.2 Khởi tạo quần thể ban đầu

Khởi tạo quần thể là giai đoạn nền tảng trong giải thuật di truyền, đóng vai trò quyết định đến khả năng hội tụ, tốc độ tìm kiếm, và chất lượng nghiệm cuối cùng của thuật toán. Một quần thể khởi đầu được thiết kế tốt phải đảm bảo sự cân bằng giữa hai yếu tố then chốt: đa dạng di truyền - nhằm hỗ trợ quá trình khám phá các vùng mới trong không gian tìm kiếm - và chất lượng ban đầu của các nghiệm, giúp thúc đẩy khai thác các giải pháp tiềm năng ngay từ đầu. Nếu quần thể thiếu đa dạng, thuật toán có nguy cơ rơi vào tình trạng hội tụ sớm (*premature convergence*), dẫn đến nghiệm cục bộ thay vì toàn cục; ngược lại, nếu chất lượng ban đầu thấp, quá trình tiến hóa có thể mất nhiều thế hệ hơn để đạt được kết quả mong muốn [84].

Có hai chiến lược khởi tạo chính được sử dụng rộng rãi trong thực tiễn:

- **Khởi tạo ngẫu nhiên:** Các cá thể trong quần thể được tạo ra một cách hoàn toàn ngẫu nhiên trong toàn bộ không gian tìm kiếm khả thi. Phương pháp này đảm bảo độ bao phủ rộng lớn và duy trì tính đa dạng cao, giúp thuật toán tránh bị kẹt ở các vùng cục bộ. Tuy nhiên, do thiếu định hướng, nó có thể dẫn đến tốc độ hội tụ chậm, đặc biệt trong các bài

toán có không gian tìm kiếm lớn hoặc phức tạp. Ví dụ, trong bài toán lập lịch, các *nhiệm sắc thể* có thể được khởi tạo bằng cách sắp xếp ngẫu nhiên thứ tự công việc, nhưng điều này có thể tạo ra nhiều nghiệm không khả thi hoặc kém chất lượng [87].

- Khởi tạo theo heuristic: Phương pháp này tích hợp các thuật toán tìm kiếm tham lam, tri thức miền, hoặc các phương pháp metaheuristic khác để tạo ra một phần quần thể với chất lượng cao hơn. Chẳng hạn, trong bài toán xếp ba lô, một số cá thể có thể được khởi tạo bằng cách ưu tiên chọn các vật phẩm có tỷ lệ giá trị/khối lượng cao nhất theo thuật toán tham lam. Mặc dù cách tiếp cận này cải thiện tốc độ hội tụ và chất lượng nghiệm ban đầu, nhưng nếu lạm dụng, nó có thể làm giảm tính đa dạng, dẫn đến tình trạng hội tụ sớm và thiếu khả năng khám phá các nghiệm tối ưu toàn cục.

Các nghiên cứu thực nghiệm đã chỉ ra rằng sự kết hợp giữa hai chiến lược trên thường mang lại hiệu suất tối ưu. Ví dụ, một tỷ lệ phỏ biến là 80% cá thể được khởi tạo ngẫu nhiên để đảm bảo đa dạng, trong khi 20% còn lại sử dụng heuristic để nâng cao chất lượng ban đầu. Tỷ lệ này có thể được điều chỉnh linh hoạt tùy theo đặc trưng của bài toán, chẳng hạn như tăng tỷ lệ heuristic trong các bài toán ràng buộc chặt chẽ hoặc giảm nó trong các bài toán có không gian tìm kiếm đa cực trị [94]. Hơn nữa, trong các biến thể nâng cao của giải thuật di truyền, các kỹ thuật như khởi tạo dựa trên quần thể hạt giống hoặc sử dụng phân bố xác suất để khởi tạo có thể được áp dụng để cải thiện thêm hiệu quả [93].

Tóm lại, việc lựa chọn chiến lược khởi tạo quần thể đòi hỏi sự cân nhắc kỹ lưỡng về bản chất bài toán, kích thước quần thể, và các tham số khác của thuật toán, nhằm đạt được sự cân bằng lý tưởng giữa khám phá và khai thác, từ đó nâng cao khả năng tìm kiếm nghiệm tối ưu toàn cục.

7.3.3 Hàm mục tiêu

Hàm mục tiêu (*fitness function*) là yếu tố cốt lõi, định hướng toàn bộ quá trình tiến hóa trong giải thuật di truyền, bằng cách đánh giá và định lượng chất lượng của từng cá thể trong quần thể. Về bản chất, hàm mục tiêu gán cho mỗi cá thể một giá trị độ thích nghi, phản ánh mức độ phù hợp của nghiệm mà cá thể đó đại diện đối với mục tiêu của bài toán tối ưu. Giống như vai trò của môi trường tự nhiên trong thuyết tiến hóa, hàm mục tiêu quyết định "sự sống còn" của các cá thể: những cá thể có độ thích nghi cao sẽ có xác suất cao hơn trong các bước lựa chọn, lai ghép, và đột biến, từ đó định hình hướng tiến hóa của toàn bộ quần thể hướng tới các nghiệm tối ưu [84].

Việc thiết kế hàm mục tiêu phụ thuộc chặt chẽ vào bản chất của bài toán, và có thể được phân loại như sau:

- Trong các bài toán tối ưu hóa tổ hợp: hàm mục tiêu thường là một hàm toán học đơn giản nhưng phản ánh rõ ràng các ràng buộc và mục tiêu. Ví dụ, trong bài toán xếp ba lô (*knapsack problem*), hàm mục tiêu có thể được định nghĩa như sau:

$$\max \sum_{i=1}^n v_i x_i \quad \text{với ràng buộc} \quad \sum_{i=1}^n w_i x_i \leq W,$$

trong đó $x_i \in \{0, 1\}$ đại diện cho việc chọn hoặc không chọn mặt hàng thứ i , v_i là giá trị, và w_i là khối lượng. Trong giải thuật di truyền, nếu nghiệm vi phạm ràng buộc, độ thích nghi có thể được điều chỉnh bằng cách trừ đi một giá trị phạt tỷ lệ với mức độ vi phạm [87].

- Trong các bài toán tối ưu liên tục hoặc không ràng buộc: hàm mục tiêu có thể là các hàm benchmark tiêu chuẩn như hàm Rastrigin (với nhiều cực trị cục bộ), hàm Rosenbrock (hình

7.3. GIẢI THUẬT DI TRUYỀN

dạng thung lũng hẹp), hoặc hàm Sphere (hình dạng đơn giản để kiểm tra hội tụ). Ví dụ, hàm Sphere được định nghĩa là:

$$f(\mathbf{x}) = \sum_{i=1}^d x_i^2,$$

với cực tiểu toàn cục tại $\mathbf{x} = \mathbf{0}$. Những hàm này giúp đánh giá khả năng xử lý không gian tìm kiếm đa chiều và đa cực trị của thuật toán [95].

Trong các bài toán ràng buộc, vì giải thuật di truyền thường xử lý các nghiêm không ràng buộc, hàm mục tiêu cần được hiệu chỉnh bằng các kỹ thuật như hàm phạt (*penalty function*) - thêm chi phí phạt cho các nghiêm vi phạm ràng buộc - hoặc chuyển đổi thành bài toán đa mục tiêu, nơi một mục tiêu tập trung vào vi phạm ràng buộc và mục tiêu khác vào tối ưu hóa chính. Các phương pháp như phạt tĩnh (*static penalty*) hoặc phạt động (*dynamic penalty*) có thể được sử dụng để điều chỉnh mức phạt theo tiến trình tiến hóa [96].

Tóm lại, một hàm mục tiêu hiệu quả phải đáp ứng các tiêu chí: (1) phân biệt rõ ràng giữa nghiêm tốt và nghiêm kém, (2) phản ánh chính xác bản chất và ràng buộc của bài toán, và (3) hỗ trợ cơ chế tiến hóa hướng tới cực trị toàn cục mà không gây ra thiên kiến không mong muốn. Việc thiết kế cẩn thận hàm mục tiêu không chỉ nâng cao hiệu suất của giải thuật di truyền mà còn đảm bảo tính khả thi và độ tin cậy của các nghiêm thu được [93].

7.3.4 Các toán tử di truyền

Trong giải thuật di truyền, quá trình tiến hóa giả lập được điều khiển thông qua ba nhóm toán tử chính: toán tử chọn lọc, toán tử lai ghép, toán tử đột biến. Mỗi toán tử đóng vai trò thiết yếu trong việc mô phỏng các cơ chế sinh học của tiến hóa tự nhiên, đồng thời định hình hành vi tìm kiếm và khả năng hội tụ của thuật toán.

7.3.4.1 Toán tử chọn lọc

Toán tử chọn lọc (*Selection Operator*) là cơ chế mô phỏng quá trình chọn lọc tự nhiên trong sinh học, đảm bảo rằng các cá thể có độ thích nghi cao hơn có xác suất được giữ lại và tái sử dụng trong thế hệ kế tiếp lớn hơn. Đây là giai đoạn định hướng cho quá trình tìm kiếm, giúp thuật toán tập trung vào các vùng không gian nghiêm có triển vọng.

Có hai kỹ thuật chọn lọc phổ biến:

1. Phép tái sinh (*reproduction*): Là phương pháp xác suất, trong đó mỗi cá thể được chọn với xác suất tỉ lệ thuận với độ thích nghi của nó. Với quần thể gồm n cá thể, gọi f_i là độ thích nghi của cá thể thứ i , tổng độ thích nghi $F = \sum_{i=1}^n f_i$, và tích lũy $f_{ti} = \sum_{j=1}^i f_j$. Một số ngẫu nhiên $r \in [0, F]$ được sinh ra, và cá thể k sao cho $f_{tk-1} < r \leq f_{tk}$ được chọn. Kỹ thuật này còn gọi là *lựa chọn bánh xe quay* (*roulette wheel selection*) [84].
2. Phép chọn cắt bỏ (*truncation selection*): Là phương pháp chọn lọc quyết định (*deterministic*), trong đó quần thể được sắp xếp theo độ thích nghi giảm dần, và chỉ giữ lại n' cá thể tốt nhất (với $n' < n$) để đưa vào bước lai ghép. Phương pháp này giúp tăng tốc độ hội tụ, nhưng có nguy cơ giảm tính đa dạng di truyền nếu sử dụng quá sớm trong tiến trình.

7.3.4.2 Toán tử lai ghép

Toán tử lai ghép (*Crossover Operator*) mô phỏng quá trình trao đổi thông tin di truyền giữa hai cá thể trong sinh học, nhằm tạo ra thế hệ con mang các đặc điểm pha trộn từ cha mẹ. Đây là

toán tử chủ đạo trong việc khai thác thông tin đã học được từ quần thể, và đóng vai trò trung tâm trong việc khám phá không gian nghiệm.

Một số kỹ thuật lai ghép phổ biến bao gồm:

- Lai ghép một điểm (*one-point crossover*): Chọn ngẫu nhiên một điểm cắt trên chuỗi nhiễm sắc thể, sau đó hoán đổi các đoạn bên phải (hoặc bên trái) giữa hai cá thể cha mẹ. Ví dụ: với hai nhiễm sắc thể $[1, 0, 1]$ và $[0, 1, 0]$, nếu điểm cắt là sau gene thứ hai, ta thu được hai con mới là $[1, 0, 0]$ và $[0, 1, 1]$ [83].
- Lai ghép hai điểm (*two-point crossover*) và lai ghép đồng nhất (*uniform crossover*) cũng được sử dụng để tăng cường mức độ tái tổ hợp giữa các cặp gene, từ đó nâng cao khả năng tìm kiếm nghiệm tối ưu.

7.3.4.3 Toán tử đột biến

Toán tử đột biến (*Mutation Operator*) mô phỏng các biến đổi ngẫu nhiên trong cấu trúc di truyền, tương ứng với đột biến gene trong sinh học. Trong giải thuật, đột biến được áp dụng bằng cách thay đổi ngẫu nhiên một hoặc nhiều giá trị trong nhiễm sắc thể của cá thể, với xác suất rất nhỏ. Mục tiêu của đột biến là bảo toàn và duy trì đa dạng di truyền trong quần thể, từ đó tránh hiện tượng *hội tụ sớm* (tức rơi vào cực trị cục bộ).

Ví dụ 7.1. Ví dụ: với một nhiễm sắc thể nhị phân $[1, 0, 1]$, nếu đột biến xảy ra tại vị trí thứ hai, giá trị 0 được thay bằng 1, tạo thành nhiễm sắc thể mới $[1, 1, 1]$. Mặc dù hiệu ứng của từng lần đột biến là nhỏ, nhưng vai trò tích lũy của nó trong việc mở rộng không gian tìm kiếm là rất đáng kể [84].

Tổng thể, ba toán tử trên hoạt động phối hợp để cân bằng giữa hai mục tiêu đối lập: khai thác và khám phá trong quá trình tối ưu hóa. Trong khi chọn lọc và lai ghép hướng đến việc khai thác các thông tin đã tích lũy từ quần thể hiện tại, đột biến đóng vai trò quan trọng trong việc khám phá các vùng mới của không gian nghiệm. Sự cân bằng hợp lý giữa ba toán tử này là yếu tố then chốt quyết định hiệu quả và độ ổn định của giải thuật di truyền.

7.4 Ứng dụng của Giải thuật di truyền

7.4.1 Bài toán One-max

7.4.1.1 Giới thiệu bài toán One-max

Bài toán One-max là một bài toán đơn giản trong tính toán tiến hoá. Với bài toán này, chúng ta cần tiến hoá các cá thể của một quần thể để tìm ra một cá thể cụ thể với độ thích nghi cao nhất. Cụ thể là, mỗi các thể được biểu diễn bằng một chuỗi các số nhị phân, việc chúng ta cần là tiến hoá các chuỗi nhị phân thành chuỗi nhị phân lớn nhất. Ví dụ như: cho tập các chuỗi ban đầu với độ dài 5, mục tiêu cần làm là tiến hoá các chuỗi để trở thành 11111.

7.4.1.2 Sử dụng giải thuật di truyền để giải bài toán One-max

Dầu tiên, chúng ta đến với bước đầu tiên, khởi tạo quần thể. Ở bước này, chúng ta sẽ tạo ra ngẫu nhiên m cá thể với số gen của mỗi cá thể là n . Hình 7.3 minh họa việc khởi tạo quần thể với $m = 5$, $n = 5$:

Bước tiếp theo, chúng ta sẽ đánh giá độ thích nghi cho từng cá thể. Cụ thể ở đây, độ thích nghi được tính bằng tổng các số nhị phân của mỗi cá thể.

7.4. ỨNG DỤNG CỦA GIẢI THUẬT DI TRUYỀN

Cá thẻ 1	<table border="1"><tr><td>1</td><td>0</td><td>1</td><td>0</td><td>1</td></tr></table>	1	0	1	0	1
1	0	1	0	1		
Cá thẻ 2	<table border="1"><tr><td>1</td><td>0</td><td>1</td><td>1</td><td>0</td></tr></table>	1	0	1	1	0
1	0	1	1	0		
Cá thẻ 3	<table border="1"><tr><td>0</td><td>0</td><td>1</td><td>0</td><td>1</td></tr></table>	0	0	1	0	1
0	0	1	0	1		
Cá thẻ 4	<table border="1"><tr><td>1</td><td>1</td><td>1</td><td>1</td><td>0</td></tr></table>	1	1	1	1	0
1	1	1	1	0		
Cá thẻ 5	<table border="1"><tr><td>1</td><td>0</td><td>0</td><td>0</td><td>0</td></tr></table>	1	0	0	0	0
1	0	0	0	0		

Hình 7.3: Khởi tạo quần thẻ

Độ thích nghi											
Cá thẻ 1	<table border="1"><tr><td>1</td><td>0</td><td>1</td><td>0</td><td>1</td></tr></table>	1	0	1	0	1		<table border="1"><tr><td>3</td></tr></table>	3		
1	0	1	0	1							
3											
Cá thẻ 2	<table border="1"><tr><td>1</td><td>0</td><td>1</td><td>1</td><td>0</td></tr></table>	1	0	1	1	0		<table border="1"><tr><td>3</td></tr></table>	3		
1	0	1	1	0							
3											
Cá thẻ 3	<table border="1"><tr><td>0</td><td>0</td><td>1</td><td>0</td><td>1</td></tr></table>	0	0	1	0	1		<table border="1"><tr><td>2</td></tr></table>	2		
0	0	1	0	1							
2											
Cá thẻ 4	<table border="1"><tr><td>1</td><td>1</td><td>1</td><td>1</td><td>0</td></tr></table>	1	1	1	1	0		<table border="1"><tr><td>4</td></tr></table>	4		
1	1	1	1	0							
4											
Cá thẻ 5	<table border="1"><tr><td>1</td><td>0</td><td>0</td><td>0</td><td>0</td></tr></table>	1	0	0	0	0		<table border="1"><tr><td>1</td></tr></table>	1		
1	0	0	0	0							
1											

Hình 7.4: Đánh giá độ thích nghi cá thẻ

CHƯƠNG 7. GIẢI THUẬT ĐI TRUYỀN

Đến với bước lựa chọn, chúng ta cần phải sắp xếp lại thứ tự của quần thể theo độ thích nghi, sau đó sử dụng phương pháp lựa chọn nhị phân. Phương pháp lựa chọn nhị phân có cơ chế hoạt động như sau: lấy ngẫu nhiên hai cá thể trong quần thể, cá thể nào có độ thích nghi tốt hơn thì được chọn. Tuy nhiên chúng ta cần.

Nói cách khác, giả sử các cá thể trong quần thể được đánh trị số (từ 0 đến 4 cho 5 cá thể). Để chọn ra được một cá thể tốt, chúng ta sẽ sinh ra hai số ngẫu nhiên i_1 và i_2 nằm trong đoạn $[0, 4]$. Sau đó, lấy hai cá thể ở vị trí i_1 và i_2 , cá thể nào có độ thích nghi cao hơn sẽ được chọn.

Index	Quần thể được sắp xếp theo độ thích nghi	Độ thích nghi			
0	1 0 0 0 0	1	3	1 0 1 1 0	3
1	0 0 1 0 1	2	4	1 1 1 1 0	4
2	1 0 1 0 1	3			
3	1 0 1 1 0	3			
4	1 1 1 1 0	4			

Hình 7.5: Ví dụ về chọn lọc

Trong hình minh họa trên, 2 cặp số ngẫu nhiên (i_1, i_2) được sinh ra là (1,3) và (2,4). Thì ta thấy rằng giữa cá thể mang trị số 1 và cá thể mang trị số 3 thì cá thể mang trị số 3 có độ thích nghi cao hơn nên được lựa chọn. Tương tự như thế với cặp cá thể (2,4). Do đó, để tạo một quần thể mới từ những cá thể tốt trong quần thể hiện tại thì cần tạo ra m cặp số ngẫu nhiên và áp dụng phương pháp lựa chọn nhị phân. Hình sau minh họa việc tạo ra một quần thể mới với 5 cặp số được tạo ra ngẫu nhiên là: (1,3), (2,4), (0,4), (1,0), (1,2).

Index	Quần thể được sắp xếp theo độ thích nghi	Độ thích nghi	Quần thể mới		Độ thích nghi
			Index cũ	Quần thể mới	
0	1 0 0 0 0	1	3	1 0 1 1 0	3
1	0 0 1 0 1	2	4	1 1 1 1 0	4
2	1 0 1 0 1	3	4	1 1 1 1 0	4
3	1 0 1 1 0	3	1	0 0 1 0 1	2
4	1 1 1 1 0	4	2	1 0 1 0 1	3

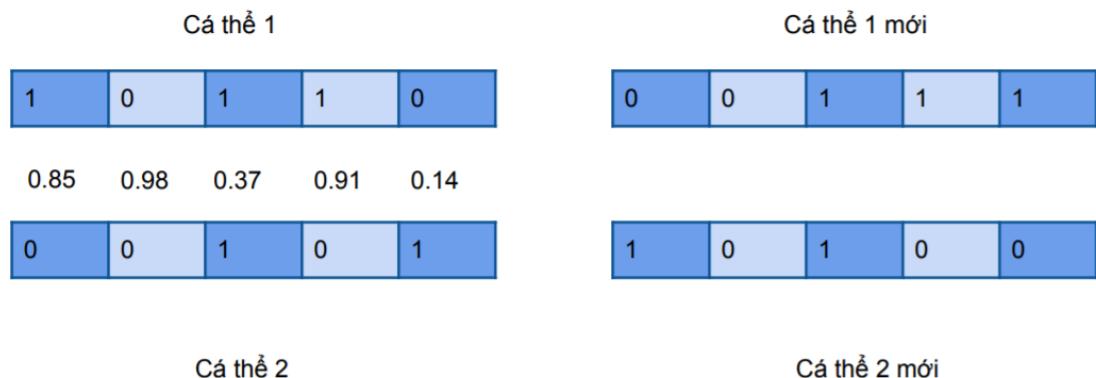
Hình 7.6: Quần thể mới từ sự chọn lọc

Chúng ta thấy được rằng ở quần thể mới, những cá thể tốt nhất ở quần thể cũ được giữ lại

7.4. ỨNG DỤNG CỦA GIẢI THUẬT DI TRUYỀN

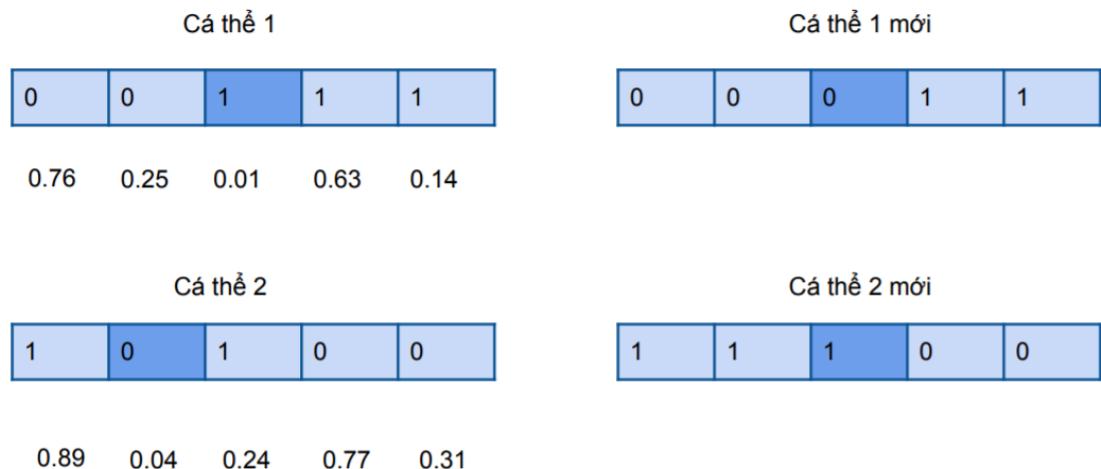
và những cá thể có độ thích nghi nhỏ hơn đã bị loại bỏ.

Tiếp theo, chúng ta đến bước lai ghép các cá thể nghĩa là các cá thể có thể trao đổi gen với nhau. Trong bài toán này, chúng ta sẽ dùng phép lai ghép nhị phân: cho trước xác suất lai ghép là 0.9, xét hai cá thể, ở mỗi vị trí gen tương ứng giữa hai cá thể, chúng ta sẽ tạo ra một số thập phân ngẫu nhiên có giá trị trong khoảng (0,1) nếu số thập phân được tạo ra có giá trị bé hơn 0.9 thì gen tại 2 vị trí này của mỗi gen sẽ được đổi chỗ cho nhau. Hình 7.7 minh họa bước lai ghép này:



Hình 7.7: Lai ghép hai cá thể

Sau khi thực hiện xong bước lai ghép, các cá thể sẽ xảy ra hiện tượng đột biến. Tương tự việc lai ghép, chúng ta cũng cần tạo ra các số thập phân ngẫu nhiên ứng với từng vị trí của gen trong mỗi cá thể. Nếu số thập phân đó bé hơn xác suất xảy ra đột biến thì gen tương ứng tại vị trí đó của cá thể sẽ xảy ra đột biến. Hình 7.8 minh họa sự đột biến này với xác suất xảy ra đột biến là 0.05:



Hình 7.8: Cá thể đột biến

Trong bài toán này, điều kiện dừng khá là đơn giản đó chính là số thế hệ xuất hiện được cho

truớc.

Source code cho bài toán One-max như sau:

```

import random
n = 10          # so luong gen
m = 20          # so luong ca the
n_generations = 20 # so the he cho truoc
fitnesses = []
def generate_random_value():
    return random.randint(0, 1)
def compute_fitness(individual):
    return sum(gen for gen in individual)
# tao 1 ca the
def create_individual():
    return [generate_random_value() for _ in range(n)]
#phep lai ghep
def crossover(individual1, individual2, crossover_rate = 0.9):
    individual1_new = individual1.copy()
    individual2_new = individual2.copy()

    for i in range(n):
        if random.random() < crossover_rate:
            individual1_new[i] = individual2[i]
            individual2_new[i] = individual1[i]

    return individual1_new, individual2_new
#phep dot bien
def mutate(individual, mutation_rate = 0.05):
    individual_m = individual.copy()

    for i in range(n):
        if random.random() < mutation_rate:
            individual_m[i] = generate_random_value()

    return individual_m
#phep chon loc
def selection(sorted_old_population):
    index1 = random.randint(0, m-1)
    while True:
        index2 = random.randint(0, m-1)
        if (index2 != index1):
            break

    individual_s = sorted_old_population[index1]
    if index2 > index1:
        individual_s = sorted_old_population[index2]

    return individual_s
#tao quan the moi sau moi buoc lap
def create_new_population(old_population, elitism=2, gen=1):
    sorted_population = sorted(old_population, key=compute_fitness)

    if gen%1 == 0:

```

7.4. ỨNG DỤNG CỦA GIẢI THUẬT DI TRUYỀN

```
fitnesses.append(compute_fitness(sorted_population[m-1]))
print("BEST:", compute_fitness(sorted_population[m-1]))

new_population = []
while len(new_population) < m-elitism:
    # buoc chon loc
    individual_s1 = selection(sorted_population)
    individual_s2 = selection(sorted_population) # duplication

    # buoc lai ghep
    individual_c1, individual_c2 = crossover(individual_s1, individual_s2)

    # buoc dot bien
    individual_m1 = mutate(individual_c1)
    individual_m2 = mutate(individual_c2)

    new_population.append(individual_m1)
    new_population.append(individual_m2)

for ind in sorted_population[m-elitism:]:
    new_population.append(ind.copy())
print(old_population)
return new_population

#khoi tao quan the
population = [create_individual() for _ in range(m)]
for i in range(n_generations):
    population = create_new_population(population, 2, i)
#ve so do
import matplotlib.pyplot as plt
plt.plot(fitnesses,'b')
plt.ylabel('Fitness')
plt.xlabel('Generation')
plt.ylim([0,11])
plt.show()
```

Giải thuật di truyền được sử dụng cho những bài toán khó, và đã được ứng dụng thành công cho một số bài toán như: lập kế hoạch, điều khiển tương thích, chương trình trò chơi, các bài toán vận tải, bài toán người đi du lịch,... Sau đây là một vài ứng dụng tiêu biểu của Giải thuật di truyền.

7.4.2 Bài toán sắp xếp thời khoá biểu

7.4.2.1 Giới thiệu bài toán sắp xếp thời khoá biểu

Chúng ta thường gặp những bài toán về sắp xếp như sắp xếp lịch làm việc, sắp xếp thời gian làm việc,... Đối với loại bài toán sắp xếp này cần phải tìm ra một nghiệm thoả mãn các ràng buộc, yêu cầu đồng thời khai thác hiệu quả tài nguyên cũng như tiết kiệm được chi phí.

Việc xếp thời khóa biểu trong trường học, đặc biệt trong Đại học là một ví dụ của bài toán nêu trên. Mỗi sinh viên trong một học kỳ đều có lịch học các môn lý thuyết và các môn thực hành. Vấn đề cần giải quyết ở đây chính là sắp xếp các môn học sao cho phù hợp, không gây xung đột, thoả mãn các ràng buộc như:

- Ràng buộc về đối tượng tham gia trong bài toán (giảng viên, sinh viên, trợ giảng, lớp học)

CHƯƠNG 7. GIẢI THUẬT DI TRUYỀN

- Ràng buộc về thời gian (thời gian bắt đầu và kết thúc của mỗi tiết học, số tiết học của một môn trong ngày, tuần)
- Ràng buộc về tài nguyên (phòng dạy lý thuyết, phòng dạy thực hành, thiết bị cơ sở vật chất dạy học,...)
- Ràng buộc về địa điểm (Các cơ sở của một trường Đại học)
- Một vài ràng buộc khác như về chuyên môn,..

Bài toán sắp xếp thời khoá biểu là một bài toán không mới và có nhiều giải thuật để giải quyết (tô màu đồ thị, xấp xỉ...) tuy nhiên các trường hợp này không mang tính tổng quát chỉ có thể dùng được cho những trường học nhỏ, ít dữ liệu cần sắp xếp. Trong vài năm gần đây, giải thuật di truyền đã thu hút nhiều sự chú ý bởi vì các đặc điểm của nó như không đòi hỏi nhiều tri thức, thích hợp với các bài toán không gian nghiệm lớn và có thể áp dụng trong nhiều trường hợp. Vì vậy, việc nghiên cứu và ứng dụng giải thuật di truyền trong việc sắp xếp thời khoá biểu cho các trường Đại học có cơ sở dữ liệu môn học lớn là một việc làm cần thiết nhằm giảm bớt các chi phí, công sức và thời gian đồng thời thoả các ràng buộc trong thực tế.

7.4.2.2 Sử dụng giải thuật di truyền để giải bài toán xếp thời khoá biểu

Một trường đại học có 4 khóa học cần được phân bổ lịch học cho các giảng viên và sinh viên. Mục tiêu là thiết kế một lịch học tối ưu dựa trên các tiêu chí sau:

- **Giảm xung đột:** Không để hai lớp trùng nhau trên cùng một khung giờ cho cả giảng viên và sinh viên.
- **Cân bằng:** Tối thiểu hóa sự chênh lệch giữa số buổi học vào ngày thường (thứ Hai đến thứ Sáu) và cuối tuần (thứ Bảy, Chủ nhật).
- **Không quá "chặt chẽ":** Tránh xếp các buổi học liên tiếp gần nhau (giữa hai khung giờ sát nhau).

Danh sách khóa học và khung giờ khả dụng

Khóa học	Slot 0	Slot 1	Slot 2	Slot 3
Advanced Software Programming (ASP)	Chiều thứ 7	Tối thứ 3	Tối thứ 2	Sáng chủ nhật
Data Engineering (DE)	Tối thứ 4	Sáng thứ 7	Sáng chủ nhật	Tối thứ 6
Big Data (BD)	Tối thứ 6	Tối thứ 2	Tối thứ 3	Tối thứ 7
natural language processing (NLP)	Sáng thứ 7	Sáng chủ nhật	Tối thứ 5	Tối thứ 6

Điểm ưu tiên của sinh viên theo khung giờ

Giải thích ý nghĩa điểm ưu tiên Tổng điểm ưu tiên theo ngày cho thấy sự phân bố mức độ ưu tiên của tất cả sinh viên cho từng ngày trong tuần. Các ngày có tổng điểm ưu tiên cao có thể cho thấy nhu cầu học cao hơn vào những ngày đó, từ đó giúp tối ưu hóa việc phân bổ lịch học, tránh trùng lịch và cân bằng lịch học trong tuần. Chúng còn là cơ sở giúp đánh giá sự cân bằng giữa lịch học ngày thường (thứ Hai đến thứ Sáu) và cuối tuần (thứ Bảy, Chủ nhật), hỗ

7.4. ỨNG DỤNG CỦA GIẢI THUẬT DI TRUYỀN

Bảng 7.1: Dữ liệu thời gian mong muốn của các sinh viên

Tên	Tối thứ 2	Tối thứ 3	Tối thứ 4	Tối thứ 5	Tối thứ 6	Sáng thứ 7	Chiều thứ 7	Tối thứ 7	Sáng chủ nhật	Chiều chủ nhật	Tối chủ nhật
Bạn A	3	1	1	1	4	3	2	2	3	4	1
Bạn B	4	4	3	1	1	4	4	1	4	1	1
Bạn C	1	4	4	4	1	1	1	1	1	1	1
Bạn D	1	4	4	2	3	1	3	4	2	1	3
Tổng	9	13	12	8	9	9	10	8	10	7	6

trợ tối ưu hóa yếu tố **cân bằng**. Việc tính toán này là một yếu tố quan trọng khi áp dụng các phương pháp tối ưu hóa như **Genetic Algorithm** để giải quyết các bài toán lập lịch học trong môi trường giáo dục.

Trong bài toán này, mỗi lịch học của học sinh sẽ được biểu diễn bằng một chuỗi các số nhị phân. Mỗi cặp số nhị phân (x_1, x_2) sẽ tương ứng với một slot học, được định nghĩa như sau:

(x_1, x_2)	Số nhị phân	Slot
(0, 0)	00	Slot 0
(0, 1)	01	Slot 1
(1, 0)	10	Slot 2
(1, 1)	11	Slot 3

Mỗi cặp (x_1, x_2) là một đại diện của một slot học trong lịch của học sinh. Việc sử dụng chuỗi số nhị phân này giúp đơn giản hóa việc mã hóa và lưu trữ thông tin lịch học. Như hình minh họa dưới đây:

	ADVANCED SOFTWARE PROGRAMMING	DATA ENGINEERING	BIG DATA	NATURAL LANGUAGE PROCESSING				
BẠN A	0	0	0	1	0	0	0	1
BẠN B	0	0	0	1	0	0	0	1
BẠN C	0	1	1	1	1	1	0	0
BẠN D	1	0	1	1	1	0	0	1

Hình 7.9: Lịch học ở dạng chuỗi số nhị phân

Danh sách lịch học tương ứng của sinh viên theo khung giờ đã được mã hóa
Công thức tính điểm Fitness

1. **Baseline:** Tổng điểm ưu tiên cơ bản cho các khung giờ đã chọn.

$$\text{Baseline} = \sum_{\text{khung giờ}} \text{Điểm ưu tiên}$$

CHƯƠNG 7. GIẢI THUẬT ĐI TRUYỀN

Bảng 7.2: Lịch học của các sinh viên

Tên	Lịch
Bạn A	Chiều thứ 7, Tối thứ 6, Tối thứ 2, Sáng chủ nhật
Bạn B	Chiều thứ 7, Sáng thứ 7, Tối thứ 6, Sáng chủ nhật
Bạn C	Tối thứ 3, Tối thứ 6, Tối thứ 7, Sáng thứ 7
Bạn D	Tối thứ 2, Tối thứ 6, Tối thứ 3, Sáng chủ nhật

2. **Conflict:** Nếu có trùng thời gian cho một sinh viên hoặc giảng viên:

$$\text{Conflict Penalty} = -\infty$$

3. **Balance:** Nếu chênh lệch giữa số buổi học vào ngày thường và cuối tuần:

$$\text{Balance Penalty} = |\text{Số buổi ngày thường} - \text{Số buổi cuối tuần}| \times (-5)$$

4. **Not too tight:** Nếu có cặp khung giờ liên tiếp:

$$\text{Not too tight Penalty} = \text{Số cặp liên tiếp} \times (-2)$$

5. **Fitness Score:** Tổng hợp các thành phần:

$$\text{Fitness} = \text{Baseline} + \text{Conflict Penalty} + \text{Balance Penalty} + \text{Not too tight Penalty}$$

Bạn A

Lịch học:

- ASP: Chiều thứ 7 (10 điểm).
- DE: Tối thứ 6 (9 điểm).
- BD: Tối thứ 2 (9 điểm).
- NLP: Sáng chủ nhật (10 điểm).

Tính điểm:

- Baseline: $10 + 9 + 9 + 10 = 38$
- Conflict: Không có xung đột (không bị trừ điểm).
- Balance:

$$\text{Penalty} = |2 - 2| \times (-5) = 0$$

- Not too tight: Không có cặp liên tiếp.

Fitness Score: $38 + 0 + 0 + 0 = 38$

Chúng ta sẽ tính tương tự cho 3 bạn còn lại là Bạn B, Bạn C, Bạn D.
Tóm tắt kết quả Fitness

7.4. ỨNG DỤNG CỦA GIẢI THUẬT DI TRUYỀN

Sinh viên	Baseline	Conflict Penalty	Balance Penalty	Not too tight Penalty	Fitness Score
Bạn A	38	0	0	0	38
Bạn B	38	0	-10	-4	24
Bạn C	39	0	0	-2	37
Bạn D	41	0	-10	-2	29

Tuy nhiên hiện tại, Bạn B và Bạn D có lịch học chưa tối ưu. Cụ thể, Bạn B học 3 buổi cuối tuần trong khi Bạn D học 3 buổi trong tuần. Điều này tạo ra sự mất cân bằng giữa các ngày học trong tuần và cuối tuần, có thể làm ảnh hưởng đến khả năng nghỉ ngơi và học tập của các bạn. Vì vậy, lai tạo lịch học giữa Bạn B và Bạn D sẽ giúp cải thiện sự cân bằng này và tối ưu hóa lịch học, giảm bớt sự xung đột giữa các môn học.

Vì thế, chúng ta sẽ tiến hành lai ghép. Dùng phép lai ghép nhị phân: cho trước xác suất lai ghép là 0.9, xét hai bạn sinh viên B và D, ở mỗi lịch học tương ứng giữa hai bạn sinh viên, chúng ta sẽ tạo ra một số thập phân ngẫu nhiên có giá trị trong khoảng (0,1) nếu số thập phân được tạo ra có giá trị bé hơn 0.9 thì lịch học tại 2 vị trí này của mỗi bạn sinh viên sẽ được đổi chỗ cho nhau.



Hình 7.10: Hình minh họa bước xác suất lai ghép lịch học giữa 2 sinh viên bạn B và bạn D

Quá trình lai tạo

Trong quá trình lai tạo, chúng ta quyết định thay đổi lịch học của môn **Advanced Software Programming (ASP)** giữa Bạn B và Bạn D:

Lịch học tương ứng của mỗi bạn sinh viên sau quá trình lai ghép

- Bạn B sẽ học **Advanced Software Programming** vào Tối thứ 2 thay vì Chiều thứ 7 như trước.
- Bạn D sẽ học **Advanced Software Programming** vào Chiều thứ 7 thay vì Tối thứ 2.

Việc hoán đổi này giúp Bạn B giảm thiểu số lượng buổi học vào cuối tuần (chỉ còn 2 buổi) và Bạn D sẽ có thêm một buổi học vào cuối tuần thay vì lịch học tập trung vào các ngày trong tuần, từ đó cân bằng lại lịch học giữa tuần và cuối tuần.

Tính lại chỉ số **Baseline** và **Fitness** cho Bạn B và Bạn D để đánh giá sự cải thiện:

- **Baseline** là điểm số phản ánh sự phù hợp giữa lịch học hiện tại và lịch học mà sinh viên mong muốn. Điểm **Baseline** cao chứng tỏ rằng lịch học đã được tối ưu hóa gần như hoàn toàn theo mong muốn của sinh viên.
- **Fitness** là điểm số tối ưu hóa lịch học, dựa trên các yếu tố như:

	ADVANCED SOFTWARE PROGRAMMING	DATA ENGINEERING	BIG DATA	NATURAL LANGUAGE PROCESSING				
BẠN A	0	0	0	1	0	0	0	1
BẠN B	1	0	0	1	0	0	0	1
BẠN C	0	1	1	1	1	1	0	0
BẠN D	0	0	1	1	1	0	0	1

Hình 7.11: Hình minh họa bước lai ghép lịch học giữa 2 sinh viên bạn B và bạn D

- Xung đột giữa các môn học: Liệu có môn học nào bị trùng lịch hoặc không thể tham gia được không?
- Cân bằng giữa các buổi học trong tuần và cuối tuần: Lịch học có hợp lý giữa các ngày trong tuần và cuối tuần hay không?
- Khoảng cách hợp lý giữa các buổi học liên tiếp: Các buổi học không nên quá gần nhau, gây mệt mỏi cho sinh viên.

Việc tính lại **Fitness** sau khi lai tạo giúp chúng ta biết được liệu lịch học mới có thực sự tối ưu hơn so với lịch học ban đầu hay không.

Kết quả sau khi lai tạo

Bảng 7.3: Bảng sau khi lai tạo

Sinh viên	Lịch học sau lai tạo	Baseline	Fitness
Bạn B	Tối thứ 2, Sáng thứ 7, Tối thứ 6, Sáng chủ nhật	38	36
Bạn D	Chiều thứ 7, Tối thứ 6, Tối thứ 3, Sáng chủ nhật	42	40

- **Bạn B** đã có sự cải thiện đáng kể về **Fitness** (từ 24 lên 36), điều này chứng tỏ lịch học của Bạn B đã được tối ưu hóa hơn và có sự cân bằng tốt hơn giữa các buổi học cuối tuần và ngày trong tuần.

- **Bạn D** tuy có sự cải thiện về **Baseline** (tăng từ 41 lên 42), nhưng điểm **Fitness** của Bạn D đã có sự cải thiện đáng kể về **Fitness** (từ 29 lên 40), có thể do sự thay đổi trong việc phân bổ thời gian học giữa các ngày trong tuần và cuối tuần. Tuy nhiên, việc cải thiện **Baseline** vẫn cho thấy rằng lịch học của Bạn D đã được cải thiện về mặt phù hợp với mong muốn cá nhân.

7.4.3 Một số bài toán khác

7.4.3.1 Bài toán người du lịch

Bài toán người du lịch (*Traveling Salesman Problem - TSP*) được mô tả như sau: Một du khách muốn thăm những thành phố anh quan tâm; mỗi thành phố thăm qua đúng một lần; rồi trở về

7.4. ỨNG DỤNG CỦA GIẢI THUẬT DI TRUYỀN

diểm khởi hành. Biết trước chi phí di chuyển giữa hai thành phố bất kỳ. Yêu cầu của bài toán là xây dựng một lộ trình thỏa các điều kiện trên với tổng chi phí nhỏ nhất.

TSP là bài toán tối ưu tổ hợp, không gian tìm kiếm là tập các hoán vị của n thành phố. Bất cứ hoán vị nào của n thành phố cũng là một nghiệm chấp nhận được. nghiệm tối ưu là một hoán vị với chi phí tối thiểu của hành trình. Không gian tìm kiếm là $n!$. Có thể giải bài toán này bằng nhiều phương pháp: phương pháp nhánh cận, phương pháp gần đúng hay những phương pháp tìm kiếm *heuristic*. Phương pháp nhánh cận đã được chứng minh đạt sự tối ưu về nghiệm, tuy nhiên phương pháp này lại mất khá nhiều thời gian khi số đỉnh của đồ thị lớn.

Trong những năm gần đây, đã xuất hiện nhiều thuật toán đạt gần đến nghiệm tối ưu của bài toán TSP: láng giềng gần nhất, đảo gần nhất, đảo xa nhất... và TSP cũng trở thành một đích ngắm của cộng đồng GAs.

Với bài toán này chúng ta sẽ đánh số các thành phố và dùng một vector nguyên để biểu diễn một NST lô trình $\mathcal{V} = \langle i_1, i_2, \dots, i_n \rangle$ biểu diễn một lô trình: từ i_1 đến i_2 , từ i_{n-1} đến i_n và trở về i_1 (\mathcal{V} là một hoán vị của vector $\langle 1, 2, \dots, n \rangle$), hàm lượng giá chính là chi phí của lô trình.

7.4.3.2 Bài toán lập lịch

Lập lịch là bài toán tổ chức sản xuất. Một công ty cần sản xuất nhiều loại hàng hóa; những hàng hóa này có thể được sản xuất theo những kế hoạch khác nhau. Mỗi kế hoạch xử lý gồm một chuỗi thao tác; những thao tác này sử dụng một số tài nguyên và cần thời gian chạy máy. Một lịch sản xuất là một kế hoạch thực hiện các đơn đặt hàng. Trong đó, một số đơn đặt hàng có thể được thực hiện với cùng những thao tác tương đương. Nhiệm vụ là lên kế hoạch, lập lịch sản xuất, để thực hiện các đơn đặt hàng nhanh nhất có thể.

Bài toán lập lịch là chọn một chuỗi các thao tác đồng thời chỉ định về thời gian bắt đầu/ kết thúc và các tài nguyên cần thiết cho mỗi thao tác. Điều cần quan tâm chính yếu là chi phí thời gian máy rõi, năng lực lao động và các đơn đặt hàng cần hoàn thành đúng hạn.Ý tưởng chính trong phương pháp là mã hóa biểu diễn của lịch phân công là các toán tử di truyền phải thực hiện theo cách có ý nghĩa, và một bộ giải mã phải luôn tạo ra một nghiệm hợp lệ cho bài toán. Thủ tục giải mã mô phỏng các thao tác của công việc theo cách mà khi một máy tính sẵn sàng chọn lựa, thì thao tác cho phép đầu tiên từ danh sách ưu tiên được lấy ra. Ví dụ nếu danh sách ưu tiên của máy m_1 là: $m_1(40\ o_3\ o_1\ o_2$ “chờ” “nhàn rõ”), thì thủ tục giải mã vào thời điểm 40 có thể thực hiện đơn đặt hàng o_3 trên máy m_1 . Nếu không được, thủ tục giải mã sẽ thực hiện đơn đặt hàng o_1 và o_2 (nghĩa là, tìm ở o_1 trước; nếu không được mới tìm ở o_2). Biểu diễn này bảo đảm tạo một lịch phân công hợp lệ.

7.4.3.3 Phân hoạch đối tượng và đồ thị

Bài toán phân hoạch là cần chia n đối tượng thành k loại. Lớp bài toán này gồm nhiều bài toán nổi tiếng như bài toán đóng thùng (gán các mặt hàng vào các thùng), bài toán tô màu đồ thị (gán các nút của đồ thị vào các màu cụ thể...).

Bài toán đóng thùng (Bin Packing - BP) được phát biểu như sau: Cho danh sách gồm n đồ vật $L = a_1, a_2, a_3, \dots, a_n$ và các thùng giống nhau với cùng sức chứa B. Kích thước của đồ vật a_i là s_i thỏa mãn $0 \leq s_{ij} \leq B \forall i = 1, 2, \dots, n$. Vấn đề đặt ra là tìm cách xếp các đồ vật vào các thùng sao cho số lượng thùng cần phải sử dụng là ít nhất.

7.4.3.4 Vạch đường cho robot di chuyển

Tìm đường là hướng dẫn robot di chuyển đến đích mà không bị lạc hay va vào những đối tượng khác. Trong bài toán này, một lộ trình được lập trước để robot đi theo, lộ trình này có thể dẫn

dắt robot di tới đích một cách hoàn hảo. Tuy nhiên, các nhà khoa học muốn cải tiến hơn bằng cách vạch lô trình nội tại, phụ thuộc vào tri thức thu được từ việc cảm nhận môi trường cục bộ để xử lý các chướng ngại chưa biết. Bộ tìm đường tiến hóa (EN) được đề xuất. Phần đầu của thuật giải là tìm lô trình toàn cục tối ưu từ điểm khởi đầu đến đích, phần thứ hai có trách nhiệm xử lý những va chạm có thể xảy ra hay những vật cản chưa được biết trước bằng cách thay một phần của lô trình toàn cục gốc bằng một lô trình con tối ưu.

7.5 Bài tập

Bài tập 7.1. Nêu ưu điểm và nhược điểm của việc sử dụng phương pháp chọn lọc Bánh xe (Roulette Wheel Selection) và phương pháp chọn lọc Giải đấu (Tournament Selection)

Bài tập 7.2. Xét một quần thể gồm nhiều cá thể, mỗi cá thể được đại diện bởi một nhiễm sắc thể có chiều dài $n = 500$. Mỗi gen trong nhiễm sắc thể có thể nhận một trong 4 giá trị sau {A, B, C, D}. Giả sử rằng số lượng cá thể là P. Từ thông tin trên, trả lời các câu hỏi sau:

a/ Có thể tạo ra được tối đa bao nhiêu loại nhiễm sắc thể.

b/ Giả sử rằng chiều dài mỗi nhiễm sắc thể và số lượng cá thể trong quần thể không thay đổi. Số lượng nhiễm sắc thể khác nhau được đánh giá độ thích nghi qua M thế hệ tối đa là bao nhiêu?

c/ Biết rằng số lượng cá thể là 10^{12} và quần thể sẽ trải qua 10^9 thế hệ. Tính p, biết p là tỉ lệ giữa số lượng cá thể được đánh giá độ thích nghi trên toàn bộ các nhiễm sắc thể. Giả sử các nhiễm sắc thể đều khác nhau.

Bài tập 7.3. Xét một quần thể gồm 5 cá thể với độ thích nghi lần lượt như sau: $f_1 = 5, f_2 = 7, f_3 = 8, f_4 = 10, f_5 = 15$. Tính xác suất mà cá thể 4 vẫn được giữ lại trong quần thể sau quá trình chọn lọc đầu tiên nếu:

a/ Sử dụng cách chọn lọc Bánh xe Roulette.

b/ Cách chọn lọc Giải đấu biết quy mô của giải đấu là 2 và xác suất để chọn được cá thể tốt nhất là 0.75.

Bài tập 7.4. Sử dụng giải thuật di truyền với một quần thể mà mỗi cá thể trong quần thể đó được đại diện bởi một nhiễm sắc thể có độ dài là 8. Mỗi gen trong nhiễm sắc thể là một số nguyên không âm bé hơn 10. Giả sử rằng độ thích nghi của từng cá thể được đánh giá bằng hàm sau: $f(x) = (a+b)-(c+d)+(e+f)-(g+h)$. Biết rằng quần thể không cố định và ban đầu, quần thể gồm 4 cá thể có nhiễm sắc thể như sau: $x_1 = 65413532, x_2 = 87126601, x_3 = 23921285, x_4 = 41852094$.

a/ Tính độ thích nghi của từng cá thể ban đầu và sắp xếp lại thứ tự của chúng dựa trên độ thích nghi (từ cao đến thấp)

b/ Thực hiện bước lai ghép như sau:

- Sử dụng phương pháp lai ghép đơn điểm hai cá thể có độ thích nghi cao nhất với điểm lai ở chính giữa nhiễm sắc thể.
- Sử dụng phương pháp lai ghép hai điểm hai cá thể có độ thích nghi cao thứ hai và ba với điểm lai đầu tiên ở ngay sau gen thứ 2 và điểm lai thứ hai ở ngay sau gen thứ 4 của nhiễm sắc thể.
- Sử dụng phương pháp lai ghép đồng nhất với hai cá thể có độ thích nghi thấp nhất.

c/ Tính lại độ thích nghi của từng cá thể mới được tạo ra và xem xét thử độ thích nghi chung của quần thể đã được cải thiện hơn so với ban đầu không?

d/ Tính độ thích nghi cao nhất mà một cá thể có thể đạt được.

7.5. BÀI TẬP

Bài tập 7.5. Giả sử chúng ta đang muốn giải bài toán sau:

$$\text{Minimize } f(x) = \sum_{i=1}^7 (x_i^2 - 10 \cos(2\pi x_i) + 10)$$

Biết rằng \mathbf{x} là một vector 7 chiều có cận dưới và cận trên của từng chiều lần lượt là:

$$[-1, -2, -3, -4, -5, -6, -7] \quad \text{và} \quad [9, 8, 7, 6, 5, 4, 3].$$

Dáp số cần làm tròn chính xác đến 3 chữ số sau dấu thập phân.

- a) Nếu chúng ta tiếp cận bài toán này bằng thuật toán di truyền (GA), cho biết một nhiễm sắc thể (chromosome) sẽ được biểu diễn bằng bao nhiêu bit và giải thích lý do.
- b) Với phương pháp chọn lọc quần thể mới đã học trên lớp, các nhiễm sắc thể có điểm fitness thấp sẽ không có cơ hội được chọn. Hãy trình bày một phương pháp chọn lọc cho phép các nhiễm sắc thể này vẫn có cơ hội được chọn, nhưng cơ hội này sẽ tỷ lệ với điểm fitness của chúng (tức là điểm fitness càng thấp thì càng ít cơ hội được chọn).

Bài tập 7.6. Hãy xem xét một trường hợp của thuật toán di truyền (GA) trong giai đoạn chọn lọc. Bảng 7.4 và Bảng 7.5 trình bày các quần thể của các thế hệ cha mẹ và con cái tương ứng, cùng với giá trị fitness của chúng.

No	Code	Fitness
1	G1	80
2	G2	93
3	G3	81
4	G4	100

Bảng 7.4: Thế hệ cha

No	Code	Fitness
1	G5	97
2	G6	102
3	G7	104
4	G8	78

Bảng 7.5: Bảng dữ liệu Code và Fitness cho thế hệ con

- a) Nếu chúng ta áp dụng chiến lược xếp hạng, tức là các nhiễm sắc thể có giá trị fitness cao hơn sẽ được giữ lại, quần thể tiếp theo sẽ là gì?
- b) Nếu chúng ta sử dụng chiến lược như đã nêu trong Bài 6a, nhiễm sắc thể G8 không thể sống sót do fitness thấp của nó. Hãy đề xuất một chiến lược cho phép tất cả các nhiễm sắc thể có cơ hội sống sót, tùy theo giá trị fitness của chúng.
- c) Sau khi áp dụng chiến lược mà bạn đề xuất, hãy chỉ ra thế hệ tiếp theo, kèm theo giải thích. Nếu bạn cần sử dụng xác suất, chúng ta giả sử rằng có một hàm `randbetween()` đã được triển khai, hàm này trả về các giá trị ngẫu nhiên theo thứ tự lần lượt như trong Bảng 7.6.

CHƯƠNG 7. GIẢI THUẬT DI TRUYỀN

Bảng 7.6 - Kết quả trả về từ `randbetween()` khi được gọi lần lượt (từ trái sang phải, từ trên xuống dưới)

Bảng 7.6: Bảng kết quả với các số ngẫu nhiên

62	45	91	11	80	89	60	54	90	46
21	79	13	90	78	64	90	82	44	77
31	30	16	14	37	16	30	83	64	47
30	80	45	67	11	41	4	60	41	88
99	18	62	51	59	62	30	41	18	53
9	57	7	31	55	53	80	76	31	19
93	86	96	43	70	41	27	31	36	64
2	65	8	2	75	15	56	23	34	34
24	87	97	35	11	5	46	76	51	48
76	29	8	56	19	38	50	90	20	2
60	24	90	3	47	79	69	69	10	14
91	23	1	1	91	15	70	99	4	28
90	45	13	45	66	17	24	34	34	73
73	36	78	12	20	92	62	35	100	72

Phần 2

Ứng dụng minh họa

Chương 8

Hệ thống Quản lý Phản hồi Khách hàng Thông minh

8.1 Đề bài

Một tổ chức muốn xây dựng hệ thống quản lý khảo sát thông minh (SCFM - Smart Customer Feedback Management) để thu thập các phản hồi của thực khách khi đến nhà hàng. Từ những kết quả khảo sát thu được, họ muốn tìm hiểu được các yếu tố có thể học được từ các kết quả khảo sát trên sử dụng các phương pháp học máy khác nhau như cây quyết định, mạng Bayes... để suy diễn các kết quả mong muốn.

Các yêu cầu chức năng chính của hệ thống

- *Tạo và thu thập các khảo sát:* Hệ thống có thể tạo/chỉnh sửa các form khảo sát; Gửi các form này tới người tiêu dùng và hiển thị tổng hợp các kết quả thu thập được.
- *Hệ thống phân tích và suy luận các kết quả khảo sát:* Hệ thống có thể áp dụng các mô hình học máy để phân tích các kết quả của khảo sát. Người dùng có thể tạo và quản lý các dự án tổng hợp kết quả từ nhiều khảo sát thị trường khác nhau; đưa ra các tiêu chí về việc ánh xạ, xử lý dữ liệu và quản lý quá trình huấn luyện dữ liệu.
- *Hệ thống quản lý người dùng:* Các hệ thống để quản lý người dùng cơ bản

8.2 Tài liệu phân tích yêu cầu

Phần này sẽ trình bày theo mẫu một tài liệu phân tích yêu cầu (*Software Requirements Specification*) cho hệ thống SCFM.

8.2.1 Giới thiệu

8.2.1.1 Mục đích

Tài liệu này cung cấp một mô tả chi tiết về các yêu cầu chức năng và phi chức năng của hệ thống “*Smart Customer Feedback Management*” (SCFM - Hệ thống Quản lý Phản hồi Khách hàng Thông minh). Mục tiêu chính của hệ thống là hỗ trợ quản lý và tạo ra các công cụ thu nhận phản hồi từ khách hàng, đồng thời cung cấp khả năng phân tích thông qua các mô hình học máy để trích xuất các ý kiến giá trị từ phản hồi của khách hàng.

8.2. TÀI LIỆU PHÂN TÍCH YÊU CẦU

8.2.1.2 Phạm vi

Hệ thống SCFM được thiết kế để phục vụ các tổ chức và doanh nghiệp có nhu cầu:

- Tạo và quản lý các biểu mẫu hoặc công cụ thu thập phản hồi khách hàng.
- Lưu trữ, quản lý và phân tích dữ liệu phản hồi khách hàng.
- Áp dụng các mô hình học máy để phân loại, trích xuất thông tin quan trọng và phân tích cảm xúc từ phản hồi.
- Cung cấp báo cáo và thống kê giúp nâng cao chất lượng dịch vụ và sản phẩm.

Hệ thống sẽ bao gồm các thành phần:

- Giao diện người dùng web cho quản trị viên và người dùng tham gia tạo khảo sát và tham gia khảo sát.
- Giao diện người dùng web cho data-analyst để quản lý, tiến hành phân tích và khảo sát của khách hàng.

8.2.1.3 Định nghĩa và viết tắt

Thuật ngữ	Định nghĩa
SCFM	Smart Customer Feedback Management - Hệ thống Quản lý Phản hồi Khách hàng Thông minh
GDPR	General Data Protection Regulation - Quy định bảo vệ dữ liệu chung
ML	Machine Learning - Học máy

Bảng 8.1: Định nghĩa các thuật ngữ phổ biến

8.2.2 Mô tả tổng quan

Chương này cung cấp mô tả tổng quan về hệ thống *Smart Customer Feedback Management*, bao gồm các chức năng chính, đối tượng sử dụng và các ràng buộc kỹ thuật; giúp định hình bức tranh toàn cảnh của hệ thống trước khi đi vào chi tiết các yêu cầu.

8.2.2.1 Phân quyền người dùng

Người dùng trong hệ thống được chia ra theo các vai trò như sau:

- **End User:** Người dùng tham gia khảo sát, cung cấp phản hồi và ý kiến.
- **Campaign Creator:** Người dùng tạo các bảng khảo sát, quản lý nội dung khảo sát và thu thập dữ liệu từ End User.
- **Data Scientist:** Người dùng quản lý các dự án học máy hoặc trích xuất dữ liệu; dữ liệu này được lấy từ các bảng khảo sát.
- **Admin:** Quản trị viên hệ thống, chịu trách nhiệm quản lý người dùng và đảm bảo vận hành ổn định của hệ thống.

8.2.2.2 Lược đồ Use-case

Lược đồ 8.1 biểu diễn tổng quan use case của hệ thống.

Các actor chính của hệ thống tương ứng với các phân quyền người dùng được mô tả ở mục 8.2.2.1

Các usecase của hệ thống được chia thành 3 nhóm chính: Nhóm quản lý các khảo sát, nhóm quản lý các dự án học máy và nhóm quản lý người dùng.

8.2.2.3 Yêu cầu chức năng của hệ thống

Các yêu cầu chức năng của hệ thống bao gồm

(a.) Customer Feedback Management (CFM)

- *Survey Creation:* Tạo các biểu mẫu khảo sát tùy chỉnh với các loại câu hỏi: checkbox, text input, rating scale; Thiết lập các logic điều kiện trong form (conditional logic) cho business flow (nhảy câu hỏi).
- *Survey Deployment:* Cung cấp link hoặc QR code để khách hàng truy cập form; Quản lý trạng thái khảo sát: active, inactive; Tích hợp gửi form qua thư điện tử
- *Survey Result Summary:* Hiển thị bảng tổng hợp các câu trả lời; Tự động vẽ các biểu đồ (biểu đồ cột, tròn, đường...) dựa trên dữ liệu khảo sát.

(b.) Data Analysis System

- *Feature Engineering:* Ánh xạ từ dữ liệu thô thành các feature cho mô hình; Cung cấp UI để thêm các rule chuẩn hóa dữ liệu như: Loại bỏ giá trị null/empty, Scale dữ liệu về khoảng 0-1 hoặc chuẩn hóa các thang điểm (1-5, 1-10), Mapping dữ liệu dạng phân loại (e.g., Tốt = 3, Trung bình = 2, Kém = 1).
- *Model Training:* Huấn luyện các mô hình; Hiển thị thông tin kết quả training, đánh giá
- *Model Prediction:* Tích hợp hệ thống để áp dụng mô hình đã huấn luyện cho dữ liệu mới; Hiển thị kết quả dự đoán và giải thích các quyết định của mô hình (e.g., tại sao điểm này là Hài lòng).

(c.) CRM Features

- *Account Management:* Tạo tài khoản khách hàng và phân quyền (admin, data scientist, feedback manager); Quản lý danh sách khách hàng và thông tin liên hệ.
- *Permission Management:* Phân quyền rõ ràng cho các nhóm người dùng;
- *Customer Summary Dashboard:* Tóm tắt thông tin khách hàng, số lần phản hồi khảo sát, đánh giá trung bình; Lịch sử feedback của từng khách hàng.

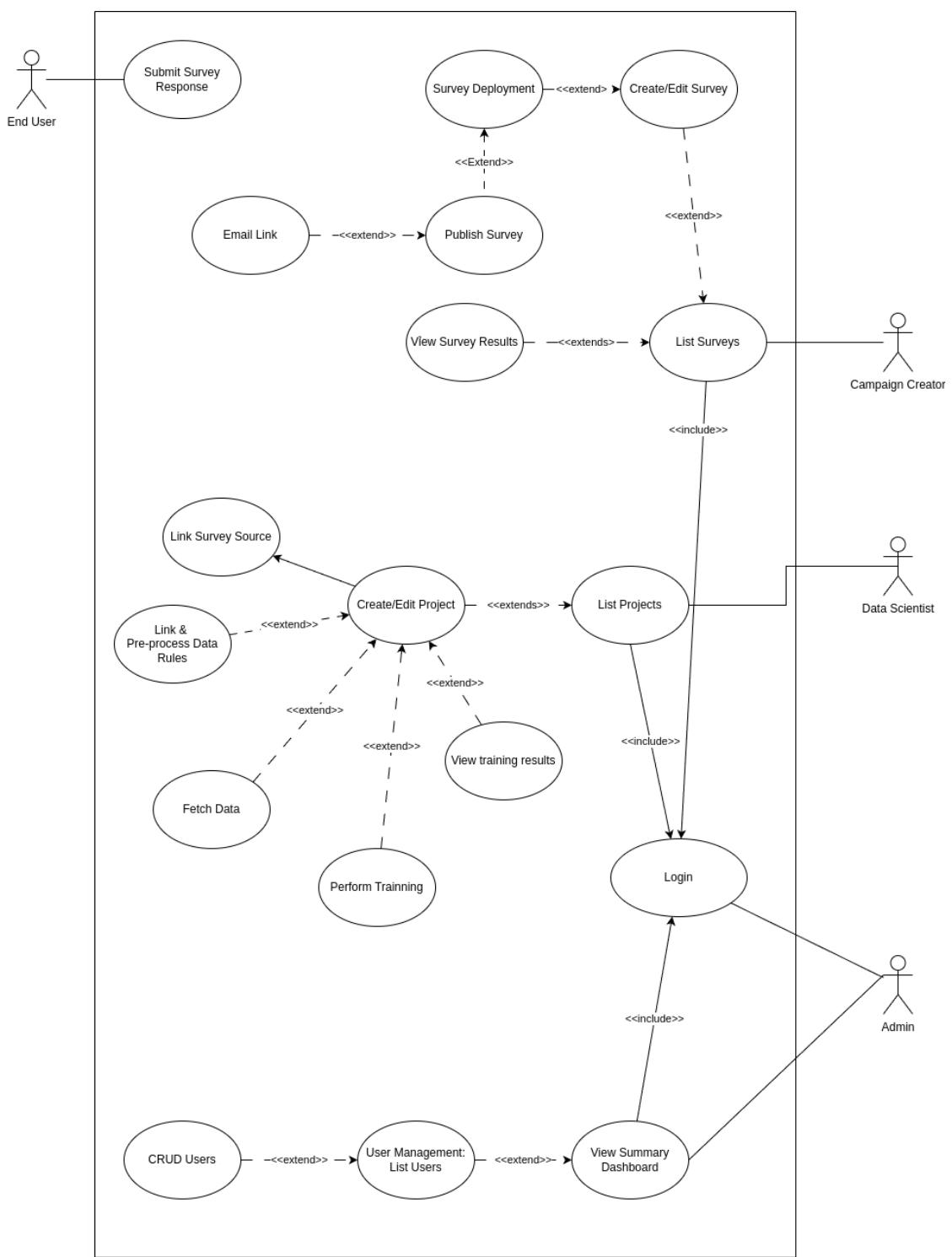
8.2.2.4 Yêu cầu phi chức năng của hệ thống

Các yêu cầu phi chức năng và ràng buộc của hệ thống bao gồm:

- Về hiệu năng

- Hệ thống có thể xử lý đồng thời ít nhất 100 khách hàng cùng lúc gửi phản hồi khảo sát.

8.2. TÀI LIỆU PHÂN TÍCH YÊU CẦU



Hình 8.1: Lược đồ use case tổng quát

- Thời gian xử lý dữ liệu và trả kết quả phân tích không quá 5 giây cho tập dữ liệu dưới 1 triệu dòng.

- Về khả năng mở rộng

- Hỗ trợ mở rộng để quản lý số lượng lớn khảo sát (> 100 form) và dữ liệu (> 10 triệu bản ghi).

- Về khả năng sử dụng

- Giao diện người dùng trực quan, hỗ trợ đa ngôn ngữ: Anh - Việt. Một người dùng sau khi đọc tài liệu hướng dẫn có thể sử dụng hệ thống ở mức cơ bản sau 1h.
- Có hướng dẫn sử dụng chi tiết cho từng chức năng.

- Về độ tin cậy

- Đảm bảo hệ thống không bị downtime quá 0.1% thời gian hoạt động.
- Lưu log cho mọi hoạt động trên hệ thống để có thể kiểm tra trong trường hợp lỗi. Thời gian lưu log là 7 ngày.

- Về bảo mật

- Mã hóa dữ liệu khách hàng và thông tin khảo sát trong khi lưu trữ và truyền tải (SSL, AES).
- Hỗ trợ xác thực hai yếu tố (2FA) cho các tài khoản admin.
- Hệ thống phân quyền nghiêm ngặt, đảm bảo chỉ người có quyền mới truy cập được dữ liệu nhạy cảm.

- Về tuân thủ các điều luật

- Tuân thủ các quy định bảo vệ dữ liệu cá nhân như GDPR.

- Về bảo trì

- Hệ thống được xây dựng trên các công nghệ phổ biến, dễ bảo trì và nâng cấp.
- Ghi chú rõ ràng trong mã nguồn và tài liệu triển khai.

- Về sao lưu và sự cố

- Sao lưu dữ liệu hàng ngày.
- Bản sao lưu được lưu trữ trong vòng 30 ngày.
- Hỗ trợ khôi phục dữ liệu trong vòng 24 giờ trong trường hợp sự cố.

8.2.2.5 Kịch bản Use Case

Phần này mô tả các kịch bản use case của hệ thống. Trong đó, các kịch bản được chia thành các nhóm chính:

- Kịch bản về quản lý khảo sát: bảng 8.2, 8.3, 8.4
- Kịch bản về quản lý dự án học máy: bảng 8.5, 8.6, 8.7, 8.8, 8.9, 8.10, 8.11
- Kịch bản về quản lý người dùng: bảng 8.12, 8.13, 8.14

8.2. TÀI LIỆU PHÂN TÍCH YÊU CẦU

Bảng 8.2: Kịch bản Thu thập khảo sát

ID	UC-SURVEY-SUBMIT-RESPONSE
Kịch bản	Thu thập khảo sát
Mô tả	Người dùng <i>End user</i> truy cập vào đường dẫn của một khảo sát được cung cấp, sau đó điền các câu trả lời theo yêu cầu trong biểu mẫu và gửi kết quả lên hệ thống.
Actors	Người dùng <i>End user</i>
Tiền điều kiện	Khảo sát đã được tạo và ở trạng thái <i>Active</i>
Hậu điều kiện	<ul style="list-style-type: none"> • Câu trả lời của người dùng được lưu trữ thành công trong hệ thống. • Người dùng có thể nhận thông báo xác nhận đã hoàn thành khảo sát.
Luồng chính	<ol style="list-style-type: none"> 1 Người dùng truy cập vào link của khảo sát. 2 Hệ thống hiển thị khảo sát cùng với các câu hỏi. 3 Người dùng điền các câu hỏi 4 Người dùng nhấn “Hoàn Thành” để nộp khảo sát. 5 Hệ thống kiểm tra các thông tin và hiển thị thông báo “Khảo sát đã được gửi thành công”.
Luồng thay thế	<p>5.1 Hệ thống kiểm tra các thông tin bắt buộc khảo sát chưa được điền đầy đủ</p> <p>5.1.1 Hệ thống hiển thị thông báo “Thông tin chưa được điền đầy đủ”. Quay về bước 3.</p>
Luồng ngoại lệ	<p>5.2 Hệ thống bị lỗi khi lưu dữ liệu.</p> <p>5.1.1 Hệ thống hiển thị thông báo “Gửi khảo sát thất bại, vui lòng thử lại.”</p> <p>5.2.2 Người dùng nhấn vào nút “Thử lại” để gửi lại khảo sát</p>

Bảng 8.3: Kịch bản Hiển thị danh sách khảo sát

ID	UC-SURVEYS-LIST
Kịch bản	Hiển thị danh sách khảo sát
Mô tả	Campaign Creator truy cập vào hệ thống để xem danh sách các khảo sát hiện có, bao gồm các khảo sát đã tạo, khảo sát đang hoạt động, hoặc đã hoàn thành.
Actors	Campaign Creator
Tiền điều kiện	Người dùng đã đăng nhập thành công
Hậu điều kiện	<ul style="list-style-type: none"> • Danh sách các khảo sát được hiển thị thành công cho người dùng. • Người dùng có thể thực hiện hành động khác như chọn khảo sát để xem chi tiết hoặc chỉnh sửa.

CHƯƠNG 8. HỆ THỐNG QUẢN LÝ PHẢN HỒI KHÁCH HÀNG THÔNG MINH

ID	UC-SURVEYS-LIST
Kịch bản	Hiển thị danh sách khảo sát
Luồng chính	<p>1 Người dùng truy cập vào trang quản lý khảo sát.</p> <p>2 Hệ thống hiển thị danh sách các khảo sát hiện có, bao gồm các thông tin cơ bản:</p> <ul style="list-style-type: none"> • ID khảo sát • Tên khảo sát • Ngày tạo • Trạng thái: New/Active/Paused/Completed. • Ngày cập nhật cuối • Số lượng response • Link Open Detail <p>Các khảo sát sắp xếp theo trật tự giảm dần về ngày cập nhật cuối.</p> <p>3 Người dùng nhấn vào 1 liên kết mở ra thông tin chi tiết khảo sát, hệ thống sẽ chuyển sang kịch bản chứa nội dung chi tiết của khảo sát.</p>
Luồng thay thế	<p>2.1 Không có khảo sát nào trong hệ thống. Hệ thống hiển thị thông báo “Hiện chưa có khảo sát nào được tạo”.</p> <p>2.2 Người dùng điền vào ô Filter để lọc khảo sát theo tên khảo sát, hệ thống sẽ hiển thị khảo sát theo tiêu chí của người dùng.</p> <p>2.3 Người dùng có thể nhấn vào từng cột của danh sách khảo sát để sắp xếp theo trật tự alphanumeric tăng/giảm trong danh sách</p> <p>3.1 Người dùng nhấn vào link “Tạo khảo sát”, hệ thống chuyển sang trang kịch bản tạo khảo sát.</p>

Bảng 8.4: Kịch bản Tạo khảo sát

ID	UC-SURVEY-CREATE
Kịch bản	Tạo khảo sát
Mô tả	Campaign Creator tạo một khảo sát mới bằng cách định nghĩa các thông tin như tiêu đề, mô tả, danh sách câu hỏi, và các thuộc tính khác liên quan đến khảo sát.
Actors	Campaign Creator
Tiền điều kiện	Người dùng đã đăng nhập thành công
Hậu điều kiện	<ul style="list-style-type: none"> • Một khảo sát mới được tạo và lưu thành công trong hệ thống. • Khảo sát ở trạng thái ban đầu là <i>New</i> (chưa hoạt động).

8.2. TÀI LIỆU PHÂN TÍCH YÊU CẦU

ID	UC-SURVEY-CREATE
Kích bản	Tạo khảo sát
Luồng chính	<p>1. Người dùng truy cập vào trang tạo khảo sát.</p> <p>2. Người dùng nhập các thông tin cơ bản về khảo sát, bao gồm:</p> <ul style="list-style-type: none"> • Tiêu đề • Mô tả • Ngày bắt đầu (nếu có) • Ngày kết thúc (nếu có) <p>3. Người dùng chọn “Thêm câu hỏi” để tạo câu hỏi bảng khảo sát. Hệ thống hiển thị thông tin để người dùng điền nội dung câu hỏi, bao gồm:</p> <ul style="list-style-type: none"> • Loại câu hỏi: trả lời ngắn/đoạn văn/Checkbox/List box/Date-time/Number • Nội dung câu hỏi • Nội dung gợi ý (nếu có) • Tuỳ chọn câu trả lời (nếu có) • Dánh dấu là bắt buộc hoặc không bắt buộc. <p>4. Người dùng nhấn “Lưu” để tạo khảo sát</p> <p>5. Hệ thống kiểm tra tính hợp lệ của thông tin, lưu khảo sát, và hiển thị thông báo: “Khảo sát đã được tạo thành công.”</p>
Luồng thay thế	<p>6.1. Người dùng nhấn “Xem trước”, hệ thống hiển thị mở một link mới để người dùng xem trước bảng khảo sát</p> <p>6.2 Người dùng nhấn “Publish”, hệ thống sẽ</p> <ul style="list-style-type: none"> • Lưu thông tin khảo sát. • Chuyển trạng thái khảo sát thành “Active” • Lưu timestamp log • Hiển thị liên kết đã publish cho hệ thống <p>6.3. Người dùng gửi bảng khảo sát qua thư điện tử Khi trạng thái khảo sát đang là “active”, người dùng nhấn vào nút “Send via Email” đã được enable</p> <p>6.3.1. Hệ thống hiển thị pop up để người dùng nhập thông tin</p> <p>6.3.2. Người dùng điền template thư điện tử và danh sách thư điện tử vào 2 trường tương ứng, nhấn vào nút Gửi.</p> <p>6.3.3. Hệ thống báo gửi “Gửi thành công”</p>

CHƯƠNG 8. HỆ THỐNG QUẢN LÝ PHÂN HỒI KHÁCH HÀNG THÔNG MINH

ID	UC-SURVEY-CREATE
Kịch bản	Tạo khảo sát
Luồng ngoại lệ	<p>5.1. Hệ thống hiển thị thông báo lỗi: “Không thể lưu khảo sát, vui lòng thử lại”. Người dùng nhấn vào nút thử lại để lưu lại.</p> <p>6.3.3.1. Hệ thống thông báo “Gửi thất bại” nếu việc gửi không thành công. Người dùng nhấn ok để quay lại.</p>

Bảng 8.5: Kịch bản Liệt kê các dự án học máy

ID	UC-ML-PROJECTS-LIST
Kịch bản	Liệt kê các dự án học máy
Mô tả	Data Scientist xem danh sách các dự án học máy hiện có trong hệ thống.
Actors	Data Scientist
Tiền điều kiện	Người dùng đã đăng nhập thành công
Hậu điều kiện	Danh sách các dự án học máy được hiển thị.
Luồng chính	<ol style="list-style-type: none"> Người dùng truy cập trang danh sách dự án học máy. Hệ thống hiển thị các dự án học máy hiện có, và các thông tin của chung của dự án, bao gồm: <ul style="list-style-type: none"> Tên dự án Người tạo Trạng thái (Training, Paused, Finished). Thời gian tạo Số lượng khảo sát đã nhận. Người dùng nhấn vào một dự án. Hệ thống chuyển sang kịch bản thông tin chi tiết của dự án học máy
Luồng thay thế	<ol style="list-style-type: none"> Nếu không có dự án nào. Hệ thống hiển thị thông báo “No project has been created”. Người dùng có thể lọc dự án bằng tên bằng cách nhập tên dự án vào thanh tìm kiếm.
Luồng ngoại lệ	<ol style="list-style-type: none"> Hệ thống hiển thị thông báo lỗi: “Không thể thực hiện truy vấn, vui lòng thử lại”. Người dùng nhấn vào nút ok để bỏ qua thông báo lỗi.

Bảng 8.6: Kịch bản Tạo dự án học máy

ID	UC-ML-PROJECT-CREATE
Kịch bản	Tạo dự án học máy
Mô tả	Data Scientist tạo một dự án học máy mới, chọn nguồn dữ liệu, mô hình và thiết lập các tham số huấn luyện.
Actors	Data Scientist
Tiền điều kiện	Người dùng đã đăng nhập thành công

8.2. TÀI LIỆU PHÂN TÍCH YÊU CẦU

ID	UC-ML-PROJECT-CREATE
Kịch bản	Tạo dự án học máy
Hậu điều kiện	Dự án học máy mới được tạo và lưu thành công.
Luồng chính	<p>1. Người dùng nhấn vào nút Create new project.</p> <p>2. Người dùng điền các thông tin chung của dự án, bao gồm:</p> <ul style="list-style-type: none"> • Tên dự án • Mô tả dự án <p>3. Nhấn “Lưu” để tạo dự án.</p>
Luồng thay thế	<p>2.1. Người dùng có thể chọn “Liên kết dữ liệu”, hệ thống sẽ</p> <ul style="list-style-type: none"> 2.1.1. Hệ thống tự động lưu thông tin dự án hiện tại. 2.1.2. Hệ thống chuyển sang kịch bản “liên kết dữ liệu”. <p>2.2. Người dùng có thể chọn cấu hình của việc huấn luyện, bao gồm chọn các giải thuật và thông số tương ứng. Các giải thuật có thể chọn bao gồm:</p> <ul style="list-style-type: none"> • Decision tree • Random forest • Logistic Regression • Naive bayes
Luồng ngoại lệ	<p>4.1. Hệ thống hiển thị thông báo lỗi: “Không thể thực tạo dự án, vui lòng thử lại”. Người dùng nhấn vào nút “Ok” để quay lại bước 2.</p>

Bảng 8.7: Kịch bản Chi tiết dự án học máy

ID	UC-ML-PROJECT-DETAIL
Kịch bản	Chi tiết dự án học máy
Mô tả	Data Scientist xem thông tin chi tiết một dự án học máy
Actors	Data Scientist
Tiền điều kiện	<ul style="list-style-type: none"> • Người dùng đã đăng nhập thành công • Dự án học máy đã được tạo thành công
Hậu điều kiện	Dự án học máy mới được lưu thành công.

CHƯƠNG 8. HỆ THỐNG QUẢN LÝ PHẢN HỒI KHÁCH HÀNG THÔNG MINH

ID	UC-ML-PROJECT-DETAIL
Kịch bản	Chi tiết dự án học máy
Luồng chính	<p>1. Người dùng truy cập vào trang thông tin của dự án học máy</p> <p>2. Hệ thống hiển thị các thông tin chung của dự án, bao gồm:</p> <ul style="list-style-type: none"> • ID dự án • Tên dự án • Mô tả dự án • Ngày tạo • Trạng thái • Mô hình huấn luyện <p>3. Người dùng có thể chọn một trong các trường <i>Tên Dự Án</i>, <i>Mô tả dự án</i>, <i>Mô hình huấn luyện</i> để chỉnh sửa</p> <ul style="list-style-type: none"> • Người dùng chọn vào trường cần chỉnh sửa và nhập nội dung mới • Khi người dùng click ra ngoài, hệ thống sẽ tự lưu những thay đổi.
Luồng thay thế	<p>4.1. Người dùng có thể chọn Liên kết Dữ liệu, hệ thống sẽ chuyển sang kịch bản 8.8</p> <p>4.2. Người dùng có thể chọn <i>Fetch dữ liệu</i>, hệ thống sẽ chuyển sang kịch bản 8.9</p> <p>4.3. Người dùng có thể chọn <i>Huấn luyện</i>, hệ thống sẽ chuyển sang kịch bản 8.10</p> <p>4.4. Người dùng có thể chọn <i>Kết quả huấn luyện</i>, hệ thống sẽ chuyển sang kịch bản 8.11</p>
Luồng ngoại lệ	

Bảng 8.8: Kịch bản Cấu hình Liên kết nguồn dữ liệu cho dự án học máy

ID	ML-PROJECT-LINK-SOURCES
Kịch bản	Cấu hình Liên kết nguồn dữ liệu cho dự án học máy
Mô tả	Data Scientist liên kết các nguồn dữ liệu cho dự án học máy
Actors	Data Scientist
Tiền điều kiện	<ul style="list-style-type: none"> • Người dùng đã đăng nhập thành công • Dự án học máy đã được tạo
Hậu điều kiện	Liên kết dữ liệu của dự án học máy được lưu trong hệ thống.

8.2. TÀI LIỆU PHÂN TÍCH YÊU CẦU

ID	ML-PROJECT-LINK-SOURCES
Kích bản	<p>Cấu hình Liên kết nguồn dữ liệu cho dự án học máy</p> <p>Luồng chính</p> <ol style="list-style-type: none"> Người dùng truy cập vào nội dung “Liên kết dữ liệu”. Hệ thống hiển thị thông tin của dự án học máy và danh sách các liên kết dữ liệu hiện tại. Người dùng chọn “Tạo liên kết mới” Hệ thống hiển thị bảng dialog để người dùng thiết lập thông tin <ol style="list-style-type: none"> Tên Feature Nguồn khảo sát: là một text field. Khi Người dùng điền thông tin vào text field, hệ thống sẽ hiển thị gợi ý các khảo sát đang có. Người dùng có thể chọn một trong những kết quả gợi ý để xác nhận. Nguồn câu hỏi: Hệ thống sẽ tải thông tin các câu hỏi trong bảng khảo sát đã chọn ở phần 4.2. Người dùng có thể chọn 1 câu hỏi trong các câu hỏi của bảng khảo sát. Chọn phương án tiền xử lý dữ liệu: là một danh sách các phương án tiền xử lý dữ liệu mà hệ thống hỗ trợ. Các phương án và thông số đi kèm, bao gồm <ul style="list-style-type: none"> • Clean up - Lược bỏ giá trị rỗng, chuỗi khoảng trắng • Clean up - Lược bỏ giá trị ngoài biên độ. Người dùng điền thông tin biên độ vào 2 ô Min-Max value • Normalize - Chuẩn hóa giá trị về khoảng Min - Max. Người dùng điền thông tin biên độ vào 2 ô Min-Max value Người dùng nhấn “Lưu” trên dialog để lưu cấu hình liên kết. Người dùng nhấn “Lưu” để lưu cấu hình các liên kết.
Luồng thay thế	<ol style="list-style-type: none"> Nếu chưa có liên kết dữ liệu nào của dự án, hệ thống sẽ hiển thị “Chưa có cấu hình liên kết dữ liệu” Người dùng có thể chọn “Xóa” để xóa một cấu hình liên kết dữ liệu trong dự án <ol style="list-style-type: none"> Khi hệ thống không tìm thấy thông tin khảo sát phù hợp với thông tin nhập người dùng, hệ thống hiển thị danh sách rỗng. Ở các bước 4.3, 4.4, nếu người dùng nhập lại thông tin của mục 4.2-Nguồn khảo sát thì các mục 4.3-Nguồn câu hỏi, 4.4-Phương án tiền xử lý dữ liệu sẽ được trả về giá trị mặc định là rỗng. Ở các bước 4.4, nếu người dùng nhập lại thông tin của mục 4.3-Nguồn câu hỏi thì Mục 4.4-Phương án tiền xử lý dữ liệu sẽ được trả về giá trị mặc định là rỗng. Người dùng có thể quay về bước 3 để chọn thêm hoặc xóa một cấu hình liên kết dữ liệu.

CHƯƠNG 8. HỆ THỐNG QUẢN LÝ PHẢN HỒI KHÁCH HÀNG THÔNG MINH

ID	ML-PROJECT-LINK-SOURCES
Kịch bản	Cấu hình Liên kết nguồn dữ liệu cho dự án học máy
Luồng ngoại lệ	6.1 Khi hệ thống không lưu được thông tin, hệ thống sẽ hiển thị thông báo “Có lỗi khi lưu dữ liệu. Vui lòng thử lại sau!”

Bảng 8.9: Kịch bản Fetch nguồn dữ liệu cho hệ thống

ID	ML-PROJECT-RUN-FETCH-DATA
Kịch bản	Fetch nguồn dữ liệu cho hệ thống
Mô tả	Hệ thống fetch dữ liệu từ các khảo sát để chuẩn bị cho phần huấn luyện mô hình
Actors	Data Scientist
Tiền điều kiện	<ul style="list-style-type: none"> • Người dùng đã đăng nhập thành công • Dự án học máy đã được tạo • Cấu hình liên kết nguồn dữ liệu đã thiết lập
Hậu điều kiện	Dữ liệu đã fetch được lưu trong hệ thống
Luồng chính	<ol style="list-style-type: none"> 1. Người dùng truy cập vào nội dung “Fetch Data Source”. 2. Hệ thống hiển thị thông tin trạng thái của quá trình fetch dữ liệu <ul style="list-style-type: none"> • Trạng thái tổng quát của quá trình: New/Fetching/Done. • Danh sách các cấu hình liên kết và trạng thái đã fetch dữ liệu: Bắt đầu/Chưa hoàn thành/Đã Hoàn Thành 3. Người dùng chọn “Bắt đầu”. 4. Hệ thống chuyển trạng thái: <i>Fetching data</i>. Nút <i>Bắt đầu</i> chuyển thành nút “Đừng” 5. Hệ thống dựa vào thông tin đã cấu hình, tiến hành lần lượt fetch các dữ liệu từ các bảng khảo sát, thực hiện các phép quy định tiền xử lý và lưu lại. Trong quá trình biến đổi, hệ thống cập nhật trạng thái của các cấu hình liên kết tương ứng. 6. Hệ thống thông báo đã hoàn tất, cập nhật trạng thái Tổng quát của quá trình là <i>Hoàn Thành</i>. Nút “Đừng” chuyển thành “Bắt đầu”.
Luồng thay thế	<p>2.1 Nếu chưa có liên kết dữ liệu nào của dự án, hệ thống sẽ hiển thị “Chưa có cấu hình liên kết dữ liệu. Nút “Bắt đầu” chuyển sang trạng thái disable.</p> <p>5.1 Người dùng nhấn “Đừng”. Hệ thống sẽ</p> <ul style="list-style-type: none"> • Các thông tin đã fetch sẽ bị xóa bỏ. • Nút <i>Đừng</i> chuyển sang <i>Bắt đầu</i>. • Trạng thái hệ thống chuyển sang: “New”

8.2. TÀI LIỆU PHÂN TÍCH YÊU CẦU

ID	ML-PROJECT-RUN-FETCH-DATA
Kịch bản	Fetch nguồn dữ liệu cho hệ thống
Luồng ngoại lệ	6.1 Khi hệ thống không lưu được thông tin, hệ thống sẽ hiển thị thông báo “Có lỗi khi lưu dữ liệu. Vui lòng thử lại sau!”

Bảng 8.10: Kịch bản Huấn luyện mô hình

ID	ML-PROJECT-RUN-TRAIN-MODEL
Kịch bản	Huấn luyện mô hình
Mô tả	Hệ thống chạy quá trình huấn luyện mô hình
Actors	Data Scientist
Tiền điều kiện	<ul style="list-style-type: none"> • Người dùng đã đăng nhập thành công • Dự án học máy đã được tạo • Dữ liệu nguồn cho dự án học máy đã fetch xong
Hậu điều kiện	<ol style="list-style-type: none"> 1. Người dùng truy cập vào nội dung “Huấn luyện mô hình”. 2. Hệ thống hiển thị thông tin trạng thái của quá trình huấn luyện mô hình <ul style="list-style-type: none"> • Trạng thái tổng quát của quá trình: New/Training/Done. • Thông tin về phương thức huấn luyện và các hệ số (nếu có) • Thông tin về log của quá trình huấn luyện 3. Người dùng chọn “Bắt đầu”. 4. Hệ thống chuyển trạng thái: <i>Training</i>. Nút <i>Bắt đầu</i> chuyển thành nút “Đừng” 5. Hệ thống dựa vào thông tin đã cấu hình, chạy quá trình huấn luyện mô hình. Hệ thống hiển thị các log tương ứng trong quá trình huấn luyện cho người dùng. 6. Hệ thống thông báo đã hoàn tất, cập nhật trạng thái Tổng quát của quá trình là <i>Hoàn Thành</i>. Nút “Đừng” chuyển thành “Bắt đầu”.
Luồng thay thế	<ol style="list-style-type: none"> 2.1 Nếu dữ liệu nguồn chưa được fetched, hệ thống sẽ hiển thị “Chưa có cấu hình liên kết dữ liệu. Nút “Bắt đầu” chuyển sang trạng thái disable. 5.1 Người dùng nhấn “Đừng”. Hệ thống sẽ <ul style="list-style-type: none"> • Nút <i>Đừng</i> chuyển sang <i>Bắt đầu</i>. • Trạng thái hệ thống chuyển sang: “New”
Luồng ngoại lệ	

CHƯƠNG 8. HỆ THỐNG QUẢN LÝ PHÂN HỒI KHÁCH HÀNG THÔNG MINH

Bảng 8.11: Kịch bản Hiển thị kết quả huấn luyện

ID	ML-PROJECT-RUN-TRAIN-RESULTS
Kịch bản	Hiển thị kết quả huấn luyện
Mô tả	Hiển thị kết quả huấn luyện và tiến hành dự đoán kết quả với input thực
Actors	Data Scientist
Tiền điều kiện	<ul style="list-style-type: none"> • Người dùng đã đăng nhập thành công • Dự án học máy đã được tạo • Dữ liệu nguồn cho dự án học máy đã fetch xong • Quá trình huấn luyện dữ liệu đã hoàn tất
Hậu điều kiện	
Luồng chính	<ol style="list-style-type: none"> 1. Người dùng truy cập vào nội dung “kết quả huấn luyện mô hình”. 2. Hệ thống hiển thị thông tin kết quả quá trình huấn luyện mô hình và các chỉ số tương ứng
Luồng thay thế	<ol style="list-style-type: none"> 3.1 Người dùng muốn thử dự đoán dựa trên mô hình đã huấn luyện <ol style="list-style-type: none"> (a) Hệ thống hiển thị 3 bảng với 2 cột: Feature và Giá trị. (b) Người dùng điền các giá trị tương ứng với một mẫu thử (c) Người dùng nhấn “Dự đoán” (d) Hệ thống chạy mô hình dự đoán và hiển thị kết quả cho người dùng.
Luồng ngoại lệ	

Bảng 8.12: Kịch bản Đăng nhập

ID	UC-IAM-LOGIN
Kịch bản	Đăng nhập
Mô tả	Người dùng đăng nhập vào hệ thống.
Actors	User
Tiền điều kiện	<ul style="list-style-type: none"> • Tài khoản của người dùng đã tồn tại trong hệ thống. • Tài khoản của người dùng Admin đã setup OTP trong hệ thống.
Hậu điều kiện	Người dùng được đăng nhập vào hệ thống và chuyển tới màn hình phù hợp.

8.2. TÀI LIỆU PHÂN TÍCH YÊU CẦU

ID	UC-IAM-LOGIN
Kích bản	Đăng nhập
Luồng chính	<p>1. Người dùng truy cập vào trang đăng nhập của hệ thống</p> <p>2. Hệ thống hiển thị trang đăng nhập và các thông tin</p> <p>3. Người dùng điền các thông tin <i>Username</i> và <i>Mật khẩu</i> và nhấn <i>Login</i></p> <p>4. Hệ thống kiểm tra thông tin của người dùng và hiển thị màn hình nhập TOTP</p> <p>5. Người dùng nhập TOTP gồm 6 số.</p> <p>6. Khi người dùng nhập 6 số, hệ thống tự động kiểm tra thông tin TOTP của người dùng. Với thông tin chính xác, người dùng được chuyển vào homepage tương ứng.</p> <ul style="list-style-type: none"> • Nếu người dùng là <i>Campaign Creator</i>, hệ thống chuyển sang màn hình 8.3 • Nếu người dùng là <i>Data Scientist</i>, hệ thống chuyển sang màn hình 8.5 • Nếu người dùng là <i>Admin</i>, hệ thống chuyển sang màn hình 8.13
Luồng thay thế	<p>4.1. Nếu người dùng không phải là Admin và chưa cấu hình TOTP thì hệ thống chuyển sang bước 6</p>
Luồng thay thế	<p>4.2. Nếu người dùng nhập thông tin sai, hệ thống sẽ hiển thị thông báo: <i>Thông tin chưa chính xác</i>. Người dùng quay về bước 3 để nhập lại.</p> <p>4.3. Nếu người dùng nhập sai quá 3 lần. Trong khoảng thời gian phạt sau đó, hệ thống xem như người dùng nhập thông tin sai tài khoản, kể cả khi người dùng nhập thông tin đúng. Thời gian phạt được tính như sau:</p> <ul style="list-style-type: none"> • Sai 3 lần, phạt 15 phút. • Sai lần thứ 4 (sau khi hết thời gian phạt 15 phút), phạt 30 phút. • Sai lần thứ 5 trở đi (sau khi hết thời gian phạt trước đó), phạt 1 giờ; <p><u>Trong thời gian phạt, không tính số lần sai.</u></p>
Ghi chú	<ul style="list-style-type: none"> • Hệ thống không cho phép người dùng <i>Khôi phục mật khẩu</i> hoặc <i>đăng ký mới</i>. Các tính năng này có trong phần 8.13

CHƯƠNG 8. HỆ THỐNG QUẢN LÝ PHẢN HỒI KHÁCH HÀNG THÔNG MINH

Bảng 8.13: Kịch bản Admin dashboard

ID	UC-ADMIN-DASHBOARD
Kịch bản	Admin dashboard
Mô tả	Admin xem tổng quan về hệ thống
Actors	Admin
Tiền điều kiện	Người dùng đã đăng nhập thành công
Hậu điều kiện	
Luồng chính	<ol style="list-style-type: none"> 1. Người dùng truy cập vào trang admin dashboard 2. Hệ thống hiển thị các thông tin chung của hệ thống <ul style="list-style-type: none"> • Số lượng user • Số lượng dự án khảo sát • Số lượng dự án học máy • Số lượng các tiến trình fetch data đang chạy • Số lượng các tiến trình train đang chạy
Luồng thay thế	<ol style="list-style-type: none"> 3.1. Người dùng nhấn vào danh sách user, hệ thống sẽ chuyển sang kịch bản 8.14 3.2. Người dùng nhấn vào danh sách user, hệ thống sẽ chuyển sang kịch bản 8.14

Bảng 8.14: Kịch bản Liệt kê danh sách user

ID	UC-IAM-ADMIN-LIST-USERS
Kịch bản	Liệt kê danh sách user
Mô tả	Hiển thị danh sách các user trong hệ thống
Actors	Admin
Tiền điều kiện	Người dùng đã đăng nhập thành công
Hậu điều kiện	
Luồng chính	<ol style="list-style-type: none"> 1. Người dùng truy cập vào trang admin list user 2. Hệ thống hiển thị danh sách các thông tin các user trong hệ thống <ul style="list-style-type: none"> • Username • Last login • Role, và nút <i>Change role</i> • Nút <i>Reset Password</i> • Nút <i>Delete</i>

8.2. TÀI LIỆU PHÂN TÍCH YÊU CẦU

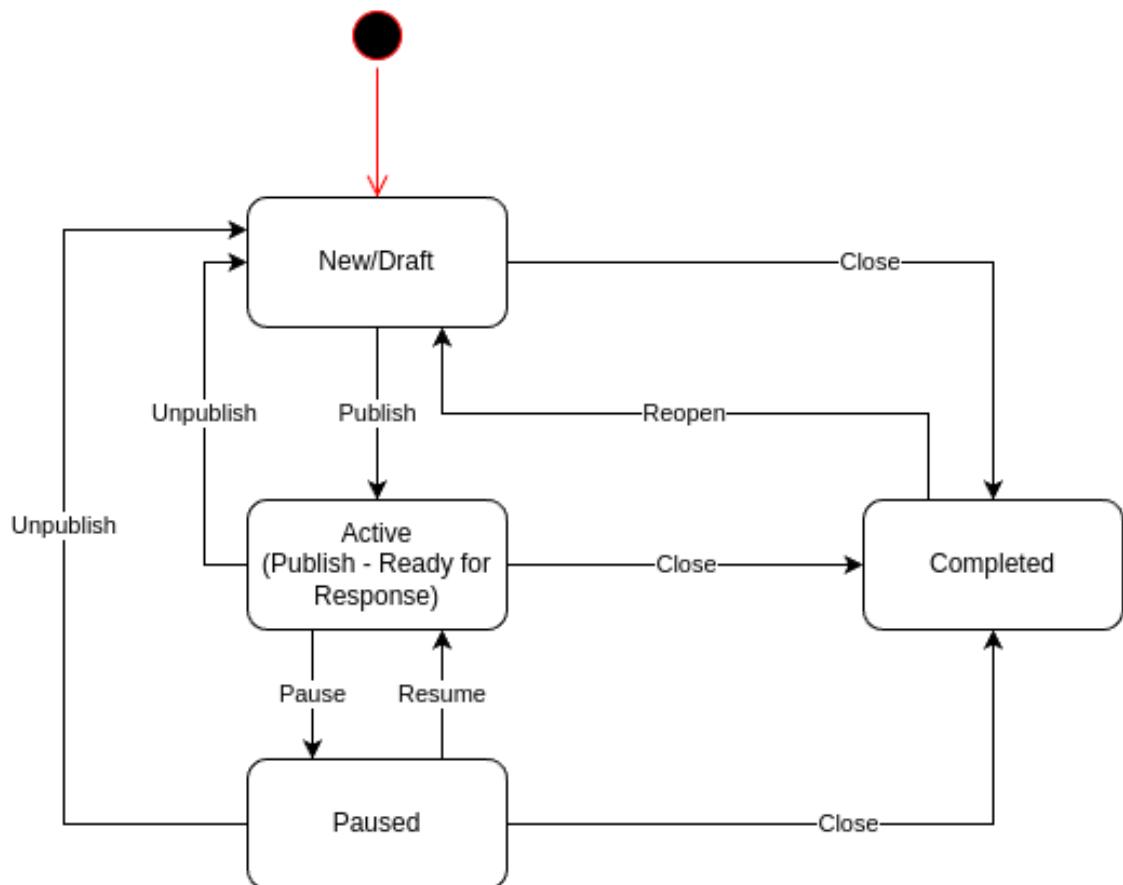
ID	UC-IAM-ADMIN-LIST-USERS
Kích bản	Liệt kê danh sách user
Luồng thay thé	<p>3.1. Admin có thể khôi phục mật khẩu của một người dùng bằng cách:</p> <ul style="list-style-type: none"> 3.1.1. Admin nhấp vào nút <i>Reset Password</i> tương ứng 3.1.2. Admin điền thông tin mật khẩu mới của người dùng 3.1.3. Hệ thống cập nhật thông tin và popup một thông báo đã cập nhật vào hệ thống. <p>3.2. Admin có thể xóa 1 user</p> <ul style="list-style-type: none"> 3.2.1. Admin nhấp vào nút <i>Delete</i> tương ứng 3.2.2. Hộp thoại hiện lên để admin xác nhận lại thao tác: <i>Remove</i> hoặc <i>Cancel</i>. 3.2.3. Hệ thống tiến hành xóa người dùng nếu admin nhấp <i>Remove</i> hoặc bỏ qua nếu Admin nhấp <i>Cancel</i> <p>3.3. Admin có thể đổi vai trò của một người dùng</p> <ul style="list-style-type: none"> 3.3.1. Admin nhấp vào nút <i>Change Role</i> tương ứng 3.3.2. Hộp thoại hiện lên để admin chọn role mới cho người dùng 3.3.3. Hệ thống cập nhật thông tin role mới và kết thúc hộp thoại. <p>3.4. Admin có thể tạo một người dùng mới</p> <ul style="list-style-type: none"> 3.4.1. Admin nhấp vào nút <i>Add user</i> 3.4.2. Hộp thoại hiện lên chờ Admin nhập vào thông tin username và password của người dùng mới. 3.4.3. Admin nhấp nút <i>Tạo</i> để tạo người dùng mới. <p>Tại bước [3.4.3.], Admin có thể nhấn nút <i>Cancel</i> để hủy bỏ việc tạo user mới.</p>

8.2.2.6 Lược đồ trạng thái của hệ thống

Trạng thái của một khảo sát

Lược đồ trạng thái của một bảng khảo sát được mô tả trong hình 8.2. Cụ thể

- **New:** Là trạng thái của một survey được tạo mới hoặc đang chỉnh sửa. Trạng thái này, survey chưa sẵn sàng để thu thập khảo sát từ người dùng
- **Active:** là trạng thái của survey sau khi đã published. Trong trạng thái này, survey đang thu thập khảo sát từ người dùng
- **Paused:** là trạng thái tạm ngưng thu thập khảo sát từ người dùng.
- **Completed:** là trạng thái survey đã hoàn thành khảo sát. Trong trạng thái này, survey không nhận kết quả khảo sát từ người dùng.



Hình 8.2: Lược đồ trạng thái của bảng khảo sát

8.3. TÀI LIỆU THIẾT KẾ HỆ THỐNG

Lưu ý: Không có trạng thái *Removed/Deleted*. Trạng thái *Pause* được đặt sẵn, chờ được thiết lập trong tương lai. Trong phiên bản này, trạng thái *Pause* không được tùy chỉnh.

Trạng thái của một dự án học máy

Hình 8.3 thể hiện các trạng thái của một dự án học máy trong hệ thống. Cụ thể

- **New:** Dự án mới được khởi tạo
- **Data-Ready:** Dự án đã fetched các dữ liệu từ kết quả khảo sát, sẵn sàng cho quá trình huấn luyện.
- **Training:** Đang huấn luyện.
- **Finished:** huấn luyện kết thúc.

8.3 Tài liệu thiết kế hệ thống

8.3.1 Mục đích

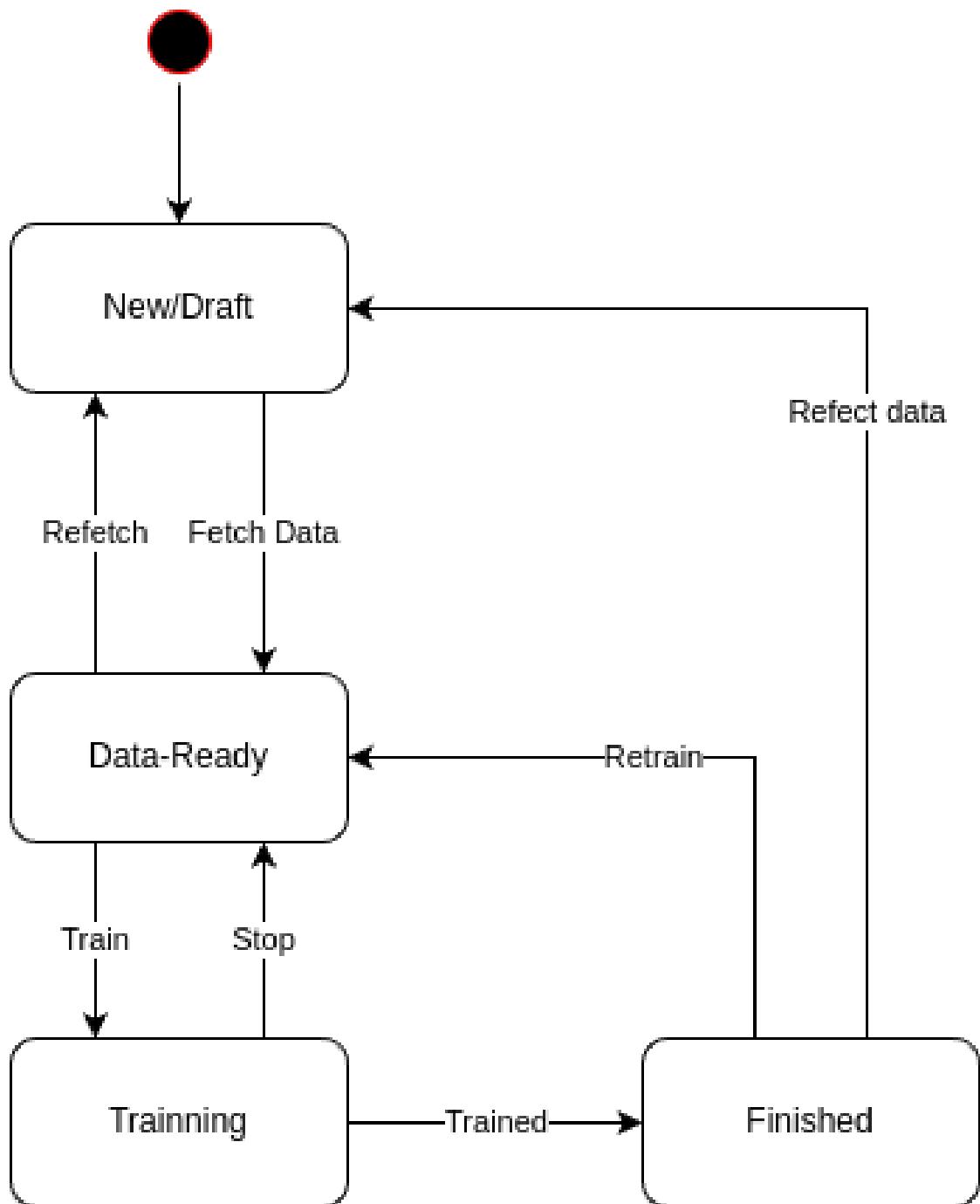
Hệ thống "Smart Customer Feedback Management" là một nền tảng quản lý và phân tích phản hồi khách hàng. Hệ thống giúp doanh nghiệp thu thập, quản lý và phân tích các phản hồi từ khách hàng, đồng thời ứng dụng các mô hình học máy để trích xuất thông tin có giá trị từ dữ liệu.

Tài liệu này mô tả kiến trúc của hệ thống và các module trong hệ thống được xây dựng; cung cấp góc nhìn tổng quan hệ thống

8.3.2 Kiến trúc hệ thống

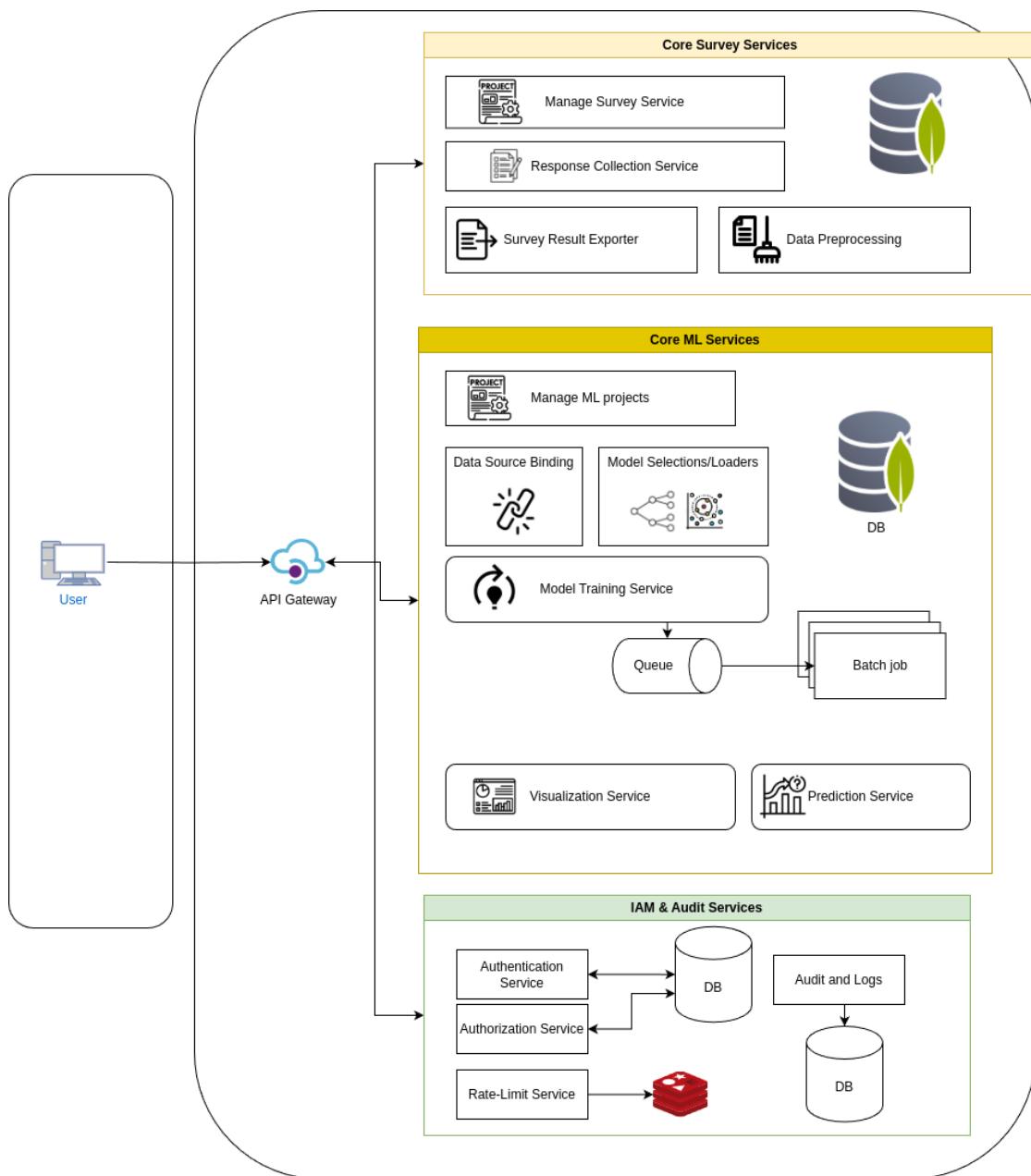
Hình 8.4 mô tả kiến trúc hệ thống SCFM. Hệ thống được đề nghị theo hướng microservices với các khối dịch vụ chính:

- Khối *Core survey*: cung cấp các dịch vụ liên quan đến vấn đề khảo sát
 - Quản lý các dịch vụ tạo khảo sát (Manage Survey)
 - Thu thập các kết quả trả về (Response Collection)
 - Làm sạch dữ liệu (Data Pre-processing),
 - Tạo interface để có thể trích xuất dữ liệu cho các dịch vụ khác (Survey Result Exporter)
- Khối *Core ML*: cung cấp các dịch vụ liên quan đến phần vận hành các mô hình học máy
 - Quản lý các dự án học máy (Manage ML projects)
 - Liên kết cấu hình các nguồn dữ liệu cho dự án học máy (Data Source Binding)
 - Cấu hình mô hình học máy (Model Selection/Loaders): có thể chọn các mô hình học máy phù hợp
 - Huấn luyện mô hình thông qua dịch vụ Model Training, kết hợp hệ thống Queue và Batch jobs để có thể chạy ngầm
 - Dịch vụ Prediction để chạy các mô hình đã train cho người dùng
 - Dịch vụ visualization các dữ liệu hiện có
- Khối *IAM & Audit* cung cấp các dịch vụ login và xác thực cho hệ thống. Đồng thời cung cấp các khả năng Audit Logs, cũng như ratelimit để tránh bị tấn công hoặc lạm dụng



Hình 8.3: Lược đồ trạng thái của một dự án học máy

8.3. TÀI LIỆU THIẾT KẾ HỆ THỐNG



Hình 8.4: Kiến trúc hệ thống

Phần 3

Một số kỹ thuật học máy khác

Chương 9

Linear Regression

9.1 Cơ sở lý thuyết

9.1.1 Giới thiệu về Linear Regression

Linear Regression (hồi quy tuyến tính) là dạng đơn giản và phổ biến nhất trong các phép phân tích dự đoán. Ý tưởng chung của hồi quy là xem xét hai yếu tố sau:

- Liệu danh sách các biến độc lập cho trước có phù hợp để dự đoán biến phụ thuộc?
- Những biến nào đóng vai trò quan trọng trong việc dự đoán và chúng sẽ tác động như thế nào đến kết quả của biến được dự đoán?

Phép ước lượng bằng hồi quy được sử dụng để thể hiện mối quan hệ giữa một biến phụ thuộc với một hay nhiều biến độc lập khác.

Ba ứng dụng chính của phép phân tích hồi quy, đó là:

- Xác định mức độ ảnh hưởng của các biến dùng để dự đoán đến biến được dự đoán.
- Dự đoán sự ảnh hưởng khi thay đổi giá trị của các biến dự đoán.
- Dự đoán xu hướng thay đổi và các giá trị tương lai. Ứng dụng này thường được sử dụng nhiều trong kinh tế để dự đoán sự thay đổi về giá một sản phẩm dựa trên thị trường.

9.1.2 Dạng của Linear Regression

Xét một ví dụ như sau: Một công ty thời trang trong năm x_1 đã đầu tư x_2 triệu đồng vào việc quảng bá các sản phẩm thuộc hàng thời trang của họ. Giả sử nếu như chúng ta cũng có những số liệu tương tự qua nhiều năm của công ty đó cũng như các công ty đối thủ liệu chúng ta có thể dự đoán được doanh thu của công ty trong năm đó. Và nếu có thì sự phụ thuộc giữa doanh thu y so với các biến x_1 và x_2 sẽ như thế nào.

Dễ thấy qua từng năm, khi kinh tế ngày càng phát triển, sự quan tâm và đầu tư của mỗi người vào thời trang cũng tăng lên và khi được quảng bá càng nhiều thì càng nhiều người sẽ biết đến nhãn hàng hơn. Do đó, ta có thể xây dựng một hàm số đơn giản $f(x)$ để mô tả mối quan hệ giữa doanh thu và 2 đại lượng đầu vào.

$$y \approx f(x) = \hat{y} \quad (9.1)$$

với

- y là doanh thu thực sự của công ty,
- và \hat{y} là doanh thu dự đoán.

Một cách đơn giản, ta có thể giả sử hàm $f(x)$ có dạng như sau:

$$f(x) = w_0 + w_1x_1 + w_2x_2 \quad (9.2)$$

trong đó: w_0, w_1, w_2 là các hằng số và w_0 được gọi là *bias*. Mỗi quan hệ $y \approx f(x)$ bên trên là một mối quan hệ tuyến tính (linear) và bài toán chúng ta đang làm là một bài toán thuộc loại hồi quy (regression). Mục đích của bài toán này là đi tìm các hệ số tối ưu $\{w_0, w_1, w_2\}$. Chính vì vậy bài toán này được gọi là bài toán Linear Regression.

Trong phương trình (9.2) như trên nếu chúng ta đặt $\mathbf{w} = [w_0, w_1, w_2]$ và $\mathbf{x} = [1, x_1, x_2]$, phương trình (9.2) có thể được viết lại dưới dạng.

$$y \approx \bar{\mathbf{x}}\mathbf{w} = \hat{y} \quad (9.3)$$

9.1.3 Sai số dự đoán

Ta đặt, e là chênh lệch giữa giá trị y và giá trị dự đoán \hat{y} . Khi đó

$$e = |y - \hat{y}| \quad (9.4)$$

Mục tiêu của chúng ta là giá trị e là nhỏ nhất. Nói cách khác, chúng ta cần tìm y để e đạt min, tương đương việc tìm min của các biểu thức sau

$$\begin{aligned} \min(e) &= \min(|y - \hat{y}|) \\ \min(e^2) &= \min((y - \hat{y})^2) \\ \min\left(\frac{1}{2}e^2\right) &= \min\left(\frac{1}{2}(y - \hat{y})^2\right) = \min\left(\frac{1}{2}(y - \bar{\mathbf{x}}\mathbf{w})^2\right) \end{aligned} \quad (9.5)$$

9.1.4 Hàm mất mát

Phương pháp phổ biến nhất để lựa chọn đường hồi quy là phương pháp bình phương nhỏ nhất (least-squares). Phương pháp này tính đường hồi quy phù hợp nhất dựa trên các mẫu có sẵn bằng cách tối thiểu tổng các bình phương của độ lệch theo trực tung từ điểm dữ liệu đến đường thẳng hồi quy. Bởi vì độ lệch này được bình phương rồi cộng lại với nhau nên sẽ không có sự khử lẫn nhau giữa các giá trị âm và dương. Giá trị của tổng các bình thường này được gọi là **Hàm mất mát** (loss function) và được ký hiệu là $\mathcal{L}(\mathbf{w})$

$$\mathcal{L}(\mathbf{w}) = \frac{1}{2} \sum_{i=1}^N (y_i - \bar{\mathbf{x}}_i \mathbf{w})^2 \quad (9.6)$$

Để tìm được đường hồi quy chính xác nhất, ta cần phải xác định \mathbf{w} sao cho giá trị của hàm mất mát $\mathcal{L}(\mathbf{w})$ là nhỏ nhất. Giá trị của \mathbf{w} làm cho hàm mất mát đặt giá trị nhỏ nhất gọi là *điểm tối ưu* (optimal point), ký hiệu:

$$\mathbf{w}^* = \arg \min_{\mathbf{w}} \mathcal{L}(\mathbf{w}) \quad (9.7)$$

Trước khi đi tìm lời giải, chúng ta đơn giản hóa phép toán trong phương trình hàm mất mát 9.6. Đặt

- $\mathbf{y} = [y_1; y_2; \dots; y_N]$ là một vector cột chứa tất cả các kết quả (*output*) của dữ liệu huấn luyện (*training data*);
- $\bar{\mathbf{X}} = [\bar{\mathbf{x}}_1; \bar{\mathbf{x}}_2; \dots; \bar{\mathbf{x}}_N]$ là ma trận dữ liệu đầu vào (mở rộng) mà mỗi hàng của nó là một điểm dữ liệu.

Khi đó hàm số mất mát $\mathcal{L}(\mathbf{w})$ được viết dưới dạng ma trận đơn giản hơn:

$$\mathcal{L}(\mathbf{w}) = \frac{1}{2} \sum_{i=1}^N (y_i - \bar{\mathbf{x}}_i \mathbf{w})^2 = \frac{1}{2} \|\mathbf{y} - \bar{\mathbf{X}}\mathbf{w}\|_2^2 \quad (9.8)$$

với $\|\mathbf{z}\|_2$ là Euclidean norm (chuẩn hóa Euclid, hay khoảng cách Euclid), nói cách khác $\|\mathbf{z}\|_2^2$ là tổng của bình phương mỗi phần tử của vector \mathbf{z} . Tới đây, ta đã có một dạng đơn giản của hàm mất mát được viết như phương trình (9.8).

9.1.5 Nghiệm của bài toán Linear Regression

Cách phổ biến nhất để tìm nghiệm cho một bài toán tối ưu là giải phương trình đạo hàm (*gradient*) bằng 0. Đạo hàm theo \mathbf{w} của hàm mất mát là:

$$\frac{\partial \mathcal{L}(\mathbf{w})}{\partial \mathbf{w}} = \bar{\mathbf{X}}^T (\bar{\mathbf{X}}\mathbf{w} - \mathbf{y}) \quad (9.9)$$

Phương trình đạo hàm bằng 0 tương đương với:

$$\bar{\mathbf{X}}^T \bar{\mathbf{X}}\mathbf{w} = \bar{\mathbf{X}}^T \mathbf{y} \triangleq \mathbf{b} \quad (9.10)$$

(ký hiệu $\bar{\mathbf{X}}^T \mathbf{y} \triangleq \mathbf{b}$ nghĩa là đặt $\bar{\mathbf{X}}^T \mathbf{y}$ bằng \mathbf{b}).

Trong trường hợp $\mathbf{A} \triangleq \bar{\mathbf{X}}^T \bar{\mathbf{X}}$ không khả nghịch thì phương trình (9.10) vô nghiệm hoặc có vô số nghiệm. Do đó chúng ta sẽ sử dụng khái niệm *giả nghịch đảo* \mathbf{A}^\dagger để chỉ giá trị nghịch đảo của \mathbf{A} trong cả trường hợp khả nghịch lẫn không khả nghịch.

Với khái niệm giả nghịch đảo, điểm tối ưu của bài toán Linear Regression có dạng:

$$\mathbf{w} = \mathbf{A}^\dagger \mathbf{b} = (\bar{\mathbf{X}}^T \bar{\mathbf{X}})^\dagger \bar{\mathbf{X}}^T \mathbf{y} \quad (9.11)$$

9.2 Chuẩn bị dữ liệu để thực hiện hồi quy tuyến tính

Hầu hết mọi mối quan hệ hồi quy giữa các biến đều có thể được biểu diễn lại dưới dạng hồi quy tuyến tính. Tuy nhiên, không phải lúc nào từ số liệu thô ban đầu, ta cũng có thể xây dựng một cách trực tiếp được hàm hồi quy một cách chính xác nhất.

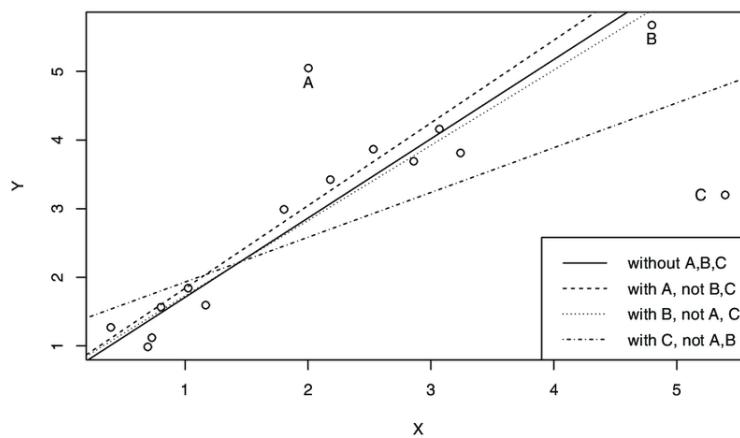
Một trong những hạn chế lớn nhất của Hồi quy tuyến tính là nó rất dễ bị ảnh hưởng bởi các điểm ngoại lai (*outlier*).

Ví dụ như trong hình 9.1, với hai điểm A, B thì hàm số bị lệch nhưng không nhiều. Trong khi đối với điểm C, hàm số đã bị sai khác đi rất nhiều, và không thể sử dụng hàm hồi quy này để dự đoán các kết quả trong tương lai được.

Vì vậy, trước khi thực hiện quá trình Hồi quy tuyến tính, các điểm ngoại lai cần phải được loại bỏ. Bước này gọi là tiền xử lý (*pre-processing*).

Các phương pháp tiền xử lý số liệu:

YUAN AND ZHONG



Hình 9.1: Sự thay đổi của hàm hồi quy khi có các giá trị nhiễu

- **Giả định tuyến tính:** Hồi quy tuyến tính xem mối quan hệ giữa input và output là tuyến tính và nó không hỗ trợ cho các dạng hồi quy khác. Do đó chúng ta cần phải chuyển đổi dạng dữ liệu để nó tuân theo mối quan hệ tuyến tính (ví dụ ta có thể chuyển đổi căn bậc hai để đưa một hàm bậc hai về hàm tuyến tính).
- **Loại bỏ nhiễu:** Hồi quy tuyến tính hoạt động tốt nhất khi các biến input và output không xuất hiện các giá trị nhiễu. Nếu chúng ta loại bỏ nhiễu tốt thì kết quả thu được sẽ càng chính xác. Có rất nhiều phương pháp¹ để thực hiện loại bỏ nhiễu để chúng ta có thể lựa chọn phương pháp phù hợp nhất.
- **Loại bỏ tính cộng tuyến:** Hồi quy tuyến tính sẽ over-fit² tập dữ liệu nếu tồn tại các biến input đa cộng tuyến³. Để giải quyết vấn đề này, chúng ta có thể tính toán mức tương quan của từng cặp input và loại bỏ biến có độ tương quan cao nhất.
- **Phân phối Gausian:** Hồi quy tuyến tính sẽ cho kết quả dự đoán đáng tin cậy hơn nếu input và output tuân theo phân phối Gaussian.
- **Thay đổi tỷ lệ dữ liệu vào:** Hồi quy tuyến tính sẽ cho kết quả dự đoán đáng tin cậy hơn nếu được điều chỉnh tỷ lệ của các dữ liệu vào (input) bằng cách chuẩn hóa.

Ví dụ 9.1. Tìm hàm số $f(x) = A + Bx$ xấp xỉ tốt nhất với Bảng 9.1.

Bảng 9.1: Bảng giá trị dữ liệu của hàm số $f(x)$

x	1	1	2	2	2	3	3	4	5	6
y	1	2	2	3	4	4	5	5	6	7

¹một số phương pháp phổ biến như Extreme Value Analysis, Probabilistic and Statistical Models, Linear Models, Proximity-based Models, Information Theoretic Models, High-Dimensional Outlier Detection

²hiện tượng hàm số được xây dựng quá fit với dữ liệu và sẽ gây phản tác dụng khi dự đoán

³hiện tượng các biến độc lập có mối quan hệ tương quan với nhau và có thể thể hiện được dưới dạng hàm số

Giá trị của hàm mất mát được nêu trong công thức 9.12

$$L = \frac{1}{2} \sum_{k=1}^n (A + Bx_k - y_k)^2 \quad (9.12)$$

Bài toán quy về tìm cực tiểu của hàm hai biến $L(A, B)$. Tọa độ điểm dừng của hàm này được xác định bởi hệ phương trình

$$\begin{cases} \frac{\partial}{\partial A} \frac{1}{2} \sum_{k=1}^n (A + Bx_k - y_k)^2 = \sum_{k=1}^n (A + Bx_k - y_k) = 0 \\ \frac{\partial}{\partial B} \frac{1}{2} \sum_{k=1}^n (A + Bx_k - y_k)^2 = \sum_{k=1}^n (A + Bx_k - y_k)x_k = 0 \end{cases} \quad (9.13)$$

$$\Leftrightarrow \begin{cases} nA + \left(\sum_{k=1}^n x_k \right) B = \sum_{k=1}^n y_k \\ \left(\sum_{k=1}^n x_k \right) A + \left(\sum_{k=1}^n x_k^2 \right) B = \sum_{k=1}^n x_k y_k \end{cases} \quad (9.14)$$

Từ bảng số trên ta có $n = 10$ và $\sum_{k=1}^n x_k = 29$, $\sum_{k=1}^n y_k = 39$, $\sum_{k=1}^n x_k^2 = 109$, $\sum_{k=1}^n x_k y_k = 140$. Hệ phương trình để xác định A, B có dạng.

$$\begin{cases} 10A + 29B = 39 \\ 29A + 109B = 140 \end{cases} \Leftrightarrow \begin{cases} A = 0.7671 \\ B = 1.0803 \end{cases}$$

Do đó đường thẳng cần tìm là $f(x) = 0.7671 + 1.0803x$

9.3 Tranning model sử dụng Linear Regression với dữ liệu thực tế

9.3.1 Mẫu dữ liệu sử dụng

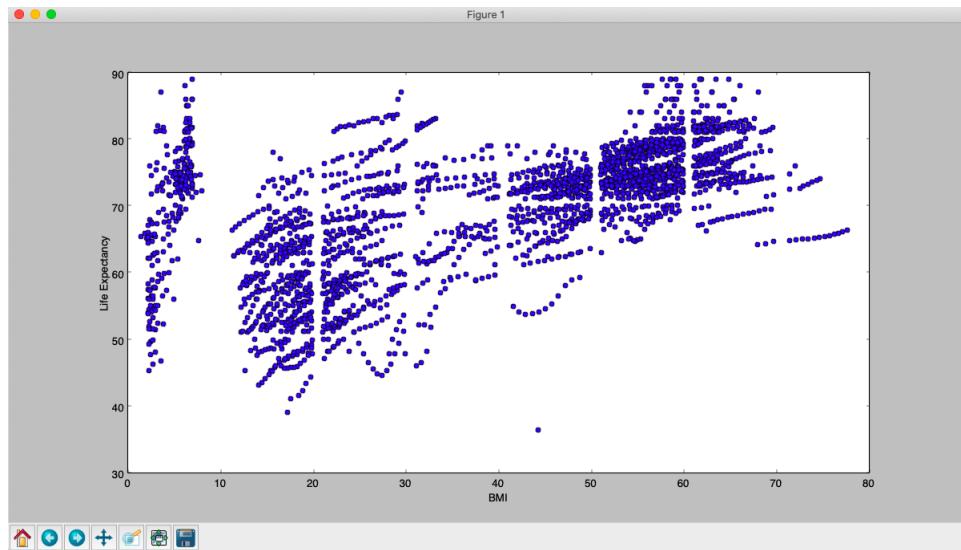
Trong ví dụ này ta dùng mẫu dữ liệu lấy từ trang kaggle.com⁴, được thu thập từ dữ liệu của Dài quan sát sức khỏe toàn cầu (GHO) thuộc Tổ chức Y tế Thế giới (WHO).

Mẫu dữ liệu này tập trung vào các yếu tố tiềm chung, yếu tố tử vong, yếu tố kinh tế, yếu tố xã hội và các yếu tố liên quan đến sức khỏe khác. Do các quan sát bộ dữ liệu này dựa trên các quốc gia khác nhau, nên một quốc gia sẽ dễ dàng xác định yếu tố dự đoán sẽ góp phần làm giảm giá trị tuổi thọ. Điều này sẽ giúp gợi ý một quốc gia mà khu vực cần được coi trọng để cải thiện hiệu quả tuổi thọ của dân số. Ngoài ra nó còn giúp mỗi cá nhân tự ý thức cải thiện bản thân, tăng cường lối sống lành mạnh để nâng cao tuổi thọ.

Bộ dữ liệu gồm 2938 dòng và 22 cột. Các dòng bao gồm: country, year, developing status, adult mortality, life expectancy, infant deaths, alcohol consumption per capita, country's expenditure on health, immunization coverage, BMI, deaths under 5-years-old, deaths due to HIV/AIDS, GDP, population, body condition, income information, và education.

⁴<https://www.kaggle.com/datasets/kumarajarshi/life-expectancy-who>

9.3.2 Ví dụ về chỉ số BMI



Hình 9.2: Đồ thị tương quan giữa chỉ số BMI và tuổi thọ

Để dễ hình dung, ta sẽ chỉ vẽ đồ thị 2D cho yếu tố chỉ số BMI để quan sát sự tương quan trong mẫu dữ liệu.

Đồ thị cho thấy chỉ số BMI tăng thì tuổi thọ sẽ có xu hướng tăng. Bỏ qua các yếu tố khác, ta có thể giải thích rằng những người béo phì tuy dễ mắc các bệnh liên quan đến tim, đường huyết,... Tuy nhiên điều kiện sống của họ đa phần là ở mức cao và được hưởng nền y tế tiên tiến. Ví dụ như ở Mỹ, tuổi thọ trung bình cao⁵ mặc dù đây là quốc gia có tỉ lệ béo phì cao⁶.

9.3.3 Huấn luyện mô hình

Trong ví dụ này, ta sẽ sử dụng tập dữ liệu trên và train model. Ta sẽ dùng Sample data là 2500 dòng đầu tiên và các cột từ 4 đến 21.

Trong đoạn code một số hàm trong `Sklearn` để chuẩn hóa dữ liệu, random dữ liệu train, test, tính Mean squared error.

Algorithm 9.1: Hiện thực linear regression

Result: Hiện thực hồi quy tuyến tính

```
1 Chuẩn hóa dữ liệu huấn luyện X, y;  
2 Khởi tạo trọng số;  
3 while i < số step đã chọn do  
4     // Tính delta  
5     delta = (chuyển vị của X) * (X * trọng số - y);  
6     // Cập nhật trọng số  
7     trọng số = trọng số - (Tỉ lệ học / Số điểm của dữ liệu) * delta;  
8 end
```

Kết quả của chương trình thu được là (Các giá trị có thể thay đổi trong các lần thực thi do việc thay đổi learning rate và number of step. Ngoài ra trong code có sử dụng phân chia ngẫu nhiên tập dữ liệu thành phần `train` và `test`).

Theo kết quả, SME là 0.2839 - sai số có thể chấp nhận được. Các hệ số:

Bảng 9.2: Kết quả sau khi train

Model's Coefficients: [[-0.19863653 0.02868794 0.12524506 0.04065507 0.04024758 -0.0048461 0.06174491 -0.05350559 0.06243888 -0.04618593 0.07045321 -0.23154371 0.05984205 0.00722812 -0.0206898 -0.00961901 0.20172049 0.14472694]]
Model's Intercept: [0.00321724]

9.3.4 Huấn luyện mô hình với Sklearn

Thay vì thực hiện thủ công như cách trên, thư viện Sklearn cung cấp sẵn các hàm để train dữ liệu một cách nhanh chóng.

⁵78,54 tuổi (2017)

⁶tỷ lệ béo phì trung bình ở người Mỹ trưởng thành là 31% (2019)

9.4. ỨNG DỤNG CHO ĐIỂM VÀ ĐƯA RA LỜI KHUYÊN SỨC KHỎE

```
# Scale data
input_scaled = StandardScaler().fit_transform(input)
output_scaled = StandardScaler().fit_transform(output)

# Split data into train and test
X_train, X_test, y_train, y_test = train_test_split(input_scaled, output_scaled)

# Train model
model = LinearRegression()
model.fit(input_scaled, output_scaled)

# Test with X_test
y_predict = model.predict(X_test)
mse = mean_squared_error(y_predict, y_test)
```

Kết quả của chương trình thu được là (Các giá trị có thể thay đổi trong các lần thực thi do hàm `train_test_split` sẽ random để chia tập `train` và `test`)

Theo kết quả, SME là 0.2571 - sai số có thể chấp nhận được. Các hệ số:

Bảng 9.3: Kết quả sau khi train với thư viện SKLearn

```
Model's Coefficients: [[-0.19245038 1.37291682 0.12846941 0.01428586
0.02682785 -0.0032278 0.0733252 -1.39138518 0.05624307 -0.03669052 0.062629
-0.2331516 0.08785803 -0.00839027 -0.02979804 -0.01735239 0.1685084
0.13921772]]
Model's Intercept: [2.7304267e-15]
```

Dể hiểu rõ hơn cách hiện thực quá trình huấn luyện mô hình của hai cách trên, các bạn có thể tham khảo code bằng cách truy cập đường link sau <https://github.com/thanhlam152/Linear-Regression>

9.4 Ứng dụng cho điểm và đưa ra lời khuyên sức khỏe

9.4.1 Ý tưởng

Theo mẫu dữ liệu ta vừa phân tích, ta có thể thấy dựa vào các yếu tố ngoại cảnh ta có thể ước lượng được tuổi thọ của con người. Từ đó ý tưởng của nhóm sẽ tạo 1 ứng dụng giúp người dùng dự đoán tuổi thọ của mình thông qua các dữ liệu được người dùng cung cấp. Từ đó đưa ra lời khuyên để người dùng cải thiện các chỉ số (Giảm nồng độ cồn tiêu thụ, tiêm ngừa Vaccin,...) để có lối sống lành mạnh và gia tăng tuổi thọ.

Mở rộng với các mẫu dữ liệu khác liên quan nhiều hơn đến sức khỏe hiện tại (tuổi, nồng độ đường và mỡ trong máu, tiền sử bệnh, chế độ ăn, ...) để đưa ra lời khuyên mang tính cá nhân và chính xác hơn. Thay vì đưa ra tuổi thọ dự đoán ta sẽ đưa ra điểm để tránh gây hoang mang cho người dùng với công thức sau:

$$\text{Điểm} = (\text{Tuổi thọ dự đoán} * 1.5) \bmod 100$$

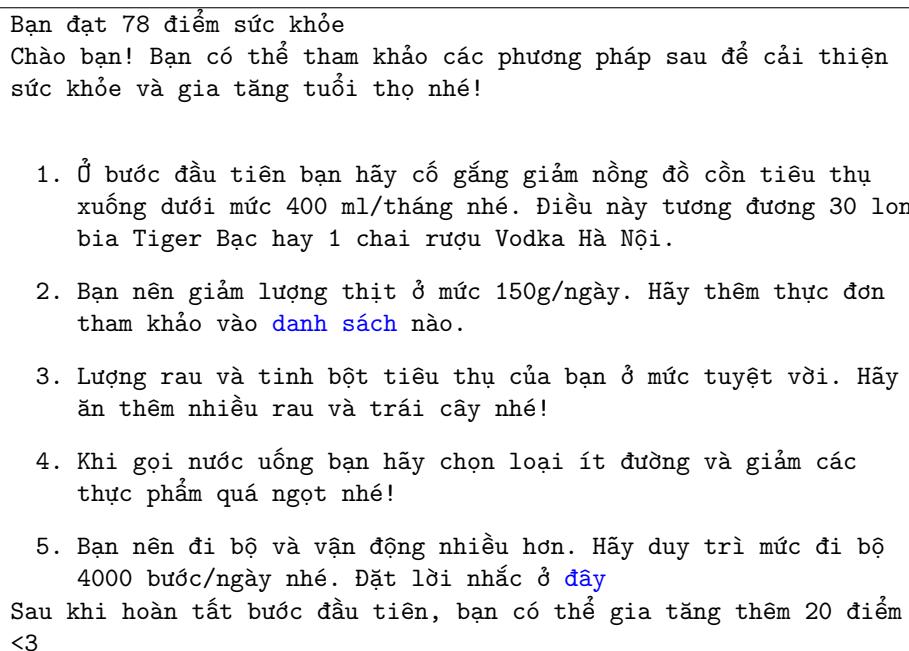
9.4.2 Mô tả ứng dụng

Input: Dữ liệu đầu vào là một mảng chứa các thông tin: tuổi hiện tại, chỉ số BMI, lượng cồn tiêu thụ (ml/tháng), lượng thịt tiêu thụ (kg/tháng), lượng rau tiêu thụ (kg/tháng), lượng tinh bột tiêu thụ (kg/tháng), lượng đường trong máu (mmol/l), số bước đi bộ trung bình (bước/ngày), thu nhập (USD/tháng). Ví dụ:

[32, 30.5, 600, 7.9, 12.0, 10.8, 13.0, 2500, 3000]

Các input sẽ được thu thập từ các cảm biến trên thiết bị đeo thông minh (Ví dụ như Apple Watch, máy đo chỉ số cơ thể, ...)

Output: Đầu ra sẽ là tuổi thọ ước tính và lời khuyên để nâng cao tuổi thọ của người dùng. Ví dụ:



Các dòng màu xanh sẽ thêm dịch vụ vào trong ứng dụng hoặc gọi đến các dịch vụ của đối tác (Ví dụ như đặt thực đơn và gọi Grab Food mang đồ ăn đã chế biến đến tận nơi).

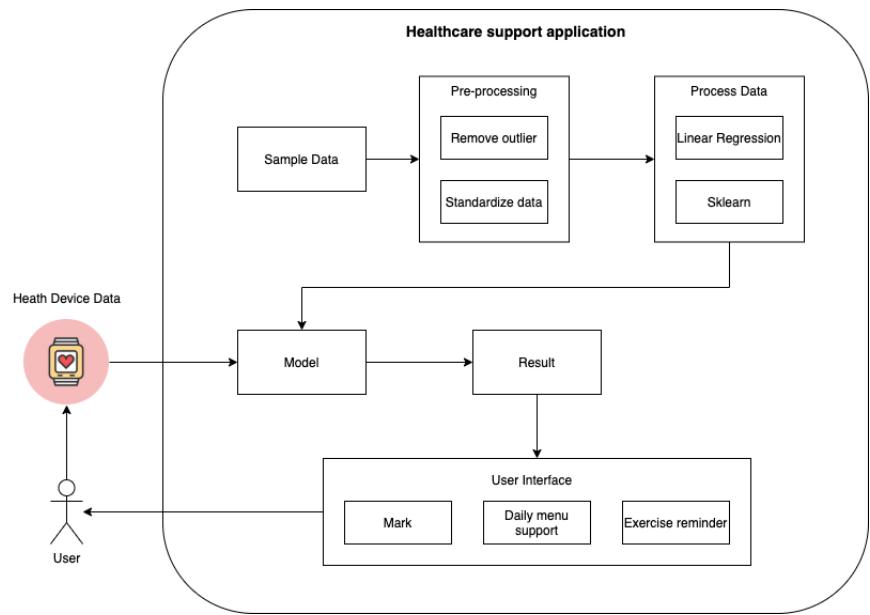
9.4.3 Kiến trúc phần mềm

9.4.4 Sequence diagram

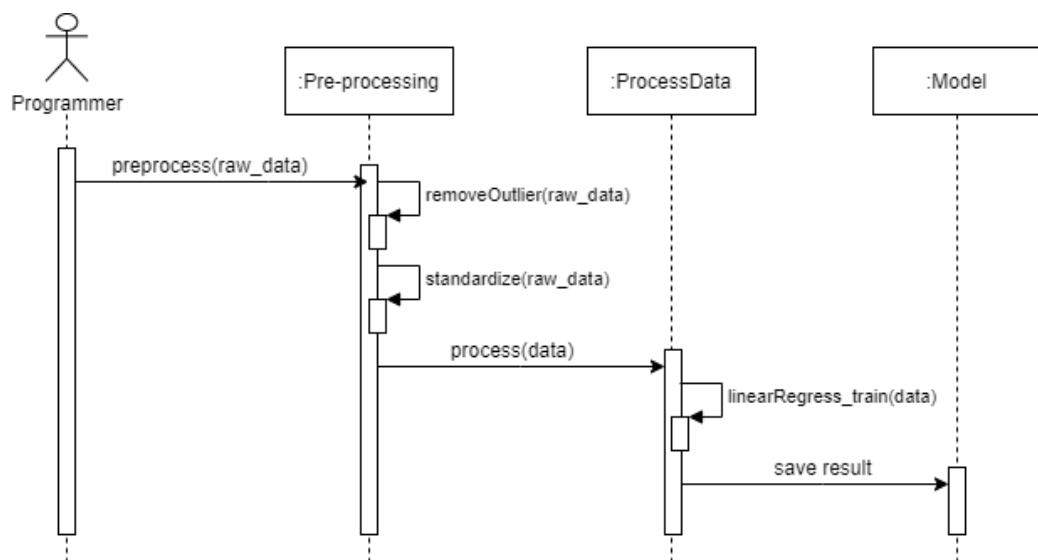
9.5 Bài tập

Hùng đang định mở một cửa hàng bánh. Hùng muốn định giá một loại bánh của mình dựa trên ba tiêu chí sau: *giá thành nguyên liệu*, *thời gian thực hiện*, *độ ngon* (độ ngon sẽ được chia theo thang từ 1-10). Giả sử các đại lượng này hồi quy tuyến tính với giá sản phẩm. Để thực hiện định giá, Hùng có đi tham khảo giá bánh của các cửa hiệu bên cạnh và thu được Bảng 9.4.

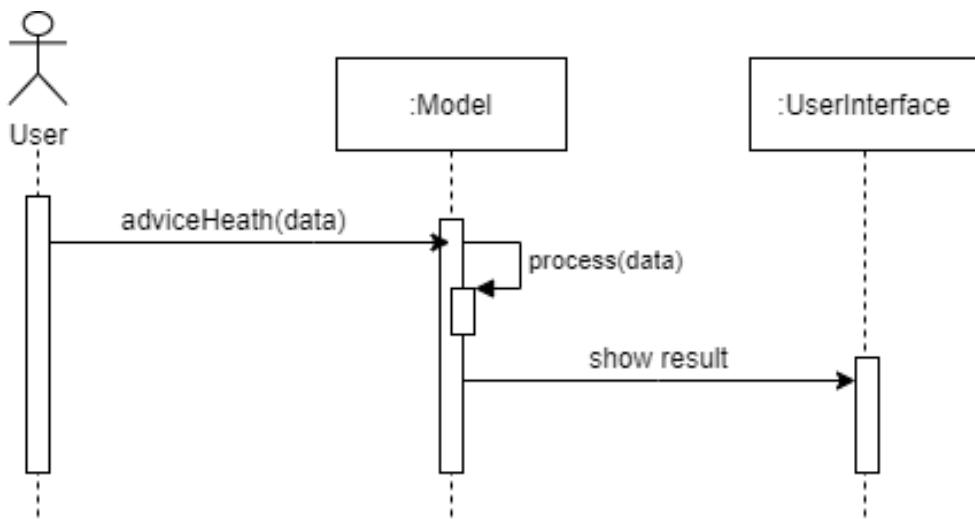
9.5. BÀI TẬP



Hình 9.3: Kiến trúc phần mềm



Hình 9.4: Sequence Diagram khi train Sample data



Hình 9.5: Sequence Diagram khi người dùng sử dụng

Bảng 9.4: Bảng dữ liệu giá thành sản phẩm của Hùng

Giá nguyên liệu (VND)	Thời gian (phút)	Độ ngọt	Giá bán (VND)
20.000	45	7	50.000
30.000	15	8	70.000
45.000	10	6	80.000
20.000	15	6	50.000
25.000	30	8	60.000
18.000	23	7	30.000
30.000	13	5	55.000
48.000	10	5	70.000
22.000	15	6	50.000
20.000	60	10	60.000

9.5. BÀI TẬP

Ta sẽ sử dụng bảng dữ liệu 9.4 cho các câu hỏi sau

- **Câu hỏi 1:** Xây dựng mô hình tuyến tính từ bảng dữ liệu trên. Từ đó dự đoán giá của một cái bánh có giá nguyên liệu là 27.000 VND, thời gian thực hiện là 55 phút và độ ngon là 7.
- **Câu hỏi 2:** Tìm và loại bỏ outlier xuất hiện trong bảng dữ liệu. Từ đó xây dựng lại mô hình hồi quy tuyến tính và thực hiện định giá lại cho câu 1.
- **Câu hỏi 3:** Sau khi sử dụng mô hình hồi quy tuyến tính để định giá bánh, tiệm bánh của Hùng được khá nhiều người ủng hộ nhờ có chất lượng và giá cả hợp lý. Do đó, Hùng quyết định đầu tư kinh doanh về nhà nghỉ. Các bạn hãy giúp Hùng chọn các biến (yếu tố) để định giá phòng dựa trên mô hình hồi quy tuyến tính.
- **Câu hỏi 4:** Phân tích và so sánh giữa ước lượng một mô hình giữa các biến bằng hồi quy tuyến tính so với phương pháp Lagrange. Những vấn đề mà phương pháp Lagrange gặp phải có thể được khắc phục bởi hồi quy tuyến tính?
- **Câu hỏi 5:** Có thể hiện thực mô hình hồi quy tuyến tính bằng cách sử dụng một mạng neural nhân tạo hay không? Nếu có hãy nêu ý tưởng thực hiện, ngược lại hãy giải thích lý do.

Chương 10

Logistic Regression

Ở chương 1, chúng ta đã tìm hiểu về hồi quy tuyến tính với đầu ra thu được là một giá trị thực, thì ở chương này sẽ giới thiệu về hồi quy logistic với đầu ra là một giá trị nhị phân (0 hoặc 1). Nhờ đó, chúng ta có thể áp dụng trong các bài toán như xác định một thư điện tử có phải là spam hay không, một khối u là u lành tính hay u ác tính.

10.1 Bài toán mở đầu

Trường trung học cơ sở ABC đang chuẩn bị tổ chức kỳ thi tuyển sinh đầu vào cho các bạn học sinh tiểu học trong địa phương. Đề thi gồm 2 nội dung: Toán và Văn. Vào một ngày đẹp trời, ban giám hiệu của trường ABC đặt ra một câu hỏi: liệu một ngày nào đó đất nước chúng ta có dịch bệnh, khi ấy công tác thi cử sẽ gặp rất nhiều khó khăn cũng như các bạn học sinh sẽ phải chịu nhiều thiệt thòi trong quá trình ôn tập, chúng ta có thể tìm ra một giải pháp nào khác để dự đoán một bạn học sinh có thể thi đậu kỳ thi đầu vào của trường chúng ta hay không?

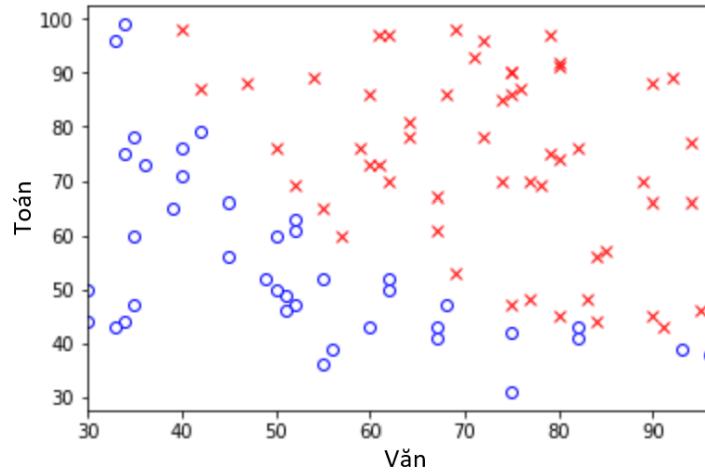
Sau khi phân tích, ban giám hiệu nhận thấy rằng hai yếu tố quyết định đến việc thi đậu hay không chính là điểm trung bình hai môn Toán và Văn ở năm học cuối cấp của các bạn học sinh. Biết dữ liệu thu được của 100 học sinh từng tham gia kỳ thi đầu vào của trường là như Hình 10.1:

Văn	Toán	Đậu												
35	78	0	67	43	0	84	56	1	75	86	1	89	70	1
30	44	0	90	66	1	52	47	0	35	47	0	95	46	1
36	73	0	51	49	0	94	66	1	56	39	0	67	67	1
60	86	1	34	44	0	82	41	0	30	50	0	57	60	1
79	75	1	78	69	1	51	46	0	45	66	0	80	91	1
45	56	0	62	70	1	62	52	0	67	41	0	68	86	1
61	97	1	80	45	1	77	70	1	40	98	1	42	79	0
75	47	1	93	39	0	98	87	1	49	52	0	75	90	1
76	87	1	62	50	0	62	97	1	80	92	1	79	97	1
84	44	1	39	65	0	92	89	1	67	61	1	52	61	0
96	38	0	61	73	1	80	74	1	33	43	0	94	77	1
75	31	0	85	57	1	99	61	1	64	78	1	90	88	1
82	76	1	52	63	0	91	43	1	72	96	1	55	36	0
69	98	1	52	69	1	35	60	0	60	73	1	74	85	1
40	76	0	40	71	0	50	50	0	59	76	1	90	45	1
54	89	1	55	52	0	50	60	0	100	72	1	83	48	1
69	53	1	34	99	0	98	69	1	47	88	1	42	87	1
68	47	0	64	81	1	33	96	0	50	76	1	99	69	1
71	93	1	75	42	0	74	70	1	60	43	0	55	65	1
77	48	1	34	75	0	72	78	1	82	43	0	75	90	1

Hình 10.1: Kết quả thi ứng với điểm trung bình (trên thang 100) các môn Văn và Toán ở năm học cuối cấp

Để dễ quan sát, ban giám hiệu phác thảo lại đồ thị liên hệ giữa điểm trung bình Toán, Văn

ở năm học cuối cấp và kết quả thi như Hình 10.2:



Hình 10.2: Đồ thị giữa điểm Toán, Văn và kết quả thi

Trong đó các điểm dữ liệu hình chữ x màu đỏ đại diện cho các học sinh thi đậu kỳ thi đầu vào, và ngược lại các điểm dữ liệu hình tròn màu xanh đại diện cho các học sinh không thi đậu.

Về mặt logic, giờ ban giám hiệu cần tìm đường thẳng phân chia giữa các điểm đậu và rớt. Sau đó sẽ quyết định một điểm mới là đậu hay rớt từ đường thẳng đó. Tuy nhiên, có thể xảy ra trường hợp số học sinh đậu trong năm đó là quá cao, đòi hỏi nhà trường phải thắt chặt hơn tiêu chí xét tuyển của mình. Điều đó đặt ra yêu cầu cho ban giám hiệu phải tính toán được xác suất một học sinh thi đậu là bao nhiêu để ưu tiên lựa chọn những học sinh có xác suất thi đậu cao hơn.

10.2 Hàm sigmoid

Trong thực tế chúng ta luôn phải đổi mặt với rất nhiều bài toán về ước lượng xác suất để một sự kiện có khả năng xảy ra là bao nhiêu. Đối với những bài toán này thì miền giá trị của biến mục tiêu chỉ nằm trong khoảng từ $[0,1]$ nên việc áp dụng mô hình hồi quy tuyến tính sẽ không hợp lý nữa do hàm tuyến tính không bị giới hạn miền giá trị trả về.

Do đó chúng ta cần xây dựng một hàm số khác đảm bảo được các tiêu chí:

- Hàm số phải liên tục và nhận miền giá trị trong khoảng $(0,1)$.
- Đạo hàm của nó nên là một hàm số đẹp. Khi đó việc áp dụng thuật toán gradient descent sẽ đơn giản hơn.
- Hàm số phải đồng biến hoặc nghịch biến đối với các biến dự đoán bởi trong thực tế một biến sẽ chỉ đóng 1 vai trò ảnh hưởng tiêu cực hoặc tích cực lên biến mục tiêu mà không thể đóng 2 vai trò cùng lúc (vừa tác động cùng chiều và ngược chiều).

- Hàm mất mát (loss function) phải tìm được nghiệm tối ưu để xác suất tiên nghiệm đối với một bộ dữ liệu biết trước là lớn nhất. Hay nói cách khác tồn tại ước lượng hợp lý tối đa (maximum loglikelihood estimation) cho hàm mất mát.

Trong số các hàm số thỏa mãn 4 tính chất nói trên thì hàm sigmoid

$$\sigma(x) = \frac{1}{1 + e^{-x}}$$

được ưa chuộng sử dụng nhiều nhất bởi vì nó thỏa mãn tính liên tục và bị chặn trong khoảng $(0,1)$.

$$\begin{aligned}\lim_{x \rightarrow -\infty} \sigma(x) &= 0 \\ \lim_{x \rightarrow \infty} \sigma(x) &= 1\end{aligned}$$

Đặc biệt hơn thế nữa sigmoid là hàm số đồng biến có đạo hàm rất đẹp và đơn giản

$$\sigma(x)' = \frac{e^{-x}}{(1 + e^{-x})^2} = \frac{1}{1 + e^{-x}} \cdot \frac{e^{-x}}{1 + e^{-x}} = \sigma(x)(1 - \sigma(x)) > 0$$

Nhờ thỏa mãn những tính chất đặc biệt như vậy mà hàm sigmoid có rất nhiều ứng dụng trong lĩnh vực Học máy, đặc biệt là ứng dụng trong việc xây dựng mô hình hồi quy Logistic.

10.3 Thiết lập bài toán

Quay trở lại vấn đề, để giải quyết bài toán mở đầu, chúng ta thực hiện lần lượt các bước:

- Thiết lập mô hình
- Thiết lập hàm mất mát
- Tìm tham số bằng việc tối ưu hàm mất mát
- Dự đoán dữ liệu mới bằng mô hình vừa tìm được

10.3.1 Mô hình

Mô hình hồi quy logistic là một mô hình hồi quy áp dụng hàm số sigmoid nhằm dự đoán giá trị đầu ra rẽ rác y ứng với một véc-tơ đầu vào \bar{x} . Việc này tương đương với chuyện phân loại các đầu vào \bar{x} vào các nhóm y tương ứng.

Điểm khác biệt rõ ràng nhất giúp ta phân biệt một bài toán nên dùng mô hình hồi quy Logistic hay các loại mô hình hồi quy khác đó chính là biến mục tiêu của bài toán đó.

Một biến mục tiêu y thường có 2 dạng chính là định tính hoặc định lượng

- Với biến mục tiêu là biến định lượng liên tục, bài toán sẽ thường được áp dụng mô hình hồi quy tuyến tính, bao gồm tuyến tính đơn biến và tuyến tính đa biến (xem lại chương 1).
- Với biến mục tiêu là biến định tính rời rạc, bài toán chủ yếu áp dụng mô hình hồi quy Logistic. Cụ thể hơn:

10.3. THIẾT LẬP BÀI TOÁN

- Chúng ta có phương pháp Nominal Logistic Regression ứng với biến mục tiêu là biến định danh. Ví dụ như bài toán phân loại nghề nghiệp gồm có Bác sĩ, Kỹ sư, Nhà khoa học.
- Chúng ta có phương pháp Ordinal Logistic Regression ứng với biến mục tiêu là biến thứ bậc. Ví dụ như bài toán phân loại thành tích gồm có Nhất, Nhì, Ba.
- Chúng ta có phương pháp Binary Logistic Regression ứng với biến mục tiêu là biến thay phiên (hay còn gọi là biến nhị phân). Dạng này gọi là biến nhị phân vì đầu ra của nó chỉ có khả năng xảy ra đúng một trong hai trường hợp. Ví dụ phân loại giới tính là Nam hoặc Nữ. Không thể cả hai mà cũng không thể không có giới tính nào.

Chính vì vậy nên với bài toán mở đầu, chúng ta chỉ có thể vận dụng mô hình hồi quy Logistic nhị phân.

Với dòng thứ i trong bảng dữ liệu, gọi $x_1^{(i)}$ là điểm trung bình môn văn và $x_2^{(i)}$ là điểm trung bình môn toán năm học cuối cấp của học sinh thứ i . Khi đó:

$p(x^{(i)} = 1) = \hat{y}_i$ là xác suất mà mô hình dự đoán học sinh thứ i là đậu.

Còn $p(x^{(i)} = 0) = 1 - \hat{y}_i$ là xác suất mà mô hình dự đoán học sinh thứ i không đậu.

Ta có công thức của mô hình hồi quy logistic cho bài toán mở đầu là:

$$\hat{y}_i = \sigma(\omega_0 + \omega_1 * x_1^{(i)} + \omega_2 * x_2^{(i)}) = \frac{1}{1 + e^{-(\omega_0 + \omega_1 * x_1^{(i)} + \omega_2 * x_2^{(i)})}} \quad (10.1)$$

10.3.2 Hàm mất mát

Sau khi đã xây dựng mô hình hồi quy Logistic cho bài toán, giờ ta cần có một hàm số để đánh giá độ tốt của mô hình. Mô hình sẽ là càng tốt nếu như \hat{y} càng gần với y .

- Nếu học sinh thứ i là đậu, tức $y_i = 1$ thì ta cũng mong muốn \hat{y} càng gần 1 càng tốt, hay mô hình dự đoán xác suất học sinh thứ i đậu càng cao càng tốt.
- Nếu học sinh thứ i là không đậu, tức $y_i = 0$ thì ta cũng mong muốn \hat{y} càng gần 0 càng tốt, hay mô hình dự đoán xác suất học sinh thứ i đậu càng thấp càng tốt.

Với mỗi điểm $(x^{(i)}, y_i)$, ta gọi hàm mất mát

$$L_i = -(y_i * \log(\hat{y}_i) + (1 - y_i) * \log(1 - \hat{y}_i)) \quad (10.2)$$

Biểu thức trên tương đương với việc khi $y_i = 1$, toán hạng thứ hai của vế phải sẽ bị triệt tiêu. Khi đó, $L_i = -\log(\hat{y}_i)$. Ngược lại khi $y_i = 0$, toán hạng thứ nhất sẽ bị triệt tiêu và còn lại $L_i = -\log(1 - \hat{y}_i)$

Dánh giá về hàm L. Với $L = -\log(\hat{y}_i)$ khi $y_i = 1$:

- Hàm L giảm dần khi \hat{y}_i đi từ 1 đến 0.
- Khi mô hình dự đoán \hat{y}_i gần 1, tức giá trị dự đoán gần với giá trị thật y_i thì L nhỏ, xấp xỉ 0
- Khi mô hình dự đoán \hat{y}_i gần 0, tức giá trị dự đoán ngược lại giá trị thật y_i thì L rất lớn

Ngược lại, với $L = -\log(1 - \hat{y}_i)$ khi $y_i = 0$:

- Hàm L giảm dần khi \hat{y}_i đi từ 0 đến 1.
- Khi mô hình dự đoán \hat{y}_i gần 0, tức giá trị dự đoán gần với giá trị thật y_i thì L nhỏ, xấp xỉ 0
- Khi mô hình dự đoán \hat{y}_i gần 1, tức giá trị dự đoán ngược lại giá trị thật y_i thì L rất lớn

Từ những đánh giá trên, ta đưa ra kết luận hàm L càng nhỏ khi giá trị mô hình dự đoán càng gần với giá trị thật và sẽ càng lớn khi mô hình dự đoán càng sai. Vậy nên bài toán tìm mô hình trở thành bài toán tìm giá trị nhỏ nhất của L.

Tổng quát hóa, ta có hàm mất mát trung bình trên toàn bộ dữ liệu là:

$$J = \frac{1}{N} \sum_{i=0}^{N-1} L_i = -\frac{1}{N} \sum_{i=0}^{N-1} (y_i * \log(\hat{y}_i) + (1 - y_i) * \log(1 - \hat{y}_i)) \quad (10.3)$$

10.3.3 Quy tắc dây chuyền (chain rule)

Cho $z = f(y)$ và $y = g(x)$. Khi đó, $z = f(g(x))$ thì quy tắc dây chuyền phát biểu như sau:

$$\frac{dz}{dx} = \frac{dz}{dy} * \frac{dy}{dx}$$

Đây là một quy tắc rất hay giúp đơn giản hóa việc tính đạo hàm của các hàm hợp phức tạp. Ví dụ, cần tính đạo hàm $z(x) = (3x + 2)^2$, có thể đặt biểu thức trên thành $z(x) = f(g(x))$ với $f(x) = x^2$ và $g(x) = 3x + 2$. Áp dụng quy tắc dây chuyền, ta có:

$$\frac{dz}{dx} = \frac{dz}{dy} * \frac{dy}{dx} = \frac{d(3x + 2)^2}{d(3x + 2)} * \frac{d(3x + 2)}{dx} = 2 * (3x + 2) * 3$$

Khi đã nắm vững quy tắc dây chuyền, chúng ta sẽ sử dụng nó để áp dụng gradient descent.

10.3.4 Gradient descent

Gradient descent là một thuật toán tìm giá trị nhỏ nhất của hàm số $f(x)$ dựa trên đạo hàm. Thuật toán này được thực hiện như sau:

1. Khởi tạo giá trị $x = x_0$ tùy ý
2. Gán $x = x - \text{learning rate} * f'(x)$ với learning rate là một hằng số không âm)
3. Tính lại $f(x)$. Nếu $f(x)$ đủ nhỏ thì dừng lại, ngược lại tiếp tục bước 2

Thuật toán sẽ lặp lại bước 2 một số lần đủ lớn (100 hoặc 1000 lần tùy vào bài toán và hệ số learning rate) cho đến khi $f(x)$ đạt giá trị đủ nhỏ.

Nhận xét:

- Thuật toán hoạt động rất tốt trong trường hợp không thể tìm giá trị nhỏ nhất bằng đại số tuyến tính.

10.3. THIẾT LẬP BÀI TOÁN

- Việc quan trọng nhất của thuật toán là tính đạo hàm của hàm số theo từng biến sau đó lặp lại bước 2.

Việc lựa chọn hệ số learning rate cực kì quan trọng, có 3 trường hợp

- Nếu learning rate nhỏ: mỗi lần hàm số giảm rất ít nên cần rất nhiều lần thực hiện bước 2 để hàm số đạt giá trị nhỏ nhất.
- Nếu learning rate hợp lý: sau một số lần lặp bước 2 vừa phải thì hàm sẽ đạt giá trị đủ nhỏ.
- Nếu learning rate quá lớn: sẽ gây hiện tượng overshoot và không bao giờ đạt được giá trị nhỏ nhất của hàm.

Quay trở lại với bài toán mở đầu, với mỗi điểm $(x^{(i)}, y_i)$, có hàm $L = -(y_i * \log(\hat{y}_i) + (1 - y_i) * \log(1 - \hat{y}_i))$ trong đó $\hat{y}_i = \sigma(\omega_0 + \omega_1 * x_1^{(i)} + \omega_2 * x_2^{(i)})$ là giá trị mà mô hình dự đoán, còn y_i là giá trị thật của dữ liệu.

Áp dụng quy tắc dây chuyền, ta có:

$$\frac{dL}{d\omega_0} = \frac{dL}{d\hat{y}_i} * \frac{d\hat{y}_i}{d\omega_0} \quad (10.4)$$

$$\frac{dL}{d\omega_0} = \frac{dL}{d\hat{y}_i} * \frac{d\hat{y}_i}{d\omega_0} \quad (10.5)$$

$$\frac{dL}{d\omega_0} = \frac{dL}{d\hat{y}_i} * \frac{d\hat{y}_i}{d\omega_0} \quad (10.6)$$

Trong đó:

$$\frac{dL}{d\hat{y}_i} = -\frac{d(y_i * \log(\hat{y}_i) + (1 - y_i) * \log(1 - \hat{y}_i))}{d\hat{y}_i} = -\left(\frac{y_i}{\hat{y}_i} - \frac{1 - y_i}{1 - \hat{y}_i}\right) \quad (10.7)$$

$$\frac{d\hat{y}_i}{d\omega_0} = \frac{\sigma(\omega_0 + \omega_1 * x_1^{(i)} + \omega_2 * x_2^{(i)})}{d\omega_0} = \hat{y}_i * (1 - \hat{y}_i) \quad (10.8)$$

$$\frac{d\hat{y}_i}{d\omega_1} = \frac{\sigma(\omega_0 + \omega_1 * x_1^{(i)} + \omega_2 * x_2^{(i)})}{d\omega_1} = x_1^{(i)} * \hat{y}_i * (1 - \hat{y}_i) \quad (10.9)$$

$$\frac{d\hat{y}_i}{d\omega_2} = \frac{\sigma(\omega_0 + \omega_1 * x_1^{(i)} + \omega_2 * x_2^{(i)})}{d\omega_2} = x_2^{(i)} * \hat{y}_i * (1 - \hat{y}_i) \quad (10.10)$$

Do đó:

$$\frac{dL}{d\omega_0} = \frac{dL}{d\hat{y}_i} * \frac{d\hat{y}_i}{d\omega_0} = -\left(\frac{y_i}{\hat{y}_i} - \frac{1 - y_i}{1 - \hat{y}_i}\right) * \hat{y}_i * (1 - \hat{y}_i) = -(y_i * (1 - \hat{y}_i) - (1 - y_i) * \hat{y}_i) = \hat{y}_i - y_i \quad (10.11)$$

Tương tự:

$$\frac{dL}{d\omega_1} = x_1^{(i)} * (\hat{y}_i - y_i) \quad (10.12)$$

$$\frac{dL}{d\omega_2} = x_2^{(i)} * (\hat{y}_i - y_i) \quad (10.13)$$

Tổng quát hóa trên toàn bộ dữ liệu:

$$\frac{dJ}{d\omega_0} = \frac{1}{N} \sum_{i=0}^{N-1} (\hat{y}_i - y_i) \quad (10.14)$$

$$\frac{dJ}{d\omega_1} = \frac{1}{N} \sum_{i=0}^{N-1} x_1^{(i)} * (\hat{y}_i - y_i) \quad (10.15)$$

$$\frac{dJ}{d\omega_2} = \frac{1}{N} \sum_{i=0}^{N-1} x_2^{(i)} * (\hat{y}_i - y_i) \quad (10.16)$$

10.3.5 Quan hệ giữa xác suất và phương trình tuyến tính

Xét đường thẳng (d) có phương trình: $y = ax + b$, gọi $f = y - (ax + b)$, khi đó đường thẳng (d) luôn chia mặt phẳng tọa độ Oxy thành 2 miền, 1 miền hàm f có giá trị dương và 1 miền hàm f có giá trị âm. Các điểm trên đường thẳng (d) có giá trị f bằng 0.

Giả sử trong bài toán mở đầu lấy mốc ở chính giữa là 50%, tức là nếu học sinh mới được mô hình dự đoán có $\hat{y}_i \geq 0.5$ thì đậu, còn nhỏ hơn 0.5 thì rớt. Khi đó:

$$\begin{aligned} \hat{y}_i &\geq 0.5 \\ \Leftrightarrow \frac{1}{1 + e^{-(\omega_0 + \omega_1 * x_1^{(i)} + \omega_2 * x_2^{(i)})}} &\geq 0.5 \\ \Leftrightarrow 2 &\geq 1 + e^{-(\omega_0 + \omega_1 * x_1^{(i)} + \omega_2 * x_2^{(i)})} \\ \Leftrightarrow e^{-(\omega_0 + \omega_1 * x_1^{(i)} + \omega_2 * x_2^{(i)})} &\leq 1 = e^0 \\ \Leftrightarrow \omega_0 + \omega_1 * x_1^{(i)} + \omega_2 * x_2^{(i)} &\geq 0 \end{aligned}$$

Tương tự, $\hat{y}_i \leq 0.5 \Leftrightarrow \omega_0 + \omega_1 * x_1^{(i)} + \omega_2 * x_2^{(i)} \leq 0$.

Vậy đường thẳng $\omega_0 + \omega_1 * x_1^{(i)} + \omega_2 * x_2^{(i)}$ chính là đường phân cách giữa các điểm đậu và rớt.

Trong trường hợp ở bài toán mở đầu, nếu số lượng học sinh đậu quá đông buộc nhà trường ưu tiên nhận những học sinh có xác suất đậu lớn hơn t ($0.5 < t < 1$). Khi đó phương trình đường phân cách sẽ là: $\omega_0 + \omega_1 * x_1^{(i)} + \omega_2 * x_2^{(i)} > -\ln(\frac{1}{t} - 1)$

10.4 Đánh giá hiệu suất

Một tiêu chí tự nhiên để đánh giá hiệu suất của một mô hình phân loại (*classification model*) đó là xác suất gây ra lỗi phân loại sai (*misclassification*). Trong đó, lỗi phân loại sai xảy ra khi sự quan sát của một điểm dữ liệu thực tế thuộc về một lớp này, nhưng mô hình phân loại nó là thành viên của một lớp khác.

Một trình phân loại không có lỗi nào sẽ được coi là hoàn hảo. Tuy nhiên, đừng hy vọng có thể xây dựng các mô hình phân loại như vậy trong thế giới thực. Bởi luôn xảy ra những nguyên nhân:

- Do dữ liệu nhiễu (*noise data*)
- Do không có tất cả các thông tin cần thiết để phân loại chính xác các trường hợp.

10.5. VÍ DỤ

Dể có được ước tính trung thực về lỗi phân loại, chúng ta sử dụng ma trận phân loại (*classification matrix*) được tính toán từ những dữ liệu đã được xác nhận là hợp lệ.

- Trước tiên chúng ta phân vùng dữ liệu thành tập huấn luyện (*training set*) và tập kiểm tra (*validation set*) bằng cách chọn ngẫu nhiên các trường hợp.
- Sau đó, xây dựng một bộ phân loại bằng cách sử dụng dữ liệu trong tập huấn luyện.
- Áp dụng nó cho dữ liệu trong tập kiểm tra
- Dự đoán phân loại cho các quan sát trong tập kiểm tra.
- Cuối cùng, tóm tắt các phân loại này thành một ma trận phân loại.

Tuy nhiên, có một khả năng sẽ xảy ra khi tính toán tính chính xác của mô hình. Đó là khi dữ liệu huấn luyện và dữ liệu kiểm tra bị lệch hẳn về một phân loại, khi đó mô hình sẽ dự đoán mọi thứ đều là thành viên của lớp đó. Ví dụ như trong dữ liệu huấn luyện có 78% số lượng thư điện tử là thư rác. Vậy nếu một người huấn luyện một mô hình dự đoán tất cả mọi thư điện tử đều là thư rác, nó sẽ chính xác 78%.

Vậy nên, ta có ma trận phân loại cho N điểm dữ liệu được xây dựng như Bảng 10.1:

Bảng 10.1: Ma trận phân loại

	Dự đoán (Sai)	Dự đoán (Đúng)
Thực tế (Sai)	Dúng phủ định (TN)	Sai khẳng định (FP)
Thực tế (Đúng)	Sai phủ định (FN)	Dúng khẳng định (TP)

Khi đó, tính chính xác (*Accuracy*) của mô hình là:

$$\frac{TP + TN}{N} \quad (10.17)$$

Tính phân loại (*Misclassification*) của mô hình là:

$$\frac{FP + FN}{N} \quad (10.18)$$

Tính đúng đắn (*Precision*) của mô hình là:

$$\frac{TP}{TP + FP} \quad (10.19)$$

10.5 Ví dụ

10.5.1 Mô tả

Dể hiểu mọi thứ một cách rõ ràng hơn, từ dữ liệu huấn luyện cho ở bài toán mở đầu, ta sẽ từng bước xây dựng lại mô hình để tìm ra kết quả cụ thể.

Dầu tiên, sau khi phân tích bài toán, ta nhận thấy:

- Biến mục tiêu cần dự đoán là tỉ lệ đậu kỳ thi đầu vào của học sinh nên giá trị này luôn nằm trong khoảng $[0, 1]$.
- Nhìn chung, kết quả thu được có quy luật là tổng điểm trung bình 2 môn Văn và Toán càng cao, xác suất để học sinh đó đậu là càng lớn (giống với tính chất hàm đồng biến). Mặc dù vẫn có một số điểm ngoại lệ nhưng nó là không đáng kể.

Từ 2 phân tích trên cho thấy bài toán thỏa mãn điều kiện áp dụng mô hình hồi quy Logistic.

10.5.2 Thiết lập bài toán

Việc thực hiện tính toán bằng tay trên cơ sở dữ liệu lớn như vậy là không khả thi và tốn rất nhiều thời gian, vậy nên đòi hỏi chúng ta phải hiện thực bằng một ngôn ngữ lập trình cụ thể (như Python hoặc R) mới thu được kết quả nhanh chóng và chính xác. Tuy nhiên, để nắm vững các công thức vừa tìm hiểu, ta sẽ lấy ra một tập dữ liệu nhỏ hơn để thử vận dụng. Tập dữ liệu đó gồm thông tin như Bảng 10.2:

Văn	Toán	Đậu
35	78	0
30	44	0
36	73	0
60	86	1
79	75	1

Bảng 10.2: Bảng dữ liệu mẫu để thử tính toán công thức bằng tay

Gọi $x_1^{(i)}$, $x_2^{(i)}$ và y_i lần lượt là điểm trung bình trên thang 100 các môn Văn, Toán và kết quả (nếu đậu là 1 và ngược lại là 0) của học sinh thứ i . Theo mô hình hồi quy Logistic, ta có xác suất để một học sinh thứ i đậu kỳ thi đầu vào của trường sẽ là:

$$\hat{y}_i = \frac{1}{1 + e^{-(\omega_0 + \omega_1 * x_1^{(i)} + \omega_2 * x_2^{(i)})}}$$

Cụ thể, xác suất để học sinh thứ 0 đậu vào trường ABC sẽ là: $\hat{y}_0 = \frac{1}{1 + e^{-(\omega_0 + \omega_1 * 35 + \omega_2 * 78)}}$

Tiếp theo, để áp dụng gradient descent, ta chọn bộ ba $[\omega_0, \omega_1, \omega_2] = [0, 0, 0]$ và learning rate = 0.001. Khi đó:

$$\hat{y}_0 = \hat{y}_1 = \hat{y}_2 = \hat{y}_3 = \hat{y}_4 = \frac{1}{1 + e^0} = 0.5$$

Kết hợp bước 2 của thuật toán gradient descent và công thức (2.14), ta cập nhật lại giá trị:

$$\omega_0 = 0 - \frac{0.001}{5}[(0.5 - 0) + (0.5 - 0) + (0.5 - 0) + (0.5 - 1) + (0.5 - 1)] = -0.0001$$

Với bộ ba mới $[\omega_0, \omega_1, \omega_2] = [-0.0001, 0, 0]$, ta phải đi tính lại các giá trị \hat{y}_i và được kết quả là:

$$\hat{y}_0 = \hat{y}_1 = \hat{y}_2 = \hat{y}_3 = \hat{y}_4 = \frac{1}{1 + e^{0.0001}} \approx 0.499975$$

Kết hợp bước 2 của thuật toán gradient descent và công thức (2.15), ta cập nhật lại giá trị:

$$\omega_1 = 0 - \frac{0.001}{5} \sum_{i=0}^5 x_1^{(i)} (\hat{y}_i - y_i) \approx 0.0038$$

Tương tự, với bộ ba mới $[\omega_0, \omega_1, \omega_2] = [-0.0001, 0.0038, 0]$, ta tiếp tục đi tính lại các giá trị \hat{y}_i và được kết quả là:

$$\begin{aligned}\hat{y}_0 &= \frac{1}{1 + e^{-(0.0001+0.0038*35)}} = 0.5332 \\ \hat{y}_1 &= \frac{1}{1 + e^{-(0.0001+0.0038*30)}} = 0.5284 \\ \hat{y}_2 &= \frac{1}{1 + e^{-(0.0001+0.0038*36)}} = 0.5341 \\ \hat{y}_3 &= \frac{1}{1 + e^{-(0.0001+0.0038*60)}} = 0.5567 \\ \hat{y}_4 &= \frac{1}{1 + e^{-(0.0001+0.0038*79)}} = 0.5745\end{aligned}$$

Kết hợp bước 2 của thuật toán gradient descent và công thức (2.16), ta cập nhật lại giá trị:

$$\omega_2 = 0 - \frac{0.001}{5} \sum_{i=0}^5 x_2^{(i)} (\hat{y}_i - y_i) \approx -0.0034$$

Đến đây là xong một lần gradient descent, ta được kết quả bộ ba $[\omega_0, \omega_1, \omega_2] = [-0.0001, 0.0038, -0.0034]$. Khi đó, tính lại các giá trị \hat{y}_i và kết hợp với công thức (2.3), ta sẽ có hàm mất mát trung bình trên toàn bộ dữ liệu có công thức là:

$$J = -\frac{1}{5} [\log(1 - \hat{y}_0) + \log(1 - \hat{y}_1) + \log(1 - \hat{y}_2) + \log(\hat{y}_3) + \log(\hat{y}_4)]$$

Bằng cách thực hiện gradient descent một số lần đủ lớn để hàm J trên nhận kết quả đủ nhỏ, ta sẽ thu được bộ ba $[\omega_0, \omega_1, \omega_2]$ tương đối chính xác để sử dụng làm công thức tính xác suất đậu của một học sinh thứ i bất kỳ.

10.5.3 Hiệu thực bài toán bằng ngôn ngữ lập trình Python

Người đọc có thể lấy dữ liệu từ file *data.csv* và tham khảo mã nguồn trong file *testLogisticRegression.ipynb* tại: <https://github.com/ToDuyHung/testLogisticRegression>

Bước 1: Sau khi khai báo các thư viện cần thiết và đọc dữ liệu, ta xây dựng hàm sigmoid theo biến z với công thức $\frac{1}{1 + e^{-z}}$ để áp dụng khi tính xác suất đậu của học sinh theo công thức:

$$\hat{y}_i = \frac{1}{1 + e^{-(\omega_0 + \omega_1 * x_1^{(i)} + \omega_2 * x_2^{(i)})}}$$

trong đó, $x_1^{(i)}$ và $x_2^{(i)}$ lần lượt là điểm thi môn Văn và Toán của học sinh thứ i.

Nhiệm vụ của chúng ta bây giờ là đi tìm bộ 3 giá trị $[\omega_0, \omega_1, \omega_2]$ để có thể tính được xác suất đậu \hat{y}_i của mỗi học sinh thứ i bất kỳ.

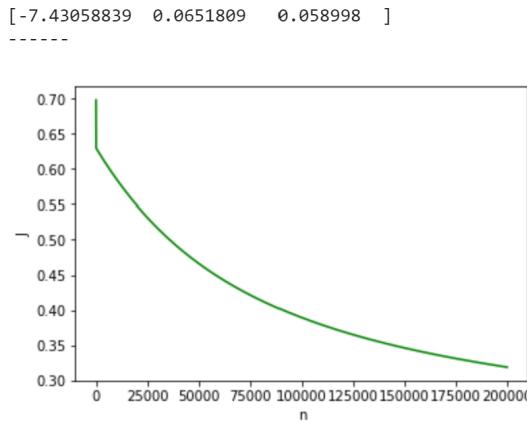
Bước 2: ta xây dựng được hàm mất mát trung bình J theo công thức:

$$J = \frac{1}{100} \sum_{i=0}^{99} \hat{y}_i = -\frac{1}{100} \sum_{i=0}^{99} (y_i * \log(\hat{y}_i) + (1 - y_i) * \log(1 - \hat{y}_i))$$

Bước 3: Sử dụng quy tắc dây chuyền, ta được:

$$\begin{aligned}\frac{dJ}{d\omega_1} &= \frac{1}{100} \sum_{i=0}^{99} (\hat{y}_i - y_i) \\ \frac{dJ}{d\omega_2} &= \frac{1}{100} \sum_{i=0}^{99} x_1^{(i)} * (\hat{y}_i - y_i) \\ \frac{dJ}{d\omega_3} &= \frac{1}{100} \sum_{i=0}^{99} x_2^{(i)} * (\hat{y}_i - y_i)\end{aligned}$$

Bước 4: Áp dụng gradient descent, chọn learning rate bằng 0.001 và số lần thực hiện thuật toán là 200000 lần. Ta được kết quả như Hình 10.3 :



Hình 10.3: Phác thảo đồ thị giá trị của J sau mỗi lần thực hiện gradient descent

Từ Hình 10.3, ta thấy hàm mất mát J đang tiến dần về giá trị đủ nhỏ xấp xỉ với 0.3. Bên cạnh đó, ta thu được kết quả là bộ ba $[\omega_0, \omega_1, \omega_2]$ có giá trị tương đối tốt lần lượt bằng -7.4306, 0.0652, 0.0590.

Khi đó, xác suất để một học sinh đậu là:

$$\hat{y}_i = \frac{1}{1 + e^{(-7.4306 + 0.0652 * x_1^{(i)} + 0.0590 * x_2^{(i)})}}$$

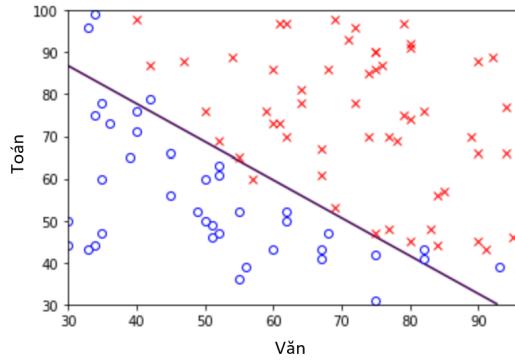
10.5.4 Kiểm tra kết quả

Thử kiểm tra lại kết quả của mô hình vừa xây dựng được với kết quả thực tế của một số học sinh trong danh sách. Ta được kết quả như Bảng 10.3:

STT	Văn	Toán	Xác suất đậu	Thực tế
88	79	97	0.97	1
89	52	61	0.39	0
90	94	77	0.96	1
91	90	88	0.97	1
92	55	36	0.15	0

Bảng 10.3: So sánh xác suất tính được với kết quả thực tế

Cuối cùng, vẽ đường phân cách đậu và rớt trên mặt phẳng hai chiều Oxy có phương trình $-7.4306 + 0.0652 * x_1^{(i)} + 0.0590 * x_2^{(i)} = 0$, như Hình 10.4



Hình 10.4: Đường phân cách các điểm dữ liệu đậu và rớt

Trong đó các điểm dữ liệu hình chữ x màu đỏ đại diện cho các học sinh có khả năng đậu kỳ thi đậu vào cao, và ngược lại các điểm dữ liệu hình tròn màu xanh đại diện cho các học sinh có khả năng đậu không cao.

10.6 Ứng dụng

Hồi quy Logistic được ứng dụng rộng rãi trong phân tích dự báo, hay sâu hơn là trong lĩnh vực học máy (*machine learning*). Kỹ thuật này xuất hiện nhiều trong các phần mềm thống kê và khai phá dữ liệu, giúp người dùng tìm hiểu mối quan hệ giữa biến mục tiêu là biến định tính và một hay nhiều biến độc lập thông qua thiết lập phương trình hồi quy logistic.

Ứng dụng hồi quy logistic trong việc xây dựng mô hình dự báo đối với các công ty ngày nay như là một phương pháp tạo nên sự khác biệt và lợi thế cạnh tranh. Vì đơn giản các mô hình dự báo sẽ giúp họ khai phá các mối quan hệ, những yếu tố sẽ tác động lên doanh thu, lợi nhuận trong tương lai, thông qua tìm hiểu hành vi của khách hàng, từ đó ra quyết định, chiến lược hiệu quả hơn. Một nhóm phân tích dữ liệu của một nhà máy sản xuất có thể sử dụng hồi quy logistic để dự báo khả năng hư hỏng của các thành phần máy móc thiết bị dựa trên khoảng thời gian chúng được lưu trữ trong kho. Với kết quả có được từ quá trình phân tích, nhà máy có thể đưa ra kế hoạch bảo dưỡng, lắp đặt hợp lý. Còn rất nhiều ứng dụng khác của hồi quy logistic trong kinh doanh khác mà điển hình như công ty có thể sử dụng để dự báo khả năng khách hàng rời dịch vụ, phân khúc khách hàng theo nhóm sản phẩm mục tiêu dựa trên đặc điểm mua hàng, thông tin cá nhân. Trong lĩnh vực y tế hồi quy logistic có thể được sử dụng để dự đoán khả năng mắc bệnh của một nhóm dân số nhất định để áp dụng các biện pháp phòng ngừa.

Một số ứng dụng tiêu biểu của hồi quy Logistic:

- Dự báo hay phân loại thư điện tử có phải spam hay không spam.
- Dự báo khả năng rời dịch vụ của khách hàng.
- Dự báo tình trạng khối u ung thư là ác tính hay lành tính trong y học.
- Dự báo khả năng khách hàng sẽ mua sản phẩm bất kỳ, hay đăng ký dịch vụ.
- Dự báo khả năng trả nợ của khách hàng.
- Dự đoán giao dịch ngân hàng có phải gian lận hay không.
- Dự đoán khoản đầu tư có sinh lời hay không.

10.6.1 Ứng dụng hồi quy Logistic vào dự án phần mềm

10.6.2 Mô tả

Thư rác, hay còn gọi là *spam email*, là những nội dung không mong muốn được gửi hàng loạt cho các người dùng với mục đích quảng cáo hoặc tấn công. Nghe có vẻ mâu thuẫn khi dù chúng ta đang sống trong một thời đại mà công nghệ thông tin đang vô cùng phát triển, vẫn đề bảo mật và riêng tư của người dùng lại luôn bị đặt ở mức dễ bị xâm phạm. Vậy nên một trong những ứng dụng phổ biến nhất của hồi quy Logistic trong những dự án thực tế đó là phân loại thư điện tử. Mục đích của ứng dụng này là nhắm loại bỏ những thư điện tử "rác" một cách tự động.

10.6.3 Phân tích yêu cầu (requirement) của ứng dụng

- Dữ liệu đầu vào: là nội dung của một thư điện tử bất kỳ (loại dữ liệu: String)
- Dữ liệu đầu ra: phân loại được nội dung đó là thư rác (spam) hay không (ham)

10.6.4 Thiết kế mô hình phân loại nội dung

- Tập dữ liệu huấn luyện: Bài báo cáo này tham khảo tập dữ liệu từ file *SMSSpamCollection* tại địa chỉ: <https://archive.ics.uci.edu/ml/machine-learning-databases/00228/>. Tập dữ liệu đó bao gồm 5572 nội dung của các thư điện tử khác nhau đã được phân loại sẵn thành spam hoặc ham như Hình. 10.5

10.6. ỨNG DỤNG

```
4197 Sorry, it's a lot of friend-of-a-friend stuff, ...
168 Hi frnd, which is best way to avoid missunders...
5455 Wishing you a beautiful day. Each moment revea...
4920 Re your call; You didn't see my facebook huh?
2944 No message..no responce..what happend?
...
4112 URGENT! Your Mobile number has been awarded a ...
5049 Yeah so basically any time next week you can g...
4439 Nothing will ever be easy. But don't be lookin...
2422 Err... Cud do. I'm going to at 8pm. I haven't...
3263 O shore are you takin the bus
Name: 1, Length: 4179, dtype: object
197 Did u got that persons story
4390 K I'm ready, &lt;#&gt; ?
82 Ok i am on the way to home hi hi
3315 Oh gei. That happend to me in tron. Maybe ill ...
1958 Take something for pain. If it moves however t...
...
4217 Er mw im filled tuth is aight
1566 The &lt;#&gt; g that i saw a few days ago, th...
1239 Dear relieved of westonzoyland, all going to p...
3177 Havent still waitin as usual... Ü come back sc...
3675 You have won a Nokia 7250i. This is what you g...
```

Hình 10.5: Tập dữ liệu huấn luyện

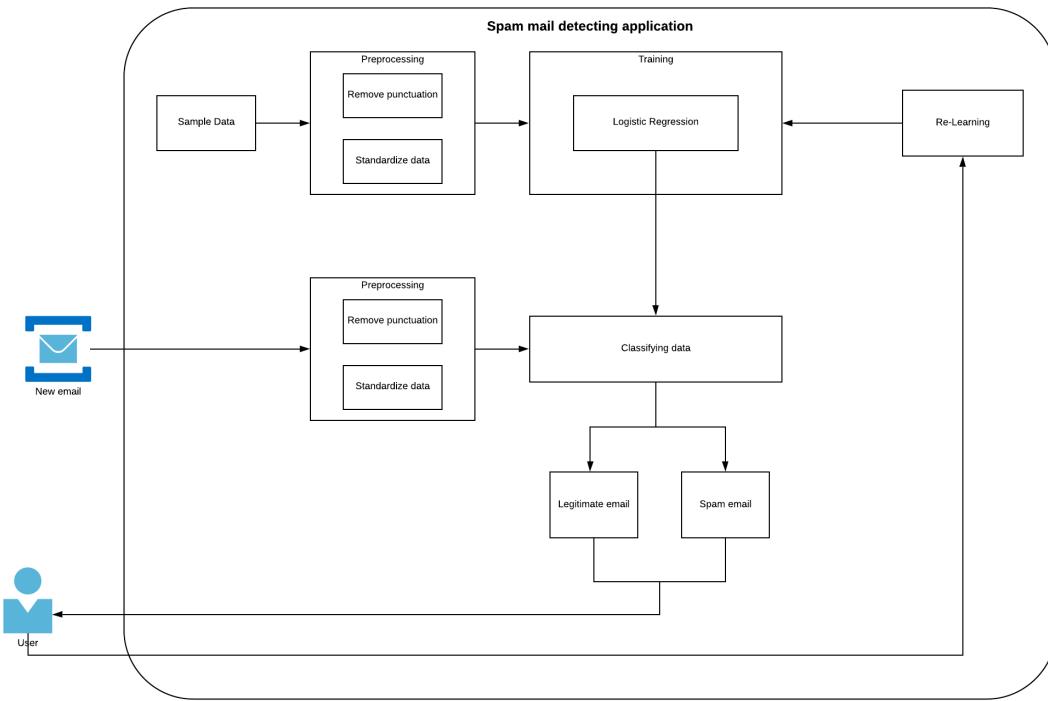
- Tiền xử lý dữ liệu (preprocessing): Ta tìm cách chuẩn hóa hóa dữ liệu đầu vào từ dạng chuỗi thành dạng số thực. Trong bài toán này, ta có thể sử dụng phương thức TfidfVectorizer được cung cấp bởi thư viện Scikit-learn. Kết quả thu được sau khi chuẩn hóa dữ liệu như Hình. 10.6
- Áp dụng mô hình hồi quy Logistic: sử dụng phương thức LogisticRegression được cung cấp bởi thư viện Scikit-learn

Với ý tưởng đó, người đọc hãy thử xây dựng lại mô hình hoàn chỉnh cho chức năng phân loại nội dung như trên.

(0, 1534)	0.22791011889690926
(0, 7148)	0.26748057546808424
(0, 7258)	0.20309114399370096
(0, 3216)	0.257943735240714
(0, 769)	0.3324115436487556
(0, 6611)	0.10791720639151749
(0, 6507)	0.23216077077780783
(0, 6727)	0.2655977445552242
(0, 723)	0.18460925638126702
(0, 4730)	0.13926555670871194
(0, 3797)	0.1495488987514727
(0, 6363)	0.23691259530073153
(0, 2938)	0.45786307495073136
(0, 4778)	0.2678408583138424
(0, 4120)	0.23951632957965968
(0, 3679)	0.1281668228915897
(0, 6155)	0.18709821425599446
(1, 4820)	0.20760319985907505
(1, 1277)	0.3799686590086109
(1, 4885)	0.21557397679720508
(1, 7308)	0.31234741691321016
(1, 4412)	0.3986669888947816
(1, 1122)	0.3799686590086109
(1, 7176)	0.23031412515300648
(1, 1286)	0.2693879063207318

Hình 10.6: Dữ liệu sau khi chuẩn hóa

10.6.5 Kiến trúc phần mềm



Hình 10.7: Kiến trúc chương trình phân loại thư điện tử

10.6.6 Sequence diagram

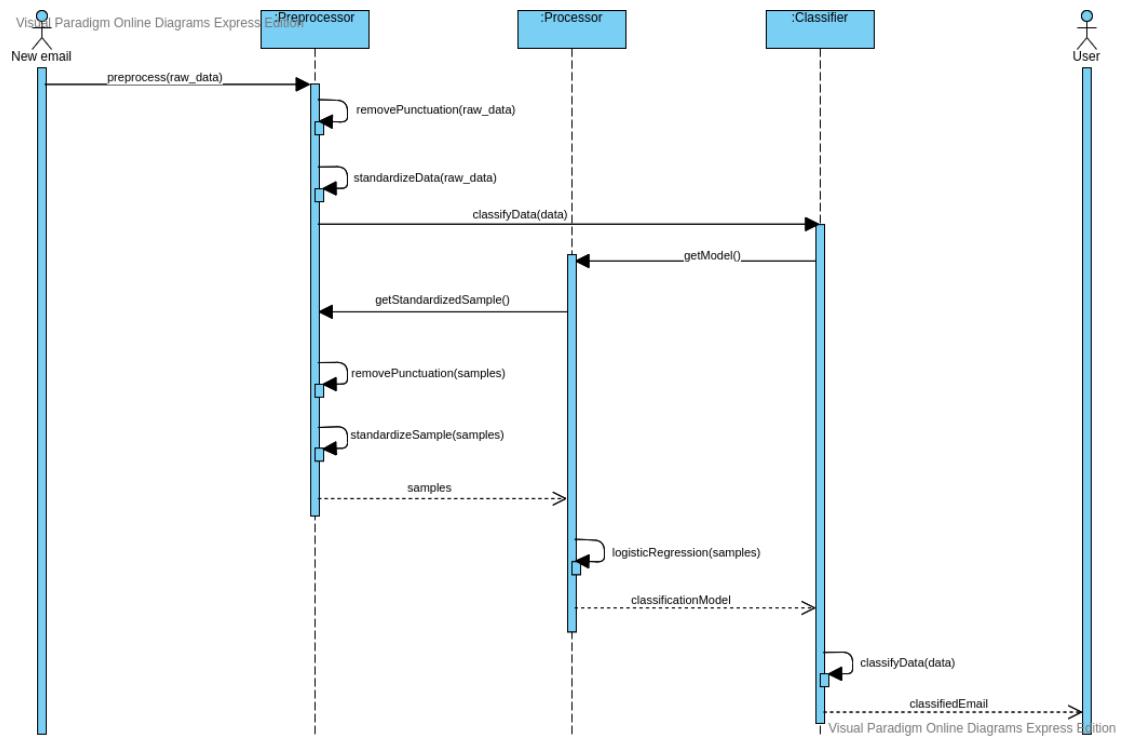
10.7 Bài tập

- Tại sao hàm sigmoid lại được chọn để sử dụng trong bài toán hồi quy logistic ? Dùng quy tắc dây chuyền tính đạo hàm của hàm sigmoid và hàm mất mát trong hồi quy logistic với từng biến. Giả sử điều chỉnh 1 số tham số như learning rate (tăng hoặc giảm), khi ấy hàm mất mát sẽ thay đổi thế nào ?
- Cho hàm số $f(x) = x^2$. Dẽ dàng nhận thấy giá trị nhỏ nhất của hàm $f(x)$ là bằng 0 khi $x = 0$. Tuy nhiên hãy thử áp dụng gradient descent để tìm giá trị nhỏ nhất của $f(x)$ với x ban đầu là 2 và learning rate là 0.001.
- Sau khi xây dựng và hiện thực một mô hình phân loại bằng hồi quy Logistic. Ta được kết quả như Bảng 10.4. Hãy tính toán tính chính xác, tính phân loại sai và tính đúng đắn của mô hình trên.

	Dự đoán thuộc lớp 1	Dự đoán thuộc lớp 0
Thực tế thuộc lớp 1	301	69
Thực tế thuộc lớp 0	16	3784

Bảng 10.4: Kết quả dự đoán và thực tế của tập dữ liệu sau khi áp dụng mô hình

CHƯƠNG 10. LOGISTIC REGRESSION



Hình 10.8: Sequence diagram cho quá trình phân loại thư điện tử

4. Biểu diễn dưới dạng ma trận cho bài toán hồi quy logistic

Chương 11

Giải thuật DBSCAN

11.1 Bài toán đặt ra

Khai thác dữ liệu là quá trình tìm kiếm và rút trích những thông tin tiềm ẩn có giá trị, hữu ích từ một khối lượng dữ liệu khá lớn ban đầu. Trong lĩnh vực giao thông vận tải, việc sử dụng kết quả từ việc phân tích dữ liệu từ các thiết bị giám sát hành trình, dữ liệu xe con di động (FCD) và dữ liệu điện thoại trực tuyến (FPD) đã đem lại những hiệu quả rõ rệt trong vấn đề giám sát và quản lý giao thông.

Bài toán đặt ra đó là phân tích hay khai thác dữ liệu hành trình thu thập được từ các thiết bị thu GPS, gọi tắt là dữ liệu GPS, để trích xuất những thông tin có giá trị và hữu ích về tình trạng ùn tắc giao thông của mạng lưới giao thông trong đô thị. Việc khai thác dữ liệu GPS mang lại khá nhiều ứng dụng hữu ích, như: dự báo tắc nghẽn giao thông, khai thác địa điểm quan trọng và lộ trình thông dụng từ dữ liệu GPS, quy hoạch sử dụng các lộ trình tối ưu. Vậy để phân tích dữ liệu GPS, giải thuật phân cụm dữ liệu, trong đó giải thuật DBSCAN với khả năng phân vùng với kích thước lớn mà không cần quan tâm hình dạng và số vùng cần chia là 1 thuật toán tối ưu.

11.2 Sơ lược về phân cụm dữ liệu

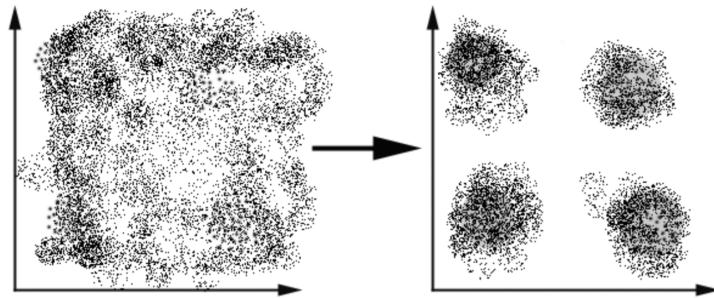
11.2.1 Khái niệm phân cụm dữ liệu

Phân cụm / gom cụm dữ liệu (Clustering / Cluster analysis) là việc nhóm dữ liệu không có nhãn sao cho các dữ liệu cùng cụm có các tính chất tương tự nhau và dữ liệu của hai nhóm khác nhau phải rất khác nhau.

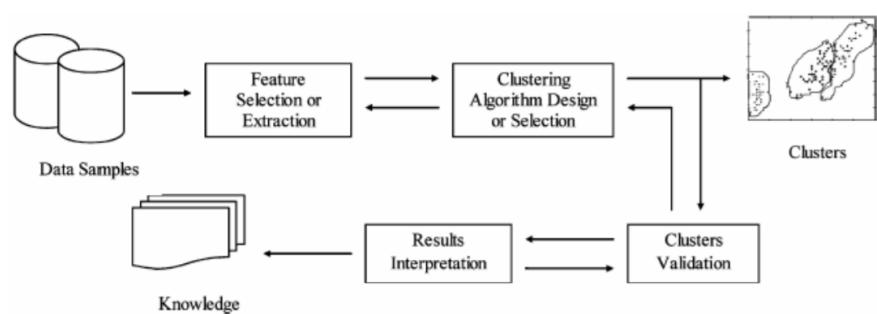
Ở hình trên, khi áp dụng phương pháp phân cụm dù thủ công hay tự động, sẽ thu được các cụm trong đó các phần tử "gần nhau" hay là "tương tự" thì chúng thuộc về các cụm khác nhau.

Bằng việc quan sát dữ liệu thu thập được (thể hiện của dữ liệu), người ta giả định rằng các dữ liệu này tuân theo một quy luật nào đó (bản chất của dữ liệu). Kế đến xây dựng mô hình dựa trên quy luật giả định, thiết lập các phương trình, tìm các tham số, sao cho mô hình phù hợp với dữ liệu ta thu thập được nhất. Mô hình thu được có thể được dùng để hiểu dữ liệu, phát hiện tri thức mới từ dữ liệu hoặc cũng có thể dùng để giải thích dữ liệu mới có liên quan.

Phân cụm dữ liệu phải giải quyết đó là hầu hết các dữ liệu chứa dữ liệu "nhiều" (noise) do các bước lấy mẫu chưa đầy đủ hoặc thiếu chính xác, do đó cần phải lập kế hoạch chiến lược ngay tại bước tiền xử lý dữ liệu để loại bỏ "nhiều" trước khi đưa vào giai đoạn tiếp theo. Khái niệm



Hình 11.1: Minh họa phân cụm dữ liệu.



Hình 11.2: Quá trình phân cụm dữ liệu.

"nhiều" được hiểu là thông tin về các đối tượng chưa chính xác, hoặc là khuyết thiếu thông tin về một số thuộc tính.

Một trong các kỹ thuật xử lý nhiễu phổ biến là việc thay thế giá trị của các thuộc tính của đối tượng "nhiều" bằng giá trị thuộc tính tương ứng của đối tượng dữ liệu gần nhất. Do vậy, phân cụm dữ liệu cần giải quyết một số vấn đề sau:

- Xây dựng hàm tính độ đo tương tự
- Xây dựng tập các tiêu chí phân cụm
- Thiết lập các cấu trúc dữ liệu cho cụm dữ liệu
- Xây dựng thuật toán phân cụm dữ liệu
- Xây dựng hệ thống phân tích và đánh giá kết quả

Ngày nay, chưa có một phương pháp phân cụm nào có thể giải quyết trọn vẹn cho tất cả các dạng cấu trúc cụm dữ liệu.

11.2.2 Các giải thuật gom cụm

Phân loại các giải thuật phân cụm dữ liệu tiêu biểu:

- Phân hoạch (partitioning): các phân hoạch được tạo ra và đánh giá theo một tiêu chí nào đó. Đây là các thuật toán được áp dụng nhiều trong thực tế như k-means, PAM (Partitioning Around Medoids), CLARA (Clustering LARge Applications), CLARANS (Clustering LARge ApplicatioNS).
- Phân cấp (hierarchical): phân rã tập dữ liệu/dối tượng có thứ tự phân cấp theo một tiêu chí nào đó. Ví dụ như: BIRCH (Balanced Iterative Reducing and Clustering Using Hierarchies), CURE(Clustering Using REpresentatives).
- Dựa trên mật độ (density-based): dựa trên connectivity and density functions. Ví dụ như: DBSCAN (Density - Based Spatial Clustering of Applications with noise), OPTICS (Ordering Points To Identify the Clustering Structure), DENCLUE (DENsity - Based CLUstErInG).
- Dựa trên lưới (grid-based): dựa trên a multiple-level granularity structure. Ví dụ như: STING và CLIQUE
- Dựa trên mô hình (model-based): một mô hình giả thuyết được đưa ra cho mỗi cụm; sau đó hiệu chỉnh các thông số để mô hình phù hợp với cụm dữ liệu/dối tượng nhất.

Trong phần tiếp theo, ta sẽ tập trung nghiên cứu chi tiết về giải thuật DBSCAN, một giải thuật tiêu biểu của phân cụm dữ liệu dựa trên mật độ.

11.3 Giải thuật DBSCAN

11.3.1 Khái niệm

Giải thuật Density-Based Spatial Clustering of Applications with Noise – một thuật toán phân cụm dựa trên mật độ (DBSCAN) được Ester, Kriegel và Sander đề xuất năm 1996 khi nghiên

11.3. GIẢI THUẬT DBSCAN

cứu các thuật toán gom cụm dữ liệu không gian dựa trên định nghĩa cụm là tập tối đa các điểm liên thông về mật độ. Giải thuật DBSCAN phát hiện các cụm có hình dạng tùy ý, khả năng phát hiện nhiều tốt. DBSCAN thực hiện tốt trên không gian nhiều chiều; thích hợp với cơ sở dữ liệu có mật độ phân bố dày đặc kể cả có phần tử nhiễu.



Hình 11.3: Dạng cụm dữ liệu được khám phá bởi giải thuật DBSCAN.

Quan sát 3 tập dữ liệu điểm mẫu ở Hình 11.3, chúng ta dễ dàng nhận ra các cụm điểm và các điểm nhiễu.

Ý tưởng chính để phát hiện ra các cụm của giải thuật DBSCAN là bên trong mỗi cụm luôn tồn tại một mật độ cao hơn bên ngoài cụm. Hơn nữa, mật độ ở những vùng nhiễu thì thấp hơn mật độ bên trong của bất kỳ cụm nào. Trong mỗi cụm phải xác định bán kính vùng lân cận (Eps) và số lượng điểm tối thiểu trong vùng lân cận của một điểm trong cụm ($MinPts$). Hình dạng vùng lân cận của một điểm được xác định dựa vào việc chọn hàm khoảng cách giữa hai điểm p và q , ký hiệu là $dist(p, q)$. Ví dụ, nếu dùng khoảng cách Manhattan trong không gian 2D thì hình dạng vùng lân cận là hình chữ nhật.

11.3.2 Định nghĩa sử dụng trong giải thuật DBSCAN

Xét một tập hợp các điểm cần phân cụm. Gọi ε là độ dài bán kính của vùng lân cận của một điểm. Trong bài toán phân cụm DBSCAN, các điểm được phân thành ba loại là điểm cốt lõi (**core point**), điểm tiếp cận được và điểm nhiễu (**noise**). Ta có một số khái niệm:

- Một điểm p là **core point** nếu có ít nhất **minPts** điểm nằm trong vùng lân cận của nó với bán kính ε (kể cả chính nó).
- Một điểm q là gần kề với p nếu điểm q nằm trong vùng lân cận của **core point** p . Một điểm chỉ có thể gần kề với **core point**.
- Một điểm q là tiếp cận được với p nếu tồn tại đường đi p_1, \dots, p_n với $p_1 = p$ và $p_n = q$, với mọi p_{i+1} là gần kề p_i . Cần chú ý tất cả các điểm trên (q có thể ngoại lệ) đều là **core point**.

Vậy, nếu p là **core point**, nó sẽ tạo thành một cụm với tất cả các điểm khác tiếp cận được với nó. Mỗi cụm chứa ít nhất một **core point**. Các điểm không phải là core point có thể nằm trong cụm, nhưng không thể tiếp cận các điểm khác, vì vậy chúng tạo thành "biên" của cụm.

Khả năng tiếp cận không là quan hệ đối xứng: chỉ **core point** mới có thể tiếp cận các điểm không là core point, và điều ngược lại không đúng, vậy nên điểm không là core point có thể được tiếp cận, nhưng không thể tiếp cận điểm khác. Vì vậy, cần có một khái niệm nữa để mở

rộng các cụm tìm thấy. Hai điểm p và q là **density-connect** nếu có một điểm o sao cho cả p và q đều có thể tiếp cận được nó. Khái niệm này có tính đối xứng.

Một cụm sẽ thỏa mãn hai tính chất:

1. Tất cả các điểm trong cụm sẽ là **density-connect**.
2. Nếu một điểm có thể được tiếp cận và có **density-connect** với điểm nào đó trong cụm, thì nó cũng thuộc cụm đó.

11.3.3 Mã giả giải thuật DBSCAN

Giải thuật DBSCAN có thể được mô tả bởi các bước:

1. Tìm các điểm lân cận nằm trong khoảng ε của tất cả các điểm, và xác định các **core point** là các điểm có nhiều hơn **minPts** các điểm lân cận.
2. Tìm các thành phần liên thông giữa các **core points** trên đồ thị các điểm lân cận, bỏ qua tất cả các điểm không là **core point**.
3. Gán nhãn cho các điểm không là **core point** theo một cụm nào đó nếu cụm đó nằm trong khoảng ε , ngược lại, gán nhãn cho nó là **noise**.

Ngoài ra, ta có thể cụ thể hóa giải thuật trên bằng mã giả:

```
DBSCAN(DB, distFunc, eps, minPts) {
    C := 0
    for each point P in database DB {
        if label(P) != undefined then continue
        Neighbors N := RangeQuery(DB, distFunc, P, eps)
        if |N| < minPts then {
            label(P) := Noise
            continue
        }
        C := C + 1
        label(P) := C
        SeedSet S := N \ {P}
        for each point Q in S {
            if label(Q) = Noise then label(Q) := C
            if label(Q) != undefined then continue
            label(Q) := C
            Neighbors N := RangeQuery(DB,
                distFunc, Q, eps)
            if |N| >= minPts then {
                S := S ∪ N
            }
        }
    }
}
```

Với hàm `RangeQuery(DB, distFunc, Q, eps)` có thể được hiện thực bằng `linear scan`:

11.3. GIẢI THUẬT DBSCAN

```

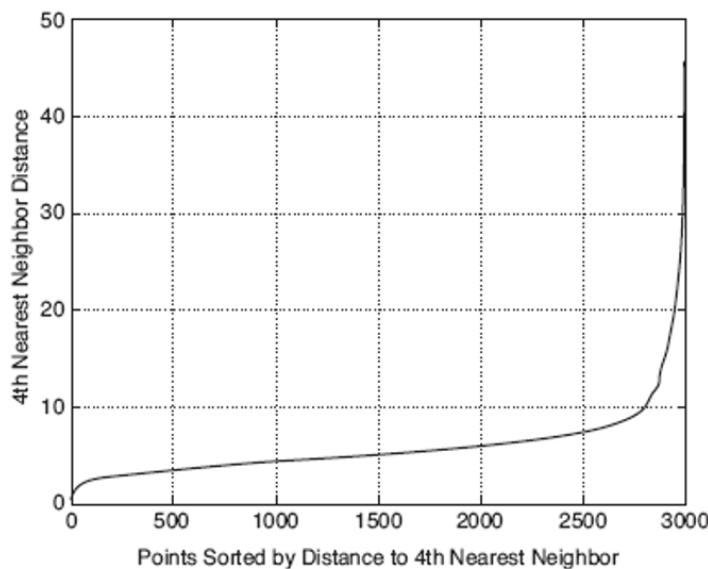
RangeQuery(DB, distFunc, Q, eps) {
    Neighbors N := empty list
    for each point P in database DB {
        if distFunc(Q, P) <= eps then {
            N := N U {P}
        }
    }
    return N
}

```

11.3.4 Xác định thông số Eps and MinPts

Thông số *Eps* và *MinPts* cho thuật toán DBSCAN có thể được xác định thông qua thuật toán heuristics. Thuật toán này dựa trên 2 quan sát sau:

1. Gọi d là khoảng cách giữa đối tượng p và đối tượng gần nhất thứ k thì vùng lân cận d của đối tượng p chứa $k+1$ đối tượng (hoặc nhiều hơn $k+1$ đối tượng khi nhiều đối tượng có cùng khoảng cách đến p).
2. Thay đổi giá trị k không dồn đến thay đổi lớn giá trị của d trừ khi k đối tượng này cùng nằm xấp xỉ trên một đường thẳng. Với giá trị k cho trước, hàm $k\text{-dist}$ là khoảng cách từ một đối tượng đến đối tượng gần nhất thứ k .



Hình 11.4: Đồ thị sorted 4-dist

Tạo đồ thị *sorted k-dist* bằng cách sắp xếp các đối tượng theo giá trị $k\text{dist}$ giảm dần như hình 11.4. Nếu chọn một đối tượng bất kỳ p , đặt thông số *Eps* là $k\text{dist}(p)$ và *MinPts* là k thì:

- Các đối tượng có khoảng cách với p nhỏ hơn hoặc bằng giá trị $k\text{dist}$ sẽ thuộc về cụm tạo bởi đối tượng p .

- Nếu tìm được đối tượng ngưỡng với giá trị k-dist lớn nhất ở trong cụm mỏng nhất của D, ta sẽ tìm được giá trị thông số mong muốn. Đối tượng ngưỡng này là đối tượng đầu tiên trong vùng lõm đầu tiên của đồ thị sorted k-dist.
- Tất cả các đối tượng với giá trị k-dist cao hơn (bên trái đối tượng ngưỡng) được xem là nhiễu.
- Các đối tượng còn lại (bên phải đối tượng ngưỡng) sẽ thuộc về một cụm nào đó.

11.3.5 Độ phức tạp

DBSCAN có thể đi qua nhiều lần mỗi điểm trong tập dữ liệu. Tuy nhiên, trong thực tế, độ phức tạp về thời gian chỉ xoay quanh số lần thực hiện `regionQuery`. DBSCAN thực hiện `regionQuery` mỗi khi đi qua một điểm, và nếu cấu trúc đánh chỉ mục được sử dụng để phân chia lân cận có độ phức tạp là $O(\log n)$, độ phức tạp về thời gian trung bình của thuật toán sẽ là $O(n \log n)$ (nếu ε được chọn phù hợp). Nếu không dùng cấu trúc đánh chỉ mục để giảm thời gian, hoặc ε quá lớn, độ phức tạp về thời gian sẽ là $O(n^2)$.

Ma trận khoảng cách với kích cỡ $(n^2 - n)/2$ có thể được sử dụng để tránh tính toán lại khoảng cách, nhưng sẽ tốn $O(n^2)$ bộ nhớ, trong khi không sử dụng ma trận chỉ tốn $O(n)$ bộ nhớ.

11.3.6 Bài toán ví dụ về Thuật Toán DBSCAN

Thuật toán DBSCAN (Density-Based Spatial Clustering of Applications with Noise) là một phương pháp phân cụm dữ liệu dựa trên mật độ. Thuật toán này có thể phát hiện các cụm có hình dạng bất kỳ và có khả năng xử lý các điểm nhiễu (noise) mà không cần biết trước số lượng cụm.

Giả sử bạn có bộ dữ liệu gồm các điểm trong không gian 2D như bảng 11.3.6:

Điểm	Tọa độ
A	(1, 2)
B	(2, 2)
C	(3, 3)
D	(8, 7)
E	(8, 8)
F	(25, 80)
G	(24, 79)

Tham số DBSCAN:

- Eps = 2: Bán kính tìm kiếm xung quanh mỗi điểm.
- MinPts = 2: Mỗi điểm cần có ít nhất 2 điểm lân cận trong bán kính Eps để được coi là điểm lõi.

Khoảng Cách Giữa Các Điểm

Khoảng cách giữa hai điểm trong không gian 2D được tính bằng công thức Euclidean:

$$d(P_1, P_2) = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

Với các điểm trong bảng 11.3.6, ta tính khoảng cách giữa tất cả các cặp điểm:

Điểm	A	B	C	D	E	F	G
A	0	1	2.24	9.22	9.22	25.70	24.83
B	1	0	1.41	7.81	7.81	24.54	23.69
C	2.24	1.41	0	6.40	6.40	23.82	22.98
D	9.22	7.81	6.40	0	1.41	17.75	16.97
E	9.22	7.81	6.40	1.41	0	17.76	16.99
F	25.70	24.54	23.82	17.75	17.76	0	1.41
G	24.83	23.69	22.98	16.97	16.99	1.41	0

Xác Định Các Điểm Lân Cận

Dựa trên bán kính $\text{Eps} = 2$, ta xác định các điểm lân cận của mỗi điểm trong bộ dữ liệu.

- Điểm A có các điểm lân cận trong bán kính Eps = 2: **B, C**.
- Điểm B có các điểm lân cận trong bán kính Eps = 2: **A, C**.
- Điểm C có các điểm lân cận trong bán kính Eps = 2: **A, B**.
- Điểm D có các điểm lân cận trong bán kính Eps = 2: **E**.
- Điểm E có các điểm lân cận trong bán kính Eps = 2: **D**.
- Điểm F và G không có điểm nào trong bán kính Eps = 2.

Xác Định Các Loại Điểm

Dựa trên số lượng điểm lân cận trong bán kính Eps, ta phân loại các điểm:

- A, B, C là điểm lõi vì mỗi điểm này có ít nhất 2 điểm lân cận trong bán kính Eps.
- D, E là điểm biên.
- F và G là điểm nhiễu.

Kết Quả Phân Cụm

Sau khi áp dụng DBSCAN, kết quả phân cụm như sau:

- Cụm 1: A, B, C (Ba điểm này là điểm lõi và có thể kết hợp với nhau để tạo thành một cụm).
- Cụm 2: D, E (Hai điểm này là điểm biên, có một điểm lõi trong bán kính Eps).
- Điểm nhiễu: F, G (Không đủ điểm lân cận trong bán kính Eps để tham gia vào cụm).

11.3.7 Ưu điểm và hạn chế

11.3.7.1 Ưu điểm

1. DBSCAN không cần biết trước số cụm sẽ xử lý.
2. DBSCAN có thể tìm ra được các hình dáng cụ thể của cụm. Ngoài ra, nó còn có thể phân cụm được một cụm bao quanh cụm khác mà không liên thông nhau.
3. Nhờ có **minPts**, giải thuật giảm thiểu được trường hợp nhầm hai cụm khi có một đường các điểm nối với nhau.

4. DBSCAN có thể nhận dạng **noise**, và không bị các điểm ngoại lai gân nhiễu.
5. DBSCAN chỉ cần hai điều kiện đầu vào và hầu như không lệ thuộc thứ tự các điểm trong dữ liệu.
6. Có thể sử dụng các cấu trúc dữ liệu để tăng tốc tìm kiếm, vd: **R* tree**,...
7. Các thông số **minPts** và ε có thể được thiết đặt trước hiệu quả nếu hiểu rõ dữ liệu đầu vào.

11.3.7.2 Hạn chế

1. DBSCAN không hoàn toàn phân chia cụ thể các cụm, một điểm không là core point nằm giữa hai cụm có thể thuộc một trong hai cụm tùy vào thứ tự dữ liệu. Hạn chế này thường rất ít ảnh hưởng tới kết quả phân cụm.
2. Chất lượng kết quả phân cụm phụ thuộc vào cách tính khoảng cách trong hàm **regionQuery(P, ε)**. Thông thường, người ta sử dụng khoảng cách Euclid cho thuật toán. Tuy nhiên, với các dữ liệu quá nhiều chiều, khoảng cách Euclid không còn hiệu quả, gây khó khăn khi xác định ε .
3. DBSCAN không thể phân loại hiệu quả nếu dữ liệu có các cụm có khoảng mật độ chênh lệch quá cao, vì các cặp **minPts**, ε không thể được lựa chọn phù hợp cho tất cả các cụm.
4. Thuật toán phụ thuộc nhiều vào ε , vì vậy rất khó khăn để chọn được ε phù hợp cho tập dữ liệu chưa được tìm hiểu rõ.

11.4 Ứng dụng vào dự án thực tế

Ứng dụng: Dự đoán và kiểm tra những điểm ùn tắc giao thông trong thành phố.

Ùn tắc giao thông trong thành phố luôn là vấn đề nhức nhối và nan giải đối với mọi thành phố trên thế giới. Ứng dụng này sẽ giúp cho người dân có thể biết được những điểm giao thông nào lưu lượng xe đông, di chuyển chậm để có thể chọn lựa được tuyến đường phù hợp. Ngoài ra ứng dụng này còn giúp nhà quản lý, quy hoạch đô thị có được thống kê đầy đủ về những điểm ùn tắc giao thông, từ đó đưa ra quyết định quy hoạch hợp lý hơn.

11.4.1 Mô tả

- Input: Dữ liệu GPS của hệ thống xe bus tại thành phố Chicago, Mỹ trong cùng lúc.
- Output: Xác định những vùng, khu vực giao thông ùn tắc dựa trên 2 tiêu chí: mật độ xe đông, và di chuyển chậm.

Ứng dụng này sẽ sử dụng dữ liệu hành trình từ các thiết bị thu GPS, gọi tắt là dữ liệu GPS, để trích xuất những thông tin hữu ích về tình trạng giao thông trong khu vực ở thời gian thực real-time (ví dụ tọa độ GPS, tốc độ xe, hướng di chuyển của xe).

Tập dữ liệu GPS được sử dụng để demo là dữ liệu hành trình của hệ thống xe bus tại thành phố Chicago, Hoa Kỳ vào thời điểm 21 giờ 40 phút 46 giây ngày 1/8/2020 tại địa chỉ dưới đây: ¹

¹<https://data.cityofchicago.org/Transportation/Chicago-Traffic-Tracker-Congestion-Estimates-by-Region/t2qc-9pjg>

11.4. ỨNG DỤNG VÀO DỰ ÁN THỰC TẾ

Dối với dự án thực tế, ta sẽ thu thập dữ liệu GPS từ các thiết bị thu GPS gắn trên các phương tiện giao thông công cộng là xe bus hay thu thập qua phần mềm viết cho các điện thoại thông minh của khách hàng.

```
1 SEGMENTID,STREET,DIRECTION,FROM_STREET,TO_STREET,LONGITUDE,LATITUDE, CURRENT_SPEED
2 937,Milwaukee,SE,Fullerton,Armitage,-87.70059031,41.92479169,20
3 494,Archer,EB,Austin,Central,-87.77208272,41.79406784,-1
4 470,95th,WB,Cottage Grove,Dr Martin L King Jr,-87.60450127,41.72222592,32
5 578,Lake Shore Dr,NW,47th,43rd,-87.58861739,41.81094268,-1
```

Hình 11.5: Dữ liệu GPS của hệ thống xe bus tại thành phố Chicago, Mỹ

Trong đó, **SEGMENT ID** là ID của tuyến bus, dữ liệu GPS $\langle Lon \text{ (Kinh độ)}, Lat \text{ (Vĩ độ)} \rangle$ được lấy từ cột **LONGITUDE** và **LATITUDE**, và dữ liệu về tốc độ tức thời được lấy từ cột **CURRENT SPEED**.

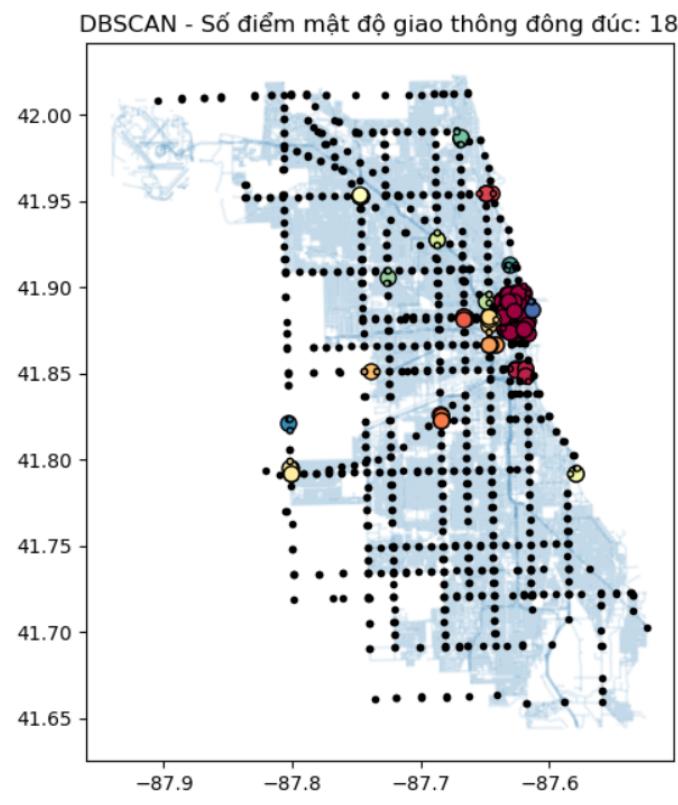
11.4.2 Thiết kế

- Bước 1: Trích xuất dữ liệu về vị trí GPS và tốc độ hiện tại của các xe từ tập dữ liệu trên.
- Bước 2: Tiền xử lý dữ liệu (preprocessing): Đa số trường hợp thiết bị giám sát hành trình ghi nhận vận tốc tức thời của phương tiện tại thời điểm thu thập thông tin về vị trí và lưu trữ vào GPS Log. Tuy nhiên một số trường hợp dữ liệu GPS Log không có thông tin về vận tốc tức thời nên được tính thông qua khoảng cách thời gian và không gian giữa hai điểm liên tiếp trong quỹ đạo.
- Bước 3: Sử dụng ngưỡng vận tốc để xác định các vị trí trên mạng lưới giao thông mà phương tiện di chuyển chậm. Tuy nhiên việc này cũng chưa thể xác định được đoạn đường ùn tắc vì nhiều nguyên nhân gây nhiễu.
- Bước 4: Áp dụng giải thuật DBSCAN cho tập hợp các vị trí ghi nhận có phương tiện di chậm, với hai tham số là khoảng cách Epsilon = 0.005 và số điểm nhỏ nhất để xác định một vùng có mật độ dày MinPts = 5.
- Bước 5: Minh họa kết quả thu được bằng các điểm (điểm nhiễu, điểm lõi) trên mạng lưới hệ thống đường bộ của thành phố.

Kết quả demo:

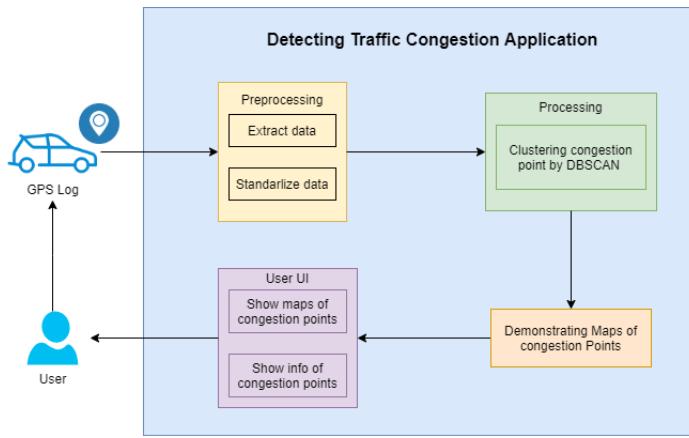
```
Estimated number of clusters: 18
Estimated number of noise points: 824
```

Hình 11.6: Kết quả phân vùng những điểm ùn tắc giao thông



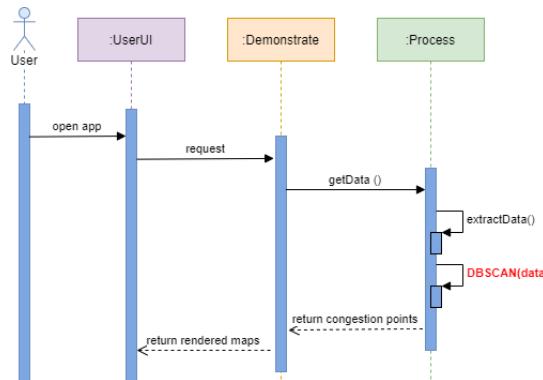
Hình 11.7: Minh họa trực quan những điểm ùn tắc giao thông trong thành phố, những điểm màu đen là điểm nhiều. Người dùng có thể di chuyển, phóng to, thu nhỏ để xem chi tiết

11.4.3 Kiến trúc phần mềm



Hình 11.8: Kiến trúc của chương trình

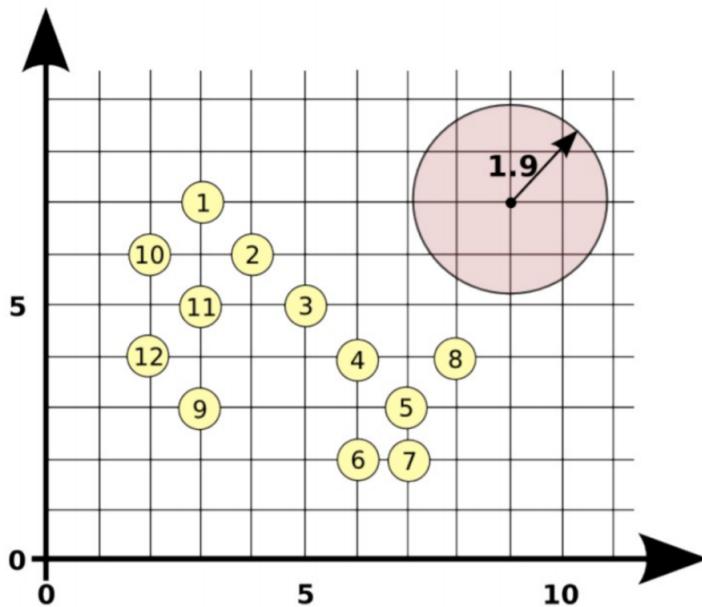
11.4.4 Sequence Diagram



Hình 11.9: Sequence Diagram khi người dùng coi tra cứu điểm ùn tắc giao thông

11.5 Bài tập

1. Áp dụng giải thuật DBSCAN với $\text{eps} = 1.9$ và $\text{MinPts}=4$ cho hình dưới đây và trả lời các câu hỏi:



Hình 11.10: Xác định trực quan

- (a) Hãy chỉ ra điểm nào là điểm lõi, điểm biên và điểm nhiễu
 - (b) Khoanh vùng các cluster nhận được
2. Cho C1 và C2 là 2 cluster kết quả thu được sau khi áp dụng DBSCAN. Hãy chứng minh tính đúng sai của các mệnh đề sau:
- (a) Nếu A là điểm lõi của cả C1 và C2 thì $C1 = C2$.
 - (b) Nếu A là điểm biên của cả C1 và C2 thì tập các điểm chung của C1 và C2 có số điểm $> \text{MinPts}$.
 - (c) C1 luôn có số điểm biên $>$ số điểm lõi.
3. Thiết kế sơ đồ kiến trúc phần mềm của hệ thống phân loại sở thích mua sắm của khách hàng và chỉ ra vị trí phần áp dụng DBSCAN.
4. Hãy cho ví dụ thực tế khi nào thì giải thuật DBSCAN tối ưu hơn giải thuật K-means và ngược lại.
5. Cho bộ dữ liệu bao gồm các điểm như bảng 5:

Điểm	Tọa độ
A	(1, 1)
B	(2, 2)
C	(3, 3)
D	(8, 8)
E	(8, 9)
F	(25, 80)
G	(24, 79)
H	(25, 81)
I	(10, 10)
J	(12, 12)

- (a) Tính khoảng cách Euclidean giữa tất cả các cặp điểm.
- (b) Với tham số Eps = 2 và MinPts = 3, xác định các điểm lõi (core point), điểm biên (border point) và điểm nhiễu (noise point). Từ đó, phân cụm các điểm dựa trên các điểm lõi, điểm biên và điểm nhiễu.
- (c) Với tham số Eps = 4 và MinPts = 3, xác định lại các điểm lõi (core point), điểm biên (border point) và điểm nhiễu (noise point). Từ đó, phân cụm các điểm dựa trên các điểm lõi, điểm biên và điểm nhiễu.
- (d) So sánh kết quả phân cụm giữa Eps = 2 và Eps = 4. Các cụm có thay đổi không? Hãy mô tả sự thay đổi trong số lượng cụm và thành phần của mỗi cụm.
- (e) Dựa trên các kết quả phân cụm, bạn có thể rút ra kết luận gì về sự ổn định của thuật toán DBSCAN trong bài toán này?

Chương 12

Giải thuật Association Rule Mining

12.1 Đặt ra vấn đề

Một cửa hàng tạp hóa nhỏ chủ yếu bán 6 sản phẩm chính là thịt heo, sữa, trứng, bột bánh, dầu ăn và quần áo. Cửa hàng có một máy tính tiền nhằm đẩy mạnh tốc độ giao dịch và máy có lưu các hóa đơn giao dịch thực hiện tại cửa hàng như sau, với giá trị 1 tương ứng với một giao dịch và 1 sản phẩm tức là sản phẩm đó được mua trong giao dịch, giá trị 0 là không được mua trong giao dịch:

Bảng 12.1: Bảng cơ sở dữ liệu chứa các giao dịch

Giao dịch	Thịt heo	Sữa	Trứng	Bột bánh	Dầu ăn	Quần áo
1	0	0	1	1	0	1
2	1	1	1	0	1	0
3	0	1	1	1	1	0
4	1	1	1	0	1	1
5	1	1	0	1	0	0
6	1	0	0	0	1	0

Chủ cửa hàng muốn từ các hóa đơn giao dịch ấy để tìm ra một quy luật buôn bán (những món hàng nào thường được mua cùng nhau) để có những quyết định kinh doanh (mua nhiều món A được tặng món B, giảm giá món C khi mua chung món D, ...) giúp tăng doanh thu cho cửa hàng. Và đây cũng là một vấn đề cần giải đáp trong chủ đề trong học máy, mang tên Luật kết hợp (Association Rule).

12.1.1 Luật kết hợp

Cho tập hợp I là là tập hợp có n tính chất khác nhau

$$I = i_1, i_2, \dots, i_n$$

12.2. CÁC TÍNH CHẤT CỦA MỘT LUẬT KẾT HỢP

D là một cơ sở dữ liệu giao dịch (transaction database) với mỗi bản ghi T chứa một tập con A các tính chất ($A \subseteq I$). Luật kết hợp là một loại tri thức khám phá được từ quá trình khai thác dữ liệu, có dạng $X \implies Y$ khi X và Y là 2 tập các tính chất, với

$$X \in \{x_1, x_2, \dots, x_n\}$$

Và Y có dạng

$$Y \in \{y_1, y_2, \dots, y_m\}$$

Ta định nghĩa tập X là tập nguyên nhân, tập Y là tập kết quả, với $X, Y \subseteq I$ và $X \cap Y = \emptyset$. Đồng thời $|A|$ kí hiệu cho số phần tử nằm trong tập A xác định.

12.2 Các tính chất của một luật kết hợp

12.2.1 Độ hỗ trợ (support)

Độ hỗ trợ của một tập $A \subseteq I$ nào đó trong D là tỉ lệ các bản ghi T trong D chứa các tính chất trong A , có công thức là:

$$supp(A) = \frac{|\{T \in D : A \subseteq T\}|}{|D|}$$

Một tập con các tính chất trong cơ sở dữ liệu giao dịch được cho là lớn (hoặc phổ biến) nếu độ hỗ trợ lớn hơn hoặc bằng ngưỡng hỗ trợ tối thiểu ($minsupp$) được đặt ra bởi người dùng.

Hệ quả:

- Tập con của một tập phổ biến cũng là một tập phổ biến.
- Tập cha của một tập không phổ biến cũng là một tập không phổ biến.

12.2.2 Độ tin cậy (confidence)

Độ tin cậy của một luật kết hợp $A \implies B$ là tỉ lệ các bản ghi T trong D chứa các tính chất của $C = A \cup B$ chia cho tỉ lệ các bản ghi đó chứa các tính chất của A . Công thức là:

$$conf(A \implies B) = \frac{supp(A \cup B)}{supp(A)}$$

Một luật kết hợp có độ tin cậy 100% là **luật kết hợp tuyệt đối**, ngược lại là **luật kết hợp tương đối**.

12.2.3 Tính hợp lệ của luật kết hợp

Theo mô hình hỗ trợ-tin cậy (Support-confidence framework, Agrawal 1993): Với I là tập tính chất trong cơ sở dữ liệu D ; $X, Y \subseteq I$ là tập con các tính chất, $X \cap Y = \emptyset$. Ta gọi $X \implies Y$ là một luật kết hợp hợp lệ nếu:

- $supp(X \cup Y) \geq minsupp$
- $conf(X \implies Y) \geq minconf$

Với độ hỗ trợ tối thiểu ($minsupp$) và độ tin cậy tối thiểu ($minconf$) là hai giá trị do người dùng đặt ra.

12.2.4 Các mục tiêu cần đạt trong việc khai thác luật kết hợp

1. Tạo ra các tập thuộc tính có tính phổ biến (độ hỗ trợ lớn hơn độ hỗ trợ tối thiểu).
2. Với mỗi tập tính chất phổ biến, mọi bộ luật với độ tin cậy tối thiểu được định nghĩa như sau: Với mỗi tập tính chất phổ biến X và $Y \subset X$, nếu

$$\frac{supp(X)}{supp(X - Y)} > minconf$$

thì $X - Y \implies Y$ là một luật kết hợp hợp lệ.

12.3 Thuật toán

12.3.1 Thuật toán Apriori

12.4 Hiện thực

12.4.1 Giới thiệu phần hiện thực: Phiên bản C# Console

Để hiện thực bằng ngôn ngữ C#, ta sẽ định nghĩa 4 class chính như sau:

1. **Class Apriori:** Chứa các hàm tìm ra các tập ứng viên và tập phổ biến.
2. **Class ItemSet:** Chứa thông tin của tập các giá trị cần tìm, kế thừa Dictionary<List<string>, int>. Trong đó:
 - **List<string>:** Là bộ các item thỏa mãn các tính chất của bài toán.
 - **int:** chứa số lượng xuất hiện các giá trị của bộ List<string>.
3. **Class AssociationRule:** Chứa các thông tin của một Association Rule. Bao gồm:
 - **string Label:** Chứa tên hiển thị của luật tương ứng.
 - **double Confidence:** Chứa độ tin cậy của một luật.
 - **double Support:** Chứa độ phổ biến của một luật.
4. **Class Service:** Giúp định dạng các biểu thức dùng để hiển thị ra màn hình của một luật.

12.4.2 Hiện thực giải thuật: Phiên bản C# Console

12.4.2.1 Class ItemSet

```
using System;
using System.Collections.Generic;
using System.Text;

namespace Implementation
{
    public class ItemSet : Dictionary<List<string>, int>
    {
        public string Label { get; set; }
        public int Support { get; set; }
    }
}
```

12.4.2.2 Class AssociationRule

```
using System;
using System.Collections.Generic;
using System.Text;

namespace Implementation
{
    public class AssociationRule
    {
        public string Label { get; set; }
        public double Confidence { get; set; }
        public double Support { get; set; }
    }
}
```

12.4.2.3 Class Service

```
using System;
using System.Collections.Generic;
using System.Text;

namespace Implementation
{
    public static class Service
    {
        public static string ToDisplay(this List<string> list, string separator = ", ")
        {
            if (list.Count == 0)
                return string.Empty;
            StringBuilder sb = new StringBuilder();
            sb.Append(list[0]);
            for (int i = 1; i < list.Count; i++)
                sb.Append(separator + list[i]);
            return sb.ToString();
        }
    }
}
```

```

    {
        sb.Append(string.Format("{0}{1}", separator, list[i]));
    }
    return sb.ToString();
}

public static string ToDisplay(this List<string> list, string exclude, string
    separator = ", ")
{
    List<string> temp = new List<string>();
    foreach (var item in list)
    {
        if (item == exclude) continue;
        temp.Add(item.ToString());
    }
    return temp.ToString();
}
}

```

Giới thiệu chức năng các hàm:

- `public static string ToDisplay(this List<string> list, string separator = ", ")`:

Trả về chuỗi gồm các phần tử nằm trong list cách nhau bởi dấu ”,” . Ví dụ: [”A”, ”B”] ⇒ ”A, B”.

- `public static string ToDisplay(this List<string> list, string exclude, string
 separator = ", ")`:

Trả về chuỗi gồm các phần tử nằm trong list khác với exclude cách nhau bởi dấu ”,” .
Ví dụ: [”A”, ”B”, ”C”], ”B” ⇒ ”A, C”

12.4.2.4 Class Apriori

```

using System;
using System.Collections.Generic;
using System.IO;
using System.Linq;
using System.Text;

namespace Implementation
{
    public class Apriori
    {
        private string filePath = @"..\..\demo.txt";
        private List<string> list;
        private List<string> DistinctValues;
        private List<ItemSet> ItemSets;

        public Apriori()
        {

```

12.4. HIÊN THỰC

```
if (File.Exists(filePath))
{
    list = File.ReadAllLines(filePath).ToList().Where(a =>
        !string.IsNullOrWhiteSpace(a)).ToList();
}
else Console.WriteLine("File doesn't exist");
ItemSets = new List<ItemSet>();

SetDistinctValues(list);

}

private void Remove_Invalid_SetItem(ref List<List<string>> itemSet,
    List<List<string>> test)
{
    List<List<string>> copy = new List<List<string>>();
    foreach (var variable in itemSet)
    {
        copy.Add(variable);
    }

    foreach (var item in copy)
    {
        for (int i = 0; i < item.Count; i++)
        {
            List<string> temp = new List<string>();
            temp.AddRange(item);
            temp.RemoveAt(i);
            bool isContain = false;
            foreach (var t in test)
            {
                //var check = t.Except(temp).ToList();
                if (t.Except(temp).ToList().Any()) continue;
                else
                {
                    isContain = true;
                    break;
                }
            }

            if (!isContain)
            {
                itemSet.Remove(item);
            }
        }
    }
}

private ItemSet Apriori_Gen(List<List<string>> input, int k, int minSupport)
{
    List<List<string>> result = new List<List<string>>();

    for (int i = 0; i < input.Count - 1; i++)
```

```

{
    for (int j = i + 1; j < input.Count; j++)
    {
        bool isValid = true;
        for (int t = 0; t < k - 1; t++)
        {
            if(input[i][t] == input[j][t]) continue;
            else
            {
                isValid = false;
                break;
            }
        }

        if (isValid == true)
        {
            List<string> temp = new List<string>();
            temp.AddRange(input[i]);
            temp.Add(input[j][k-1]);
            result.Add(temp);
        }
        else continue;
    }
}

Remove_Invalid_SetItem(ref result, input);
ItemSet itemSet = new ItemSet();

foreach (var item in result)
{
    int count = 0;
    foreach (var variable in list)
    {
        var temp = variable.Trim().Split(" ");
        bool isValid = true;
        foreach (var t in item)
        {
            if(temp.Contains(t)) continue;
            else
            {
                isValid = false;
                break;
            }
        }

        if (isValid)
            count++;
    }
    if(count>=minSupport)
        itemSet.Add(item, count);
}

return itemSet;

```

```

    }

    private ItemSet Get_First_Candidate_ItemSet(int k)
    {
        ItemSet itemSet = new ItemSet();
        foreach (var item in DistinctValues)
        {
            List<string> data = new List<string>();
            data.Add(item);
            itemSet.Add(data, 0);
        }

        return itemSet;
    }

    private void SetDistinctValues(List<string> values)
    {
        List<string> data = new List<string>();
        foreach (var item in values)
        {
            var row = item.Split(' ');
            foreach (var item2 in row)
            {
                if (string.IsNullOrWhiteSpace(item2)) continue;
                if (!data.Contains(item2))
                    data.Add(item2);
            }
        }
        DistinctValues = new List<string>();
        DistinctValues.AddRange(data.OrderBy(a => a).ToList());
    }

    internal ItemSet Get_First_Frequent_ItemSet(int k, int minSupport)
    {
        ItemSet itemSet = Get_First_Candidate_ItemSet(k:1);
        foreach (var item in itemSet.Keys.ToList())
        {
            int count = 0;
            foreach (var item2 in list)
            {
                var temp = item2.Trim().Split(" ");
                if (temp.Contains(item[0]))
                    count++;
            }

            if (count >= minSupport)
                itemSet[item] = count;
            else itemSet.Remove(item);
        }

        itemSet.Label = "L" + k.ToString();
        itemSet.Support = minSupport;
    }
}

```

```

        return itemSet;
    }

    internal ItemSet Get_Frequent_ItemSet(int k, ItemSet itemSet, int minSupport)
    {
        ItemSet result = new ItemSet();
        result = Apriori_Gen(itemSet.Keys.ToList(), k, minSupport);
        result.Label = "L" + (k + 1).ToString();
        result.Support = minSupport;

        if (itemSet.Count > 0)
        {
            ItemSets.Add(itemSet);
        }

        return result;
    }

    internal List<AssociationRule> GetRules(ItemSet frequent)
    {
        List<AssociationRule> rules = new List<AssociationRule>();
        foreach (var item in frequent)
        {
            foreach (var set in item.Key)
            {
                rules.Add(GetSingleRule(set, item));
                if (item.Key.Count > 2)
                    rules.Add(GetSingleRule(item.Key.ToString(exclude: set), item));
            }
        }

        return rules.OrderByDescending(a => a.Support)
            .ThenByDescending(a => a.Confidence).ToList();
    }

    private AssociationRule GetSingleRule(string set, KeyValuePair<List<string>,
        int> item)
    {
        var setItems = set.Split(',');
        for (int i = 0; i < setItems.Count(); i++)
        {
            setItems[i] = setItems[i].Trim();
        }
        AssociationRule rule = new AssociationRule();
        StringBuilder sb = new StringBuilder();
        sb.Append(set).Append(" => ");
        List<string> list = new List<string>();
        foreach (var set2 in item.Key)
        {
            if (setItems.Contains(set2)) continue;
            list.Add(set2);
        }
    }
}

```

12.4. HIỆN THỰC

```
        sb.Append(list.ToString());
        rule.Label = sb.ToString();
        int totalSet = 0;
        foreach (var first in ItemSets)
        {
            var myItem = first.Keys.Where(a => a.ToString() == set);
            if (myItem.Count() > 0)
            {
                first.TryGetValue(myItem.FirstOrDefault(), out totalSet);
                break;
            }
        }
        rule.Confidence = Math.Round(((double)item.Value / totalSet) * 100, 2);
        rule.Support = Math.Round(((double)item.Value / this.list.Count) * 100, 2);
        return rule;
    }
}
}
```

Giới thiệu chức năng các hàm:

- `private ItemSet Apriori_Gen(List<List<string>> input, int k, int minSupport):`

Hàm trả về tập phổ biến gồm k+1 phần tử từ tập phổ biến k phần tử và số lần xuất hiện.

- `private void Remove_Invalid_SetItem(ref List<List<string>> itemSet,
 List<List<string>> test):`

Hàm giúp xóa những tập giá trị nằm trong itemSet mà tồn tại một tập con các phần tử không có trong tập test.

- `private ItemSet Get_First_Candidate_ItemSet(int k):`

Hàm trả về tập ứng viên một phần tử.

- `internal ItemSet Get_First_Frequent_ItemSet(int k, int minSupport):`

Hàm trả về tập phổ biến một phần tử và số lần xuất hiện.

- `private void SetDistinctValues(List<string> values):`

Hàm trả về danh sách các phần tử phân biệt trong tập dữ liệu được sắp xếp theo thứ tự.

- `internal List<AssociationRule> GetRules(ItemSet frequent):`

Hàm trả về danh sách tất cả các luật thỏa mãn từ tập phổ biến.

- `private AssociationRule GetSingleRule(string set, KeyValuePair<List<string>, int item):`

Hàm tính toán các thuộc tính: độ phổ biến (support), độ tin cậy (confidence) và tên luật tương ứng. Trả về danh sách các luật và các giá trị tương ứng của luật.

12.4.2.5 Class Program: Class chứa hàm Main

```
using System;
using System.Collections.Generic;
using System.IO;

namespace Implementation
{
    class Program
    {
        static void Main(string[] args)
        {
            Apriori apriori = new Apriori();
            List<ItemSet> ItemSets = new List<ItemSet>();
            List<List<AssociationRule>> AssociationRules = new
                List<List<AssociationRule>>();

            int k = 1;
            int minSupport = 2;

            var firstCandidateItemSet = apriori.Get_First_Frequent_ItemSet(k,
                minSupport);

            if (firstCandidateItemSet.Count > 0)
                ItemSets.Add(firstCandidateItemSet);
            else return;

            Console.WriteLine("*****");
            Console.WriteLine("{0} :", firstCandidateItemSet.Label);
            foreach (var item in firstCandidateItemSet.Keys)
            {
                Console.WriteLine("{0}, {1}", string.Join(" ", item),
                    firstCandidateItemSet[item]);
            }

            var temp = firstCandidateItemSet;

            bool next;
            do
            {
                next = false;
                var frequent = apriori.Get_Frequent_ItemSet(k, temp, minSupport);

                Console.WriteLine("*****");
                Console.WriteLine("{0} :", frequent.Label);
                foreach (var item in frequent.Keys)
                {
                    Console.WriteLine("{0}, {1}", string.Join(" ", item), frequent[item]);
                }
            }
        }
    }
}
```

```
        if (frequent.Count > 0)
            ItemSets.Add(frequent);
        else break;
        ++k;
        temp = frequent;
        List<AssociationRule> rules = new List<AssociationRule>();
        rules = apriori.GetRules(frequent);
        AssociationRules.Add(rules);
        Console.WriteLine("Association Rule L{0}", k);
        Console.WriteLine(string.Format("{0, -10} | {1, -10} |
            {2,5}", "Rule", "Confidence", "Support"));
        foreach (var variable in rules)
        {
            Console.WriteLine(string.Format("{0, -10} | {1,-10} | {2,5}",
                variable.Label, variable.Confidence, variable.Support));
        }
        next = true;

    } while (next);
}
}
}
```

12.4.3 Kết quả

Tập dữ liệu:

A F
A B F
A C F
A B C F
S F D A S
B C S
B D A
B F S
D S F
A V C D R

Hình 12.1: Tập dữ liệu.

Kết quả với Min Support = 2:

Tập L1:

```
*****  
L1 :  
A, 7  
B, 5  
C, 4  
D, 4  
F, 7  
S, 4  
*****
```

Hình 12.2: Kết quả cho tập L1.

Tập L2:

```
L2 :  
A B, 3  
A C, 3  
A D, 3  
A F, 5  
B C, 2  
B F, 3  
B S, 2  
C F, 2  
D F, 2  
D S, 2  
F S, 3  
  
Association Rule L2  
Rule | Confidence | Support  
A => F | 71.43 | 50  
F => A | 71.43 | 50  
C => A | 75 | 30  
D => A | 75 | 30  
S => F | 75 | 30  
B => A | 60 | 30  
B => F | 60 | 30  
A => B | 42.86 | 30  
A => C | 42.86 | 30  
A => D | 42.86 | 30  
F => B | 42.86 | 30  
F => S | 42.86 | 30  
C => B | 50 | 20  
S => B | 50 | 20  
C => F | 50 | 20  
D => F | 50 | 20  
D => S | 50 | 20  
S => D | 50 | 20  
B => C | 40 | 20  
B => S | 40 | 20  
F => C | 28.57 | 20  
F => D | 28.57 | 20  
*****
```

Hình 12.3: Kết quả cho tập L2.

12.5. ỨNG DỤNG VÀO DỰ ÁN PHẦN MỀM

Tập L3:

L3 :		
A B F, 2	A C F, 2	D F S, 2
Association Rule L3		
Rule	Confidence	Support
C, F => A	100	20
D, S => F	100	20
D, F => S	100	20
B, F => A	66.67	20
A, B => F	66.67	20
A, C => F	66.67	20
F, S => D	66.67	20
C => A, F	50	20
D => F, S	50	20
S => D, F	50	20
B => A, F	40	20
A, F => B	40	20
A, F => C	40	20
A => B, F	28.57	20
F => A, B	28.57	20
A => C, F	28.57	20
F => A, C	28.57	20
F => D, S	28.57	20

Hình 12.4: Kết quả cho tập L3.

12.5 Ứng dụng vào dự án phần mềm

Chúng ta sẽ áp dụng vào một dự án phần mềm đơn giản, cho phép người quản lý từ một tập dữ liệu có sẵn các giao dịch khách hàng trên ứng dụng, từ đó hiển thị ra kết quả khi chạy giải thuật Association Rule Mining hiển thị ra các thông tin, từ những thông tin đó thì người quản lý có thể xem danh sách các sản phẩm có xu hướng mua cùng với nhau, từ đó có thể tạo ra các chiến lược để bán hàng hiệu quả hơn (sắp xếp các món hàng có xu hướng mua lại với nhau, hoặc tạo một bộ sản phẩm và có cách chính sách ưu đãi,...) giúp nâng cao doanh thu.

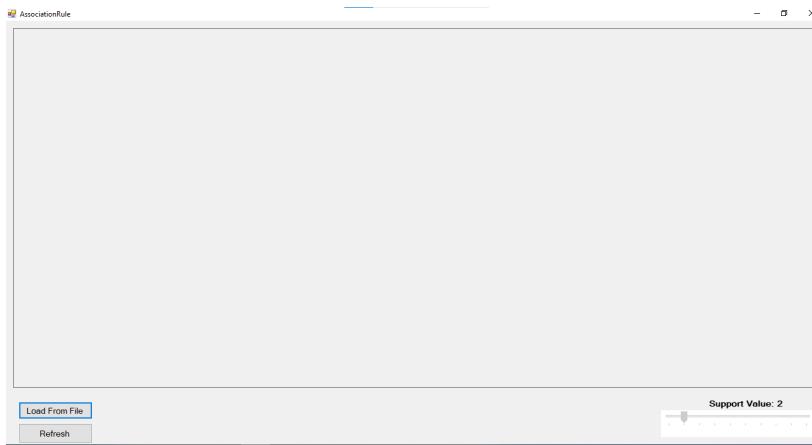
Tập dữ liệu

```
A F
A B F
A C F
A B C F
S F D A S
B C S
B D A
B F S
D S F
A V C D R
```

Hình 12.5: Tập dữ liệu.

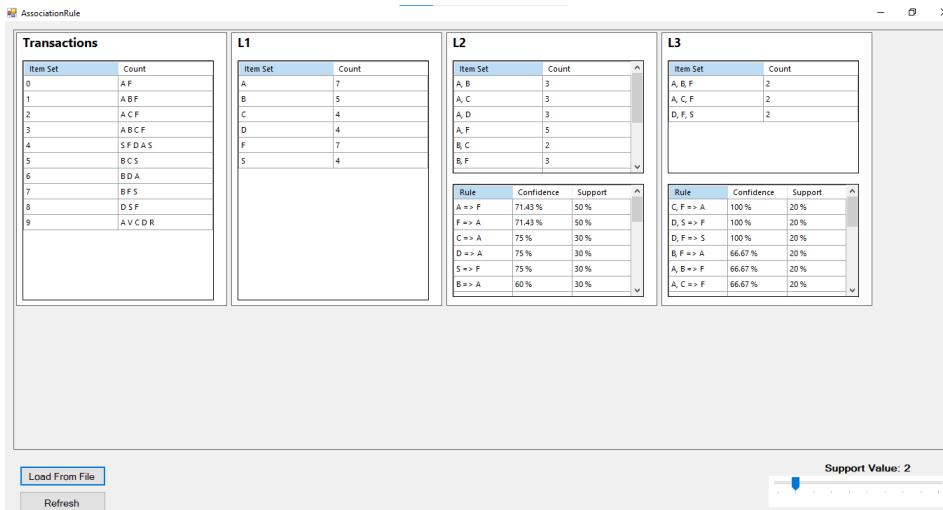
CHƯƠNG 12. GIẢI THUẬT ASSOCIATION RULE MINING

Ứng dụng chúng ta đơn giản sẽ có giao diện như sau:



Hình 12.6: Giao diện ban đầu của ứng dụng

Ta sẽ chọn vào button Load From File và sau đó hộp thoại OpenFileDialog xuất hiện, ta chọn file dữ liệu cần thiết để xem các thông tin khi chạy file đó với giải thuật Association Rule Mining (ở ví dụ này ta dùng file demo.txt), phía dưới cùng bên phải có một thanh trackBar để chọn độ Support cho giải thuật (mặc định ban đầu là 2). Sau khi chọn file thành công, ta có giao diện như sau:



Hình 12.7: Giải thuật với Support = 2

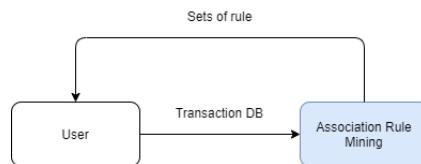
Sau đó ta có thể xem các thông tin, các xác suất các món hàng hay được khách hàng mua kèm với nhau. Phần (Item set, Count) là thông tin của Apriori và (Rule, Confidence, Support) là thông tin của Association Rule. Ta có thể chọn giá trị Support khác qua thanh

12.5. ỨNG DỤNG VÀO DỰ ÁN PHẦN MỀM

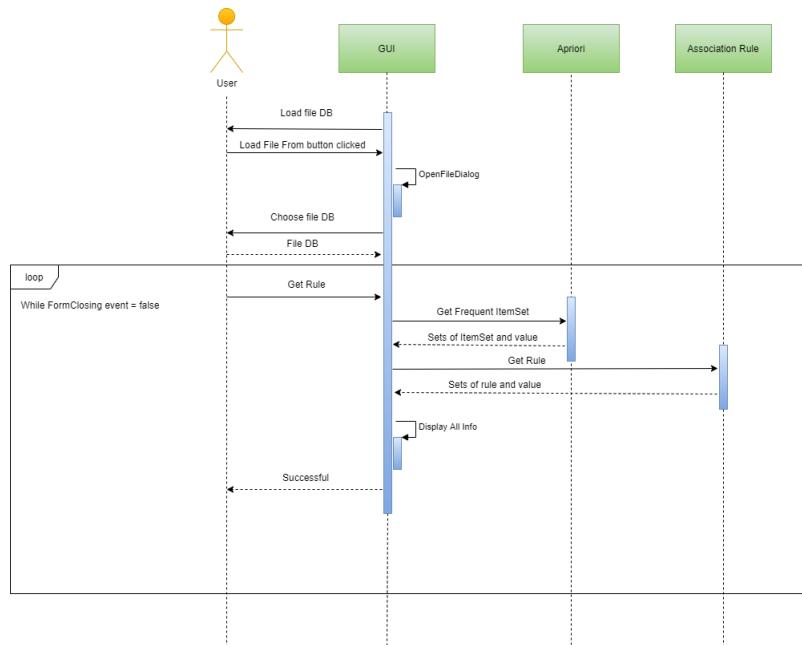
trackBar, khi sự kiện Scroll xảy ra thì ứng dụng cũng được tự động cập nhật các bảng tương ứng để hiển thị ra giao diện.

Các bạn có thể truy cập đường dẫn phía dưới để tải và chạy thử ứng dụng (ứng dụng hỗ trợ cho người quản lý - version: winform application C#):

<https://github.com/OnceUponATimeMathley/SoftwareEngineering-Extend>



Hình 12.8: Kiến trúc phần mềm



Hình 12.9: Sequence Diagram chức năng hiển thị các luật tương ứng khi chạy giải thuật Association Rule Mining với tập dữ liệu có sẵn

12.6 Bài tập

Câu 1: Hãy so sánh thuật toán Apriori và thuật toán FP-growth.

Câu 2: Cho cơ sở giao dịch bên

1. Sử dụng thuật toán Apriori để tìm tập phỏ biến với minsupp = 22 %.
2. Liệt kê các tập phỏ biến tối đại và tập bao phỏ biến.
3. Tìm tất cả các luật kết hợp thỏa mãn:
 - (a) Minconf = 50 %.
 - (b) Minconf = 70 %.

TID	Items
1	M1,M2,M5
2	M2,M4
3	M2,M3
4	M1,M2,M4
5	M1,M3
6	M2,M3
7	M1,M3
8	M1,M2,M3,M5
9	M1,M2,M3

Câu 3: Cho cơ sở giao dịch bên với 5 giao dịch, độ hỗ trợ tối thiểu là 0.6, độ tin cậy tối thiểu là 0.8

1. Tìm mọi tập tối thiểu phỏ biến bằng giải thuật Apriori.
2. Tìm mọi luật kết hợp bằng giải thuật FP-Growth và chỉ ra các luật kết hợp có "B" là tiền đề ("B" nằm ở về bên trái).

TID	Items
1	A,B,C,D,E,F
2	B,C,D,E,F,G
3	A,D,E,H
4	A,D,F,I,J
5	B,D,E,K

Câu 4:

Ta thường quan tâm các luật kết hợp có độ tin cậy cao. Tại sao các luật kết hợp có độ tin cậy 100% nên được bỏ qua? Tại sao các luật kết hợp có độ tin cậy gần bằng 100% (ví dụ: 99.8%, 99%, ..) nên được để ý kĩ?

Chương 13

Matrix Factorization for Collaborative filtering

13.1 Tổng quan về Recommendation System

13.1.1 Giới thiệu

Hệ thống khuyến nghị (*Recommendation System*) thực sự bùng nổ khoảng 10 - 15 năm trở lại đây do sự bùng nổ của Internet cũng như các ứng dụng thương mại điện tử và mạng xã hội. Có hai thực thể chính trong một recommendation system là user và item. User là người dùng; item là sản phẩm, ví dụ như các bộ phim, bài hát, cuốn sách, clip, hoặc cũng có thể là các người dùng khác trong bài toán gọi ý kết bạn. Mục đích chính của các Recommendation System là dự đoán mức độ quan tâm của một người dùng tới một sản phẩm nào đó, qua đó có chiến lược recommendation phù hợp. Chẳng hạn, khi bạn mở YouTube, trang chủ sẽ hiển thị một loạt các video mà bạn cảm thấy hoàn toàn liên quan đến sở thích (gu âm nhạc, thể loại phim ảnh, ...) hoặc thói quen theo dõi gần đây. Điều này khiến bạn cảm thấy thuận tiện hơn trong việc tìm kiếm, cũng như gợi ý ra những thể loại video mà bạn hoàn toàn có thể sẽ thích dù chưa từng theo dõi. Đó là một ứng dụng của Recommendation System.

Các Recommendation System thường được chia thành hai nhóm lớn:

- **Content-based system:** khuyến nghị dựa trên đặc tính của sản phẩm. Chẳng hạn, một người dùng xem rất nhiều các bộ phim có tính chất hài hước, giải trí thì hệ thống sẽ gợi ý một bộ phim có tính chất tương tự như vậy, chẳng hạn Gia đình là số 1. Cách tiếp cận này yêu cầu việc sắp xếp các sản phẩm vào từng nhóm hoặc đi tìm các đặc trưng của từng sản phẩm. Tuy nhiên, có những sản phẩm không có nhóm cụ thể và việc xác định nhóm hoặc đặc trưng của từng sản phẩm đôi khi là rất khó.
- **Collaborative Filtering:** khuyến nghị các sản phẩm dựa trên sự tương quan (similarity) giữa các người dùng và/hoặc sản phẩm. Chẳng hạn, ta đã biết ba người dùng A, B, C đều thích nhạc của ca sĩ Sơn Tùng MTP, và ngoài ra còn biết thêm rằng B, C đều thích nhạc của ca sĩ Charlie Puth, thì có khả năng A cũng sẽ thích nhạc của ca sĩ Charlie Puth. Đây là ví dụ của việc đề nghị dựa trên những người dùng có sở thích tương đồng.

13.1.2 Utility matrix

Như đã đề cập, có hai thực thể chính trong các Recommendation System là user và item. Mỗi user sẽ có mức độ quan tâm (degree of preference) tới từng item khác nhau. Mức độ quan tâm này, nếu đã biết trước, được gán cho một giá trị ứng với mỗi cặp user-item. Thông tin về mức độ quan tâm của một user tới một item có thể được thu thập thông qua một hệ thống đánh giá (review và rating); hoặc có thể dựa trên việc user đã click vào thông tin của item trên website; hoặc có thể dựa trên thời gian và số lần một user xem thông tin của một item.

Dưới đây là ví dụ cho một hệ thống rating, mức độ quan tâm của một user tới một item được đo bằng giá trị user đó đã đánh giá cho item đó, chẳng hạn số sao trên tổng cộng năm sao. Tập hợp tất cả các rating, bao gồm cả những giá trị chưa biết cần được dự đoán, tạo nên một ma trận gọi là ma trận utility. Các giá trị "?" tương ứng với các ô chưa có dữ liệu. Công việc của Recommendation System là hoàn thiện ma trận trên ở các vị trí có dấu "?", để từ đó gợi ý cho user những item mà giá trị có được từ vị trí (user - item) là lớn. Rõ ràng, càng nhiều ô đã được điền trong ma trận thì giá trị dự đoán ở các ô chưa có giá trị là càng chính xác.

	A	B	C	D	E	F	Item's feature vectors
Người phán xử	5	5	0	0	?	1	$X_1 = [0.99, 0.01]^T$
Chạy án	5	?	?	0	?	?	$X_2 = [0.91, 0.11]^T$
Conan Thám tử lừng danh	?	0	5	?	1	?	$X_3 = [0.69, 0.96]^T$
Doraemon và những người bạn	1	2	4	?	?	4	$X_4 = [0.01, 0.99]^T$
One Punch Man	0	0	5	3	?	?	$X_5 = [0.06, 0.98]^T$
User's model	w_1	w_2	w_3	w_4	w_5	w_6	\leq Need to optimize

Hình 13.1: Bài toán Recommendation tương đương với việc hoàn thiện ma trận Utility

13.2 Content-based System

13.2.1 Giới thiệu

Trong các hệ thống content-based, tức dựa trên nội dung của mỗi item, chúng ta cần xây dựng một bộ hồ sơ (profile) cho mỗi item. Profile này được biểu diễn dưới dạng toán học là một vector đặc trưng. Trong những trường hợp đơn giản, vector này được trực tiếp trích xuất từ item. Chẳng hạn, trong ví dụ trên, mỗi bộ phim đều có một bộ hồ sơ là một vector dạng cột gồm 2 đặc trưng : tính hình sự và tính hoạt hình. Ở đây để đơn giản, ta chỉ chọn 2 đặc trưng, tuy nhiên thực tế có thể cần nhiều hơn 2. Ví dụ với bộ phim "Người phán xử", rõ ràng đây là một bộ phim có tính hình sự và không có tính hoạt hình, vậy thì hồ sơ cho nó có thể là: $[0.99, 0.1]^T$. Quá trình xây dựng item profile có thể hoàn toàn độc lập với rating từ user mà chỉ dựa vào đặc tính riêng của mỗi item.

Tương tự như thế, hành vi của mỗi user cũng có thể được mô hình hóa dưới dạng tập các tham số w_u . Dữ liệu huấn luyện để xây dựng mỗi mô hình w_u là các cặp (item profile, rating)

tương ứng với các item mà user đó đã đánh giá. Khi đã có hồ sơ của một item và hành vi của một user (x_i và w_u , là các vector đặc trưng), đầu ra của hệ thống khuyến nghị là dự đoán mức độ yêu thích của user này cho item đó, được biểu diễn bằng một hàm $f(w_u, x_i)$. Chẳng hạn, nếu có dạng tuyến tính, $y_{ui} = f(w_u, x_i) = w_u x_i + b_u$.

Thảo luận:

- Đặc điểm của phương pháp Content-based recommendation system là việc xây dựng mô hình cho mỗi user không phụ thuộc vào các user khác.
- Việc xây dựng mô hình cho mỗi user có thể được coi như bài toán regression hoặc classification với dữ liệu huấn luyện là các cặp (item profile, rating) mà user đó đã đánh giá. Item profile không phụ thuộc vào user mà phụ thuộc vào các đặc điểm mô tả của item.
- Việc làm này có lợi thế là tiết kiệm bộ nhớ và thời gian tính toán.

13.2.2 Xây dựng hàm mất mát

Mô hình tuyến tính: Giả sử rằng ta có thể tìm được một mô hình cho mỗi user, minh họa bởi vector cột hệ số \mathbf{w}_n và bias b_n sao cho mức độ quan tâm của một user tới một item có thể tính được bằng một hàm tuyến tính:

$$y_{mn} = \mathbf{x}_m \mathbf{w}_n + b_n \quad (12.2.1)$$

(Chú ý rằng \mathbf{x}_m là một vector hàng, \mathbf{w}_n là một vector cột.)

Xét một user thứ n bất kỳ, nếu ta coi training set là tập hợp các thành phần đã được điền của \mathbf{y}_n (cột thứ n của ma trận \mathbf{Y}), ta có thể xây dựng hàm mất mát tương tự như Ridge Regression như sau:

$$\mathcal{L}_n = \frac{1}{2s_n} \sum_{m : r_{mn}=1} (\mathbf{x}_m \mathbf{w}_n + b_n - y_{mn})^2 + \frac{\lambda}{2s_n} \|\mathbf{w}_n\|_2^2 \quad (12.2.2)$$

Trong đó, thành phần thứ hai là regularization term và λ là một tham số dương, s_n là số lượng các items mà user thứ n đã đánh giá. Nói cách khác:

$$s_n = \sum_{m=1}^M r_{mn} \quad (12.2.3)$$

là tổng các phần tử trên cột thứ n của ma trận rated or not \mathbf{R} .

Vì biểu thức loss function chỉ phụ thuộc vào các items đã được rated, ta có thể rút gọn nó bằng cách đặt $\hat{\mathbf{y}}_n$ là sub vector của \mathbf{y} được xây dựng bằng cách trích các thành phần khác dấu ? ở cột thứ n , tức đã được rated bởi user thứ n trong Utility Matrix \mathbf{Y} . Đồng thời, đặt $\hat{\mathbf{X}}_n$ là sub matrix của ma trận feature \mathbf{X} , được tạo bằng cách trích các hàng tương ứng với các items đã được rated bởi user thứ n). Khi đó, biểu thức hàm mất mát của mô hình cho user thứ n được viết gọn thành:

$$\mathcal{L}_n = \frac{1}{2s_n} \|\hat{\mathbf{X}}_n \mathbf{w}_n + b_n \mathbf{e}_n - \hat{\mathbf{y}}_n\|_2^2 + \frac{\lambda}{2s_n} \|\mathbf{w}_n\|_2^2 \quad (12.2.4)$$

Trong đó, \mathbf{e}_n là vector cột chứa s_n phần tử 1. Ta sẽ xem xét ví dụ dưới đây để hiểu rõ hơn về hàm mất mát này.

13.2.3 Ví dụ về hàm mất mát cho User

Quay trở lại với ví dụ trong Hình 13.1 về ma trận Utility cho hệ thống đề xuất phim ảnh, ở đây để đơn giản hóa ta xét mỗi vector đặc trưng cho từng bộ phim là một vector 2 chiều (tính hình sự và tính hoạt hình) và tạm coi các vector này đã được xác định bằng một cách nào đó, feature matrix cho các items (mỗi hàng tương ứng với một item) là:

$$\mathbf{X} = \begin{bmatrix} 0.99 & 0.01 \\ 0.91 & 0.11 \\ 0.69 & 0.96 \\ 0.01 & 0.99 \\ 0.06 & 0.98 \end{bmatrix} \quad (12.2.5)$$

Xét trường hợp của user F với $n = 6$, $\mathbf{y}_6 = [1, ?, ?, 4, ?]^T \Rightarrow \mathbf{r}_6 = [1, 0, 0, 1, 0]^T$. Vì F mới chỉ rated cho items thứ nhất và thứ tư nên $s_6 = 2$. Hơn nữa:

$$\hat{\mathbf{x}}_6 = \begin{bmatrix} 0.99 & 0.01 \\ 0.01 & 0.99 \end{bmatrix}, \hat{\mathbf{y}}_6 = \begin{bmatrix} 1 \\ 4 \end{bmatrix}, \mathbf{e}_6 = \begin{bmatrix} 1 \\ 1 \end{bmatrix} \quad (12.2.6)$$

Khi đó, hàm mất mát cho hệ số tương ứng với user F là:

$$\mathcal{L}_6 = \frac{1}{4} \left\| \begin{bmatrix} 0.99 & 0.01 \\ 0.01 & 0.99 \end{bmatrix} \mathbf{w}_6 + b_6 \begin{bmatrix} 1 \\ 1 \end{bmatrix} - \begin{bmatrix} 1 \\ 4 \end{bmatrix} \right\|_2^2 + \frac{\lambda}{4} \|\mathbf{w}_6\|_2^2 \quad (12.2.7)$$

13.3 Content-based System

13.3.1 Giới thiệu

Trong các hệ thống content-based, tức dựa trên nội dung của mỗi item, chúng ta cần xây dựng một bộ hồ sơ (profile) cho mỗi item. Profile này được biểu diễn dưới dạng toán học là một vector đặc trưng. Trong những trường hợp đơn giản, vector này được trực tiếp xuất từ item. Chẳng hạn, trong ví dụ trên, mỗi bộ phim đều có một bộ hồ sơ là một vector dạng cột gồm 2 đặc trưng : tính hình sự và tính hoạt hình. Ở đây để đơn giản, ta chỉ chọn 2 đặc trưng, tuy nhiên thực tế có thể cần nhiều hơn 2. Ví dụ với bộ phim "Người phán xử", rõ ràng đây là một bộ phim có tính hình sự và không có tính hoạt hình, vậy thì hồ sơ cho nó có thể là: $[0.99, 0.1]^T$. Quá trình xây dựng item profile có thể hoàn toàn độc lập với rating từ user mà chỉ dựa vào đặc tính riêng của mỗi item.

Tương tự như thế, hành vi của mỗi user cũng có thể được mô hình hoá dưới dạng tập các tham số w_u . Dữ liệu huấn luyện để xây dựng mỗi mô hình w_u là các cặp (item profile, rating) tương ứng với các item mà user đó đã đánh giá. Khi đã có hồ sơ của một item và hành vi của một user (x_i và w_u , là các vector đặc trưng), đầu ra của hệ thống khuyến nghị là dự đoán mức độ yêu thích của user này cho item đó, được biểu diễn bằng một hàm $f(w_u, x_i)$. Chẳng hạn, nếu có dạng tuyến tính, $y_{ui} = f(w_u, x_i) = w_u x_i + b_u$.

Thảo luận:

- Đặc điểm của phương pháp Content-based recommendation system là việc xây dựng mô hình cho mỗi user không phụ thuộc vào các user khác.
- Việc xây dựng mô hình cho mỗi user có thể được coi như bài toán regression hoặc classification với dữ liệu huấn luyện là các cặp (item profile, rating) mà user đó đã đánh giá. Item profile không phụ thuộc vào user mà phụ thuộc vào các đặc điểm mô tả của item.

- Việc làm này có lợi thế là tiết kiệm bộ nhớ và thời gian tính toán.

13.3.2 Xây dựng hàm mất mát

Mô hình tuyến tính: Giả sử rằng ta có thể tìm được một mô hình cho mỗi user, minh họa bởi vector cột hệ số \mathbf{w}_n và bias b_n sao cho mức độ quan tâm của một user tới một item có thể tính được bằng một hàm tuyến tính:

$$y_{mn} = \mathbf{x}_m \mathbf{w}_n + b_n \quad (12.2.1)$$

(Chú ý rằng \mathbf{x}_m là một vector hàng, \mathbf{w}_n là một vector cột.)

Xét một user thứ n bất kỳ, nếu ta coi training set là tập hợp các thành phần đã được điền của \mathbf{y}_n (cột thứ n của ma trận \mathbf{Y}), ta có thể xây dựng hàm mất mát tương tự như Ridge Regression như sau:

$$\mathcal{L}_n = \frac{1}{2s_n} \sum_{m : r_{mn}=1} (\mathbf{x}_m \mathbf{w}_n + b_n - y_{mn})^2 + \frac{\lambda}{2s_n} \|\mathbf{w}_n\|_2^2 \quad (12.2.2)$$

Trong đó, thành phần thứ hai là regularization term và λ là một tham số dương, s_n là số lượng các items mà user thứ n đã đánh giá. Nói cách khác:

$$s_n = \sum_{m=1}^M r_{mn} \quad (12.2.3)$$

là tổng các phần tử trên cột thứ n của ma trận rated or not \mathbf{R} .

Vì biểu thức loss function chỉ phụ thuộc vào các items đã được rated, ta có thể rút gọn nó bằng cách đặt $\hat{\mathbf{y}}_n$ là sub vector của \mathbf{y} được xây dựng bằng cách trích các thành phần khác dấu ? ở cột thứ n , tức đã được rated bởi user thứ n trong Utility Matrix \mathbf{Y} . Đồng thời, đặt $\hat{\mathbf{X}}_n$ là sub matrix của ma trận feature \mathbf{X} , được tạo bằng cách trích các hàng tương ứng với các items đã được rated bởi user thứ n). Khi đó, biểu thức hàm mất mát của mô hình cho user thứ n được viết gọn thành:

$$\mathcal{L}_n = \frac{1}{2s_n} \|\hat{\mathbf{X}}_n \mathbf{w}_n + b_n \mathbf{e}_n - \hat{\mathbf{y}}_n\|_2^2 + \frac{\lambda}{2s_n} \|\mathbf{w}_n\|_2^2 \quad (12.2.4)$$

Trong đó, \mathbf{e}_n là vector cột chứa s_n phần tử 1. Ta sẽ xem xét ví dụ dưới đây để hiểu rõ hơn về hàm mất mát này.

13.3.3 Ví dụ về hàm mất mát cho User

Quay trở lại với ví dụ trong Hình 13.1 về ma trận Utility cho hệ thống đề xuất phim ảnh, ở đây để đơn giản hóa ta xét mỗi vector đặc trưng cho từng bộ phim là một vector 2 chiều (tính hình sự và tính hoạt hình) và tạm coi các vector này đã được xác định bằng một cách nào đó, feature matrix cho các items (mỗi hàng tương ứng với một item) là:

$$\mathbf{X} = \begin{bmatrix} 0.99 & 0.01 \\ 0.91 & 0.11 \\ 0.69 & 0.96 \\ 0.01 & 0.99 \\ 0.06 & 0.98 \end{bmatrix} \quad (12.2.5)$$

Xét trường hợp của user F với $n = 6$, $\mathbf{y}_6 = [1, ?, ?, 4, ?]^T \Rightarrow \mathbf{r}_6 = [1, 0, 0, 1, 0]^T$. Vì F mới chỉ rated cho items thứ nhất và thứ tư nên $s_6 = 2$. Hơn nữa:

$$\hat{\mathbf{X}}_6 = \begin{bmatrix} 0.99 & 0.01 \\ 0.01 & 0.99 \end{bmatrix}, \hat{\mathbf{y}}_6 = \begin{bmatrix} 1 \\ 4 \end{bmatrix}, \mathbf{e}_6 = \begin{bmatrix} 1 \\ 1 \end{bmatrix} \quad (12.2.6)$$

Khi đó, hàm mất mát cho hệ số tương ứng với user F là:

$$\mathcal{L}_6 = \frac{1}{4} \left\| \begin{bmatrix} 0.99 & 0.01 \\ 0.01 & 0.99 \end{bmatrix} \mathbf{w}_6 + b_6 \begin{bmatrix} 1 \\ 1 \end{bmatrix} - \begin{bmatrix} 1 \\ 4 \end{bmatrix} \right\|_2^2 + \frac{\lambda}{4} \|\mathbf{w}_6\|_2^2 \quad (12.2.7)$$

13.4 Matrix Factorization for Collaborative filtering

13.4.1 Giới thiệu

Trong Content-based System, item profile được xây dựng dựa trên thông tin mô tả của item và quá trình xây dựng này độc lập với quá trình tìm kiếm hệ số phù hợp cho mỗi user. Việc xây dựng item profile đóng vai trò rất quan trọng và có ảnh hưởng trực tiếp lên hiệu năng của mô hình. Tuy nhiên, không phải khi nào cũng có thể xây dựng profile cho mỗi item (dữ liệu được thêm vào ngày càng nhiều, item không thể hiện được rõ ràng tính chất). Mặt khác, xây dựng từng mô hình riêng lẻ cho mỗi user dẫn đến kết quả chưa thực sự tốt vì không khai thác được mối quan hệ giữa các user.

Bây giờ, giả sử rằng ta không cần xây dựng từ trước các item profile \mathbf{X} mà vector đặc trưng cho mỗi item này có thể được huấn luyện đồng thời với mô hình của mỗi user (ở đây là một vector hệ số). Điều này nghĩa là, biến số trong bài toán tối ưu là cả \mathbf{X} và \mathbf{W} chứ không đơn thuần là \mathbf{W} như với Content-based System; trong đó, \mathbf{X} là ma trận của toàn bộ item profile, mỗi cột tương ứng với một item, \mathbf{W} là ma trận của toàn bộ user model, mỗi cột tương ứng với một user. Với phương pháp Matrix factorization collaborative filtering (MFCF) (Phân tích đa thức thành nhân tử), ta sẽ cố gắng xấp xỉ ma trận utility $\mathbf{Y} \in \mathbb{R}^{M \times N}$ bằng tích của hai ma trận $\mathbf{X} \in \mathbb{R}^{K \times M}$ và $\mathbf{W} \in \mathbb{R}^{K \times N}$. Thông thường, K được chọn là một số nhỏ hơn rất nhiều so với M, N . Khi đó, cả hai ma trận \mathbf{X} và \mathbf{W} đều có rank không vượt quá K và ma trận utility \mathbf{Y} sẽ xấp xỉ:

$$\mathbf{Y} \approx \mathbf{X}^T \times \mathbf{W} \quad (13.1)$$

Thảo luận

- Ý tưởng chính đằng sau Matrix factorization cho recommendation system là tồn tại các đặc trưng ẩn (latent feature) mô tả sự liên quan giữa các item và các user. Chẳng hạn với một hệ thống khuyến nghị các bộ phim, đặc trưng ẩn có thể là tính chất hình sự, hoạt hình, hài kịch, ... hoặc là một sự kết hợp của các tính chất này; hoặc là một tính chất bất kỳ nào đó mà ta không cần đặt tên và không rõ về nó. Mỗi item mang một vector \mathbf{x} chứa hệ số biểu thị tính chất ẩn của nó, và hệ số càng cao thể hiện item mang tính chất ẩn đó càng rõ rệt. Tương tự với user cũng mang một vector \mathbf{w} chứa các hệ số thể hiện xu hướng thích một tính chất ẩn nào đó mô tả trong item. Giá trị $\mathbf{x}^T \mathbf{w}$ càng cao chứng tỏ các thành phần trong \mathbf{x} và \mathbf{w} cũng cao, tức là item mang một tính chất ẩn và tính chất đó cũng được user đó thích.
- Để tìm nghiệm của bài toán tối ưu, ta phải lần lượt đi tìm \mathbf{X} và \mathbf{W} khi thành phần còn lại được cố định. Như vậy, mỗi cột của \mathbf{X} sẽ phụ thuộc vào toàn bộ các cột của \mathbf{W} . Ngược lại, mỗi cột của \mathbf{W} lại phụ thuộc vào toàn bộ các cột của \mathbf{X} . Như vậy, có những mối quan hệ ràng buộc chằng chịt giữa các thành phần của hai ma trận trên. Tức chúng ta cần sử dụng thông tin của tất cả để suy ra tất cả. Vậy nên phương pháp này cũng được xếp vào Collaborative Filtering.

13.4.2 Xây dựng và tối ưu hàm mất mát

13.4.2.1 Hàm mất mát

Như đã đề cập, đánh giá của user n lên item m có thể được xấp xỉ bởi $\hat{y}_{mn} \approx \mathbf{x}_m^T \mathbf{w}_n$. Do đó hàm mất mát có thể được viết thành :

$$\mathcal{L}(\mathbf{X}, \mathbf{W}) = \frac{1}{2s} \sum_{n=1}^N \sum_{m:r_{mn}=1} (y_{mn} - \mathbf{x}_m^T \mathbf{w}_n)^2 + \frac{\lambda}{2} (\|\mathbf{X}\|_F^2 + \|\mathbf{W}\|_F^2) \quad (12.3.1)$$

trong đó $r_{mn} = 1$ nếu item thứ m đã được đánh giá bởi user thứ n , s là số lượng rating trong ma trận Utility. Thành phần thứ nhất chính là trung bình sai số của mô hình. Thành phần thứ hai trong hàm mất mát phía trên là l_2 regularization, giúp tránh overfitting.

13.4.2.2 Tối ưu hàm mất mát

Khi cố định \mathbf{X} , việc tối ưu \mathbf{W} chính là bài toán tối ưu trong Content-based Recommendation Systems:

$$\mathcal{L}(\mathbf{W}) = \frac{1}{2s} \sum_{n=1}^N \sum_{m:r_{mn}=1} (y_{mn} - \mathbf{x}_m^T \mathbf{w}_n)^2 + \frac{\lambda}{2} \|\mathbf{W}\|_F^2 \quad (12.3.2)$$

Khi cố định \mathbf{W} , việc tối ưu \mathbf{X} được đưa về tối ưu hàm:

$$\mathcal{L}(\mathbf{X}) = \frac{1}{2s} \sum_{n=1}^N \sum_{m:r_{mn}=1} (y_{mn} - \mathbf{x}_m^T \mathbf{w}_n)^2 + \frac{\lambda}{2} \|\mathbf{X}\|_F^2 \quad (12.3.3)$$

Hai bài toán này sẽ được tối ưu bằng Gradient Descent.

Đối với bài toán (12.3.2), ta tách thành \mathbf{N} bài toán nhỏ ứng với mỗi n :

$$\mathcal{L}(\mathbf{w}_n) = \frac{1}{2s} \sum_{m:r_{mn}=1} (y_{mn} - \mathbf{x}_m^T \mathbf{w}_n)^2 + \frac{\lambda}{2} \|\mathbf{w}_n\|_2^2 \quad (12.3.4)$$

Vì biểu thức trong dấu \sum chỉ phụ thuộc vào các items đã được rated bởi user đang xét, ta có thể đơn giản nó bằng cách đặt $\hat{\mathbf{X}}_n$ là ma trận được tạo bởi các hàng tương ứng với các items đã được rated đó, và $\hat{\mathbf{y}}_n$ là các ratings tương ứng. Khi đó:

$$\mathcal{L}(\mathbf{w}_n) = \frac{1}{2s} \|\hat{\mathbf{y}}_n - \hat{\mathbf{X}}_n^T \mathbf{w}_n\|^2 + \frac{\lambda}{2} \|\mathbf{w}_n\|_2^2 \quad (12.3.5)$$

Đạo hàm của (12.3.4) :

$$\frac{\partial \mathcal{L}(\mathbf{w}_n)}{\partial \mathbf{w}_n} = -\frac{1}{s} \hat{\mathbf{X}}_n^T (\hat{\mathbf{y}}_n - \hat{\mathbf{X}}_n^T \mathbf{w}_n) + \lambda \mathbf{w}_n \quad (12.3.6)$$

Vậy công thức cập nhật cho các giá trị \mathbf{w}_n của (12.3.2) là :

$$\mathbf{w}_n = \mathbf{w}_n - \eta \left(-\frac{1}{s} \hat{\mathbf{X}}_n^T (\hat{\mathbf{y}}_n - \hat{\mathbf{X}}_n^T \mathbf{w}_n) + \lambda \mathbf{w}_n \right) \quad (12.3.7)$$

Tương tự như thế, với bài toán (12.3.3), công thức cập nhật cho các giá trị của \mathbf{x}_m là :

$$\mathbf{x}_m = \mathbf{x}_m - \eta \left(-\frac{1}{s} (\hat{\mathbf{y}}^m - \mathbf{x}_m^T \hat{\mathbf{W}}_m) \hat{\mathbf{W}}_m^T + \lambda \mathbf{x}_m \right) \quad (12.3.8)$$

13.5. ÁP DỤNG MFCF VÀO HỆ THỐNG KHUYẾN NGHỊ PHIM ẢNH TRÊN WEBSITE XEM PHIM TRỰC TUYẾN

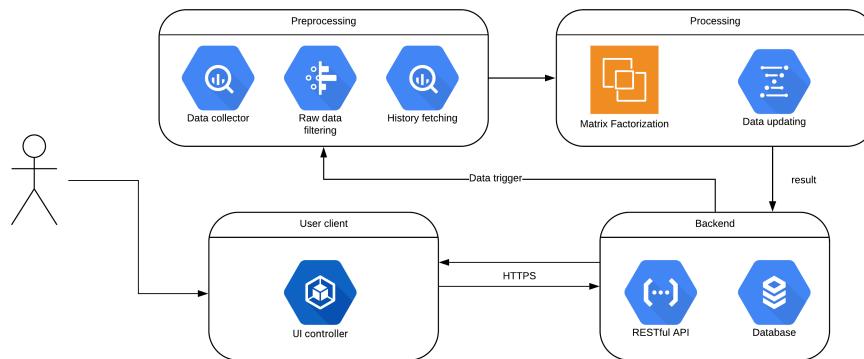
13.5 Áp dụng MFCF vào hệ thống khuyến nghị phim ảnh trên Website xem phim trực tuyến

13.5.1 Dataset cho quá trình huấn luyện

Bộ cơ sở dữ liệu MovieLens 100k (<https://goo.gl/BzHgtq>) được công bố năm 1998 bởi GroupLens (<https://grouplens.org>). Bộ cơ sở dữ liệu này bao gồm 100,000 (100k) đánh giá từ 943 user cho 1682 bộ phim. Chúng ta có thể sử dụng bộ dữ liệu trên để thực hiện huấn luyện Model cho hệ thống khuyến nghị, chi tiết mã nguồn và hiện thực Model xem tại [đây](#). Một số file quan trọng trong bộ dữ liệu này :

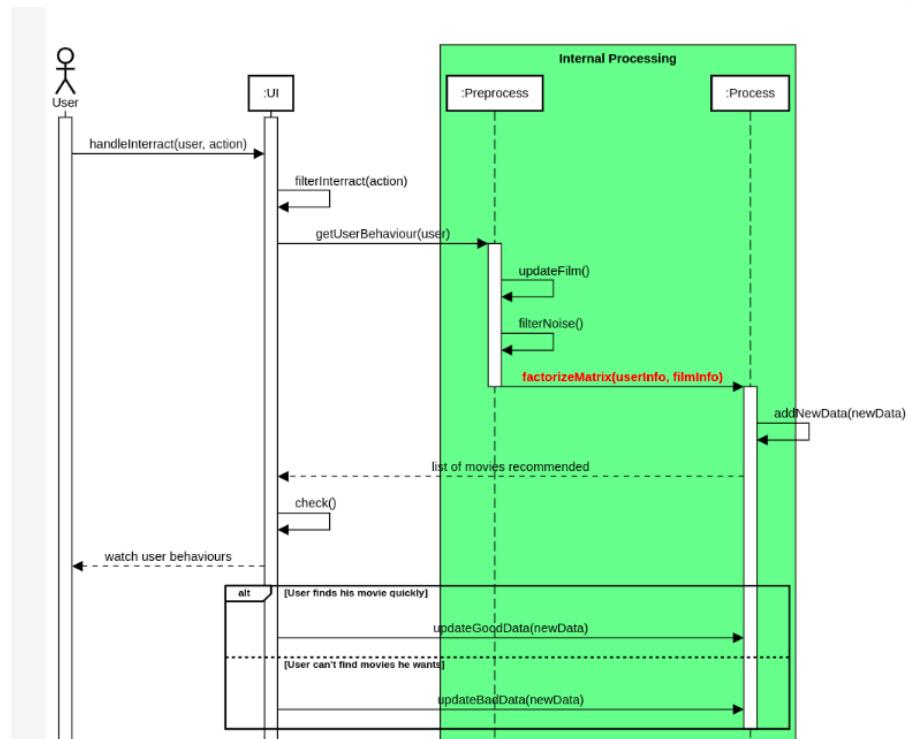
- **u.data**: Chứa toàn bộ các đánh giá của 943 người dùng cho 1682 bộ phim. Mỗi người dùng đánh giá ít nhất 20 movie.
 - **u.genre** : Chứa tên của 19 thể loại phim. Các thể loại bao gồm: unknown, Action, Adventure, Animation, Children's, Comedy, Crime, Documentary, Drama, Fantasy, Film-Noir, Horror, Musical, Mystery, Romance, Sci-Fi, Thriller, War, Western.
 - **u.item**: Thông tin về mỗi bộ phim. Trong mỗi dòng, chúng ta sẽ thấy id của phim, tên phim, ngày phát hành, link trên imdb, và các số nhị phân 0, 1 phía cuối để chỉ ra bộ phim thuộc các thể loại nào trong 19 thể loại đã cho trong u.genre. Một bộ phim có thể thuộc nhiều thể loại khác nhau. Một vài dòng đầu tiên của file:

13.5.2 Kiến trúc của hệ thống khuyến nghị



Hình 13.2: Kiến trúc của hệ thống khuyến nghị trong một ứng dụng xem phim trực tuyến

13.5.3 Sequence Diagram quá trình hoạt động của hệ thống khuyến nghị



Hình 13.3: Sequence Diagram quá trình hoạt động của hệ thống khuyến nghị

13.6 Bài tập

Bài tập 13.1. Nêu hai nhược điểm cơ bản của phương pháp Content - based.

Bài tập 13.2. Tương tự như ví dụ ở phần **12.2.3** áp dụng trên ma trận Utility cho hệ thống khuyến nghị phim ảnh (**Hình 13.1**), hãy xây dựng hàm mất mát cho các User còn lại (**A, B, C, D, E**) theo phương pháp Content - based.

Bài tập 13.3. Thực tế cho thấy trên các ứng dụng tương tác, Utility Matrix thay đổi liên tục vì có thêm users, items cũng như các ratings mới hoặc user muốn thay đổi ratings của họ, thậm chí ta còn dựa trên tương tác của người dùng trên ứng dụng để đánh giá mức độ yêu thích, vì vậy hai ma trận **X** và **W** thường xuyên được cập nhật. Điều này đồng nghĩa với việc ta không thể đưa ra khuyến nghị ngay lập tức vì quá trình training vốn tốn khá nhiều thời gian. Hãy đề xuất giải pháp cập nhật User model **W** nhằm khắc phục vấn đề này

Gợi ý : Trong các lựa chọn về số điểm dữ liệu sử dụng để cập nhật mô hình trong Gradient Decent, lựa chọn nào giúp quá trình cập nhật hệ số trên 1 vòng lặp là nhanh nhất ?.

Chương 14

Giải thuật ARIMA

14.1 Giới thiệu

14.1.1 Bài toán mở đầu

Trong đời sống tự nhiên, khi chuồn chuồn bay thấp hoặc khi thấy mây đen xuất hiện trên bầu trời, ta dự đoán rằng trời sắp mưa. Trong lĩnh vực chứng khoán, người kinh doanh dựa trên những biểu đồ kinh tế của các công ty để đưa ra quyết định đầu tư. Tương tự như vậy, vô số các lĩnh vực khác thường sử dụng các dữ liệu từ quá khứ để có thể dự báo các giá trị ở tương lai để phục vụ cho lợi ích của mình.

Trong lĩnh vực học máy (machine learning), nhiều mô hình dự đoán đã được tạo ra để sử dụng trong đời sống. ARIMA là một trong nhiều mô hình được sử dụng dựa trên dữ liệu là các chuỗi thời gian (time series).

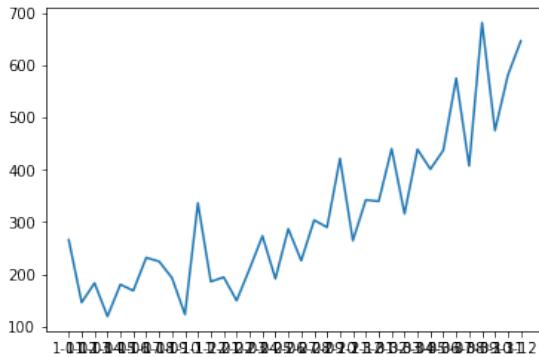
14.1.2 Giới thiệu về chuỗi thời gian

Dự báo chuỗi thời gian là một trong những mảng đề tài lớn của thống kê, kinh tế lượng và machine learning. Các dự báo này được sử dụng rộng rãi và có một số ứng dụng tiêu biểu như dự báo nhu cầu thị trường, dự báo chứng khoán, dự báo giá dầu mỏ và nhiều ứng dụng trong các lĩnh vực khác.

Trong dự báo, chuỗi thời gian là một tập hợp các dữ liệu quan sát được thực hiện tuần tự theo thời gian. Hình 14.1 là một ví dụ rõ ràng về một chuỗi thời gian.

Ta giả sử một dãy các giá trị Y_1, Y_2, \dots, Y_T có ký hiệu là $\{Y_t, t \in T\}$. Chuỗi thời gian này khi được sử dụng trong machine learning sẽ được sửa đổi để có các đặc trưng và thường sẽ đáp ứng các yêu cầu về:

- Chuỗi dừng: Một chuỗi thời gian được gọi là có tính dừng khi nó có tính chất lặp lại (chu kỳ):
 - Kỳ vọng không đổi theo thời gian: $E(Y_t) = \mu$
 - Phản sai (Variance) không đổi theo thời gian: $Var(Y_t) = E((Y_t - \mu)^2) = \sigma^2$
- Nhiều trắng (*White noise*): Một chuỗi được gọi là nhiều trắng nếu kì vọng của nó bằng 0, các giá trị với các độ trễ khác nhau không có hiện tượng tự tương quan và phản sai sai số không đổi. Nó thể hiện cho tính không thể dự báo của model do không có quy luật.



Hình 14.1: Một chuỗi thời gian (time series) về doanh số của một cửa hàng trong 36 tháng

- **Tự tương quan (Auto Correlation Regression):** Hầu hết các chuỗi thời gian sẽ có sự tương quan với giá trị trễ của nó. Các giá trị càng gần nhau hoặc cùng thuộc chu kỳ thì tương quan càng mạnh. Hệ số này thường dùng để tìm ra độ trễ cần thiết để xây dựng mô hình.

Trong trường hợp đặc biệt, nếu quá khứ lặp lại ở tương lai, ta sẽ dùng dữ liệu quá khứ để giải thích cho hiện tại. Để xử lý vấn đề trên, ta có thể sử dụng một số mô hình như: ARIMA, SARIMA, ARIMAX, và GARCH.

14.1.3 Giới thiệu về autocorrelation function

Phương sai (variance) của một chuỗi Y_t (kí hiệu là $Var[Y_t]$) quen thuộc mà chúng ta đã biết là một trường hợp đặc biệt của hiệp phương sai tự động (auto-covariance) khi mà độ trễ (lag) $k = 0$. Hiệp phương sai tự động (auto-covariance) chỉ phụ thuộc vào độ trễ k :

$$Cov[Y_t, Y_{t+k}] = E[(X_t - \mu)(X_{t+k} - \mu)] = \gamma_k, \forall t \quad (14.1)$$

Một tập hợp của tất cả các hệ số γ_k trên tạo thành phương trình tự tương quan (autocorrelation function) của quá trình. Trong đó, $\gamma_0 = \sigma^2$. Ngoài ra, ACF thường được chuyển hóa về dưới dạng $\{\rho_t\}$, $k = 0, 1, 2, \dots$ với $\rho_t = \frac{\gamma_k}{\gamma_0}$. Lúc này, Với p phần tử, các phần tử trong ACF lúc này sẽ có dạng:

$$\rho_1 = \alpha_1 + \alpha_2\rho_1 + \dots + \alpha_p\rho_{p-1} \quad (14.2)$$

$$\rho_2 = \alpha_1\rho_1 + \alpha_2 + \dots + \alpha_p\rho_{p-2} \quad (14.3)$$

...

$$\rho_p = \alpha_1\rho_{p-1} + \alpha_2\rho_{p-2} + \dots + \alpha_p \quad (14.4)$$

Giả sử $p = 1$ thì $\rho_1 = \alpha_1 = a_1$. Cứ tiếp tục như vậy, ta tìm được một chuỗi a_1, a_2, a_3, \dots . Ta gọi chuỗi số này là đặc trưng tự tương quan riêng phần (partial autocorrelation function - PACF) của chuỗi thời gian. ACF, PACF là các thành phần quan trọng giúp chúng ta tìm ra hệ số của mô hình ARIMA.

14.1.4 ARIMA

ARIMA hay còn gọi là “*Auto Regressive Integrated Moving Average*” là lớp các mô hình mà dự đoán một khoảng thời gian được cho dựa trên giá trị quá khứ của nó kết hợp cùng với độ trễ của nó và các lỗi dự đoán do độ trễ.

Nói một cách dễ hiểu hơn, khi đáp ứng các điều kiện chuỗi dừng và phương sai sai số không đổi cùng như một vài yếu tố khác như ở 14.1.2, mô hình này sẽ lấy đầu vào là các dữ liệu từ quá khứ để dự báo cho tương lai. Các dữ liệu này bao gồm:

- Chuỗi tự hồi quy AR (auto regression) được đặc trưng bằng p
- Chuỗi trung bình trượt MA (moving average) được đặc trưng bằng q
- Trong đa số trường hợp, các dữ liệu thường không có tính dừng. Lúc này, ta sẽ cần biến đổi sang chuỗi dừng bằng sai phân I (order of integration) được đặc trưng bằng d. Đây chính là đặc trưng thứ 3 của ARIMA

Vậy ARIMA(p, d, q) được đặc trưng bởi ba tham số p, d, q.

14.2 Lý thuyết mô hình ARIMA

Các chuỗi thời gian đều có sự tương quan giữa giá trị trong quá khứ với giá trị hiện tại. Mức độ tương quan càng lớn khi chuỗi càng gần thời điểm hiện tại. Trong mô hình ARIMA, ta sẽ tìm cách đưa vào các biến trễ (lag) nhằm tạo ra một mô hình dự báo có khả năng dự đoán với tỉ lệ tốt hơn.

ARIMA được cấu tạo nên từ hai mô hình là Auto Regressive (AR) và Moving Average (MA).

Trong mô hình Auto Regressive, Y_t bị phụ thuộc trực tiếp bởi độ trễ của bản thân nó và có phương trình:

$$Y_t = \alpha + \beta_1 Y_{t-1} + \beta_2 Y_{t-2} + \dots + \beta_p Y_{t-p} + \epsilon_t \quad (14.5)$$

Trong đó,

- Y_{t-1} là giá trị quá khứ lùi về 1 của Y_t
- β là hệ số độ trễ 1 mà mô hình xấp xỉ
- α là giá trị chẵn cung được mô hình xấp xỉ
- ϵ_t là độ nhiễu trắng

Còn với mô hình Moving Average, Y_t chỉ phụ thuộc vào sai phân dự đoán độ trễ (lagged forecast errors) và có phương trình:

$$Y_t = \alpha + \epsilon_t + \phi_1 \epsilon_{t-1} + \phi_2 \epsilon_{t-2} + \dots + \phi_q \epsilon_{t-q} \quad (14.6)$$

Trong đó,

- α là trung bình của chuỗi
- $\phi_1..\phi_q$ là các tham số cho sẵn
- $\epsilon_t..\epsilon_{t-q}$ là độ nhiễu trắng

Từ hai mô hình trên cùng với việc lấy sai phân hợp lí, ta kết hợp nó lại để được mô hình ARIMA:

$$Y_t = \alpha + \beta_1 Y_{t-1} + \beta_2 Y_{t-2} + \dots + \beta_p Y_{t-p} + \epsilon_t + \phi_1 \epsilon_{t-1} + \phi_2 \epsilon_{t-2} + \dots + \phi_q \epsilon_{t-q} \quad (14.7)$$

14.2.1 Giải quyết tính dừng

Để có thể sử dụng model ARIMA, bước đầu tiên là cần làm cho chuỗi dữ liệu (chuỗi thời gian) trở thành chuỗi dừng (time series stationary). Bởi vì, mô hình ARIMA sử dụng mô hình hồi qui tuyến tính (linear regression model) mà sử dụng độ trễ của nó để dự đoán. Với việc ứng dụng mô hình đó, nó hoạt động tốt nhất khi các dự đoán có sự tương quan và độc lập với nhau.

Để giải quyết vấn đề này, cách tiếp cận thông thường nhất chính là “difference it”, nghĩa là chúng ta sẽ trừ giá trị hiện tại cho giá trị quá khứ. Đôi lúc, nếu chuỗi quá phức tạp, chúng ta cần nhiều hơn một lần lấy sai phân(differencing).

Giá trị d là số lượng tối thiểu sai phân (diferencing) cần để làm cho chuỗi trở thành chuỗi dừng. Và nếu chuỗi thời gian đã có tính dừng thì $d = 0$.

Trước khi sai phân, ta phải đảm bảo rằng chuỗi được sử dụng là không có tính dừng. Để làm việc này, ta cần dùng một số công cụ kiểm định như Augmented Dickey Fuller test áp dụng lên mẫu dữ liệu. Sau đó, sử dụng output là P -value để đánh giá chuỗi dữ liệu này. Nếu P -value > 0.05 , chuỗi được cho là không có tính dừng và ta có thể bắt đầu tìm sai phân để làm cho chuỗi có tính dừng.

Bình thường, một chuỗi dữ liệu chỉ cần 1 đến 2 lần lấy sai phân. Trong trường hợp nếu sau khi lấy sai phân, các giá trị của biểu đồ tự tương quan chạy đến vùng âm khá nhanh, chuỗi đã cho có thể đã bị lấy sai phân quá đà. Khi này, ta cần giảm trị d đến khi không còn tình trạng trên và có thể chấp nhận giá trị d đó.

Để tránh các bước thủ công trên, ta có thể sử dụng hàm ndiffs kết hợp cùng với các loại kiểm định khác để chương trình tự tính toán giá trị d phù hợp để chuỗi trở thành chuỗi dừng.

14.2.2 Auto Regressive và Moving Average

Quá trình tự hồi qui (AR) bậc p khi quan sát giá trị hiện tại X_t được tạo ra bằng trọng số trung bình của p giá trị quan sát ở quá khứ kết với giá trị đặc trưng cho sự nhiễu loạn ngẫu nhiên tại hiện tại.

Một chuỗi thời gian $\{X_t\}$ là một quá trình tự hồi qui bậc p (AR(p)) nếu với mỗi quan sát X_t được kí hiệu như sau:

$$Y_t = \alpha_1 Y_{t-1} + \alpha_2 Y_{t-2} + \dots + \alpha_p Y_{t-p} + \delta + \epsilon_t \quad (14.8)$$

Trong đó,

- ϵ_t là các nhiễu trắng với $E[\epsilon_t] = 0$, phương sai σ^2 và hiệp phương sai $y_k = 0, k \neq 0$
- δ (or ϕ_0) là một hằng số liên quan đến giá trị trung bình (mean) của quá trình ngẫu nhiên, có thể bằng không.

Để có thể biết được liệu chúng ta có cần giá trị p trong mô hình Auto Regressive, ta cần quan sát biểu đồ tự tương quan riêng phần (Partial Autocorrelation - PACF).

Tự tương quan riêng phần là sự tự tương quan giữa chuỗi dữ liệu và độ trễ của nó, sau khi đã bao gồm các giá trị độ trễ trung gian. Từ đó, chúng ta biết được độ trễ đó có cần thiết phải sử dụng hay không.

Giống với mô hình trên, Moving Average cũng cần xác định một hệ số q . Hệ số này có thể được quan sát thông qua biểu đồ ACF. Biểu đồ ACF cho chúng ta biết được rằng cần bao nhiêu q để có thể xóa các sự tự tương quan trong chuỗi đã trở thành chuỗi dừng.

Hai hệ số này sẽ được nói rõ hơn khi chúng ta tiến vào phần hiện thực.

14.3 Xây dựng mô hình ARIMA trong Python

Ta sẽ xây dựng mô hình ARIMA trên tập dữ liệu chuỗi thời gian đơn giản sau :
<https://raw.githubusercontent.com/jbrownlee/Datasets/master/shampoo.csv>

Dây là tập dữ liệu về doanh số bán đầu tiên theo tháng trong chu kỳ 3 năm.
Ta đọc tập dữ liệu và xem thử kết quả 5 dòng đầu của tập :

```
import pandas as pd

df = pd.read_csv("""https://raw.githubusercontent.com/
jbrownlee/Datasets/master/shampoo.csv""",
index_col=0)

print(df.head())
```

Month	Sales
1-01	266.0
1-02	145.9
1-03	183.1
1-04	119.3
1-05	180.3

14.3.1 Tìm bậc của sai phân

Một trong những điều kiện tiền đề khi hồi qui các mô hình chuỗi thời gian đó là chuỗi phải dừng. Chúng ta chỉ lấy sai phân khi chuỗi ban đầu không có tính dừng, ngược lại thì không cần lấy sai phân, nghĩa là $d=0$.

Để kiểm định tính dừng chúng ta có thể sử dụng kiểm định Argument Dickey Fuller (ADF) hay còn gọi là kiểm định nghiệm đơn vị. Giả thiết null (H_0) của kiểm định ADF là chuỗi thời gian không có tính dừng. Vì vậy nếu giá trị p của kiểm định nhỏ hơn mức ý nghĩa (0.05) thì ta sẽ bác bỏ giả thiết H_0 và kết luận chuỗi thời gian đó có tính dừng.

Trên python đã hỗ trợ kiểm định ADF thông qua package statsmodels

```
from statsmodels.tsa.stattools import adfuller
result = adfuller(df)
print('ADF Statistic: %f' % result[0])
print('p-value: %f' % result[1])
```

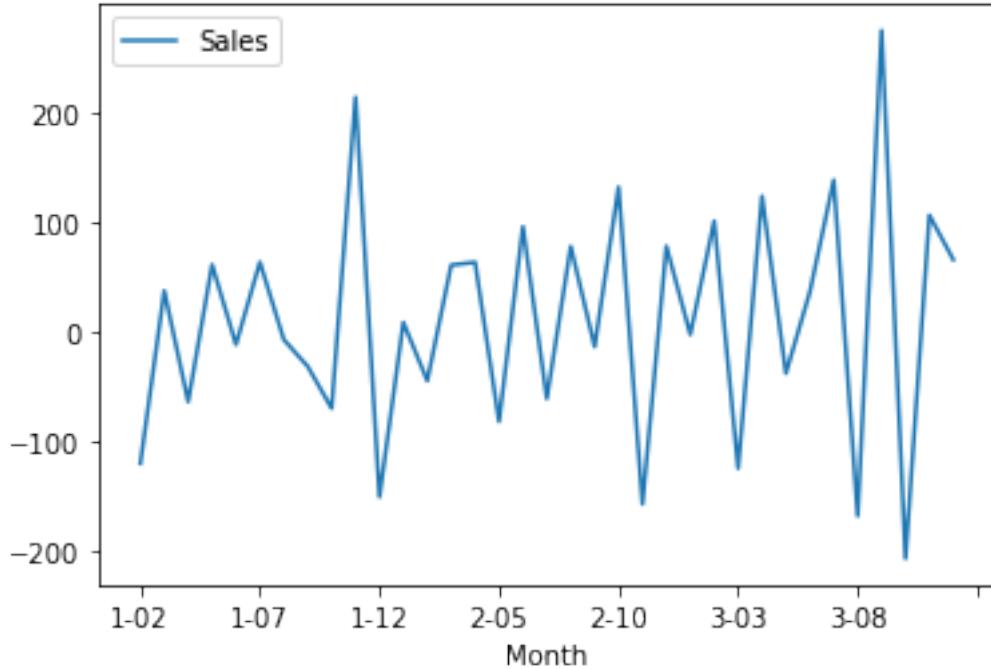
```
ADF Statistic: 3.060142
p-value: 1.000000
```

Ta có thể thấy giá trị p (p-value) khá cao, vì vậy ta sẽ lấy sai phân một lần và chạy lại bài kiểm định tính dừng.

```
data_diff = df.diff()[1:]
result = adfuller(data_diff)
print('ADF Statistic: %f' % result[0])
print('p-value: %f' % result[1])
data_diff.plot()
```

14.3. XÂY DỰNG MÔ HÌNH ARIMA TRONG PYTHON

```
ADF Statistic: -7.249074
p-value: 0.000000
```



Hình 14.2: Đồ thị của dữ liệu sau khi lấy sai phân bậc 1

Từ giá trị p cũng như đồ thị ta có thể thấy chuỗi thời gian của ta sẽ có tính dừng sau một lần sai phân. Vì vậy ta sẽ chọn $d = 1$.

14.3.2 Tìm bậc của tham số MA (q)

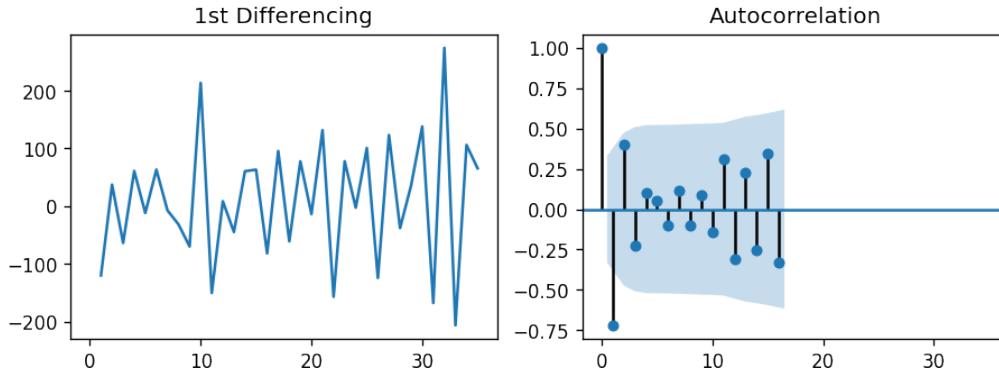
Tự tương quan (ACF - AutoCorrelation Function): Tự tương quan là một khái niệm quan trọng trong chuỗi thời gian. Hầu hết các chuỗi thời gian sẽ có sự tương quan với giá trị trễ của nó và các giá trị càng gần nhau thì tương quan càng mạnh hoặc các giá trị cùng thuộc 1 chu kỳ của chuỗi thì sẽ có tương quan cao (chẳng hạn như cùng tháng trong chu kỳ năm hay cùng quý trong chu kỳ năm). Chính vì vậy hệ số này mới có tên là tự tương quan. Hệ số tự tương quan được viết tắt là ACF và thường dùng để tìm ra độ trễ của quá trình trung bình trượt MA(q) để xây dựng các mô hình như ARIMA, GARCH, ARIMAX,... và kiểm tra yếu tố mùa vụ.

Dể tìm q, ta sẽ nhìn vào biểu đồ ACF và xác định tại lag nào mà phần lớn các lag phía sau không còn nhiều ý nghĩa.

```
import pandas as pd
from statsmodels.graphics.tsaplots import plot_acf, plot_pacf
import matplotlib.pyplot as plt
plt.rcParams.update({'figure.figsize':(9,3), 'figure.dpi':120})

fig, axes = plt.subplots(1, 2, sharex=True)
axes[0].plot(df.Sales.diff()[1:]); axes[0].set_title('1st Differencing')
```

```
plot_acf(df.Sales.diff().dropna(), ax=axes[1])
plt.show()
```



Hình 14.3: Đồ thị của dữ liệu sai phân bậc 1 (trái) và đồ thị ACF tương ứng (phải)

Từ đồ thị ACF trên, ta có thể chọn $q=1$ hoặc $q=2$ vì lag 2 khá sát biên của miền ý nghĩa (vùng màu xanh). Ở đây ta sẽ chọn $q=2$ cho những bước tiếp theo.

14.3.3 Tìm bậc của tham số AR (p)

Tự tương quan riêng phần (PACF - Partial AutoCorrelation Function): Về cơ bản tương quan riêng phần cũng là chỉ số đo lường hệ số tương quan như ACF. Tuy nhiên vẫn có sự khác biệt đó là hệ số tương quan này loại bỏ ảnh hưởng của các chuỗi độ trễ trung gian. Để tìm p , ta cũng sẽ làm giống như tìm q là từ đồ thị PACF để xác định tại lag nào mà phần lớn các lag phía sau không còn nhiều ý nghĩa.

```
# PACF plot of 1st differenced series
plt.rcParams.update({'figure.figsize':(9,3), 'figure.dpi':120})

fig, axes = plt.subplots(1, 2, sharex=True)
axes[0].plot(df.Sales.diff()); axes[0].set_title('1st Differencing')
#axes[1].set(ylim=(0,5))
plot_pacf(df.Sales.diff().dropna(), ax=axes[1])

plt.show()
```

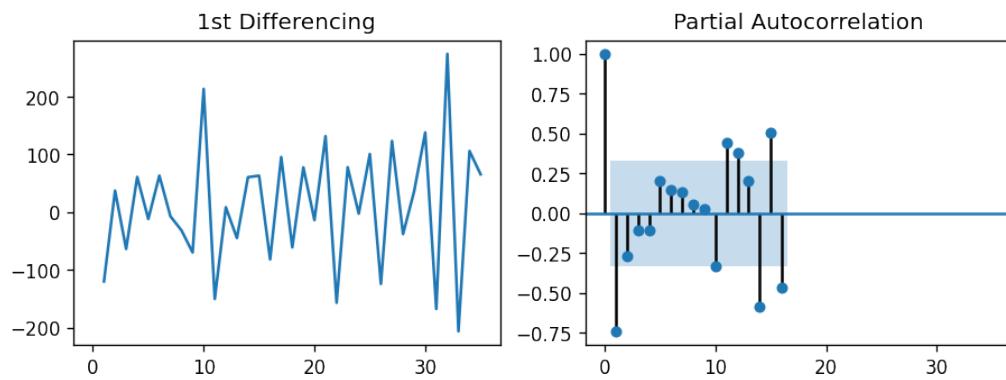
Từ đồ thị ta chọn $p=1$

14.3.4 Xây dựng mô hình ARIMA

Sau khi xác định được bậc của p , d , q thì mô hình ARIMA có thể được xây dựng khá dễ dàng trên python thông qua package statsmodels. Điều mà chúng ta cần thực hiện chỉ là khai báo bậc của mô hình ARIMA. Giả sử cần xây dựng một mô hình ARIMA(1, 1, 2) ta thực hiện như sau:

```
from statsmodels.tsa.arima_model import ARIMA
```

14.3. XÂY DỰNG MÔ HÌNH ARIMA TRONG PYTHON



CHƯƠNG 14. GIẢI THUẬT ARIMA

Ta có thể thấy hệ số ước lượng của biến AR1 khá gần 0 và giá trị p-value của nó (0.623) lớn hơn khá nhiều so với 0.05. Vì vậy hãy xây dựng lại mô hình ARIMA với việc bỏ đi biến AR1.

```
# 0,1,2 ARIMA Model  
model = ARIMA(df.Sales, order=(0,1,2))  
model_fit = model.fit(disp=0)  
print(model_fit.summary())
```

```
ARIMA Model Results  
=====  
Dep. Variable: D.Sales No. Observations: 35  
Model: ARIMA(0, 1, 2) Log Likelihood: -195.803  
Method: css-mle S.D. of innovations: 61.594  
Date: Mon, 13 Jul 2020 AIC: 399.606  
Time: 13:57:22 BIC: 405.828  
Sample: 1 HQIC: 401.754  
=====  
coef std err z P>|z| [0.025 0.975]  
-----  
const 10.1715 6.423 1.584 0.113 -2.418 22.761  
ma.L1.D.Sales -1.3068 0.164 -7.956 0.000 -1.629 -0.985  
ma.L2.D.Sales 0.8950 0.197 4.535 0.000 0.508 1.282  
Roots  
=====  
Real Imaginary Modulus Frequency  
-----  
MA.1 0.7300 -0.7644j 1.0570 -0.1287  
MA.2 0.7300 +0.7644j 1.0570 0.1287  
=====
```

Giá trị AIC của mô hình đã giảm bớt và các giá trị p-value cũng đã được cải thiện. Ta có thể chấp nhận mô hình ARIMA(0, 1 ,2) phù hợp với chuỗi thời gian của mình.

14.3.5 Phương pháp Auto ARIMA

Chúng ta thấy rằng việc lựa chọn mô hình tốt nhất chỉ đơn thuần dựa trên chỉ số AIC, khá đơn giản. Do đó chúng ta hoàn toàn có thể tự động thực hiện qui trình này. Trên python đã hỗ trợ tìm kiếm mô hình ARIMA phù hợp thông qua package auto arima. Chúng hoạt động như một grid search mà tham số chúng ta truyền vào chỉ là các hệ số giới hạn trên của các bậc (p,d,q). Mọi việc còn lại hãy để thuật toán tự giải quyết. Nếu bạn làm quen với R thì cũng hỗ trợ chức năng auto ARIMA tương tự như vậy. Để sử dụng phương pháp auto arima chúng ta phải dùng tới package pmdarima.

Xây dựng phương trình hồi qui theo phương pháp Auto ARIMA:

```
from statsmodels.tsa.arima_model import ARIMA  
import pmdarima as pm  
  
model = pm.auto_arima(df.Sales, start_p=1, start_q=1,  
                      test='adf',      # use adftest to find optimal 'd'  
                      max_p=3, max_q=3, # maximum p and q  
                      m=1,            # frequency of series
```

14.3. XÂY DỰNG MÔ HÌNH ARIMA TRONG PYTHON

```
d=None,           # let model determine 'd'  
seasonal=False, # No Seasonality  
start_P=0,  
D=0,  
trace=True,  
error_action='ignore',  
suppress_warnings=True,  
stepwise=True)  
  
print(model.summary())
```

Ở đây chúng ta sẽ khai báo điểm bắt đầu và kết thúc của p,q, bậc của d sẽ được quyết định bằng bài kiểm định ADF và phương pháp điều chỉnh mô hình là stepwise.

Chiến lược stepwise: Là một chiến lược được sử dụng khá phổ biến trong việc lựa chọn biến cho mô hình hồi qui. Chiến lược này sẽ trải qua nhiều bước. Mỗi một bước tìm cách thêm vào hoặc bớt đi một biến giải thích nhằm tạo ra một mô hình mới sao cho sai số giảm so với mô hình gốc. Như vậy càng qua nhiều bước mô hình sẽ càng chuẩn xác hơn và sau khi kết thúc chiến lược ta sẽ thu được một mô hình cuối cùng là mô hình tốt nhất. Tiêu chuẩn để đo lường sai số và ra quyết định lựa chọn thường là AIC.

Trong trường hợp mô hình có yếu tố mùa vụ thì ta sẽ cần thiết lập seasonal = True và kết hợp thêm chu kỳ của mùa vụ. Chẳng hạn chu kỳ là 12 tháng thì có thể khai báo D = 12. Khi đó mô hình ARIMA sẽ trở thành mô hình SARIMA (seasonal ARIMA). Ta thu được kết quả tối ưu nhất từ auto_arima là mô hình ARIMA(0,1,2)

```
SARIMAX Results  
=====  
Dep. Variable: y No. Observations: 36  
Model: SARIMAX(0, 1, 2) Log Likelihood: -195.763  
Date: Mon, 13 Jul 2020 AIC: 399.527  
Time: 14:03:05 BIC: 405.748  
Sample: 0 HQIC: 401.675  
- 36  
Covariance Type: opg  
=====  
coef std err z P>|z| [0.025 0.975]  
-----  
intercept 9.4590 7.825 1.209 0.227 -5.877 24.795  
ma.L1 -1.3227 40.789 -0.032 0.974 -81.267 78.622  
ma.L2 0.9996 61.662 0.016 0.987 -119.856 121.856  
sigma2 3526.2648 2.17e+05 0.016 0.987 -4.22e+05 4.29e+05  
=====  
Ljung-Box (Q): 36.34 Jarque-Bera (JB): 0.04  
Prob(Q): 0.36 Prob(JB): 0.98  
Heteroskedasticity (H): 0.84 Skew: 0.06  
Prob(H) (two-sided): 0.76 Kurtosis: 2.87  
=====  
Warnings:  
[1] Covariance matrix calculated using the outer product of gradients (complex-step).
```

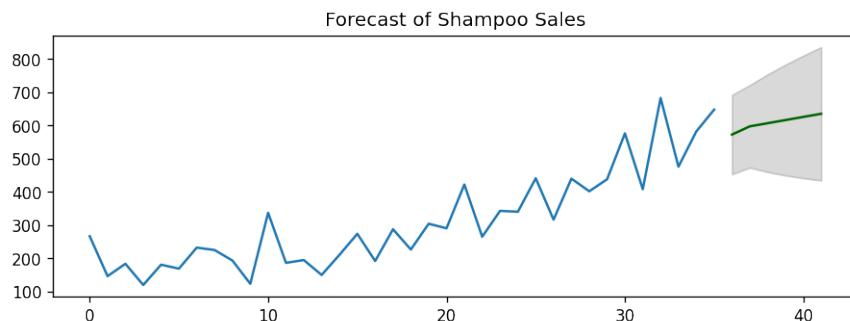
14.3.6 Dự báo

Sau khi đã tìm ra được mô hình ARIMA tốt nhất. Chúng ta sẽ dự báo cho khoảng thời gian tiếp theo. Dự báo cho chuỗi thời gian khá đặc thù và khác biệt so với các lớp mô hình dự báo khác vì giá trị time step liền trước sẽ được sử dụng để dự báo cho time step liền sau. Do đó đòi hỏi phải có một vòng lặp liên tiếp dự báo qua các bước thời gian. Rất may mắn là hàm `predict()` đã tự động giúp ta thực hiện việc đó. Ta sẽ chỉ phải xác định số lượng phiên tiếp theo muốn dự báo là bao nhiêu. Ta sẽ dự báo trước doanh số bán đầu gội của 6 tháng tiếp theo.

```
# Forecast
n_periods = 6
fc, confint = model.predict(n_periods=n_periods, return_conf_int=True)
index_of_fc = np.arange(len(df.Sales), len(df.Sales)+n_periods)

# make series for plotting purpose
fc_series = pd.Series(fc, index=index_of_fc)
lower_series = pd.Series(confint[:, 0], index=index_of_fc)
upper_series = pd.Series(confint[:, 1], index=index_of_fc)

# Plot
plt.plot(df.Sales)
plt.plot(fc_series, color='darkgreen')
plt.fill_between(lower_series.index,
                 lower_series,
                 upper_series,
                 color='k', alpha=.15)
plt.title("Forecast of Shampoo Sales")
plt.show()
```



Hình 14.5: Đồ thị doanh số bán hàng và kết quả dự đoán của 6 tháng tiếp theo

14.4 Ứng dụng ARIMA vào dự án phần mềm

Từ ví dụ trên, ta có thể thấy ARIMA có thể được tích hợp vào các hệ thống phần mềm quản lý bán hàng nhằm đưa ra các dự đoán về doanh số bán hàng theo thời gian để giúp các chủ cửa hàng có thể điều chỉnh chiến lược kinh doanh phù hợp. Ở đây ta sẽ xét ví dụ việc tích hợp ARIMA vào hệ thống Smart Food Court - Nhà ăn thông minh (SFCS).

14.5. BÀI TẬP

SFCS là hệ thống quản lý nhà ăn cho phép người dùng đặt món thông qua các màn hình cảm ứng tại quầy hoặc thông qua ứng dụng di động. Về phía các chủ cửa hàng trong nhà ăn, SFCS cung cấp nhiều tính năng cho phép các chủ cửa hàng quản lý cửa hàng của mình, quản lý các đơn hàng cũng như xem báo cáo kinh doanh,... Đối với hệ thống này, việc tích hợp ARIMA sẽ giúp người bán hàng có thể xem trước những dự đoán về doanh số bán hàng, món ăn nào sẽ được ưa chuộng hơn theo thời gian,... Từ đó các chủ cửa hàng có thể điều chỉnh thực đơn cũng như có các chiến lược kinh doanh hợp lý theo thời gian. Kiến trúc của hệ thống SFCS sẽ gồm ba thành phần chính như sau:

- UI : giao diện của khách hàng cũng như chủ cửa hàng
- SFCS Controller: thành phần xử lý các sự kiện của người dùng cũng như thực thi các tác vụ luận lí của hệ thống
- Database: cơ sở dữ liệu lưu trữ các dữ liệu cần thiết cho hệ thống

Thành phần ARIMA của ta sẽ được tích hợp trong SFCS Controller dưới tên ARIMA_Forecaster như trong Hình 14.6. Thành phần ARIMA sẽ sử dụng dữ liệu bán hàng được cung cấp thông qua thành phần SalesDataManagement của hệ thống SFCS và xây dựng model cũng như đưa ra dự đoán cho chủ cửa hàng. Chi tiết các tương tác giữa các thành phần trong hệ thống đối với chức năng đưa ra dự đoán doanh số bán hàng được trình bày trong biểu đồ tuần tự (Sequence Diagram) trong Hình 14.7.

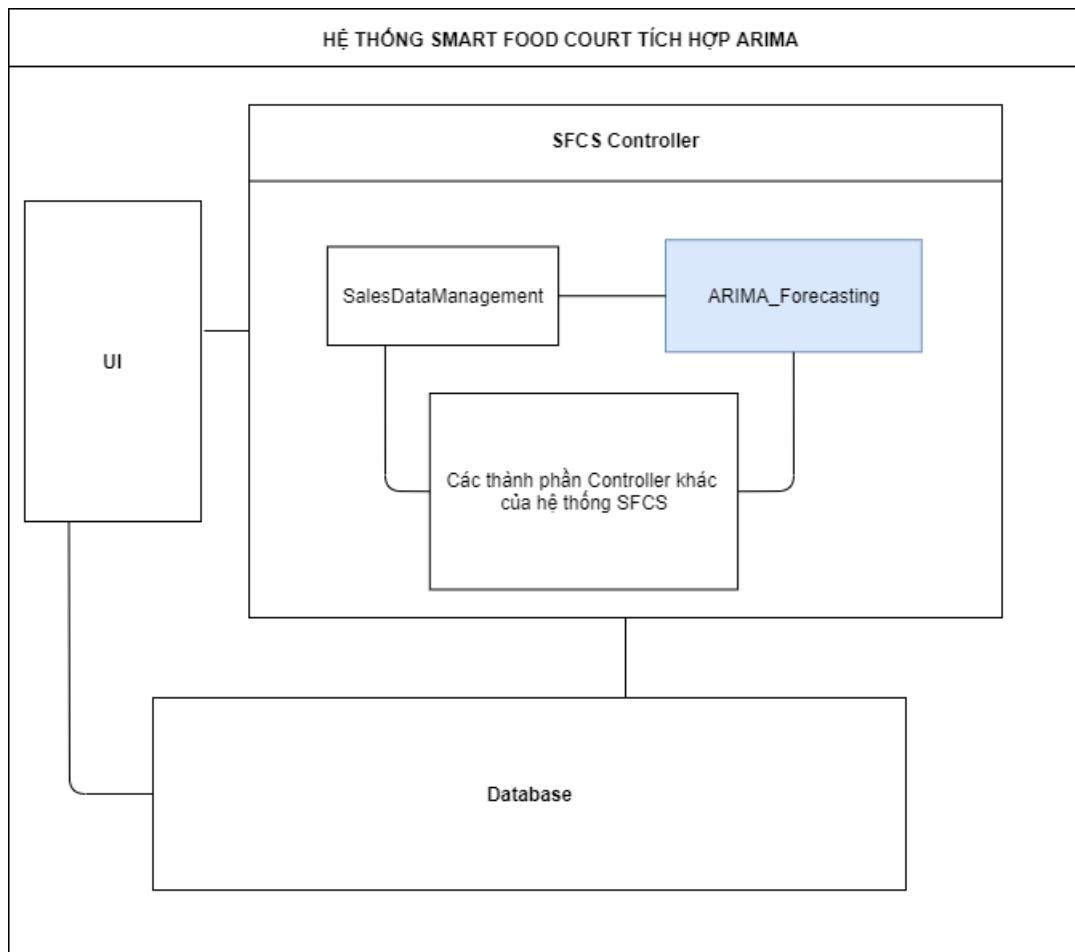
14.5 Bài tập

Câu 1: Ở mô hình AR(1), khi nào thì series xảy ra hiện tượng "zigzag"?

Câu 2: Giải thích sự khác nhau giữa ACF và PACF.

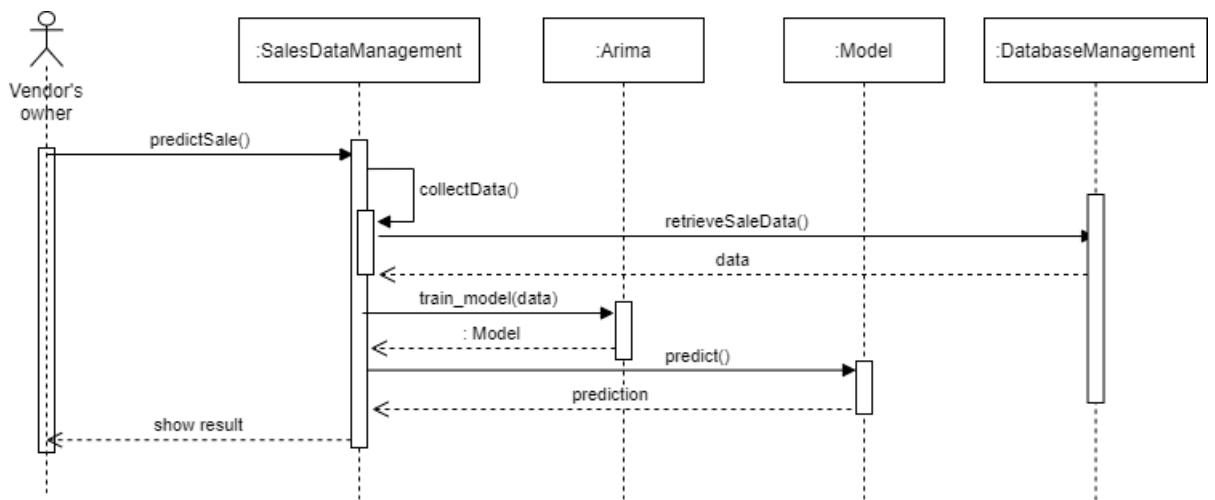
Câu 3: Trong một phần mềm dự báo thời tiết, những thành phần nào có thể dùng mô hình ARIMA áp dụng vào? Vẽ component/package diagram minh họa?

Câu 4: Vì sao mô hình ARIMA thất bại khi có một sự thay đổi bất thường về time series?



Hình 14.6: Kiến trúc phần mềm của hệ thống SFCS tích hợp ARIMA

14.5. BÀI TẬP



Hình 14.7: Biểu đồ tuần tự của chức năng dự đoán doanh số bán hàng

Chương 15

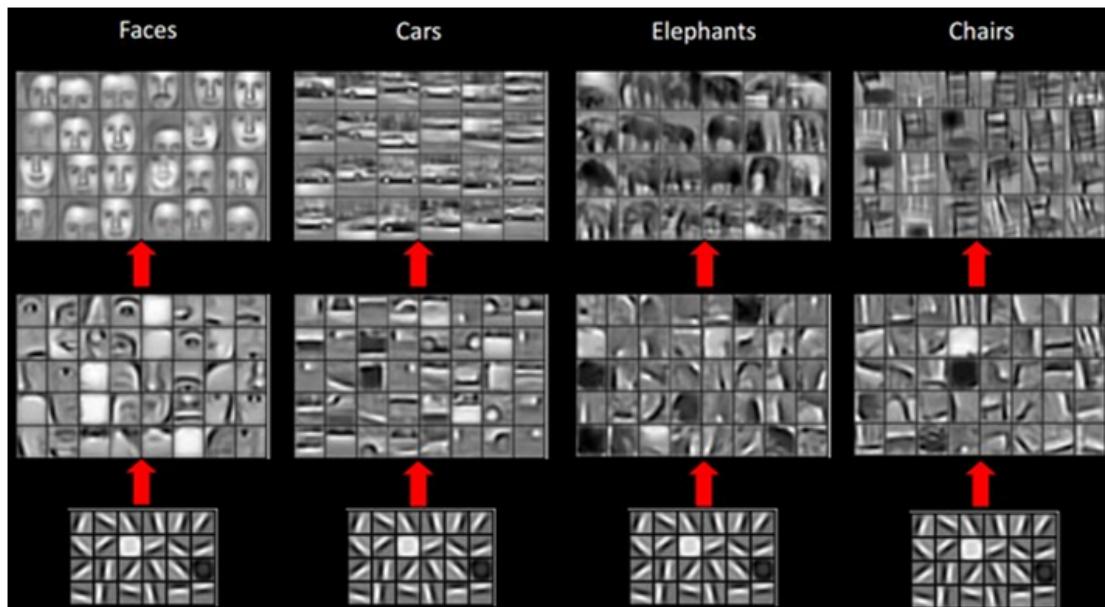
Topic Modeling

15.1 Mạng thần kinh tích chập (Convolutional neural network)

15.1.1 Giới thiệu

Thuật ngữ *Học sâu* (Deep learning) hay *Mạng thần kinh sâu* (Deep neural network) dùng để đề cập đến *Mạng thần kinh nhân tạo* (Artificial neural network - ANN) với nhiều lớp khác nhau. Trong một vài thập kỷ gần đây, nó đã trở thành một trong số những công cụ mạnh nhất và trở nên rất phổ biến với các nhà khoa học với khả năng xử lý được một lượng cực lớn dữ liệu. Các nghiên cứu về các mạng thần kinh sâu gần đây đã vượt qua hiệu suất của các phương pháp cổ điển trong nhiều lĩnh vực khác nhau, đặc biệt là trong vấn đề nhận dạng mẫu. Một trong số những mạng thần kinh sâu phổ biến nhất hiện nay là *Mạng thần kinh tích chập* (Convolutional neural network - CNN)[97]. Mạng thần kinh này sử dụng các phép toán tuyến tính giữa các ma trận gọi là tích chập. CNN có nhiều lớp khác nhau, bao gồm lớp tích chập (convolutional layer), lớp phi tuyến tính (non-linearity layer), lớp tổng hợp (pooling layer) và lớp kết nối đầy đủ (fully-connected layer). Lớp tích chập và lớp kết nối đầy đủ chứa các tham số, còn các lớp phi tuyến tính và lớp tổng hợp không có tham số.

Mạng CNN có hiệu suất ấn tượng trong việc giải quyết các vấn đề Học máy. Nó đã có bước đột phá lớn trong thập kỷ qua trong các lĩnh vực liên quan đến phân loại mẫu, xử lý ảnh, thị giác máy tính, nhận dạng giọng nói[98] và cả xử lý ngôn ngữ tự nhiên. Lợi ích to lớn nhất của mạng CNN là làm giảm số lượng tham số so với mạng ANN thông thường. Những thành tựu này đã thúc đẩy các nhà nghiên cứu và nhà phát triển xây dựng các mô hình lớn hơn để giải quyết những công việc phức tạp, vốn không thể với các mạng ANN cổ điển. Giả định quan trọng nhất về những vấn đề được mạng CNN giải quyết là không nên có các đặc tính phụ thuộc vào không gian. Nghĩa là, trong một ứng dụng nhận diện khuôn mặt, ta không cần để ý vị trí khuôn mặt xuất hiện ở đâu trong bức hình mà cái ta quan tâm là nhận diện chúng bất kể vị trí trong những hình ảnh đầu vào. Một khía cạnh quan trọng khác của CNN, là trích xuất được các đặc điểm trừu tượng của dữ liệu đầu vào làm dữ liệu truyền vào các lớp sâu hơn. Ví dụ, trong một bài toán phân loại ảnh, các cạnh sẽ được xác định ở lớp đầu tiên, sau đó lớp thứ hai sẽ xác định các hình khối cơ bản và các đặc điểm phức tạp hơn như nhận diện vật thể, khuôn mặt, ... sẽ được xác định ở các lớp tiếp theo.



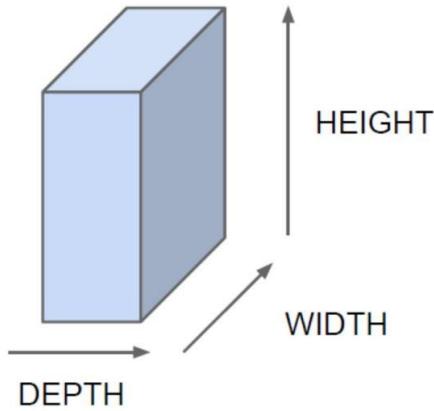
Hình 15.1: Các lớp trong mạng CNN trích xuất các đặc tính từ dữ liệu đầu vào

15.1.2 Lớp tích chập

Giả sử dữ liệu đầu vào của mạng thần kinh có hình dạng như Hình 15.2. Nó có thể là hình ảnh (ví dụ một hình ảnh có chiều rộng và chiều cao là 32x32, với chiều sâu là 3 ứng với số kênh màu RGB), hay video (video chia tỉ lệ với chiều rộng và chiều cao là độ phân giải, còn chiều sâu là số khung hình trong video).

Bây giờ, ta sẽ giả sử dữ liệu đầu vào của mạng thần kinh là các hình ảnh được thể hiện dưới dạng ma trận các điểm ảnh có kích thước là 32x32x3. Để biểu thị hết nội dung của lớp đầu ta cần truyền vào tất cả các điểm ảnh ($32 \times 32 \times 3 = 3072$), do đó lớp này sẽ có 3072 nodes. Để kết nối lớp đầu vào này với một neuron trong lớp ẩn tiếp theo, mạng ANN cổ điển sẽ sử dụng kết nối đầy đủ, tức là có 3072 kết nối có trọng số mới sẽ được tạo ra. Nếu ta thêm vào một neuron mới vào lớp ẩn, ta sẽ tạo ra thêm 3072 kết nối mới nữa và sẽ có tổng cộng 6144 kết nối. Ta phải sử dụng tối hơn 6000 kết nối chỉ để kết nối tới 2 neuron. Tuy nhiên, 2 neuron là không đủ để giải quyết bài toán xử lý ảnh hiệu quả, vì vậy ta sẽ phải thêm các neuron mới vào mạng thần kinh. Giả sử số lượng neuron trong lớp ẩn là 1000, ta sẽ có tổng cộng $32 \times 32 \times 3 \times 1000 = 3072000$ (hơn 3 triệu) kết nối và đó mới chỉ là số lượng kết nối của lớp đầu vào với lớp ẩn đầu tiên trong mạng thần kinh, chưa kể đến những lớp khác của mạng thần kinh. Hơn nữa, nếu kích thước ảnh tăng lên, ví dụ 512x512 thì số lượng kết nối sẽ tăng lên vô cùng nhanh và mạng thần kinh sẽ bị quá tải. Rõ ràng, cách tiếp cận này hoàn toàn không hiệu quả, ta cần phải tìm một phương án khác hiệu quả hơn.

Để tìm phương pháp hiệu quả hơn, thay vì sử dụng kết nối đầy đủ với toàn bộ hình ảnh đầu vào, ta chỉ tạo kết nối đầy đủ cho 1 vùng nhất định trong hình ảnh, ví dụ một vùng 3x3 nào đó trong hình ảnh. Khi đó, để kết nối tới một neuron ở lớp ẩn tiếp theo, ta chỉ cần tạo 3x3 kết nối mới, thay vì 32x32 kết nối mới như trước đây. Nếu lớp ẩn có 1000 neuron, ta sẽ có tổng cộng $3 \times 3 \times 3 \times 1000 = 27000$ kết nối được tạo ra. So sánh với 3072000 kết nối được đề cập ở trên, số lượng kết nối đã được giảm đáng kể. Phương pháp tiếp cận này đã đem lại hiệu quả cao hơn rất nhiều so với các phương pháp cổ điển. Tuy nhiên, số lượng kết nối vẫn còn khá lớn, buộc mạng



Hình 15.2: Dữ liệu đầu vào 3 chiều

thần kinh phải xử lý nhiều tính toán phức tạp, đưa ra các kết quả đầu ra chậm và tốn nhiều thời gian huấn luyện để đạt được hiệu quả mong muốn.

15.1.2.1 Ma trận tích chập

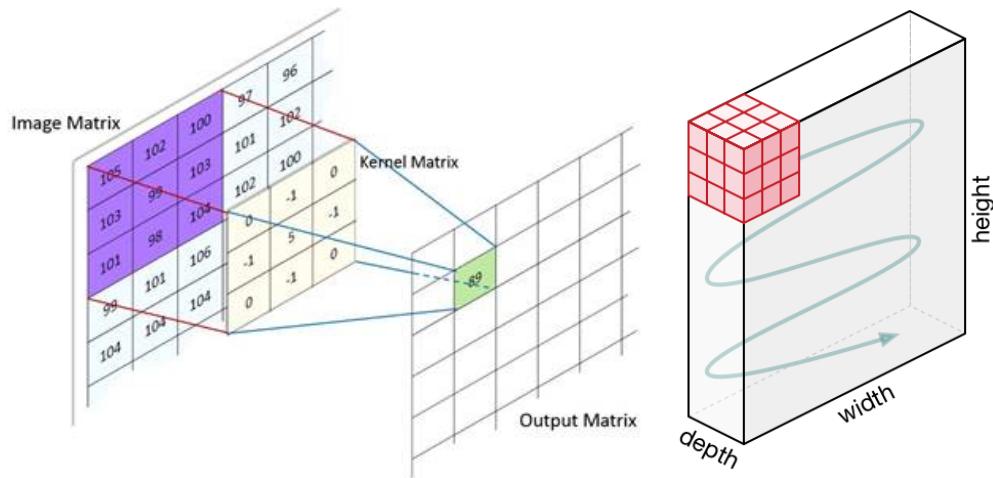
Để đơn giản hóa phương pháp tiếp cận ở trên, thay vì tạo ra các kết nối dãy đủ cho từng vùng 3×3 trong hình ảnh, ta sẽ cố định các trọng số vào một ma trận tích chập (convolution matrix) gọi là lõi (kernel) và sử dụng lõi này để tích chập và kết nối các vùng 3×3 trong hình ảnh tới các neuron trong lớp ẩn tiếp theo (Hình 15.3)[99]. Sau đó lõi này sẽ truyền từ từ qua tất cả dữ liệu trong hình ảnh và kết nối tới các neuron tiếp theo (Hình 15.3). Một lợi ích cực kì to lớn của phương pháp này là số lượng tham số cần phải tính toán chỉ phụ thuộc vào kích thước lõi mà không phụ thuộc vào kích thước dữ liệu đầu vào hay số lượng neuron trong lớp tiếp theo, và bằng đúng kích thước của lõi. Trong ví dụ này, số lượng tham số cần tính toán sẽ là $3 \times 3 \times 3 = 27$ bất kể kích thước của hình ảnh đầu vào và của lớp tiếp theo. Ngoài ra, cách tiếp cận này còn cho phép trích xuất các đặc tính quan trọng bất kể vị trí của chúng trong hình ảnh đầu vào, đây là lý do ta gọi đây là lớp tích chập.

Trong trường hợp tổng quát, các neuron trong lớp ẩn sẽ được tính theo công thức sau:

$$net[i, j] = (\chi * w)[i, j] = \sum_{k=i}^{k+m-1} \sum_{l=j}^{l+m-1} \chi[k, l] * w[k, l]$$

Trong đó:

- net : Ma trận các neuron của lớp tích chập
- χ : Ma trận hình ảnh đầu vào
- w : Ma trận tích chập
- m : Kích thước của ma trận tích chập



Hình 15.3: Sử dụng ma trận tích chập để kết nối các neuron

Dể cho thấy hiệu quả đáng kinh ngạc của ma trận tích chập, Hình 15.4 đã mô tả điều gì sẽ xảy ra nếu chúng ta chọn thủ công các trọng số trong ma trận tích chập 3×3 . Ta có thể thấy trong Hình 15.4, ma trận này có thể được thiết lập để phát hiện các cạnh trong hình ảnh hay làm mờ hình ảnh. Các ma trận này còn được gọi là bộ lọc bởi vì chúng hoạt động như các bộ lọc trong kỹ thuật xử lý ảnh. Tuy nhiên, trong mạng thần kinh tích chập các ma trận này được khởi tạo và huấn luyện cùng với mạng thần kinh tích chập nhằm tìm ra các trọng số thích hợp nhất để giải quyết vấn đề đặt ra.

Dể mạng thần kinh hoạt động hiệu quả hơn, ta có thể thêm vào nhiều lớp sau lớp đầu vào, mỗi lớp có thể được áp dụng các bộ lọc khác nhau. Nhờ đó, nó có thể trích xuất được nhiều đặc tính quan trọng từ các hình ảnh đầu vào.

15.1.2.2 Stride

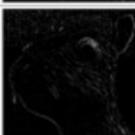
Trong thực tế, lớp tích chập có nhiều thiết lập khác nhau cung cấp các khả năng để giảm số lượng các tham số, đồng thời làm giảm các tác dụng phụ, một trong số đó là *stride*. Trong các ví dụ được đề cập ở trên, ta giả sử ma trận tích chập trượt lần lượt trên hình ảnh đầu vào, mỗi lần di chuyển 1 đơn vị. Ta có thể thay đổi độ dời của ma trận tích chập mỗi lần di chuyển bằng cách thay đổi *stride*. Ta có thể thấy trong Hình 15.5, với ma trận đầu vào có kích thước 7×7 , nếu ta di chuyển ma trận tích hợp 1 đơn vị mỗi lần cho đến hết thì ma trận kết quả sẽ có kích thước 5×5 . Nếu ta thay đổi $stride=2$, tức là di chuyển ma trận tích hợp 2 đơn vị mỗi lần thì ma trận kết quả sẽ chỉ có kích thước 3×3 .

Tổng quát hóa, với ma trận đầu vào có kích thước $N \times N$, ma trận tích hợp có kích thước $F \times F$ và giá trị *stride* là S thì ma trận kết quả O sẽ có kích thước là:

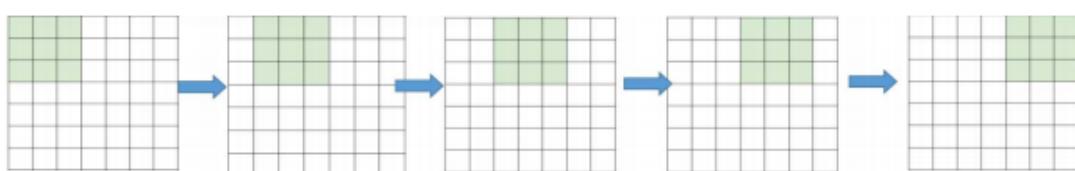
$$O = \frac{N - F}{S} + 1$$

15.1.2.3 Padding

Một trong những hạn chế của ma trận tích chập là có thể làm mất thông tin quan trọng tồn tại trên đường viền của hình ảnh. Để khắc phục hạn chế này, một giải pháp đơn giản đã được đề ra

Operation	Filter	Convolved Image
Identity	$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$	
Edge detection	$\begin{bmatrix} 1 & 0 & -1 \\ 0 & 0 & 0 \\ -1 & 0 & 1 \end{bmatrix}$	
	$\begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$	
Sharpen	$\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$	
	$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix}$	
Box blur (normalized)	$\frac{1}{9} \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$	
Gaussian blur (approximation)	$\frac{1}{16} \begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$	

Hình 15.4: Các hiệu quả khác nhau của ma trận tích chập



Hình 15.5: Sử dụng stride=1 với ma trận tích chập

là sử dụng kỹ thuật *zero padding*. Kỹ thuật này cho phép thêm vào các số 0 bên ngoài ma trận đầu vào như trong Hình 15.6, nhằm tạo ra đường viền ảo cho hình ảnh và sử dụng ma trận tích chập trích xuất các thông tin có trên đường viền cũ của hình ảnh. Khi sử dụng zero padding như Hình 15.6, với ma trận đầu vào kích thước 7×7 và $stride=1$, ta sẽ thu được ma trận kết quả có kích thước 7×7 thay vì 5×5 như ở trên. Kỹ thuật này giúp chúng ta ngăn chặn việc kích thước đầu ra giảm dần khi tới các lớp sâu hơn trong mạng thần kinh, cho phép triển khai các mạng học sâu phức tạp với nhiều lớp tích chập.

0	0	0	0	0	0	0	0	0
0								0
0								0
0								0
0								0
0								0
0								0
0	0	0	0	0	0	0	0	0

Hình 15.6: Kỹ thuật zero padding

Tổng quát hóa, với ma trận đầu vào có kích thước $N \times N$, ma trận tích hợp có kích thước $F \times F$, giá trị $stride$ là S và giá trị $padding$ là P thì ma trận kết quả O sẽ có kích thước là:

$$O = \frac{N + 2P - F}{S} + 1$$

15.1.3 Lớp phi tuyến tính

Lớp tiếp theo sau lớp tích chập mà ta sẽ giới thiệu là lớp phi tuyến tính. Lớp phi tuyến tính có thể được sử dụng để điều chỉnh hoặc cắt bỏ các giá trị trong ma trận đầu ra. Lớp này được áp dụng để bao hòa hay giới hạn các giá trị được tạo ra trong ma trận đầu ra.

Trong suốt nhiều năm, các hàm phi tuyến tính được sử dụng nhiều nhất là sigmoid và tanh. Tuy nhiên, hiện tại hàm Rectified Linear Unit (ReLU) được sử dụng phổ biến hơn bởi những lý do sau:

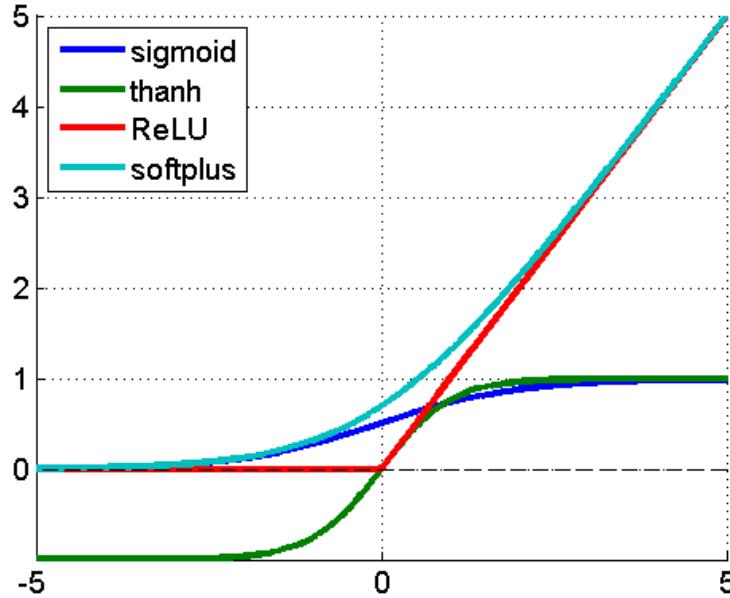
- ReLU có định nghĩa đơn giản về cả hàm số lăn đạo hàm.

$$ReLU(x) = \max(0, x)$$

$$\frac{d}{dx} ReLU(x) = \begin{cases} 1, & \text{nếu } x > 0 \\ 0, & \text{nếu } x \leq 0 \end{cases}$$

- Các hàm bao hòa như sigmoid gây ra các vấn đề trong việc truyền ngược. Khi mạng thần kinh được thiết kế sâu hơn, các tín hiệu độ dốc sẽ bị "biến mất", không có ý nghĩa trong quá trình huấn luyện mạng thần kinh. Điều này xảy ra do giá trị độ dốc của các hàm đó rất gần bằng 0 hầu như ở khắp mọi nơi trừ vùng gần giá trị trung bình của hàm. Tuy nhiên, ReLU có một độ dốc không đổi cho đầu vào dương và bằng 1.

- ReLU tạo ra các thể diện thừa hơn, bởi vì các giá trị âm đều được chuyển hoàn toàn về 0. Ngược lại, các hàm sigmoid và tanh luôn có kết quả khác 0 về độ dốc, có thể gây bất lợi cho việc huấn luyện mạng thần kinh.



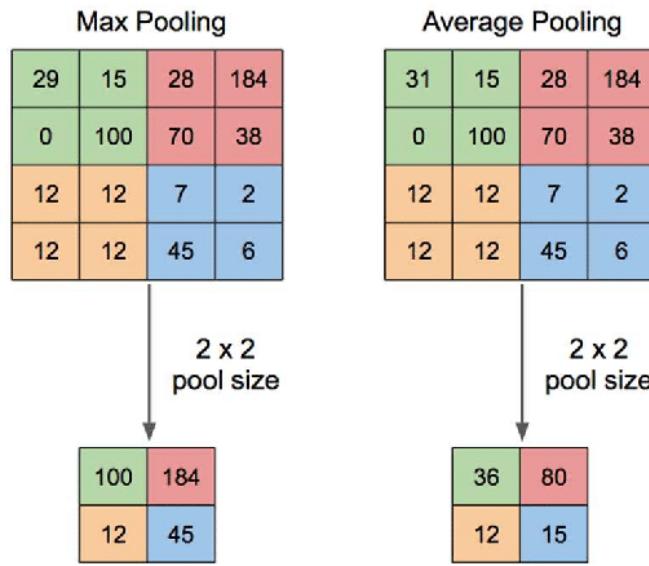
Hình 15.7: Các hàm phi tuyến tính phổ biến

15.1.4 Lớp tổng hợp

Lớp tổng hợp thường được sử dụng giữa các lớp tích chập để đơn giản hóa thông tin đầu vào nhằm làm giảm số lượng neuron ở các lớp tiếp theo trong khi vẫn giữ lại được các đặc tính quan trọng của dữ liệu đầu vào. Điều này giúp làm giảm độ phức tạp và khối lượng tính toán của mạng thần kinh đồng thời vẫn bảo toàn được hiệu suất tính toán và độ chính xác của nó. Trong lĩnh vực xử lý ảnh, việc này tương tự với giảm độ phân giải của hình ảnh.

Các phương pháp tổng hợp được dùng phổ biến hiện nay là tổng hợp trung bình (*average pooling*) và tổng hợp cực đại (*max pooling*). Phương pháp tổng hợp cực đại chia hình ảnh ban đầu thành nhiều vùng nhỏ hơn có cùng kích thước và chọn giá trị lớn nhất trong vùng đó làm giá trị đầu ra của 1 neuron trong lớp tiếp theo. Các kích thước vùng thường được dùng để tổng hợp là 2×2 và 4×4 . Ta có thể thấy trong Hình 15.8, khi sử dụng lớp tổng hợp cực đại với kích thước vùng là 2×2 , nó sẽ chọn giá trị cực đại trong mỗi vùng (các vùng xanh lá, đỏ, vàng, xanh dương) và tổng hợp thành ma trận kết quả, ma trận kết quả đã giảm kích thước đi một nửa so với ma trận đầu vào.

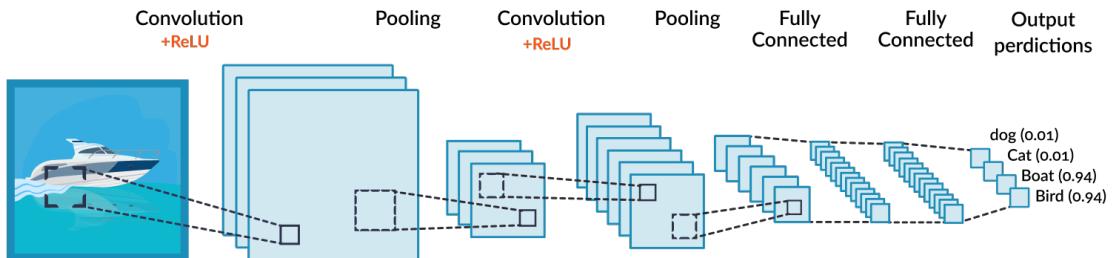
Việc sử dụng các lớp tổng hợp sẽ làm thay đổi vị trí tương đối của các điểm ảnh trong hình ảnh, do đó sẽ không bảo toàn được thông tin ban đầu. Vì vậy, các lớp tổng hợp chỉ nên được sử dụng khi cần trích xuất và bảo toàn các thông tin quan trọng của hình ảnh. Ngoài ra, ta có thể sử dụng các lớp tổng hợp với *stride* khác nhau để tăng độ hiệu quả cho việc trích xuất các đặc tính của dữ liệu đầu vào.



Hình 15.8: Các lớp tổng hợp phổ biến

15.1.5 Lớp kết nối đầy đủ

Lớp kết nối đầy đủ được xây dựng tương tự như cách sắp xếp các neuron trên một lớp trong các mạng thần kinh nhân tạo cổ điển. Trong đó, mỗi neuron ở lớp sau được kết nối tới tất cả các neuron ở lớp trước như trong Hình 15.9. Đây là những lớp rất quan trọng đối với quá trình huấn luyện mạng thần kinh tích chập, giúp mạng thần kinh “học hỏi” được các đặc tính quan trọng của hình ảnh và đưa ra các kết quả dự đoán chính xác nhất.



Hình 15.9: Lớp kết nối đầy đủ

Hạn chế chính của lớp kết nối đầy đủ là nó chứa nhiều tham số và yêu cầu nhiều thao tác tính toán phức tạp trong quá trình huấn luyện mạng thần kinh. Chúng ta có thể loại bỏ một phần các kết nối nhằm tăng tốc độ huấn luyện mà vẫn bảo toàn được hiệu suất và độ chính xác của mạng thần kinh bằng kỹ thuật loại bỏ (*dropout*) [100].

15.2 Ứng dụng mạng thần kinh tích chập vào thực tế

15.2.1 Đặt vấn đề

Hiện nay, theo báo cáo của Tổ chức động vật thế giới, các nhà khoa học đã phát hiện hơn 360 giống chó khác nhau được chia thành 7 nhóm khác nhau dựa trên mục đích và khả năng của từng loài chó hoặc về hình dáng và kích thước của nó. 7 nhóm đó là:

- *Sporting*: Các giống chó trong nhóm này được lai tạo để hỗ trợ những người thợ săn trong việc săn bắt các loài động vật nhỏ
- *Hound*: Tất cả các giống chó trong nhóm này đều được lai tạo để săn bắt các loài động vật máu nóng, bao gồm chó rừng, hươu, cá, vịt và chim.
- *Working*: Các giống chó trong nhóm này được huấn luyện để làm một số công việc đặc thù như kéo xe, canh nhà, cứu hộ, ...
- *Herding*: Nhóm này được lai tạo để hỗ trợ người nông dân trong việc chăn thả gia súc như cừu, bò, dê và thậm chí cả tuần lộc.
- *Terrier*: Những con chó trong nhóm này ban đầu được lai tạo để giúp kiểm soát quần thể động vật gặm nhấm.
- *Toy*: Nhóm này bao gồm các giống chó có kích thước nhỏ.
- *Non-Sporting*: Các giống chó trong nhóm này rất đa dạng, có thể thực hiện nhiều công việc khác nhau của các nhóm khác nên khó phân chia vào 6 nhóm còn lại.

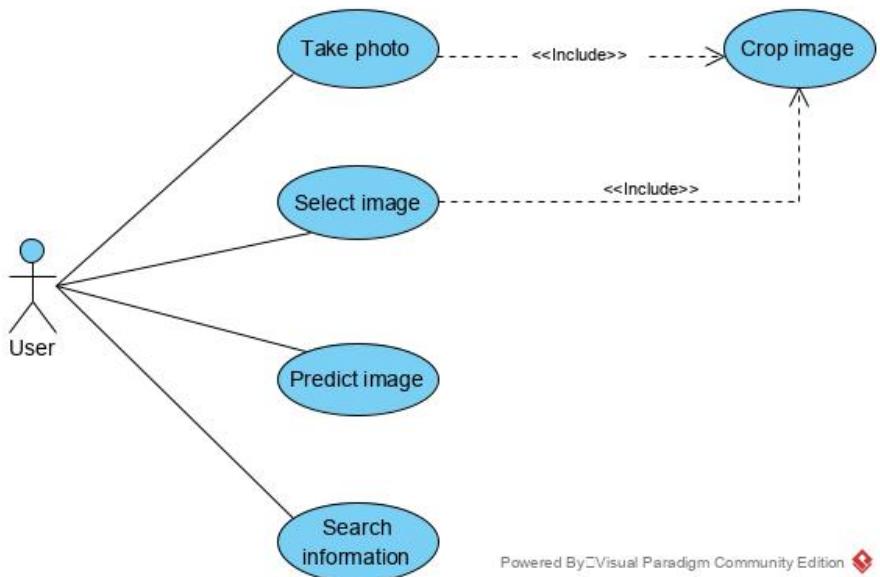
Trong số các giống chó đã được phát hiện, có nhiều loài chó có nhiều điểm tương đồng về ngoài và cả kích thước, chúng ta khó có thể phân biệt bằng mắt thường và cần quan sát tỉ mỉ mới có thể nhận biết được. Hơn nữa, với tổng số lượng hơn 360 giống chó khác nhau trên thế giới, ta khó có thể ghi nhớ hết các thông tin cần thiết về từng giống chó và có thể gây nhầm lẫn trong việc nhận biết các giống chó. Nếu có một cuốn sách ghi chép đầy đủ thông tin về các giống chó thì việc mang theo cũng khá bất tiện và việc tìm kiếm thông tin cũng làm chúng ta tốn nhiều thời gian không cần thiết.

Vì vậy, trong bài báo cáo này, ta sẽ tiến hành xây dựng một ứng dụng mobile giúp người dùng có thể nhanh chóng nhận biết các giống chó khác nhau thông qua hình ảnh được đưa vào trong ứng dụng. Ứng dụng này cũng có thể giúp các nhà khoa học phân tích nhanh chóng và hiệu quả các đặc tính nhất định của các giống chó.

Ứng dụng này cho phép người dùng chụp một hình ảnh mới từ camera hoặc chọn một hình ảnh có sẵn trong thiết bị và đưa vào mô hình mạng thần kinh tích chập đã được huấn luyện để phân tích hình ảnh và đưa ra kết quả dự đoán nhanh chóng, chính xác. Ngoài ra, từ kết quả dự đoán của ứng dụng, người dùng cũng có thể tìm kiếm thêm các thông tin liên quan về kết quả dự đoán để cập nhật thêm nhiều kiến thức mới.

15.2.2 Phân tích yêu cầu

15.2.2.1 Use case diagram của hệ thống



Powered By Visual Paradigm Community Edition

Hình 15.10: Use case diagram của hệ thống

15.2.2.2 Đặc tả use case

Use Case ID:			
Use Case Name:	Take photo		
Process Owner:	Lê Đỗ Thanh Bình	Last Updated By:	Lê Đỗ Thanh Bình
Date Created:	12/08/2020	Date Last Updated:	12/08/2020
Actor:	User		
Description:	Cho phép người dùng chụp hình từ camera		
Triggers:	Người dùng chọn nút “Select Image” và mở camera		
Pre-conditions:	Không		
Post-conditions:	Hiển thị hình ảnh đã chụp		
Normal Flow:	1. Người dùng chọn nút “Select Image” và mở camera 2. Chụp hình từ camera 3. Cắt một phần hình đã chụp nếu cần thiết 4. Chọn “OK” 5. Hiển thị hình ảnh đã chụp		
Alternative Flow:	<i>Alternative 1: ở bước 2</i> 2a. Người dùng không hài lòng với hình ảnh đã chụp và chụp lại Continue step 3 in normal flow.		
Exceptions:	<i>Exception 1: ở bước 1</i> 1a. Thiết bị không có camera <i>Exception 2: ở bước 2</i> 2b. Người dùng dừng việc chụp hình		
Notes and Issues:	Không		
Non-functional:	Không		

15.2. ỨNG DỤNG MẠNG THẦN KINH TÍCH CHẬP VÀO THỰC TẾ

Use Case Name:	Select image		
Process Owner:	Lê Đỗ Thanh Bình	Last Updated By:	Lê Đỗ Thanh Bình
Date Created:	12/08/2020	Date Last Updated:	12/08/2020
Actor:	User		
Description:	Cho phép người chọn hình từ kho lưu trữ hình ảnh		
Triggers:	Người dùng chọn nút “Select Image” và mở kho lưu trữ hình ảnh		
Pre-conditions:	Thiết bị phải có kho lưu trữ hình ảnh		
Post-conditions:	Hiển thị hình ảnh đã chọn		
Normal Flow:	<ol style="list-style-type: none">1. Người dùng chọn nút “Select Image” và mở kho lưu trữ hình ảnh2. Chọn hình từ kho lưu trữ hình ảnh3. Cắt một phần hình đã chọn nếu cần thiết4. Chọn “OK”5. Hiển thị hình ảnh đã chọn		
Alternative Flow:	<i>Alternative 1: ở bước 2</i> 2a. Người dùng không hài lòng với hình ảnh đã chụp và chụp lại Continue step 3 in normal flow.		
Exceptions:	<i>Exception 1: ở bước 2</i> 2b. Người dùng dừng việc chọn hình		
Notes and Issues:	Không		
Non-functional:	Không		

Use Case ID:			
Use Case Name:	Predict Image		
Process Owner:	Lê Đỗ Thanh Bình	Last Updated By:	Lê Đỗ Thanh Bình
Date Created:	12/08/2020	Date Last Updated:	12/08/2020
Actor:	User		
Description:	Đự đoán loài chó từ hình ảnh đã chọn (chụp) của người dùng và hiển thị kết quả dự đoán		
Triggers:	Không		
Pre-conditions:	Không		
Post-conditions:	Hiển thị kết quả dự đoán		
Normal Flow:	1. Xử lý dữ liệu hình ảnh đầu vào 2. Đưa dữ liệu vào mô hình mạng thần kinh tích chập để dự đoán kết quả 3. Xử lý kết quả dự đoán 4. Hiển thị kết quả dự đoán		
Alternative Flow:	Không		
Exceptions:	Không		
Notes and Issues:	Không		
Non-functional:	Kết quả dự đoán phải được đưa ra nhanh (nhỏ hơn 1s) và chính xác (độ chính xác > 80%)		

15.3. THIẾT KẾ HỆ THỐNG

Use Case ID:			
Use Case Name:	Search information		
Process Owner:	Lê Đỗ Thanh Bình	Last Updated By:	Lê Đỗ Thanh Bình
Date Created:	12/08/2020	Date Last Updated:	12/08/2020
Actor:	User		
Description:	Tìm kiếm thêm thông tin về loài chó được dự đoán từ hình ảnh đầu vào		
Triggers:	Người dùng chọn “Search”		
Pre-conditions:	1. Mạng thàn kính tích chập đã đưa ra kết quả dự đoán 2. Thiết bị được kết nối internet		
Post-conditions:	Hiển thị kết quả tìm kiếm		
Normal Flow:	1. Người dùng chọn “Search” 2. Tìm kiếm thêm thông tin về loài chó được dự đoán từ hình ảnh đầu vào 3. Hiển thị kết quả tìm kiếm		
Alternative Flow:	Không		
Exceptions:	Không		
Notes and Issues:	Không		
Non-functional:	Không		

15.2.2.3 Yêu cầu phi chức năng của hệ thống

- Tính khả thi:
 - a
 - Người dùng có thể sử dụng ứng dụng dễ dàng sau 5 phút hướng dẫn.
- Hiệu suất:
 - Tốc độ phản hồi của ứng dụng nhanh (nhỏ hơn 1s)
 - Độ chính xác của kết quả dự đoán phải ổn định (lớn hơn 80%)
- Bảo mật:
 - Ứng dụng phải bảo mật thông tin người dùng
- Môi trường hoạt động:
 - Ứng dụng chạy được trên tất cả các điện thoại có phiên bản hệ điều hành Android từ 5.0 trở lên
- Kích thước:
 - Ứng dụng phải có kích thước vừa phải (nhỏ hơn 100MB)

15.3 Thiết kế hệ thống

15.3.1 Xây dựng mạng thần kinh tích chập

Trong phần này, chúng ta sẽ sử dụng kỹ thuật chuyển dịch học tập (transfer learning) bằng cách sử dụng một mô hình mạng thần kinh tích chập đã được huấn luyện từ trước với độ chính xác cao và điều chỉnh lại mạng thần kinh này để giải quyết bài toán đặt ra.

Một mô hình được huấn luyện từ trước là một mạng thần kinh đã được huấn luyện trên một tập dữ liệu lớn, trong một khoảng thời gian đủ dài và được kiểm chứng độ hiệu quả và độ chính xác của nó. Tư tưởng chính của kỹ thuật này là nếu một mô hình mạng thần kinh tích chập đã được huấn luyện trên một tập dữ liệu đủ lớn và tổng quát thì mô hình này cũng sẽ giải quyết hiệu quả các vấn đề tương tự khác trên thế giới. Do đó, ta có thể sử dụng lại các đặc tính được trích xuất ra từ mô hình này mà không cần phải huấn luyện lại một mô hình mới từ đầu để giải quyết bài toán đặt ra.

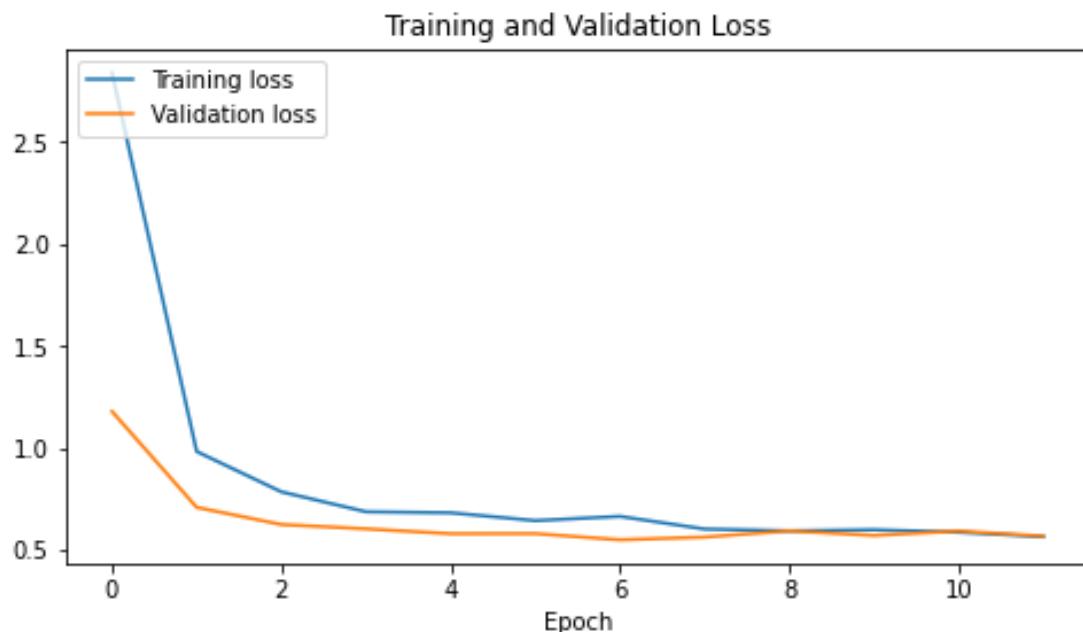
Kỹ thuật này cho phép ta trích xuất được những đặc tính có nghĩa từ tập dữ liệu huấn luyện. Sau đó ta chỉ cần thêm một số lớp để phân loại dữ liệu ở trên đầu của mạng thần kinh đã được huấn luyện và phân loại thành các nhóm tương ứng.

Tập dữ liệu đầu vào được sử dụng trong báo cáo này được tổng hợp tại <http://vision.stanford.edu/aditya86/ImageNetDogs/main.html>, tập dữ liệu này bao gồm khoảng 20000 hình ảnh được chia thành 196 giống chó khác nhau. Ta sẽ huấn luyện mạng thần kinh trên tập dữ liệu này để tìm ra phương pháp nhận biết các giống chó khác nhau. Ta sẽ sử dụng mạng thần kinh tích chập NasNetMobile[101] được phát triển đã đưa ra nhiều kết quả chính xác và nhanh chóng. Ngoài ra, ta cũng sử dụng hàm tối ưu Adam[102] để tối thiểu hóa hàm chi phí của mô hình trong quá trình huấn luyện.

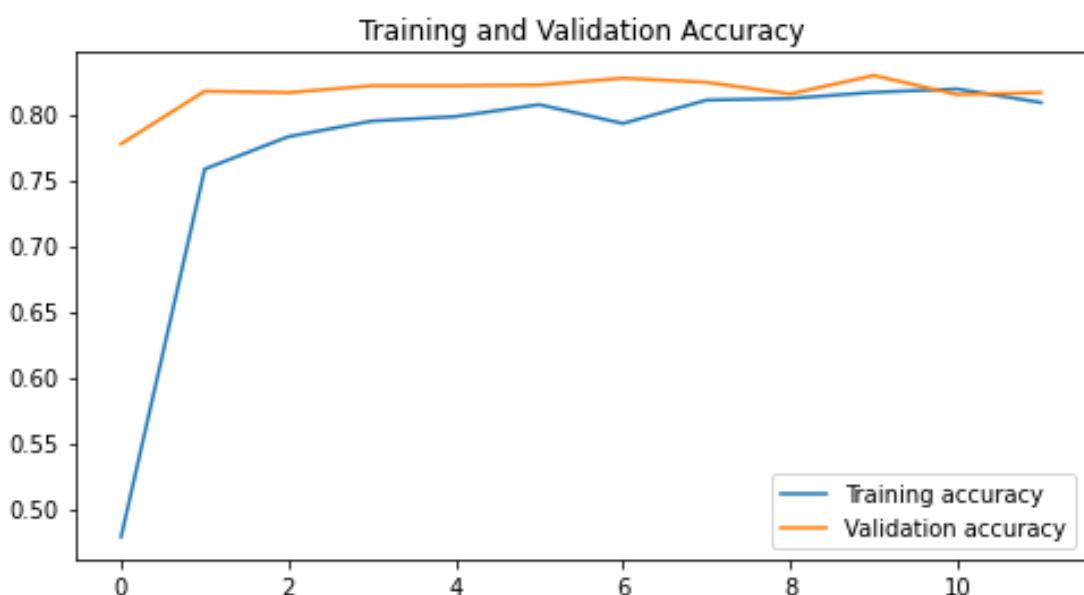
Để tăng tính đa dạng của dữ liệu huấn luyện cho mạng thần kinh tích chập, ta sẽ sử dụng một kỹ thuật khác là tăng cường dữ liệu (data augmentation)[103] bằng cách áp dụng các phép biến đổi ngẫu nhiên (nhưng thực tế) như xoay, cắt hình ảnh, tăng giảm độ sáng và độ tương phản của hình ảnh,... Sử dụng kỹ thuật này giúp ta tạo nên nhiều dữ liệu để huấn luyện, tránh tình trạng mạng thần kinh gặp lại cùng một dữ liệu nhiều lần dẫn đến tình trạng "ghi nhớ" dữ liệu khiến cho mạng thần kinh dễ dàng xử lý các dữ liệu huấn luyện mà không xử lý được các dữ liệu thực tế chưa từng gặp phải và khiến cho độ chính xác của mạng thần kinh giảm xuống. Dưới đây là các kết quả sau khi huấn luyện mô hình:

15.3.2 Activity diagram của hệ thống

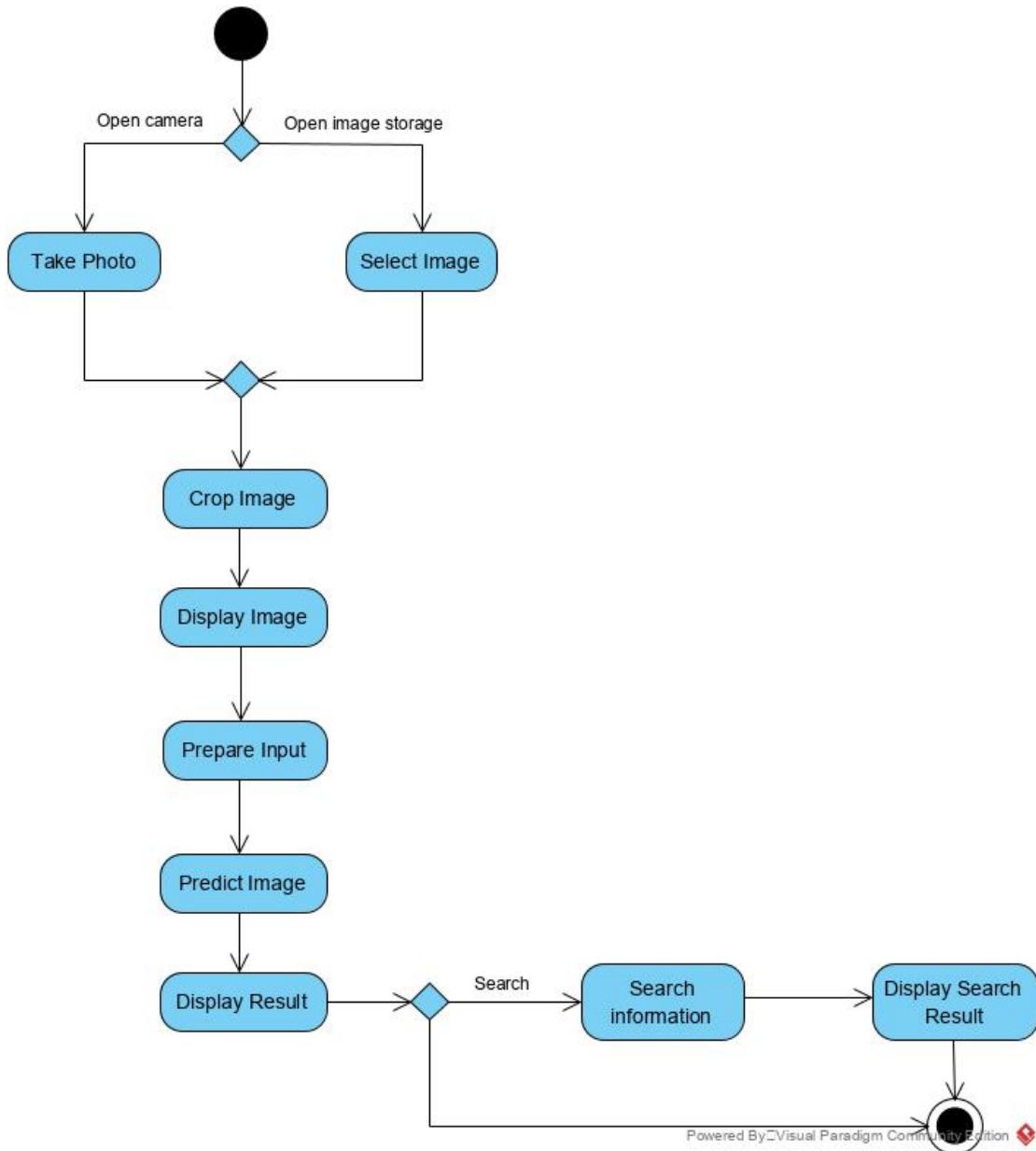
15.3. THIẾT KẾ HỆ THỐNG



Hình 15.11: Sự thay đổi của hàm chi phí trong quá trình huấn luyện

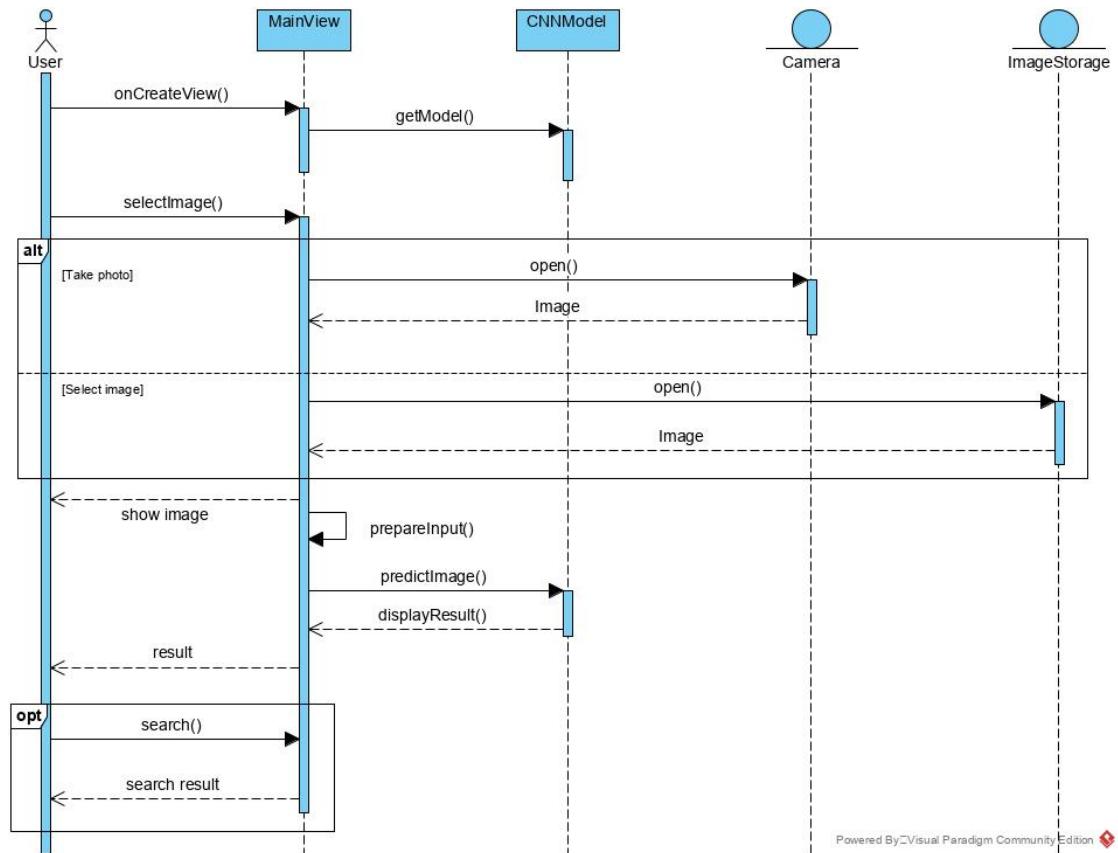


Hình 15.12: Sự thay đổi của độ chính xác trong quá trình huấn luyện



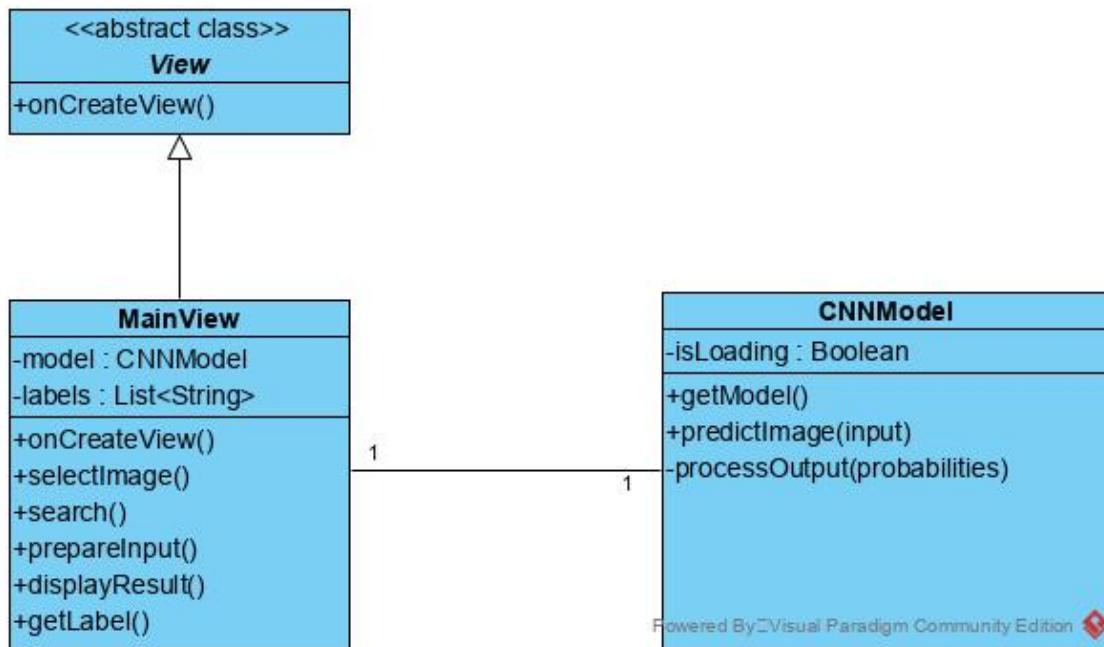
Hình 15.13: Activity diagram của hệ thống

15.3.3 Sequence diagram của hệ thống



Hình 15.14: Sequence diagram của hệ thống

15.3.4 Class diagram của hệ thống



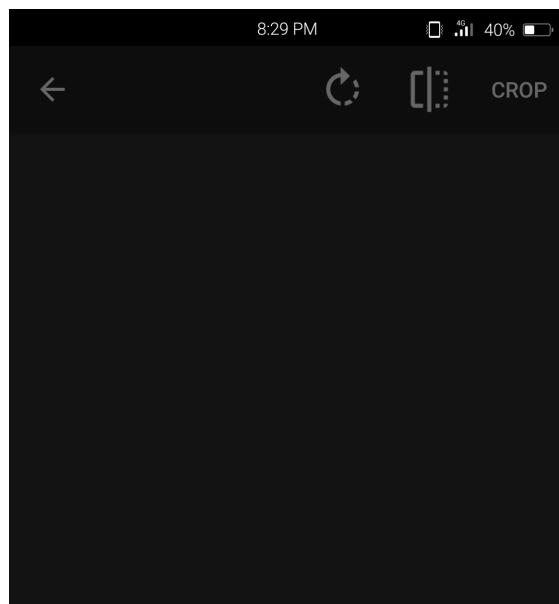
Hình 15.15: Class diagram của hệ thống

15.4 Demo ứng dụng

Mã nguồn ứng dụng được để tại liên kết:

<https://gitlab.com/binhnd234/convolution-neural-network-project>. Về cơ bản, ứng dụng đã đáp ứng được các yêu cầu đặt ra và cho một số kết quả khả quan như đã đề cập ở trên. Tuy nhiên, với một số hình ảnh đặc thù ứng dụng vẫn chưa có được kết quả chính xác.

15.4. DEMO ỨNG DỤNG



Select source



Camera



Downloads



File explorer



Drive



Photos



Gallery



File Manager

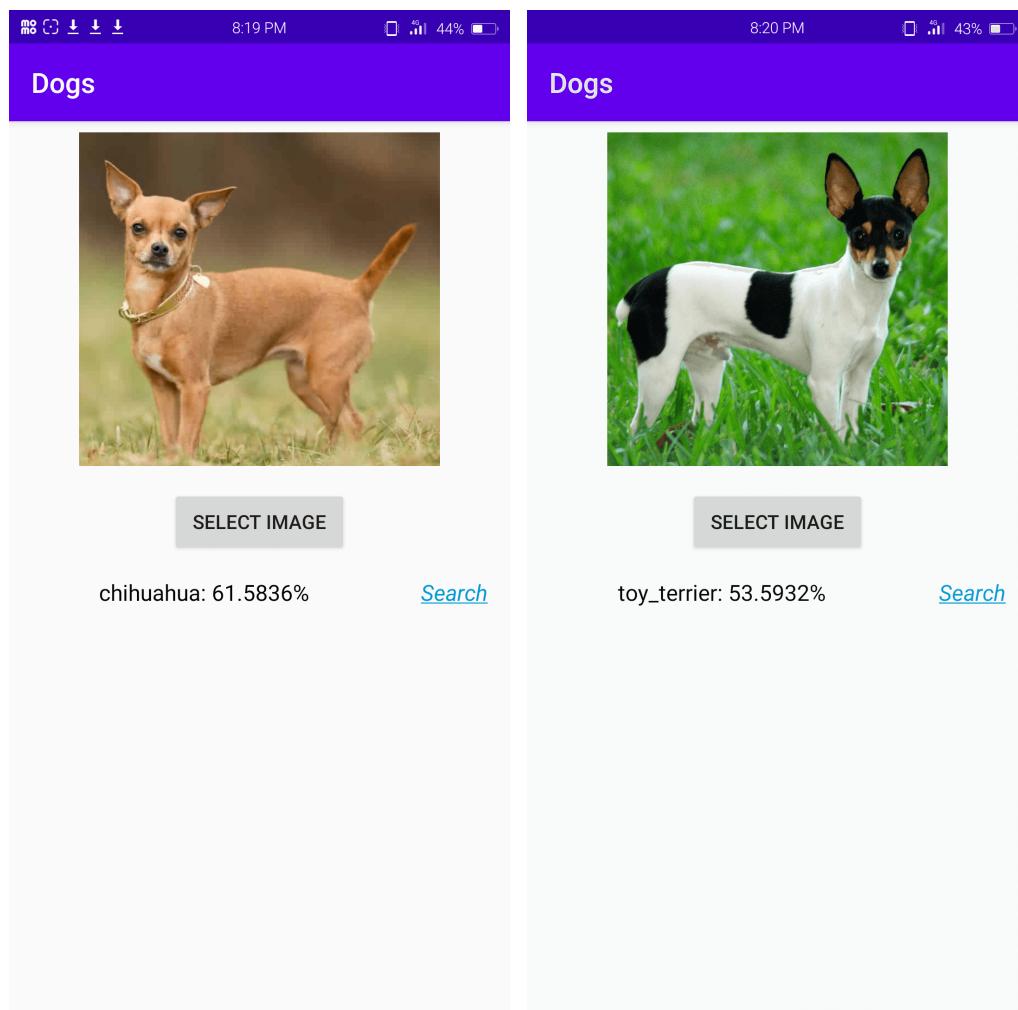


File Manager

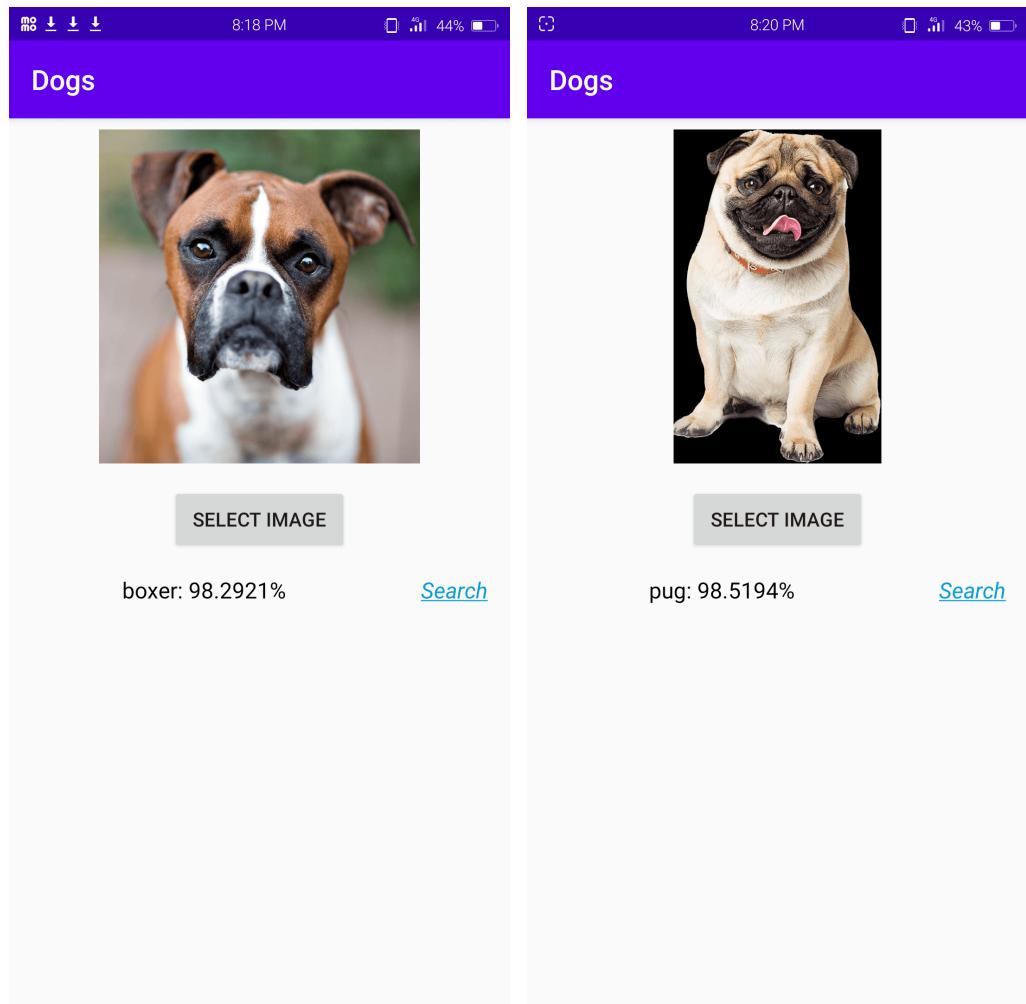
Hình 15.16: Chức năng chụp (chọn) hình ảnh

Bảng 15.1: Bảng mô tả các chức năng

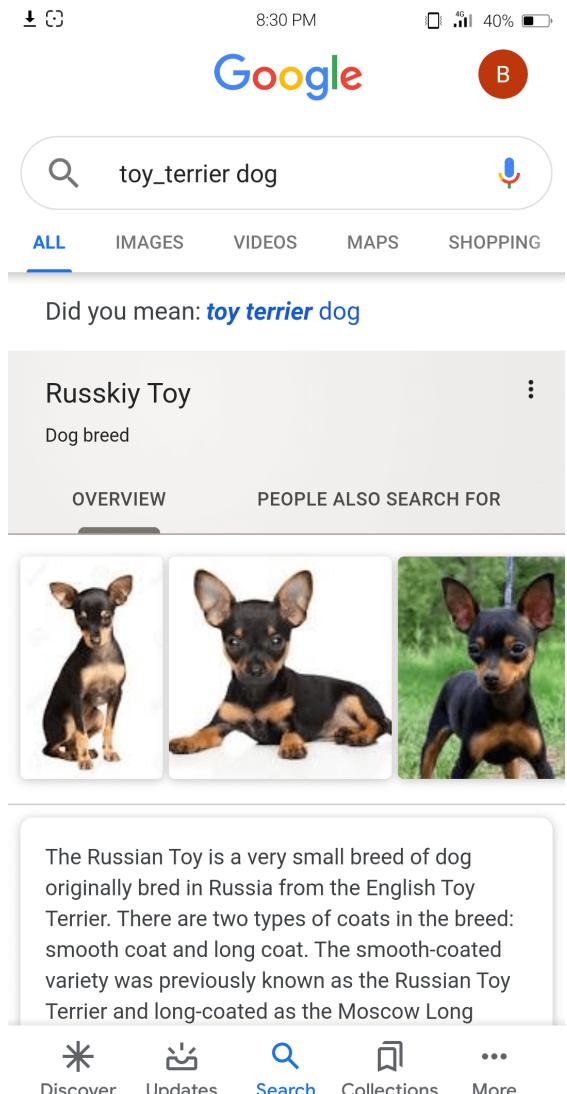
Usecase	Mô tả
View	
onCreateView()	Khởi tạo và hiện thị giao diện lên màn hình thiết bị
CNNModel	
getModel()	Lấy mô hình mạng thần kinh tích chập đã huấn luyện trước đó để sử dụng
predictImage(input)	Sử dụng mạng thần kinh tích chập để xử lý dữ liệu và đưa ra kết quả dự đoán
processOutput(probability)	Xử lý kết quả dự đoán của mạng thần kinh tích chập
MainView	
selectImage()	Cho phép người dùng chụp hoặc chọn một hình ảnh mới
search()	Tìm kiếm thông tin dựa trên kết quả dự đoán của mạng thần kinh tích chập
prepareInput()	Xử lý dữ liệu từ hình ảnh đầu vào để đưa vào mạng thần kinh tích chập
displayResult()	Hiển thị kết quả dự đoán của mạng thần kinh tích chập
getLabel()	Lấy các nhãn dự đoán của tập dữ liệu



15.4. DEMO ỨNG DỤNG



Hình 15.17: Chức năng dự đoán từ hình ảnh đầu vào



Hình 15.18: Chức năng tìm kiếm thông tin

Từ điển chú giải

chỉ số F1 (*F1-score*) là trung bình điều hoà của độ chính xác dự đoán và độ thu hồi, giúp cân bằng giữa hai chỉ số này.. 82

cây quyết định (*decision tree*) là một mô hình học có giám sát dùng để giải bài toán phân loại và hồi quy, thường biểu diễn dưới dạng cây nhị phân. 53

dương tính giả (*false positive*) số mẫu thuộc lớp âm nhưng mô hình dự đoán nhầm là dương (hay còn gọi là lỗi loại I). 80, 284

dương tính thật (*true positive*) số mẫu thuộc lớp dương được mô hình dự đoán đúng là dương. 80, 284

dưới khớp (*underfitting*) là hiện tượng xảy ra khi mô hình quá đơn giản để nắm bắt được quy luật trong dữ liệu, dẫn đến hiệu suất thấp cả trên tập huấn luyện và tập kiểm tra. 42

entropy một thước đo độ bất định hoặc hỗn loạn trong một tập dữ liệu. 52, 54, 55, 284

hàm giá trị (*cost function*) là một hàm dùng để đo tổng sai số của mô hình trên toàn bộ tập dữ liệu huấn luyện. 24

hàm hợp lý (*likelihood function*) là một hàm dùng để đánh giá xác suất quan sát được dữ liệu thực tế, giả định rằng tham số của mô hình là cố định. 24

hàm khoảng cách (*metric*) là một hàm số đo lường mức độ khác biệt hoặc độ xa giữa hai điểm dữ liệu trong một không gian. Một hàm được gọi là hàm khoảng cách nếu thỏa mãn bốn tính chất: không âm, đồng nhất, đối xứng và bất đẳng thức tam giác. 32

hàm mất mát (*loss function*) là một hàm đánh giá mức độ sai lệch giữa giá trị dự đoán của mô hình và giá trị thực tế. 23

học có giám sát (*supervised learning*) là một phương pháp học máy trong đó dữ liệu huấn luyện bao gồm cả đầu vào và nhãn đầu ra tương ứng. 21

học không có giám sát (*unsupervised learning*) là một phương pháp học máy trong đó dữ liệu huấn luyện không có nhãn đầu ra, thường dùng để khám phá cấu trúc tiềm ẩn trong dữ liệu, như phân cụm hoặc giảm chiều. 27, 284

khoảng cách Minkowski (*Minkowski Distance*) là một dạng tổng quát của các khoảng cách Euclid và Manhattan, dùng để đo độ khác biệt giữa hai điểm trong không gian.. 36

làm mượt Laplace (*Laplace smoothing*) là một kỹ thuật làm mượt xác suất trong mô hình Naïve Bayes để tránh vấn đề xác suất bằng 0. 78, 83

ma trận nhầm lẫn (*confusion matrix*) là một bảng dùng để đánh giá hiệu năng của mô hình phân loại, thể hiện số lượng dự đoán đúng và sai của mô hình so với nhãn thực tế. Các ô trong ma trận biểu diễn quan hệ giữa dương tính thật, âm tính thật, dương tính giả, và âm tính giả. 80

phương pháp khuỷu tay (*Elbow Method*) là một kỹ thuật được sử dụng để xác định số cụm k tối ưu trong thuật toán thuật toán k -means dựa trên việc phân tích đồ thị WCSS.. 117

quá khớp (*overfitting*) là hiện tượng xảy ra khi mô hình học quá sát với tập dữ liệu huấn luyện, bao gồm cả nhiễu và ngoại lệ, dẫn đến không tổng quát hóa tốt trên dữ liệu chưa thấy. 25, 54

rừng ngẫu nhiên (*Random Forest*) là một mô hình học có giám sát sử dụng tập hợp nhiều cây quyết định. 64

thuật toán k -means là thuật toán học không có giám sát phân chia dữ liệu thành k cụm dựa trên khoảng cách giữa các điểm và tâm cụm.. 119, 284

thuật toán leo đồi (*Hill Climbing*) là một thuật toán tìm kiếm cục bộ lặp lại việc cải thiện nghiệm hiện tại bằng cách chuyển sang nghiệm láng giềng tốt hơn cho đến khi đạt cực trị địa phương. 125

thuật toán Naïve Bayes (*Naïve Bayes algorithm*) là thuật toán phân loại dựa trên định lý Bayes cùng giả định các đặc trưng đầu vào độc lập có điều kiện theo nhãn lớp. 72

tập dữ liệu huấn luyện (*training dataset*) là tập dữ liệu được sử dụng để huấn luyện mô hình học máy. 21

âm tính giả (*false negative*) số mẫu thuộc lớp dương nhưng mô hình dự đoán nhầm là âm (hay còn gọi là lỗi loại II). 80, 284

âm tính thật (*true negative*) số mẫu thuộc lớp âm được mô hình dự đoán đúng là âm. 80, 284

điểm số Silhouette (*Silhouette Score*) là thước đo đánh giá chất lượng phân cụm bằng cách xem xét đồng thời độ chặt bên trong cụm và độ tách biệt giữa các cụm.. 118

độ bất thuần chủng (*impurity*) là một chỉ số đo lường mức độ hỗn hợp của các lớp trong một nhóm. Mức độ bất thuần chủng cao xảy ra khi các phần tử trong nhóm phân bố đều giữa nhiều lớp khác nhau. 51

độ chính xác dự đoán (*precision*) là tỉ lệ số mẫu dương tính thật được mô hình dự đoán đúng trên tổng số mẫu được mô hình dự đoán là dương.. 82, 283

độ chính xác tổng thể (*accuracy*) là tỉ lệ số dự đoán đúng trên tổng số mẫu.. 81

độ lợi thông tin (*information gain*) là độ giảm của bất thuần chủng (thường đo bằng entropy) khi phân chia tập dữ liệu theo một thuộc tính. 52, 55

độ thu hồi (*recall*, còn gọi là *sensitivity* hoặc *true positive rate*) là tỉ lệ số mẫu dương tính thật được mô hình dự đoán đúng trên tổng số mẫu thực sự thuộc lớp dương.. 81, 283

độ thuần chủng (*purity*) là một chỉ số đo lường mức độ mà các phần tử trong một nhóm thuộc cùng một lớp. 52

Acronyms

ANN Mạng nơ-ron nhân tạo (Artificial Neural Networks). 25

DBSCAN Density-Based Spatial Clustering of Applications with Noise – một thuật toán phân cụm dựa trên mật độ. 205, 206

ID3 Iterative Dichotomiser 3 – một thuật toán xây dựng cây quyết định dựa trên entropy và độ lợi thông tin. 53, 54

MAP Maximum A Posteriori – phương pháp Tối đa Hậu nghiệm. 73, 74

ReLU Rectified Linear Unit. 264

SVM Support Vector Machine. 25

TF-IDF Tần suất-Nghịch đảo tài liệu (Term Frequency-Inverse Document Frequency). 26

WCSS *tổng bình phương khoảng cách trong cụm – Within-Cluster Sum of Squares.* 117–119, 284

Bibliography

- [1] M. Mohri, A. Rostamizadeh, and A. Talwalkar, *Foundations of Machine Learning*, 2nd. The MIT Press, 2018, ISBN: 0262039400.
- [2] T. Bayes, “Lii. an essay towards solving a problem in the doctrine of chances. by the late rev. mr. bayes, frs communicated by mr. price, in a letter to john canton, amfr s,” *Philosophical transactions of the Royal Society of London*, no. 53, pp. 370–418, 1763.
- [3] G. Salton and C. Buckley, “Term-weighting approaches in automatic text retrieval,” *Information Processing & Management*, vol. 24, no. 5, pp. 513–523, 1988.
- [4] T. Mikolov, K. Chen, G. Corrado, and J. Dean, “Efficient estimation of word representations in vector space,” *arXiv preprint arXiv:1301.3781*, 2013.
- [5] D. R. Cox, “The regression analysis of binary sequences (with discussion),” *Journal of the Royal Statistical Society: Series B (Methodological)*, vol. 20, no. 2, pp. 215–242, 1958.
- [6] C. Cortes and V. Vapnik, “Support-vector networks,” *Machine learning*, vol. 20, no. 3, pp. 273–297, 1995.
- [7] S. Hochreiter and J. Schmidhuber, “Long short-term memory,” *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.
- [8] A. Vaswani et al., “Attention is all you need,” in *Advances in neural information processing systems*, vol. 30, 2017.
- [9] Y. LeCun, Y. Bengio, and G. Hinton, “Deep learning,” *Nature*, vol. 521, no. 7553, pp. 436–444, 2015. doi: [10.1038/nature14539](https://doi.org/10.1038/nature14539).
- [10] K. P. Murphy, *Machine Learning: A Probabilistic Perspective*. Cambridge, MA, USA: MIT Press, 2012.
- [11] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*, 2nd. Cambridge, MA: MIT Press, 2018, ISBN: 9780262039246.
- [12] X. Zhu, “Semi-supervised learning literature survey,” University of Wisconsin-Madison, Computer Sciences Technical Report 1530, 2007. [Online]. Available: http://pages.cs.wisc.edu/~jerryzhu/pub/ssl_survey.pdf.
- [13] J. MacQueen, “Some methods for classification and analysis of multivariate observations,” in *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability*, University of California Press, vol. 1, 1967, pp. 281–297.
- [14] F. Nielsen, “Hierarchical clustering,” in *Introduction to HPC with MPI for Data Science*, Springer, 2016, ch. 8, pp. 195–211, ISBN: 978-3-319-21903-5.
- [15] D. Reynolds, “Gaussian mixture models,” in *Encyclopedia of biometrics*, Springer, 2015, pp. 827–832.

BIBLIOGRAPHY

- [16] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu, “A density-based algorithm for discovering clusters in large spatial databases with noise,” in *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining (KDD)*, AAAI Press, 1996, pp. 226–231.
- [17] T. Cover and P. Hart, “Nearest neighbor pattern classification,” *IEEE Transactions on Information Theory*, vol. 13, no. 1, pp. 21–27, 1967. DOI: [10.1109/TIT.1967.1053964](https://doi.org/10.1109/TIT.1967.1053964).
- [18] M. M. Deza and E. Deza, *Encyclopedia of Distances*. Berlin, Heidelberg: Springer, 2009, ISBN: 978-3-642-00233-5. DOI: [10.1007/978-3-642-00234-2](https://doi.org/10.1007/978-3-642-00234-2).
- [19] R. W. Hamming, “Error detecting and error correcting codes,” *Bell system technical journal*, vol. 29, no. 2, pp. 147–160, 1950.
- [20] G. Salton and M. J. McGill, “Introduction to modern information retrieval,” in *McGraw-Hill Book Company*, Defines and uses cosine similarity in information retrieval, 1983.
- [21] R. O. Duda, P. E. Hart, and D. G. Stork, *Pattern Classification*, 2nd. Wiley, 2001.
- [22] G. N. Lance and W. T. Williams, “A general theory of classificatory sorting strategies: 1. hierarchical systems,” *The Computer Journal*, vol. 9, no. 4, pp. 373–380, 1967.
- [23] J. R. Bray and J. T. Curtis, “An ordination of the upland forest communities of southern wisconsin,” *Ecological Monographs*, vol. 27, no. 4, pp. 325–349, 1957.
- [24] R. W. Hamming, “Error detecting and error correcting codes,” *Bell System Technical Journal*, vol. 29, no. 2, pp. 147–160, 1950.
- [25] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*, 2nd. New York: Springer, 2009, ISBN: 9780387848570.
- [26] K. Q. Weinberger and L. K. Saul, “Distance metric learning for large margin nearest neighbor classification,” *Journal of Machine Learning Research*, vol. 10, pp. 207–244, 2009.
- [27] H. Minkowski, *Geometrie der Zahlen*, German. Leipzig: B.G. Teubner, 1896.
- [28] D. Jurafsky and J. H. Martin, *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition*, 2nd. Upper Saddle River, NJ, USA: Prentice Hall, 2009, ISBN: 978-0-13-187321-6.
- [29] K. V. Mardia and P. E. Jupp, *Directional Statistics*. Chichester: John Wiley & Sons, 2000, ISBN: 978-0-471-95333-3. DOI: [10.1002/9780470316979](https://doi.org/10.1002/9780470316979).
- [30] J. Han, M. Kamber, and J. Pei, *Data Mining: Concepts and Techniques*, 3rd. San Francisco, CA, USA: Morgan Kaufmann, 2011.
- [31] S. A. Dudani, “The distance-weighted k-nearest-neighbor rule,” *IEEE Transactions on Systems, Man, and Cybernetics*, vol. SMC-6, no. 4, pp. 325–327, 1976.
- [32] R. Kohavi, “A study of cross-validation and bootstrap for accuracy estimation and model selection,” in *Proceedings of the 14th International Joint Conference on Artificial Intelligence (IJCAI)*, San Francisco, CA: Morgan Kaufmann, 1995, pp. 1137–1143. [Online]. Available: <https://ai.stanford.edu/~ronnyk/accEst.pdf>.
- [33] D. Wettschereck, D. W. Aha, and T. Mohri, “A review and empirical evaluation of feature weighting methods for a class of lazy learning algorithms,” *Artificial Intelligence Review*, vol. 11, no. 1-5, pp. 273–314, 1997.
- [34] D. W. Aha, D. Kibler, and M. K. Albert, “Instance-based learning algorithms,” *Machine Learning*, vol. 6, no. 1, pp. 37–66, 1991.

BIBLIOGRAPHY

- [35] J. L. Bentley, “Multidimensional binary search trees used for associative searching,” *Communications of the ACM*, vol. 18, no. 9, pp. 509–517, 1975.
- [36] S. M. Omohundro, “Five balltree construction algorithms,” International Computer Science Institute, Tech. Rep. TR-89-063, 1989.
- [37] I. Guyon and A. Elisseeff, “An introduction to variable and feature selection,” *Journal of Machine Learning Research*, vol. 3, pp. 1157–1182, 2003.
- [38] K. Beyer, J. Goldstein, R. Ramakrishnan, and U. Shaft, “When is “nearest neighbor” meaningful?” In *Proceedings of the 7th International Conference on Database Theory (ICDT)*, ser. Lecture Notes in Computer Science, vol. 1540, Springer, 1999, pp. 217–235.
- [39] Y. Freund and L. Mason, “The alternating decision tree learning algorithm,” in *Proceedings of the Sixteenth International Conference on Machine Learning*, ser. ICML ’99, San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1999, pp. 124–133, ISBN: 1558606122.
- [40] T. M. Mitchell, *Machine Learning*. McGraw-Hill Education, 1997.
- [41] Y. Ben-Haim and E. Tom-Tov, “A streaming parallel decision tree algorithm..,” *Journal of Machine Learning Research*, vol. 11, no. 2, 2010.
- [42] J. R. Quinlan, “Induction of decision trees,” *Machine Learning*, vol. 1, no. 1, pp. 81–106, 1986.
- [43] L. Breiman, J. Friedman, R. A. Olshen, and C. J. Stone, *Classification and Regression Trees*. Wadsworth, 1984.
- [44] S. Theodoridis and K. Koutroumbas, *Pattern Recognition*. Academic Press, 2008.
- [45] J. R. Quinlan, *C4.5: Programs for Machine Learning*. Morgan Kaufmann, 1993.
- [46] C. E. Shannon, “A mathematical theory of communication,” *Bell System Technical Journal*, vol. 27, no. 3, pp. 379–423, 1948.
- [47] H. Almuallim, “An efficient algorithm for optimal pruning of decision trees,” *Artificial Intelligence*, vol. 83, no. 2, pp. 347–362, 1996.
- [48] F. Esposito, D. Malerba, and G. Semeraro, “An experimental comparison of pruning methods for decision tree induction,” *Machine Learning*, vol. 27, no. 2, pp. 169–201, 1997.
- [49] J. Surowiecki, *The Wisdom of Crowds*. Anchor Books, 2005.
- [50] L. Breiman, “Random forests,” *Machine Learning*, vol. 45, no. 1, pp. 5–32, 2001.
- [51] T. K. Ho, “Random decision forests,” in *Proceedings of the Third International Conference on Document Analysis and Recognition*, 1995, pp. 278–282.
- [52] N. Bhagat and S. Patil, “Enhanced random forest algorithm,” *International Journal of Advanced Research in Computer Science*, vol. 6, no. 5, 2015.
- [53] T. F. Cootes et al., “Robust and accurate shape model fitting using random forest regression voting,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 9, pp. 1740–1754, 2012.
- [54] M. Sahami, S. Dumais, D. Heckerman, and E. Horvitz, “A bayesian approach to filtering junk e-mail,” *AAAI Workshop on Learning for Text Categorization*, pp. 55–62, 1998.
- [55] I. Androutsopoulos, J. Koutsias, K. V. Chandrinou, and C. D. Spyropoulos, “An experimental comparison of naive bayesian and keyword-based anti-spam filtering with personal e-mail messages,” *Proceedings of the 23rd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 160–167, 2000.

BIBLIOGRAPHY

- [56] A. McCallum and K. Nigam, “A comparison of event models for naive bayes text classification,” *AAAI Workshop on Learning for Text Categorization*, pp. 41–48, 1998.
- [57] D. D. Lewis, “Naive (bayes) at forty: The independence assumption in information retrieval,” *European Conference on Machine Learning (ECML)*, pp. 4–15, 1998.
- [58] S. J. Russell and P. Norvig, “Artificial intelligence: A modern approach,” 2016.
- [59] H. Zhang, “The optimality of naive bayes,” *Proceedings of the Seventeenth International Florida Artificial Intelligence Research Society Conference (FLAIRS)*, 2004.
- [60] P. Domingos and M. Pazzani, “Beyond independence: Conditions for the optimality of the simple bayesian classifier,” *Machine Learning*, vol. 29, pp. 103–130, 1996.
- [61] H. He and E. A. Garcia, “Learning from imbalanced data,” *IEEE Transactions on Knowledge and Data Engineering*, vol. 21, no. 9, pp. 1263–1284, 2009.
- [62] I. Kononenko, “Machine learning for medical diagnosis: History, state of the art and perspective,” *Artificial Intelligence in Medicine*, vol. 23, no. 1, pp. 89–109, 2001.
- [63] S. Garera, N. Provos, M. Chew, and A. D. Rubin, “A framework for detection and measurement of phishing attacks,” in *Proceedings of the 2007 ACM Workshop on Recurring Malcode*, 2007, pp. 1–8.
- [64] J. Ma, L. K. Saul, S. Savage, and G. M. Voelker, “Identifying suspicious urls: An application of large-scale online learning,” in *Proceedings of the 26th International Conference on Machine Learning*, 2009, pp. 681–688.
- [65] H. Drucker, D. Wu, and V. N. Vapnik, “Support vector machines for spam categorization,” *IEEE Transactions on Neural Networks*, vol. 10, no. 5, pp. 1048–1054, 1999.
- [66] B. E. Boser, I. M. Guyon, and V. N. Vapnik, “A training algorithm for optimal margin classifiers,” in *Proceedings of the Fifth Annual Workshop on Computational Learning Theory*, 1992, pp. 144–152.
- [67] V. N. Vapnik, *Statistical Learning Theory*. Wiley-Interscience, 1998.
- [68] C. J. C. Burges, “A tutorial on support vector machines for pattern recognition,” *Data Mining and Knowledge Discovery*, vol. 2, no. 2, pp. 121–167, 1998.
- [69] T. Joachims, “Text categorization with support vector machines: Learning with many relevant features,” in *Proceedings of the 10th European Conference on Machine Learning*, 1998, pp. 137–142.
- [70] B. Schölkopf and A. J. Smola, *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. MIT Press, 2002.
- [71] D. W. Hosmer, S. Lemeshow, and R. X. Sturdivant, *Applied Logistic Regression*. Wiley, 2013.
- [72] N. Cristianini and J. Shawe-Taylor, *An Introduction to Support Vector Machines and Other Kernel-Based Learning Methods*. Cambridge University Press, 2000.
- [73] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge University Press, 2004.
- [74] J. Nocedal and S. J. Wright, *Numerical Optimization*. Springer, 2006.
- [75] K. P. Bennett and C. Campbell, “Support vector machines: Hype or hallelujah?” *ACM SIGKDD Explorations Newsletter*, vol. 2, no. 2, pp. 1–13, 2000.
- [76] J. A. Hartigan and M. A. Wong, “Algorithm as 136: A k-means clustering algorithm,” *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, vol. 28, no. 1, pp. 100–108, 1975.

BIBLIOGRAPHY

- [77] A. K. Jain, M. N. Murty, and P. J. Flynn, *Data Clustering: A Review*. ACM, 1999, vol. 31, pp. 264–323.
- [78] D. Arthur and S. Vassilvitskii, “K-means++: The advantages of careful seeding,” in *Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, SIAM, 2007, pp. 1027–1035.
- [79] D. Aloise, A. Deshpande, P. Hansen, and P. Popat, “Np-hardness of euclidean sum-of-squares clustering,” *Machine Learning*, vol. 75, no. 2, pp. 245–248, 2009.
- [80] S. Z. Selim and M. A. Ismail, “K-means-type algorithms: A generalized convergence theorem and characterization of local optimality,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PAMI-6, no. 1, pp. 81–87, 1984.
- [81] R. L. Thorndike, “Who belongs in the family?” *Psychometrika*, vol. 18, no. 4, pp. 267–276, 1953.
- [82] P. J. Rousseeuw, “Silhouettes: A graphical aid to the interpretation and validation of cluster analysis,” *Journal of Computational and Applied Mathematics*, vol. 20, pp. 53–65, 1987.
- [83] J. H. Holland, *Adaptation in Natural and Artificial Systems*. MIT Press, 1992.
- [84] D. E. Goldberg, *Genetic Algorithms in Search, Optimization, and Machine Learning*. Addison-Wesley, 1989.
- [85] G. Mendel, “Versuche über pflanzenhybriden,” *Verhandlungen des naturforschenden Vereines in Brünn*, vol. 4, pp. 3–47, 1866.
- [86] G. H. Hardy, “Mendelian proportions in a mixed population,” *Science*, vol. 28, no. 706, pp. 49–50, 1908.
- [87] M. Mitchell, *An Introduction to Genetic Algorithms*. MIT Press, 1996.
- [88] C. Darwin, *On the Origin of Species by Means of Natural Selection*. John Murray, 1859.
- [89] D. J. Futuyma, *Evolution*, 3rd. Sinauer Associates, 2013.
- [90] J. Huxley, *Evolution: The Modern Synthesis*. London: George Allen & Unwin, 1942.
- [91] S. Russell and P. Norvig, *Artificial Intelligence: A Modern Approach*, 3rd. Prentice Hall, 2010.
- [92] C. R. Reeves and J. E. Rowe, *Genetic Algorithms: Principles and Perspectives — A Guide to GA Theory*. Springer, 2002.
- [93] T. Bäck, D. B. Fogel, and Z. Michalewicz, *Handbook of Evolutionary Computation*. Oxford University Press, 1997.
- [94] C. R. Reeves, “Genetic algorithms,” in *Handbook of Evolutionary Computation*, T. Bäck, D. B. Fogel, and Z. Michalewicz, Eds., Oxford University Press, 1995, G1.2:1–G1.2:20.
- [95] X. Yao, Y. Liu, and G. Lin, “Evolutionary programming made faster,” *IEEE Transactions on Evolutionary Computation*, vol. 3, no. 2, pp. 82–102, 1999.
- [96] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, “A fast and elitist multiobjective genetic algorithm: Nsga-ii,” *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 2, pp. 182–197, 2002.
- [97] S. Albawi, T. A. Mohammed, and S. Al-Zawi, “Understanding of a convolutional neural network,” in *2017 International Conference on Engineering and Technology (ICET)*, IEEE, 2017, pp. 1–6.

BIBLIOGRAPHY

- [98] O. Abdel-Hamid, L. Deng, and D. Yu, “Exploring convolutional neural network structures and optimization techniques for speech recognition.,” in *Interspeech*, vol. 11, 2013, pp. 73–5.
- [99] V. Dumoulin and F. Visin, “A guide to convolution arithmetic for deep learning,” *arXiv preprint arXiv:1603.07285*, 2016.
- [100] I. Goodfellow, D. Warde-Farley, M. Mirza, A. Courville, and Y. Bengio, “Maxout networks,” in *International conference on machine learning*, 2013, pp. 1319–1327.
- [101] A. S. Winoto, M. Kristianus, and C. Premachandra, “Small and slim deep convolutional neural network for mobile device,” *IEEE Access*, vol. 8, pp. 125 210–125 222, 2020.
- [102] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” *arXiv preprint arXiv:1412.6980*, 2014.
- [103] A. Mikołajczyk and M. Grochowski, “Data augmentation for improving deep learning in image classification problem,” in *2018 international interdisciplinary PhD workshop (IIPhDW)*, IEEE, 2018, pp. 117–122.