

**ĐẠI HỌC QUỐC GIA HÀ NỘI
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ**

Hoàng Xuân Huấn

Giáo trình
HỌC MÁY

MỤC LỤC

LỜI NÓI ĐẦU.....	7
CHƯƠNG 1 GIỚI THIỆU	9
1.1. HỌC MÁY LÀ GÌ?	9
1.1.1. Khái niệm học máy	9
1.1.2. Tại sao cần nghiên cứu học máy?	10
1.1.3. Một số lĩnh vực liên quan.....	11
1.1.4. Các bài toán học thiết lập đúng đắn	11
1.2. MỘT SỐ LỐP BÀI TOÁN ỨNG DỤNG ĐIỀN HÌNH	12
1.3. KIẾN TRÚC VÀ THIẾT KẾ MỘT HỆ HỌC	14
1.3.1. Đồi sánh vân tay.....	14
1.3.1.1 Lược đồ đồi sánh dựa trên đặc trưng chi tiết.....	14
1.3.1.2. Các thành phần của hệ nhận dạng mẫu.....	16
1.3.1.3. Thiết kế hệ nhận dạng.....	17
1.3.2. Tìm đường đi tối ưu cho robot	19
1.3.2.1. Thuật toán tăng cường để tìm gần đúng đường đi tối ưu.....	19
1.3.2.2. Thiết kế hệ học cho bài toán tổng quát.....	20
KẾT LUẬN	22
BÀI TẬP	22
CHƯƠNG 2 HỌC CÓ GIÁM SÁT.....	23
2.1 CÁC BÀI TOÁN VÀ VÍ DỤ ĐIỀN HÌNH.....	23
2.1.1. Bài toán học khái niệm	23
2.1.2. Bài toán học nhiều lớp	26
2.1.3. Bài toán nội suy và hồi quy	26
2.2. HỌC QUY NẠP.....	27
2.2.1. Phát biểu bài toán	27
2.2.2. Một số khái niệm cơ bản	27
2.3. HỌC KHÁI NIỆM VÀ BÀI TOÁN TÌM KIẾM.....	28
2.3.1. Thuật toán tìm kiếm giả thuyết chi tiết nhất	29
2.3.2. Các thuật toán loại trừ ứng cử.....	32
2.3.2.1. Thuật toán liệt kê loại trừ ứng cử(List-then-eliminate algorithm).....	33
2.3.2.2. Một cách biểu diễn compact đối với không gian tướng thuật.....	33
2.3.2.3. Thuật toán loại trừ ứng cử.....	34
2.4. HỌC HÀM NỘI SUY VÀ HỒ QUY.....	37
2.4.1. Phương pháp trực tiếp tìm hàm nội suy	37

2.4.2. Tìm hàm hồi quy	38
2.5. MỘT SỐ VẤN ĐỀ LIÊN QUAN.....	39
2.5.1. Khuynh hướng quy nạp.....	39
2.5.2. Học gần đúng theo xác suất	40
2.5.3. Chiều Vapnik-Chervonenkis.....	42
KẾT LUẬN	43
BÀI TẬP	44
CHƯƠNG 3 HỌC BẰNG CÂY QUYẾT ĐỊNH	46
3.1. BIỂU DIỄN GIẢ THUYẾT BẰNG CÂY QUYẾT ĐỊ NH	46
3.2. CÁC THUẬT TOÁN HỌC.....	49
3.2.1. Thuật toán học ID3	49
3.2.1.1. Các khái niệm.....	49
3.2.1.2. Mô tả thuật toán	51
3.2.1.3. Nhận xét về ID3	53
3.2.2. Thuật toán C4.5	55
3.3. MỘT SỐ VẤN ĐỀ KHÁC TRONG HỌC BẰNG CÂY QUYẾT ĐỊ NH.....	57
3.3.1. Phù hợp trội và cách khắc phục	57
3.3.2. Tiêu chuẩn chọn thuộc tính	59
3.3.3. Xử lý giá trị thuộc tính bị thiếu của mẫu	60
3.3.4. Cây phân lớp và hồi quy	61
KẾT LUẬN	62
BÀI TẬP	63
CHƯƠNG 4 PHÂN LỚP MẪU NHỜ HÀM PHÂN BIỆT	65
4.1. HÀM PHÂN BIỆT VÀ MIỀN QUYẾT ĐỊ NH.....	65
4.2. CÁC MÔ HÌNH TUYẾN TÍNH	68
4.2.1. Tách được bởi siêu phẳng	68
4.2.2. Thuật toán học perceptron	70
4.2.3. Thuật toán bình phương tối thiểu	72
4.2.4. Phân lớp khoảng cách cực tiểu	75
4.2.4.1. Các metric trong không gian đặc trưng	76
4.2.4.2. Phân biệt tuyến tính Euclidean	77
4.2.4.3. Phân biệt tuyến tính Mahalanobis	78
4.2.5. Máy vectơ tia	79
4.2.5.1. Các lớp tách được tuyến tính	79
4.2.5.2. Các lớp không tách được tuyến tính	81
4.2.5.3. Hàm nhân (Kernel function)	82
4.3. BÀI TOÁN TÝ LỆ CHIỀU	83
KẾT LUẬN	85
BÀI TẬP	85
CHƯƠNG 5 HỌC THỐNG KÊ	87
5.1. LÝ THUYẾT QUYẾT ĐỊ NH BAYES	87
5.1.1. Bài toán và các quy tắc quyết định	87
5.1.2. Quyết định Bayes trong học khái niệm	89
5.1.3. Giả thuyết có khả năng nhất và sai số bình phương tối thiểu	91

5.2. PHÂN LỚP BAYES.....	92
5.2.1. Các quy tắc phân lớp MAP và ML.....	93
5.2.2. Phân lớp cực tiểu rủi ro.....	95
5.2.3. Bộ phân lớp tối ưu bayes	98
5.2.4. Phân lớp Bayes ngây thơ(Naïve Bayes).....	99
5.2.5. Phân lớp Bayes khi mỗi lớp có phân bố chuẩn	101
5.3. QUY TẮC QUYẾT ĐỊ NH K LÁNG GIỀNG GẦN NHẤT (K-NN).....	103
5.4. LỰA CHỌN ĐẶC TRƯNG VÀ PHÂN TÍCH THÀNH PHẦN CHÍNH.....	104
5.4.1. Lựa chọn đặc trưng	104
5.4.2. Phân tích thành phần chính (PCA).....	105
5.5. ĐÁNH GIÁ CÁC BỘ PHÂN LỚP	106
5.5.1. Ước lượng lỗi của bộ phân lớp	106
5.5.2. Phương pháp k-tập (k-folds) đánh giá phương pháp học	107
5.5.3. So sánh các bộ phân lớp	108
5.5.4. Một số đại lượng và thông tin khác	109
KẾT LUẬN	110
BÀI TẬP	111
CHƯƠNG 6 HỌC KHÔNG GIÁM SÁT.....	112
6.1. ƯỚC LƯỢNG HÀM MẶT ĐỘ.....	112
6.1.1. Kỹ thuật có tham số.....	112
6.1.2. Kỹ thuật phi tham số	114
6.2. PHÂN CỤM DỮ LIỆU	117
6.2.1. Bài toán phân cụm dữ liệu	117
6.2.2 . Vấn đề chuẩn hóa dữ liệu	119
6.3.3. Phương pháp phân cấp	120
6.3.4. Phương pháp phân hoạch	122
6.3.5. Phân cụm bán giám sát.....	127
KẾT LUẬN	128
BÀI TẬP	129
CHƯƠNG 7 MẠNG NƠRON NHÂN TẠO.....	130
7.1. GIỚI THIỆU	130
7.1.1. Cấu tạo và đặc điểm của mạng nơron sinh học	130
7.1.2. Mô hình và kiến trúc mạng nơron	131
7.1.2.1. Cấu tạo của nơron	132
7.1.2.2. Các kiểu kiến trúc mạng nơron.....	133
7.1.3. Đặc điểm của mạng nơron.....	135
7.2. PERCEPTRON	137
7.2.1. Perceptron của Roseblatt	137
7.2.2. Mạng ADALINE	138
7.3. PERCEPTRON NHIỀU TẦNG (MẠNG MLP)	143
7.3.1. Kiến trúc mạng.....	143
7.3.2. Thuật toán huấn luyện lan truyền ngược (BP).....	144
7.4. MẠNG HÀM COSBÁN KÍNH (MẠNG RBF)	151
7.4.1. Kiến trúc mạng RBF	151

7.4.2. Huấn luyện mạng RBF	152
7.4.2.1. Các thuật toán dựa trên tìm cực tiểu SSE	152
7.4.2.2. Phương pháp huấn luyện lắp mạng nội suy	153
KẾT LUẬN	158
BÀI TẬP	158
CHƯƠNG 8 CÁC MÔ HÌNH HỌC ĐỊA PHƯƠNG.....	160
8.1. HỌC K-LÁNG GIỀNG GẦN NHẤT (K-NN)	160
8.1.1. Lược đồ tổng quát	160
8.1.2. Thuật toán nghị ch đảo khoảng cách	161
8.2. HỒ QUY TRỌNG SỐ ĐỊ A PHƯƠNG	162
8.2.1. Lược đồ tổng quát	162
8.2.2. Hồi quy tuyến tính đị a phuong	163
8.3. MẠNG NƠRON RBF HỒ QUY	165
8.3.1. Đặt vấn đề	165
8.3.2. Xây dựng mạng nơron RBF hồi quy.....	165
8.4. MẠNG RBF ĐỊ A PHƯƠNG	166
8.4.1. Kiến trúc và thủ tục xây dựng mạng	166
8.4.2. Thuật toán phân cụm nhòcây k-d.....	168
8.5. LẬP LUẬN DỰA TRÊN TÌNH HUỐNG (CBR)	170
KẾT LUẬN	171
BÀI TẬP	173
CHƯƠNG 9 HỌC TĂNG	174
9.1. TÁC TỬ VÀ CÁC BÀI TOÁN HỌC	174
9.1.1. Một số ví dụ	174
9.1.2. Tác tử.....	176
9.1.3. Các bài toán học	178
9.2. HỌC Q (Q learning)	180
9.2.1. Học Q trong các bài toán đơn đị nh.....	180
9.2.1.1. Hàm Q	181
9.2.1.2. Một thuật toán học Q	181
9.2.2. Học Q trong các bài toán ngẫu nhiên	187
9.3. PHƯƠNG PHÁP TỐ ƯU ĐÀN KIẾN (ACO)	189
9.3.1. Phát biểubài toán tối ưu tổ hợp tổng quát	189
9.3.2. Thuật toán tổng quát.....	190
KẾT LUẬN	193
BÀI TẬP	194
CHƯƠNG 10 KẾT HỢP CÁC BỘ HỌC	195
10.1. LÝ DO NÊN HỌC TẬP THỂ.....	195
10.2. PHƯƠNG PHÁP BỔ PHIẾU	196
10.3. KỸ THUẬT TẠO VÀ KẾT HỢP BỘ NHÂN DẠNG COSỞ.....	198
10.3.1. Nhặt theo gói.....	198
10.3.2. Nhặt đị nh hướng.....	199
10.3.3. Rùng ngẫu nhiên.....	202
10.4. KIẾN TRÚC BẮC THANG.....	203

KẾT LUẬN	204
BÀI TẬP	205
TÀI LIỆU THAM KHẢO	206
BẢNG CHỮ VIẾT TẮT	207

LỜI NÓI ĐẦU

Học máy là một phần của lĩnh lực trí tuệ nhân tạo, mục đích của nó là phát triển các phương pháp xây dựng các chương trình máy tính tự động cải tiến chất lượng thực hiện nhờ sử dụng dữ liệu hoặc kinh nghiệm đã có. Những nghiên cứu trong lĩnh vực này nhằm đáp ứng các bài toán đa dạng trong thực tiễn, bắt đầu từ các bài toán nhận dạng ngôn ngữ, tiếng nói, phân tích dữ liệu thống kê đến các bài toán điều khiển tự động...

Cùng với sự phát triển nhanh chóng của công nghệ thông tin, đã có nhiều ứng dụng thành công của học máy trong nhiều lĩnh vực khác nhau như: các hệ nhận dạng tiếng nói, chữ viết tay; tiếp thị (marketing) nhờ phân tích dữ liệu bán lẻ, các hệ thống điều khiển tự động Trong đó, ứng dụng trung tâm của học máy là khám phá tự động tri thức từ dữ liệu nhằm trợ giúp quyết định.

Giáo trình này giới thiệu kiến thức nhập môn trong thời lượng ba tín chỉ cho sinh viên của các ngành trong khoa công nghệ thông tin. Vì học máy là lĩnh vực quan trọng, hướng ứng dụng, phát triển nhanh và đa dạng nên với thời lượng như vậy không thể chuyển tải nhiều. Hơn nữa, để hiểu thấu đáo các kỹ thuật học máy thì sinh viên phải có nền tảng tốt về toán, đặc biệt là giải tích ngẫu nhiên nên chúng tôi chú trọng vào các phương pháp cơ bản và hướng dẫn sử dụng mà không thể đi vào các kỹ thuật phát triển quá sâu.

Các phương pháp và thuật toán học là trung tâm của lĩnh vực học máy, ở đây được trình bày theo từng nhóm hướng tiếp cận và bài toán giải quyết thể hiện ở tiêu đề chương để người đọc dễ theo dõi và thấy được toàn cảnh. Tuy nhiên, việc phân loại các tiếp cận của phương pháp học có tính nhập nhằng, chẳng hạn, các thuật toán ở các chương 4 và chương 6 có thể đưa ra cách nhìn để xếp vào tiếp cận học thống kê. Vì vậy, chúng tôi chọn cách diễn đạt để nội dung phù hợp với tiếp cận của mỗi chương. Ngoại trừ hai chương đầu, nội dung cả các chương được viết tương đối độc lập và người dùng có thể đọc riêng rẽ. Tuy vậy, nếu đủ thời gian thì nên theo đúng trình tự được trình bày thì dễ tiếp thu hơn.

Chương đầu của giáo trình giới thiệu một định nghĩa cho học máy, phác họa bức tranh chung về bài học và quy trình thiết kế hệ học. Chương 2 các bài toán học

có giám sát, bắt đầu từ khái niệm chung về học quy nạp và hai thuật toán đơn giản để tìm tập luật trong học khái niệm nhằm cho người đọc thấy được đặc điểm chính của việc tìm kiếm giả thuyết trong không gian giả thuyết, sau đó là bài toán hồi quy và các vấn đề liên quan đến học quy nạp. Các phương pháp học cây quyết định cơ bản để tìm tập luật trong học có giám sát được trình bày trong chương 3. Chương 4 giới thiệu các mô hình tuyến tính để phân biệt mẫu. Các phương pháp dựa trên mô hình xác suất như lý thuyết quyết định Bayes và ứng dụng, phương pháp phân lớp k-láng giêng gần nhất, một số vấn đề liên quan tới bài toán học có giám sát như chọn đặc trưng và đánh giá bộ phân lớp được giới thiệu trong chương 5. Chương 6 giới thiệu các phương pháp học không giám sát thường gặp trong ước lượng hàm mật độ và phân cụm dữ liệu. Chương 7 dành cho giới thiệu các kiến thức chung về mạng noron và một số mạng truyền tải thường gặp nhất. Một số phương pháp học địa phương cho bài toán hồi quy và phương pháp lập luận dựa trên tình huống được trình bày trong chương 8. Chương 9 dành cho giới thiệu bài toán học tăng cường và ứng dụng. Một số phương pháp kết hợp các bộ học để tăng độ chính xác và một số tiếp cận để xây dựng các bộ nhận dạng cơ sở được giới thiệu trong chương cuối.

Trong trình bày, chúng tôi chú trọng giới thiệu các thuật toán dưới dạng dễ hiểu và dễ vận dụng. Về hình thức, tùy theo ngữ cảnh mà các thuật toán có thể được đặc tả dưới dạng giả mã hoặc mô tả rõ từng bước thực hiện để người đọc làm quen với cả hai dạng thông dụng này.

Dựa trên giáo trình này, Các độc giả muốn mở rộng kiến thức chung có thể tham khảo các tài liệu [1-3], khi cần tìm hiểu sâu hơn về học thống kê có thể tham khảo các tài liệu [4-7], để có nội dung đầy đủ hơn về nhận dạng mẫu có thể tham khảo các tài liệu [8-12], còn để có kiến thức hệ thống và đầy đủ hơn về mạng noron thì có thể tham khảo các tài liệu [13-15].

Mặc dù đã được dùng làm bài giảng từ nhiều năm nay nhưng là lần đầu xuất bản nên chắc chắn giáo trình còn nhiều thiếu sót, chúng tôi mong nhận được các ý kiến góp ý để giáo trình được hoàn thiện hơn.

Tác giả

Chương 1

GIỚI THIỆU

1.1. HỌC MÁY LÀ GÌ?

1.1.1. Khái niệm học máy

Khái niệm học có nghĩa rộng giống như sự thông minh, bao gồm cả quá trình và khó có một định nghĩa chính xác. Theo nghĩa tự điển, học là quá trình thu nhận kiến thức, kỹ năng do người khác truyền lại hoặc đọc đi, đọc lại, nghiên ngẫm ghi nhớ (học thuộc lòng). Rộng hơn, học bao gồm cả quá trình đúc rút tri thức từ các quan sát, trải nghiệm thực tiễn.

Học máy (machine learning) mang hai nghĩa thông dụng: 1) sử dụng máy tính để khám phá tri thức từ dữ liệu, 2) sự học trong máy (tác tử: agent). Về phương diện công nghệ, học máy là một lĩnh vực của trí tuệ nhân tạo, trong đó nghiên cứu các kỹ thuật xây dựng và phát triển các chương trình máy tính có thể thích nghi và "học" từ các dữ liệu mẫu hoặc kinh nghiệm. Đến nay, đã có nhiều định nghĩa cho khái niệm này, tuy nhiên khó có một định nghĩa thỏa đáng được mọi người thừa nhận. Định nghĩa sau phát triển từ định nghĩa của T. Mitchell cho ta cách nhìn toán học của một chương trình học khi nghiên cứu, thiết kế.

Định nghĩa 1.1. Một chương trình máy tính được gọi là học từ dữ liệu/kinh nghiệm E đối với lớp nhiệm vụ T và độ đo mức thực hiện P nếu việc thực hiện các nhiệm vụ T của nó khi đo bằng P được cải tiến nhờ dữ liệu hoặc kinh nghiệm E.

Theo định nghĩa này, người ta cần tối ưu hóa độ đo thực hiện P dựa trên phân tích dữ liệu/ kinh nghiệm E để tìm cách thực hiện nhiệm vụ T tốt nhất. Để hiểu rõ hơn, ta hãy làm quen với một số ví dụ và một số vấn đề liên quan.

Một số ví dụ

Ví dụ 1. Phân tích dữ liệu bán lẻ của siêu thị

Hàng ngày các siêu thị bán ra một lượng lớn những mặt hàng phong phú và lưu lại các hóa đơn thanh toán (bản sao giờ hàng). Từ các dữ liệu bán lẻ có được, ta có thể phân tích các giờ hàng để tiên đoán được một khách hàng mua mặt hàng A thì sẽ mua mặt hàng B với xác suất bao nhiêu? Nếu xác suất này là lớn thì ta nên xếp

các mặt hàng này gần nhau, như thế tiện cho khách hàng và lượng hàng bán được cũng tăng lên so với việc để khách hàng phải tìm kiếm khắp nơi.

Rộng hơn, nếu có mô hình phân tích tốt, ta cũng có thể dự đoán được lượng hàng cần đáp ứng trong thời gian tới, xu thế sở thích của khách hàng, trên cơ sở đó có được quyết sách thích ứng. Trong ví dụ này T là dự báo, E là dữ liệu bán lẻ lưu trữ và P là độ chính xác của kết quả dự báo.

Ví dụ 2. Đổi sánh vân tay

Bài toán đổi sánh vân tay bắt nguồn từ hai bài toán truy nguyên và xác thực vân tay. Trong bài toán truy nguyên, người ta phải đổi sánh một ảnh vân tay thu được khi điều tra với các ảnh vân tay trong kho lưu trữ để xác định xem có vân tay nào trong kho lưu trữ là do cùng một ngón lăn ra với ảnh điều tra không. Trong bài toán xác thực, người ta cần xác minh ảnh vân tay đăng nhập (ta cũng sẽ gọi là điều tra) có đúng là cùng ngón sinh ra với ảnh đã đăng ký hay không? Cả hai bài toán này được đưa về bài toán đổi sánh cặp ảnh vân tay điều tra I_q với ảnh lưu trữ I_t để trả lời xem chúng cùng hay khác ngón sinh ra. Để xây dựng chương trình đổi sánh vân tay, người ta cần một tập dữ liệu bao gồm các cặp ảnh do cùng ngón và khác ngón sinh ra. Dựa trên tập dữ liệu này, một thuật toán được áp dụng để xây dựng chương trình. Người đọc dễ dàng xác định được các thành phần T, E, P cho bài toán này. Việc giải bài toán sẽ được đề cập ở mục 1.3

Ví dụ 3. Tìm đường đi ngắn nhất cho robot

Một mạng lưới gồm n trạm hoạt động tự động, khoảng cách giữa chúng khác nhau. Định kỳ, một robot cần đi kiểm tra các trạm này một lần. Giả sử Robot ghi nhớ được các trạm đã qua và độ dài đường đi giữa chúng, biết được các trạm cần kiểm tra tiếp. Khi đó qua từng trạm nó sẽ tìm đường đi tới trạm tiếp theo. Nếu có chiến lược học tốt, càng ngày robot sẽ tìm được đường đi ngắn hơn, thậm chí là tối ưu. Trong trường hợp này, P và E tương ứng là độ dài và các đường đi kiểm tra đã tìm được, T là tìm đường đi kiểm tra.

Bài toán này được tổng quát hóa như sau. Cho một hệ thống gồm một tập hữu hạn trạng thái S , với mỗi trạng thái s_i trong S có một tập tác động có thể $A(s_i)$ để khi hệ ở trạng thái này và chọn tác động $a \in A(s_i)$ thì hệ chuyển sang trạng thái mới $s(a, s_i)$ và mất chi phí nhất định. Ban đầu hệ ở trạng thái s_0 và kết thúc ở trạng thái thuộc tập F , ta cần tìm chuỗi tác động sao cho chi phí tích lũy nhỏ nhất.

1.1.2. Tại sao cần nghiên cứu học máy?

Sự thâm nhập mạnh mẽ của công nghệ thông tin kinh tế, xã hội công nghệ tri thức phát triển và tạo nên nhu cầu ứng dụng rộng rãi. Sau đây là một số phạm vi nghiên cứu, ứng dụng điển hình:

- Xây dựng các hệ nhận dạng mẫu dùng cho các thiết bị nghe nhìn cho robot và trong lĩnh vực tự động hóa, nhận dạng chữ viết tay, chuyển đổi các bài nói thành văn bản, phân tích ảnh tự động...
- Tạo ra các chương trình máy tính có thể hoạt động thích nghi với môi trường thay đổi hay thực hiện các nhiệm vụ mà ban đầu chưa xác định rõ, chẳng hạn, hệ lái tự động (máy bay, ôtô, tàu thủy...), trò chơi hay các điều khiển robot đa năng.
- Khám phá tri thức từ dữ liệu, đặc biệt là các cơ sở dữ liệu lớn, để trợ giúp ra quyết định. Chẳng hạn, phân tích thị trường, chẩn đoán bệnh của bệnh nhân và xác định phương án điều trị nhờ phân tích các bệnh án lưu trữ...

1.1.3. Một số lĩnh vực liên quan

Trong mấy chục năm qua, các nghiên cứu khoa học và ứng dụng của học máy phát triển nhanh, kết hợp các tiến bộ của nhiều lĩnh vực khác. Sau đây là các lĩnh vực góp phần quan trọng cho nghiên cứu học máy:

- Lý thuyết xác suất và thống kê: Là tiền thân của lĩnh vực học máy, trong đó, cho phép suy luận (inference) từ quan sát cụ thể để có kết luận khái quát nhờ thành tựu của giải tích ngẫu nhiên.
- Mô hình thần kinh sinh học. Việc nghiên cứu cơ chế hoạt động, xử lý phi tuyến và cấu tạo hệ thần kinh sinh học nói chung cho phép tạo nên các mô hình và thuật toán phỏng sinh học, đặc biệt là các mạng nơron.
- Lý thuyết độ phức tạp tính toán. Cho phép ước lượng độ phức tạp của các nhiệm vụ học đo qua các ví dụ đào tạo, số lỗi và các thủ tục tính toán...
- Lý thuyết điều khiển thích nghi. Các thủ tục học để điều khiển quá trình nhằm tối ưu hoá mục đích định trước hay học cách đoán các trạng thái tiếp theo của quá trình điều khiển...
- Tâm lý học: Cho phép mô phỏng các đáp ứng thực tế của con người, xây dựng các mô hình xử lý hiệu quả, chẳng hạn, học tăng cường.
- Các mô hình tiến hóa. Việc nghiên cứu các mô hình tiến hóa cho phép chúng ta đưa ra các thuật toán học mô phỏng tự nhiên như: thuật toán di truyền (GA), tối ưu đòn kiến (ACO), tối ưu bầy đàm (PSO), hệ miễn dịch nhân tạo (AIS)....

1.1.4. Các bài toán học thiết lập đúng đắn

Bài toán học được coi là thiết lập đúng khi thực sự có thể cải tiến được P qua kinh nghiệm E. Thông thường mô hình toán học để xây dựng thuật toán cho một bài

toán học đòi hỏi phải đúng đắn theo Hadamard. Trong các bài toán thực tế, Hadamard cho rằng một mô hình toán học ứng dụng được xem là thiết lập đúng đắn (well-posed problem) nếu nó có các tính chất:

- 1- Luôn tồn tại lời giải
- 2- Chỉ có duy nhất một lời giải
- 3- Khi các điều kiện ban đầu thay đổi ít thì *lời giải* cũng thay đổi ít.

Tuy nhiên, trong nhiều bài toán, điều kiện duy nhất một lời giải nhiều khi khó đáp ứng. Trong trường hợp đó người ta hay dùng phương pháp chính quy hóa (hiệu chỉnh hàm mục tiêu) để bài toán trở nên thiết lập đúng đắn.

Bài toán học phải được xác định đúng đắn dựa trên việc xác định rõ nhiệm vụ cụ thể, độ đo việc thực hiện và nguồn dữ liệu/ kinh nghiệm.

Phương pháp thông dụng nhất để đưa ra thuật toán cho các bài toán học là xây dựng một mô hình toán học phụ thuộc các tham số và dùng dữ liệu hoặc kinh nghiệm đã có để xác định giá trị thích hợp cho các tham số này.

1.2. MỘT SỐ LỚP BÀI TOÁN ỨNG DỤNG ĐIỂN HÌNH

Trong mục trên, ta đã làm quen với 3 ví dụ về bài toán học máy cụ thể. Các ứng dụng của học máy rất đa dạng, sau đây điểm qua một số lớp bài toán ứng dụng thường gặp.

Học các kết hợp

Trong nghiên cứu thị trường, người ta thường quan tâm tới các sự kiện X và Y cùng xảy ra và ước lượng xác suất có điều kiện $P(Y/X)$ để Y xảy ra với điều kiện X xảy ra. Công việc này gọi là *học các kết hợp*. Chẳng hạn, trong ví dụ 1 mục trước, nhà cung cấp cần phân tích giỏ hàng của khách hàng qua các hóa đơn để tìm xác suất $P(Y/X)$ để nếu khách mua sản phẩm X thì cũng mua sản phẩm Y, nhờ đó người ta có thể dự đoán được khả năng một khách hàng khi mua sản phẩm X thì sẽ mua sản phẩm Y.

Phân loại mẫu.

Các đối tượng thuộc tập X được phân thành k lớp dựa trên một tập con D đã biết nhãn. Chẳng hạn, các chữ số viết tay có 10 lớp, còn bài toán đối sánh vân tay thuộc loại hai lớp: trùng với ảnh lưu trữ hay không. Bài toán phân loại thuộc về học có giám sát và là bài toán thường gặp nhất trong ứng dụng.

Nhiều khi, người ta dùng từ *phân lớp* (classify) để thay cho *phân loại* (categorize), mặc dù thuật ngữ phân lớp có nghĩa rộng hơn, bao gồm cả *phân cụm* (cluster). Về sau, khi không gây nên nhầm lẫn, hai từ này có thể dùng thay cho nhau. Một ứng dụng

quan trọng của bài toán này là phân tích hồ sơ người vay để đánh giá rủi ro trong hoạt động tín dụng, trong đó dựa trên các yếu tố đặc trưng về khả năng tài chính của người vay, ngân hàng cần đoán nhận xem khách hàng có khả năng trả nợ đúng hạn không để cho vay.

Hồi quy hàm số

Trong thực tiễn, ta thường phải xác định giá trị hàm số tại những điểm chưa biết dựa trên giá trị hàm đã biết tại một số điểm. Bài toán này phát biểu như sau. Có một hàm chưa biết $f: R^n \rightarrow R$, nhưng biết được tập $D\{(x^j, y^j)\}_{j=1}^N$ trong $R^n \times R$ gồm N đối tượng quan sát được:

$$y = f(x_1^j, \dots, x_n^j) + \varepsilon(\mathbf{x}), \forall j = i, \dots, N, \quad (1.1)$$

trong đó $\varepsilon(\mathbf{x})$ là nhiễu trắng (các biến ngẫu nhiên độc lập, cùng phân bố và có kỳ vọng bằng không), $\mathbf{x}^j = (x_1^j, \dots, x_n^j) \in R^n$. Ta cần tìm hàm gần đúng $g(x_1, \dots, x_n)$ của $f(x)$ cho các đối tượng khác của X . Hàm g sẽ được gọi là hàm hồi quy của f . Nếu không quan tâm tới phân bố nhiễu thì ta gọi là bài toán xấp xỉ hàm. Khi các $\{\mathbf{x}^j\}_{j=1}^N$ phân bố rộng trên tập X và đòi hỏi:

$$g(\mathbf{x}^j) = y^j \quad \forall j = i, \dots, N \quad (1.2)$$

thì bài toán xấp xỉ này gọi là bài toán nội suy và hàm g sẽ được gọi là hàm nội suy của hàm f .

Học không giám sát

Các bài toán trên thuộc loại học có giám sát, trong đó ta biết được nhãn của tập dữ liệu quan sát được. Trong học không giám sát, ta chỉ đơn thuần phân tích đặc điểm của tập dữ liệu để có thông tin. Ba bài toán học không có giám sát thường gặp là: ước lượng hàm mật độ, phân cụm dữ liệu và đóng hàng (align) dựa trên cấu trúc..

Trong bài toán ước lượng hàm mật độ, có một tập mẫu dữ liệu lấy ngẫu nhiên cùng phân bố, ta cần dựa trên đó để ước lượng hàm mật độ của phân bố này.

Trong bài toán phân cụm dữ liệu, người ta chia tập dữ liệu thành các tập con (cụm) sao cho các phần tử trong cùng một cụm thì *giống nhau* hơn các phần tử khác cụm. Đặc tính *giống nhau* này thường được xác định bởi *khoảng cách*, đối tượng A giống đối tượng B hơn đối tượng C nếu khoảng cách từ A đến B nhỏ hơn khoảng cách từ A đến C. Khi tập dữ liệu cần xử lý lớn thì việc phân cụm cho phép ta giảm thời gian chạy của các ứng dụng. Tuy nhiên bài toán này là bài toán thiết lập không đúng đắn (ill-posed) và thường không duy nhất nghiệm.

Phân tích các dữ liệu có cấu trúc xâu/ trình tự (string/sequence) hoặc mạng dẫn đến các bài toán đóng hàng trong xử lý ngôn ngữ tự nhiên và tin sinh học. Việc đóng hàng các trình tự DNA, RNA, Protein và các mạng tương tác protein cho phép hiểu được các tính tương đồng và khác biệt về nhiều đặc điểm sinh học của các cá thể sinh vật và loài.

Học tăng cường

Trong nhiều trường hợp, đầu ra của hệ thống là một chuỗi tác động. Khi đó mỗi tác động riêng lẻ không quan trọng mà điều quan trọng là chuỗi tác động này cần đạt được mục đích định trước. Chẳng hạn, trong các trò chơi, một nước đi không thực sự quan trọng mà quan trọng là chuỗi nước đi đưa đến kết quả thắng, ví dụ 3 nêu ở trên là trường hợp riêng của loại này.

Tương tự như phương thức học nhò trải nghiệm cuộc sống, người ta có thể tạo ngẫu nhiên nhiều lời giải chấp nhận được và sau mỗi lần lặp điều chỉnh trọng số định hướng lựa chọn tác động để càng về sau chuỗi tác động có trọng số cao giúp ta đạt được mục đích cần có.

Bài toán học tăng cường sẽ khó hơn với các bài toán chỉ quan sát được từng phần hoặc cần hợp tác của nhiều tác tử (agent) để đạt được đích.

1.3. KIẾN TRÚC VÀ THIẾT KẾ MỘT HỆ HỌC

Kiến trúc và việc thiết kế hệ học khá đa dạng tùy theo từng bài toán mà các vấn đề then chốt phải giải quyết khác nhau.

Trong mục này ta xét bài toán nhận dạng vân tay và bài toán tìm đường đi tối ưu nhò học tăng cường để minh họa.

1.3.1. Đối sánh vân tay

Trở lại với bài toán đối sánh vân tay trong ví dụ 2: Dựa trên tập mẫu các cặp vân tay cùng hoặc khác nhau sinh ra, ta cần kết luận hai ảnh vân tay điều tra I_q và lưu trữ I_t có cùng ngón sinh ra hay không? Kiến trúc và quy trình thiết kế cho hệ này mang đặc điểm chung của một hệ nhận dạng mẫu (pattern recognition). Trước hết, ta cần làm quen với phương pháp đối sánh vân tay dựa trên đặc trưng chi tiết.

1.3.1.1 Lược đồ đối sánh dựa trên đặc trưng chi tiết

Các điểm kỳ dị trên vân tay như các điểm cùt, rẽ nhánh, tâm... được gọi là các điểm đặc trưng chi tiết (ĐTCT). Người ta có thể dựa trên tính tương đồng/ khác biệt

của các tập ĐTCT trên hai ảnh vân tay để kết luận chúng có cùng ngón sinh ra không. Lược đồ của phương pháp này như sau.

Từ hai ảnh vân tay, ta xác định hai tập điểm ĐTCT M_t và M_q tương ứng của I_t và I_q :

$$M_q = \{m_1, m_2, \dots, m_M\}, \quad m_i = (x_i, y_i, \theta) \quad \forall i = 1, \dots, M; \quad (1.3)$$

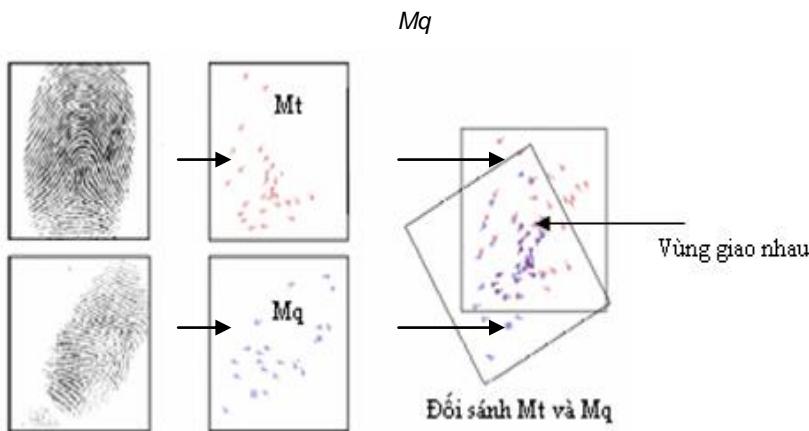
$$M_t = \{m'_1, m'_2, \dots, m'_N\}, \quad m'_i = (x'_i, y'_i, \theta') \quad \forall i = 1, \dots, N, \quad (1.4)$$

trong đó m_i/m'_i là vectơ điểm ĐTCT trích chọn từ ảnh tương ứng, $(x_i, y_i)/(x'_i, y'_i)$ là tọa độ của m_i/m'_i trên mặt phẳng chứa nó còn θ_i/θ'_i là hướng vân tại m_i/m'_i .

Vì miền sinh ra ảnh I_t và I_q không trùng nhau nên nói chung $M \neq N$. Để đối sánh, người ta xây dựng phép biến đổi T ít thay đổi tôpô của một ảnh để chồng ảnh I_q lên ảnh I_t sao cho các điểm ĐTCT tương ứng của chúng cũng “trùng khớp” với nhau từng đôi một nếu cùng một ngón sinh ra (xem hình 1.1). Một điểm ĐTCT m'_i trên I_t gọi là *trùng khớp* với một điểm ĐTCT m_k trên I_q nếu sau biến đổi ảnh m''_i của m'_i rơi vào một lân cận đủ bé của m_k như được mô tả trong hình 1.1. Tuy nhiên, do chất lượng ảnh hoặc hiện tượng biến dạng ngón tay khi lấy dấu tay nên chỉ có n cặp điểm *trùng khớp* và chúng được gọi là cặp điểm tương ứng. Sau khi xây dựng và dùng các phép biến đổi, độ giống nhau của hai ảnh vân tay được đặc trưng bằng độ đo tương tự $S(I_t, I_q)$ cho bởi công thức

$$S(I_t, I_q) = n^2 / M_t \cdot M_q. \quad (1.5)$$

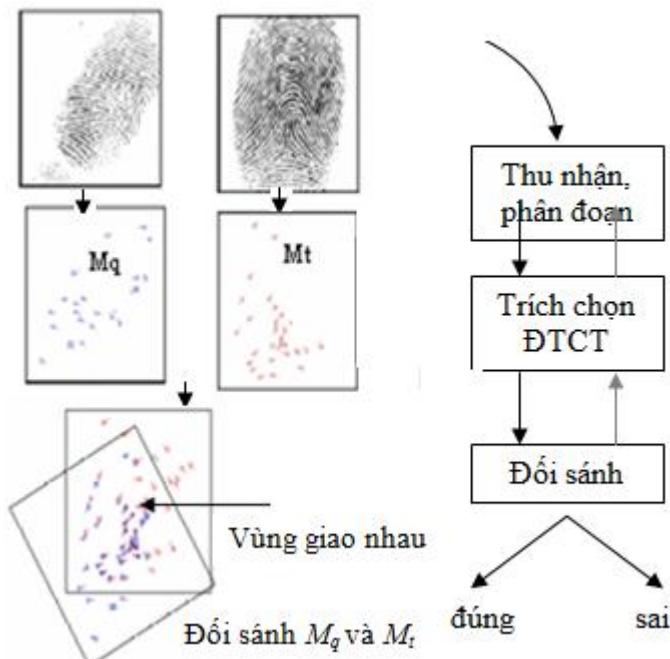
Nếu độ tương tự này nhỏ hơn ngưỡng cho trước thì nói rằng chúng không cùng một ngón sinh ra, ngược lại, ta kết luận cùng ngón. Ngưỡng này được xác định dựa trên tập mẫu đã có.



Hình 1.1. Xác định các cặp điểm ĐTCT tương ứng giữa hai tập M_t và M_q

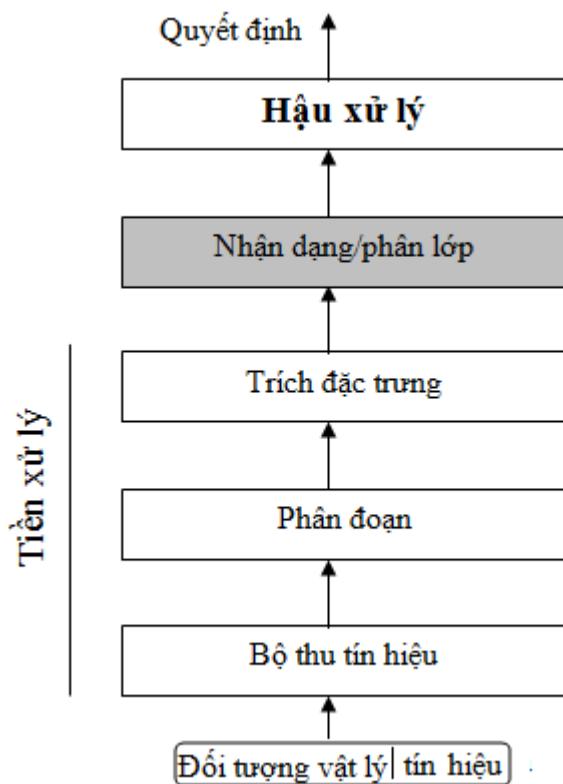
1.3.1.2. Các thành phần của hệ nhận dạng mẫu

Kiến trúc của hệ đối sánh vân tay được mô tả trong hình 1.2, trong đó bao gồm các thành phần: thu nhận, phân đoạn ảnh; trích chọn ĐTCT và đối sánh. Sau khi thu nhận ảnh từ chỉ bản hoặc sensor ta cần phân đoạn để tách phần ảnh rõ để trích chọn các điểm ĐTCT, sau đó sử dụng thành phần đối sánh để kết luận chúng cùng nhau sinh ra. Kết luận này sẽ được xem xét sử dụng.



Hình 1.2. Sơ đồ hệ đối sánh vân tay

Kiến trúc trên là trường hợp riêng của một hệ nhận dạng mẫu, được khái quát trong hình 1.3. bao gồm các thành phần: Tiền xử lý, nhận dạng/phân lớp và hậu xử lý.



Hình 1.3. Kiến trúc của một hệ nhận dạng

Trong thành phần tiền xử lý có các modun: Thu tín hiệu, phân đoạn và trích chọn đặc trưng. Tín hiệu vật lý được xử lý ở bộ thu để có tín hiệu rõ và tách tín hiệu quan tâm ở modun phân đoạn trước khi trích chọn đặc trưng để đưa vào bộ nhận dạng/phân lớp.

Bộ nhận dạng/phân lớp (về sau gọi gọn là *nhận dạng*) là trung tâm của hệ thống. Dựa trên phân tích đặc điểm của các giá trị thuộc tính được trích chọn, bộ này đưa ra nhãn cho đối tượng.

Đầu ra của bộ nhận dạng sẽ đưa vào bộ hậu xử lý xem kết luận đã tốt chưa, cần phân tích bổ sung nữa hay không và đưa ra quyết định cuối cùng.

Như vậy một hệ nhận dạng có khâu chính: tiền xử lí dữ liệu, phân lớp và hậu xử lí. Trong đó phân lớp là trung tâm của cả hệ thống.

1.3.1.3. Thiết kế hệ nhận dạng

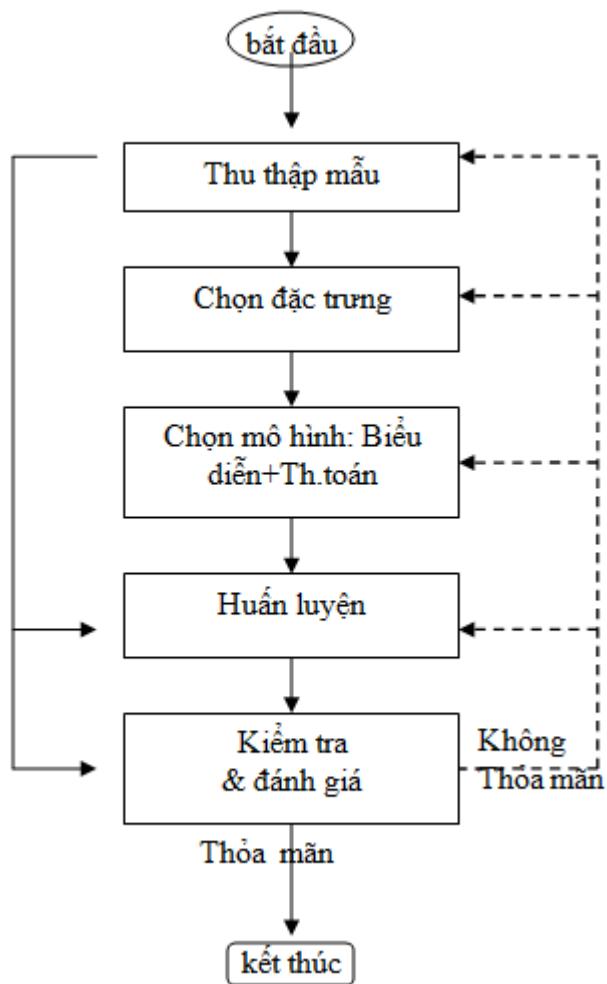
Khi có dự án xây dựng bộ nhận dạng, việc đầu tiên là thu thập tập mẫu hay dữ liệu. Đối với nhiều trường hợp học, cơ sở lý thuyết về lấy mẫu thống kê có vai trò quan trọng để đảm bảo chất lượng của hệ thống. Các dữ liệu thu được cần có giải pháp làm sạch và làm giàu để sử dụng.

Dựa trên đặc điểm bài toán và mô hình dự kiến áp dụng, cần quyết định chọn các đặc trưng: bao nhiêu đặc trưng là đủ, đặc trưng nào sẽ được chọn.

Khâu trung tâm là xây dựng mô hình bao gồm biểu diễn bài toán và quan trọng nhất, đưa ra thuật toán giải hiệu quả. Thông thường, một bài toán có thể có nhiều mô hình, thuật toán để giải, cần cân nhắc để chọn mô hình, thuật toán thích hợp.

Sau đó, người ta sử dụng dữ liệu và mô hình để huấn luyện bộ nhận dạng. Vì có nhiều mô hình, thuật toán giải, hơn nữa, các kỹ thuật cài đặt cũng ảnh hưởng tới hiệu quả của hệ nên cần phải đánh giá, kiểm tra qua thực nghiệm để đánh giá hiệu quả để có quyết định cuối cùng. Bởi vì các bước trên liên quan mật thiết với nhau nên việc thiết kế luôn có mối liên hệ ngược (Feedback) khi thực hiện và điều chỉnh mỗi bước. Quá trình này chính là quá trình thiết kế hệ, trong đó trung tâm là xây dựng và cài đặt thuật toán học.

Hình 1.4 biểu diễn quá trình xây dựng một bộ nhận dạng trong đó các bước có thể lặp lại cho tới khi đạt yêu cầu.



Hình 1.4. Quá trình thiết kế bộ nhận dạng

1.3.2. Tìm đường đi tối ưu cho robot

Ta quay lại với bài toán tìm đường đi cho robot trong ví dụ 3, khi biểu diễn mạng lưới các trạm bởi một đồ thị đầy đủ với mỗi nút là một trạm, các cạnh nối các đỉnh (i,j) có trọng số là độ dài đường đi giữa các đỉnh này (ban đầu robot chưa biết) thì bài toán có thể phát biểu như sau: Tìm chu trình Hamilton có độ dài ngắn nhất cho robot đi kiểm tra các trạm. Ngay cả khi robot biết trước độ dài các cạnh thì bài toán này cũng thuộc loại NP-khó nên không giải đúng được khi số trạm lớn.

Giả sử robot thực hiện nhiều lần tìm kiếm nhanh đường đi (không thực hiện kiểm tra) bằng việc lựa chọn ngẫu nhiên các đỉnh tiếp theo, khi đó thuật toán đơn giản sau có thể áp dụng.

1.3.2.1. Thuật toán học tăng cường để tìm gần đúng đường đi tối ưu

Trong thuật toán này, robot sẽ tìm cách đánh dấu các cạnh thuộc đường đi tốt nhất bằng chỉ số nghịch đảo độ dài đường đi này, các cạnh không thuộc đường đi ngắn nhất sẽ đánh dấu bằng nghịch đảo độ dài đường đi dài nhất. Điểm thú vị ở đây là độ dài mỗi cạnh chỉ góp phần vào độ dài đường đi mà nó thuộc vào chứ không có tính quyết định và robot chỉ tìm lời giải gần đúng, như vậy các chỉ số được gán cũng chỉ là “*xấp xỉ*”. Thuật toán thực hiện như sau.

Với số tự nhiên k và số L_0 cho trước, khởi tạo chỉ số cạnh bằng L_0 . Robot thực hiện lặp việc xây dựng đường đi ngẫu nhiên và điều chỉnh trọng số cạnh nhờ thủ tục bước ngẫu nhiên sau.

Khởi tạo từ đỉnh xuất phát u_0 , xây dựng tuần tự các đỉnh tiếp theo: nếu đang ở đỉnh i robot chọn đỉnh j tiếp sau một cách ngẫu nhiên với xác suất là

$$P_{i,j} = \frac{L_{i,j}}{\sum_{q \in J(i)} L_{i,q}}, \quad (1.6)$$

trong đó $L_{i,j}$ là chỉ số của cạnh (i,j) tương ứng còn $J(i)$ là những trạm chưa đến.

Quá trình tiếp tục cho đến khi đi hết các đỉnh thì xác định độ dài đường đi và các cạnh đã qua. Cứ sau một chu kỳ k lần thì robot thực hiện so sánh để tìm đường đi ngắn nhất. Ký hiệu L_{best} và L_{worst} tương ứng là nghịch đảo đường đi ngắn nhất và dài nhất tìm được, robot sẽ điều chỉnh chỉ số cho các cạnh theo công thức:

$$L_{i,j} \leftarrow (1 - \rho)L_{i,j} + \rho\Delta_{i,j} \quad (1.7)$$

trong đó $\rho \in (0,1)$ là hệ số cho trước, là đại lượng cho bởi công thức:

$$\Delta_{i,j} = \begin{cases} L_{best} & \text{nu } (i,j) \text{ thuc cong thuc} \\ L_{worst} & \text{cach canh con li} \end{cases} \quad (1.8)$$

Quá trình học thường dùng sau số bước lặp chọn trước. Sau khi huấn luyện, robot sẽ chọn cạnh có chỉ số cao nhất cho đường đi tới đỉnh tiếp theo khi thực hiện kiểm tra. Thuật toán này được đặc tả trong hình 1.5.

Người ta chứng minh được sau một số lần lặp đủ nhiều, lời giải tìm được có độ dài gần với lời giải tối ưu và chỉ số tương ứng trên cạnh của nó gần với chỉ số mong muốn. Hơn nữa, người ta có thể cố định trước giá trị chỉ số L_{best} và L_{worst} mà không cần phải thay đổi theo độ dài đường đi trong từng bước lặp, khi chọn thích hợp thì có thể cho kết quả tốt hơn.

1.3.2.2. Thiết kế hệ học cho bài toán tổng quát

Bài toán tìm đường đi tối ưu minh họa cho các vấn đề khi thiết kế hệ học cho lớp bài toán tìm chuỗi tác động tối ưu trong hệ thống hữu hạn trạng thái nêu ở trên. Sau đây là một số vấn đề cần lưu ý khi thiết kế hệ học:

```
Procedure : Tìm đường đi của robot;  
Begin  
    Initialize; // số k, chỉ số L0 cho các cạnh;  
    Repeat  
        Construct k solutions;// sử dụng thủ tục bước ngẫu nhiên;  
        Evaluate solutions;  
        Update edge indexes  
    End for;  
    Until End condition;  
    Choose the solutions;  
End;
```

Hình 1.5. Đặc tả thuật toán tìm đường đi robot

1) Chọn phương thức tạo sinh kinh nghiệm đào tạo

Kinh nghiệm đào tạo là tập các mẫu (lời giải) ta có, phương thức tạo ra /thu nhận mẫu này là vấn đề đầu tiên gặp phải khi thiết kế hệ. Ba vấn đề sau, chúng ta phải đối diện.

Thứ nhất. Cần chọn kiểu tạo sinh kinh nghiệm đào tạo để máy có thể sinh **trực tiếp** hay **gián tiếp** và các mẫu tạo sinh sau có thể có liên hệ với lời giải trước hoặc không. Các mẫu trong thuật toán nêu trên được sinh trực tiếp, còn trong sinh gián

tiếp thì mẫu có thể lấy từ các biên bản thực hiện trước đó hoặc có thêm thông tin hướng dẫn bổ sung.

Thứ hai. Chọn mức độ điều khiển của bộ học đối với chuỗi mẫu đào tạo. Trong trường hợp trên, chuỗi mẫu được sinh tự động theo thuật toán mà không có tương tác nào thêm của người thiết kế. Nhiều trường hợp, bộ học có thể trao đổi với người thiết kế (thầy) để định hướng tạo mẫu tiếp theo.

Thứ ba. Làm thế nào để phân bố thực hiện thí nghiệm dựa trên đánh giá độ đo đích thực P qua các độ đo trên mẫu đào tạo. Ở trên, ta tìm kiếm các mẫu mới và đánh giá theo định hướng quanh lời giải tốt nhất tìm được. Chúng ta cũng có thể khám phá và đánh giá theo hướng khác.

2) Chọn hàm đích

Trong trường hợp trên, hàm đích là độ dài đường đi, để dễ cài đặt thuật toán, ta xét nghịch đảo độ dài để đưa về bài toán tìm cực tiểu. Nhiều trường hợp, độ đo thực hiện không tường minh, chẳng hạn, khi thiết kế trò chơi với độ đo là tỷ lệ thắng thì mục tiêu này còn tùy thuộc trình độ đối thủ. Ngoài ra ta có mục tiêu thứ cấp (như là tham số) có thể chọn bằng nhiều cách như đã nêu trên.

3) Chọn biểu diễn và xấp xỉ tham số

Hàm đích/tham số đạt được trong quá trình học thường là giá trị gần đúng của hàm đích/tham số đặt ra. Ở trên cho thấy ta có nhiều cách chọn chỉ số L_{best} và L_{worst} và cách chọn tự nhiên dựa trên độ dài cạnh chưa hẳn tốt, việc cập nhật chỉ số này cũng có nhiều cách khác nhau, chẳng hạn, đơn giản nhất là chọn tham số ρ khác nhau. Khi đó, sau mỗi bước lặp ta có giá trị chỉ số khác nhau.

Quá trình trên thực chất là ta thiết lập và chọn thuật toán giải bài toán tối ưu

4) Xây dựng hệ học

Chất lượng của hệ học được đo bằng độ đo thực hiện, nó được quyết định bởi nhiều yếu tố, trong đó quyết định nhất là thuật toán học. Ngoài độ đo thực hiện, thời gian huấn luyện cũng là yếu tố được quan tâm khi đánh giá thuật toán. Thời gian chạy này tùy thuộc vào kỹ thuật cài đặt khi xây dựng bộ học. Chẳng hạn, nếu ta chọn kiểu dữ liệu khác nhau thì thời gian xử lý sẽ khác nhau và do đó thời gian huấn luyện cũng khác nhau.

Việc đánh giá hiệu năng của hệ chỉ thực hiện được qua thực nghiệm. Vì vậy phải đảm bảo tính khách quan và tin cậy được dựa vào các kết luận thống kê

KẾT LUẬN

Học máy nghiên cứu làm sao xây dựng các chương trình mà chúng có thể cải tiến chất lượng thực hiện nhiệm vụ dựa trên dữ liệu hoặc kinh nghiệm đã có. Sau đây là một số điểm cần lưu ý.

- Các thuật toán đã chứng tỏ rất có ý nghĩa trong thực tế ở lĩnh vực ứng dụng khác nhau như: Khám phá tri thức từ dữ liệu (Data mining), các công cụ hoạt động tự động có thể thích nghi với môi trường thay đổi, nghiên cứu y-sinh học...
- Học máy kết hợp ý tưởng và kỹ thuật từ nhiều lĩnh vực khác nhau như xác suất và thống kê, trí tuệ nhân tạo, tâm lý và thần kinh học, lý thuyết điều khiển, lý thuyết tính toán....
- Bài toán học phải được xác định đúng đắn dựa trên việc xác định rõ nhiệm vụ cụ thể, độ đo việc thực hiện và nguồn dữ liệu/ kinh nghiệm.
- Việc thiết kế hệ học bao gồm việc thu thập xử lý dữ liệu/ tạo sinh kinh nghiệm, xác định hàm đích phản ánh độ đo thực hiện, quan trọng nhất là một thuật toán học để học hàm đích dựa trên dữ liệu/kinh nghiệm đào tạo.

BÀI TẬP

1. Xác định T, E, P và đặc điểm thiết kế cho các bài toán sau:
 - a) Rô bot học lái trên đường 4 làn đường học lái xe robot
 - b) Nhận dạng chữ viết tay
 - c) Dựa trên các bệnh án đã có, chẩn đoán bệnh và đề xuất phác đồ điều trị cho bệnh nhân
2. Giả sử anh hay chị sẽ xây dựng hệ nhận dạng chữ viết tay/ mặt người. Hãy mô tả hệ nhận dạng và phác họa thiết kế.
3. So sánh các khâu thiết kế hệ học choi cờ và hệ đối sánh vân tay, tìm đường đi tối ưu của robot.

Chương 2

HỌC CÓ GIÁM SÁT

Chương này giới thiệu phương pháp học có giám sát từ tập mẫu quan sát được, bắt đầu từ các khái niệm chung về học quy nạp một bài toán phân lớp và các thuật toán đơn giản để học một khái niệm, sau đó xét bài toán hồi quy với thuộc tính và đầu ra nhận giá trị liên tục, cuối cùng thảo luận một số vấn đề liên quan như khuynh hướng, lỗi của giả thuyết và chiêu Vapnik-Chevonekis.

2.1 CÁC BÀI TOÁN VÀ VÍ DỤ ĐIỂN HÌNH

Hiểu theo nghĩa rộng, học có giám sát (supervised learning) dùng để chỉ các bài toán học mà trong dữ liệu/kinh nghiệm hay quá trình học có các thông tin định hướng để cải thiện độ đo thực hiện.

Hai bài toán học có giám sát thường gặp nhất là phân lớp và hồi quy hàm số dựa trên một tập mẫu đã biết nhãn. Tùy theo ngữ cảnh của bài toán cụ thể, chúng có tên gọi khác nhau: học khái niệm/ phân lớp/phân loại, hồi quy/ xấp xỉ/nội suy hàm số. Mô tả tổng quát cho các bài toán này như sau.

Cho một tập mẫu quan sát được $D = \{(x^k, y^k)\}_{k=1}^N$, trong đó $D_x = \{x^k\}_{k=1}^N$ là tập con của tập đối tượng X , còn $\{y^k\}_{k=1}^N$ là nhãn trong tập Y của các đối tượng tương ứng, ta cần tìm nhãn cho các đối tượng x mới trong X . Trước khi đi vào các phương pháp học, ta làm quen với các bài toán điển hình.

2.1.1. Bài toán học khái niệm

Nhiều trường hợp ta cần học một khái niệm tổng quát như: trâu, bò, mèo, chó, ôtô, xe máy... từ các trường hợp quan sát cụ thể và ta gọi là học khái niệm. Trong các trường hợp quan sát được, ta biết các đối tượng/ sự kiện (ví dụ) có phù hợp (dương tính) với khái niệm cần học hay không (âm tính), từ đó mà xác định xem các đối tượng/sự kiện khác là phù hợp hoặc không với khái niệm cần học.

Để giải bài toán này ta cần một *mô hình biểu diễn* các đối tượng nhò đặc điểm của chúng. Thông thường, mỗi đối tượng được đặc trưng bởi tập giá trị của các thuộc tính tương ứng. Nhãn của mỗi đối tượng có giá trị boolean: bằng 1 khi đối tượng dương tính và bằng 0 nếu âm tính. Như vậy nếu ký hiệu tập đối tượng là x và được biểu diễn bởi n thuộc tính A_1, \dots, A_n thì mỗi $x \in X$ có biểu diễn là $x = (x_1, \dots, x_n)$ trong đó x_i nhận giá trị trong tập giá trị V_i tương ứng của thuộc tính A_i . Từ các mẫu đã biết nhãn, ta cần xác định hàm logic $c: X \rightarrow \{0,1\}$ sao cho khi x đã biết nhãn thì $c(x)$ trùng với nhãn này.

Ví dụ 1 (giá trị thuộc tính rời rạc). Giả sử ta học khái niệm: *thời tiết của ngày bạn A thích chơi môn thể thao dưới nước*. Trước hết, ta cần có mô hình biểu diễn cho đặc điểm thời tiết ảnh hưởng tới quyết định chơi thể thao của bạn A.

Mô hình biểu diễn. Ta chọn biểu diễn thời tiết với các thuộc tính cho trong bảng 2.1: tiết trời, nhiệt độ, độ ẩm, gió, nước (nhiệt độ) với các giá trị có thể nhận được cho tương ứng trong ngoặc nhọn.

Bảng 2.1. Các thuộc tính biểu thị thời tiết ngày ưa thể thao

-Tập mẫu X: Các ngày với giá trị thuộc tính có thể nhận.
A ₁ : Tiết trời: {nắng, nhiều mây, mưa}
A ₂ : Nhiệt độ: {ấm, lạnh}
A ₃ : Độ ẩm: {trung bình, cao}
A ₄ : Gió: {mạnh, yếu}
A ₅ : Nước: {ấm, lạnh}
A ₆ : Dự báo: {không đổi, đổi}

Dựa trên tập dữ liệu thu được từ các ngày bạn A chơi hoặc không (chẳng hạn, được cho trong bảng 2.2.) ta sẽ dự đoán bạn A có đi chơi hoặc không khi biết giá trị các thuộc tính liên quan của một ngày mới. Ta sẽ tìm hàm dự báo cho dưới dạng tập luật: {nếu x_1 là a_1, \dots, x_n là a_n thì $c(x)=y$ }.

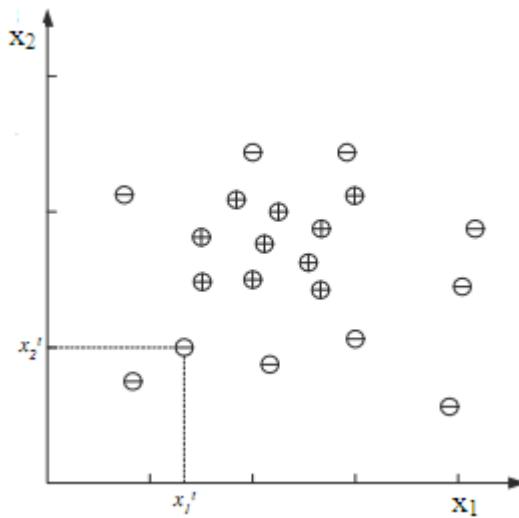
Bảng 2.2. Các ví dụ về những ngày chơi hoặc không của A

	Tiết trời	nhiệt độ	độ ẩm	gió	nước	dự báo	thích chơi?
1	nắng	ấm	trung bình	mạnh	ấm	không đổi	có
2	nắng	ấm	cao	mạnh	ấm	không đổi	có
3	mưa	lạnh	cao	mạnh	ấm	đổi	không
4	nắng	ấm	cao	mạnh	lạnh	đổi	có

Trong ví dụ trên, các thuộc tính nhận giá trị rời rạc, ví dụ sau minh họa bài toán có giá trị thuộc tính liên tục.

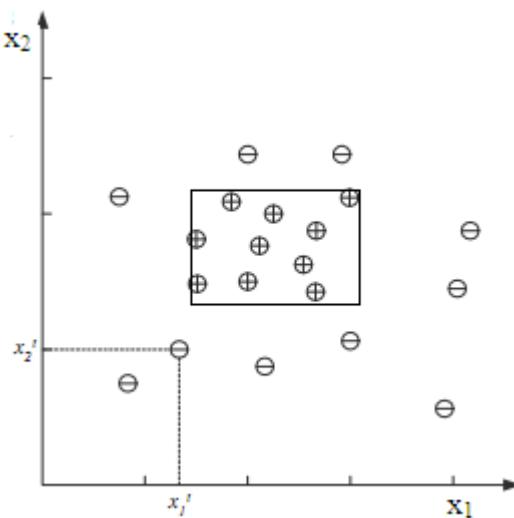
Ví dụ 2 (giá trị thuộc tính liên tục).

Bây giờ giả sử ta cần học khái niệm *người có sức khỏe tốt* dựa trên xét nghiệm hai chỉ số sinh hóa x_1 và x_2 nhận giá trị thực với tập mẫu quan sát được cho trong hình 2.1, trong đó đối tượng \oplus là dương tính (biểu thị sức khỏe tốt) còn đối tượng \ominus là âm tính (sức khỏe không tốt).



Hình 2.1. Biểu thị sức khỏe qua hai thuộc tính sinh hóa

Một lời giải khả dĩ cho trường hợp này là: nếu $p_1 \leq x_1 \leq \bar{p}_1$ và $p_2 \leq x_2 \leq \bar{p}_2$ thì $c(x) = \oplus$ ngược lại thì $c(x) = \ominus$, tức là những đối tượng có đặc trưng thuộc hình chữ nhật được minh họa trong hình 2.2 sẽ có sức khỏe tốt.



Hình 2.2. Một miền xác định người sức khỏe tốt dựa trên hai chỉ số sinh hóa

2.1.2. Bài toán học nhiều lớp

Bài toán học khái niệm có thể xem là bài toán học 2 lớp, lớp dương tính và lớp âm tính. Bài toán phân lớp tổng quát từ tập mẫu quan sát được phát biểu như sau.

Có k lớp C_1, \dots, C_k và tập quan sát $D = \{(x^j, y^j)\}_{j=1}^N, \{x^j: j = 1, \dots, N\} \subset X; \{y^j: j = 1, \dots, N\} \subset \{C_1, \dots, C_k\}$, trong đó X là tập đối tượng, ta cần phân lớp cho các đối tượng mới x trong X . Chẳng hạn, D là tập các chữ viết tay mẫu còn $\{C_1, \dots, C_k\}$ là tập các chữ cái, ta cần đoán nhận các chữ cái mới. Bài toán này cũng có thể xem gồm k bài toán học khái niệm, mỗi khái niệm là một lớp. Về sau, nếu không có gì đặc biệt, ta chỉ xét trường hợp hai lớp và người đọc có thể tổng quát hóa cho trường hợp nhiều lớp như là bài tập thực hành.

2.1.3. Bài toán nội suy và hồi quy

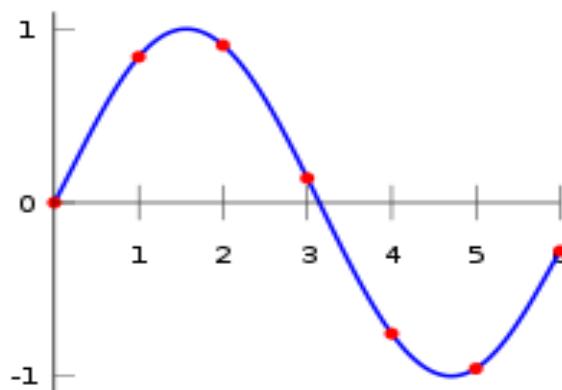
Ta trở lại với *bài toán nội suy* đã được nói đến trong mục 1.2: Có một hàm chưa biết $f: X \subset R^n \rightarrow R$, nhưng biết được tập quan sát $D = \{(x^j, y^j)\}_{j=1}^N$ trong $X \times R$:

$$y^j = f(x_1^j, \dots, x_n^j); j = i, \dots, N, \quad (2.1)$$

trong đó $x^j = (x_1^j, \dots, x_n^j)$. Ta cần xác định giá trị hàm $f(x)$ tại các điểm chưa quan sát được $x = (x_1, \dots, x_n)$. Các giá trị $f(x)$ này sẽ được xác định nhờ tìm một hàm $g: X \rightarrow R$ sao cho:

$$g(x_1^j, \dots, x_n^j) = y^j \forall j = i, \dots, N \quad (2.2)$$

và $g(x)$ được dùng để tính gần đúng $f(x)$. Hàm g sẽ gọi là hàm nội suy của f . Nếu x thuộc miền có các điểm $\{x^k\}_{k=1}^N \setminus \{x^j\}$ thì bài toán xác định $f(x)$ là *bài toán nội suy* (interpolation) còn x ra ngoài miền thì gọi là *ngoại suy* (extrapolation). Hình 2.3 minh họa đồ thị một hàm nội suy 2 biến dựa trên 7 mẫu quan sát được.



Hình 2.3. Một hàm nội suy từ 7 ví dụ quan sát

Khi dữ liệu quan sát được có nhiều thì đòi hỏi tìm hàm hồi quy thỏa mãn chặt biểu thức (2.2) không cần thiết mà người ta chỉ đòi hỏi gần đúng và tốt theo một tiêu chuẩn nào đó.

Bây giờ ta phát biểu bài toán học có giám sát cho bài toán học quy nạp từ các mẫu quan sát được.

2.2. HỌC QUY NẠP

Học hàm tổng quát hóa từ các mẫu quan sát được là trung tâm của học máy và được gọi là học quy nạp. Trước hết ta cần phát biểu toán học cho bài toán này.

2.2.1. Phát biểu bài toán

Cho X là tập đối tượng, Y là tập nhãn và một ánh xạ $c: X \rightarrow Y$ chưa biết nhưng có tập mẫu đã biết $D = \{(x, c(x)) / \forall x \in D_x \subset X\}$, tìm cần tìm ánh xạ $h: X \rightarrow Y$ sao cho:

$$h(x) = c(x) \quad \forall x \in D_x. \quad (2.3)$$

Ta xem ánh xạ h tìm được là ánh xạ c cần tìm. Tập D còn được gọi là tập huấn luyện hay tập đào tạo.

Dễ dàng thấy rằng các bài toán học khái niệm, phân lớp và nội suy đều trên là trường hợp riêng của bài toán này, còn khi xét biểu thức (2.3) theo nghĩa gần đúng thì nó bao gồm cả bài toán hồi quy.

2.2.2. Một số khái niệm cơ bản

Biểu diễn đối tượng

Thông thường, các đối tượng trong X được biểu diễn qua các giá trị trong tập các thuộc tính được chọn $\{A_1, \dots, A_n\}$ hay còn gọi là các đặc trưng của chúng. Mỗi thuộc tính A_i của đối tượng nhận giá trị trong tập giá trị V_i tương ứng. Với mỗi tập đối tượng X có thể có nhiều cách biểu diễn với các tập đặc trưng phù hợp. Việc trích và chọn đặc trưng vượt ra khỏi khuôn khổ của giáo trình này, và ta chỉ để cập tới một vài điểm then chốt mà thôi. Mỗi cách biểu diễn với các đặc trưng được chọn ta gọi là một **mô hình biểu diễn**.

Cho mô hình biểu diễn với tập đặc trưng $\{A_1, \dots, A_n\}$, khi đó mỗi đối tượng x trong X ứng với một vecto và ký hiệu là $x = \langle x_1, \dots, x_n \rangle$, trong đó các x_k có thể nhận giá trị thực hoặc giá trị định danh tùy theo đặc trưng của nó.

Không gian giả thuyết

Mỗi ánh xạ $h: X \rightarrow Y$ có thể xem xét làm lời giải (không nhất thiết thỏa mãn (2.3)) cho bài toán. Tức là nó xác định một cách gán nhãn cho các đối tượng trong X và

được gọi là một giả thuyết. Vì số ánh xạ từ X vào Y có lực lượng là $|Y|^{|X|}$, trong đó $|S|$ chỉ số phần tử của tập S , nên sẽ có rất nhiều giả thuyết khi X và Y là tập lớn. Người ta thường tìm gần đúng ánh xạ c trong tập con H nào đó của tập tất cả các ánh xạ mà ta có thể dự đoán hàm đích thuộc vào. Tập H này sẽ được gọi là không gian giả thuyết ((hypothesis), trong trường hợp này, mỗi $h \in H$ sẽ gọi là một giả thuyết nếu không xảy ra nhầm lẫn. Thuật toán tìm h sẽ gọi là *thuật toán học*, mỗi đối tượng x trong X cũng được gọi là một *mẫu* (instance).

Giả thuyết học quy nạp

Học quy nạp chỉ áp dụng được cho các bài toán thỏa mãn giả thiết rằng: bất cứ giả thuyết nào xấp xỉ tốt hàm đích trên tập mẫu đủ lớn thì cũng xấp xỉ tốt hàm đích trên các mẫu chưa biết.

Các bài toán học có tập giả thuyết thỏa mãn giả thiết trên gọi là bài toán thỏa mãn *giả thuyết học quy nạp*.

Giả thuyết phù hợp và không gian tường thuật

Định nghĩa 2.1. (*Giả thuyết phù hợp*) Một giả thuyết h gọi là *phù hợp* với tập dữ liệu đào tạo D nếu và chỉ nếu $h(x) = c(x)$ với mỗi $\langle x, c(x) \rangle$ trong D .

$$\text{consistent}(h, D) \equiv (\forall \langle x, c(x) \rangle \in D) h(x) = c(x).$$

Tập tất cả các giả thuyết phù hợp này gọi là không gian tường thuật đối với không gian giả thuyết H và tập mẫu đào tạo D .

Định nghĩa 2.2. (*không gian tường thuật*) Không gian tường thuật (được ký hiệu là $VS_{H,D}$) đối với không gian giả thuyết H và tập dữ liệu huấn luyện D là tập con của H phù hợp với tập mẫu D :

$$VS_{H,D} = \{h \in H \mid \text{consistent}(h, D)\}. \quad (2.4)$$

Khi các tập X và Y đều là hữu hạn thì VS_{HD} có thể xác định nhờ tìm kiếm trên toàn không gian giả thuyết. Tuy nhiên, như đã nói ở trên, khi X và Y lớn thì việc tìm kiếm như vậy không khả thi. Để minh họa xét bài toán tìm kiếm trong học khái niệm.

2.3. HỌC KHÁI NIỆM VÀ BÀI TOÁN TÌM KIẾM

Xét bài toán học khái niệm với thuộc tính nhận giá trị trên tập hữu hạn “*thời tiết của ngày bạn A thích chơi môn thể thao dưới nước*” nêu ở ví dụ 1 mục trước với tập mẫu X cho trong bảng 2.2.

Dễ dàng thấy có $3 \cdot 2 \cdot 2 \cdot 2 \cdot 2 = 96$ mẫu khác biệt trong X và có 2^{96} ánh xạ khác nhau từ X vào tập $\{0,1\}$. Như vậy, với mỗi tập dữ liệu huấn luyện D đã cho, số lượng giả thuyết có thể xem xét là quá nhiều nên việc tìm kiếm vét cạn không dễ.

Ta sẽ xét một thuật toán giải bài toán này theo tiếp cận hạn chế không gian giả thuyết.

2.3.1. Thuật toán tìm kiếm giả thuyết chi tiết nhất

Trước khi trình bày thuật toán, ta quy ước các thuộc tính có thể nhận những giá trị sau:

- Ký hiệu bởi ? nếu bất cứ giá trị nào của thuộc tính cũng được chấp nhận
- Ký hiệu bởi ϕ nếu không giá trị nào được chấp nhận
- Một giá trị cụ thể của thuộc tính.

Chẳng hạn, giả thuyết A chỉ chơi khi trời lạnh và độ ẩm cao được biểu diễn bởi:

$\langle ?, \text{lạnh}, \text{cao}, ?, ?, ? \rangle$.

Khi đó giả thuyết tổng quát nhất : mọi ngày đều chơi là

$\langle ?, ?, ?, ?, ?, ? \rangle$,

còn giả thuyết chi tiết nhất: không ngày nào chơi là

$\langle \phi, \phi, \phi, \phi, \phi, \phi \rangle$.

Bây giờ ta sẽ tìm lời giải cho bài toán này trong không gian giả thuyết H bao gồm tất cả các giả thuyết có dạng một liên kết các giá trị thuộc tính, chúng có thể là ?:; ϕ hoặc giá trị cụ thể, giá trị khái niệm đích c (tra thể thao) thuộc tập Θ_1 . Để dàng tính được có $5 \cdot 4 \cdot 4 \cdot 4 \cdot 4 \cdot 4 = 5120$ các giả thuyết khác nhau trong H. Tuy nhiên, các giả thuyết có ký hiệu ϕ đều như nhau nên chỉ còn lại $1 + 4 \cdot 3 \cdot 3 \cdot 3 \cdot 3 \cdot 3 = 973$ giả thuyết khác nhau về ngữ nghĩa. Với số lượng giả thuyết như vậy, ta có thể tìm được thuật toán tìm kiếm có hiệu quả không gian tàng thuật $VS_{H,D}$ nếu nó khác rỗng trong không gian giả thuyết này. Để giới thiệu thuật toán, ta cần đến cấu trúc thứ tự của tập các giả thuyết H vừa nêu ở trên.

Cấu trúc thứ tự “tổng quát đến chi tiết” của các giả thuyết.

Nhiều thuật toán học khái niệm được giải bằng cách tìm kiếm trong không gian giả thuyết nhờ dùng cấu trúc thứ tự *tổng quát đến chi tiết của không gian giả thuyết*. Nhờ cấu trúc này, ta có thể tìm kiếm vét cạn trong không gian giả thuyết (thậm chí có thể vô hạn) mà không cần đánh số giả thuyết. Để minh họa thứ tự này ta xét hai giả thuyết:

$$h_1 = \langle \text{nắng}, ?, ?, \text{mạnh}, ?, ?, ? \rangle$$

$$h_2 = \langle \text{nắng}, ?, ?, ?, ?, ?, ? \rangle$$

và xét tập mẫu được h_1, h_2 phân lớp dương. Bởi vì h_2 ít ràng buộc hơn nên nó có nhiều mẫu dương hơn h_1 . Khi đó ta nói h_2 tổng quát hơn h_1 .

Trước khi định nghĩa chính xác quan hệ thứ tự tổng quát ta cần khái niệm mẫu x thỏa mãn giả thuyết h .

Định nghĩa 2.3: (đối tượng thỏa mãn giả thuyết) Với mọi đối tượng $x \in X$ và giả thuyết $h \in H$, ta nói x **thỏa mãn** h nếu $h(x) = 1$.

Chú ý rằng khái niệm *thỏa mãn* này khác với khái niệm phù hợp trong định nghĩa 2.1, ở đây chỉ xét với mẫu dương. Nay giờ ta định nghĩa thứ tự tổng quát (chi tiết) của các giả thuyết.

Định nghĩa 2.4. (Quan hệ tổng quát-chi tiết) Giả sử h_j và h_k là hai hàm giá trị boolean xác định trên X . Ta nói h_j tổng quát hơn h_k (và viết là $h_j \geq_g h_k$) nếu và chỉ nếu bất cứ mẫu nào thỏa mãn h_k thì cũng thỏa mãn h_j :

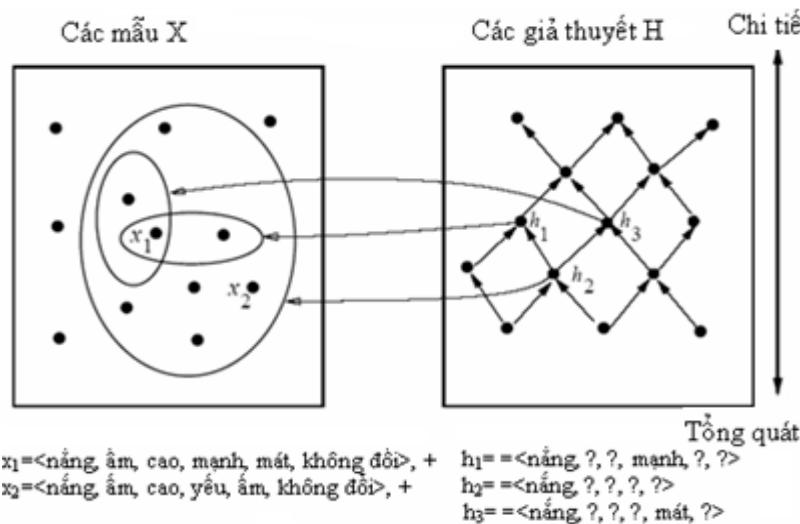
$$(\forall x \in X) [h_k(x)=1 \rightarrow h_j(x)=1]. \quad (2.5)$$

Ta cũng nói h_j tổng quát chặt hơn h_k (và viết là $h_j >_g h_k$) nếu và chỉ nếu:

$$(h_j \geq_g h_k) \wedge (h_k \neq h_j). \quad (2.6)$$

Trong các trường hợp trên ta cũng nói h_k chi tiết (chặt) hơn h_j .

Để minh họa định nghĩa này ta xét ba giả thuyết h_1, h_2, h_3 trong hình 2.4 ở bài toán của ví dụ đang xét. Trong hình này, bên trái biểu thị tập mẫu X , bên phải biểu thị tập H tất cả các giả thuyết, mỗi giả thuyết ứng với một tập con các mẫu mà nó phân lớp dương. Trong hình, h_2 tổng quát hơn h_1 và h_3 , còn h_1 và h_3 không so sánh được với nhau.



Hình 2.4. Cấu trúc thứ tự của các giả thuyết: bên trái là tập mẫu X , bên phải là tập giả thuyết H , mũi tên chỉ quan hệ “tổng quát hơn”

Chú ý rằng quan hệ \geq_g và $>_g$ là độc lập với khái niệm đích ; chúng chỉ phụ thuộc vào các mẫu thoả mãn hai giả thuyết mà không phụ thuộc vào sự phân lớp các mẫu phù hợp với khái niệm đích. Quan hệ \geq_g rất quan trọng vì nó cho một cấu trúc có ích trên không gian giả thuyết H đối với mọi bài toán học khái niệm. Thuật toán sau cho phép tìm giả thuyết chi tiết nhất sao cho mọi đối tượng dương trong tập mẫu đều thỏa mãn nó.

Thuật toán Find-S (tìm một giả thuyết chi tiết nhất)

Giả sử h là một giả thuyết phù hợp với tập mẫu D và không gian giả thuyết nêu trên chứa giả thiết phù hợp với D thì dễ dàng kiểm tra được giả thiết chi tiết nhất mà mọi mẫu dương đều thỏa mãn sẽ là giả thuyết phù hợp. Để tìm một giả thuyết này, ta bắt đầu từ giả thuyết chi tiết nhất $h = <\phi, \phi, \phi, \phi, \phi, \phi>$ và nhờ cấu trúc thứ tự đã biết, tổng quát hoá giả thuyết này mỗi khi có một mẫu dương không thỏa mãn nó. Thuật toán Find-S được trình bày trong bảng 2.3.

Chẳng hạn nếu đưa ví dụ 1 trong bảng 2.2:

$x_1 = <\text{nắng}, \text{âm}, \text{trung bình}, \text{mạnh}, \text{âm}, \text{không đổi}>$
thì $h \leftarrow <\text{nắng}, \text{âm}, \text{trung bình}, \text{mạnh}, \text{âm}, \text{không đổi}>$.

Nếu đưa thêm ví dụ 2: $x_2 = <\text{nắng}, \text{âm}, \text{cao}, \text{mạnh}, \text{âm}, \text{không đổi}>$
thì $h \leftarrow <\text{nắng}, \text{âm}, ?, \text{mạnh}, \text{âm}, \text{không đổi}>$.

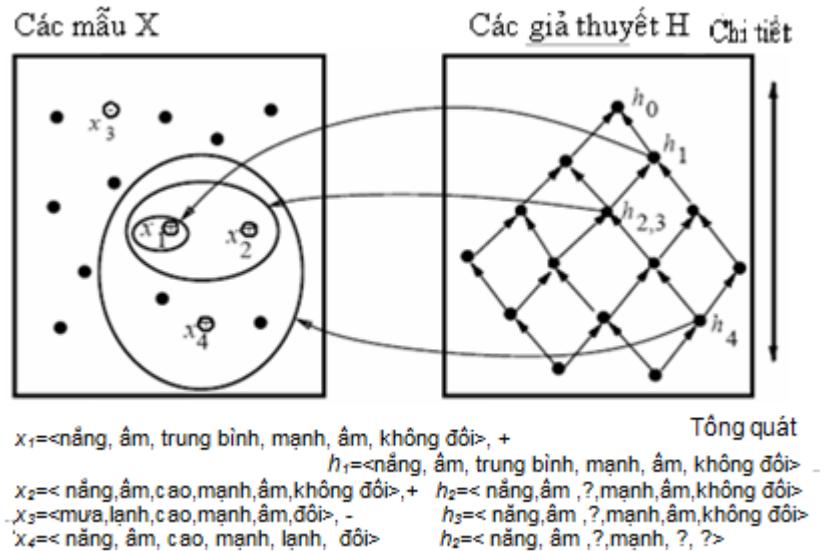
Bảng 2.3: Mô tả thuật toán Find-S

1. Khởi tạo h là giả thuyết chi tiết nhất; // $h \leftarrow <\phi, \phi, \phi, \phi, \phi, \phi>$
2. Với mỗi mẫu đào tạo dương x thực hiện:
Với mỗi thuộc tính A_i trong h , nếu ràng buộc a_i thoả mãn thì giữ nguyên,
ngược lại thay a_i trong h bởi ràng buộc tổng quát hơn thoả mãn x .

Thuật toán này bỏ qua các thí dụ sai (thứ 3) và tới ví dụ thứ tư:

$x_4 = <\text{nắng}, \text{âm}, \text{cao}, \text{mạnh}, \text{lạnh}, \text{đổi}>$ thì $h \leftarrow <\text{nắng}, \text{âm}, ?, \text{mạnh}, ?, ?, ?>$.

Thuật toán áp dụng cho ví dụ được minh họa trong hình 2.5.



Hình 2.5. Thuật toán Find-S cho ví dụ ở bảng 2.2

Thuật toán này hạn chế không gian giả thuyết trên tập các *giả thuyết cho bởi liên kết các ràng buộc của thuộc tính*. Khi đó ta tìm được giả thuyết chi tiết nhất phù hợp với các mẫu dương và nó cũng sẽ đúng với mẫu âm khi giả thuyết đúng và tập mẫu cho đúng. Tuy vậy còn một số câu hỏi chưa trả lời đốivới thuật toán này là:

- Thuật toán có đảm bảo hội tụ tới giả thuyết phù hợp với D không? Nếu có thì ngoài giả thiết tìm được, còn có bao nhiêu giả thuyết phù hợp và liệu ta có thể tìm ra giả thuyết đúng hay không? Nếu không thì ta cũng cần ước lượng được tính không chắc chắn của nó.
- Tại sao ta ưa giả thuyết chi tiết nhất? các giả thuyết khác thì sao? (tổng quát nhất hoặc trung gian)
- Nếu tập mẫu D có ví dụ sai thì sao?

Sau đây là một tiếp cận khác, trong đó ta tìm VS_{HD} mà không hạn chế không gian giả thuyết.

2.3.2. Các thuật toán loại trừ ứng cử

Trong mục này đưa ra thuật toán vét cạn để tìm tất cả các giả thuyết phù hợp với tập mẫu đào tạo, tức là toàn bộ không gian tường thuật:

$$VS_{H,D} = \{h \in H \mid \text{consistent}(h, D)\}$$

Sau đó giới thiệu một biến thể của nó dựa trên biểu diễn compact của không gian tường thuật.

2.3.2.1. Thuật toán liệt kê loại trừ ứng cử (List-then-eliminate algorithm)

Một cách đơn giản để biểu diễn không gian tường thuật là liệt kê mọi phần tử của nó. Ý tưởng này dẫn tới thuật toán liệt kê-loại trừ ứng cử được mô tả trong bảng 2.4. Thuật toán này khởi tạo toàn bộ không gian giả thuyết rồi sau đó loại trừ dần các giả thuyết không phù hợp khi đối sánh tuân tự với các mẫu quan sát được.

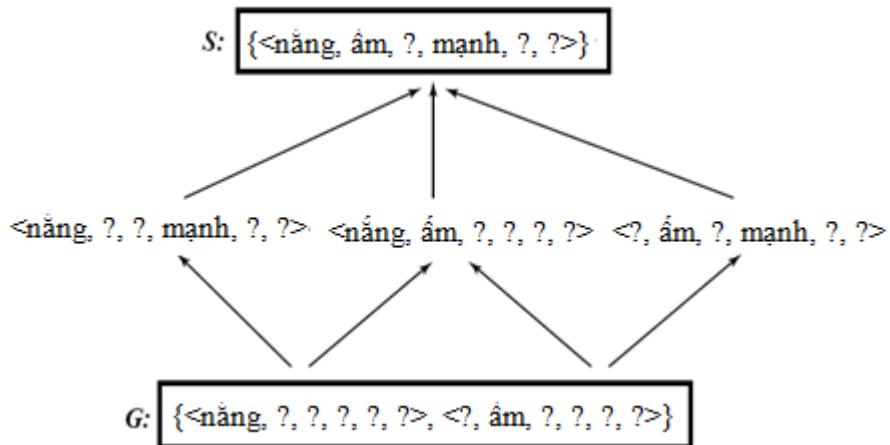
Bảng 2.4. Thuật toán liệt kê loại trừ ứng cử

1. $VS_{H,D} \leftarrow$ danh sách các giả thuyết trong H
2. Với mỗi $\langle x, c(x) \rangle$ trong D
loại trừ các giả thuyết $h \in VS_{H,D}$ mà $h(x) \neq c(x)$
3. Đầu ra $VS_{H,D}$

Về nguyên tắc thì thuật toán này có thể áp dụng khi không gian giả thuyết H hữu hạn nhưng trong thực hành thì thường không thể liệt kê và tìm kiếm vét cạn H . Một biến thể của thuật toán này được thực hiện dựa trên biểu diễn sau của không gian tường thuật

2.3.2.2. Một cách biểu diễn compact đối với không gian tường thuật

Người ta có thể biểu diễn không gian tường thuật nhờ thứ tự tổng quát đến chi tiết bằng cách chỉ ra cận dưới chi tiết nhất và cận trên tổng quát nhất của $VS_{H,D}$ mà không phải liệt kê chúng. Để minh họa cho cách biểu diễn này, ta trở lại với bài toán học khái niệm thích chơi thể thao với các ví dụ trong bảng 2.2. Thuật toán Find-S cho ta giả thuyết chi tiết nhất là: $\langle \text{nắng}, \text{âm}, ?, \text{mạnh}, ?, ? \rangle$ thực ra có sáu giả thuyết trong H phù hợp với tập mẫu trong bảng 2.2, các giả thuyết này được mô tả trong hình 2.6.



Hình 2.6. VSHD với biên tổng quát và chi tiết của khái niệm học từ bảng 2.2

Ý tưởng này dẫn đến thuật toán loại trừ ứng cử. Để trình bày thuật toán ta cần đến định nghĩa các tập biên tổng quát G và chi tiết S đối với không gian giả thuyết H và tập dữ liệu D .

Định nghĩa 2.5. (*Biên tổng quát và biên chi tiết*)

i) *Biên tổng quát* G đối với không gian giả thuyết H và tập dữ liệu đào tạo D là tập các giả thuyết tổng quát nhất của H phù hợp với D :

$$G \equiv \{g \in H \mid \text{consistent}(g, D) \wedge (\neg g' \in H) \mid g' >_g g \} \wedge \text{consistent}(g, D)$$

ii) *Biên chi tiết* S đối với không gian giả thuyết H và tập dữ liệu đào tạo D là tập các giả thuyết chi tiết nhất của H phù hợp với D :

$$S \equiv \{s \in H \mid \text{consistent}(s, D) \wedge (\neg s' \in H) \mid s >_g s' \} \wedge \text{consistent}(s, D)$$

Tập G và S này hoàn toàn có thể đặc tả $V_{H,D}$ nhò định lý sau.

Định lý 2.1. (*Biểu diễn không gian tường thuật*). Giả sử X là tập mẫu tùy ý và H là tập giả thuyết giá trị boolean xác định trên X . Cho $c: X \rightarrow \{0,1\}$ là một khái niệm đích xác định trên X và D là tập mẫu quan sát được tuỳ ý của nó : $D = \{x, c(x)\}$, khi đó với mọi X, H, c và D sao cho G và S được xác định thì

$$V_{H,D} = \{h \in H \mid (\exists s \in S)(\exists g \in G)(g \geq_g h \geq_g s) \}. \quad (2.7)$$

2.3.2.3. Thuật toán loại trừ ứng cử

Trong thuật toán này ta khởi tạo G và S bởi giả thuyết G_0 và S_0 là các giả thuyết tổng quát và chi tiết nhất:

$$G \leftarrow G_0 = \langle ?, ?, ?, ?, ?, ? \rangle$$

$$S \leftarrow S_0 = \langle \phi, \phi, \phi, \phi, \phi, \phi \rangle.$$

Khi mỗi mẫu đào tạo được xem xét, G và S được chi tiết hóa và tổng quát hóa để loại trừ khỏi $VS_{H,D}$ các giả thuyết không phù hợp với dữ liệu tương ứng. Sau khi tập mẫu D đã được xét hết thì $VS_{H,D}$ chỉ chứa các giả thuyết phù hợp với D trong H . Thuật toán được mô tả trong bảng 2.5.

Một ví dụ minh họa

Hình 2.7. biểu diễn thuật toán khi áp dụng hai mẫu đầu trong bảng 2.2. Tập biên khởi tạo bởi G_0 và S_0 . Khi xem xét mẫu thứ nhất (ở đây là mẫu dương), thuật toán kiểm tra biên S và thấy nó không phù hợp với mẫu này và đưa vào S_1 như trong hình, còn $G_1 = G_0$ vì G_0 vẫn phù hợp với mẫu này. Khi xét mẫu 2, cũng là mẫu dương, tương tự trước ta tổng quát hóa S_1 thành S_2 còn $G_2 = G_0$ không đổi. Ta thấy xử lý hai mẫu dương này tương tự như trong thuật toán Find S.

Bảng 2.5. Thuật toán loại trừ ứng cử

Bước 1. Khởi tạo G là tập giả thuyết tổng quát nhất trong H ,

Khởi tạo S là tập giả thuyết chi tiết nhất trong H ;

Bước 2.(Lặp)Với mỗi ví dụ đào tạo $d=\{x, c(x)\}$, thực hiện:

2.1. Nếu d là ví dụ dương tính ($c(x)=1$):

2.1.2. Lấy khỏi G các giả thuyết không phù hợp với d ;

2.1.2. Với mỗi s trong S không phù hợp với d :

Lấy s khỏi S ;

Thêm vào S các tổng quát hoá chi tiết nhất h của s mà h phù hợp với d và có một phần tử trong G tổng quát hơn h ;

Lấy khỏi S các giả thuyết tổng quát hơn các giả thuyết khác trong S

2.2. Nếu d là ví dụ âm tính ($c(x)=0$)

2.2.1. Lấy khỏi S các giả thuyết không phù hợp với d

2.2.2. Với mỗi g trong G không phù hợp với d :

Lấy g khỏi G ;

Thêm vào G các chi tiết hoá nhỏ nhất h của g mà h phù hợp với d và có một phần tử trong S chi tiết hơn h ;

Lấy khỏi G các giả thuyết ít tổng quát hơn các giả thuyết trong G

Mẫu huấn luyện:

1: <nắng, ám, trung bình, mạnh, ám, không đổi> $c=1$
 2: <nắng, ám, cao, mạnh, ám, không đổi> $c=1$

$S_0:$ $\boxed{\langle \phi, \phi, \phi, \phi, \phi, \phi \rangle}$

S_1 $\boxed{\langle \text{nắng, ám, trung bình, mạnh, ám, không đổi} \rangle}$

S_2 $\boxed{\langle \text{nắng, ám, ?, mạnh, ám, không đổi} \rangle}$

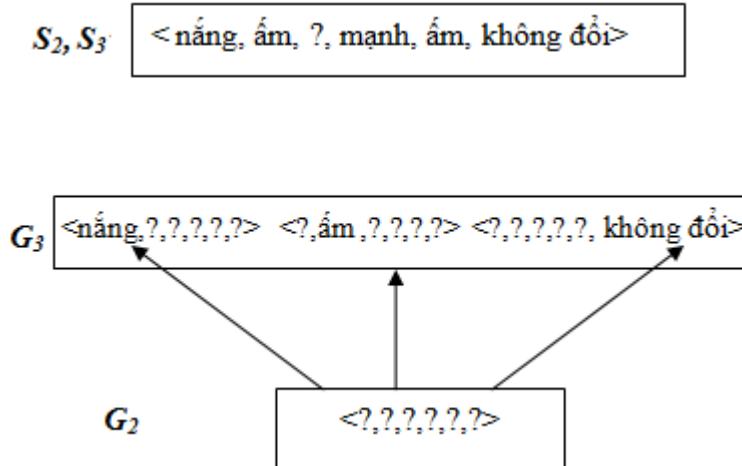
$G_0, G_1, G_2:$ $\boxed{\langle ?, ?, ?, ?, ?, ? \rangle}$

Hình 2.7. Kết quả xử lý các ví dụ 1 và 2

Trong hai bước này, các mẫu dương có thể làm cho biên chi tiết của không gian tường thuật tăng tính tổng quát hơn. Còn ví dụ âm sẽ tăng tính chi tiết của biên tổng quát như khi xét mẫu tiếp theo.

Bây giờ ta xét mẫu 3: \langle mưa, lạnh, cao, mạnh, ám, đổi $\rangle c = 0$ như minh họa trong hình 2.8. Trong trường hợp này S_3 vẫn như S_2 còn G_3 chứa ba giả thuyết chi tiết hóa nhỏ nhất của G_2 .

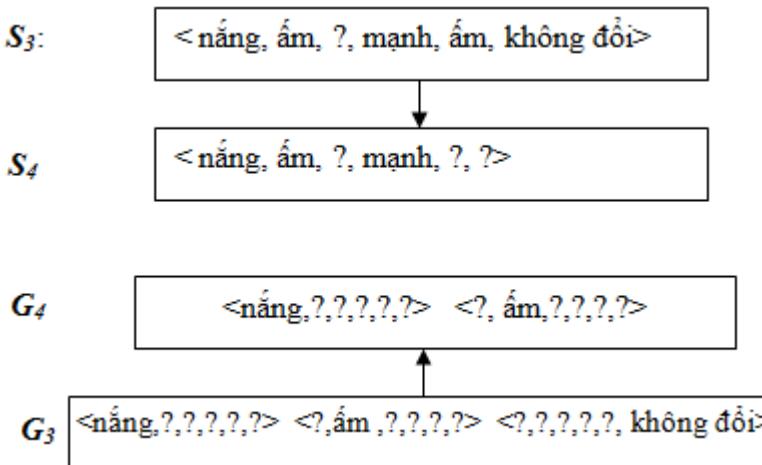
Mẫu: \langle mưa, lạnh, cao, mạnh, ám, đổi $\rangle c = 0$



Hình 2.8. Kết quả xử lý mẫu 3

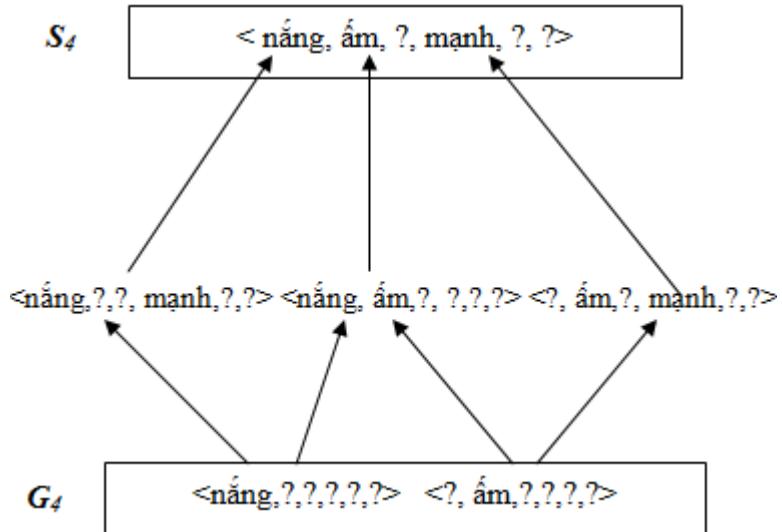
Tiếp theo, ta xét mẫu thứ tư: \langle nắng, ám, cao, mạnh, lạnh, đổi $\rangle c = 1$, kết quả được minh họa trong hình 2.9. Trong biên tổng quát G_3 , mẫu này không thỏa mãn giả thuyết $\langle ?, ?, ?, ?, ?, không đổi \rangle$ nên bị loại và G_4 chỉ còn hai giả thuyết. Giả thuyết trong S_3 không phù hợp với mẫu mới nên được thay bằng giả thuyết tổng quát hơn nhỏ nhất và chi tiết hơn giả thuyết trong G_4 , ta có S_4 .

Mẫu: \langle mưa, ám, cao, mạnh, ám, đổi $\rangle c = 1$



Hình 2.9. Kết quả xử lý mẫu 4

Cuối cùng, biểu diễn compact của $V_{H,D}$ được mô tả trong hình 2.10.



Hình 2.10. Biểu diễn compact của $VS_{H,D}$

Nhận xét. Trong các thuật toán trên, ta ngầm hiểu các giá trị thuộc tính được lấy trên tập hữu hạn và các mẫu quan sát được không có nhiều (đều là mẫu đúng). Khi các thuộc tính nhận giá trị thực, ta có thể rời rạc hóa miền giá trị của mỗi thuộc tính bởi các khoảng. Tuy nhiên với bài toán nội suy thì việc giải đúng hệ phương trình 2.2 thường rất khó thực hiện mà thường phải giải gần đúng như bài toán hồi quy. Bây giờ ta chuyển sang việc giải các bài toán này

2.4. HỌC HÀM NỘI SUY VÀ HỒI QUY

2.4.1. Phương pháp trực tiếp tìm hàm nội suy

Xét bài toán nội suy đã nêu trong mục 2.1.3: dựa trên tập quan sát được $D = \{(x^j, y^j)\}_{j=1}^N$ trong $R^n \times R$, trong đó là y^j giá trị quan sát của hàm chưa biết f tại điểm x^j : $y^j = f(x^j); j = i, \dots, N$. Cần tìm hàm g có dạng đã chọn thỏa mãn hệ phương trình (2.2):

$$g(x^j) = y^j \quad \forall j = i, \dots, N$$

để nội suy hàm f tại các điểm x mới.

Việc xây dựng hàm g thường tuân theo lược đồ tổng quát như sau: chọn trước một hàm dạng tổng quát phụ thuộc N tham số $\varphi(x, c_1, \dots, c_N)$ và hàm nội suy g có dạng:

$$g(x) = \varphi(x, c_1, \dots, c_N), \tag{2.8a}$$

trong đó các tham số c_1, \dots, c_N được xác định dựa trên hệ phương trình (2.2).

Khi chọn trước hệ N hàm độc lập tuyến tính $\{\varphi_k\}_{k=1}^N$ và hàm φ là tổ hợp tuyến tính của chúng:

$$\varphi(\mathbf{x}) = \sum_{k=1}^N c_k \varphi_k(\mathbf{x}) \quad (2.8b)$$

thì hàm nội suy g có thể xác định nhờ giải hệ phương trình tuyến tính để tìm các hệ số c_k :

$$\sum_{k=1}^N c_k \varphi_k(x^j) = y^j \quad \forall j = 1, \dots, N. \quad (2.8c)$$

Với $n = 1$ và $\varphi_k(x) = x^{k-1}$, ta tìm được hàm nội suy Lagrange quen biết. Khi N lớn, việc giải hệ (2.8c) có thể có sai số lớn do tích lũy sai số tính toán, ta có thể tìm gần đúng các hệ số c_k nhờ cực tiểu tổng bình phương sai số' như trong mục sau.

2.4.2. Tìm hàm hồi quy

Phương pháp thông dụng giải bài toán hồi quy là tìm hàm hồi quy g có biểu diễn tham số tương tự như trong biểu thức (2.8a):

$$g(\mathbf{x}) = \varphi(\mathbf{x}, c_1, \dots, c_M), \quad (2.9a)$$

trong đó M có thể nhỏ hơn N .

Việc xác định các hệ số c_k được thực hiện nhờ tìm cực tiểu tổng bình phương sai số' (sum of squared errors viết tắt là SSE) E :

$$E(c) = \sum_{k=1}^N [\varphi(x^k) - y^k]^2. \quad (2.9b)$$

Với bài toán nội suy thì cực tiểu của E tìm được sẽ bằng không. Trường hợp tổng quát, vectơ c có thể tìm bằng thuật toán gradient¹ (hoặc một biến thể của nó). Một biến thể đơn giản của thuật toán gradient được đặc tả trong bảng 2.6.

Bảng 2.5. Thuật toán gradient tìm cực tiểu hàm E

Bước 1. Khởi tạo và vectơ $c = c^0 \in R^M$ tùy ý;

Bước 2. Thực hiện lặp:

2.1. Tính $E'(c^0) = \left[\frac{\partial E}{\partial c_1}, \dots, \frac{\partial E}{\partial c_M} \right]^T$; // chỉ số trên T là vectơ chuyển vị

2.2. $\alpha_1 = \alpha$ // $\alpha \in (0,1]$ cho trước

2.3. $c^1 = c^0 - \alpha_1 E'(c^0)$;

2.3. Nếu $E(c^1) < E(c^0)$ thì $c^0 \leftarrow c^1$ và quay lại 2.1;

2.4. ngược lại: $\alpha_1 \leftarrow \frac{\alpha_1}{2}$ và trở lại 2.3;

¹ Nếu cần, người đọc có thể tham khảo tài liệu [13] để hiểu thêm phương pháp này

Thuật toán dừng khi $\|E'(\cdot)\|$ đủ bé, trong đó $\|u\| = \sqrt{\sum_{i=1}^M u_i^2}$ là chuẩn Euclide của vecto u trong R^M .

Trường hợp tuyến tính

Khi φ có dạng tuyến tính của các hàm cơ sở $\{\varphi_k(x)\}_{k=1}^N$:

$$\varphi(x) = \sum_{k=1}^N c_k \varphi_k(x) \quad (2.9b)$$

thì ta dễ dàng xác định được hệ phương trình để tìm điểm dừng của E: cho bởi $\frac{\partial E}{\partial c_i} = 0$ với mọi $i=1,\dots,M$

$$\frac{\partial E}{\partial c_i} = 2 \sum_{k=1}^N \left[\sum_{j=i}^M c_j \varphi_j(x^k) - y^k \right] \varphi_i(x^k) = 0 \quad (2.9c)$$

$$\Leftrightarrow \sum_{k=1}^N \left[\sum_{j=i}^M c_j \varphi_j(x^k) - y^k \right] \varphi_i(x^k) = 0 \quad \forall i \quad (2.9d)$$

$$\Leftrightarrow \sum_{j=1}^M c_j \left[\sum_{k=1}^N \varphi_j(x^k) \varphi_i(x^k) \right] = \sum_{k=1}^N y^k \varphi_i(x^k) \quad \forall i. \quad (2.9e)$$

Biểu thức (2.9c) cũng cho ta xác định được vecto gradient $E'(c)$. Nếu M không lớn, ta có thể giải hệ phương trình tuyến tính (2.9e) để xác định hàm hồi quy.

2.5. MỘT SỐ VẤN ĐỀ LIÊN QUAN

2.5.1. Khuynh hướng quy nạp

Như đã biết trong mục 2.3, nếu không gian giả thuyết H là tất cả các hàm có thể có từ X lên Y thì nói chung có nhiều lời giải phù hợp với tập dữ liệu huấn luyện D đã cho. Như vậy, bài toán học thiết lập không đúng đắn. Hơn nữa việc tìm kiếm trên không gian này thường không khả thi trong thực tế. Vì vậy người ta thường đưa ra các giả thuyết cho tập dữ liệu để thuật toán học duy nhất nghiệm và hạn chế không gian tìm kiếm. Tập các giả thiết này gọi là khuynh hướng quy nạp(bias) của thuật toán học.

Chẳng hạn, trong thuật toán Find-S ta giả thiết không gian H chỉ chứa các liên kết ràng buộc và các ví dụ âm luôn đúng khi giả thuyết thỏa mãn các ví dụ dương. Đối với trường hợp cho trong bảng 2.6, giả thuyết này sẽ không thỏa mãn.

Bảng 2.6. Một tập mẫu mới về những ngày chơi hoặc không của A

Ví dụ	Bầu trời	nhiệt độ	độ ẩm	gió	nước	dự báo	thích chơi
1	nắng	ấm	trung bình	mạnh	mát	đổi	có
2	mây	ấm	trung bình	mạnh	mát	đổi	có
3	mưa	ấm	trung bình	mạnh	mát	đổi	không

Thực vậy, khi đó giả thuyết chi tiết nhất phù hợp với hai mẫu đầu của tập là:

$$S_2 = <?, \text{ấm}, \text{trung bình}, \text{mạnh}, \text{mát}, \text{đổi}>$$

Tuy nhiên, giả thuyết này không phù hợp với mẫu thứ ba. Sở dĩ như vậy vì khuynh hướng học: H chỉ là các liên kết ràng buộc và các ví dụ âm luôn đúng khi giả thuyết thỏa mãn các ví dụ dương không thỏa mãn dẫn đến không có lời giải.

2.5.2. Học gần đúng theo xác suất

Ta trở lại với bài toán học khái niệm theo cách đánh giá khả năng học gần đúng theo xác suất (ký hiệu là PAC: Probably Approximately Correct) của bộ phân lớp đối với một lớp khái niệm. Cho tập đối tượng X (chẳng hạn, tập người) được mô tả bởi các đặc trưng (chẳng hạn, các chỉ số xét nghiệm sinh hóa, và số đo vật lý) và một lớp khái niệm cần học C (chẳng hạn, các loại bệnh lý có thể gặp). Mỗi khái niệm $c \in C$ ứng với một hàm giá trị boolean $c: X \rightarrow \{0,1\}$ sao cho nếu $x \in X$ là đúng với khái niệm thì $c(x) = 1$, ngược lại $c(x) = 0$ nếu x là âm tính.

Giả sử để học khái niệm, các mẫu trong X được lấy ngẫu nhiên với phân bố xác suất \mathcal{D} nào đó. Các mẫu huấn luyện được tạo sinh nhò lấy ngẫu nhiên các mẫu x cũng theo phân bố \mathcal{D} , và quan sát hàm đích $c(x)$ để có tập dữ liệu quan sát cho bộ học L . Bộ học này sẽ tìm khái niệm đích trên không gian giả thuyết H nào đó. Sau khi sử dụng tập dữ liệu quan sát tìm được, bộ học sẽ cho một giả thuyết h trong không gian giả thuyết H để dùng làm ước lượng hàm c . Ta sẽ đánh giá sự thành công của bộ học L thông qua khả năng thực hiện đoán nhận của h trên các mẫu mới được lấy ra một cách ngẫu nhiên trong X theo cùng phân bố \mathcal{D} này.

Ở đây ta sẽ chú ý tới khả năng thực hiện của các bộ học khác nhau khi sử dụng các không gian giả thuyết khác nhau và với các khái niệm đích lấy ra từ các lớp C khác nhau. Bởi vì ta cần bộ học phải học được đủ tốt bất cứ một khái niệm đích nào trong C mà không để ý đến phân bố các mẫu đào tạo, nên ta thường chú ý phân tích trong các trường hợp xấu nhất trên tập khái niệm đích có thể có trong C và mọi phân bố mẫu \mathcal{D} có thể gặp.

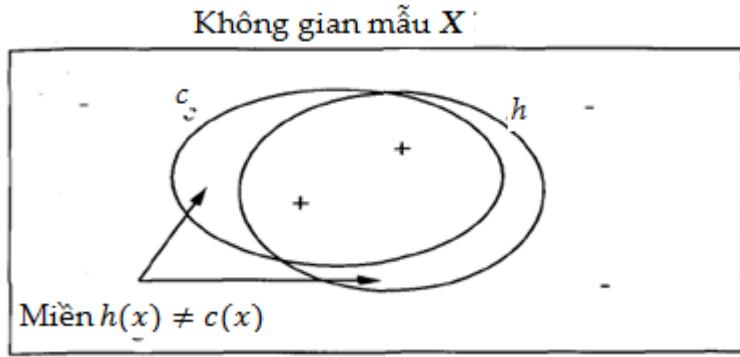
Lỗi của một giả thuyết

Ta hãy bắt đầu với khái niệm *lỗi đúng* (true error) của một giả thuyết h đối với khái niệm đích c và một phân bố mẫu \mathcal{D} .

Định nghĩa 2.6. (*Lỗi đúng*) Lỗi đúng của một giả thuyết h đối với khái niệm đích c và phân bố \mathcal{D} (được ký hiệu là $error_{\mathcal{D}}(h)$) là xác suất h phân lớp sai một mẫu lấy ngẫu nhiên theo phân bố \mathcal{D} :

$$error_{\mathcal{D}}(h) \equiv Pr_{x \in \mathcal{D}}[c(x) \neq h(x)], \quad (2.10)$$

trong đó $Pr_{x \in \mathcal{D}}$ là ký hiệu xác suất thu được trên phân bố mẫu \mathcal{D} . Hình 2.11 phác họa miền lỗi, trên đó hai hàm h và c nhận giá trị khác nhau, xác suất ở (2.11) là độ đo hai miền này theo phân bố mẫu \mathcal{D} .



Hình 2.11. Xác suất lỗi của giả thuyết

Khi \mathcal{D} là phân bố đều và X là các điểm trên hình chữ nhật thì lỗi đúng là tỷ lệ diện tích miền $c(x) \neq h(x)$ và diện tích toàn hình chữ nhật. Khi phân bố \mathcal{D} khác nhau thì nói chung xác suất lỗi khác nhau. Tuy nhiên, ta không ước lượng trực tiếp được lỗi này mà phải ước lượng qua lỗi đào tạo ($training error_{\mathcal{D}}(h)$) trên tập quan sát được D .

Định nghĩa 2.7. (*Lỗi đào tạo*) Lỗi đào tạo trên tập D là xác suất sai của giả thuyết trên tập huấn luyện này:

$$Training error_D(h) = Pr_{x \in D}(c(x) \neq h(x)). \quad (2.11)$$

Lỗi đào tạo dùng để ước lượng khoảng cho lỗi đúng, ta sẽ quay lại chủ đề này trong chương 5.

Khả năng học gần đúng theo xác suất

Theo giả thuyết quy nạp, số lượng mẫu quan sát được càng nhiều thì lỗi đúng càng nhỏ và ta luôn mong muốn tìm được giả thuyết h có lỗi đúng $error_{\mathcal{D}}(h)=0$. Tuy nhiên điều này là không thể trong thực tế, trừ khi tập mẫu quan sát được là toàn bộ tập X .

Người ta khắc phục khó khăn này bằng cách chỉ đòi hỏi khi tập mẫu đào tạo đủ lớn thì với xác suất lớn hơn $1 - \delta$ trong đó δ là hằng số đủ nhỏ tùy ý, bộ học tìm được giả thuyết h có lỗi đúng nhỏ hơn hằng số ϵ bé tùy ý cho trước. Khi đó ta nói bộ học L có khả năng học gần đúng xác suất (PAC) lớp khái niệm C . Mô hình PAC được Valiant đề xuất (1984), Mitchell phát triển thành định nghĩa sau.

Định nghĩa 2.8. (*Học PAC*) Xét một lớp các khái niệm C trên một tập X các mẫu có độ dài n và một bộ học L sử dụng không gian giả thuyết H . Ta nói C có thể học PAC bằng L nhờ dùng H nếu với mọi khái niệm $c \in C$, phân bố \mathcal{D} trong X và các hằng số ϵ , δ trong khoảng $(0, 1/2)$ thì với xác suất ít nhất là $(1 - \delta)$, bộ học L đưa ra một được giả thuyết $h \in H$ sao cho $error_{\mathcal{D}}(h) \leq \epsilon$, với thời gian là đa thức theo $1/\epsilon$, $1/\delta$, n và $size(c)$. (Ở đây $size(c)$ là độ dài đã được mã hoá của c trong C)

Trong định nghĩa trên, ta có hai yêu cầu với L . Thứ nhất: với xác suất cao tùy ý (lớn hơn $1 - \delta$), L phải đưa ra được một giả thuyết h với một xác suất lỗi bé tùy ý

(nhiều nhất là ε). Thứ hai, L phải làm việc rất hiệu quả, theo thời gian nó chỉ tăng theo đa thức với các tỉ số $1/\varepsilon$, $1/\delta$, n và $\text{size}(c)$ về độ phức tạp.

2.5.3. Chiều Vapnik-Chervonenkis

Chiều *Vapnik-Chervonenkis* (VC) dùng để đo độ phức tạp của không gian giả thuyết H biểu thị qua khả năng phân biệt mẫu của nó. Xét tập mẫu S trong X . Mỗi giả thuyết h trong H sẽ tách S thành hai phần: tập mẫu dương tính và tập mẫu âm tính. Tập S có $2^{|S|}$ tập con có thể là tập dương tính. Một bài toán đặt ra là không gian giả thuyết H đã cho có phân tách (shatter) được một tập con bất kỳ của S như là tập dương tính hay không? Trước khi định nghĩa chiều VC, ta cần định nghĩa sau về khả năng tách được của H đối với tập S

Định nghĩa 2.9. (Tính tách được) Một tập mẫu S được gọi là tách được bởi không gian giả thuyết H nếu và chỉ nếu với mọi tập con của S luôn tồn tại một giả thuyết h trong H sao cho tập con này dương tính và phần bù của nó trong S âm tính. Khi đó ta cũng nói H tách được S .

Không gian giả thuyết H được xem là lý tưởng nếu nó tách được toàn bộ không gian mẫu X . Khi H không thể tách X , nhưng lại có thể phân tách một tập con S của X có cỡ (size) lớn thì sao? Bằng trực giác, có thể nói, tập con của X có khả năng tách được tập con càng lớn thì H càng có ý nghĩa. Chiều VC của H biểu thị độ đo này

Định nghĩa 2.10. (Chiều Vapnik-Chervonenkis) Chiều VC của không gian giả thuyết H (kí hiệu $VC(H)$) trên không gian mẫu X là số cực đại các đối tượng trong X tách được bởi H . Nếu tập con hữu hạn tuỳ ý của X có thể tách được bởi H thì $VC(H) = \infty$

Dễ dàng kiểm tra được với mọi không gian giả thuyết hữu hạn ta đều có ước lượng:

$$VC(H) \leq \log_2 |H|. \quad (2.12)$$

Chứng minh điều này dành cho độc giả như bài tập

Ví dụ 1.

Giả sử không gian mẫu X là tập các số thực $X = \mathbb{R}$ (ví dụ, mô tả chiều cao của người) và H là tập các khoảng số thực dạng: $a < x < b$ với a và b là các hằng số thực. Khi đó $VC(H)$ là gì?

Để trả lời câu hỏi này, chúng ta cần tìm tập lớn nhất của X có thể tách được bởi H . Xét một tập con S gồm 2 mẫu khác nhau trong tập $\{1, 5, 7\}$, ta thấy 4 giả thuyết $\{x < 2\}, \{x < 4\}, \{x < 7\}, \{x < 1\}$ làm được việc này (tương ứng thể hiện 4 phân tách trên S : không có mẫu nào, một trong 2 mẫu và cả 2 mẫu). Bởi vì chúng ta đã tìm ra một tập có cỡ 2 tách được bởi H , do đó $VC(H)$ tối thiểu là 2. Vậy giờ ta cần trả lời câu hỏi: có tập nào cỡ bằng 3 tách được bởi H không? Giả sử tập $S = \{x_1, x_2, x_3\}$

chứa 3 mẫu, không mất tính tổng quát ta cho $x_0 < x_1 < x_2$. Rõ ràng tập này không tách được bởi H vì mọi khoảng chứa tập con $\{x_0, x_2\}$ đều chứa x_1 . Vậy thì $VC(H)=2$. Chú ý rằng H vô hạn nhưng $VC(H)$ là hữu hạn.

Ví dụ 2

Tiếp theo, ta xét tập X có các mẫu tương ứng với các điểm trong mặt phẳng hai chiều x,y (xem hình 2.12). H là tập tất cả quyết định tuyến tính trong mặt phẳng (chia mặt phẳng bằng đường thẳng). Ta có thể dễ dàng thấy bất cứ 2 điểm nào trong mặt phẳng đều có thể tách được bởi H . Vậy $VC(H) \geq 2$.

Với 3 điểm không thẳng hàng, ta có thể tìm được 2^3 mặt tuyến tính tách chúng. Dĩ nhiên 3 điểm thẳng hàng không thể phân tách được (lý do giống như 3 điểm trên trực số thực không thể phân tách được ở ví dụ trên). Vậy $VC(H)$ tối thiểu là 3. Định nghĩa về chiều VC chỉ ra rằng nếu chúng ta tìm ra bất kỳ tập mẫu nào kích thước d mà H tách được thì $VC(H) \geq d$. Để chứng minh $VC(H) < d$, chúng ta phải chỉ ra không có tập nào kích thước d có thể phân tách được. Trong ví dụ này, không có tập nào kích thước 4 tách được bởi H (bài tập 9) thì $VC(H)=3$ (hình 2.12). Tổng quát hơn, kích thước D của các mặt tuyến tính trong một không gian r chiều là $r+1$.



(a) Tập 3 điểm có thể được phân tách bởi quyết định tuyến tính.
(b) Tập 3 điểm không thể phân tách

Hình 2.12. Chiều VC của các đường thẳng trong mặt phẳng là 3

KẾT LUẬN

Học quy nạp từ tập mẫu quan sát được có vị trí quan trọng trong học máy. Trong đó, học khái niệm có thể thực hiện nhờ tìm kiếm trong một không gian giả thuyết lớn cho trước. Dựa trên cấu trúc thứ tự tổng quát đến chi tiết, thuật toán Find-S thực hiện tìm kiếm giả thuyết chi tiết nhất phù hợp với tập mẫu đào tạo, trong khi thuật toán loại trừ ứng cử xác định không gian tường thuật nhờ các biến chi tiết và tổng quát của nó.

Bài toán hồi quy có thể thực hiện nhờ dùng thuật toán gradient hoặc một biến thể của nó để tìm hàm hồi quy trong không gian giả thuyết biểu diễn bằng một hàm phụ thuộc tham số. Nói riêng khi hàm nội suy và hồi quy là tuyến tính đối với họ

hàm cơ sở đã cho thì chúng có thể tìm bằng giải trực tiếp hệ phương trình tuyến tính đối với các hệ số.

Bài toán học quy nạp thường không duy nhất nghiệm và phải tìm kiếm trên không giả thuyết lớn, dùng khuynh hướng quy nạp là một tiếp cận để khắc phục hiện tượng này.

Mô hình gần đúng xác suất xét các thuật toán học trong lớp khái niệm C với tập mẫu được lấy ngẫu nhiên với phân bố xác suất nào đó. Mô hình đòi hỏi với mỗi khái niệm c , bộ học có khả năng tìm được giả thuyết gần đúng cho khái niệm có lỗi đúng nhỏ hơn ϵ bé tùy ý với xác suất ít nhất là $(1 - \delta)$ bằng một thuật toán thời gian đa thức theo $1/\epsilon$, $1/\delta$, n và c (size) của X , c .

BÀI TẬP

- 1- Giải thích tại sao trong bài toán ở mục 2.2.3 lại có 96 mẫu và 973 giả thuyết khác nhau, số mẫu và giả thuyết sẽ là bao nhiêu nếu thêm thuộc tính *dòng nước* với các giá trị *nhỏ*, *trung bình* và *xiết*
- 2- Chứng minh rằng lời giả của thuật toán Find-S là duy nhất
- 3- Xây dựng thứ tự tổng quát-chi tiết và thuật toán tương tự Find-S cho bài toán học khái niệm với thuộc tính liên tục
- 4- Cho biết thứ tự lấy ví dụ trong thuật toán loại trừ ứng cử có ảnh hưởng tới kết quả không, tại sao?
- 5- Cho tập mẫu gồm các đối tượng có hai thuộc tính đặc trưng, đặc trưng thứ nhất nhận hai giá trị 0 hoặc 1, đặc trưng thứ hai nhận giá trị trong 5 chữ cái. Dữ liệu quan sát được gồm hai lớp cho bởi bảng sau:

$\omega_1: (0,A) (0,B) (1,B) (0,E) (0,D)$
$\omega_2: (1,A) (0,C) (1,C) (1,E) (1,D)$

- a) Tính số đối tượng khác nhau của bài toán
- b) Áp dụng thuật toán Find-S cho tập mẫu này và tính số giả thuyết khác nhau trong không gian giả thuyết
- c) Áp dụng thuật toán loại trừ ứng cử cho tập mẫu này
- 6- Cho hàm hai biến $z = f(x,y)$ với tập quan sát cho bởi bảng:

x	-2	-1	0	1	2	1	0	-1
y	-1	0	2	-1	1	-2	1	1
z	2	5	8	3	21	4	2	-4

- a) Tìm hàm hồi quy bậc hai: $g(x,y) = ax^2 + by^2 + cxy + dx + ey + h$ cho hàm z

- b) Tìm hàm nội suy của z dạng:

$$g(x,y) = ax^3 + by^3 + cx^2y + dxy^2 + exy + hx + ky + l$$

7- Phát triển lược đồ áp dụng các thuật toán Find-S và loại trừ ứng cử cho bài toán học nhiều lớp

8-Chứng minh công thức (2.12)

9-Chứng minh rằng một tập gồm 4 điểm tùy ý trong mặt phẳng không thể tách được bởi không gian giả thuyết H trong ví dụ 2 ở mục 2.5.3

Chương 3

HỌC BẰNG CÂY QUYẾT ĐỊNH

Chương này giới thiệu phương pháp học có giám sát nhờ cây quyết định, bắt đầu từ bài toán phân lớp các đối tượng có thuộc tính nhận giá trị rời rạc, sau đó phát triển xử lý các trường hợp có thuộc tính liên tục và một số vấn đề liên quan.

3.1. BIỂU DIỄN GIẢ THUYẾT BẰNG CÂY QUYẾT ĐỊNH

Chương trước, giới thiệu tiếp cận giải các bài toán *phân lớp/hồi quy hàm số* nhờ tìm kiếm trong không gian giả thuyết. Học bằng cây quyết định để xây dựng trực tiếp giả thuyết cần tìm là một phương pháp thông dụng trong ứng dụng.

Để đơn giản, ta bắt đầu từ các bài toán mà các thuộc tính của đối tượng nhận giá trị trong tập hữu hạn. Bài toán được biểu như sau: Có hàm đích $c: X \rightarrow Y$ trong đó $\mathbf{x} = \langle x_1, \dots, x_n \rangle$, x_i là giá trị trong miền giá trị của thuộc tính A_i tương ứng, tập đích Y là các tập hữu hạn. Từ tập mẫu quan sát được $D = \{ \mathbf{x}^k, y^k \}_{k=1}^M$ ta cần xác định một giả thuyết gần đúng h cho hàm c dưới dạng một tập luật để xác định hàm đích dựa trên giá trị thuộc tính dạng:

$$\{ \text{Nếu } \Lambda \mathbf{x}_i = v_i / i \in I \subset [1, \dots, n] \text{ thì } h(\mathbf{x}) = y \}. \quad (3.1)$$

Các giả thuyết này có thể biểu diễn bằng một cây quyết định, trong đó mỗi đường đi xuôi từ gốc đến lá cho ta một luật có biểu diễn dạng (3.1) theo các ràng buộc giá trị-thuộc tính trên đường đi này.

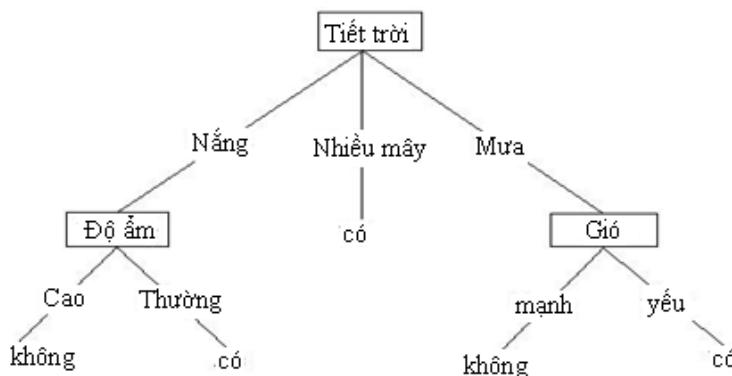
Nút gốc và mỗi nút trong trên cây sẽ định rõ một thuộc tính/điều kiện kiểm tra của mẫu và mỗi nhánh đi xuống từ mỗi nút tương ứng với một trong những giá trị/giá trị *chân lý* có thể của thuộc tính/điều kiện này. Nhãn của các mẫu phù hợp với đường đi này được gắn ở nút lá. Với mỗi mẫu mới, nhãn của nó được xác định bằng cách khởi đầu từ nút gốc của cây, kiểm tra thuộc tính/điều kiện ở nút này và sau đó di chuyển xuống nhánh tương ứng với giá trị/giá trị chân lý xác định ở lá.

Ta xét tập mẫu quan sát về thời tiết thích hợp để bạn A chơi cầu lông dựa trên 4 thuộc tính {tiết trời, nhiệt độ, độ ẩm, gió} cho trong bảng 3.1.

Bảng 3.1. Tập mẫu về thời tiết phù hợp để chơi cầu lông

Ngày	Tiết trời	Nhiệt độ	Độ ẩm	Gió	Chơi cầu lông
D1	Nắng	Nóng	Cao	Yếu	Không
D2	Nắng	Nóng	Cao	Mạnh	Không
D3	Nhiều mây	Nóng	Cao	Yếu	Có
D4	Mưa	Mát	Cao	Yếu	Có
D5	Mưa	lạnh	Thường	Yếu	Có
D6	Mưa	lạnh	Thường	Mạnh	Không
D7	Nhiều mây	lạnh	Thường	Mạnh	Có
D8	Nắng	Mát	Cao	Yếu	Không
D9	Nắng	lạnh	Thường	Yếu	Có
D10	Mưa	Mát	Thường	Yếu	Có
D11	Nắng	Mát	Thường	Mạnh	Có
D12	Nhiều mây	Mát	Cao	Mạnh	Có
D13	Nhiều mây	Nóng	Thường	Yếu	Có
D14	Mưa	Mát	Cao	Mạnh	Không

Hình 3.1 biểu thị một cây quyết định cho bài toán học khái niệm *thời tiết phù hợp để bạn A chơi cầu lông* dựa trên dữ liệu đào tạo cho trong bảng 3.1 và được xây dựng nhờ thuật toán ID3 trong mục 3.2.

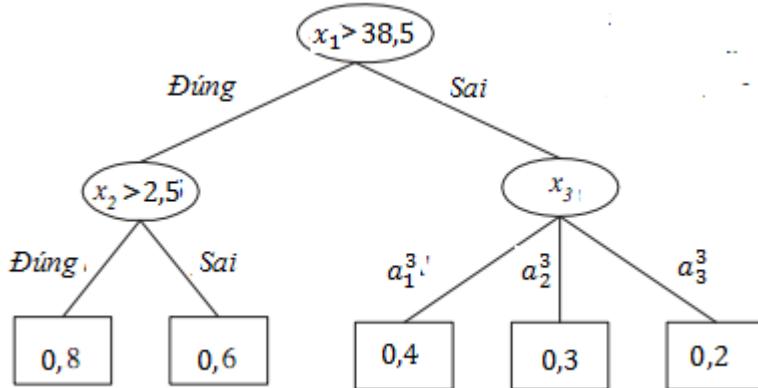


Hình 3.1. Cây quyết định học được từ mẫu cho trong bảng 3.1

Trong ví dụ này, các nút biểu thị thuộc tính kiểm tra, các nhánh từ mỗi nút ứng với giá trị kiểm tra của mẫu. Cây này tương ứng với tập gồm 5 luật:

$\{ < \text{Nếu Tiết trời = nắng} \wedge \text{độ ẩm = cao} \text{ thì } h(x) = \text{không};$
 $< \text{Nếu tiết trời = nắng} \wedge \text{độ ẩm = thường} \text{ thì } h(x) = \text{có};$
 $< \text{Nếu tiết trời = nhiều mây} \text{ thì } h(x) = \text{có};$
 $< \text{Nếu tiết trời = mưa} \wedge \text{gió = mạnh} \text{ thì } h(x) = \text{Không};$
 $< \text{Nếu tiết trời = mưa} \wedge \text{gió = yếu} \text{ thì } h(x) = \text{có} \}$

Một cây hồi quy của một hàm ba biến trong đó có hai biến thực, biến x_3 nhận một trong tập $\{a_1^3, a_2^3, a_3^3\}$ được biểu thị trong hình 3.2.



Hình 3.2. Một cây hồi quy của hàm 3 biến với 2 biến thực

Tập luật tương ứng của cây này là:

$\{ < \text{Nếu } (x_1 > 38,5) \wedge (x_2 > 2,5) \text{ thì } h(x) = 0,8>;$
 $< \text{Nếu } (x_1 > 38,5) \wedge (x_2 \leq 2,5) \text{ thì } h(x) = 0,6>;$
 $Nếu (x_1 \leq 38,5) \wedge (x_3 = a_1^3) \text{ thì } h(x) = 0,4>;$
 $Nếu (x_1 \leq 38,5) \wedge (x_3 = a_2^3) \text{ thì } h(x) = 0,3;$
 $Nếu (x_1 \leq 38,5) \wedge (x_3 = a_3^3) \text{ thì } h(x) = 0,2\}$

Quá trình xây dựng cây quyết định dựa trên tập mẫu đào tạo sẽ được gọi là học cây quyết định. Mặc dù phương pháp học bằng cây quyết có miền ứng dụng khá rộng, nhưng thích hợp nhất với các bài toán thuộc tính nhận giá trị rời rạc, dữ liệu quan sát được có thể có lỗi hoặc mất/thiếu giá trị thuộc tính. Chẳng hạn, đối tượng dữ liệu thường có nhiều hoặc bị thiếu giá trị của một vài thuộc tính trong các bài toán:

- Chẩn đoán và điều trị bệnh: dựa trên theo dõi triệu chứng lâm sàng và các kết quả xét nghiệm để chẩn đoán bệnh và đề xuất phương án điều trị.
- Phân tích rủi ro ngân hàng: dựa vào các đặc điểm khách hàng để đoán nhận khách hàng có khả năng thanh toán hay không, trên cơ sở đó quyết định cho vay hoặc không.

3.2. CÁC THUẬT TOÁN HỌC

3.2.1. Thuật toán học ID3

3.2.1.1. Các khái niệm

Thuật toán ID3 (Quinlan 1986) học cây quyết định cho bài toán học khái niệm bằng cách xây dựng bắt đầu từ gốc và phát triển dần đến các nút lá. Mỗi nút gốc hoặc nút trong biểu thị một thuộc tính kiểm tra, các cạnh biểu thị giá trị kiểm tra của thuộc tính tương ứng. Thuộc tính tốt nhất của tập dữ liệu đào tạo được chọn làm nút gốc theo tiêu chuẩn cực đại lượng *thu hoạch thông tin* (Information gain). Để định nghĩa thu hoạch thông tin, ta cần đến khái niệm Entropy của một tập.

Entropy

Cho một tập S , gồm các mẫu nhận giá trị đích dạng boolean xác định bởi hàm c , ta dùng các ký hiệu:

$$\begin{aligned} S_+ &= \{s \in S : c(s) = 1\}, \\ S_- &= \{s \in S : c(s) = 0\}; \end{aligned} \quad (3.2a)$$

$$\begin{aligned} p_+ &= |S_+|/|S|, \\ p_- &= |S_-|/|S| \end{aligned} \quad (3.2b)$$

Khi đó Entropy của S là đại lượng xác định bởi:

$$\text{Entropy}(S) = -p_+ \log_2 p_+ - p_- \log_2 p_- \quad (3.3a)$$

Ví dụ. Để minh họa, ta giả sử có một người có chơi cầu lông hay không dựa vào đặc điểm thời tiết: Tiết trời, Nhiệt độ, Độ ẩm, Gió. Tập mẫu quan sát được S cho trong bảng 3.1 gồm 14 mẫu với 9 mẫu dương tính và 5 mẫu âm tính (chúng ta dùng ký hiệu [9+,5-]), thì entropy của S là:

$$\text{Entropy}([9+,5-]) = -(9/14)\log_2(9/14) - (5/14)\log_2(5/14) = 0.940$$

Lưu ý rằng $\text{Entropy}(S)$ lớn nhất khi các mẫu dương tính và âm tính của S có số lượng bằng nhau và $\text{Entropy}(S) = 0$ nếu tất cả các phần tử của S đều thuộc cùng một lớp.

Tổng quát hơn, trong trường hợp có k lớp và ký hiệu p_k là tỷ lệ mẫu có nhãn c_k trong S thì khi đó $\text{Entropy}(S)$ ứng với tập này là:

$$\text{Entropy}(S) = \sum_{i=1}^k -p_i \log_2 p_i. \quad (3.3b)$$

Thu hoạch thông tin

Thu hoạch thông tin của một thuộc tính A ứng với một tập mẫu S được ký hiệu là $Gain(S,A)$, đo sự giảm kỳ vọng entropy theo thuộc tính này, được xác định như sau:

$$Gain(S,A) \equiv Entropy(S) - \sum_{v \in Values(A)} \frac{|S_v|}{|S|} Entropy(S_v), \quad (3.4)$$

trong đó, $Values(A)$ là một tập các giá trị mà thuộc tính A có thể nhận, và S_v là tập hợp con của S nhận giá trị v ở thuộc tính này: $S_v = \{s \in S \mid A(s)=v\}$. Số hạng đầu trong vế phải của (3.4) là entropy của toàn bộ tập S còn số hạng thứ hai là giá trị kỳ vọng của entropy sau khi tách S ra bằng thuộc tính A . Chính vì vậy $Gain(S,A)$ là chiết giảm entropy mong đợi dựa trên các giá trị biết được của thuộc tính A .

Trở lại tập mẫu về các ngày chơi cầu lông phù hợp với thời tiết cho trong bảng 3.1, thuộc tính gió có thể nhận giá trị yếu hoặc mạnh. S có phân bố [9+,5-], trong đó tập mẫu giá trị thuộc tính *Tiết trời* là *Nắng* có phân bố [2+,3-] và là mưa có phân bố [3+,2-], các mẫu còn lại có thuộc tính này là nhiều mây với phân bố [4+,0-]. Thu hoạch thông tin thu được của S theo thuộc tính *Tiết trời* được tính như sau:

$$Values(\text{tiết trời}) = \text{Nắng}, \text{Mưa}, \text{N_mây}$$

$$S \leftrightarrow [9+,5-];$$

$$S_{\text{nắng}} \leftrightarrow [2+,3-], Entropy(S_{\text{nắng}}) = -2/5. \log(2/5) - 3/5.\log(3/5)=0,971;$$

$$S_{\text{mưa}} \leftrightarrow [3+,2-], Entropy(S_{\text{mưa}}) = -3/5.\log(3/5)- 2/5. \log(2/5)=0,971;$$

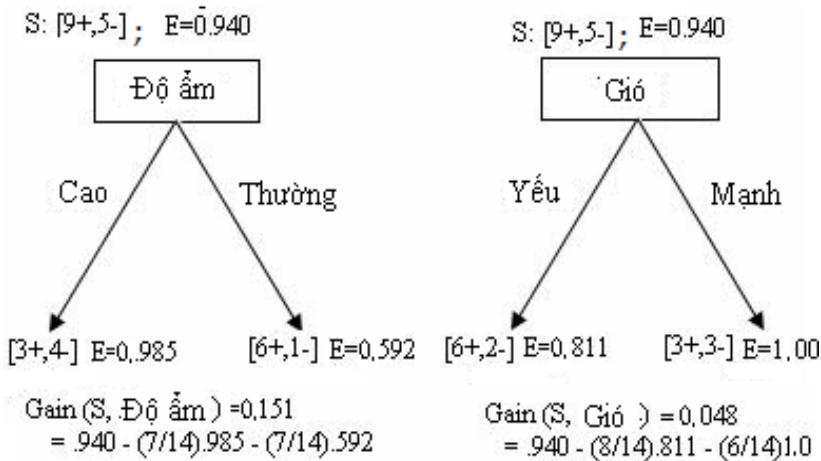
$$S_{\text{N_mây}} \leftrightarrow [4+,0-], Entropy(S_{\text{N-mây}}) = -4/4. \log(4/4)- 0.\log(0/4)=0;$$

$$Gain(S, \text{Tiết trời}) = Entropy(S) - \sum_{v \in Values(\text{Gio})} \frac{|S_v|}{|S|} Entropy(S_v)$$

$$= Entropy(S) - (5/14)Entropy(S_{\text{nắng}}) - (5/14)Entropy(S_{\text{Mưa}}) - (4/14)S_{\text{N_mây}}$$

$$= 0.94 - (5/14).0,971 - (4/14). 0,00 - (5/14).0,971 = 0.246.$$

Tương tự, lượng thu hoạch thông tin của S theo các thuộc tính độ ẩm và gió được tính và biểu thị tóm tắt trong hình 3.3, trong đó E để chỉ entropy.



Hình 3.3. Thu hoạch thông tin của S theo độ ẩm và gió

Bây giờ ta trở lại mô tả thuật toán với qui ước: *tập mẫu* là tập quan sát được đang xét, *tập nhãn* các nhãn lớp cần dự đoán cho mẫu, còn *tập thuộc tính* là các thuộc tính không phải thuộc tính đích của mẫu. Để đơn giản, ta chỉ xét bài toán học khái niệm, độc giả có thể mở rộng thuật toán áp dụng cho bài toán học nhiều lớp.

3.2.1.2. Mô tả thuật toán

Thuật toán ID3 thực hiện như sau. Trước hết, mở một nút gốc cho cây, nếu nhãn của *tập mẫu* có giá trị như nhau thì nút này được gán nhãn chung này, còn nếu *tập thuộc tính* là rỗng thì nút được gán nhãn theo đa số và thuật toán kết thúc. Ngược lại, thuộc tính tốt nhất được chọn làm nút gốc của cây và tạo ra các nhánh từ nút gốc tương ứng với các giá trị của thuộc tính được chọn.

Sau đó, tạo ra một nút con cho mỗi nhánh ứng với *tập mẫu huấn luyện* sẽ được sắp xếp phù hợp với các nút con, tức là, xếp các mẫu vào nhánh tương ứng với giá trị của mẫu theo thuộc tính này và thuộc tính này cũng bị loại khỏi danh sách *tập thuộc tính*.

Toàn bộ quá trình trên được lặp lại cho các *tập mẫu* mới ở các nút con để phát triển cây cho tới khi dữ liệu thuần nhất hoặc các thuộc tính đã kiểm tra hết thì nút trở thành nút lá và gán nhãn tương ứng hoặc theo đa số. Kết quả cho một cây quyết định phân loại đúng (hoặc gần đúng nếu mẫu có nhiều) các *tập mẫu* đào tạo. Thuật toán ID3 được mô tả trong bảng 3.2.

Ví dụ

Ta trở lại với bài toán dự báo một người chơi cầu lông theo đặc điểm thời tiết dựa trên *tập mẫu D* có phân bố [9+,5-] được cho trong bảng 3.1.

Bảng 3.2. Thuật toán ID3 (Tập mẫu, tập nhãn, tập thuộc tính)

Bước 0. Khởi tạo: D, tập nhãn, tập thuộc tính;

Bước 1. Tạo nút gốc (Root) cho cây;

Bước 2. // Gán nhãn cho nút nếu dữ liệu thuần nhất hoặc tập thuộc tính là rỗng.

2.1. Nếu mọi mẫu đều dương tính thì nhãn nút gốc = \oplus ;

2.2. Nếu tất cả các mẫu là âm tính thì nhãn nút gốc = \ominus ;

2.3. Nếu tập các thuộc tính là rỗng trả về cây một nút gốc có nhãn = giá trị phổ biến nhất của thuộc tính đích trong tập các mẫu;

2.4 Các trường hợp khác sang bước 3;

Bước 3.

3.1. Xác định thuộc tính phân loại tập mẫu tốt nhất trong tập thuộc tính;

3.2. $A \leftarrow$ thuộc tính phân lớp tốt nhất;

3.4. Với mỗi giá trị có thể v_i của thuộc tính A, thực hiện:

3.4.1. Thêm một nhánh mới dưới nút gốc với mỗi điều kiện $A=v_i$;

3.4.2. Xác định $Examples_{v_i} = \{x \in \text{tập mẫu}: x \text{ có giá trị } v_i \text{ ở thuộc tính } A\}$;

3.4.3. Nếu $Examples_{v_i}$ là rỗng thì thêm dưới nhánh một nút lá có nhãn là nhãn phổ biến nhất của các mẫu trong *tập mẫu*;

3.4.4. Ngược lại, trở lại bước 1 với khởi tạo:

($D = Examples_{v_i}, \text{tập nhãn}, \text{tập thuộc tính} - \{A\}$)

Vì tập mẫu không thuần nhất nên ta cần trả lời câu hỏi “Nên kiểm tra thuộc tính nào đầu tiên?” để gán nhãn cho nút gốc. Thuật toán ID3 xác định lượng *Thu hoạch thông tin* của các thuộc tính rồi chọn thuộc tính có lượng *Thu hoạch thông tin* cao nhất. Lượng *Thu hoạch thông tin* cho bốn thuộc tính theo các công thức (3.3) và (3.4):

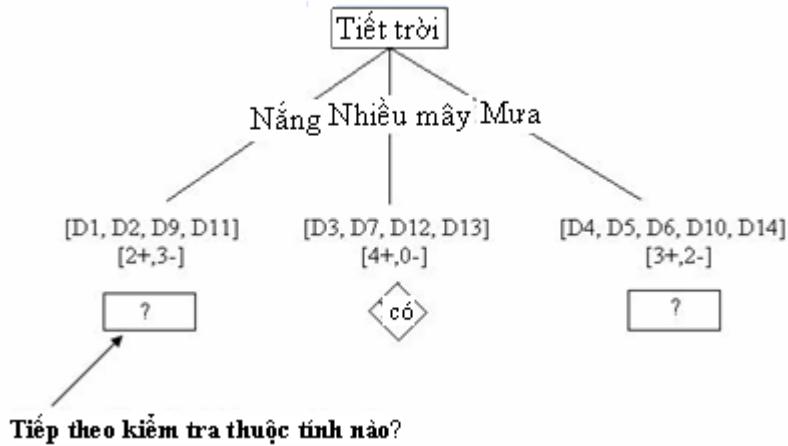
$$\text{Gain}(S, \text{Tiết trời}) = 0.246$$

$$\text{Gain}(S, \text{Độ ẩm}) = 0.151$$

$$\text{Gain}(S, \text{Gió}) = 0.048$$

$$\text{Gain}(S, \text{Nhiệt độ}) = 0.029$$

Thuộc tính *Tiết trời* là tốt nhất theo tiêu chuẩn thu hoạch thông tin cực đại. Vì vậy, thuộc tính *Tiết trời* được lựa chọn làm thuộc tính quyết định tại nút gốc, và các nhánh được tạo ra dưới nút này ứng với các giá trị tương ứng bao gồm: Nắng, nhiều mây, Mưa. Kết quả, một phần của cây quyết định được chỉ ra trong hình 3.4 cùng với các mẫu huấn luyện được phân loại cho mỗi nút con mới.



Hình 3.4. Phần cây quyết định với thuộc tính tiết trời tại gốc

Ta thấy mọi mẫu mà $Tiết\ trời = \text{nhiều\ mây}$ đều là mẫu khẳng định nên nút này của cây trở thành một nút lá với nhãn phân loại $Choi\ câu\ lồng = \text{Có}$. Ngược lại, các nút con tương ứng với $Tiết\ trời = \text{Nắng}$ và $Tiết\ trời = \text{Mưa}$ vẫn có entropy khác 0, và cây quyết định cần phát triển thêm nữa xuống dưới các nút này như trong hình 3.4. Nếu xét tập mẫu có thuộc tính tiết trời là nắng ta có $S_{\text{nắng}} = [D1, D2, D8, D9, D11]$ và cần tính độ thu hoạch thông tin của $S_{\text{nắng}}$ theo các thuộc tính Gió, Độ ẩm và Nhiệt độ để phát triển theo nhánh này.

Quá trình lựa chọn một thuộc tính mới và việc phân các mẫu huấn luyện sẽ được lặp lại với mỗi nút không phải là nút con cuối cùng và nó chỉ sử dụng các mẫu phù hợp với nút đang xét. Các thuộc tính đã được tổ chức ở phía trên của cây được loại ra, vì vậy bất kỳ một thuộc tính nào được đưa ra cũng có thể xuất hiện nhiều nhất là một lần dọc theo bất kỳ đường đi nào từ gốc tới lá trên cây. Quá trình này tiếp tục tới mỗi nút lá khi gặp phải một trong 2 trường hợp sau:

- 1) Mọi thuộc tính đều đã nằm trên đường đi này.
- 2) Các mẫu phù hợp với nút lá này đều thuộc cùng một lớp (tức là entropy của chúng = 0)

Cây quyết định được xây dựng bằng thuật toán ID3 trong ví dụ này đã được chỉ ra trong hình 3.1.

3.2.1.3. Nhận xét về ID3

ID3 cho ta một phương pháp hiệu quả để tìm một giả thuyết phù hợp với tập mẫu huấn luyện dưới dạng tập luật được biểu diễn bằng cây quyết định. Các luật phân lớp này tiện dùng trong các cơ sở tri thức của các hệ thống minh. Thuật toán có hiệu quả cả khi dữ liệu có nhiều nhò quyết định theo đa số khi có xung đột ở nút lá. Tuy nhiên vẫn còn những vấn đề sau cần được khắc phục.

- 1) ID3 tìm kiếm giả thuyết trên toàn bộ không gian các hàm có giá trị rời rạc nên không gian giả thuyết này luôn chứa hàm đích. Tuy nhiên, khi gặp thuộc tính có giá trị liên tục thì thuật toán không áp dụng được.
- 2) ID3 thực hiện xây dựng cây từ đơn giản đến phức tạp theo phương thức tìm kiếm leo đồi được định hướng nhờ tiêu chuẩn thuộc tính có *Thu hoạch thông tin cực đại* trong không gian giả thuyết. Tiêu chuẩn này chưa chắc đã tốt, Chẳng hạn, đối với dữ liệu nhân sự có thuộc tính ngày sinh khác nhau cho mỗi mẫu thì thuộc tính này thường có lượng *Thu hoạch thông tin* lớn nhất nên cho ta cây độ sâu bằng 1 và không có ý nghĩa. Có cách gì để khắc phục không?
- 3) ID3 không bao giờ quay lui trong quá trình tìm kiếm nên nó dễ mắc phải nhược điểm thường gặp của việc tìm kiếm leo đồi là tập trung vào các giải pháp tối ưu địa phương mà không phải là các giải pháp tối ưu toàn cục.
- 4) ID3 chỉ duy trì một giả thuyết đơn hiện thời khi tìm kiếm trong không gian của các cây quyết định nên nó đánh mất các khả năng mô tả rõ ràng tất cả các giả thuyết phù hợp. Chẳng hạn, nó không có khả năng xác định xem có bao nhiêu cây quyết định có thể lựa chọn phù hợp với dữ liệu huấn luyện hay đưa ra những vấn đề mới đòi hỏi phải giải quyết tối ưu giữa các giả thuyết đang cạnh tranh này.
- 5) Phương pháp dựa trên thống kê mà ID3 sử dụng để phát triển giả thuyết hiện thời của nó có ưu điểm là dùng thông tin của tất cả các mẫu, nhờ đó mà ít gặp lỗi hơn là các thuật toán học trên từng mẫu riêng lẻ. Vì vậy ID3 có thể được mở rộng một cách dễ dàng để dùng được cho dữ liệu huấn luyện có nhiều bằng cách chấp nhận các giả thuyết không phù hợp lầm với dữ liệu huấn luyện. Tuy nhiên thuật toán chưa cho cách xử lý dữ liệu bị mất.
- 6) Chiến lược tìm kiếm của thuật toán ID3 có hai đặc điểm chính: Ưa các cây ngắn hơn là các cây dài, ưu tiên hơn cho các cây quyết định nhỏ gọn hơn so với các quyết định phức tạp; Ưa những cây mà các thuộc tính có lượng *Thu hoạch thông tin* cao nhất nằm ở gần nút gốc nhất. Thuật toán tìm kiếm trên toàn bộ không gian giả thuyết nên cây tìm được có thể rất lớn, khó áp dụng khi tập dữ liệu đào tạo lớn và có nhiều thuộc tính. Một câu hỏi đặt ra là: nếu cây quá lớn thì ta có thể “tia” cho nhỏ và làm đơn giản đi ra sao?
- 7) Khi dữ liệu chứa nhiễu thì có thể xảy ra hiện tượng *phù hợp trội* (overfitting), tức là cây quyết định tìm được phù hợp tốt trên dữ liệu đào tạo nhưng có thể đoán nhận tồi hơn cây khác đối với mẫu mới. Thuật toán ID3 xử lý theo phương pháp thống kê nhưng ta chưa cho biết cách tránh hiện tượng phù hợp .

3.2.2. Thuật toán C4.5

Thuật toán này do Quinlan (1993) đề xuất để khắc phục các nhược điểm của chính của ID3. Thuật toán này thực hiện theo lược đồ của ID3 nhưng có các cải tiến sau:

- 1) Ngoài việc áp dụng tiêu chuẩn *Thu hoạch thông tin cực đại*, C4.5 còn đề xuất sử dụng tiêu chuẩn *Tỷ lệ thu hoạch thông tin cực đại (Gainratio)* để dùng cho các trường hợp mà tiêu chuẩn trước áp dụng không tốt (nhược điểm thứ nhất của ID3).
- 2) Áp dụng kỹ thuật chặn sớm sự phát triển của cây dựa trên thống kê để tránh phù hợp trội và cây không quá lớn.
- 3) Đề xuất giải pháp xử lý trường hợp mẫu có thuộc tính thiếu giá trị
- 4) Đề xuất phương pháp áp dụng cho thuộc tính nhận giá trị liên tục.

Ba giải pháp đầu sẽ được bàn đến trong mục sau, ở đây sẽ giới thiệu phương pháp xử lý thuộc tính có giá trị liên tục của C4.5.

Giả sử dữ liệu có thuộc tính A có giá trị liên tục, trong tập mẫu quan sát được, các mẫu nhận các giá trị khi được xếp thứ tự tăng dần là a_1, a_2, \dots, a_m . Thuật toán C4.5 xử lý trường hợp này như sau. Đối với mỗi giá trị a_i , chia tập mẫu thành hai lớp để tính thu hoạch thông tin: lớp thứ nhất gồm tập mẫu có giá trị thuộc tính A nhỏ hơn hoặc bằng a_i và lớp thứ hai là các mẫu còn lại. *Thu hoạch thông tin/ Tỷ lệ thu hoạch thông tin* của thuộc tính sẽ là cực đại của các đại lượng tính được theo các cách chia và a_k tương ứng với nó sẽ được chọn làm điểm chia rẽ nhánh cho thuộc tính này. Nếu thuộc tính liên tục này được chọn làm thuộc tính tốt nhất (theo tiêu chuẩn đang xét) thì tại nút tương ứng có thể gán nhãn là điều kiện kiểm tra ($x > a_k$) và hai nhánh tương ứng với giá trị logic như minh họa trong hình 3.2.

Ví dụ

Để minh họa, ta trở lại trường hợp các mẫu trong bảng 3.1 nhưng thuộc tính *Độ ẩm* nhận giá trị liên tục như được cho trong bảng 3.3.

Trong ví dụ này, sẽ dùng tiêu chuẩn *Thu hoạch thông tin cực đại*, việc dùng tiêu chuẩn *Tỷ lệ thu hoạch thông tin cực đại* được tính tương tự. Ta sẽ tính lượng *Thu hoạch thông tin* cho thuộc tính này, tức là trả lời câu hỏi:

$$Gain(D, Đ m) = ?$$

Trước hết, ta sắp các giá trị thuộc tính này của tập mẫu theo thứ tự tăng dần như sau:

$$\{65, 70, 70, 70, 75, 78, 82, 82, 82, 86, 90, 90, 95, 96\}$$

Sau đó loại bỏ các giá trị lặp ta có tập giá trị:

$$\{65, 70, 75, 78, 82, 86, 90, 95, 96\}$$

Bảng 3.3. Tập mẫu ở bảng 3.1 với giá trị thuộc tính Độ ẩm liên tục

Ngày	Tiết trời	Nhiệt độ	Độ ẩm	Gió	Chơi cầu lông
D1	Nắng	Nóng	86	Yếu	Không
D2	Nắng	Nóng	90	Mạnh	Không
D3	Nhiều mây	Nóng	78	Yếu	Có
D4	Mưa	Mát	96	Yếu	Có
D5	Mưa	lạnh	82	Yếu	Có
D6	Mưa	lạnh	70	Mạnh	Không
D7	Nhiều mây	lạnh	65	Mạnh	Có
D8	Nắng	Mát	95	Yếu	Không
D9	Nắng	lạnh	70	Yếu	Có
D10	Mưa	Mát	82	Yếu	Có
D11	Nắng	Mát	70	Mạnh	Có
D12	Nhiều mây	Mát	90	Mạnh	Có
D13	Nhiều mây	Nóng	75	Yếu	Có
D14	Mưa	Mát	82	Mạnh	Không

Ký hiệu $\text{Info}(S, A)$ là lượng thu hoạch thông tin của tập S theo thuộc tính A , trong đó S là tập mẫu nhận được từ D nhờ phân chia nó theo giá trị thuộc tính a_i tương ứng của thuộc tính này. Bảng 3.4 cho ta kết quả tính $\text{Gain}(D, \text{Độ ẩm}) = 0,102$ và lượng thu hoạch thông tin ứng với mỗi cách chia. Các bước khác được thực hiện như trong thuật toán ID3 và không mô tả lại để tránh nhầm chán.

Bảng 3.4. Tính $\text{Gain}(D, \text{Độ ẩm})$ sử dụng C4.5

	65		70		75		78		82		86		90		95		96	
Khoảng	\leq	$>$	\leq	$>$	\leq	$>$	\leq	$>$	\leq	$>$	\leq	$>$	\leq	$>$	\leq	$>$	\leq	$>$
có	1	8	3	6	4	5	5	4	7	2	7	2	8	1	8	1	9	0
Không	0	5	1	4	1	4	1	4	2	3	3	2	4	1	5	0	5	0
Entropy	0	0,961	0,811	0,971	0,721	0,991	0,65	1	0,764	0,971	0,881	1	0,918	1	0,961	0	0,94	0
$\text{Info}(S, A)$	0,892		0,925		0,8950		0,85		0,838		0,915		0,929		0,892		0,94	
Gain	0,048		0,015		0,045		0,09		0,102		0,025		0,011		0,048		0	

Lưu ý rằng, phương pháp xử lý này cho cây quyết định thường được dùng nhiều hơn tiếp cận rác hóa bằng cách chia khoảng như trong thống kê truyền thống.

3.3. MỘT SỐ VẤN ĐỀ KHÁC TRONG HỌC BẰNG CÂY QUYẾT ĐỊNH

Bây giờ ta xem xét các vấn đề khác phát sinh khi học bằng cây quyết định theo thuật toán ID3 và cách khắc phục trong C4.5 cùng với các phát triển của nó.

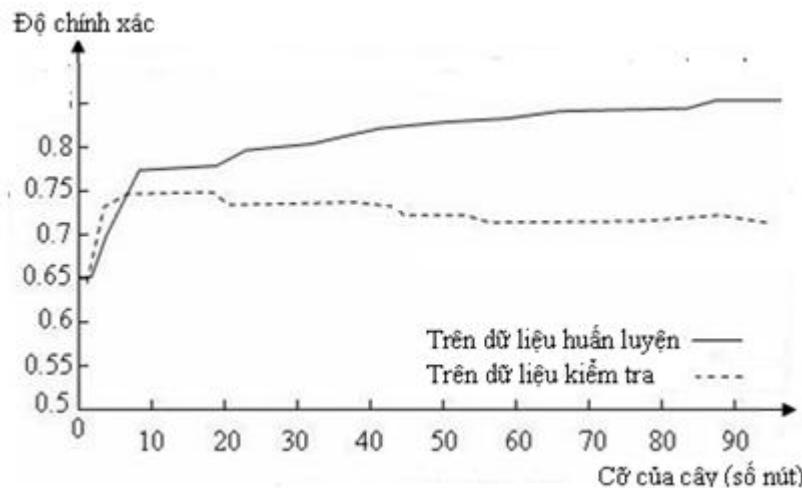
3.3.1. Phù hợp trội và cách khắc phục

Các thuật toán trên phát triển mỗi nhánh cây đủ sâu để giả thuyết tìm được phù hợp với tất cả tập mẫu huấn luyện khi dữ liệu không có nhiễu và bài toán thiết lập đúng đắn. Tuy nhiên, khi dữ liệu có nhiễu thì giả thuyết tìm được có thể có lỗi đúng lớn hơn lỗi tìm được bằng tập mẫu ít hơn. Trong trường hợp đó có hiện tượng phù hợp trội. Sau đây là định nghĩa chính xác cho hiện tượng này.

Định nghĩa 3.1. (*Phù hợp trội*) Cho một không gian giả thuyết H , ta nói một giả thuyết $h \in H$ là phù hợp trội với dữ liệu huấn luyện nếu tồn tại một giả thuyết khác $h' \in H$ mà h có sự sai lệch ít hơn h' trên tập các mẫu huấn luyện nhưng h' lại ít sai lệch hơn h trên toàn bộ tập mẫu có thể gặp.

Để đánh giá tính phù hợp trội, người ta phải chia dữ liệu quan sát được thành hai tập, một để huấn luyện còn một (tập kiểm tra) để đánh giá tính phù hợp theo tiêu chuẩn thống kê.

Ảnh hưởng của sự phù hợp trội trong một trường hợp điển hình của phương pháp học bằng cây quyết định được minh họa trong hình 3.5. Trong đó, trực hoành của đồ thị chỉ tổng số các nút của cây quyết định tại thời điểm đang xây dựng, trực tung chỉ độ chính xác của các dự đoán từ cây, đường nét liền chỉ độ chính xác của các cây quyết định trên tập mẫu huấn luyện, còn đường nét đứt chỉ độ chính xác trên tập mẫu kiểm tra. Thông thường, độ chính xác của cây quyết định trên tập đào tạo tăng dần đều khi cây được phát triển.



Hình 3.5. Hiện tượng phù hợp trội khi nghiên cứu bệnh nội tiết

Tuy nhiên độ chính xác khi tính trên tập mẫu kiểm tra độc lập, ban đầu tăng lên sau đó giảm dần. Như đã thấy, khi cõi của cây vượt quá 20 nút, việc phát triển thêm nữa của cây làm giảm độ chính xác của nó trên tập các mẫu kiểm tra bắt chấp sự tăng độ chính xác của nó trên tập mẫu huấn luyện.

Sở dĩ cây h phù hợp với các mẫu huấn luyện tốt hơn h' , nhưng lại xấu hơn trong các mẫu mới là vì tập dữ liệu huấn luyện chứa các mẫu sai, khi đó, cây học được phù hợp với các mẫu sai này sẽ cho kết quả kiểm tra nhiều lỗi và do đó lỗi đúng cũng lớn hơn.

Do tính ngẫu nhiên, nhiều khi dữ liệu không có nhiễu nhưng vẫn có thể xảy ra hiện tượng phù hợp trội. Chẳng hạn, cùng một thời tiết, người chơi có thể đi hoặc không đi với xác suất khác nhau. Trong trường hợp này, cây quyết định học được trên tập mẫu huấn luyện có thể có chất lượng tồi hơn một cây xấp xỉ hàm đích trên tập mẫu.

Có hai cách tiếp cận thông dụng để khắc phục hiện tượng phù hợp trội nhòe dựa vào kỹ thuật thống kê: tia bót để giảm lỗi và lược sau luật.

Tia bót để giảm lỗi

Quinlan đề xuất phương pháp này cho C4.5, trong đó, khi phát triển một nút mới cần dựa vào kết quả thống kê trên tập kiểm tra để đánh giá xem nên phát triển tiếp hay biến nó thành nút lá.

Đề xuất này thực chất là chặn sớm sự phát triển của cây. Tuy nhiên, trong các ứng dụng, người ta thường áp dụng tia cây sau khi đã học xong cây quyết định người ta cắt bỏ cây con từ các nút trong có nghi vấn. Theo cách tiếp cận này, tập dữ liệu quan sát được chia thành ba tập con: *tập huấn luyện*, *tập kiểm tra để ước lượng độ chính xác* và *tập đánh giá*. Ban đầu, người ta dùng *tập huấn luyện* và *tập kiểm tra để xây dựng* cây theo phương thức chặn sớm để có một cây với lỗi kiểm tra nhỏ nhất. Sau đó người ta dùng *tập đánh giá* để cắt bỏ các cành con (tia) nhằm giảm lỗi trên *tập đánh giá*.

Khi muốn *tia* một cành/cây con từ một nút nghi vấn, người ta gán nhãn phổ biến nhất của tập dữ liệu đào tạo phù hợp với nó (tức là tương thích với đường đi từ gốc tới nút này), nếu cây mới thu được có lỗi thấp hơn trên *tập đánh giá* thì cành/cây con này được tia và nút này trở thành nút lá. Trong trường hợp lỗi đánh giá không giảm thì giữ nguyên cây con trên cây.

Quá trình tia tiếp tục đến khi việc tia này là có hại (tức là nó làm giảm độ chính xác của cây trên *tập đánh giá*). Bằng cách này, những nút gần lá bị loại liên tục và độ chính xác của cây trên *tập kiểm tra tăng lên*.

Phương pháp này cho phép ta “quay lui” trong việc tìm kiếm giả thuyết, khắc phục được nhược điểm thứ ba và thứ sáu của thuật toán ID3. Hơn nữa, vì thuật toán ID3 ưu tiên cây ngắn và các thuộc tính có thu hoạch thông tin cao thì ở gần gốc nên các nút gần gốc ứng với thuộc tính có thu hoạch thông tin cao không được tách, phương pháp lược sau luật là một giải pháp tốt cho nhược điểm này và nhược điểm thứ tư của ID3.

Lược sau luật

Mặc dù biện pháp đầu tiên là trực tiếp hơn, nhưng trong thực tế, kỹ thuật lược sau luật lại thành công hơn việc tăng độ chính xác của giả thuyết tìm được. Kỹ thuật này thực hiện như sau:

1. Xây dựng cây quyết định từ tập huấn luyện cho đến khi dữ liệu huấn luyện phù hợp nhiều nhất đến mức có thể và cho phép xuất hiện phù hợp trội.
2. Chuyển đổi cây sang tập các luật { Nếu... thì...} tương đương bằng cách tạo ra một luật cho mỗi đường đi từ gốc đến lá.
3. Đơn giản (tổng quát hóa) mỗi luật bằng cách bỏ ra bất kỳ tiền điều kiện nào dẫn đến sự cải thiện độ chính xác của nó.
4. Sắp xếp các luật đã được lược bớt theo độ chính xác của nó, và xem xét chúng trong chuỗi này trong các trường hợp phân loại sau đó.

Để minh họa, ta xem lại cây quyết định trong hình 3.1. Mỗi thuộc tính kiểm tra theo một đường từ gốc đến lá tương ứng với một tiền điều kiện của luật và kết luận phân loại tại nút lá tương ứng. Chẳng hạn, nhánh bên trái nhất của cây được luật hóa bởi luật:

Nếu (Tiết trời = Nắng) Λ (Độ ẩm = cao) thì (chơi cầu lông = không)

Luật này có thể được tổng quát hóa thành một trong hai luật:

Nếu (Tiết trời = Nắng) thì (chơi cầu lông = không),

hoặc Nếu Độ ẩm = cao) thì (chơi cầu lông = không).

Luật mới nào khi thay vào luật ban đầu trong tập luật tìm được mà làm giảm lỗi trên tập đánh giá nhiều nhất thì được thay thế. Còn không có luật nào giảm được thì thôi. Sau đó rút gọn tập luật mới tìm được.

3.3.2. Tiêu chuẩn chọn thuộc tính

Ở trên, ta đã thấy rằng khi có một thuộc tính nhận quá nhiều giá trị thì lượng *Thu hoạch thông tin* của thuộc tính này thường lớn nhưng lại cho ít thông tin khi chọn nó làm nút gốc. Để khắc phục, Quinlan đề xuất có thể dùng *Tỷ lệ thu hoạch*(Gain Ratio) thay cho lượng *Thu hoạch thông tin*.

Tỷ lệ thu hoạch này phạt các thuộc tính có nhiều giá trị bằng cách thêm vào vào một hạng tử gọi là *thông tin chia* (SplitInformation), đại lượng này rất nhạy cảm với việc đánh giá tính rộng và đồng nhất khi chia tách dữ liệu theo giá trị thuộc tính:

$$SplitInformation(S, A) = -\sum_{i=1}^k \frac{|S_i|}{|S|} \log_2 \frac{|S_i|}{|S|} \quad (3.5)$$

trong đó S_1 đến S_k là k tập con của các mẫu huấn luyện khi phân chia S theo k giá trị của thuộc tính A . Lưu ý rằng $SplitInformation(S, A)$ thực tế là entropy của S ứng với giá trị của thuộc tính A . Khi đó Tỷ lệ thu hoạch thông tin của tập S đối với thuộc tính A được xác định như sau:

$$GainRatio(S, A) = \frac{Gain(S, A)}{SplitInformation(S, A)} \quad (3.6)$$

Khi sử dụng *GainRatio* thay cho *Gain* để lựa chọn các thuộc tính thì nảy sinh vấn đề là mẫu số ở biểu thức (3.6) có thể bằng 0 hoặc rất nhỏ khi $|S_i| \approx |S|$. Khi áp dụng, người ta thường khắc phục hiện tượng này nhờ kết hợp cả hai tiêu chuẩn: Đầu tiên, tính *Gain* cho mỗi thuộc tính, sau đó áp dụng *GainRatio* cho các thuộc tính có giá trị *Gain* trên trung bình để chọn thuộc tính tốt nhất.

3.3.3. Xử lý giá trị thuộc tính bị thiếu của mẫu

Trong ứng dụng, nhiều khi dữ liệu quan sát được của ta có thể có các mẫu bị thiếu giá trị của một số thuộc tính, chẳng hạn, sử dụng bệnh án của bệnh nhân để dự đoán bệnh và phương án điều trị những kết quả xét nghiệm sinh hóa ở một số bệnh nhân không có. Trong trường hợp này, các giá trị có được thường dùng để ước lượng giá trị thuộc tính thiếu theo hai cách:

Cách thứ nhất. Gán cho giá trị bị thiếu của mẫu bằng giá trị phổ biến nhất của các mẫu học tại nút đang xét và cùng lớp với nó.

Cách thứ hai. Áp dụng cho trường hợp có nhiều mẫu bị thiếu giá trị ở cùng một thuộc tính. Lúc đó người ta gán giá trị ngẫu nhiên cho tập mẫu bị thiếu này với phân bố bằng tần suất xuất hiện của giá trị tương ứng trong tập mẫu tại nút đang xét. Chẳng hạn, giả sử thuộc tính A giá trị boolean và nếu nút đang xét chứa 70 mẫu đã biết với giá trị thuộc tính này bằng 1 và 30 mẫu đã biết có giá trị thuộc tính này bằng 0. Khi đó ta ước lượng xác suất để $A(x)=1$ là 0,7 và xác suất để $A(x) = 0$ là 0,3. Bằng cách này, các mẫu thiếu giá trị thuộc tính này ở nút đang xét được được tạo ra với tỷ lệ khoảng 0,7 xuống nhánh ứng với thuộc tính $A=1$ còn với tỷ lệ khoảng 0,4 xuống nhánh thuộc tính $A=0$ của cây và dùng chúng để tính thu hoạch thông tin cho bước phát triển.

3.3.4. Cây phân lớp và hồi quy

Trong chương trước, ta đã làm quen với phương pháp giải tích để xác định hàm hồi quy cho bài toán hồi quy với các thuộc tính và hàm đều nhận giá trị thực. Nhiều trường hợp, tập thuộc tính có thể có cả thuộc tính nhận giá trị rời rạc và thuộc tính nhận giá trị thực. Trong các trường trường hợp này, ta có thể áp dụng kỹ thuật xử lý thuộc tính liên tục của Quinlan trong thuật toán C4.5 nêu trong mục 3.2.2 để xây dựng cây quyết định và gọi nó là cây hồi quy. Khi tập mẫu quan sát được quá lớn, có thể lấy giá trị trung bình giá trị hàm của các mẫu đào tạo trên khoảng nhỏ có nhiều giá trị hàm rời rạc.

Một tiếp cận khác, phương pháp cây phân lớp và hồi quy (Classification and Regression Trees: CART) do Breiman và các cộng sự (1984) để xuất, hiện nay thường được áp dụng cho bài toán phân lớp với giá trị thuộc tính thực, đặc biệt là cho các bài toán hồi quy. Phương pháp này xây dựng cây quyết định nhị phân dựa vào tiêu chuẩn thuần nhất Gini để chọn thuộc tính ưu tiên và rẽ nhánh. Thuật toán này thực hiện tương tự như ID3, chỉ khác ở chọn thuộc tính kiểm tra và cách rẽ nhánh ở mỗi nút. Để đơn giản cho trình bày, dưới đây xét các trường hợp giá trị thuộc tính thực.

Độ thuần nhất Gini

Xét tập mẫu quan sát S có nhãn thuộc tập k giá trị $\{y_i\}_{i=1}^k$, ký hiệu $f_i(S)$ là tần suất mẫu có nhãn y_i trong S . Khi đó độ thuần nhất Gini của S được ký hiệu là $GI(S)$ và được tính bởi công thức:

$$GI(S) = 1 - \sum_{i=1}^k f_i^2(S) \quad (3.7)$$

Thuộc tính ưu tiên và quy tắc chia

Ta xét biến x_i của không gian mẫu và giá trị biến này của các đối tượng trong S đều thuộc khoảng $[x_{imin}, x_{imax}]$. Với mọi $x_i^c \in [x_{imin}, x_{imax}]$, S được chia thành hai tập

$$S_L = \{x \in S: x_i < x_i^c\} \text{ và } S_R = \{x \in S: x_i \geq x_i^c\}. \quad (3.8)$$

Khi đó ta định nghĩa đại lượng $\Delta GI(S, x_i = x_i^c)$ đo thu hoạch Gini ở nhát cắt x_i^c :

$$\Delta GI(S, x_i = x_i^c) = GI(S) - [p_L GI(S_L) + p_R GI(S_R)]. \quad (3.9a)$$

Khi đó thu hoạch Gini của biến x_i được ký hiệu là $\Delta GI(S, x_i)$ được tính như sau:

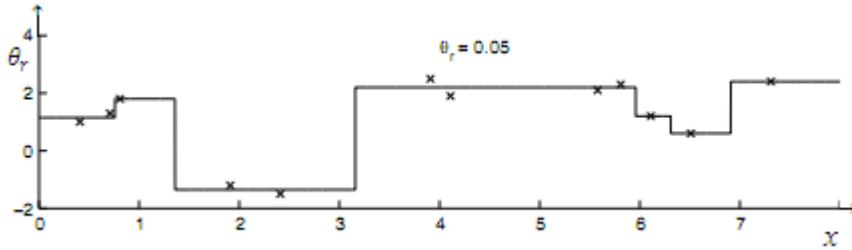
$$\Delta GI(S, x_i) = \max_{x_i^c} \{\Delta GI(S, x_i = x_i^c)\}, \quad (3.9b)$$

nhát cắt ứng với x_i^G của biến x_i làm cực đại thu hoạch Gini của biến này

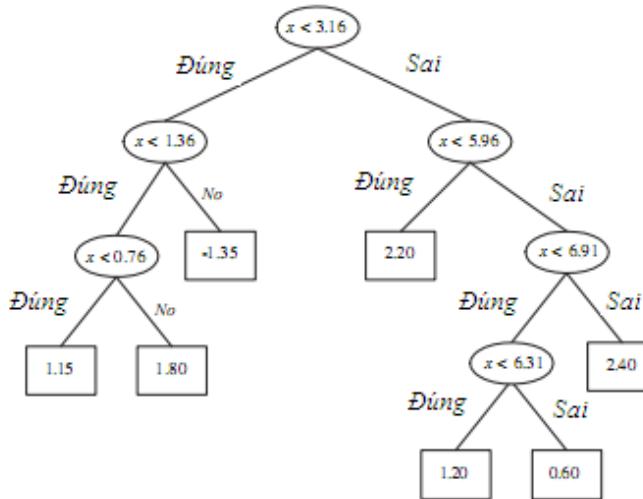
$$x_i^G = \arg \max_{x_i^c} \{\Delta GI(S, x_i = x_i^c)\} \quad (3.9c)$$

được gọi là nhát cắt Gini của biến x_i . Trong mỗi bước xây phát triển cây, CART chọn biến có thu hoạch Gini lớn nhất và nhát cắt Gini của nó làm điều kiện kiểm tra cho nút mới. Như vậy mỗi nút sẽ có hai nhánh tương ứng với S_L và S_R .

Cây quyết định trong hình 3.7 biểu diễn cây hồi quy của một hàm hồi quy làm tròn của hàm θ_r , có đồ thị trong hình 3.6, trong đó tập điểm được đánh dấu chéo là dữ liệu đào tạo.



Hình 3.6. Đồ thị hàm làm tròn cây hồi quy ở hình 3.7 của hàm của hàm θ_r



Hình 3.7. Một cây hồi quy của hàm θ_r có dữ liệu đào tạo trong hình 3.6

Mô tả chi tiết thuật toán và phát triển cho thuộc tính có giá trị rời rạc dành cho độc giả như là bài tập. Những phát triển của CART, người ta sử dụng thêm các tiêu chuẩn thống kê để chọn nút kiểm tra và nhánh phát triển.

KẾT LUẬN

Cây quyết định là một tiếp cận hữu hiệu cho các bài toán phân lớp với giá trị thuộc tính rời rạc. Thuật toán ID3 phát triển cây từ gốc xuống lá theo kiểu ăn tham để chọn thuộc tính phát triển và không quay lui nên có một số nhược điểm mà các thuật toán khác tìm cách khắc phục.

Phương pháp học bằng cây quyết định có thể áp dụng hiệu quả cả khi dữ liệu quan sát được bị nhiễu hoặc có mẫu bị thiếu giá trị thuộc tính. Thuật toán C4.5 cải

tiến ID3 dùng được cho cả các bài toán dữ liệu có thuộc tính liên tục, đặc biệt, có thể dùng để xây dựng cây hồi quy.

Hiện tượng phù hợp trội là hiện tượng quan trọng trong học có giám sát. Phương pháp tia cây và lược sau luật dựa trên kết quả thống kê là các giải pháp cho hiện tượng này. Trong đó, phương pháp lược sau luật cho giả thuyết chính xác hơn.

BÀI TẬP

- Cho tập mẫu với 2 thuộc tính A_1 và A_2 như sau:

Mẫu	A_1	A_2	Lớp
x_1	T	T	ω_1
x_2	T	T	ω_1
x_3	T	F	ω_2
x_4	F	F	ω_1
x_5	F	T	ω_2
x_6	F	T	ω_2

- a) Hãy tính Entropy tập dữ liệu theo phân lớp.
 b) Tính độ thu hoạch thông tin của thuộc tính A_2
- Xây dựng cây quyết định với dữ liệu cho trong bảng 3.2 và có thêm dữ liệu hai quan dữ liệu quan sát được::

D15: <nắng, mát, thường, yếu, có> và D16: <nắng, mát, thường, mạnh>.

- Giải thích vì sao nói tiêu chuẩn thu hoạch thông tin có khuynh hướng ưu tiên chọn thuộc tính có nhiều giá trị?
- Mô tả thuật toán CART cho trường hợp giá trị thuộc tính thực.
- Mô tả thuật toán CART cho trường hợp giá trị thuộc tính rời rạc.
- Mỗi đối tượng có hai thuộc tính đặc trưng, đặc trưng thứ nhất nhận hai giá trị 0 hoặc 1, đặc trưng thứ hai nhận giá trị trong 5 chữ cái. Dự liệu quan sát được gồm hai lớp cho bởi bảng:

$\omega_1: 1A\ 0E\ 0B\ 1B\ 0D$
$\omega_2: 0A\ 0C\ 1C\ 0B\ 1D$

- Xây dựng cây quyết định theo thuật toán ID3 sử dụng tiêu chuẩn thu hoạch thông tin
- Xây dựng cây quyết định theo thuật toán ID3 sử dụng tiêu chuẩn tỷ lệ thu hoạch thông tin.

7. Xây dựng cây quyết định nhờ dùng các thuật toán ID3 và CART nếu dữ liệu quan sát của hai lớp có hai đặc trưng thu được như sau:

Mẫu	a_1	a_2	Lớp
x_1	0,15	0,84	ω_1
x_2	0,8	0,54	ω_1
x_3	0,28	0,36	ω_1
x_4	0,38	0,70	ω_1
x_5	0,52	0,48	ω_1
x_6	0,58	0,74	ω_1
x_7	0,74	0,76	ω_1
x_8	0,08	0,14	ω_2
x_9	0,24	0,16	ω_2
x_{10}	0,70	0,20	ω_2
x_{11}	0,90	0,28	ω_2
x_{12}	0,64	0,90	ω_2
x_{13}	0,74	0,36	ω_2
x_{14}	0,10	0,30	ω_2

8. Xây dựng cây hồi quy cho hàm số ở bài tập 6 chương 2

Chương 4

PHÂN LỚP MẪU NHỜ HÀM PHÂN BIỆT

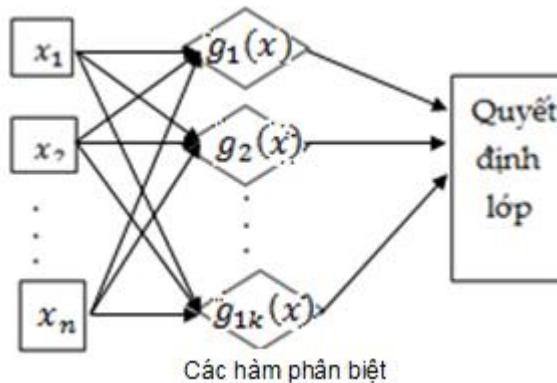
Hai chương trước đã giới thiệu các phương pháp phân lớp bằng cách tìm tập luật biểu diễn hàm đích dạng liên kết giá trị-thuộc tính với các thuộc tính nhận giá trị trong tập hữu hạn. Trong thực tế, ta thường gặp bài toán phân lớp mà các đối tượng có đặc trưng là vectơ n-chiều. Chương này giới thiệu tiếp cận giải tích để xử lý bài toán này nhờ sử dụng các hàm phân biệt tuyến tính.

4.1. HÀM PHÂN BIỆT VÀ MIỀN QUYẾT ĐỊNH

Trong các bài toán phân lớp mẫu, mỗi mẫu mỗi mẫu x trong tập đối tượng X thường được biểu diễn bởi n đặc trưng: $x = (x_1, \dots, x_n)$. Để phân X thành k lớp $\{\omega_1, \omega_2, \dots, \omega_k\}$, người ta thường dùng k *hàm phân biệt* hay còn gọi là *hàm quyết định*: $g_i(x)$ $i = 1, \dots, k$, mỗi hàm xác định một lớp. Mỗi đối tượng có vectơ đặc trưng x được gán cho lớp ω_i nếu giá trị hàm phân biệt $g_i(x)$ lớn nhất, tức là:

$$x \in \omega_i \text{ nếu } g_i(x) > g_j(x) \forall j \neq i \quad (4.1)$$

Bằng cách này, bộ phân lớp được xem như một mạng hay là một máy dùng để tính k hàm phân biệt và chọn lớp ứng với giá trị hàm phân biệt lớn nhất để gán cho mỗi đối tượng như được minh họa trong hình 4.1.



Hình 4.1. Cấu trúc bộ phân lớp có n đầu vào và k hàm phân biệt

Trong trường hợp các đặc trưng nhận giá trị thực, các hàm phân biệt chia không gian thành k miền $\{R_i\}_{i=1}^k$, mỗi miền R_i xác định một lớp:

$$R_i = \{x \in X : g_i(x) = \max\{g_j(x) \mid j \leq k\}\}$$

và được gọi là *miền quyết định* của lớp ω_i tương ứng. Khi đó, quy tắc quyết định là: x thuộc R_i (được gán vào lớp ω_i nếu $g_i(x) > g_j(x) \forall j \neq i$). (4.2a)

Các miền quyết định này được tách bởi các *biên quyết định*, chúng là hợp của các phần mặt/siêu mặt trong không gian đặc trưng chứa các điểm có giá trị hàm phân biệt lớn nhất bằng với giá trị của ít nhất một hàm phân biệt khác. Các biên quyết định tách các lớp trong không gian đặc trưng.

Khi chỉ có hai lớp và $n \geq 3$, biên quyết định còn được gọi là *mặt/siêu mặt* (khi $n > 3$) *quyết định*. Trong trường hợp này, thay vì dùng hai hàm phân biệt $g_1(x)$ và $g_2(x)$ ta chỉ cần dùng một hàm phân biệt $g(x)$ là đủ:

$$g(x) = g_1(x) - g_2(x) \quad (4.2b)$$

và quy tắc quyết định lúc đó là:

$$x \text{ thuộc } \omega_1 \text{ nếu } g(x) > 0 \text{ và thuộc } \omega_2 \text{ nếu } g(x) < 0. \quad (4.2.c)$$

Lúc này mặt/siêu mặt (biên) quyết định là tập $\{x \in R^n : g(x) = 0\}$.

Hàm phân biệt tuyến tính

Trường hợp không gian đặc trưng n -chiều, hàm phân biệt tuyến tính được biểu diễn bởi các trọng số w_0, w_1, \dots, w_n và có dạng:

$$g_i(x) = \sum_{i=1}^n w_i x_i + w_0 = \mathbf{w}' \mathbf{x} + w_0. \quad (4.3a)$$

Công thức (4.3a) có thể viết gọn lại là:

$$g_i(x) = \mathbf{w}' \mathbf{x} + w_0 = \mathbf{w}^* \mathbf{x}^*, \quad (4.3b)$$

trong đó $\mathbf{w}^* = [w_0, w_1, \dots, w_n]'$, và $\mathbf{x}^* = [1, x_1, \dots, x_d]'$ (dấu ' để chỉ vectơ chuyển vị).

Như đã nói ở trên, trường hợp hai lớp, các *mặt/siêu mặt* quyết định được xác định bởi tập $H = \{x \mid g(x) = 0\}$ là các *mặt phẳng/siêu phẳng quyết định* được đặc trưng bởi khoảng cách D_0 từ gốc tọa độ và vectơ pháp tuyến \mathbf{n} cho bởi công thức:

$$D_0 = \frac{|w_0|}{\|\mathbf{w}\|}; \quad \mathbf{n} = \frac{\mathbf{w}}{\|\mathbf{w}\|}. \quad (4.4a)$$

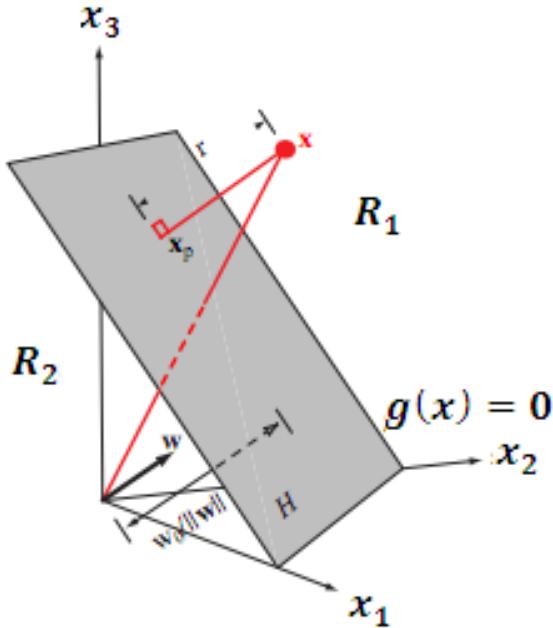
H chia không gian đặc trưng thành hai nửa không gian: R_1 ứng với lớp ω_1 và R_2 ứng với lớp ω_2 . Hàm phân biệt đo khoảng cách từ x tới H . Để trực quan hóa, ta xét biểu diễn của x dưới dạng:

$$\mathbf{x} = \mathbf{x}_p + r \frac{\mathbf{w}}{\|\mathbf{w}\|}, \quad (4.4b)$$

trong đó \mathbf{x}_p là hình chiếu trực giao của \mathbf{x} lên H còn r là khoảng cách đai số từ \mathbf{x} tới H (dương nếu \mathbf{x} thuộc miền dương và âm nếu \mathbf{x} thuộc miền âm của hàm quyết định). Bởi vì $\mathbf{g}(\mathbf{x}_p) = 0$ nên $\mathbf{g}(\mathbf{x}) = \mathbf{w}'\mathbf{x} + w_0 = r\|\mathbf{w}\|$.

$$\text{hay } r = \frac{\mathbf{g}(\mathbf{x})}{\|\mathbf{w}\|}. \quad (4.4c)$$

Một minh họa hình học được giới thiệu trong hình 4.2.



Hình 4.2. Một mặt phẳng quyết định H trong R^3

Chú ý. Khái niệm hàm tuyến tính ở đây khác với hàm tuyến tính của họ hàm cơ sở $\{\varphi_k(x)\}_{k=1}^N$ trong mục 2.4 ở chương 2.

Các dạng hàm phân biệt khác

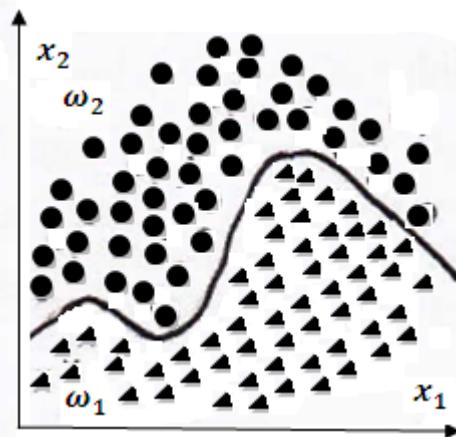
Phân biệt tuyến tính có cơ sở toán học vững chắc, cho phép ta trả lời được các câu hỏi “ có tồn tại hàm phân biệt không? nếu có thì tìm như thế nào?”, tuy nhiên miễn áp dụng được khá hạn chế. Khi mở rộng không gian giả thuyết nhờ dùng dạng hàm phân biệt phức tạp hơn thì khó xác định hàm tách các tập mẫu quan sát được. Vì vậy, dạng hàm phân biệt thường được chọn đủ đơn giản để dễ xác định nhưng phải đáp ứng được việc phân biệt các lớp. Hàm phân biệt là tổ hợp tuyến tính của các hàm cơ sở cho trước là một dạng hàm được ưa dùng. Cho trước họ hàm độc lập tuyến tính $\{f_k\}_{k=1}^m$, dựa trên tập mẫu quan sát được, ta tìm hàm phân biệt dạng:

$$g(\mathbf{x}) = \sum_{i=1}^m w_i f_i(\mathbf{x}) \quad (4.5.a)$$

trong đó w_1, w_2, \dots, w_m là các tham số cần tìm. Nói riêng, khi hàm phân biệt là đa thức thì các tham số w_i là các hệ số của đa thức. Hình 4.5 minh họa biên quyết định cho

trường hợp hai lớp trong không gian hai chiều với hàm phân biệt là một đa thức bậc bốn:

$$g(x) = w_{14}x_1^4 + w_{13}x_2^4 + w_{12}x_1^2x_2^2 + w_{10}x_1x_2^3 + w_9x_1^3 + w_8x_2^3 + w_7x_1^2x_2 + w_6x_1x_2^2 + w_5x_1^2 + w_4x_2^2 + w_3x_1x_2 + w_2x_1 + w_1x_2 + w_0 \quad (4.5b)$$



Hình 4.3. Miền và biên quyết định đối với hàm phân biệt bậc bốn

Trong không gian đặc trưng 2-chiều, với hàm phân biệt bậc 4 ta cần xác định 15 trọng số (14 trọng số đa thức và một khuynh hướng). Tổng quát hơn, khi dùng hàm phân biệt bậc k cho không gian đặc trưng d -chiều ta cần xác định :

$$C_{d+k}^k = \frac{d+k!}{d!k!} \text{ hệ số đa thức} \quad (4.5c)$$

Với $d=2; k=4$ ta có :

$$C_{d+k}^k = \frac{d+k!}{2!4!} = \frac{5.6}{2} = 15 \text{ hệ số đa thức.}$$

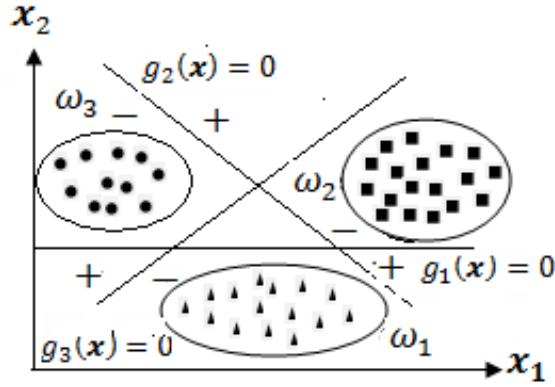
4.2. CÁC MÔ HÌNH TUYẾN TÍNH

4.2.1. Tách được bởi siêu phẳng

Thông thường các mẫu trong mỗi lớp được đặc trưng bởi một phân bố ngẫu nhiên. Trong hình 4.1, các ellip biểu diễn “*biên*” của phân phối và còn gọi là *giới hạn lớp*. Khi có nhiều lớp, các hàm phân biệt tuyến tính có thể tách các lớp dưới hai dạng: *tách tuyệt đối* và *tách từng cặp*. Ta gọi chung các trường hợp này là tách được bởi siêu phẳng hay tách được tuyến tính.

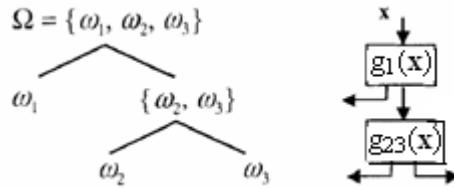
Tách tuyệt đối

Khi mỗi lớp được tách hẳn khỏi các lớp còn lại ta nói tập mẫu tách được tuyệt đối. Hình 4.4 minh họa một trường hợp ba lớp tách được tuyệt đối trong R^2 .



Hình 4.4. Ví dụ tách tuyệt đối bởi các siêu phẳng trong R^2

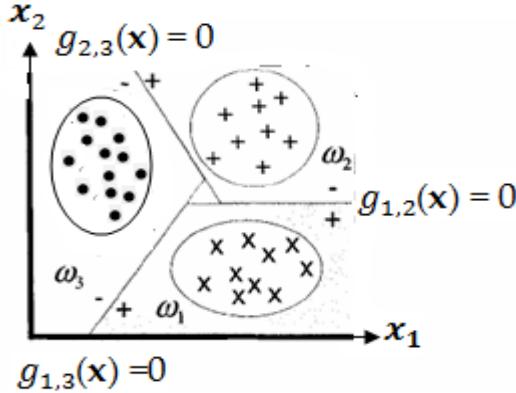
Khi các lớp tách được tuyệt đối thì hệ nhận dạng sẽ có cấu trúc phân cấp mô tả bởi cây quyết định nhị phân như minh họa trong hình 4.5, trong đó $g_{23}(\mathbf{x}) = g_2(\mathbf{x}) - g_3(\mathbf{x})$ để tách ω_1 với ω_2 .



Hình 4.5. Cấu trúc phân cấp của hệ phân 3 lớp tách được tuyệt đối

Tách theo cặp

Nhiều khi tập mẫu không tách được tuyệt đối nhưng có thể tách theo từng cặp như minh họa trong hình 4.6.



Hình 4.6. Ba lớp tách được từng cặp trong R^2

Chú ý rằng các trường hợp tách được từng cặp không mô tả được bằng cây quyết định. Trường hợp này người ta xác định $\frac{k(k-1)}{2}$ siêu phẳng tách từng cặp lớp nhò dùng các hàm phân biệt thỏa mãn:

$$g_{ij}(\mathbf{x}) > 0 \quad \forall \mathbf{x} \in \omega_i \text{ và } g_{ij}(\mathbf{x}) < 0 \quad \forall \mathbf{x} \in \omega_j, (g_{ij}(\mathbf{x}) = -g_{ij}(\mathbf{x})). \quad (4.6a)$$

Miền quyết định lúc này là:

$$R_i = \{\mathbf{x}; g_{ij}(\mathbf{x}) > 0, i, j = 1, \dots, k, j \neq i\} \quad (4.6b)$$

Để xây dựng các bộ phân lớp, ta dùng các thuật toán tìm vecto \mathbf{w} và hệ số w_0 của hàm phân biệt như trong biểu thức (4.3a). Các thuật toán này còn gọi là thuật toán học hay đào tạo.

4.2.2. Thuật toán học perceptron

Để đơn giản, ta xét bài toán có 2 lớp ω_1, ω_2 và không gian đặc trưng n-chiều. Cho tập dữ liệu quan sát được $D = D_1 \cup D_2$ trong đó D_i là các mẫu thuộc lớp ω_i tương ứng, ta tìm vecto \mathbf{w} và hệ số w_0 sao cho:

$$g(\mathbf{x}) = \mathbf{w}'\mathbf{x} + w_0 > 0 \text{ với mọi } \mathbf{x} \in D_1 \text{ và } g(\mathbf{x}) < 0 \text{ với mọi } \mathbf{x} \in D_2. \quad (4.7)$$

Trước hết ta định nghĩa hai hàm *hardlim*: $R \rightarrow R$ và $t: D \rightarrow R$ như sau:

$$\text{hardlim}(\mathbf{x}) = \begin{cases} 1 & \forall x > 0 \\ 0 & \forall x \leq 0 \end{cases} \quad (4.8a)$$

$$t(\mathbf{x}) = \begin{cases} 1 & \forall \mathbf{x} \in D_1 \\ 0 & \forall \mathbf{x} \in D_2 \end{cases} \quad (4.8b)$$

Với mỗi \mathbf{w}, w_0 đã biết và $\mathbf{x} \in D$, đặt:

$$a(\mathbf{x}) = \text{hardlim}(\mathbf{w}'\mathbf{x} + w_0) \quad (4.8c)$$

$$e(\mathbf{x}) = t(\mathbf{x}) - a(\mathbf{x}) \quad (4.8e)$$

Nếu D_1 và D_2 tách được bởi siêu phẳng thì thuật toán học perceptron do Rosenblatt đề xuất (1953) luôn hội tụ sau hữu hạn bước. Thuật toán khởi tạo ngẫu nhiên các giá trị ban đầu cho \mathbf{w} và w_0 , sau đó thực hiện lặp thủ tục lấy lần lượt các mẫu quan sát được trong D để tính $e(\mathbf{x})$ và hiệu chỉnh các tham số cần tìm theo quy tắc:

- nếu $e(\mathbf{x})=0$ (nhận dạng đúng) thì để nguyên

$$\bullet \text{ nếu } e(\mathbf{x})=1 \text{ thì } \mathbf{w}^{mới} = \mathbf{w}^{c\underline{u}} + \mathbf{x}; w_0^{mới} = w_0^{c\underline{u}} + 1 \quad (4.9b)$$

$$\bullet \text{ nếu } e(\mathbf{x})=-1 \text{ thì } \mathbf{w}^{mới} = \mathbf{w}^{c\underline{u}} - \mathbf{x}; w_0^{mới} = w_0^{c\underline{u}} - 1 \quad (4.9c)$$

Các công thức (4.14a-c) có thể viết chung là:

$$\mathbf{w}^{m\hat{o}i} = \mathbf{w}^{c\hat{u}} + e(\mathbf{x})\mathbf{x}; \quad w_0^{m\hat{o}i} = w_0^{c\hat{u}} + e(\mathbf{x}) \quad (4.9d)$$

Thuật toán được đặc tả trong hình 4.7.

```

Procedure of Thuật toán học Perceptron .
Begin
    Initialize  $\mathbf{w}$ ,  $w_0$  // khởi tạo  $\mathbf{w}$  và  $w_0$  tùy ý ;
    Repeat
        For all  $\mathbf{x} \in D$  do
            Tính  $e(\mathbf{x})$ ;
             $\mathbf{w} \leftarrow \mathbf{w} + e(\mathbf{x})\mathbf{x}$  // hiệu chỉnh  $\mathbf{w}$ ,
             $w_0 \leftarrow w_0 + e(\mathbf{x})$ , hiệu chỉnh  $w_0$ ;
        until  $\mathbf{w}$  không đổi//phân lớp đúng;
    End

```

Hình 4.7. Thuật toán học Perceptron

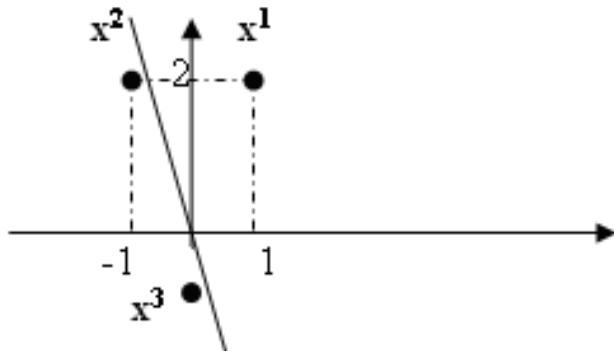
Ví dụ: Cho $D_1 = \{\mathbf{x}^1 = \begin{pmatrix} 1 \\ 2 \end{pmatrix}\}$, $D_2 = \{\mathbf{x}^2 = \begin{pmatrix} -1 \\ 2 \end{pmatrix}, \mathbf{x}^3 = \begin{pmatrix} 0 \\ -1 \end{pmatrix}\}$ như đặc tả trong hình 4.7. Trong trường hợp này, để đơn giản ta cho $w_0 = 0$ và chỉ tìm \mathbf{w} .

Lấy ngẫu nhiên $\mathbf{w} = (1, 0, -0, 8)'$ và thủ tục học thực hiện lặp hiệu chỉnh \mathbf{w} theo các mẫu quan sát:

- Cho $\mathbf{x} = \mathbf{x}^1$ khi đó $a(\mathbf{x}^1) = \text{hardlim}(\mathbf{w}' \mathbf{x}^1) = \text{hardlim}((1, 0, -0, 8)' \begin{pmatrix} 1 \\ 2 \end{pmatrix}) = \text{hardlim}(-0, 6) = 0$
 $e(\mathbf{x}^1) = t - a = 1 - 0 = 1$: $\mathbf{w}^{m\hat{o}i} = \mathbf{w}^{c\hat{u}} + e(\mathbf{x}^1) \mathbf{x}^1 = \mathbf{w} + \mathbf{x}^1 = (1, 0, -0, 8)' + \begin{pmatrix} 1 \\ 2 \end{pmatrix} = \begin{pmatrix} 2 \\ 1, 2 \end{pmatrix}$
- Cho $\mathbf{x} = \mathbf{x}^2$ khi đó $a(\mathbf{x}^2) = \text{hardlim}(\mathbf{w}' \mathbf{x}^2) = \text{hardlim}((2, 0, 1, 2)' \begin{pmatrix} -1 \\ 2 \end{pmatrix}) = \text{hardlim}(0, 44) = 1$
 $e(\mathbf{x}^2) = t - a = 0 - 1 = -1$, $\mathbf{w}^{m\hat{o}i} = \mathbf{w}^{c\hat{u}} + e(\mathbf{x}^2) \mathbf{x}^2 = \mathbf{w} - \mathbf{x}^2 = (2, 0, 1, 2)' - \begin{pmatrix} -1 \\ 2 \end{pmatrix} = \begin{pmatrix} 3 \\ -0.8 \end{pmatrix}$
- Cho $\mathbf{x} = \mathbf{x}^3$ khi đó $a(\mathbf{x}^3) = \text{hardlim}(\mathbf{w}' \mathbf{x}^3) = \text{hardlim}((3, -0.8)' \begin{pmatrix} 0 \\ -1 \end{pmatrix}) = \text{hardlim}(0.8) = 1$
 $e(\mathbf{x}^3) = t - a = 0 - 1 = -1$, $\mathbf{w}^{m\hat{o}i} = \mathbf{w}^{c\hat{u}} + e(\mathbf{x}^3) \mathbf{x}^3 = \mathbf{w} - \mathbf{x}^3 = (3, -0.8)' - \begin{pmatrix} 0 \\ -1 \end{pmatrix} = \begin{pmatrix} 3 \\ 0.2 \end{pmatrix}$

Bây giờ ở vòng lặp thứ hai ta thấy $e(\mathbf{x}^1) = e(\mathbf{x}^2) = e(\mathbf{x}^3) = 0$ và \mathbf{w} không đổi. Vậy ta có hàm phân biệt được cho trong công thức (4.14d) và được minh họa trong hình 4.8.

$$g(\mathbf{x}) = 3x_1 + 0,2 x_2 \quad (4.9e)$$



Hình 4.8. Ví dụ tách hai tập $\{x^1\}$ và $\{x^2, x^3\}$

Chú ý. Thuật toán này chỉ dùng khi tập mẫu D là tách được tuyến tính, ngoài ra, có thể có nhiều siêu phẳng tách khi chọn giá trị khởi tạo khác nhau.

Để khắc phục nhược điểm của thuật toán Perceptron, thuật toán sau đây cho một giải pháp lựa chọn.

4.2.3. Thuật toán bình phương tối thiểu

Ta xét bài toán như mục trên với tập quan sát được $D = \{\mathbf{x}^i\}$ ($i=1,\dots,N$) đã biết nhãn $t(\mathbf{x}^i)$ tương ứng, $a(\mathbf{x})$ tính bởi:

$$a(\mathbf{x}) = \sum_{i=1}^N w_i x_i + w_0. \quad (4.10a)$$

Quy tắc phân lớp sẽ là:

$$\mathbf{x} \in \omega_1 \text{ nếu } a(\mathbf{x}) > 0 \text{ và } \mathbf{x} \in \omega_2 \text{ nếu } a(\mathbf{x}) < 0 \quad (4.10b)$$

Nói chung tập mẫu này không đảm bảo tách được bởi siêu phẳng và không đảm bảo thuật toán học perceptron hội tụ nên Widrow và Hoff (1960) đề xuất dùng phương pháp bình phương tối thiểu (LMS) để xác định w và w_0 .

Sai số trung bình phương

Với tập D , vecto w và giá trị w_0 đã cho, sai số trung bình phương của bộ phân lớp là:

$$E = E(D, w, w_0) = \frac{1}{N} \sum_{i=1}^N [t(\mathbf{x}^i) - a(\mathbf{x}^i)]^2, \quad (4.11a)$$

trong đó $t(\mathbf{x})$ tính theo (4.8b) còn $a(\mathbf{x})$ tính theo (4.10a). Khi \mathbf{x} lấy ngẫu nhiên cùng phân bố với D trong không gian đặc trưng thì $E(D, w, w_0)$ là xấp xỉ của kỳ vọng bình phương sai số:

$$E[t(\mathbf{x}^i) - a(\mathbf{x}^i)]^2. \quad (4.11b)$$

Các trọng số w_i ($i=0, \dots, n$) sẽ được tìm để E trong biểu thức (4.11a) đạt cực tiểu. Tương tự như trong mục 2.4 chương 2, bài toán tìm cực tiểu hàm E luôn có nghiệm và có thể tìm cực tiểu nhờ giải hệ phương trình:

$$\frac{\partial E}{\partial w_k} = 0 \text{ với } k=0, \dots, n \quad (4.12a)$$

Để ý rằng các $t(\mathbf{x}^i)$ là hằng số còn $a(\mathbf{x}^i)$ tuyến tính đối với các w_k . Hệ phương trình (4.12a) sẽ tương đương với hệ 4.12b-c), trong đó (4.12c) ứng với đạo hàm theo biến w_0 :

$$\sum_{i=1}^N [t(\mathbf{x}^i) - a(\mathbf{x}^i)] x_k^i = 0, \quad k=1, \dots, N \quad (4.12b)$$

$$\sum_{i=1}^N [t(\mathbf{x}^i) - a(\mathbf{x}^i)] = 0 \quad (4.12c)$$

Khi N lớn, việc giải hệ (4.12b-c) sai số lớn và độ phức tạp là $O(N^3)$ nên khó thực hiện. Widrow và Hoff đề xuất dùng phương pháp gradient để tìm các w_k , họ chỉ ra rằng nếu lấy mẫu ngẫu nhiên trong D thì thay cho sai số E (D, w, w_0) ở bước k có thể dùng xấp xỉ bởi sai số $\hat{E}(\mathbf{x}) = [t(\mathbf{y}^k) - a(\mathbf{y}^k)]^2 = e(\mathbf{y}^k)^2$ trong đó \mathbf{y}^k là mẫu lấy ngẫu nhiên trong D ở bước lặp k . Thuật toán thực hiện như sau. Sau khi khởi tạo ngẫu nhiên các kệ số w_i ($i=0, \dots, n$) ban đầu, thực hiện lặp theo k . Lấy ngẫu nhiên \mathbf{y}^k trong tập mẫu D và hiệu chỉnh trọng số theo công thức:

$$w^{moi} = w^{cu} + 2\alpha e(\mathbf{y}^k) \mathbf{y}^k \quad (4.13a)$$

$$w_0^{moi} = w_0^{cu} + 2\alpha e(\mathbf{y}^k) \quad (4.13b)$$

trong đó α là tốc độ học thỏa mãn điều kiện $0 < \alpha < 1/\lambda_{max}$ với λ_{max} là giá trị riêng lớn nhất của ma trận tương quan mẫu :

$$R = E(\mathbf{x}\mathbf{x}') = \frac{1}{N} \sum_{i=1}^N [\mathbf{x}^i(\mathbf{x}^i)'] \quad (4.13c)$$

Điều kiện kết thúc có thể là số lần lặp hoặc độ lệch của sai số trung bình phương nhỏ hơn ε cho trước. Thuật toán được đặc tả trong hình 4.9.

Chú ý. Tốc độ học α có thể lấy bằng phương pháp thử sai hoặc giảm dần về không mà không phải tính giá trị riêng của R .

Ví dụ.

Cho D gồm hai vecto $\{\mathbf{x}^1 = \begin{bmatrix} 1 \\ -1 \end{bmatrix}, \mathbf{x}^2 = \begin{bmatrix} 1 \\ 1 \\ -1 \end{bmatrix}\}$ và $\{t(\mathbf{x}^1) = -1\}, t(\mathbf{x}^2) = 1$ (ở đây t lấy ± 1

cho chong hội tụ).

```

Procedure thuật toán LMS .
Begin
    Initialize  $w, w_0$  //khởi tạo giá trị ban đầu tùy ý;
    Repeat
        Lấy ngẫu nhiên  $y$  trong  $D$ ;
        Tính  $a(y)$  và  $e(y)$ ;
        Hiệu chỉnh  $w, w_0$ // theo (4.13a-b);
    until điều kiện kết thúc
End

```

Hình 4.9. Thuật toán Widrow-Hoff

$$\begin{aligned}
 \text{Ta có } R = E(\mathbf{x}\mathbf{x}') &= \frac{1}{2} \mathbf{x}^1(\mathbf{x}^1)' + \frac{1}{2} \mathbf{x}^2(\mathbf{x}^2)' = \frac{1}{2} \begin{bmatrix} 1 \\ -1 \\ -1 \end{bmatrix} [1 \ -1 \ -1] + \frac{1}{2} \begin{bmatrix} 1 \\ 1 \\ -1 \end{bmatrix} [1 \ 1 \ -1] \\
 &= \begin{bmatrix} 1 & 0 & -1 \\ 0 & 1 & 0 \\ -1 & 0 & 1 \end{bmatrix}
 \end{aligned}$$

R có 3 giá trị riêng là $\lambda_1=0,1$, $\lambda_2=0$, $\lambda_3=2$. Vậy tốc độ học ổn định là

$$\alpha < 1 / \lambda_{\max} = 1/2 = 0,5.$$

Ta lấy $\alpha = 0,2$ và $\mathbf{w}(0)=[0 \ 0 \ 0]'$ để đơn giản luôn lấy $w_0=0$ và tín hiệu vào lần lượt là $\mathbf{x}^1, \mathbf{x}^2, \mathbf{x}^1, \mathbf{x}^2, \dots$

- Với $\mathbf{y}^1 = \mathbf{x}^1 = \begin{bmatrix} 1 \\ -1 \\ -1 \end{bmatrix}$; $t(\mathbf{x}^1)=-1$ thì ta có

$$a(\mathbf{y}^1) = \mathbf{w}(0)' \mathbf{y}^1 = \mathbf{w}(0)' \mathbf{x}^1 = [0 \ 0 \ 0] \begin{bmatrix} 1 \\ -1 \\ -1 \end{bmatrix} = [0].$$

$$e(\mathbf{y}^1) = t(\mathbf{y}^1) - a(\mathbf{y}^1) = -1 - 0 = -1$$

$$\mathbf{w}(1) = \mathbf{w}(0) + 2\alpha e(\mathbf{y}^1) \mathbf{y}^1 = [0 \ 0 \ 0] + 2(0.2)(-1) \begin{bmatrix} 1 \\ -1 \\ -1 \end{bmatrix} = [-0.4 \ 0.4 \ 0.4]'$$

- Với $\mathbf{y}^2 = \mathbf{x}^2$, $t(\mathbf{x}^2)=1$ tacó :

$$a(\mathbf{y}^2) = \mathbf{w}(\mathbf{y}^2)' \mathbf{y}^2 = [-0,4 \ 0,4 \ 0,4] \begin{bmatrix} 1 \\ 1 \\ -1 \end{bmatrix} = -0,4;$$

$$e(\mathbf{y}^2) = t(\mathbf{y}^2) - a(\mathbf{y}^2) = 1 - (-0,4) = 1,4$$

$$\mathbf{w}(2) = \mathbf{w}(1) + 2\alpha e(\mathbf{y}^2) \mathbf{y}^2 = [-0,4 \ 0,4 \ 0,4]' + 2(0,2)(1,4) \begin{bmatrix} 1 \\ 1 \\ -1 \end{bmatrix} = [0,16 \ 0,96 \ -0,16]'$$

- Với $\mathbf{y}^3 = \mathbf{x}^l = \begin{bmatrix} 1 \\ -1 \\ -1 \end{bmatrix}$; $t(\mathbf{y}^3) = -1$ ta có :

$$a(\mathbf{y}^3) = \mathbf{w}(\mathbf{y}^3)' \mathbf{y}^3 = [0,16 \ 0,96 \ 0,16] \begin{bmatrix} 1 \\ -1 \\ -1 \end{bmatrix} = -0,64$$

$$e(\mathbf{y}^3) = t(\mathbf{y}^3) - a(\mathbf{y}^3) = -1 - (-0,64) = -0,36$$

$$\mathbf{W}(3) = \mathbf{W}(2) + 2\alpha e(\mathbf{y}^3) \mathbf{y}^3 = [0,16 \ 0,96 \ 0,16]' + 2(0,2)(-0,36) \begin{bmatrix} 1 \\ -1 \\ -1 \end{bmatrix} = [0,016 \ 1,1040 \ -0,0160]'$$

Lặp tiếp thì $\mathbf{w}(k)$ hội tụ tới $[0 \ 1 \ 0]$.

Widrow và Hoff đã chứng minh rằng với tốc độ học đủ nhỏ thì thuật toán luôn hội tụ tới nghiệm duy nhất của (4.12a).

Hai phương pháp trên không chú ý tới đặc điểm phân bố của mỗi lớp trong tập D , khi các đối tượng là trộn lẫn của các lớp phân bố ngẫu nhiên có tâm thì có thể tìm hàm phân biệt nhanh hơn nhiều.

4.2.4. Phân lớp khoảng cách cực tiểu

Xét trường hợp tập đối tượng là dữ liệu trộn của k lớp với trung bình mẫu tương ứng là tập vecto $\{\mathbf{m}_i\}_{i=1,\dots,k}$ ($\mathbf{m}_i = \frac{1}{N_i} \sum_{x \in D_i} x$), $D = \bigcup_{i=1}^k D_i$. Khi đó có thể dùng các hàm phân biệt:

$$g_i(x) = -d(\mathbf{x}, \mathbf{m}_i) \quad (4.14)$$

trong đó $d(\mathbf{x}, \mathbf{y})$ là khoảng cách xác định bởi metric chọn trước. Theo phương pháp này, sau khi tính trung bình mẫu của mỗi lớp, ta lấy chúng là tâm lớp và mỗi đối

tương sẽ được xếp vào lớp mà nó gần tâm nhất. Vì vậy ta gọi là phân lớp theo phương pháp khoảng cách cực tiểu (*minimum distance*). Khi dùng metric Euclidean (4.15a) hoặc Mahalanobis (4.15e), các khoảng cách này xác định bởi tích vô hướng thì các hàm phân biệt bậc hai được đưa về phân biệt tuyến tính. Trước hết ta làm quen với các metric thông dụng trong không gian đặc trưng.

4.2.4.1. Các metric trong không gian đặc trưng.

Học phân biệt mẫu thường dựa trên độ đo sự tương tự giữa các mẫu, những mẫu giống nhau thường ở cùng một lớp. Độ đo tính tương tự này thường được xác định qua khoảng cách giữa chúng, các mẫu có khoảng cách càng bé thì càng tương tự nhau. Độ đo khoảng cách $d(\mathbf{x}, \mathbf{y})$ gọi là metric giữa \mathbf{x} và \mathbf{y} là hàm không âm d thỏa mãn các tính chất sau:

$$d(\mathbf{x}, \mathbf{y}) \geq 0, \quad \forall \mathbf{x}, \mathbf{y} \quad (4.15a)$$

$$d(\mathbf{x}, \mathbf{y}) = 0, \text{ khi và chỉ khi } \mathbf{x} = \mathbf{y} \quad (4.15b)$$

$$d(\mathbf{x}, \mathbf{y}) = d(\mathbf{y}, \mathbf{x}), \quad \forall \mathbf{x}, \mathbf{y} \quad (4.15c)$$

$$d(\mathbf{x}, \mathbf{y}) \leq d(\mathbf{x}, \mathbf{z}) + d(\mathbf{z}, \mathbf{y}), \quad \forall \mathbf{x}, \mathbf{y}, \mathbf{z} \quad (4.15d)$$

Khi không gian đặc trưng là không gian số học d -chiều và metric có tính chất:

$$d(a\mathbf{x}, a\mathbf{y}) = |a| d(\mathbf{x}, \mathbf{y}) \quad \forall a \in R \quad (4.15e)$$

thì metric này sinh ra chuẩn và $\|\mathbf{x} - \mathbf{y}\| = d(\mathbf{x}, \mathbf{y})$.

Sau đây là chuẩn thông dụng nhất:

$$\text{Chuẩn Euclidean: } \|\mathbf{x} - \mathbf{y}\|_e = \left(\sum_{j=1}^n |x_j - y_j|^2 \right)^{1/2} \quad (4.16a)$$

$$\text{Chuẩn Minkowski: } \|\mathbf{x} - \mathbf{y}\|_m = \left(\sum_{j=1}^n |x_j - y_j|^r \right)^{1/r}, \quad r \geq 1 \quad (4.16b)$$

$$\text{Chuẩn Manhattan: } \|\mathbf{x} - \mathbf{y}\|_{mi} = \sum_{j=1}^n |x_j - y_j| \quad (r=1) \quad (4.16c)$$

$$\text{Chuẩn Chebyshev: } \|\mathbf{x} - \mathbf{y}\|_t = \max \left| \sum_{j=1}^n |x_j - y_j| \right| \quad (r \rightarrow \infty) \quad (4.16d)$$

$$\text{Chuẩn Mahalanobis: } \|\mathbf{x} - \mathbf{y}\|_{ma} = (\mathbf{x} - \mathbf{y})^\top A (\mathbf{x} - \mathbf{y})^{1/2} \quad (4.16e)$$

trong đó A là ma trận đối xứng xác định dương. Một phương pháp hay dùng là A có dạng đường chéo xác định bởi trọng số của các thuộc tính:

$$\|x - y\|_{ma} = \left(\sum_{j=1}^n \rho_j^2 |x_j - y_j| \right)^{1/2} \quad (4.16f)$$

Khi x_i, y_i nhận giá trị của một thuộc tính định danh thì có thể xét metric rời rạc đối với thuộc tính này:

$$d(x_i, y_i) = \begin{cases} 1 & \forall x_i \neq y_i \\ 0 & \forall x_i = y_i \end{cases} \quad (4.16g)$$

$$\text{Metric sẽ là } d(\mathbf{x}, \mathbf{y}) = \sum_{i=1}^n \rho_i d(x_i, y_i) \quad (4.16h)$$

$$\text{hoặc } d(\mathbf{x}, \mathbf{y}) = \sqrt{\sum_{i=1}^n \rho_i^2 d^2(x_i, y_i)} \quad (4.16i)$$

trong đó $d(x_i, y_i) = |x_i - y_i|$ nếu là thuộc tính số hoặc xác định theo (4.16g) nếu là thuộc tính định danh. Ngoài ra, metric sau thường được dùng với các thuộc tính định danh khi giá trị thuộc tính là xâu ký hiệu có độ dài n :

$$d(\mathbf{x}, \mathbf{y}) = \frac{n-m}{n} \quad (4.16j)$$

trong đó m là số ký hiệu trùng nhau. Các metric cũng dùng để làm tên gọi khoảng cách tương ứng.

4.2.4.2. Phân biệt tuyến tính Euclidean

Ta trở lại với phân lớp khoảng cách cực tiểu khi sử dụng metric Euclidean, với hàm quyết định (4.14). Khi đó quy tắc quyết định là:

$$\mathbf{x} \in \omega_i \text{ nếu } -\|\mathbf{x} - \mathbf{m}_i\| > -\|\mathbf{x} - \mathbf{m}_j\| \quad \forall j \neq i \quad (4.17a)$$

$$\Leftrightarrow \|\mathbf{x}\|^2 + \|\mathbf{m}_i\|^2 - 2\mathbf{m}_i' \mathbf{x} > \|\mathbf{x}\|^2 + \|\mathbf{m}_j\|^2 - 2\mathbf{m}_j' \mathbf{x} \quad \forall j \neq i ,$$

$$\text{hay } \mathbf{x} \in \omega_i \text{ nếu } -\mathbf{m}_i' \mathbf{x} + 0,5 \|\mathbf{m}_i\|^2 > -2\mathbf{m}_j' \mathbf{x} + \|\mathbf{m}_j\|^2 \quad \forall j \neq i . \quad (4.17b)$$

Như vậy, mặc dù metric là hàm bậc hai nhưng ta đưa được về hàm phân biệt tuyến tính:

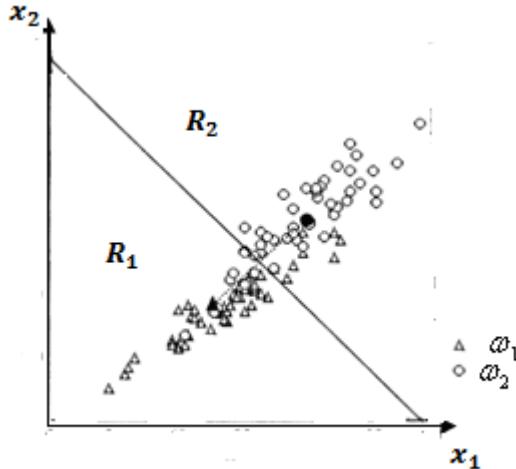
$$g_i(\mathbf{x}) = -\mathbf{m}_i' \mathbf{x} + 0,5 \|\mathbf{m}_i\|^2 \quad (4.17c)$$

và biên quyết định ứng với trường hợp hai lớp là:

$$(\mathbf{m}_1 - \mathbf{m}_2)' \mathbf{x} + 0,5 (\|\mathbf{m}_j\|^2 - \|\mathbf{m}_i\|^2) = 0. \quad (4.17d)$$

Biên quyết định là siêu phẳng trực giao với đoạn thẳng nối hai trung bình mẫu và đi qua điểm giữa của đoạn thẳng này.

Biên quyết định cho một ví dụ 2 lớp trong không gian đặc trưng hai chiều được minh họa trong hình 4.10.



Hình 4.10. Phân biệt tuyến tính Euclidean trong mặt phẳng

4.2.4.3. Phân biệt tuyến tính Mahalanobis

Ta xét tập dữ liệu $S = \{\mathbf{x}^i\} \subset R^n$ ($i=1, \dots, Q$), ma trận hiệp phương sai $C = (c_{i,j})$ được xác định bởi công thức sau:

$$c_{i,j} = \frac{1}{Q-1} \sum_{k=1}^Q (\mathbf{x}_k^i - \mathbf{m}_i)(\mathbf{x}_k^j - \mathbf{m}_j), \quad (4.18a)$$

Trường hợp k lớp và các lớp có ma trận hiệp phương sai C như nhau, khi đó nếu ta dùng metric Mahalanobis xác định bởi :

$$d^2(\mathbf{x}, \mathbf{y}) = (\mathbf{x} - \mathbf{y})' C^{-1} (\mathbf{x} - \mathbf{y}), \quad (4.18b)$$

thì metric này bất biến khi ta thay đổi đơn vị đo các đặc trưng. Khi đó hàm phân biệt khoảng cách cực tiểu của lớp ω_1 sẽ là

$$-d_i^2(\mathbf{x}) = -(\mathbf{x} - \mathbf{m}_i)' C^{-1} (\mathbf{x} - \mathbf{m}_i) \quad (4.18c)$$

Lập luận như với khoảng cách Euclidean, sau khi loại bỏ số hạng bậc hai, ta được hàm phân biệt tuyến tính:

$$g_i(\mathbf{x}) = w_i' \mathbf{x} + w_{i,0} \quad (4.19a)$$

$$\text{trong đó } w_i = C^{-1} \mathbf{m}_i; w_{i,0} = -0,5 \mathbf{m}_i' C^{-1} \mathbf{m}_1 \quad (4.19b)$$

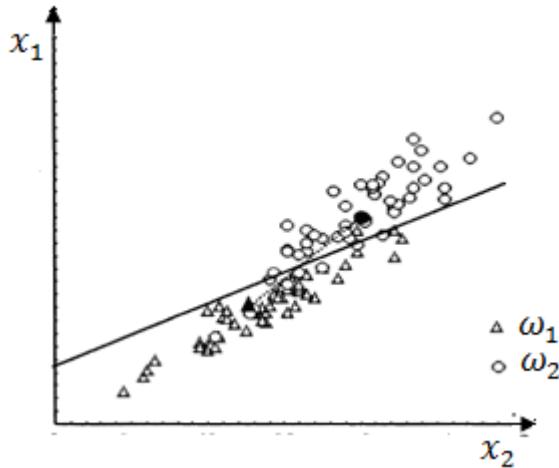
Trường hợp có 2 lớp, hàm quyết định sẽ là:

$$g(\mathbf{x}) = (\mathbf{m}_1 - \mathbf{m}_2)' C^{-1} \mathbf{x} - 0,5 (\mathbf{m}_1' C^{-1} \mathbf{m}_1 - \mathbf{m}_2' C^{-1} \mathbf{m}_2). \quad (4.19c)$$

Hàm phân biệt tuyến tính này cho siêu phẳng tách hai lớp đi qua điểm giữa đoạn nối các điểm trung bình nhưng khác với khoảng cách Euclidean là không trực giao với đường nối tâm mà trực giao với vectơ $C^{-1}(\mathbf{m}_1 - \mathbf{m}_2)$.

Phương pháp này được gọi là *phân biệt tuyến tính Mahalanobis*. Đặc biệt khi $C=s^2I$, bộ phân lớp Mahalanobis trùng với bộ phân lớp tuyến tính Euclidean. Trong thực hành, các ma trận hiệp phương sai thường không trùng nhau. Tuy nhiên nó không quá nhạy cảm với bài toán nên ta có thể dùng ma trận hiệp phương sai trung bình.

Ví dụ cho trong hình 4.10 khi xét metric Mahalanobis được minh họa trong hình 4.11 khi dùng khoảng cách Mahalanobis.



Hình 4.11. Phân biệt tuyến tính Mahalanobis trong R^2

Chú ý. Khi dùng metric Mahalanobis như trong công thức (4.16e) với ma trận A tùy ý thì ta đều tìm được phân biệt tuyến tính.

4.2.5. Máy véctơ tựa

Khi các lớp tách được tuyến tính thì có nhiều hàm phân biệt có thể tách chúng. Liệu có thể tìm một hàm phân biệt tối ưu hay không? Phương pháp máy vecto tựa ((Support vector machine: SVM) cho ta một tiếp cận tối ưu để tìm siêu phẳng tách. Phương pháp này có thể dùng cho cả các trường hợp không tách được tuyến tính và tách ra rất hiệu quả trong ứng dụng. Trước hết ta xét trường hợp các lớp trong tập mẫu tách được tuyến tính.

4.2.5.1. Các lớp tách được tuyến tính

Để đơn giản, ta sẽ xét trường hợp có 2 lớp, tập mẫu là $D = \{(\mathbf{x}^t, y^t) / t=1,..N\}$ trong đó $y^t = +1$ nếu $\mathbf{x}^t \in \omega_1$ và $y^t = -1$ nếu $\mathbf{x}^t \in \omega_2$. Bài toán được đặt ra là tìm siêu phẳng tách "tốt nhất" cho hai lớp ω_1 và ω_2 cho bởi hàm phân biệt $g(\mathbf{x}) = \mathbf{w}^T \mathbf{x} + w_0$ sao cho:

$$g(\mathbf{x}^t) = \mathbf{w}^T \mathbf{x}^t + w_0 \geq +1 \text{ khi } y^t = +1 \quad (4.20a)$$

$$g(\mathbf{x}^t) = \mathbf{w}^T \mathbf{x}^t + w_0 \leq -1 \text{ khi } y^t = -1.$$

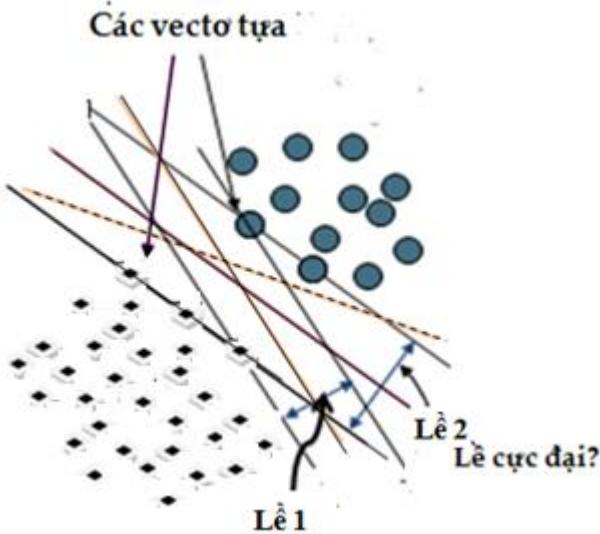
Các điều kiện trong các bất đẳng thức (4.20a) có thể lại là

$$y^t(\mathbf{w}^T \mathbf{x}^t + w_0) \geq 1. \quad (4.20b)$$

Lưu ý rằng ở đây không chỉ yêu cầu đơn giản $y^t(\mathbf{w}^T \mathbf{x}^t + w_0) \geq 0$ như mà trước mà muốn chúng có khoảng cách tới siêu phẳng tách *tốt hơn*. Tiêu chuẩn "tốt nhất" được xác định dựa trên khái niệm lề của siêu phẳng tách.

Lề của siêu phẳng tách

Giả sử ta có một siêu phẳng tách hai tập mẫu quan sát được của hai lớp đang xét. Ta tính tiến song song siêu phẳng này về phía tập mẫu của mỗi lớp cho đến khi nó *tựa* vào tập này, tức là có ít nhất một điểm phuộc siêu phẳng và không thể tịnh tiến thêm nữa thì ta gọi siêu phẳng này là *siêu phẳng tựa (support)* hay *siêu phẳng lề* của lớp tương ứng, các vecto \mathbf{x}^t thuộc các siêu phẳng này gọi là các vecto tựa hay vecto *lề*, *hành lang* nằm giữa hai siêu phẳng tựa gọi là *miền lề* và khoảng cách giữa hai siêu phẳng này gọi là *lề* của siêu phẳng tách. Siêu phẳng tách tốt nhất là siêu phẳng có lề cực đại, phương pháp tìm nó ta gọi là phương pháp máy vecto tựa (SVM). Hình 4.12 minh họa trực quan lề, vecto tựa và siêu phẳng tựa của hai siêu phẳng tách (là siêu phẳng cách đều hai siêu phẳng tựa tương ứng), trong đó lề 2 tốt hơn lề 1 còn siêu phẳng tách biểu diễn bởi đường gạch đứt không xét lề.



Hình 4.12. Lề và siêu phẳng tựa: lề 2 tốt hơn lèle 1

Khoảng cách từ điểm x^t tới siêu phẳng tách được tính theo công thức (4.4c) là $\frac{g(x^t)}{\|\mathbf{w}\|}$ nên ta có thể co giãn \mathbf{w} và w_0 để $g(x^t)$ tại các vecto tựa bằng 1 nếu nó thuộc lớp ω_1 và bằng -1 nếu nó thuộc lớp ω_2 . Từ (4.20a) ta có:

$$\text{Lề của siêu phẳng} = \frac{1}{\|\mathbf{w}\|} + \frac{1}{\|\mathbf{w}\|} = \frac{2}{\|\mathbf{w}\|} \quad (4.21a)$$

$$\text{trong đó } \mathbf{w}, w_0 \text{ thỏa mãn: } \mathbf{w}^T \mathbf{x}^t + w_0 \geq +1 \quad \forall \mathbf{x}^t \in \omega_1 \quad (4.21b)$$

$$\text{và } \mathbf{w}^T \mathbf{x}^t + w_0 \leq -1 \quad \forall \mathbf{x}^t \in \omega_1. \quad (4.21c)$$

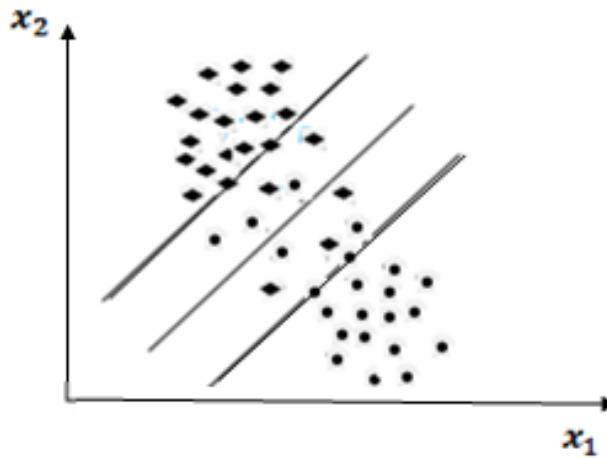
Bởi vì $y^t = +1$ nếu $\mathbf{x}^t \in \omega_1$ và $y^t = -1$ nếu $\mathbf{x}^t \in \omega_2$ nên hệ trên đưa về tìm \mathbf{w}, w_0 trong bài toán:

$$\text{Cực tiểu hàm } J(\mathbf{w}) = \frac{\|\mathbf{w}\|^2}{2}$$

với ràng buộc $y^t(\mathbf{w}^T \mathbf{x}^t + w_0) \geq 1 \quad \forall t=1, \dots, N$. Bài toán quy hoạch bậc hai này có nghiệm duy nhất và luôn tìm được nhờ thuật toán có độ phức tạp $O(N^3)$. Trong trường hợp có k lớp ta có thể xét k bài toán 2 lớp, mỗi bài toán tách một lớp với các lớp khác.

4.2.5.2. Các lớp không tách được tuyế́n tính

Phương pháp trên dựa trên khái niệm lề và các lớp tách được tuyế́n tính nên khi các lớp không tách được tuyế́n tính thì không áp dụng ngay được. Để mở rộng miền áp dụng, ta chấp nhận trong miền lề có lỗi nhưng ngoài miền lề phải phân lớp đúng. Đối với các bài toán này, ta xét hàm phân biệt với lề là các điểm $y^t(\mathbf{w}^T \mathbf{x}^t + w_0) = 1$ sao cho các điểm ngoài miền lề được phân lớp đúng như minh họa trong hình 4.13.



Hình 4.13. Trường hợp không tách được tuyế́n tính

Người ta sử dụng lề mềm nhò thêm biến chùng ξ biểu thị mức lệch $\xi^t \geq 0$ của điểm \mathbf{x}^t tới siêu phẳng lề:

$$y^t(\mathbf{w}^T \mathbf{x}^t + w_0) \geq 1 - \xi^t \quad \forall t=1, \dots, N; \quad (4.22a)$$

trong đó $\xi^t = 0$ nếu \mathbf{x}^t ở ngoài miền lề:

$$y^t(\mathbf{w}^T \mathbf{x}^t + w_0) \geq 1, \quad (4.22b)$$

$\xi^t \in (0,1)$ nếu \mathbf{x}^t ở trong miền lề và phân lớp đúng:

$$0 \leq y^t(\mathbf{w}^T \mathbf{x}^t + w_0) \leq 1, \quad (4.22c)$$

$\xi^t > 1$ nếu phân lớp sai:

$$y^t(\mathbf{w}^T \mathbf{x}^t + w_0) < 0. \quad (4.22d)$$

Ta muốn tìm hàm quyết định có lề lớn nhất và số điểm có $\xi > 0$ nhỏ nhất. Việc này dẫn tới bài toán quy hoạch lồi sau để tìm \mathbf{w} và w_0 :

$$\text{Cực tiểu hàm} \quad J(\mathbf{w}, w_0, \xi) = \frac{\|\mathbf{w}\|^2}{2} + C \sum_t \xi^t \quad (4.23a)$$

$$\text{với các ràng buộc: } y^t(\mathbf{w}^T \mathbf{x}^t + w_0) \geq 1 - \xi^t \quad \forall (\mathbf{x}^t, y^t) \in D, \quad (4.23b)$$

$$\xi^t \geq 0 \quad \forall t=1, \dots, N; \quad (4.23c)$$

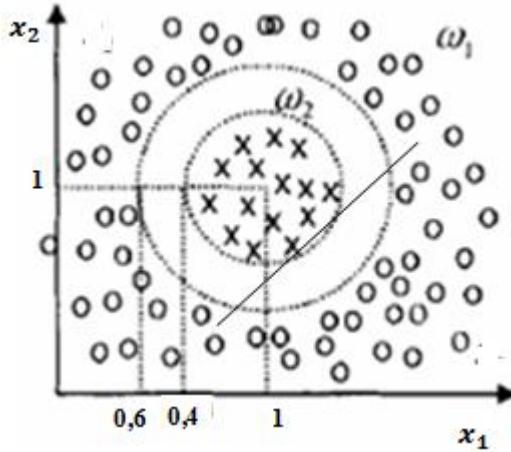
trong đó C là hằng số dương biểu thị mức phạt các điểm phân lớp sai, trường hợp tách được tuyếen tính ứng với $C \rightarrow \infty$.

4.2.5.3. Hàm nhân (Kernel function)

Nhiều bài toán ta không áp dụng được phương pháp SVM nhưng khi nhúng các đối tượng vào không gian khác bằng một ánh xạ phi tuyến thì có thể áp dụng được. Chẳng hạn, hai lớp trong hình 4.14 sẽ tách được tuyếen tính khi dùng phép biến đổi không gian đặc trưng $\Phi: X \rightarrow Z$:

$$\Phi(x_1, x_2) = (z_1, z_2) = ((x_1 - 1)^2, (x_2 - 1)^2) \quad (4.24)$$

thì tập mẫu quan sát được sẽ tách được tuyếen tính bởi hàm $g(z) = z_1 + z_2 - 0,25$, khi kết hợp hàm hợp ta có hàm phân biệt $g(x) = (x_1 - 1)^2 + (x_2 - 1)^2 - 0.25$



Hình 4.14. Hàm $g(x) = (x_1 - 1)^2 + (x_2 - 1)^2 - 0.25$ phân biệt hai lớp

Dựa trên các ví dụ như thế, thay cho không gian đặc trưng gốc, người ta có thể dùng ánh xạ $\Phi: X \rightarrow Z$ để nhúng nó lên không gian Z (thường có chiều lớn hơn) để xác định hàm phân biệt dễ hơn hoặc có dạng đơn giản hơn.

Giả sử tập mẫu quan sát được trong \mathbf{X} là $\{(\mathbf{x}^t, \mathbf{y}^t)\}_{t=1}^N$, ký hiệu \mathbf{z}^t tương ứng là ảnh của \mathbf{x}^t trong không gian \mathbf{Z} . Khi đó vectơ hệ số \mathbf{w} của hàm phân biệt tuyến tính trong \mathbf{Z} là tổ hợp tuyến tính của $\{\mathbf{z}^t\}_{t=1}^N$:

$$\mathbf{w} = \sum_t a_t \mathbf{z}^t = \sum_t a_t \Phi(\mathbf{x}^t) \quad (4.25a)$$

và hàm phân biệt là

$$g(\mathbf{x}) = \mathbf{w}' \Phi(\mathbf{x}) = \sum_t a_t \Phi(\mathbf{x}^t) \Phi(\mathbf{x}). \quad (4.25b)$$

Để không phải biểu diễn tường minh hàm Φ khi tính các tích trong $\Phi(\mathbf{x}^t) \Phi(\mathbf{x})$ người ta sử dụng hàm cơ sở gọi là hàm nhân $K(\mathbf{x}^t, \mathbf{x})$, khi đó hàm phân biệt sẽ là:

$$g(\mathbf{x}) = \sum_t a_t K(\mathbf{x}^t, \mathbf{x}). \quad (4.25c)$$

Phương pháp sử dụng hàm nhân để tìm hàm phân biệt gọi là phương pháp hàm nhân. Sau đây là các hàm nhân thông dụng.

- *Đa thức bậc p:*

$$K(\mathbf{x}^t, \mathbf{x}) = (\mathbf{x}^T \mathbf{x}^t + 1)^p$$

trong đó \mathbf{x}^T là ký hiệu vecto chuyển vị của \mathbf{x} . Nói riêng, khi $p=2$ và không gian đặc trưng của \mathbf{X} 2 chiều ta có:

$$\begin{aligned} K(\mathbf{x}, \mathbf{y}) &= (\mathbf{x}^T \mathbf{y} + 1)^2 (x_1 y_1 + x_2 y_2 + 1)^2 \\ &= 1 + 2x_1 y_1 + 2x_2 y_2 + 2x_1 y_1 x_2 y_2 + x_1^2 y_1^2 + x_2^2 y_2^2, \end{aligned}$$

tương ứng với $\Phi(\mathbf{x}) = (1, \sqrt{2}x_1, \sqrt{2}x_2, \sqrt{2}x_1 x_2, x_1^2, x_2^2)$.

- *Hàm cơ sở bán kính (radial basis function):*

$$K(\mathbf{x}^t, \mathbf{x}) = \exp \left[-\frac{\|\mathbf{x}^t - \mathbf{x}\|^2}{\sigma^2} \right].$$

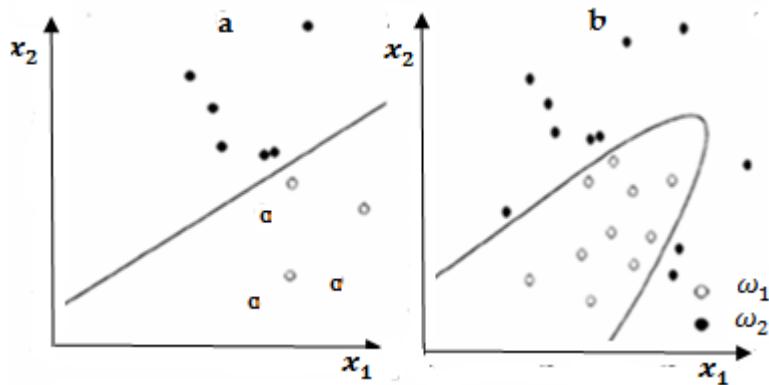
Ta sẽ trở lại với hàm này trong chương 7 và chương 8.

4.3. BÀI TOÁN TỶ LỆ CHIỀU

Ta gọi tỷ lệ $\frac{N}{d}$ giữa số lượng mẫu dữ liệu N và số đặc trưng d của dữ liệu là tỷ lệ chiều. Nói chung, tỷ lệ chiều nhỏ không phản ánh đúng cấu trúc lớp nhưng nhiều như thế nào là đủ thì vẫn chưa có câu trả lời. Một vấn đề đặt ra trong phân lớp có giám sát là có bao nhiêu mẫu quan sát thì đủ để xây dựng bộ phân lớp trong không gian đặc trưng d chiều? Câu hỏi này tới nay vẫn chưa có thời giải thỏa đáng và được gọi là vẫn đề tỷ lệ chiều. Chẳng hạn, xét ví dụ về xét bài toán 2 lớp với dữ liệu hai chiều, từng biến mỗi lớp có phân bố chuẩn:

Lớp 1	x_1 có phân bố $N(1; 1)$	x_2 có phân bố $N(1; 1)$
Lớp 2	x_1 có phân bố $N(1,1; 1)$	x_2 có phân bố $N(1,2; 2)$

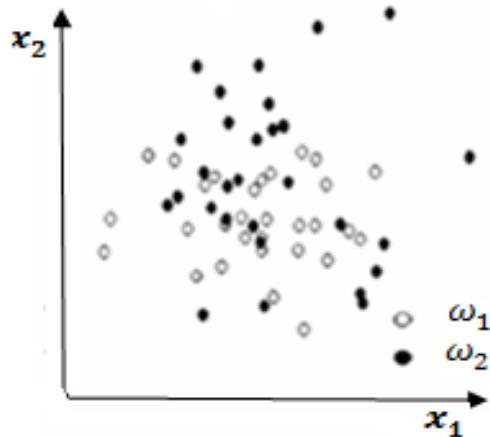
Ban đầu ta có $n = 6$ mẫu cho mỗi lớp như trong hình 4.15a($N/d=6$), trực quan hình học cho ta thấy hai lớp này tách được tuyến tính mà không có lỗi. Sau đó ta thu thập thêm dữ liệu cho mỗi lớp là 10 ($N/d=10$) như trong hình 4.15b.



Hình 4.15. Hai lớp tách được khi tỷ lệ chiều thấp: a) $N/d = 6$; b) $N/d=10$

Cảm giác trực quan cho thấy bây giờ chúng không thể tách tuyến tính được nhưng hình như hàm quyết định bậc hai có thể tách chúng.

Khi ta tăng lên 30 mẫu mỗi lớp tức là tỷ lệ chiều $N/d=30$ như minh họa trong hình 4.16 thì thấy chúng không thể tách được bằng một hàm phân biệt đơn giản.



Hình 4.16. Phân bố hai lớp với tỷ lệ chiều $N/d = 30$

Khi số chiều lớn, người ta có thể dùng công cụ thống kê để phân tích các thành phần độc lập nhằm loại bỏ các đặc trưng phụ thuộc nhau nhiều hoặc dùng phương pháp phân tích thành phần chính để giảm chiều dữ liệu. Ta sẽ trở lại với chủ đề này ở chương sau.

KẾT LUẬN

Trong các hệ phân lớp có giám sát, người ta thường dùng hàm phân biệt để xác định lớp, các hàm này phân không gian đặc trưng thành các miền gọi là miền quyết định. Đối tượng x sẽ thuộc miền R_i của lớp ω_i khi hàm của nó nhận giá trị lớn nhất so với các hàm quyết định khác.

Hàm phân biệt tuyến tính là dạng đơn giản và dễ xác định trong các trường hợp ứng dụng được nhờ nền tảng toán học của đại số tuyến tính. Phương pháp học *Perceptron* chỉ áp dụng được khi các lớp tách được tuyến tính, khi đó thuật toán luôn kết thúc sau hữu hạn bước. Thuật toán Widrow-Hoff tìm phân biệt tuyến tính theo phương pháp bình phương tối thiểu nhưng khi áp dụng thì huấn luyện tuần tự từng mẫu quan sát một.

Máy vecto tựa (SVM) là một tiếp cận để tìm siêu phẳng tách tối ưu và có thể dùng cả cho trường hợp không tách được tuyến tính. Phương pháp này tỏ ra hiệu quả trong ứng dụng thực tiễn. Nhiều trường hợp, người ta nhúng không gian đặc trưng lên không gian lớn hơn, nhờ đó mà tập mẫu có thể tách được tuyến tính. Hàm nhân cho phép ta dễ dàng xác định các tích trong mà không cần xác định tường minh ánh xạ nhúng.

Tỷ lệ chiều nhỏ thường cho ta cảm nhận sai về phân bố lớp, tuy nhiên tỷ lệ này bao nhiêu là đủ để có nhận thức đúng về các lớp vẫn là câu hỏi khó.

BÀI TẬP

1. Một bộ phân lớp dùng hàm phân biệt tuyến tính trong R^3 :

$$g(x) = 2x_1 + x_2 + x_3 + 1$$

- a) Tính khoảng cách từ gốc tọa độ đến biên quyết định
 - b) Tính khoảng cách từ vecto đặc trưng $x = [-1, 1, 0]'$ đến biên quyết định
 - c) Cho ví dụ có mẫu nằm trên biên quyết định
2. Nếu không gian đặc trưng là 3 chiều và dùng hàm phân biệt là đa thức bậc 3 thì cần xác định bao nhiêu hệ số của đa thức?
 3. Cho ví dụ một tập gồm 30 dữ liệu của 3 lớp có phân bố chuẩn 2 chiều.

- a) Kiểm tra xem các tập dữ liệu có phân biệt tuyến tính được không? Nếu có thì tách được tuyệt đối hay tương đối?
- b) Áp dụng thuật toán perceptron để phân biệt lớp..
4. Phác họa các hình cầu đơn vị cho các metric đã biết.
5. Cho các điểm thuộc 2 lớp có tập quan sát được cho mỗi lớp tương ứng:
- $$S_1 = \{(0; 0)', (0,5; 0)', (0; 1)', (-1,5; 0)', (0; 2)'\}$$
- $$S_2 = \{(1; 1)', (0; 1)', (1; 0)', (2; 1)', (1; 2)'\}$$
- a) Tính ma trận hiệp phương sai của S_1, S_2 và $S = S_1 \cup S_2$
- b) Xác định tâm của S_1 và S_2 và tạo ra tập mẫu bổ sung với phân bố chuẩn để mỗi lớp có 12 đối tượng. Dùng hai phương pháp khoảng cách cực tiểu đã biết phân lớp và xác định tỷ lệ phân lớp đúng tương ứng với mỗi phương pháp.
6. Tao tập mẫu gồm 2 lớp tương tự như trong mục 4.3 nhưng với tâm và tham số phân bố chuẩn khác, sau đó kiểm tra khi N bằng bao nhiêu thì các lớp không tách được tuyến tính.
7. Cho tập dữ liệu quan sát trong R^2 gồm 3 lớp :
- $$S_1 = \{(0,0), (1,0), (0,1)\}$$
- $$S_2 = \{(1,1), (2,1)\}$$
- $$S_3 = \{(-1,0), (-1,-1), (1,-2), (1,-1)\}$$
- a) Áp dụng thuật toán học perceptron để phân biệt S_1 và S_2 với khởi tạo là:
 $\mathbf{w} = (1,1)^T$ và $w_0 = 0$
- b) Chỉ ra các hàm phân biệt cho mỗi lớp khi dùng phương pháp khoảng cách cực tiểu theo khoảng cách Euclidean
- c) Chỉ ra các hàm phân biệt cho mỗi lớp khi dùng phương pháp khoảng cách cực tiểu theo khoảng cách Mahalanobis có ma trận A là $\begin{pmatrix} 2 & 1 \\ 1 & 2 \end{pmatrix}$.
8. Xây dựng một ví dụ nhò sử dụng Matlab để minh họa các phương pháp tách được tuyến tính.

Chương 5

HỌC THỐNG KÊ

Học thống kê bao gồm một lớp rộng các phương pháp dựa trên mô hình xác suất đang được dùng rộng rãi để phân tích tự động dữ liệu. Chương này giới thiệu một số phương pháp cơ bản áp dụng cho bài toán học có giám sát. Bắt đầu từ lý thuyết tổng quát về quyết định Bayes sau đó ứng dụng cho bài toán phân lớp, tiếp theo, lần lượt giới thiệu phương pháp phân lớp k-láng giềng gần nhất, vấn đề chọn đặc trưng và đánh giá chất lượng các bộ phân lớp.

5.1. LÝ THUYẾT QUYẾT ĐỊNH BAYES

5.1.1. Bài toán và các quy tắc quyết định

Ta xét bài toán học có giám sát, trong đó đã biết xác suất xảy ra của mỗi giả thuyết trong không gian giả thuyết H nào đó và ước lượng được xác suất xuất hiện của tập dữ liệu quan sát được khi mỗi giả thuyết xảy ra. Khi đó ta cần dự đoán giả thuyết H cho ta dữ liệu quan sát này. Trước hết ta xét ví dụ sau.

Ví dụ 1

Trong một hộp có chứa 8 đồng tiền không đồng chất với xác suất ngửa (N) khi tung ngẫu nhiên khác nhau. Lấy ngẫu nhiên một đồng tiền trong hộp và tung ngẫu nhiên 20 lần, kết quả quan sát có 6 lần được mặt ngửa và 14 lần mặt sấp (S), lần lượt là:

N, S, S, N, N, S, S, S, S, H, S, S, N, S, N, S, S, N, S, S.

Ta cần đoán xem đồng tiền này thuộc loại nào nếu biết rằng:

a) Xác suất ngửa của 8 đồng tiền trong hộp tương ứng là:

$$P(N) = \frac{1}{3}, \frac{1}{2}, \frac{1}{2}, \frac{1}{2}, \frac{1}{2}, \frac{1}{2}, \frac{1}{2}, \frac{2}{3}. \quad (5.1a)$$

b) Chỉ biết trong hộp có 3 loại tiền với xác suất ngửa tương ứng là:

$$P(N) = \frac{1}{3}, \frac{1}{2}, \frac{2}{3}. \quad (5.1b)$$

Bài toán này được tổng quát hóa như sau.

Bài toán tổng quát

Cho không gian giả thuyết $H = \{h_1, \dots, h_N\}$ và dữ liệu quan sát được D . Giả sử ta đã biết được các xác suất $P(h_i)$ để giả thuyết h_i xảy ra khi chưa có tập dữ liệu quan sát và các xác suất có điều kiện $P(D|h_i)$ xuất hiện D khi giả thuyết h_i xảy ra. Dựa trên các xác suất tiền nghiệm đã biết này, ta cần đoán nhận giả thuyết h nào xảy ra.

Trong ví dụ trên, ta có ba giả thuyết $\{h_1, h_2, h_3\}$ tương ứng với ba trường hợp đồng tiền có xác suất ngửa là $\frac{1}{3}, \frac{1}{2}, \frac{2}{3}$ còn các xác suất $P(D|h_i)$ được tính theo công thức Bernoulli:

$$P(D|h_i) = C_{20}^6(\theta)^6(1-\theta)^{14}, \quad (5.1c)$$

trong đó $\theta \in \left\{\frac{1}{3}, \frac{1}{2}, \frac{2}{3}\right\}$ tương ứng với mỗi giả thuyết.

Quy tắc xác suất hậu nghiệm cực đại (MAP)

Để trả lời câu hỏi nêu trên, ta tính các xác suất hậu nghiệm $P(h_i|D)$: xác suất xảy ra h_i với điều kiện D . Giả thuyết được dự đoán xảy ra là giả thuyết có xác suất hậu nghiệm cực đại, tức là $h = h_{MAP}$:

$$h_{MAP} = \arg \max \mathbb{P}(h_j | D) : h_j \in H \quad (5.2)$$

Để tính các xác suất hậu nghiệm, ta sử dụng công thức Bayes cho hai sự kiện A và B :

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}. \quad (5.3a)$$

Khi A là giả thuyết h_i trong H và B là quan sát D ta có:

$$P(h_i | D) = \frac{P(D|h_i)P(h_i)}{P(D)}, \quad (5.3b)$$

trong đó $P(D)$ là xác suất xuất hiện quan sát D . Vì mẫu số trong vế phải của (5.3b) như nhau và không phụ thuộc vào các giả thuyết cụ thể nên giả thuyết tốt nhất là giả thuyết có tử số đạt cực đại.

$$\begin{aligned} h_{MAP} &= \arg \max \mathbb{P}(h_i | D) : h_i \in H \xrightarrow{\text{vì}} \arg \max \left\{ \frac{P(D|h_i)P(h_i)}{P(D)} : h_i \in H \right\} \\ &= \arg \max \mathbb{P}(h_i | D)P(h_i) : h_i \in H \end{aligned} \quad (5.3c)$$

Quy tắc (5.3c) còn được gọi là quy tắc quyết định **Bayes** vì nó nhận được từ trên công thức xác suất Bayes.

Trở lại với câu hỏi a) của ví dụ 1. Ta dễ dàng tính được:

$$P(h_1) = \frac{1}{8}; P(h_2) = \frac{6}{8}; P(h_3) = \frac{1}{8}$$

$$P(D/h_1) = C_{20}^6 \left(\frac{1}{3}\right)^6 \left(\frac{2}{3}\right)^{14} = 0,18213$$

$$P(D/h_2) = C_{20}^6 \left(\frac{1}{2}\right)^{20} = 0,03696$$

$$P(D/h_3) = C_{20}^6 \left(\frac{2}{3}\right)^6 \left(\frac{1}{3}\right)^{14} = 0,00071$$

và $P(h_1/D) = 0,45015;$

$$P(h_2/D) = 0,54810;$$

$$P(h_3/D) = 0,00175.$$

Vậy h_2 là giả thuyết MAP.

Quy tắc giả thuyết có khả năng nhất (ML)

Khi không có thông tin về xác suất đúng của các giả thuyết trong H , ta giả thiết rằng mọi giả thuyết h thuộc H có cùng xác suất tiền nghiệm, tức là:

$$P(h_i) = P(h_j) \quad \forall i, j. \quad (5.4a)$$

Khi đó công thức (5.3c) cho thấy giả thuyết h có $P(D/h)$ cực đại cũng là giả thuyết có xác suất hậu nghiệm cực đại và được gọi *giả thuyết có khả năng nhất* (maximum likelihood) hay *hợp lý nhất* và được ký hiệu là h_{ML} :

$$h_{ML} = \arg \max h(D/h) : h \in H \quad (5.4b)$$

Phương pháp này được gọi là phương pháp hợp lý nhất/có khả năng nhất. Trong câu hỏi b) của ví dụ 1 ở trên, giả thuyết h_1 là giả thuyết có khả năng nhất.

Trước khi xét bài toán phân lớp, ta xét một ứng dụng lý thuyết của phương pháp này cho bài toán học khái niệm.

5.1.2. Quyết định Bayes trong học khái niệm

Trở lại với bài toán học khái niệm đã được giới thiệu trong chương 2. Giả sử hệ học quan tâm đến một tập hữu hạn các giả thuyết H xác định trên không gian mẫu X , nhiệm vụ là học một khái niệm đích $c: X \rightarrow \{0,1\}$ dựa trên tập mẫu quan sát được $\{(x_1, d_1), \dots, (x_m, d_m)\}$, trong đó các x_i là mẫu trong X và d_i là giá trị hàm mục tiêu của x_i , tức là $d_i = c(x_i)$. Để đơn giản, ta xem dãy (x_1, \dots, x_m) là cố định và tập dữ liệu D sẽ được viết qua tập giá trị đích $D = (d_1, \dots, d_m)$.

Nếu ta có tri thức tiền nghiệm $P(h)$ và $P(D/h)$ đối với mọi giả thuyết h trong H thì phương pháp MAP cho ta thuật toán học cưỡng bức để tìm giả thuyết h_{MAP} được mô tả trong bảng 5.1. Thuật toán này không có ý nghĩa thực tiễn vì khi không gian giả thuyết lớn, việc áp dụng công thức Bayes cho mọi giả thuyết h trong H để tính $P(h/D)$ là không khả thi.

Bảng 5.1. Thuật toán học MAP cưỡng bức (brute-force)

Bước 1. Với mỗi h trong H , tính xác suất hậu nghiệm:



(5.5a)

Bước 2. Chọn h_{MAP} với xác suất hậu nghiệm cao nhất.

$$h_{MAP} = \arg \max h(D/h : h \in H) \quad (5.5b)$$

Tuy nhiên thuật toán vẫn còn có ý nghĩa vì nó cho ta một tiêu chuẩn để đánh giá hiệu suất các thuật toán học khái niệm.

Để đơn giản, ta giả thiết tập dữ liệu quan sát được và không gian giả thuyết thỏa mãn các điều kiện sau:

1. Tập dữ liệu huấn luyện D là không có nhiễu (noise free) tức là $d_i = c(x_i)$ với mọi i .
2. Hàm đích c có trong H
3. Chúng ta không có tri thức nào để cho là giả thuyết này có khả năng hơn giả thuyết khác, tức là $P(h_i) = P(h_j) \forall h_i, h_j \in H$ (khả năng mọi giả thuyết như nhau).

Khi đó, do khả năng mọi giả thuyết như nhau nên $P(h) = \frac{1}{|H|}$, trong đó $|H|$ là ký hiệu số phần tử trong H ; vì tập D không nhiễu nên $P(D/h) = 1$ nếu D phù hợp h và $P(D/h) = 0$ nếu ngược lại, hay là:

$$P(D/h) = \begin{cases} 1: & d_i = h(x_i) \forall d_i \in D \\ 0: & \text{khác} \end{cases} \quad (5.5c)$$

Trước hết, ta tính $P(D)$ theo công thức xác suất đầy đủ:

$$P(D) = \sum_{h_i \in H} P(D/h_i)P(h_i) = \sum_{h_i \in VS_{HD}} 1 \cdot \frac{1}{|H|} + \sum_{h_i \notin VS_{HD}} 0 \cdot \frac{1}{|H|} = \frac{|VS_{HD}|}{|H|}, \quad (5.5d)$$

trong đó VS_{HD} là không gian tường thuật đối với không gian giả thuyết H và tập dữ liệu đào tạo D .

Bây giờ ta xét bước 1 của thuật toán: dùng công thức Bayes để tính xác suất hậu nghiệm $P(h/D)$. Từ (5.5a) và (5.5c) ta có :

- Nếu h không phù hợp với D thì $P(h/D) = \frac{P(D/h)P(h)}{P(D)} = \frac{0 \cdot P(h)}{P(D)} = 0$.

- Ngược lại, nếu h phù hợp với D thì $P(D/h)=1$ theo (5.5c), chú ý tới (5.4d) ta có:

$$P(D) = \frac{1}{|H|} \cdot \frac{1}{|VS_{HD}|} \cdot \frac{1}{|VS_{HD}|} = \frac{1}{|H|}. \quad (5.5e)$$

Tóm lại, công thức Bayes cho xác suất hậu nghiệm:

$$P(h/D) = \begin{cases} \frac{1}{|VS_{HD}|} & \forall h \in VS_{HD} \\ 0 & \forall h \notin VS_{HD} \end{cases} \quad (5.6)$$

Như vậy, với giả thiết nêu trên về phân bố $P(h)$ và $P(D/h)$ ta thấy các giả thuyết phù hợp với D là giả thuyết MAP.

5.1.3. Giả thuyết có khả năng nhất và sai số bình phương tối thiểu

Trong phần này ta quan tâm đến bài toán học một hàm đích có giá trị liên tục. Phân tích Bayesian cho thấy rằng với một số giả thiết, thuật toán bình phương tối thiểu sẽ đưa ra một giả thuyết có khả năng nhất.

Xét hệ học L trên không gian mẫu X và không gian các giả thuyết H bao gồm một lớp các hàm giá trị thực xác định trên X (với mỗi $h \in H$, $h: X \rightarrow R$; R là tập số thực). Nhiệm vụ của hệ học L là phải học một hàm đích chưa biết $f: X \rightarrow R$ trong H . Ta xét tập m dữ liệu huấn luyện D với giá trị đích có nhiều trắcn phán bố chuẩn, tức là, mỗi mẫu đào tạo có dạng (x_i, d_i) với $d_i = f(x_i) + e_i$. Trong đó $f(x_i)$ là một giá trị không nhiễu của hàm đích, e_i là một biến ngẫu nhiên độc lập có phân phối chuẩn với kỳ vọng bằng 0 và phương sai σ^2 chưa biết. Ta giả thiết xác suất tiền nghiệm của mọi giả thuyết như nhau và tìm giả thuyết hợp lý nhất h_{ML} . Phân tích toán học sẽ chỉ ra rằng với một số giả thiết, phương pháp bình phương tối thiểu (LMS) cho ta giả thuyết h_{ML} .

Trước khi chỉ ra tại sao một giả thuyết LMS lại là một giả thuyết có khả năng nhất ta nhắc lại hai khái niệm cơ bản của lý thuyết xác suất: phân phối chuẩn và mật độ xác suất

Ta dùng chữ p viết thường để chỉ hàm mật độ xác suất để phân biệt với xác suất P viết hoa. Mật độ xác suất $p(x_0)$ là giới hạn:

$$p(x_0) = \lim_{\varepsilon \rightarrow 0} \frac{1}{\varepsilon} P(x_0 \leq x < x_0 + \varepsilon).$$

Phân bố chuẩn $N(\mu, \sigma)$ có hàm mật độ là: $\frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{1}{2\sigma^2}(x-\mu)^2}$

Ta sẽ chỉ ra rằng giả thuyết LMS là giả thuyết có khả năng nhất. Ta bắt đầu với công thức (5.4b) nhưng dùng với hàm mật độ

$$h_{ML} = \arg \max_{h \in H} p(D/h),$$

Trong đó tập mẫu huấn luyện $D=\{d_1, \dots, d_m\}$ cho giá trị tại tập mốc cố định $\{x_1, \dots, x_m\}$ với $d_i = f(x_i) + e_i$ như đã nói ở trên. Giả sử rằng đối với h đã cho, các mẫu độc lập nhau, khi đó $p(D/h)$ viết được dưới dạng tích:

$$h_{ML} = \arg \max \prod_{i=1}^m p(d_i/h)$$

Khi đó mỗi d_i cũng tuân theo phân bố chuẩn với phương sai σ^2 với tâm là giá trị đích $f(x_i)$ khác không. Vì vậy $p(d_i/h)$ với h đã cho có thể được viết như một phân phối chuẩn với phương sai là σ^2 ; kỳ vọng là $\mu = f(x_i)$ và ta có:



Dùng biến đổi logarit để tìm h_{ML} ta suy ra:



$$\begin{aligned} &\Leftrightarrow h_{ML} = \arg \max \sum_{i=1}^m -\frac{1}{2\sigma^2} (d_i - h(x_i))^2 = \arg \min \sum_{i=1}^m \frac{1}{2\sigma^2} (d_i - h(x_i))^2 \\ &= \arg \min \sum_{i=1}^m (d_i - h(x_i))^2 \end{aligned} \quad (5.7)$$

Công thức (5.7) cho thấy rằng giả thuyết có khả năng nhất h_{ML} làm cực tiểu tổng bình phương sai số của các giá trị huấn luyện quan sát được d_i với giá trị giả thuyết dự đoán $h(x_i)$.

Bằng cách này ta sẽ thấy thuật toán huấn luyện mạng MLP trong chương 7 là tìm giả thuyết khả năng nhất khi giá trị của mẫu có nhiều tráng dạng phân bố chuẩn.

5.2. PHÂN LỐP BAYES

Phân lớp Bayes là ứng dụng thường gặp nhất của lý thuyết quyết định Bayes. Ta xét bài toán phân lớp, trong đó đồng nhất tập mẫu X với tập đặc trưng của chúng và có k lớp $\mathcal{A}_1, \mathcal{A}_2, \dots, \mathcal{A}_k$. Giả sử rằng từ dữ liệu huấn luyện ta đã biết được các xác suất tiên

nghiệm $P(\omega_i)$ và các phân bố xác suất có điều kiện $P(x/\omega_i)$ với mỗi $i=1,..,k$ và x thuộc X .

5.2.1. Các quy tắc phân lớp MAP và ML

Trong mục này, trước hết ta áp dụng lý thuyết quyết định Bayes cho trường hợp tổng quát có nhiều lớp và sau đó minh họa cụ thể bằng trường hợp 2 lớp.

a) Trường hợp nhiều lớp.

Giả sử có đối tượng x , dựa trên các xác suất tiên nghiệm $P(\omega_i)$ và $P(x/\omega_i)$, ta cần xác định lớp cho nó. Khi đó ta ngầm hiểu ω_i là giả thuyết x thuộc ω_i (giả thuyết h_i). Công thức Bayes trong trường hợp này là :

$$P(\omega_i/x) = \frac{P(x/\omega_i)P(\omega_i)}{P(x)} \quad (5.8a)$$

trong đó $P(x)$ được tính theo công thức xác suất đầy đủ :

$$P(x) = \sum_{i=1}^k P(x/\omega_i)P(\omega_i). \quad (5.8b)$$

Hàm quyết định cho mỗi lớp ω_i là :

$$g_i(x) = P(\omega_i/x) \text{ hoặc } P(\omega_i)P(x/\omega_i). \quad (5.8c)$$

Quy tắc xác suất hậu nghiệm cực đại là quyết định x thuộc lớp ω_{MAP} :

$$\omega_{MAP} = \arg \max\{P(\omega_i/x : i \leq k\} = \arg \max\{P(x/\omega_i)P(\omega_i) : i \leq k\}. \quad (5.8e)$$

Nếu các xác suất $P(\omega_i)$ như nhau với mọi i thì ta có quy tắc hợp lý nhất ML :

$$\omega_{ML} = \arg \max\{P(x/\omega_i) : i \leq k\}. \quad (5.8f)$$

b) Trường hợp 2 lớp

Trở lại với trường hợp hai lớp, công thức (5.8e) có thể viết lại như sau:

$$\text{Nếu } P(x/\omega_1)P(\omega_1) > P(x/\omega_2)P(\omega_2) \text{ ta quyết định } x \in \omega_1, \quad (5.9a)$$

ngược lại,

$$\text{nếu } P(x/\omega_1)P(\omega_1) < P(x/\omega_2)P(\omega_2) \text{ ta quyết định } x \in \omega_2; \quad (5.9b)$$

Khi đặt $v(x) = \frac{P(x/\omega_1)}{P(x/\omega_2)}$ thì quy tắc trên có thể phát biểu là:

$$\text{Nếu } v(x) > \frac{P(\omega_2)}{P(\omega_1)} \text{ thì } x \in \omega_1, \text{ ngược lại thì } x \in \omega_2. \quad (5.9c)$$

Đại lượng $v(x)$ trong(5.9c) được gọi là tỷ số khả năng (likelihood ratio), quyết định tùy thuộc vào kết quả so sánh nó với tỷ số ngưỡng $P(\omega_2)/P(\omega_1)$.

Ví dụ 2: Chẩn đoán ung thư

Một địa phương có 0,8% dân số bị bệnh ung thư. Xét nghiệm sinh hóa T cho thấy 98% người mắc bệnh ung thư (UT) cho kết quả dương tính (+) và 97% người không mắc bệnh ung thư (NUT) cho kết quả âm tính (-). Một người xét nghiệm sinh hóa T cho kết quả dương tính, ta có kết luận được người này bị ung thư hay không?

Từ giả thiết, ta có được tri thức tiền nghiệm như sau:

$$P(UT) = 0,008; P(NUT) = 0,992$$

$$P(+/UT) = 0,98; P(-/UT) = 0,02;$$

$$P(-/NUT) = 0,97; P(+/NUT) = 0,03$$

Muốn kết luận người đó có ung thư không ta áp dụng quy tắc quyết định MAP theo công thức (5.9a). Ta có:

$$P(+/UT).P(UT) = 0,98 \times 0,008 = 0,00784$$

$$P(+/NUT).P(NUT) = 0,03 \times 0,992 = 0,02976.$$

Như vậy $P(+/NUT).P(NUT) > P(+/UT).P(UT)$ và $h_{MAP} = \text{không ung thư}$, tức là ta không kết luận người này bị ung thư.

Xác suất trên có thể chuẩn hóa với tổng bằng 1, tức là:

$$P(UT/+) = \frac{0,00784}{0,00784 + 0,02976} = 0,21 \quad (5.9d)$$

$$P(NUT/+) = \frac{0,02976}{0,00784 + 0,02976} = 0,79$$

Ví dụ 3: phân loại sản phẩm

Giả sử các nút chai dùng cho rượu vang gồm 2 loại: $\omega_1 = \{\text{nút tốt}\}$ và $\omega_2 = \{\text{nút bình thường}\}$. Hơn nữa tập mẫu cho biết có $n_1 = 1802840$ nút thuộc lớp ω_1 và $n_2 = 2704260$ nút thuộc lớp ω_2 . Kết quả thống kê cho thấy nút chai với đặc trưng x có xác suất thuộc mỗi loại tương ứng là $P(x/\omega_1) = 0,96$ và $P(x/\omega_2) = 0,883$. Cần xác định xem nút chai này thuộc loại nào.

Từ giả thiết, ta xác định được các xác suất tiền nghiệm để một nút bất kỳ thuộc mỗi lớp là :

$$P(\omega_1) = n_1/n = 0,4; P(\omega_2) = n_2/n = 0,6.$$

Với các xác suất này, nếu phải đoán mò một nút chai bất kỳ thuộc loại nào thì ta có thể đoán nó thuộc lớp ω_2 . Khi đó xác suất sai là 40%.

Bây giờ ta áp dụng công thức (5.9a). Ta có:

$$P(\omega_1) P(x/\omega_1) = 0,4 \cdot 0,96 = 0,384$$

$$P(\omega_2) P(x/\omega_2) = 0,6 \cdot 0,883 = 0,49981.$$

Ta thấy $P(\omega_2) P(x/\omega_2)$ lớn hơn $P(\omega_1) P(x/\omega_1)$ quyết định nút này thuộc loại nút thường (lớp ω_2).

Nếu chuẩn hóa ta có :

$$P(\omega_1/x) = \frac{0,384}{0,384 + 0,49981} = 0,4345; P(\omega_2/x) = \frac{0,49981}{0,384 + 0,4998} = 0,5655$$

5.2.2. Phân lớp **cực tiểu rủi ro**

Sau khi quyết định lớp cho mỗi đối tượng, ta thường có hành động tương ứng với quyết định được chọn. Hành động này đòi hỏi chi phí (thiệt hại) tương ứng được tính theo một độ đo nào đó. Chẳng hạn, nếu ta quyết định đối tượng bị ung thư thì cần chi phí cho điều trị sớm, còn không ung thư thì chỉ cần một ít thuốc bổ hoặc chỉ nghỉ ngoại một thời gian... Khi đó ta cân nhắc quyết định dựa trên kỳ vọng của giá trị chi phí/thiệt hại mà hành động gây nên, và gọi là quyết định theo quy tắc **cực tiểu rủi ro**.

a) Trường hợp nhiều lớp

Trở lại với trường hợp phân làm k lớp, khi quyết định lớp cho đối tượng x thuộc ω_i , ta có hành động tương ứng α_i . Tuy nhiên thực chất đối tượng lại có thể thuộc lớp ω_j với xác suất tương ứng là $P(\omega_j/x)$. Khi đối tượng thuộc lớp ω_j mà ta áp dụng hành động α_i thì chi phí/thiệt hại tương ứng là $\lambda(\alpha_i/\omega_j)$. Ta gọi *rủi ro* khi quyết định đối tượng x thuộc lớp ω_i là kỳ vọng của các chi phí/ thiêt hại do hành động α_i gây nên khi đối tượng có thể thuộc các lớp một cách ngẫu nhiên. Lượng rủi ro này được ký hiệu là $R(\alpha_i/x)$ và tính bởi công thức :

$$R(\alpha_i/x) = \sum_{j=1}^k \lambda(\alpha_i/\omega_j) P(\omega_j/x). \quad (5.10)$$

Ta cần chọn quyết định cực tiểu lượng rủi ro này. Khi đó quy tắc quyết định sẽ là :

$$\text{quyết định } x \text{ thuộc } \omega_i \text{ nếu } R(\alpha_i/x) = \min\{R(\alpha_j/x) : j = 1, \dots, k\}. \quad (5.11)$$

Quy tắc này gọi là *cực tiểu rủi ro*.

b) Trường hợp 2 lớp

Khi chỉ có hai lớp, ký hiệu $\lambda_{i,j} = \lambda(\alpha_i/\omega_j)$ tổn thắt t trung bình khi hành động ứng với mỗi quyết định phân lớp là:

$$R(\alpha_1/x) = \lambda_{1,1}P(\omega_1/x) + \lambda_{1,2}P(\omega_2/x), \quad (5.12a)$$

$$R(\alpha_2/x) = \lambda_{2,1}P(\omega_1/x) + \lambda_{2,2}P(\omega_2/x). \quad (5.12b)$$

Quy tắc quyết định cực tiểu rủi ro sẽ là:

quyết định ω_1 nếu $R(\alpha_1/x) \leq R(\alpha_2/x)$, ngược lại ω_2 .

(5.13c)

Chú ý tới các công thức (5.12a) và (5.12b), quy tắc quyết định này có thể viết lại dưới dạng xác suất:

ta quyết định ω_1 nếu $(\lambda_{2,1} - \lambda_{1,1})P(\omega_1/x) \geq (\lambda_{1,2} - \lambda_{2,2})P(\omega_2/x)$,

(5.13d)

và ω_2 trong trường hợp ngược lại.

Khi thay các xác suất hậu nghiệm bởi các xác suất tiên nghiệm, quy tắc quyết định là:

quyết định ω_1 nếu: $(\lambda_{2,1} - \lambda_{1,1})P(x/\omega_1)P(\omega_1) \geq (\lambda_{1,2} - \lambda_{2,2})P(x/\omega_2)P(\omega_2)$ (5.13e)

và ω_2 trong trường hợp ngược lại.

Giả sử $\lambda_{2,1} > \lambda_{1,1}$ thì ta có biểu diễn khác cho (5.13e):

quyết định ω_1 nếu $\frac{P(x/\omega_1)}{P(x/\omega_2)} > \frac{(\lambda_{12} - \lambda_{22})P(\omega_2)}{(\lambda_{21} - \lambda_{11})P(\omega_1)}$, ngược lại ω_2 . (5.13f)

Khi quyết định đúng không có thiệt hại và quyết định sai thiệt hại như nhau thì quy tắc này trùng với quy tắc phân lớp MAP.

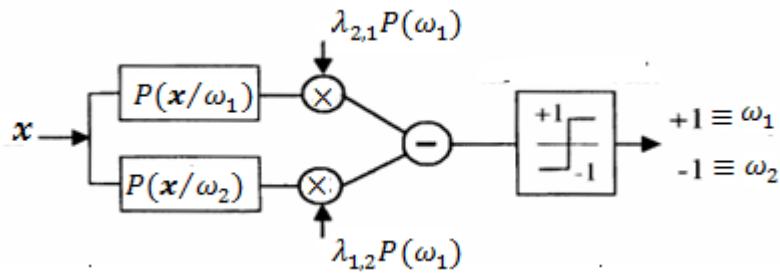
Bây giờ ta xét trường hợp thiệt hại khi quyết định sai khác nhau, và quyết định đúng không thiệt hại, từ (5.13f) ta có quy tắc quyết định là :

quyết định ω_1 nếu $\lambda_{21}P(\omega_1/x) > \lambda_{12}P(\omega_2/x)$

$\Rightarrow \frac{P(x/\omega_1)}{P(x/\omega_2)} > \frac{\lambda_{12}P(\omega_2)}{\lambda_{21}P(\omega_1)}$ ngược lại ω_2 .

(5.13g)

Quy tắc này được minh họa trong hình 5.1.



Hình 5.1. Quy tắc cực tiểu rủi ro cho 2 lớp với thiệt hại khi quyết định sai khác nhau.

Các ví dụ

- *Điều trị ung thư.* Trở lại với ví dụ 2 về chẩn đoán bệnh ở trên, nếu quyết định ung thư (UT) thì cho điều trị sớm (DTS) với chi phí 10 triệu đồng còn khi để muộn hơn (DTM) mà phải điều trị thì tốn 100 triệu đồng. Trong trường hợp đó ta nên quyết định thế nào?

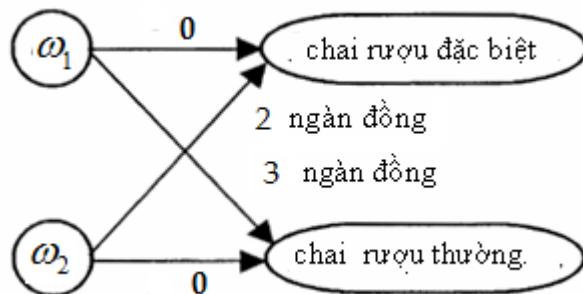
Dựa vào biểu thức (5.9d), ta dễ dàng tính được:

$$R(\text{DTS}/+) = 10 \text{ triệu đồng},$$

$$R(\text{DTM}/+) = 100 \times 0,21 = 21 \text{ triệu đồng}$$

Như vậy quyết định cực tiểu rủi ro là cho điều trị sớm mặc dù xác suất bị ung thư bé hơn.

- *Phân loại nút chai.* Trở lại với bài toán phân loại nút chai trong ví dụ 3, giả sử rằng giá một nút chai loại tốt (ω_1) là 5 ngàn đồng còn loại thường (ω_2) là 2 ngàn đồng và nút loại tốt dùng cho chai rượu đặc biệt còn nút thường dùng cho rượu thường. Khi phân lớp sai nút loại tốt thành loại thường sẽ bị thiệt lượng $\lambda_{2,1} + 5 - 2 = 3$ ngàn đồng còn phân lớp sai nút loại thường thì bộ phận đóng chai loại bỏ và mất lượng $\lambda_{1,2} = 2$ ngàn đồng (xem hình 5.2).



Hình 5.2: Sơ đồ thiệt hại của hai lớp nút chai với quyết định đúng không tồn thắt

Xét một nút chai có đặc trưng x đã nêu, có $P(x/\omega_1) = 0,96$ và $P(x/\omega_2) = 0,883$. Khi đó áp dụng quy tắc quyết định (9.13g) ta có:

$$\frac{P(x/\omega_1)}{P(x/\omega_2)} = \frac{0,96}{0,883} = 1,087 > \frac{\lambda_{1,2} P(\omega_2)}{\lambda_{2,1} P(\omega_1)} = \frac{2 \times 0,6}{3 \times 0,4} = 1$$

Trong trường hợp này ta quyết định x thuộc lớp ω_1 , tức là xếp vào loại nút tốt thay vì xếp loại nút thường như ở trên.

5.2.3. Bộ phân lớp tối ưu bayes

Ở trên cho thấy rằng đối với bài toán phân lớp dựa trên dữ liệu đào tạo D đã cho và có không gian giả thuyết H , ta có hai cách tiếp cận: 1) tìm giả thuyết có khả năng nhất và dùng giả thuyết này để đoán nhận lớp của đối tượng mới; 2) tìm ước lượng của lớp có khả năng nhất cho đối tượng mới. Ta sẽ xét liên hệ giữa hai cách tiếp cận này.

Để đơn giản, giả sử không gian H gồm ba giả thuyết h_1, h_2, h_3 và xác suất hậu nghiệm của ba giả thuyết này khi cho trước dữ liệu đào tạo D là 0,4; 0,3; 0,3 tương ứng. Khi đó h_1 là giả thuyết MAP, theo cách tiếp cận thứ nhất thì mọi đối tượng mới sẽ được phân lớp theo giả thuyết h_1 . Nay giờ xét mẫu mới x được phân loại là dương đối với giả thuyết h_1 nhưng là âm với các giả thuyết h_2, h_3 . Để dàng tính được xác suất để x dương là 0,4, và âm là 0,6. Theo cách nhìn thứ hai này, kết quả phân loại x khác với kết quả phân loại được thực hiện dựa trên giả thuyết MAP theo cách thứ nhất.

Theo tiếp cận này, việc phân lớp mỗi đối tượng mới sẽ kết hợp các dự đoán của mọi giả thuyết dựa trên trọng số của các xác suất hậu nghiệm. Nếu như nhãn lớp của mẫu mới có thể là giá trị v_j bất kỳ từ tập V thì xác suất $P(v_j|D)$ là:

$$P(v_j|D) = \sum_{h \in H} P(v_j|h) P(h|D)$$

Khi đó, cách phân lớp tối ưu cho mẫu mới là giá trị v_j mà $P(v_j|D)$ cực đại, tức là theo quy tắc sau đây.

Quy tắc phân lớp tối ưu Bayes: Nhãn của đối tượng mới x sẽ là giá trị

$$v = \operatorname{argmax} \left\{ \sum_{h \in H} P(v_i/h) P(h/D) : v_i \in V \right\} \quad (5.14)$$

Trong ví dụ đang xét, tập các lớp có thể cho các mẫu lói là $V = \{+, -\}$ và

$$P(h_1/D) = 0,4; P(+/h_1) = 1; P(-/h_1) = 0;$$

$$P(h_2/D) = 0,3; P(+/h_2) = 0; P(-/h_2) = 1;$$

$$P(h_3/D) = 0,3; P(+/h_3) = 0; P(-/h_3) = 1.$$

$$\text{Do đó: } P(+/D) = \sum_{h \in H} P(+/h) P(h/D) = 0,4 \text{ và } P(-/D) = \sum_{h \in H} P(-/h) P(h/D) = 0,6$$

nên $\operatorname{argmax} \left\{ \sum_{h \in H} P(v_i / h) P(h / D) : v_i \in H \right\} = -$. Ta kết luận đối tượng có nhãn âm tính

5.2.4. Phân lớp Bayes ngây thơ (Naïve Bayes)

Giả sử không gian mẫu được đặc trưng bởi n thuộc tính, và tập giá trị đích/nhãn lớp V . Khi áp dụng phân lớp Bayes cho mỗi mẫu mới $a = (a_1, \dots, a_n)$, ta phải ước lượng xác suất đồng thời $P((a_1, \dots, a_n) | v)$ cho mọi v trong V như trong biểu thức (5.8b) khi thay x bởi a và các ω bởi v . Trong nhiều trường hợp, khi số mẫu ít, việc ước lượng các xác suất $P((a_1, \dots, a_n) | v)$ khó khăn, phương pháp phân lớp Bayes ngây thơ dưới đây thường được áp dụng và cho kết quả thực tế khá tốt.

Theo quy tắc quyết định MAP, mẫu mới a sẽ được gán các giá trị đích có xác suất hậu nghiệm lớn nhất v_{MAP} :

$$\begin{aligned} v_{MAP} &= \arg \max_{v \in V} (v | (a_1, a_2, \dots, a_n)), \\ \text{hay: } v_{MAP} &= \arg \max_{v \in V} \frac{P((a_1, a_2, \dots, a_n) | v) P(v)}{P(a_1, a_2, \dots, a_n)} \\ &= \arg \max_{v \in V} ((a_1, a_2, \dots, a_n) | v) P(v) \end{aligned} \quad (5.15a)$$

Ta cần ước lượng hai thừa số trong biểu thức (5.15a) dựa trên dữ liệu đào tạo. Thừa số $P(v)$ dễ dàng ước lượng nhờ tính tần suất của giá trị v xuất hiện trong dữ liệu đào tạo. Để ước lượng $((a_1, a_2, \dots, a_n) | v)$, ta giả thiết các giá trị thuộc tính là độc lập có điều kiện với nhau (*ngây thơ*), tức là:

$$P(a_1, a_2, \dots, a_n | v) = \prod_{i=1}^n P(a_i | v) \quad (5.15b)$$

Thay các đại lượng tính được này vào (5.15a), quy tắc phân lớp Bayes ngây thơ sẽ quyết định giá trị đích của mẫu mới $a = (a_1, \dots, a_n)$ là v_{NB} :

$$v_{NB} = \arg \max_{v \in V} \left\{ P(v) \prod_{i=1}^n P(a_i | v) \right\} \quad (5.15c)$$

Khi đó các giá trị $P(a_i | v)$ ước lượng từ dữ liệu đào tạo dễ hơn xác suất đồng thời $P((a_1, a_2, \dots, a_n) | v)$. Vì giả thiết các xác suất có điều kiện độc lập theo các thuộc tính khó chấp nhận về mặt toán học nên phương pháp này có tên gọi là *ngây thơ*.

Tóm lại, phương pháp học mạng Bayes ngây thơ bao gồm các bước tính toán $P(v), P(a_i | v)$ cho mọi v thuộc tập giá trị đích V dựa trên tần suất xuất hiện trong dữ liệu đào tạo, sau đó các kết quả này được dùng để phân lớp mẫu mới theo biểu thức (5.15c) với giả thiết các giá trị thuộc tính là độc lập có điều kiện.

Ví dụ 4

Ta trở lại bài toán học phân loại những ngày mà một người nào đó đi *choi câu lông* trong chương 3 với tập mẫu huấn luyện trong bảng 3.1 nhưng việc quyết định đi *choi phụ* thuộc ngẫu nhiên vào đặc điểm thời tiết. Tập mẫu huấn luyện này gồm 14 mẫu cho quyết định *choi câu lông*, trong đó mỗi ngày được mô tả bởi các thuộc tính: *Tiết trời*, *Nhiệt độ*, *Độ ẩm* và *Gió*. Tỷ lệ chiều nhỏ nên không ước lượng các phân bố xác suất tiền nghiệm được. Ta sử dụng phân lớp Bayes ngay tho và tập dữ liệu này để phân lớp mẫu a: (*Tiết trời=nắng*, *Nhiệt độ=mát*, *Độ ẩm=cao*, *Gió=mạnh*).

Nhiệm vụ của chúng ta là dự đoán giá trị người này là có đi hay không, tức là $V = \{\text{có}, \text{không}\}$. Áp dụng phương trình (5.14c) ta có:

$$\begin{aligned} v_{NB} &= \arg \max_{v \in V} \left\{ P(v) \prod_i P(a_i | v) \right\} \\ &= \arg \max_{v \in V} \{P(v)P(\text{Tiết trời=nắng}/v)P(\text{Nhiệt độ=mát}/v)P(\text{Độ ẩm=cao}/v)P(\text{Gió=mạnh}/v)\} \end{aligned}$$

Chú ý là trong biểu thức cuối cùng a_i là các giá trị thuộc tính cụ thể của mẫu mới. Để xác định v_{NB} , từ dữ liệu đào tạo ta cần tính 10 xác suất. Đầu tiên, xác suất của các giá trị mục tiêu khác nhau được tính dựa trên tần suất của chúng trên 14 mẫu đào tạo:

$$P(\text{choi câu =có}) = 9/14 = 0,64,$$

$$P(\text{choi câu =không}) = 5/14 = 0,3.$$

Tương tự ta ước lượng các xác suất có điều kiện, chẳng hạn, xác suất *gió=mạnh* là:

$$P(\text{Gió=mạnh}/\text{Choi câu =có}) = 3/9 = 0,33$$

$$P(\text{Gió=mạnh}/\text{Choi câu =không}) = 3/5 = 0,6$$

Sử dụng các ước lượng tương tự ta tính được:

$$P(\text{có}).P(\text{nắng/có}).P(\text{mát/có}).P(\text{cao/có}).P(\text{mạnh/có}) = 0,0053,$$

$$P(\text{không}).P(\text{nắng/không}).P(\text{mát/không}).P(\text{cao/không}).P(\text{mạnh/không}) = 0,02026.$$

Kết quả trên cho thấy phân lớp Bayes ngay tho gán các giá trị mục tiêu *Choi câu lông = không* cho mẫu mới này dựa trên ước lượng xác suất học từ dữ liệu đào tạo.

Chuẩn hóa các đại lượng trên, ta tính được xác suất không *choi câu lông* với điều kiện a là:

$$P(\text{Choi câu lông = không}/a) = \frac{0,0206}{0,0206 + 0,0053} = 0,795.$$

Phương pháp này thường được áp dụng có hiệu quả để phân lớp văn bản.

5.2.5. Phân lớp Bayes khi mỗi lớp có phân bố chuẩn

Bây giờ ta xét phân bố xác suất của mỗi lớp ω_i có phân bố chuẩn tức là hàm mật độ của nó có dạng:

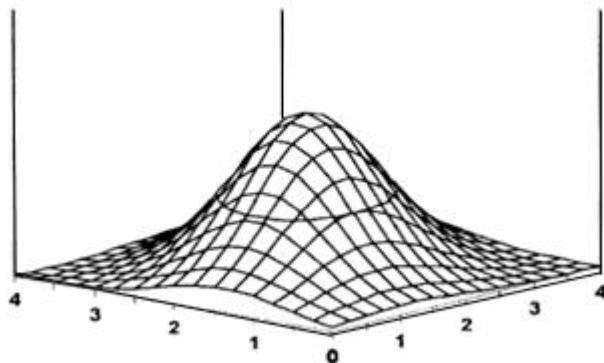
$$p(\mathbf{x}/\omega_i) = \frac{1}{(2\pi)^{n/2}|\Sigma_i|^{1/2}} \exp\left(-\frac{1}{2}(\mathbf{x} - \boldsymbol{\mu})' \Sigma_i^{-1} (\mathbf{x} - \boldsymbol{\mu})\right), \quad (5.16a)$$

trong đó:

$$\boldsymbol{\mu}_i = E_i[\mathbf{x}] \text{ là vectơ trung bình của lớp } \omega_i, \quad (5.16b)$$

$$\Sigma_i = E_i[(\mathbf{x} - \boldsymbol{\mu}_i)(\mathbf{x} - \boldsymbol{\mu}_i)'] \text{ là ma trận hiệp phương sai của lớp } \omega_i \quad (5.16c)$$

Lưu ý rằng $\boldsymbol{\mu}_i, \Sigma_i$ là các tham số của phân bố và chúng được ước lượng bởi \mathbf{m}_i và \mathbf{C}_i tương ứng như trong mục 4.2.4.3 chương trước. Đồ thị của một phân bố chuẩn hai chiều được mô tả trong hình 5.3.



Hình 5.3. Đồ thị của phân bố chuẩn hai chiều

Một tập dữ liệu trộn của hai lớp có phân bố chuẩn được minh họa trong hình 5.4.



Hình 5.4. Một tập dữ liệu trộn của hai lớp có phân bố chuẩn

Xét trường hợp tổng quát có nhiều lớp, ta tính hàm quyết định (5.8c) cho lớp ω_i với mật độ phân bố chuẩn nhiều chiều:

$$\begin{aligned} g_i(\mathbf{x}) &= P(\omega_i) \cdot p(\mathbf{x}/\omega_i) \\ &= P(\omega_i) \cdot \frac{1}{(2\pi)^{n/2} |\Sigma_i|^{1/2}} \exp\left(-\frac{1}{2}(\mathbf{x} - \boldsymbol{\mu})' \Sigma_i^{-1} (\mathbf{x} - \boldsymbol{\mu})\right) \end{aligned} \quad (5.16d)$$

Áp dụng biến đổi đơn điệu logarit ta có hàm quyết định mới:

$$d_i(\mathbf{x}) = \ln(g_i(\mathbf{x})) = -\frac{1}{2}(\mathbf{x} - \boldsymbol{\mu})' \Sigma_i^{-1} (\mathbf{x} - \boldsymbol{\mu}) - \frac{n}{2} \ln 2\pi - \frac{1}{2} \ln |\Sigma_i| + \ln P(\omega_i); \quad (5.16e)$$

Khi các hiệp phương sai của các lớp như nhau ($\Sigma_i = \Sigma$), ta bỏ qua các hằng số giống nhau $-\frac{n}{2} \ln 2\pi - \frac{1}{2} \ln |\Sigma_i|$ thì nhận được:

$$d_i(\mathbf{x}) = \ln(g_i(\mathbf{x})) = -\frac{1}{2}(\mathbf{x} - \boldsymbol{\mu})' \Sigma_i^{-1} (\mathbf{x} - \boldsymbol{\mu}) + \ln P(\omega_i); \quad (5.16f)$$

Đối với trường hợp hai lớp, ta dùng hàm phân biệt $d(\mathbf{x}) = d_1(\mathbf{x}) - d_2(\mathbf{x})$, dễ dàng tính được :

$$d(\mathbf{x}) = \mathbf{w}' \mathbf{x} + w_0 \quad (5.16g)$$

$$\text{trong đó } \mathbf{w} = \Sigma^{-1}(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2) \quad (5.16h)$$

$$\text{và } w_0 = -\frac{1}{2}(\boldsymbol{\mu}_1' \Sigma^{-1} \boldsymbol{\mu}_1 - \boldsymbol{\mu}_2' \Sigma^{-1} \boldsymbol{\mu}_2) + \ln \frac{P(\omega_1)}{P(\omega_2)}. \quad (5.16k)$$

Như vậy, ta có hàm quyết định tuyến tính tương tự như công thức (4.19c) trong chương trước, trong đó thay gần đúng các $\boldsymbol{\mu}_i$ và Σ bằng các ước lượng \mathbf{m}_i và C .

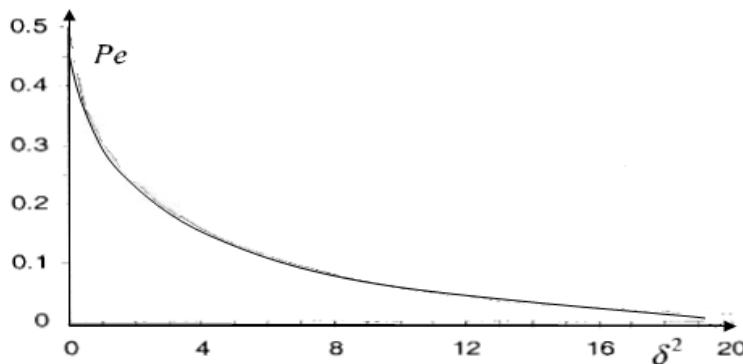
Ký hiệu khoảng cách Mahalanobis của hiêu hai giá trị trung bình lớp là δ , tức là :

$$\delta^2 = \mathbf{w} = (\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)' \Sigma^{-1} (\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2), \quad (4.17a)$$

Fukunaga đưa ra ước lượng lỗi Pe :

$$Pe = 1 - erf(\delta/2) \text{ trong đó } erf(x) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^x e^{-t^2/2} dt. \quad (5.17b)$$

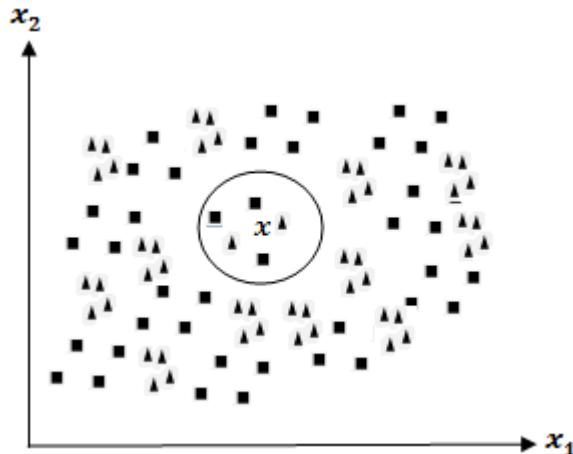
Dáng điệu biến thiên của Pe đối với δ^2 được mô tả trong hình 5.5.



Hình 5.5. Xác suất lỗi của hai lớp có phân bố chuẩn cùng hiệp phương sai

5.3. QUY TẮC QUYẾT ĐỊNH K LÁNG GIỀNG GẦN NHẤT (K-NN)

Đối với bài toán học có giám sát, một phương pháp địa phương đơn giản để phân lớp là dùng quy tắc k-láng giềng gần nhất (k-nearest neighbours) và ký hiệu là k-NN. Giả sử tập mẫu đã biết nhãn là $D = \{x^1, \dots, x^N\}$ và k là số cho trước. Với mỗi mẫu có đặc trưng x , ta tìm k đối tượng trong D gần với nó nhất và gán nhãn của lớp có nhiều phần tử nhất trong số k đối tượng này. Hình 5.6 minh họa quy tắc này trong không gian đặc trưng 2-chiều với $k=5$.



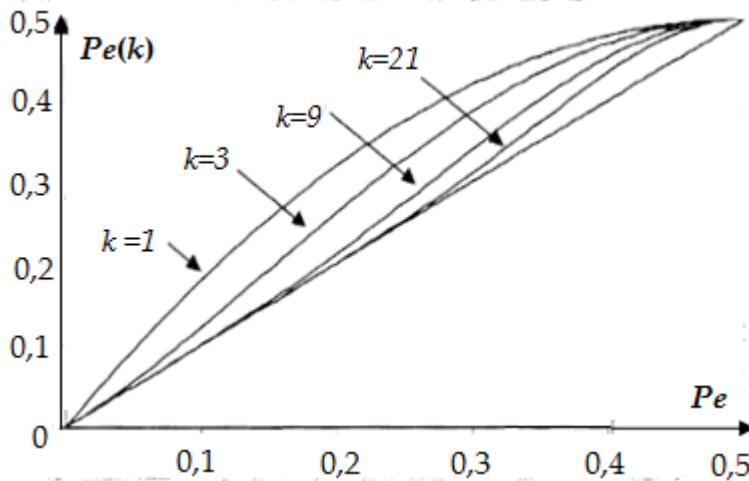
Hình 5.6. Quy tắc k-NN với $k=5$, gán điểm vuông cho x

Ký hiệu $Pe(k)$ là tỷ lệ lỗi khi dùng phương pháp k-NN, thì khi k tăng, tỷ lệ này hội tụ tới tỷ lệ lỗi Pe của phân lớp Bayes.

Trong trường hợp chỉ có 2 lớp, ta có ước lượng sau cho các tỷ lệ lỗi:

$$Pe \leq Pe(k) \leq \sum_{i=1}^{(k-1)/2} C_k^i [Pe^{i+1}(1-Pe)^{k-i} + Pe^{k-i}(1-Pe)^{i+1}], \quad (5.18)$$

đặc biệt với 1-NN thì có cận trên: $Pe(k) \leq 2Pe(1-Pe)$. Đặc điểm tiệm cận của tỷ lệ lỗi được minh họa trong hình 5.7.



Hình 5.7. Tính tiệm cận của $Pe(k)$ như là hàm của Pe

Lưu ý rằng khi áp dụng phương pháp này, ta không xác định được dạng tường minh cho giả thuyết học được dựa trên dữ liệu đào tạo nên chỉ thích hợp với các bài toán có số mẫu mới cần phân lớp không quá nhiều.

5.4. LỰA CHỌN ĐẶC TRƯNG VÀ PHÂN TÍCH THÀNH PHẦN CHÍNH

Cuối chương trước, ta đã đề cập tới việc giảm số đặc trưng để có tỷ lệ chiều cao thích hợp cho xây dựng bộ phân lớp. Mục này giới thiệu phương pháp chọn đặc trưng và phân tích thành phần chính để giảm chiều dữ liệu.

5.4.1. Lựa chọn đặc trưng

Thông thường, các đặc trưng không cho thông tin về ý nghĩa của nó với chất lượng bộ phân lớp. Vì vậy ta cần kiểm định thống kê để chọn đặc trưng.

Khi có quá nhiều đặc trưng, ta có thể sử dụng phân tích thành phần độc lập để loại bỏ các đặc trưng có tương quan cao. Trong phương pháp này, ta xác định một ngưỡng θ gần bằng 1, nếu hai đặc trưng x_i và x_j mà hệ số tương quan $r_{i,j}$ của chúng :

$$r_{i,j} = \sum_{k=1}^N \frac{(x_k^i - m_i)(x_k^j - m_j)}{(N-1)s_i s_j} \quad (s_j \text{ là các phương sai mẫu tương ứng}) \quad (5.19a)$$

lớn hơn ngưỡng θ thì chỉ cần giữ lại một đặc trưng mà thôi. Các đặc trưng này ta xem là các đặc trưng gốc và thực hiện chọn tập con đặc trưng thích hợp.

Có nhiều cách để chọn đặc trưng, phương pháp thông dụng nhất là tìm kiếm theo một tiêu chuẩn đánh giá nào đó. Giả sử F_t là tập t đặc trưng gốc, F là một tập con d đặc trưng của nó, tức là $|F| = d$ và $J(F)$ là hàm một hàm đã cho để đánh giá khi chọn đặc trưng. Ta tìm tập đặc trưng F^* sao cho :

$$J(F^*) = \operatorname{argmax}\{J(F) / F \subset F_t; |F| = d\}. \quad (5.19b)$$

Chẳng hạn, có thể chọn $J(F) = 1 - Pe$ trong đó Pe là tỷ lệ lỗi khi dùng các đặc trưng F tuy rằng cách đánh giá này còn phụ thuộc vào bộ phân lớp được chọn. Hai cách thông dụng để tìm F tối ưu trong biểu thức (5.19) là phương pháp tìm kiếm vét cạn và phương pháp nhánh-cận, chi tiết hơn có thể tham khảo tài liệu [10].

5.4.2. Phân tích thành phần chính (PCA)

Khi không cần giữ lại đặc trưng gốc, phân tích thành phần chính (Principal component analysis, viết tắt là PCA) là một phương pháp hiệu quả để giảm chiều dữ liệu. Trong phương pháp này, từ tập mẫu quan sát D , ta xây dựng ma trận hiệp phương sai C :

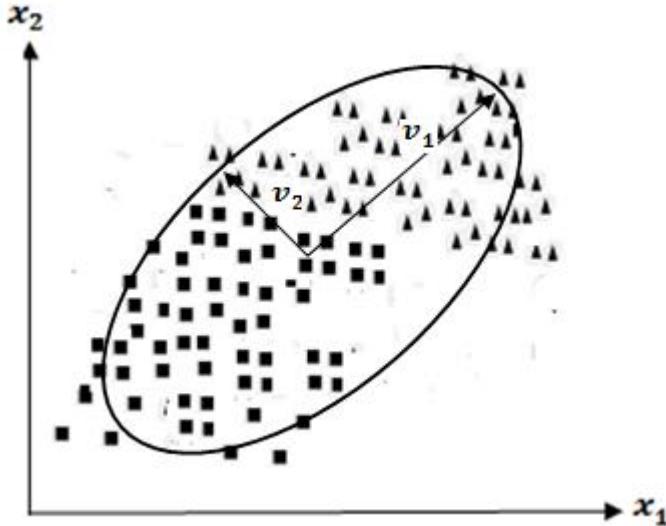
$$C = \begin{bmatrix} c_{11} & c_{12} & \dots & c_{1d} \\ c_{21} & c_{22} & \dots & c_{2d} \\ \dots & \dots & \dots & \dots \\ c_{d1} & c_{d2} & \dots & c_d \end{bmatrix}, \quad (5.20a)$$

trong đó các hệ số $c_{i,j}$ xác định theo công thức (4.18a) chương trước:

$$c_{i,j} = \frac{1}{N-1} \sum_{k=1}^N (x_k^i - m_i)(x_k^j - m_j). \quad (5.20b)$$

Ta tìm các vec tơ riêng ứng với k giá trị riêng lớn nhất của C làm vecto đơn vị và chiếu các đối tượng lên các vecto này làm đặc trưng tương ứng. Tức dùng phép biến đổi trực giao $y = Ax$ ứng với ma trận A trên không gian đặc trưng để ma trận hiệp phương sai C_y có dạng đường chéo sao cho các giá trị riêng λ_i của C_y giảm dần, khi đó đặc trưng mới là k thành phần đầu của hệ tọa độ mới. Hình 5.8 minh họa 2 vecto riêng của C , nếu lấy 1 đặc trưng thì phương pháp PCA sẽ chiếu vecto đặc trưng gốc lên vecto v_1 .

Bản chất của phương pháp PCA là ta tìm các hướng mà dữ liệu phân bố với biên độ lớn quanh tâm của tập dữ liệu (giá trị riêng lớn) làm vecto cơ sở cho không gian đặc trưng mới. Chi tiết hơn về phương pháp này có thể tham khảo [4,6].



Hình 5.8. Hai vectơ riêng của ma trận hiệp phương sai

Lưu ý rằng khi dùng phương pháp PCA để giảm đặc trưng cần cẩn thận vì các lý do sau:

- Các thành phần chính là *bien đổi tuyến tính* của đặc trưng gốc nên khi giảm chiều có thể làm mất thông tin *phi tuyến* trong dữ liệu.
- Các thành phần chính với đóng góp nhỏ trong phương sai toàn phần nhiều khi vẫn có ý nghĩa
- Rất khó giải thích ngữ nghĩa của thành phần chính trong khi các ý nghĩa rất rõ ràng khi dùng đặc trưng gốc

Tuy có các nhược điểm trên nhưng PCA là công cụ hữu hiệu để giảm chiều dữ liệu trong nhiều ứng dụng.

5.5. ĐÁNH GIÁ CÁC BỘ PHÂN LỐP

Trong mục này ta sẽ đánh giá các bộ phân lớp theo hai trường hợp : 1) Chúng dùng chung dữ liệu huấn luyện, khi đó chỉ đơn thuần so sánh ước lượng lỗi ; 2) Mỗi bộ phân lớp sử dụng một tập dữ liệu khác nhau.

5.5.1. Ước lượng lỗi của bộ phân lớp

Trong trường hợp các mẫu được lấy ngẫu nhiên độc lập cùng phân bố, người ta dùng phương pháp *đánh giá chéo* (*cross-validation*) bằng cách chia tập mẫu quan sát được thành hai tập, một tập để huấn luyện và một tập kiểm tra. Ký hiệu \$Pe\$ là lỗi thực

(tỷ lệ lỗi thực chưa biết) và $Pe_t(n)$ là tỷ lệ lỗi kiểm tra trong tập con gồm n đối tượng dữ liệu. Khi đó $Pe_t(n)$ là ước lượng không chêch của Pe :

$$Pe = E[Pe_t(n)] \quad (5.21a)$$

$$\text{và độ lệch chuẩn là: } \sqrt{\frac{Pe(1-Pe)}{n}}. \quad (5.21b)$$

Nếu số mẫu kiểm tra $n > 30$, lý thuyết thống kê cho phép kết luận $Pe_t(n)$ xấp xỉ phân bố chuẩn với trung bình Pe và phương sai là $Pe(1-Pe)/n$. Khi đó ta dùng lý thuyết thống kê để ước lượng khoảng cho Pe , chẳng hạn, với xác suất 95% thì lỗi thực Pe nằm trong khoảng:

$$Pe_t(n) \pm 1,96 \sqrt{\frac{Pe_t(n)(1-Pe_t(n))}{n}} \quad (5.21c)$$

Tổng quát hơn, với khoảng tin cậy $N=1-\alpha$ (còn gọi là độ tin cậy) thì Pe thuộc khoảng:

$$Pe_t(n) \pm Z_N \sqrt{\frac{Pe_t(n)(1-Pe_t(n))}{n}} \quad (5.20e)$$

trong đó Z_N được xác định bằng bảng phân bố chuẩn theo giá trị $\alpha/2$.

Ví dụ 5

Giả sử rằng dữ liệu kiểm tra gồm $n = 40$ mẫu và giả thiết có $r = 12$ lỗi. Như vậy lỗi mẫu $Pe_t(40) = 12/40 = 0,30$. Ước lượng lỗi thực Pe với xác suất 95% thuộc khoảng:

$$0,30 \pm (1,96*0,07) = 0,30 \pm 0,14.$$

Đối với bộ phân lớp Bayes, ký hiệu P_i là ước lượng tiền nghiệm cho lớp ω_i tương ứng, $i=1,\dots,k$. Khi đó, ngoài việc ước lượng lỗi gộp như trên, lỗi thực Pe của bộ phân lớp được ước lượng qua các ước lượng lỗi thực Pe_i của các lớp theo công thức:

$$Pe = \sum_{i=1}^c P_i Pe_i. \quad (5.22a)$$

Ước lượng này có phương sai là:

$$D[Pe_t(n)] = \sigma^2 = \sum_{i=1}^c P_i^2 \frac{Pe_i(1-Pe_i)}{n_i} \quad (5.22b)$$

5.5.2. Phương pháp k-tập (k-folds) đánh giá phương pháp học

Như đã nói ở trên, đánh giá các bộ phân lớp có chung dữ liệu chỉ đơn thuần là so sánh các ước lượng lỗi bằng các kỹ thuật thống kê. Phương pháp k-tập (k-folds) tăng độ chính xác cho ước lượng lỗi.

Trong phương pháp này, với $k > 1$ cho trước, người ta chia ngẫu nhiên tập dữ liệu thành k tập con $\{D_i \mid i=1,..,k\}$. Lần lượt, người ta loại ra một tập con D_i và huấn luyện bộ phân lớp trên $k-1$ tập còn lại rồi kiểm tra lỗi Pe_i trên tập D_i này. Lỗi thực được ước lượng bằng lỗi kiểm tra Pe_t là trung bình của các lỗi này :

$$Pe_t = \sum_{i=1}^k \frac{1}{k} Pe_i, \quad (5.23a)$$

và phương sai ước lượng bởi :

$$s^2 = \frac{1}{k(k-1)} \sum_{i=1}^k (Pe_i - Pe_t)^2 \quad (5.23b)$$

5.5.3. So sánh các bộ phân lớp

Bây giờ ta xét hai bộ phân lớp được thiết kế và đã được kiểm tra trên hai tập mẫu S_1, S_2 bao gồm n_1, n_2 mẫu ngẫu nhiên tương ứng. Ta muốn ước lượng sự khác biệt lỗi thật sự d giữa hai bộ phân lớp này:

$$d = P^1 e(S_1) - P^2 e(S_2) \quad (5.24a)$$

Khi đó d được ước lượng bởi \bar{d} của hai tập mẫu

$$\bar{d} = P^1 e_t(n_1) - P^2 e_t(n_2). \quad (5.24b)$$

Mặc dù không chứng minh nhưng ta có thể thấy rằng \bar{d} là một ước lượng không chêch của d ; $E[\bar{d}] = d$. Khi n_1, n_2 lớn hơn 30, cả hai hàm $P^1 e_t(n_1)$ và $P^2 e_t(n_2)$ đều được xấp xỉ bởi phân phối chuẩn. Do hiệu của hai phân phối chuẩn cũng là phân phối chuẩn nên \bar{d} cũng được xấp xỉ bởi phân phối chuẩn, với kỳ vọng là d . Ngoài ra thì phương sai của phân phối này là :

$$\sigma_{\bar{d}}^2 \approx \frac{P^1 e_t(n_1)(1-P^1 e_t(n_1))}{n_1} + \frac{P^2 e_t(n_2)(1-P^2 e_t(n_2))}{n_2} \quad (5.24c)$$

Với độ tin cậy m ước lượng khoảng cho d là:

$$d = \bar{d} \pm z_N \sqrt{\frac{P^1 e_t(n_1)(1-P^1 e_t(n_1))}{n_1} + \frac{P^2 e_t(n_2)(1-P^2 e_t(n_2))}{n_2}} \quad (5.24d)$$

trong đó z_N là hằng số được xác định trong bảng phân bố chuẩn ứng với giá trị $\alpha/2$ và $\alpha = 1-m$. Tương tự ta có thể dùng lý thuyết thống kê để đánh giá bộ phân lớp nào tốt hơn.

5.5.4. Một số đại lượng và thông tin khác

Khi đánh giá chất lượng của các hệ phân lớp và các hệ truy hồi thông tin (information retrieval), ngoài tỷ lệ lỗi, trong một số lĩnh vực người ta còn quan tâm tới các đặc trưng sau.

Ma trận nhầm lẫn(Confusion matrix). Giả sử bộ phân lớp dự đoán N đối tượng dữ liệu gồm k lớp, ma trận nhầm lẫn $A = [a_{i,j}]$ là ma trận vông cấp k trong đó phần tử ở hàng i cột j biểu thị số đối tượng có nhãn ω_i nhưng được dự đoán thuộc lớp ω_j .

Độ hồi tưởng (recall). Độ hồi tưởng R_i của lớp ω_i là tỷ lệ phần trăm của số đối tượng dữ liệu thuộc lớp này được dự đoán đúng trên số đối tượng dữ liệu được dự đoán thuộc lớp này:

$$R_i = \frac{a_{i,i}}{\sum_{j=1}^k a_{i,j}}. \quad (5.25a)$$

Độ chính xác (precision). Độ chính xác P_i của lớp ω_i là tỷ lệ phần trăm của số đối tượng dữ liệu thuộc lớp này được dự đoán đúng trên số đối tượng được dự đoán là thuộc lớp này :

$$P_i = \frac{a_{i,i}}{\sum_{j=1}^k a_{j,i}} \quad (5.25b)$$

Độ chính xác chung (overall accuracy). Độ chính xác chung Ac là tỷ lệ phần trăm của số đối tượng đã được dự đoán đúng :

$$Ac = \frac{1}{N} \sum_{i=1}^K a_{i,i} \quad (2.25c)$$

Đối với bộ nhận dạng một khái niệm, ngoài các đại lượng này (độ hồi tưởng R , độ đúng P và độ chính xác Ac) người ta còn quan tâm tới các đại lượng sau.

Độ đo F (F-Measure). Độ đo F là kết hợp hài hòa có trọng số giữa độ chính xác và độ hồi tưởng, được tính theo công thức:

$$\frac{P.R}{\alpha R + (1-\alpha)P} \quad (2.25d)$$

trong đó trọng số $\alpha \in [0,1]$. Khi $\alpha = \frac{1}{2}$ độ đo này được gọi là độ đo F cân bằng (balanced F-measure) ký hiệu là F_1 , và $F_1 = 2 \frac{P.R}{P+R}$.

Tỷ lệ lỗi chấp nhận sai (false positive ratio). Tỷ lệ lỗi chấp nhận sai FPR (hoặc là FAR : False Acceptance Rate) là tỷ lệ phần trăm số đối tượng không thuộc khái niệm đang xét nhưng nhận dạng nhầm:

$$FPR = \frac{a_{2,1}}{N}. \quad (2.25e)$$

Tỷ lệ lỗi bác bỏ sai (false negative ratio). Tỷ lệ lỗi bác bỏ sai FNR (hoặc là FRR : False Rejection Rate) là tỷ lệ phần trăm số đối tượng thuộc khái niệm đang xét nhưng bác bỏ nhầm:

$$FNR = \frac{a_{1.2}}{N}. \quad (2.25f)$$

KẾT LUẬN

Lý thuyết quyết định Bayes làm nền tảng để giải quyết nhiều bài toán học có yếu tố ngẫu nhiên. Việc sử dụng công thức Bayes để tìm xác suất hậu nghiệm cho ta quy tắc MAP từ các xác suất tiền nghiệm cho ta tên gọi của phương pháp này. Trong bài toán học khái niệm, với một số giả thuyết kèm theo, quyết định MAP cho ta giả thuyết phù hợp. Trong bài toán hồi quy với nhiều tráng có phân bố chuẩn, giả thuyết có khả năng nhất h_{ML} cho ta lời giải của phương pháp bình phương tối thiểu.

Bài toán phân lớp dựa trên các xác suất tiền nghiệm được xác định bởi kỹ thuật thống kê là một ứng dụng thường gặp của lý thuyết Bayes. Nhiều trường hợp, ta giả thiết tỷ lệ các lớp như nhau, khi đó ta có quy tắc quyết định ML (hợp lý nhất/ có khả năng nhất). Khi mỗi quyết định được làm cơ sở cho hành động sau đó, ta cần tính đến chi phí phát sinh bởi hành động tương ứng, lúc đó quyết định tốt nhất là cực tiểu rủi ro.

Trong trường hợp có nhiều giá trị thuộc tính, các dữ liệu quan sát không đủ để ước lượng các xác suất có điều kiện cho mỗi lớp, ta có thể áp dụng phương pháp Bayes ngây thơ. Mặc dù giả thiết về tính độc lập có điều kiện của các thuộc tính khó chấp nhận về mặt toán học nhưng nhiều trường hợp ứng dụng cho kết quả tốt.

Khi dữ liệu là hỗn hợp các tập có phân bố chuẩn nhiều chiều ta có thể dễ dàng tìm được các hàm quyết định. Nói riêng, khi các lớp này có hiệp phương sai như nhau thì ta có phân biệt tuyến tính tương tự như phân lớp khoảng cách cực tiểu với metric Mahalanobis trong chương trước.

Quy tắc k-NN cho ta một phương pháp phân lớp đơn giản, có thể áp dụng để tìm nhanh nhãn lớp của mẫu mới khi đối tượng cần xác định lớp không nhiều.

Khi muốn giảm chiều dữ liệu, sau khi đã phân tích thành phần độc lập, ta có thể chọn tập con đặc trưng nhờ phương pháp thống kê. Phân tích thành phần chính (PCA) cũng là một phương pháp hiệu quả để giảm chiều dữ liệu, tuy nhiên các đặc trưng mới không cho ta rõ ý nghĩa của nó, ngoài ra, cần thận trọng khi áp dụng cho các bài toán phi tuyến.

Việc đánh giá, so sánh các phương pháp học và các bộ phân lớp dựa trên việc so sánh lỗi. Khi ước lượng lỗi để so sánh các phương pháp học có thể dùng phương pháp k-folds. Các kỹ thuật kiểm định giả thuyết thống kê là công cụ hữu hiệu cho những công việc này.

BÀI TẬP

- 1) Phát biểu và giải bài toán đoán nhận đồng tiền như trong ví dụ 1 với một số thay đổi.
 - a) Chỉ có 5 đồng tiền và có 5 lần được mặt ngửa trong 20 lần tung ngẫu nhiên
 - b) Có 6 đồng tiền và 7 lần được mặt ngửa trong 20 lần tung
- 2) Với bài toán như trong ví dụ 2 nếu nghiên cứu thực tế cho biết 0,7% dân số mắc bệnh ung thư, 98% người mắc bệnh ung thư cho kết quả dương tính, 96% người không mắc bệnh ung thư cho kết quả âm tính.
- 3) Giả sử có một trang trại nuôi giống gà nuôi chung một giống gà nước ngoài GN (1200 con) và hai giống gà trong nước G1 (900 con) và G2 (900 con). Sau 3 tháng tuổi ta lọc các con lớn nuôi riêng. Biết rằng gà ta có trọng lượng 400g- 500g là lớn nhưng gà ngoại nhập thì là bé và các xác suất này trong mỗi loài đều bằng 0,5. Một con gà ngẫu nhiên có trọng lượng thuộc khoảng này, chỉ dựa vào trọng lượng để xác định loại gà lớn hay bé.
- 4) Chỉ ra rằng trong trường hợp 2 lớp, quy tắc quyết định MAP có:

$$P(\tilde{l}|\tilde{x}) = \min\{P(\omega_1|\tilde{x}), P(\omega_2|\tilde{x})\}$$

- 5) Xây dựng ba tập dữ liệu có phân bố chuẩn hai chiều cùng phương sai với tâm khác nhau trong R^2 , mỗi tập có 40 đối tượng
 - a) Xác định biên quyết định cho các tập dữ liệu này theo phân lớp Bayes.
 - b) Tạo thêm mỗi lớp 10 đối tượng ngẫu nhiên và dùng phương pháp K-NN để phân lớp với k=4
 - c) So sánh các sai số của các phương pháp trên
- 6) Một vùng có 0,7% dân số nhiễm một loại bệnh lạ. Biết rằng trong số nhiễm bệnh có 97% bị sốt còn những người không nhiễm bệnh này có 1% bị sốt. Trong vùng dịch có một người bị sốt.
 - a) Hãy dùng tiêu chuẩn MAP để chẩn đoán người này có nhiễm bệnh này hay không?
 - b) Một liều thuốc điều trị sớm là 1 triệu đồng còn điều trị muộn là 100 triệu đồng, hãy quyết định xem có nên cho người này dùng thuốc không?

Chương 6

HỌC KHÔNG GIÁM SÁT

Những chương trước giới thiệu một số phương pháp tiếp cận cho học có giám sát, trong đó nhấn của các đối tượng trong tập dữ liệu huấn luyện đã biết. Chương này sẽ giới thiệu một số phương pháp thông dụng giải hai bài toán học không giám sát thường gặp: ước lượng hàm mật độ và phân cụm dữ liệu.

6.1. ƯỚC LƯỢNG HÀM MẬT ĐỘ

Trong nhiều trường hợp, ta cần ước lượng các hàm mật độ xác suất của các *biến vectơ ngẫu nhiên* dựa trên các tập dữ liệu quan sát. Các hàm mật độ tìm được sẽ được dùng trong *mô phỏng bài toán đang xét/ học tạo sinh* (generative learning) hoặc dùng để giải bài toán rộng hơn. Để tìm các hàm mật độ này ta có thể dùng hai cách tiếp cận có tham số và không có tham số.

6.1.1. Kỹ thuật có tham số

Giả sử ta có tập dữ liệu huấn luyện \mathcal{D} gồm N mẫu lấy ngẫu nhiên độc lập cùng phân bố có hàm mật độ có dạng đã biết $p(x|\theta)$ phụ thuộc vào vectơ tham số θ . Các phương pháp có tham số sẽ tìm cách ước lượng gần đúng vectơ $\hat{\theta}$ của θ dựa trên một tiêu chuẩn nào đó. Mục này giới thiệu phương pháp ước lượng khả năng nhất (Maximum-likelihood estimation) để giúp độc giả hình dung được các cách tiếp cận theo kỹ thuật này.

Bởi vì các mẫu trong \mathcal{D} được lấy độc lập nên ta có:

$$p(\mathcal{D}|\theta) = \prod_{k=1}^N p(x_k|\theta). \quad (6.1)$$

Lưu ý rằng, đại lượng $p(\mathcal{D}|\theta)$ được gọi là *khả năng* của θ đối với tập mẫu \mathcal{D} . Ta cần tìm ước lượng $\hat{\theta}$ để cực đại $p(\mathcal{D}|\theta)$.

Nếu là vectơ có n thành phần $\theta = (\theta_1, \dots, \theta_n)^T$, ta ký hiệu ∇_θ là toán tử gradient:

$$\nabla_{\theta} = \begin{bmatrix} \frac{\partial}{\partial \theta_1} \\ \vdots \\ \frac{\partial}{\partial \theta_n} \end{bmatrix} \quad (6.2a)$$

Xét hàm log-likelihood $l(\theta)$:

$$l(\theta) = \ln p(\mathcal{D}|\theta) \quad (6.2b).$$

Bởi vì hàm logarit là đơn điệu tăng nên $\hat{\theta}$ cực đại $l(\theta)$ thì cũng cực đại $\ln p(\mathcal{D}|\theta)$, tức là:

$$\hat{\theta} = \arg \max_{\theta} l(\theta) \quad (6.2c)$$

trong đó ẩn đi sự phụ thuộc vào dữ liệu \mathcal{D} . Chú ý tới (6.1) thì (6.2b) được viết lại là:

$$l(\theta) = \sum_{k=1}^N \ln p(x_k|\theta) \quad (6.2d)$$

$$\text{và } \nabla_{\theta} l = \sum_{k=1}^N \nabla_{\theta} \ln p(x_k|\theta) \quad (6.2e)$$

Như vậy các điều kiện cần để ước lượng khả năng nhất cho tham số θ có thể đạt được từ tập n phương trình:

$$\nabla_{\theta} l = \mathbf{0} \quad (6.2f)$$

Để làm ví dụ, ta xét trường hợp phân bố chuẩn nhiều chiều, hàm mật độ có dạng:

$$p(x|\theta) = \frac{1}{\sqrt{(2\pi)^n |\Sigma|}} \exp \left[-\frac{1}{2} (x - \mu)^T \Sigma^{-1} (x - \mu) \right], \quad (6.3a)$$

trong đó μ là kỳ vọng và Σ là ma trận hiệp phương sai. Dưới đây ta xét hai trường hợp: μ chưa biết, cả μ và Σ chưa biết.

a) Trường hợp chưa biết μ

Để đơn giản, trước hết ta xét trường hợp chưa biết μ . Với điều kiện này, ta xét một điểm mẫu x_k , đồng nhất θ với μ trong các biểu thức (6.2d-f) và tính:

$$\ln p(x_k|\mu) = -\frac{1}{2} \ln(2\pi)^n |\Sigma| - \frac{1}{2} (x_k - \mu)^T \Sigma^{-1} (x_k - \mu); \quad (6.3b)$$

$$\Rightarrow \nabla_{\mu} \ln p(x_k|\mu) = \Sigma^{-1} (x_k - \mu) \quad (6.3c)$$

Các biểu thức (6.2e-f) và (6.3c) cho thấy ước lượng hợp lý nhất đối với μ phải thỏa mãn :

$$\sum_{k=1}^N \Sigma^{-1} (x_k - \hat{\mu}) = \mathbf{0} \quad (6.3d)$$

Nhân cả hai vế của phương trình trên với Σ , rồi rút gọn ta thu được:

$$\hat{\mu} = \frac{1}{N} \sum_{k=1}^N x_k \quad (6.3e)$$

Kết quả này chỉ ra rằng ước lượng khả năng nhất đối với kỳ vọng chưa biết là trung bình số học của những mẫu quan sát được.

b) Trường hợp chưa biết μ và Σ

Trong nhiều trường hợp cả kỳ vọng μ và ma trận hiệp phương sai Σ đều chưa biết. Trước hết, ta xét trường hợp một chiều: $\theta_1 = \mu$ và $\theta_2 = \sigma^2$. Khi đó tại mỗi điểm mẫu x_k ta có:

$$\ln p(x_k | \theta) = -\frac{1}{2} \ln 2\pi\theta_2 - \frac{1}{2\theta_2}(x_k - \theta_1)^2 \quad (6.4a)$$

và lấy đạo hàm ta được: $\nabla_{\theta} l = \nabla_{\theta} \ln p(x_k | \theta) = \begin{bmatrix} \frac{1}{\theta_2}(x_k - \theta_1) \\ -\frac{1}{2\theta_2} + \frac{(x_k - \theta_1)^2}{2\theta_2^2} \end{bmatrix} \quad (6.4b)$

Áp dụng (6.2f), ta có các điều kiện để xác định $\hat{\theta}$ là:

$$\sum_{k=1}^N \frac{1}{\hat{\theta}_2} (x_k - \hat{\theta}_1) = 0 \quad (6.4c)$$

và $-\sum_{k=1}^N \frac{1}{\hat{\theta}_2} + \sum_{k=1}^N \frac{(x_k - \hat{\theta}_1)^2}{\hat{\theta}_2^2} = \mathbf{0} \quad (6.4d)$

với $\hat{\theta}_1$ và $\hat{\theta}_2$ là ước lượng của khả năng nhất đối với θ_1 và θ_2 tương ứng.

Thay thế $\mu = \hat{\theta}_1$ và $\sigma^2 = \hat{\theta}_2$ và sắp xếp lại chúng ta thu được ước lượng khả năng nhất đối với μ và σ^2 :

$$\hat{\mu} = \frac{1}{N} \sum_{k=1}^N x_k, \quad (6.4e)$$

$$\text{và } \hat{\sigma}^2 = \frac{1}{N} \sum_{k=1}^N (x_k - \mu)^2 \quad (6.4f)$$

Trong trường hợp tổng quát, phân tích là tương tự ta có ước lượng khả năng nhất của μ và Σ là:

$$\hat{\mu} = \frac{1}{N} \sum_{k=1}^N x_k, \quad (6.4g)$$

$$\text{và } \hat{\Sigma} = \frac{1}{N} \sum_{k=1}^N (x_k - \hat{\mu})(x_k - \hat{\mu})^T. \quad (6.4h)$$

Một lần nữa chúng ta thấy rằng ước lượng khả năng nhất đối với kỳ vọng chính là trung bình mẫu và ước lượng khả năng nhất đối với ma trận hiệp phương sai là giá trị trung bình số học của N ma trận $(x_k - \hat{\mu})(x_k - \hat{\mu})^T$.

6.1.2. Kỹ thuật phi tham số

Có nhiều phương pháp phi tham số thích hợp cho việc nhận dạng hàm mật độ, mục này giới thiệu phương pháp cửa sổ Parzen và phương pháp k-láng giềng gần nhất để ước lượng trực tiếp hàm mật độ tại mỗi điểm được xét.

Xét vecto đặc trưng có hàm mật độ $p(\mathbf{x})$ tại điểm \mathbf{x} . Khi đó với mỗi miền đủ nhỏ nhô R có thể tích V và tâm \mathbf{x} , định lý về giá trị trung bình cho ta ước lượng:

$$p(\mathbf{x})V \approx \int_R p(\mathbf{u})d\mathbf{u} = P(\mathbf{x} \in R) = P. \quad (6.5a)$$

Giả sử tập dữ liệu gồm n mẫu được lấy là độc lập cùng phân phối (i.i.d.) thì xác suất để k trong n mẫu này rơi vào miền R được tính bởi luật phân bố nhị thức

$$P_k = C_n^k P^k (1 - P)^{n-k} \quad (6.5b)$$

và giá trị kỳ vọng cho k là:

$$E[k] = nP \quad (6.5c)$$

nên P được ước lượng bởi công thức:

$$P \approx \frac{k}{n} \approx p(\mathbf{x})V, \quad (6.5e)$$

$$\text{hay là: } p(\mathbf{x}) \approx \frac{k/n}{V} \quad (6.5f)$$

Để ước lượng giá trị hàm mật độ $p(\mathbf{x})$ tại điểm \mathbf{x} , ta tìm một dãy miền có thể tích giảm dần R_1, R_2, \dots , chứa điểm \mathbf{x} và tạo dãy mẫu độc lập có cùng phân bố $p(\mathbf{u})$ sao cho số mẫu rơi vào mỗi miền tăng dần. Ký hiệu V_n là thể tích của R_n và k_n là số lượng các mẫu rơi vào miền R_n tương ứng. Khi đó ước lượng $p_n(\mathbf{x})$ cho $p(\mathbf{x})$ ở công thức (6.5f) trở thành:

$$p_n(\mathbf{x}) = \frac{k_n/n}{V_n}. \quad (6.5g)$$

Để thấy rằng nếu $p(\mathbf{u})$ liên tục tại \mathbf{x} và các điều kiện sau thỏa mãn thì $p_n(\mathbf{x})$ hội tụ tới $p(\mathbf{x})$:

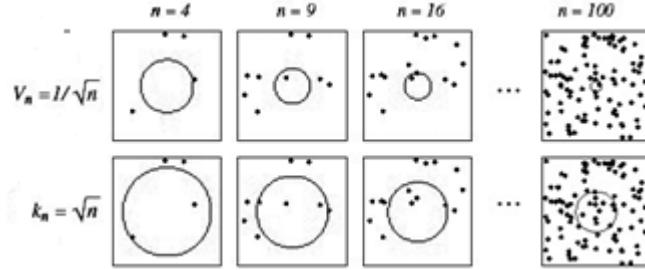
$$\text{a)} \lim_{n \rightarrow \infty} V_n = 0 \quad (6.5h1)$$

$$\text{b)} \lim_{n \rightarrow \infty} k_n = \infty \quad (6.5h2)$$

Tuy nhiên, trong thực tế, số lượng mẫu n có hạn và ta chỉ xác định được miền R đủ nhỏ thay cho dãy miền R_n . Khi đó nếu thể tích V của R quá bé thì số mẫu rơi vào miền R quá ít và ước lượng của công thức (6.5g) ít tin cậy, ngược lại nếu V lớn thì miền R rộng và số mẫu rơi vào miền này nhiều nhưng công thức (6.5a) sẽ cho ước lượng sai số lớn. Mâu thuẫn này được xử lý theo 2 hướng.

1) Phương pháp cửa sổ Parzen. Cố định thể tích V_n phụ thuộc tổng số mẫu n tức là mẫu càng nhiều thì thể tích càng bé (chẳng hạn $V_n = 1/\sqrt{n}$), sau đó xác định k_n là số điểm dữ liệu rơi vào miền R_n có thể tích V_n này.

2) Phương pháp k láng giềng gần nhất (k -NN). Cố định số điểm dữ liệu k_n rơi vào miền R_n phụ thuộc vào n (chẳng hạn $k_n = \sqrt{n}$) và tính thể tích V_n . Hai cách tiếp cận nêu trên với $V_n = 1/\sqrt{n}$ và $k_n = \sqrt{n}$ được minh họa trong hình 6.1, trong đó phương pháp cửa sổ Parzen ứng với hàng trên và k -NN ứng với hàng dưới.



Hình 6.1. Hai phương pháp xác định hàm mật độ tại điểm tâm ô vuông, R_n là hình tròn, $V_n = 1/\sqrt{n}$ và $k_n = \sqrt{n}$. Phương pháp cửa sổ Parzen ứng với hàng trên.

1) Phương pháp cửa sổ Parzen

Phương pháp này dùng *hàm cửa sổ* (*window function*) để xác định dãy R_n và tính $k(n)$. Có nhiều cách chọn hàm cửa sổ, cách đơn giản là chọn cửa sổ lập phương.

Để xác định $p(x)$ ta chọn miền R_n là một hình siêu lập phương (hypercube) đủ bé với tâm x . Nếu R_n là siêu lập phương d -chiều có độ dài cạnh là $h(n)$ thì thể tích của nó là:

$$V(n) = h^d(n) \quad (6.6a)$$

Hàm cửa sổ (*window function*) $\varphi(u)$ trên siêu lập phương đơn vị với tâm tại gốc tọa độ được xác định như sau:

$$\varphi(u) = \begin{cases} 1 & |u_j| \leq \frac{1}{2} \quad \forall j = 1, \dots, d \\ 0 & \text{ngược li} \end{cases}. \quad (6.6b)$$

Khi đó số $k(n)$ các điểm của tập n mẫu rơi vào siêu lập phương R_n là:

$$k(n) = \sum_{i=1}^n \varphi\left(\frac{x - x_i}{h(n)}\right). \quad (6.6c)$$

Đại lượng $k(n)$ cho ước lượng $p_n(x)$ của $p(x)$ theo công thức (6.5g) và được viết lại như sau:

$$\bar{p}_n(x) = \frac{k(n)}{nV(n)} = \frac{1}{nV(n)} \sum_{i=1}^n \varphi\left(\frac{x - x_i}{h(n)}\right) \quad (6.6d)$$

Công thức này cho thấy số điểm dữ liệu tích tụ quanh x càng nhiều thì ước lượng của $p(x)$ càng lớn. Hàm $p(x)$ có thể xác định nhờ nội suy dựa trên kết quả tính được ở

một số điểm. Việc xác định $h(n)$ rất quan trọng vì nó quyết định lượng điểm rơi vào miền R và do đó quyết định độ chính xác của ước lượng, Dudart gợi ý nên chọn $h(n)$:

$$h(n) = V_1 / \sqrt{n}^{\gamma_d} \quad (6.6e)$$

trong đó V_1 là thể tích của một siêu lập phương chọn trước, chẳng hạn có độ dài cạnh bằng 1.

2) Phương pháp ước lượng k-láng giềng gần nhất (k -NN).

Như đã nói ở trên, trong phương pháp cửa sổ Parzen khi số điểm rơi vào cửa sổ quá ít thì ước lượng không đủ in cậy. Một cách tiếp cận khác là cố định $k(n)$ số điểm rơi vào cửa sổ phụ thuộc theo n , sau đó điều chỉnh miền có tâm x sao cho miền này chứa $k(n)$ điểm dữ liệu. Lúc đó miền này có thể tích $V(n)$ và giá trị hàm mật độ $p(x)$ tại x được ước lượng bằng $\bar{p}_n(x)$ theo công thức:

$$\bar{p}_n(x) = \frac{k(n)/n}{V(n)} \quad (6.6f)$$

Dễ thấy rằng khi tập dữ liệu có nhiều điểm quanh x thì $V(n)$ bé và do đó $p(x)$ lớn, ngược lại nếu ít điểm quanh x thì $V(n)$ lớn và $p(x)$ bé. Điều kiện cần và đủ để $p_n(x)$ hội tụ tới $p(x)$ theo xác suất tại tất cả các điểm mà $p(x)$ là liên tục là $\lim_{n \rightarrow \infty} k_n = \infty$ và $\lim_{n \rightarrow \infty} k_n/n = 0$, Dudart gợi ý dùng $k(n) = \sqrt{n}$.

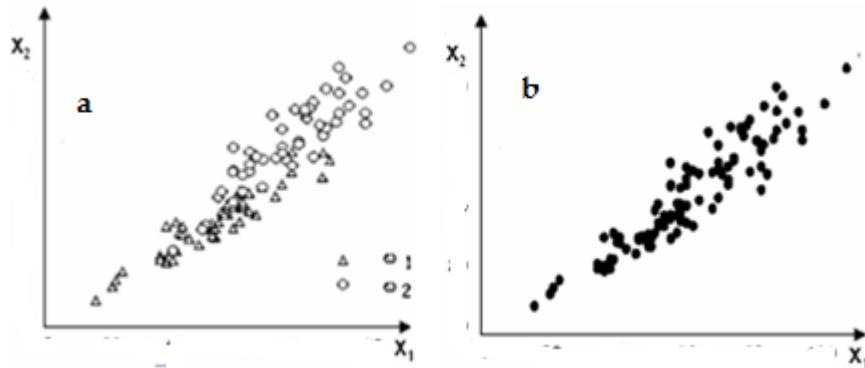
6.2. PHÂN CỤM DỮ LIỆU

Trong các chương trước ta đã xét bài toán phân lớp có giám sát (supervised learning), trong đó ta có thông tin về nhãn (lớp) của một số mẫu đào tạo hoặc phân bố của lớp trong không gian đặc trưng. Trong mục này ta xét bài toán phân lớp chỉ dựa vào phân tích cấu trúc nội tại của tập đối tượng dữ liệu được xét. Bài toán này được gọi là phân lớp không giám sát (unsupervised classification) hay phân cụm dữ liệu (data clustering).

6.2.1. Bài toán phân cụm dữ liệu

Trong bài toán phân cụm dữ liệu, dựa vào phân tích tính tương tự của các đối tượng trong tập dữ liệu được xét, ta chia nó thành các cụm sao cho các đối tượng trong cùng một cụm thì giống nhau hơn khi so với các đối tượng khác cụm. Hình 6.2 minh họa bài toán phân cụm tập dữ liệu đã xét trong hình 4.10 (hình a) khi không biết trước nhãn lớp (hình b). Để dàng nhận thấy trong bài toán phân cụm (không có *thầy*) có thể có nhiều cách phân cụm và kết quả phân cụm không đảm bảo chắc chắn. Bài toán này thuộc loại *thiết lập không đúng đắn*. Tuy nhiên, có nhiều ứng dụng và

phân cụm giúp ta giảm được độ phức tạp khi xử lý dữ liệu lớn nên hiện nay đang là chủ đề được nhiều người quan tâm.



Hình 6.2. Tập dữ liệu có giám sát (a) khi chưa biết nhãn lớp (b)

Đặc biệt, bài toán phân cụm đang được ứng dụng rộng rãi trong các lĩnh vực sau.

Kỹ thuật phân lớp.

- Tóm tắt và giải thích dữ liệu bài toán trong các trường hợp mà dữ liệu cần được giải thích theo nhóm chứ không phải theo từng mẫu. Trong nhiều bài toán việc phân tích theo cụm cũng rất có ý nghĩa, chẳng hạn, phân tích tiến hóa sinh học có thể thực hiện theo từng loài hoặc các nhóm loài gần nhau.
- Tạo mẫu cho tiếp cận thống kê. Để giảm thời gian và chi phí thu thập dữ liệu, việc phân cụm dữ liệu thường được áp dụng cho giai đoạn đầu để ước lượng phân phối cho các tập mẫu nhỏ
- Tạo tâm cho các nhóm nhân tạo trong các bộ phân lớp. Khi xây dựng các mạng neural, người ta thường dùng vectơ trung bình của các cụm làm tâm neural để nhận biết các mẫu có đặc trưng gần nó.

Khám phá tri thức từ dữ liệu lớn. Trong những bài toán phải xử lý dữ liệu rất lớn, nhiều khi các thuật toán với độ phức tạp thời gian đa thức cũng không dùng được. Khi đó việc chia dữ liệu thành nhiều cụm để xử lý là một phương pháp hữu hiệu

Sinh học: Trong sinh học, việc phân cụm dữ liệu đang được sử dụng để xác định quan hệ giữa các loài, phân loại biểu diễn gene.

Tùy theo đặc điểm về tính tương đồng của các đối tượng trong các bài toán đang xét, có nhiều cách tiếp cận cho thuật toán phân cụm. J. Han và M. Kamber phân các thuật toán này làm 4 loại chính.

- Phương pháp phân cấp (Hierarchical Data Clustering).
- Phương pháp phân hoạch(Partition Based Data Clustering).
- Phương pháp dựa trên mật độ (Density Based Data Clustering).

- Phương pháp dựa trên lưới (Grid Based Data Clustering).

Trong đó, thông dụng nhất là hai phương pháp phân cấp và phân hoạch. Ở đây chỉ giới thiệu hai phương pháp này.

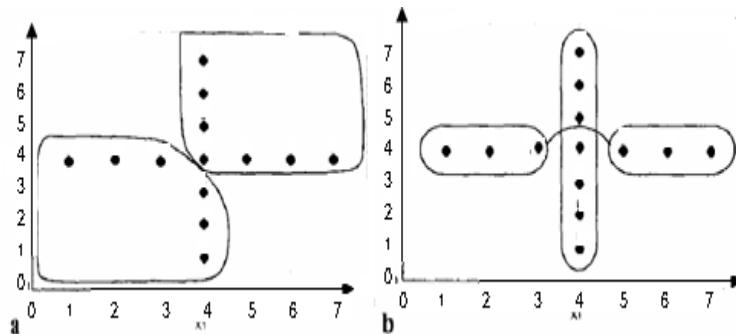
Trước khi giới thiệu các phương pháp phân cụm, ta xem xét vấn đề chuẩn hóa dữ liệu.

6.2.2 . Vấn đề chuẩn hóa dữ liệu

Tính tương đồng của các đối tượng dữ liệu thường được xác định nhờ các khoảng cách trong tập dữ liệu, khoảng cách giữa hai đối tượng càng bé thì tính tương đồng càng cao. Kết quả phân cụm phụ thuộc nhiều vào việc chọn metric và đơn vị chia cho mỗi đặc trưng của dữ liệu. Hình 6.3 biểu diễn tập dữ liệu hình chữ thập, nếu chia tập dữ liệu thành hai cụm theo tiêu chuẩn cực tiểu sai số trung bình

$$E = \sum_{i=1}^2 \frac{1}{n_i} \sum_{x,y \in \omega_i} d(x, y)$$

và dùng chuẩn Euclidean thì ta được hai cụm ở hình **a** còn dùng chuẩn Manhattan thì ta được hai cụm ở hình **b**.



Hình 6.3. Dữ liệu chữ thập chia làm 2 cụm với các metric khác nhau: a) Euclidean; b) Manhattan

Dễ thấy rằng việc chọn đơn vị cho đặc trưng ảnh hưởng tới khoảng cách của các đối tượng nên cũng ảnh hưởng tới kết quả phân cụm. Một cách hạn chế ảnh hưởng này là chuẩn hóa dữ liệu. Nói chung không phải dữ liệu nào cũng cần chuẩn hóa, đặc biệt với các bài toán cần thông tin từ đặc trưng gốc.

Sau đây là một số cách thông dụng để chuẩn hóa các đặc trưng của dữ liệu:

$$\bullet \quad y_i = (x_i - m_i)/s_i, \quad (6.7a)$$

trong đó m_i , s_i tương ứng là trung bình và độ lệch chuẩn của đặc trưng x_i ;

$$\bullet \quad y_i = (x_i - \min\{x_i\})/(\max\{x_i\} - \min\{x_i\}); \quad (6.7b)$$

$$\bullet \quad y_i = x_i/(\max\{x_i\} - \min\{x_i\}); \quad (6.7c)$$

$$\bullet \quad y_i = x_i/a \quad (6.7d)$$

Tóm lại ta cần lưu ý ba điểm sau:

- Để thực hiện phân cụm, ta cần chọn metric thích hợp
- Metric thích hợp phụ thuộc vào phương pháp chuẩn hóa
- Để chọn phương pháp chuẩn hóa ta cần biết về kiểu cụm muốn có.

6.3.3. Phương pháp phân cấp

Phương pháp phân cấp (Hierarchical Data Clustering) còn được gọi là phương pháp phân cụm cây, trong đó sắp xếp một tập dữ liệu đã cho thành một cấu trúc có dạng hình cây, cây phân cấp này được xây dựng theo kỹ thuật đệ quy. Cây phân cụm có thể được xây dựng theo hai phương pháp tổng quát: phương pháp dưới lên (Bottom up) và phương pháp trên xuống (Top down). Các thuật toán theo phương pháp dưới lên còn gọi là thuật toán trộn (merging algorithm) còn phương pháp trên xuống còn được gọi là phương pháp tách.

Với quy tắc liên kết để chọn cặp cụm trộn cho trước, thuật toán trộn được mô tả trong bảng 6.1.

Bảng 6.1. Thuật toán phân cụm trộn

- | |
|--|
| <ol style="list-style-type: none"> Khởi tạo mỗi phần tử làm một cụm $\omega_i = \{x_i\}$ Khi $c \geq 1$ thực hiện lặp: <ol style="list-style-type: none"> Xác định hai cụm gần nhất ω_i và ω_j theo quy tắc đã chọn; Trộn ω_i và ω_j thành $\omega_{ij} = \omega_i \cup \omega_j$ //còn $c-1$ cụm; $c \leftarrow c - 1$ |
|--|

. **Ví dụ.** Trong mục này ta giả thiết đã có quy tắc liên kết và không bàn cụ thể tới cách chọn cụm trộn. Phương pháp "dưới lên" phân cụm tập dữ liệu $S=\{A,B,C,D,E,F,G\}$ có thể thực hiện như sau:

Bước 0: Mỗi đối tượng dữ liệu được gán cho mỗi cụm, như vậy các cụm ban đầu là $\{A\}, \{B\}, \{C\}, \{D\}, \{E\}, \{F\}, \{G\}$;

Bước 1: $\{B\}$ và $\{C\}$ được gộp vào thành một cụm lớn hơn là $\{B,C\}$ và các cụm thu được là : $\{A\}, \{B,C\}, \{D\}, \{E\}, \{F\}, \{G\}$;

Bước 2: gộp các cụm $\{D\}, \{E\}$ thành $\{D,E\}$. Các cụm thu được là $\{A\}, \{B,C\}, \{D,E\}, \{F\}, \{G\}$;

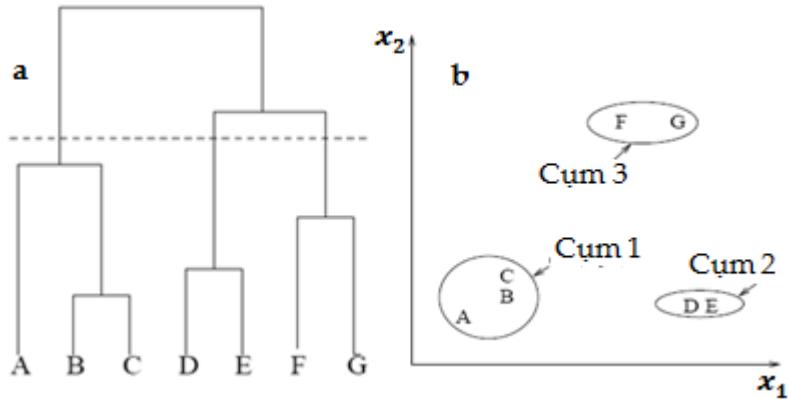
Bước 3; Gộp các cụm $\{F\}, \{G\}$ thành $\{F,G\}$. Các cụm thu được là $\{A\}, \{B,C\}, \{D,E\}, \{F,G\}$;

Bước 4: Gộp cụm hai cụm $\{A\}, \{B, C\}$ thành $\{A,B,C\}$. Thu được 3 cụm là $\{A,B,C\}, \{D,E\}, \{F,G\}$;

Bước 5: Gộp hai cụm $\{D, E\}$ và $\{F, G\}$ thành $\{D, E, F, G\}$. Thu được hai cụm $\{A, B, C\}, \{D, E, F, G\}$.

Gộp hai cụm còn lại ta thu được một cụm.

Quá trình này được mô tả trong hình 6.4. Cây được mô tả trong hình 6.4a, nếu lấy mức theo đường đứt, ta thu được 3 cụm như trong hình 6.4b.



Hình 6.4. Phân cụm phân cấp tập $S=\{A, B, C, D, E, F, G\}$ theo phương pháp dưới lên

Các quy tắc liên kết

Trong các thuật toán trộn, hai cụm ω_i và ω_j được chọn để trộn là cặp có "khoảng cách" (hoặc giả khoảng cách) $d(\omega_i, \omega_j)$ nhỏ nhất. Các khoảng cách này được định nghĩa khác nhau cho các thuật toán và kết quả phân cụm khác nhau.

Sau đây là một số quy tắc liên kết.

1) *Liên kết đơn* (Nearest neighbour). Còn gọi là quy tắc láng giềng gần nhất và được ký hiệu là NN. Trong quy tắc này khoảng cách $d(\omega_i, \omega_j)$ của hai cụm ω_i, ω_j là khoảng cách nhỏ nhất của hai mẫu tương ứng thuộc chúng:

$$d(\omega_i, \omega_j) = \min \|x - y\| : x \in \omega_i, y \in \omega_j \quad (6.8a)$$

2) *Liên kết đầy* (Furthest neighbour). Còn gọi là phương pháp láng giềng xa nhất và được ký hiệu là FN, trong đó khoảng cách $d(\omega_i, \omega_j)$ của hai cụm ω_i, ω_j là khoảng cách lớn nhất của hai mẫu tương ứng thuộc chúng:

$$d(\omega_i, \omega_j) = \max \|x - y\| : x \in \omega_i, y \in \omega_j \quad (6.8b)$$

3) *Liên kết trung bình giữa các nhóm*. (un-weighted pair-group method using arithmetic averages). Được ký hiệu là quy tắc UPGMA, trong đó khoảng cách $d(\omega_i, \omega_j)$ của hai cụm ω_i, ω_j là khoảng cách trung bình của các cặp mẫu tương ứng thuộc chúng:

$$d(\omega_i, \omega_j) = \frac{1}{n_i n_j} \sum_{\mathbf{x} \in \omega_i} \sum_{\mathbf{y} \in \omega_j} \|\mathbf{x} - \mathbf{y}\| \quad (6.8c)$$

4) *Liên kết trung bình trong phạm vi nhóm.* (un-weighted within-group method using arithmetic averages). Được ký hiệu là quy tắc UWGM, trong đó khoảng cách $d(\omega_i, \omega_j)$ của hai cụm ω_i, ω_j là khoảng cách trung bình của các cặp mẫu thuộc cụm sau khi đã trộn:

$$d(\omega_i, \omega_j) = \frac{1}{C_{n_i+n_j}} \sum_{\mathbf{x}, \mathbf{y} \in \omega_i \cup \omega_j} \|\mathbf{x} - \mathbf{y}\| \quad (6.8d)$$

5) *Phương pháp Ward.* Trong đó khoảng cách $d(\omega_i, \omega_j)$ của hai cụm ω_i, ω_j là bằng trung bình của bình phương khoảng cách của mỗi dữ liệu tới tâm cụm sau khi trộn:

$$d(\omega_i, \omega_j) = \frac{1}{n_i + n_j} \sum_{\mathbf{x} \in \omega_i \cup \omega_j} \|\mathbf{x} - \mathbf{m}\|^2 \quad (6.8e)$$

6.3.4. Phương pháp phân hoạch

Phân hoạch (Partition) là phương pháp phân cụm đang được dùng phổ biến nhất, đặc biệt cho các tập dữ liệu lớn. Lược đồ chung của các thuật toán này như sau: Với tập dữ liệu D gồm n đối tượng trong không gian d chiều, và số lượng cụm k thường được xác định trước hoặc đặt dưới dạng tham số, người ta thực hiện một quá trình lặp để phân các đối tượng dữ liệu thành k cụm cực tiểu mục tiêu chọn trước. Sau đây là một số thuật toán loại này.

1) Thuật toán k-means.

a) Lược đồ tổng quát

Thuật toán K-Means (MacQueen, 1967) và các biến thể của nó đang được sử dụng nhiều nhất. Thuật toán này phân tập dữ liệu thành k cụm nhờ dùng giá trị trung bình của các đối tượng trong một cụm là tâm cụm nên gọi là k-mean. Hàm mục tiêu được sử dụng trong thuật toán là tổng bình phương độ lệch của các đối tượng dữ liệu đến tâm cụm:

$$E = \sum_{j=1}^k \sum_{x \in \omega_j} \|\mathbf{x} - \mathbf{m}_j\|^2 \quad (6.9a)$$

trong đó \mathbf{m}_j là tâm cụm ω_j . Với số lượng cụm k cho trước, thuật toán này được mô tả trong bảng 6.2.

Điều kiện dừng của thuật toán thường chọn một trong các điều kiện sau:

1. $t = T_{max}$ trong đó T_{max} là số lần lặp cho trước
2. $|E^t - E^{t-1}| < \Delta$ trong đó Δ là hằng số bé cho trước

3. Tới khi các cụm không đổi.

Bảng 6.2. Thuật toán k-means

- Bước 1. Khởi tạo chọn ngẫu nhiên k tâm cụm;
- Bước 2. Lặp khi điều kiện dừng chưa thỏa mãn
- 2.1. Gán mỗi đối tượng vào cụm mà nó gần tâm nhất;
 - 2.2. Tính lại các tâm cụm của bước 2.1.

Khi tập dữ liệu không quá lớn thì người ta dùng điều kiện dừng 3 và lời giải tìm được là cực trị địa phương của E .

b) Khi dữ liệu là hỗn hợp các tập dữ liệu có phân bố chuẩn nhiễu chiều

Trong thực tế, ta hay gặp trường hợp tập dữ liệu là hỗn hợp của các tập con có phân bố chuẩn nhiễu chiều, tức là lớp ω_i có phân bố $N(\mathbf{m}_i, \Sigma_i)$. Khi đó khoảng cách của mỗi điểm \mathbf{x}_i đến tâm cụm \mathbf{m}_j của cụm ω_j được tính theo công thức:

$$d(\mathbf{x}_i, \mathbf{m}_j) = \|\mathbf{x}_i - \mathbf{m}_j\| = ((\mathbf{x}_i - \mathbf{m}_j)\Sigma_i^{-1}(\mathbf{x}_i - \mathbf{m}_j))^{\frac{1}{2}} \quad (6.9b)$$

trong đó, các ma trận hiệp phương sai Σ_i được ước lượng theo công thức (6.4h) từ dữ liệu huấn luyện của mỗi lớp nếu có trước tập mẫu cho mỗi lớp.

Thuật toán k-mean có độ phức tạp là $O(nkt)$ trong đó t là số lần lặp trong bước 2. Vì độ phức tạp của thuật toán thấp nên rất thích hợp khi phải xử lý dữ liệu lớn. Tuy nhiên, thuật toán có các nhược điểm của thuật toán k-mean cơ bản sau:

- 1- Nếu tâm khởi tạo không tốt thì có cụm rỗng (hay tâm chết) vì trong mọi lần lặp ở bước 2 không có đối tượng nào gần tâm này.
- 2- Khi dữ liệu có nhiễu thì có thể làm lệch tâm nên thuật toán này không ổn định.
- 3- Thường thì thuật toán chỉ cho lời giải là cực trị địa phương của E .

Để khắc phục nhược điểm thứ 3, người ta thường khởi tạo nhiều bộ tâm để chạy thuật toán nhiều lần và chọn lời giải có E nhỏ nhất. Hai nhược điểm đầu có thể khắc phục bằng thuật toán k-centroid dưới đây .

2) Thuật toán phân cụm k-centroid

Thuật toán này khởi tạo các tâm cụm là các đối tượng chọn ngẫu nhiên trong tập dữ liệu và thay cho việc lấy giá trị trung bình làm tâm, nó dùng một đối tượng trong một cụm gần tâm nhất làm tâm cụm. Điều này làm cho thuật toán K-centroid tránh được ảnh hưởng của nhiễu và phần tử ngoại lai. Với k cho trước và điều kiện dừng như thuật toán k-mean, thuật toán k-centroid thực hiện tương tự như k-mean và được mô tả trong bảng 6.3.

Bảng 6.3. Thuật toán k- centroid

Bước 1. Khởi tạo ngẫu nhiên k đối tượng trong D làm tâm cụm;

Bước 2. Lặp khi điều kiện dừng chưa thỏa mãn:

2.1 Gán mỗi đối tượng vào cụm mà nó gần tâm nhất;

2.2. Tính trung bình các cụm mới ở bước 2.1. và chọn phần tử trong cụm gần nó nhất làm tâm mới.

Nhờ thay đổi cách khởi tạo và lấy phần tử tâm người ta đưa ra nhiều biến thể của thuật toán này, chẳng hạn: PAM (Partitioning Around Medoids); CLARA (Clustering Large Applications); CLARANS....

3) Thuật toán phân cụm ngưỡng.

Các thuật toán loại k-mean thường đòi hỏi biết trước số cụm và cho phép bán kính của mỗi cụm lớn tùy ý. Thuật toán phân cụm ngưỡng có cách tiếp cận khác hơn, nó xác định trước một ngưỡng “bán kính” cụm Δ và các *tâm/phân tử tâm* được khởi tạo tuân tự bằng cách tăng thêm một *tâm/phân tử tâm* mới mỗi khi khoảng cách mẫu mới tới tâm gần nhất lớn hơn ngưỡng này. Với ngưỡng Δ cho trước và điều kiện dừng như thuật toán k-mean, thuật toán thực hiện như trong bảng 6.4.

Bảng 6.4. Thuật toán phân cụm ngưỡng

Bước 1. Chọn z_1 thuộc D làm tâm cụm thứ nhất;

Bước 2. $\forall x \in D$, nếu khoảng cách tới tâm gần nhất nhỏ hơn Δ thì nó thuộc cụm có tâm này, ngược lại nó sẽ là tâm cụm mới;

Bước 3. Lặp khi điều kiện dừng chưa thỏa mãn:

3.1. Tính lại tâm cụm như thuật toán centroid;

3.2. Phân lại các mẫu vào cụm gần tâm nhất

Các thuật toán trên dùng cho dữ liệu có các thuộc tính nhận giá trị thực. Khi dữ liệu có các thuộc tính định danh thì ta không tính được giá trị trung bình theo cách thông dụng. Thuật toán k-tâm được dùng cho trường hợp này.

4) Thuật toán k-tâm.

Trong thuật toán này, ta khái niệm mode thay cho trọng tâm của mỗi tập dữ liệu có thuộc tính nhận giá trị định danh. Xét tập dữ liệu D gồm n đối tượng $\{x^i\}_{i=1}^n$ trong đó $x^i = (x_1^i, \dots, x_d^i)$ có các thành phần x_j^i ($\in Dom(A_j)$) của thuộc tính A_j là giá trị định danh với mọi $j \geq m$, còn khi $j < m$ thì x_j^i nhận giá trị thực. Ta gọi dữ liệu như vậy là *dữ liệu hỗn hợp*.

Mode của tập dữ liệu.

Định nghĩa mode của tập dữ liệu hỗn hợp, dựa trên khái niệm j -mode của các thuộc tính A_j ($j \leq d$).

Định nghĩa 6.1. Cho C là tập con của tập dữ liệu hỗn hợp D .

i) Với mọi $j \leq n$, j -mode của C (kí hiệu là j -mode(C)) là giá trị có tần suất nhiều nhất trong tập giá trị thuộc tính A_j của C khi A_j là thuộc tính định danh và là trung bình cộng của các giá trị thuộc tính này của C khi A_j là thuộc tính số. Nếu A_j là thuộc tính định danh và tập giá trị thuộc tính này của C có nhiều giá trị có tần suất lớn nhất thì j -mode(C) có thể không duy nhất và ta chọn giá trị nào cũng được.

ii) Mode của tập hợp C ký hiệu là mode(C) là phần tử $z = (z_1, \dots, z_d)$ trong đó

$$z_j = j\text{-mode}(C), \forall j \leq n \quad (6.10a)$$

Để tính khoảng cách của các đối tượng, ta cần định nghĩa metric trên dữ liệu hỗn hợp. Metric này được định nghĩa dựa trên metric của từng thuộc tính.

Định nghĩa 6.2. Giả sử $\text{DOM}(A_j)$ là miền giá trị của thuộc tính A_j . $\forall x, y \in \text{DOM}(A_j)$ hàm $d_j(x, y)$ xác định như sau

i) Nếu A_j là thuộc tính số thì $d_j(x, y) = |x - y|$ (6.10b)

ii) Nếu A_j là thuộc tính định danh thì $d_j(x, y) = \begin{cases} 0 & \text{khi } x = y \\ 1 & \text{khi } x \neq y \end{cases}$ (6.10c)

Bây giờ ta định nghĩa khoảng cách cho mọi cặp đối tượng dữ liệu hỗn hợp.

Định nghĩa 6.3. Giả sử $x = (x_1, \dots, x_d)$ và $y = (y_1, \dots, y_d)$ là hai đối tượng dữ liệu hỗn hợp trong D , khoảng cách $d(x, y)$ được tính bởi công thức:

$$d(x, y) = \sqrt{\sum_{j=1}^d \rho_j^2 d_j^2(x_j, y_j)} \quad (6.10d)$$

trong đó các $d_j(x, y)$ được tính theo các công thức (6.10b-c) và ρ_j là các trọng số dương cho bởi các chuyên gia tùy theo mức quan trọng của thuộc tính.

Đặc tả thuật toán k-tâm

Với định nghĩa Mode và khoảng cách cho tập dữ liệu hỗn hợp đã nêu, thuật toán k-tâm thực hiện như thuật toán k-means hoặc k-centroid ở trên. Thuật toán được đặc tả như trong hình 6.5.

Procedure k-tâm**Begin**Chọn các trọng số ρ_j , các hàm f_j , xác định k.Chọn k phần tử ban đầu $\frac{z^1}{z^k}$ của D làm tâm các cụmXếp mỗi $x \in D$ vào cụm C_j mà nó gần tâm nhất;**For** $j=1, \dots, k$ **do** $z_j \leftarrow \text{mode}(C_j)$;**Repeat**

Phân bố lại cụm theo tâm mới// như k-means;

Cập nhật lại tâm cho các cụm // nhờ tính mode

Until các cụm không đổi;

Xác định các cụm;

End**Hình 6.5** Đặc tả thuật toán k-tâm**Xử lý thuộc tính thứ tự**

Nhiều trường hợp, thuộc tính định danh có giá trị được sắp thứ tự toàn phần (sắp thẳng), chẳng hạn : $\text{DOM}(A_j) = \{\text{không đau, hơi đau, đau và rất đau}\}$, ta nói thuộc tính này là thuộc tính thứ tự. Khi đó ta có thể áp dụng định nghĩa khoảng cách sau cho thuộc tính này để tăng chất lượng thuật toán.

Định nghĩa 6.4. Giả sử A_j là thuộc tính thứ tự và $\text{DOM}(A_j) = a_j^1, \dots, a_j^k$ trong đó $a_j^1 < a_j^2 < \dots < a_j^k$, ta lấy một hàm đơn điệu tùy ý $f_j : \text{DOM}(A_j) \rightarrow [0,1]$ sao cho $f_j(a_j^1) = 0$ và $f_j(a_j^k) = 1$ (Hàm này có thể là: $f_j(a_j^i) = \frac{i-1}{k-1}$). Khi đó khoảng cách của hai giá trị x, y trong $\text{DOM}(A_j)$ được xác định bởi :

$$d_j(x, y) = |f_j(x) - f_j(y)|. \quad (6.10e)$$

Công thức này được áp dụng trong (6.10d) để tính khoảng cách của hai đối tượng dữ liệu. Cũng dựa trên định nghĩa này, ta luôn có thể xem các thuộc tính thứ tự có miền giá trị là đoạn $[0,1]$ để tìm mode (các giá trị trên thuộc tính này của D là tập con) và nó cũng được xem là thuộc tính số khi không xảy ra nhầm lẫn. Khi cần ta sẽ chuyển ảnh của nó về giá trị định danh gốc.

Lưu ý rằng khi xem miền giá trị của các thuộc tính có thứ tự là đoạn $[0,1]$ và thì $\text{mode}(C)$ cực tiểu hàm:

$$E(\mathbf{x}) = \sum_{y \in C} d^2(y, x) \quad (6.11)$$

Nhận xét. 1) Khi thuật toán k -tâm kết thúc, các đối tượng tâm có thể không thuộc tập D do ta đã chuyển đổi giá trị của thuộc tính có thứ tự. Để tìm phần tử đại diện cho mỗi cụm, ta lấy phần tử thuộc cụm gần với tâm của nó nhất.

2) Trong thực tế, mỗi thuộc tính số có thể dùng những đơn vị đo khác nhau nên việc chọn trọng số ρ_i thích hợp cho mỗi thuộc tính là cần thiết.

6.3.5. Phân cụm bán giám sát

Như đã nói ở trên, nhược điểm của các phương pháp học không giám sát là kết quả không chắc chắn, vì vậy khó đáp ứng đầy đủ yêu cầu người dùng. Để tăng hiệu quả sử dụng cho các thuật toán, một hướng tiếp cận mới là sử dụng thêm thông tin hợp tác từ người dùng và được gọi là *phân cụm bán giám sát* (semi-supervised clustering). Các với thông tin bổ trợ này thường có được nhờ quan sát một tập con dữ liệu và gán nhãn (phân cụm) cho chúng hoặc xác định được các đối tượng nào thuộc cùng một cụm (*must-link*) hay không thể (*cannot-link*) để cùng một cụm.

Dưới đây giới thiệu thuật toán đơn giản dựa trên thuật toán k -mean gọi là Seeded-KMeans.

Thuật toán Seeded-KMeans

Thuật toán này được Basu và các cộng sự (2002) đề xuất áp dụng cho trường hợp khi trong tập D có tập con $S = \bigcup_{h=1}^k S_h$ gọi là tập giống, trong đó mỗi tập con S_i nên để cùng một cụm C_i . Thuật toán Seeded-KMeans sử dụng thông tin bổ trợ từ tập giống để khởi tạo cho thuật toán K -Means hoặc biến thể của nó. Trong thực tế, nhiều khi không tìm đủ tập giống cho tất cả k cụm, khi đó các cụm còn lại được khởi tạo ngẫu nhiên trên phần bù của S trong D . Với tập dữ liệu D trong không gian d-chiều đã cho, số cụm k và tập giống $S = \bigcup_{h=1}^k S_h$ đã biết, thuật toán Seeded-KMeans hoạt động tương tự như k -mean và dễ dàng mở rộng cho các biến thể của nó. Với các điều kiện dùng như thuật toán k -mean, thuật toán này được mô tả trong bảng 6.5.

Nhờ thông tin bổ trợ từ tập giống mà thuật toán Seeded-KMeans có nhiều ưu điểm vượt trội so với thuật toán K -Means. Kết quả thí nghiệm cho thấy Seeded-KMeans luôn cho chất lượng xét theo hàm mục tiêu E cho bởi (6.9a) và khả năng kháng nhiễu.

Bảng 6.5. Thuật toán Seeded-KMeans

Bước 0. Nhập D, k, S

Bước 1. Khởi tạo các tâm cụm: $\mu_j^0 \leftarrow \frac{1}{|S_j|} \sum_{x \in S_j} x \quad \forall j = 1, \dots, k ; t \leftarrow 0$.

Bước 2. Lặp cho tới khi thỏa mãn điều kiện dừng:

2.1. Gán mỗi đối tượng dữ liệu x vào cụm h^* nó gần tâm nhất;

2.2. Ước lượng tâm: $\mu_h^{(t+1)} \leftarrow \frac{1}{|X_h^{(t+1)}|} \sum_{x \in X_h^{(t+1)}} x$

2.3. $t \leftarrow t+1$

KẾT LUẬN

Ước lượng hàm mật độ và phân cụm dữ liệu là hai bài toán học không giám sát thường gặp. Có hai cách tiếp cận để ước lượng hàm mật độ: kỹ thuật có tham số và phi tham số. Trong kỹ thuật có tham số, người ta giả thiết tập mẫu được lấy ngẫu nhiên độc lập cùng phân bố có hàm mật độ có dạng đã biết $p(x/\theta)$ phụ thuộc vào vecto tham số θ và tìm cách ước lượng gần đúng vecto $\hat{\theta}$ của θ dựa trên một tiêu chuẩn nào đó. Có nhiều phương pháp có tham số, trong đó ước lượng có khả năng nhất là phương pháp đơn giản, dễ sử dụng.

Trong thực tế, ta thường không biết được dạng hàm mật độ và phải ước lượng trực tiếp giá trị hàm mật độ tại một tập mốc được xét nhò kỹ thuật phi tham số. Cửa sổ Parzen và k-NN là hai phương pháp thông dụng trong kỹ thuật phi tham số. Cá hai phương pháp này đều dùng một miền nhỏ R có thể tích tùy thuộc vào số mẫu quan sát được và xác định tỷ số điểm thuộc miền trên tổng số dữ liệu để ước lượng mật độ. Phương pháp k-NN đòi hỏi số điểm thuộc miền R đúng với giá trị định trước nên thường chính xác hơn.

Phân cụm dữ liệu thuộc loại bài toán thiết lập không đúng đắn (ill posed), tuy nhiên hiện có nhiều ứng dụng thực tiễn. Kết quả phân cụm phụ thuộc vào việc chọn metric và đơn vị của thuộc tính. Nhiều trường hợp, chuẩn hóa dữ liệu cho ta kết quả phân cụm tin cậy hơn.

Phân cụm phân cấp và phân hoạch là hai cách tiếp cận thông dụng hiện nay. Phân cụm phân cấp theo phương pháp trộn thông dụng hơn phương pháp trên xuống. Kết quả phân cụm phân cấp phụ thuộc nhiều vào quy tắc trộn được chọn. Phải tùy theo đặc điểm bài toán mà chọn quy tắc trộn thích hợp.

Phương pháp phân hoạch điển hình nhất là thuật toán k -mean và các biến thể của nó. Các thuật toán này thực hiện lặp điểu chỉnh cụm nhò phân các đối tượng dữ

liệu vào cụm mà nó gần với trung bình cụm/tâm nhất. Thuật toán cetroid giảm tác động của nhiễu và khắc phục được hiện tượng cụm chết của k-mean.

Khi có thuộc tính định danh, người ta dùng mode của thuộc tính thay cho giá trị trung bình. Nhờ vậy thuật toán k-tâm dùng được cho dữ liệu hỗn hợp.

Khi có thông tin bổ trợ từ người dùng về một tập nhỏ dữ liệu, chẳng hạn, nên cùng hay khác cụm thì chất lượng phân cụm có thể được cải thiện. Trong trường hợp biết được một tập giống gồm các tập con nên phân cùng một cụm, thuật toán Seeded-Kmean dùng tâm các tập con trong tập giống để khởi tạo tâm cho thuật toán k-mean cho chất lượng phân cụm tốt hơn.

BÀI TẬP

1. Tạo ngẫu nhiên 100 điểm (x_n, y_n) trong đó X và Y có phân bố chuẩn $N(1,2)$ và $Y(N(1,3))$.
 - a) Sử dụng kỹ thuật có tham số ước lượng lại phân bố của X và Y.
 - b) Dùng các kỹ thuật phi tham số để ước lượng giá trị $p(1,1)$ và so sánh với kết quả phần a.
2. Cho ví dụ về các tập dữ liệu trong mặt phẳng để minh họa các thuật toán phân cụm phân cấp và phân hoạch đã biết.
3. Giả sử D có N mẫu và phân hoạch thành c cụm $\mathcal{D}_i \subset D$ ($N > c$) theo thuật toán k-mean nhằm cực tiểu hàm mục tiêu:

$$J = \sum_{D_i \neq \emptyset} \sum_{\mathbf{x} \in D_i} \|\mathbf{x} - \mathbf{m}_i\|^2.$$
 - a) Chỉ ra rằng phân hoạch cực tiểu J không có tập con D_i nào rỗng.
 - b) Ta luôn có $J(k) > J(k+1)$, trong đó $J(k)$ là cực tiểu của J khi dùng thuật toán k-mean chia D thành k cụm
4. Tìm ví dụ minh họa việc dùng tiêu chuẩn liên kết khác nhau trong phân cụm phân cấp cho kết quả khác nhau.
5. Tạo hai tập dữ liệu gồm hai lớp có phân chuẩn hai chiều, mỗi lớp 30 dữ liệu, lớp thứ nhất có $\mathbf{m}_1 = \begin{pmatrix} -1 \\ -1 \end{pmatrix}$, $\Sigma_1 = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$ và lớp hai có $\mathbf{m}_2 = \begin{pmatrix} 1 \\ 1 \end{pmatrix}$, $\Sigma_2 = \begin{pmatrix} 2 & 0 \\ 0 & 2 \end{pmatrix}$. Áp dụng thuật toán k-mean với đối tượng khởi tạo gần tâm lớp theo lược đồ tổng quát và khi tính khoảng cách có quan tâm đến hiệp phương sai. So sánh kết quả tìm được.

Chương 7

MẠNG NORON NHÂN TẠO

Mạng noron nhân tạo (Artificial Neural Network: ANN) mô phỏng kiến trúc và hoạt động của hệ thần kinh người, đang được nghiên cứu đa dạng và ứng dụng rộng rãi. Chương này các mạng học có giám sát cơ bản bao gồm perceptron một tầng và nhiều tầng (MLP), mạng hàm cơ sở bán kính (RBF).

7.1. GIỚI THIỆU

Kiến trúc và mô hình hoạt động của mạng noron nhân tạo được xây dựng dựa trên mạng noron sinh học. Trước khi đi vào các dạng mạng noron, ta điểm qua các đặc điểm chính của hệ thần kinh sinh học.

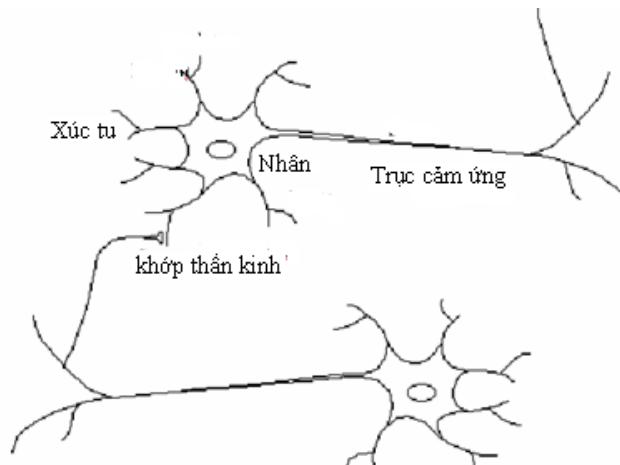
7.1.1. Cấu tạo và đặc điểm của mạng noron sinh học

Nghiên cứu về não và hệ thần kinh được bắt đầu từ đầu thế kỷ 20, và phát triển mạnh mẽ từ giữa thế kỷ này tới nay. Ngày nay, sự phát triển của thần kinh học cho chúng ta hiểu biết nhiều về hệ thần kinh nói chung và cơ chế học của các bộ não sinh học nói riêng. Kết quả nghiên cứu đầu tiên thuộc về Ramón Y Cajal (1911), ông nhận thấy rằng hệ thần kinh cấu tạo từ các noron có kiến trúc tương đối giống nhau và được kết nối đa dạng với nhau bởi để thực hiện các chức năng khác biệt. Mô hình cấu trúc của các noron tự nhiên được mô tả trong hình 7.1, mỗi noron có các thành phần chính sau:

- *Các khớp kết nối* (Synapse). Chúng kết nối các noron với nhau, nhờ đó mà các tín hiệu thần kinh lan truyền được. Cơ chế điều khiển khả năng truyền dẫn ở các khớp này dựa trên các tính chất hóa lý: Điện ↔ Hoá.
- *Các xúc tu* (dendrites). Thu nhận và truyền thông tin về nhân qua khớp kết nối.
- *Nhân* hay *thân tế bào* (cell body). Tổng hợp các tín hiệu nhận được từ các xúc tu và khi tín hiệu tổng hợp này đủ mạnh thì có tín hiệu ra ở trực cảm ứng, lúc

đó ta nói noron cháy. Nếu tín hiệu từ noron A góp phần làm noron B cháy thì ta nói noron A kích hoạt noron B.

- *Trục cảm ứng* (Axon). Cuối trục có nhiều đầu ra có khớp kết nối, khi noron cháy, nó đưa tín hiệu ra từ nhân và truyền tới các noron khác qua các đầu ra này.



Hình 7.1. Mô hình noron sinh học thể hiện qua 2 noron

Mỗi người có khoảng 10^{11} noron, mỗi noron có khoảng 10^4 khớp kết nối. Các khớp khi mới sinh ít kết nối với nhau, chúng được kết nối với nhau nhờ quá trình học. Định đề sau của Hebb (1949) về quá trình học là cơ sở cho ý tưởng thiết kế quá trình học nhờ điều chỉnh trọng số kết nối của mạng mạng noron về sau.

Định đề Hebb. Khi trục thần kinh cảm ứng của một tế bào A đủ gần để tham gia kích hoạt tế bào B và làm cháy nó một cách thường xuyên hoặc lặp đi lặp lại thì xảy ra sự chuyển hóa ở một trong hai tế bào này làm gia tăng tác động kích hoạt của A lên B. Kết quả này của Hebb làm.

Đặc điểm xử lý thông tin của não. Mặc dù đã có nhiều tiến bộ trong thần kinh học, đến nay những hiểu biết chi tiết về cơ chế xử lý thông tin trong bộ não còn rất hạn chế. Tuy nhiên, người ta biết được rằng bộ não xử lý tín hiệu chậm hơn (10^{-3} s) so với các thiết bị vật lý (10^{-9} s), tuy nhiên tốc độ năng lượng hơn và thực hiện được nhiều chức năng hơn. Các tính năng này có được nhờ cơ chế xử lý phi tuyến, song song và phân tán.

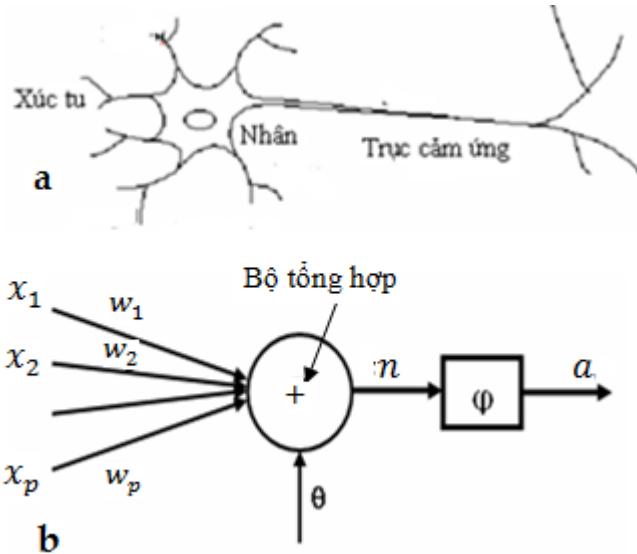
Các mạng noron nhân tạo, về sau sẽ gọi là *mạng noron* cho gọn, dựa trên mô phỏng kiến trúc và các đặc điểm quá trình học của mạng noron sinh học.

7.1.2. Mô hình và kiến trúc mạng noron

Tương tự mạng noron sinh học, các mạng noron được xây dựng nhờ kết nối các thành phần xử lý thông tin có kiến trúc tương tự nhau được gọi là noron.

7.1.2.1. Cấu tạo của nơron

Các nơron là mô phỏng kiến trúc của nơron sinh học, mô hình và kiến trúc của chúng được mô tả trong hình 7.2, trong đó mỗi nơron là một đơn vị xử lý thông tin có kiến trúc như ở hình 7.2.b.



Hình 7.2. Mô hình nơron (b) mô phỏng nơron tự nhiên (a)

Một nơron bao gồm các *liên kết* nhận tín hiệu vào bằng số có các trọng số kết nối w_i tương ứng với tín hiệu vào x_i , giá trị ngưỡng θ , một *bộ tổng hợp* và một *hàm chuyển* (hay *hàm kích hoạt*) φ , để tạo tín hiệu ra dựa trên giá trị hàm tổng.

Liên kết: Mỗi liên kết thứ i sẽ nhận giá trị vào x_i và có trọng số kết nối w_i tương ứng đưa vào bộ tổng hợp.

Trọng số kết nối. Các trọng số kết nối w_i (về sau gọi đơn giản là trọng số khi không gây nên nhầm lẫn) của mỗi liên kết mô phỏng khả năng truyền tín hiệu qua các khớp kết nối của nơron tự nhiên, chính là yếu tố then chốt của nơron. Phương pháp xác định các trọng số này thể hiện phương pháp học.

Bộ tổng hợp. Bộ này tổng hợp tích hợp các tín hiệu vào, trọng số kết nối và giá trị ngưỡng để cho ra hàm n như là thông tin tổng hợp từ tín hiệu vào và trọng số kết nối.

Trường hợp đơn giản hàm này có dạng: $n = \sum_{i=1}^k w_i x_i - \theta$ hoặc $n = \|x - w\|^2$. Ta sẽ dùng dấu $+$ để ký hiệu bộ này.

Hàm chuyển / hàm kích hoạt. Hàm chuyển φ nhận giá trị vào là đầu ra n của bộ tổng hợp và cho giá trị ra $a = \varphi(n)$. Hàm chuyển φ là một hàm không giảm cụ thể nào đó được chọn tùy theo bài toán thực tế và thường phải dễ tính đạo hàm để áp dụng khi

huấn luyện theo phương pháp hiệu chỉnh gradient. Sau đây là một số dạng hàm chuyển thông dụng:

$$Hàm ngưỡng: a = hardlim(n) = \begin{cases} 1 & s \geq 0 \\ 0 & s < 0 \end{cases};$$



$$Hàm ngưỡng đối xứng: a = hardlims(n) = \begin{cases} 1 & n \geq 0 \\ -1 & n < 0 \end{cases};$$



$$Hàm tuyến tính: a = purelin(n) = n$$



Hàm tuyến tính trên đoạn: (satlin)

$$a = satlin(n) = \begin{cases} 1 & n \geq 1 \\ n & 0 \leq n \leq 1 \\ 0 & n \leq 0 \end{cases};$$



Hàm tuyến tính bao giờ đối xứng: (satlins)

$$a = satlins(n) = \begin{cases} 1 & n \geq 1 \\ n & n \in (0,1) \\ -1 & n \leq -1 \end{cases} ;$$



$$Hàm log_sig: a = logsig(n) = \frac{1}{1 + e^{-n}};$$



$$Hàm tanghyperbolic: a = tansig(n) = \frac{e^n - e^{-n}}{e^n + e^{-n}};$$



$$Hàm tuyến tính dương: a = poslin(n) = \begin{cases} n & n \geq 0 \\ 0 & n < 0 \end{cases}$$



$$Hàm cạnh tranh: a = compet(n) = \begin{cases} 1 & n = \max \\ 0 & n < \max \end{cases}$$

Ký hiệu:



7.1.2.2. Các kiểu kiến trúc mạng nơron

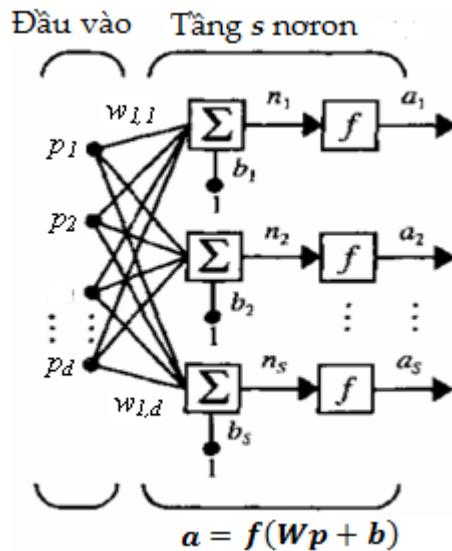
Từ các nơron đơn, người ta kết nối chúng với các tópô khác nhau và cách học các trọng số kết nối khác nhau để có nhiều nhiều kiểu mạng nơron với công dụng phong phú. Trong đó thông dụng nhất là các mạng truyền tới một hoặc nhiều tầng truyền tới ((Feed-forward), mạng hồi quy (recurrent)...).

Các kiểu kiến trúc mạng và tính năng của chúng rất phong phú, giáo trình này chỉ tập trung vào một số mạng truyền tới điển hình, độc giả muốn tìm hiểu sâu hơn có thể tham khảo [12,13]

a) *Mạng một tầng truyền tới*

Mạng này gồm s noron chung tín hiệu vào đầu song song với nhau. Các trọng số được ký hiệu bởi ma trận $W = \begin{pmatrix} w_{11} & \dots & w_{1d} \\ \dots & \dots & \dots \\ w_{s1} & \dots & w_{sd} \end{pmatrix}$ với quy ước rằng chỉ số thứ nhất chỉ

noron còn chỉ số thứ hai chỉ tín hiệu vào, vecto tín hiệu vào p và vecto khuynh hướng b được ký hiệu tương ứng là $p = (p_1, \dots, p_d)^T$, $b = (b_1, \dots, b_s)^T$. Khi đó đầu ra của các bộ tổng hợp là vecto tổng hợp $n = Wp + b$, còn đầu ra của các noron là vecto $a = f(n) = f(Wp + b) = (f(n_1), \dots, f(n_s))^T$ như được mô tả trong hình 7.3.



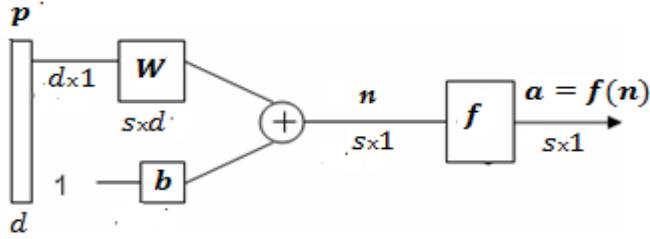
Hình 7.3. Mạng noron một tầng truyền tới

Biểu diễn rút gọn của mạng noron một tầng

Để tiện cho trình bày, mạng một tầng noron ở hình 7.3 được biểu diễn dưới dạng rút gọn cho trong hình 7.4.

b) *Mạng nhiều tầng truyền tới*

Mạng nhiều tầng truyền tới là mạng có hai hoặc nhiều tầng noron. Trong đó các tín hiệu vào đi qua các tầng noron tới tầng ra mà không quay lại tầng trước hoặc tầng chứa nó nên gọi là mạng truyền tới. Nếu mạng có n tầng noron thì ta nói mạng $(n+1)$ tầng hoặc nói rõ có n tầng noron khi cần nhấn mạnh số tầng noron, các tín hiệu vào sẽ được gọi là tầng vào.

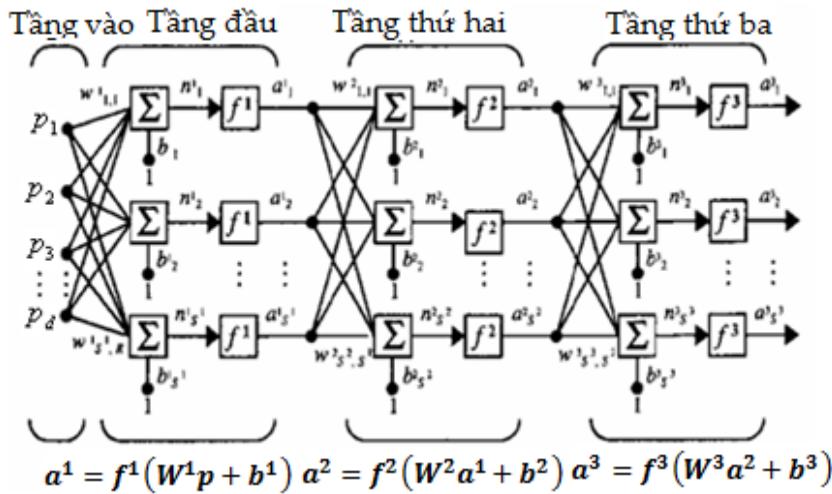


Hình 7.4. Biểu diễn rút gọn của một nơron

Một mạng nơron 4 tầng được minh họa trong hình 7.5, trong đó tầng thứ i có s^i nơron, các đầu ra \mathbf{n}^i và \mathbf{a}^i tương ứng là:

$$\mathbf{n}^i = \mathbf{W}^i \mathbf{p} + \mathbf{b}^i; \quad \mathbf{n}^i = \mathbf{W}^i \mathbf{a}^{i-1} + \mathbf{b}^i; \quad (7.1a)$$

$$\mathbf{a}^i = f(\mathbf{n}^i) = (f(n_1^i), \dots, f(n_{s^i}^i))^T. \quad (7.1b)$$



Hình 7.5. Một mạng truyền tới 4 tầng, tầng thứ i có s^i nơron

c) Mạng hồi quy

Mạng hồi quy khác với các mạng truyền tới ở chỗ các tín hiệu ra ở mỗi tầng nơron có thể quay lại làm đầu vào cho nó hoặc các nơron ở tầng trước.

7.1.3. Đặc điểm của mạng nơron

Nhờ các phương pháp kết nối nơron và cách xác định trọng số kết nối khác nhau, người ta tạo được các mạng nơron như là *các máy/chương trình* có những tính năng tương tự mà mạng sinh học đã có nhờ mô phỏng thể hiện qua các điểm:

- Tri thức thu nhận được nhờ quá trình học. Trong mạng nơron, quá trình học thể hiện qua phương pháp xác định trọng số kết nối hay *luật học*.
- Tính năng có được nhờ kiến trúc mạng và tính chất kết nối

Trong ứng dụng, mạng nơron có thể để dưới dạng chương trình hoặc cài đặt dạng cứng hóa cùng với các phần cứng như song song hóa, bộ trễ, bộ tích phân... và có các đặc điểm chính sau:

1. Phi tuyến. Cho phép xử lý phi tuyến:
2. Cơ chế ánh xạ vào ra ($x \rightarrow d(x)$) để học có giám sát
3. Cơ chế thích nghi. Thay đổi tham số nhò hiệu chỉnh dần để phù hợp với môi trường.
4. Đáp ứng theo mẫu đào tạo. Được thiết kế không những cung cấp thông tin về mẫu đào tạo mà còn cho biết mức tin cậy của nó.
5. Thông tin theo ngũ cảnh. Tri thức được biểu diễn tùy theo trạng thái và kiến trúc của mạng.
6. Cho phép có lỗi (fault tolerance).
7. Tích hợp khối lượng lớn (Very large scale integrated: VLSI).
8. Phóng sinh học. Kiến trúc và chức năng của mạng được thiết kế dựa trên kiến trúc và chức năng của mạng nơron sinh học.

Các bài toán thích hợp với mạng nơron truyền tải

Mạng nơron nhân tạo hiện nay là một cách tiếp cận có miền ứng dụng rất rộng bao gồm cả học có giám sát và không giám sát. Trong đó các mạng nơron truyền tải một hoặc nhiều tầng được dùng phổ biến nhất, thích hợp cho các bài toán có đặc điểm sau.

- Các mẫu được biểu diễn bởi nhiều cặp giá trị thuộc tính. Hàm đích được xác định trên các mẫu là vectơ đặc trưng với giá trị có tương quan hoặc độc lập.
- Hàm đích có thể nhận giá trị rời rạc hoặc thực của các vectơ có các thuộc tính giá trị thực hoặc rời rạc.
- Các mẫu đào tạo có thể có lỗi.
- Có thể chấp nhận thời gian huấn luyện lâu. So với cây quyết định thì huấn luyện mạng nơron thường lâu hơn nhiều.
- Có thể đòi hỏi đánh giá nhanh giá hàm đích học được. Đây là ưu điểm của mạng nơron so với các phương pháp dựa trên mẫu (k-NN)
- Không bắt buộc người dùng hiểu rõ hàm đích. Khi dùng mạng nổn, người dùng không cần biết và giải thích giá trị trọng số đang sử dụng.

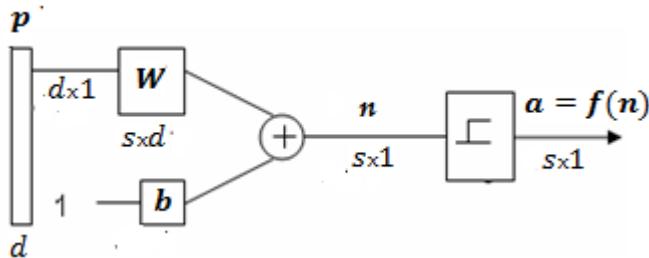
7.2. PERCEPTRON

7.2.1. Perceptron của Rosenblatt

McCulloch và Pitts (1943) là những người đầu tiên đưa ra mô hình của mạng nơron như là tổng có trọng số của các tín hiệu vào. Rosenblatt (1958) đề xuất mạng nơron để nhận dạng mẫu với luật học perceptron và chứng minh rằng luật học này hội tụ khi tập mẫu tách được tuyến tính. Ngày nay, perceptron dùng để chỉ các nơron đơn dùng để nhận dạng mẫu với hàm tổng hợp tuyến tính và hàm kích hoạt là hàm ngưỡng hoặc tuyến tính. Tuy nhiên, tùy theo ngữ cảnh mà ta có thể hiểu perceptron là mạng nơron do Rosenblatt đề xuất để phân biệt mẫu.

Kiến trúc của perceptron

Một perceptron thường được hiểu là một mạng nơron đơn, khi phân biệt nhiều lớp các perceptron có thể kết nối song song với nhau để phân biệt tuyến tính dữ liệu (tuyệt đối hoặc từng cặp), lúc đó ta gọi là mạng perceptron, nếu không xảy ra nhầm lẫn thì cũng có thể gọi gọn là perceptron. Một mạng perceptron gồm s nơron/perceptron với d tín hiệu và với hàm kích hoạt f là hàm ngưỡng $hardlim$ được mô tả trong hình 7.6, khi $s=1$ ta có một perceptron



Hình 7.6. Mô hình một mạng perceptron gồm s nơron

Các phương pháp huấn luyện

Các trọng số kết nối của perceptron có thể huấn luyện bằng thuật toán học perceptron hoặc thuật toán LMS của Widrow-Hoff trong mục 4.2, chương 4. Dưới đây các luật học này được tóm tắt như sau.

Luật học perceptron

Xét tập huấn luyện gồm hai lớp $D = \{(\mathbf{p}_i, t_i)\}_{i=1}^N, t_i \in \{0,1\}$. Khởi tạo vecto trọng số \mathbf{w} và giá trị ngưỡng w_0 tùy ý, thực hiện lặp thủ tục sau.

Lấy các mẫu \mathbf{p} trong D (tuần tự/ ngẫu nhiên), tính $a(\mathbf{p}) = hardlim(\mathbf{w}^T \mathbf{p} + w_0)$ và $e(p) = t(p) - a(p)$. Điều chỉnh trọng số theo công thức:

$$\mathbf{w}^{mi} = \mathbf{w}^{ci} + e(p)\mathbf{p}; \quad w_0^{mi} = w_0^{ci} + e(p) \quad (7.2a)$$

Thuật toán dừng khi không có mẫu dữ liệu nào cho sai số $e(p)$ khác không.

Đối với mạng perceptron, công thức tổng quát cho điều chỉnh trọng số trong mỗi bước lặp là:

$$W^{m\tilde{o}i} = W^{c\tilde{u}} + e p^T; \quad (7.2b)$$

$$b^{m\tilde{o}i} = b^{c\tilde{u}} + e. \quad (7.2c)$$

trong đó b là vectơ giá trị ngưỡng, $e = t(p) - a(p)$, t là vectơ giá trị đích của mẫu và a là đầu ra của mạng.

Như đã nói trong chương 4, luật học perceptron chỉ hội tụ khi tập mẫu quan sát được là tách được tuyến tính, luật học Widrow-Hoff không đòi hỏi điều kiện này.

Luật học Widrow-Hoff (thường dùng hàm kích hoạt là hardlims thay cho harlim)

Luật học này cũng khởi tạo giá trị các trọng số và ngưỡng ban đầu tùy ý và thực hiện lặp lấy ngẫu nhiên hoặc tuân tự các dữ liệu quan sát p để điều chỉnh trọng số theo công thức:

$$\mathbf{w}^{mi} = \mathbf{w}^{c\tilde{u}} + 2\alpha e(\mathbf{p})\mathbf{p}; \quad w_0^{mi} = w_0^{c\tilde{u}} + 2\alpha e(\mathbf{p}), \quad (7.3a)$$

trong đó α là tốc độ học thỏa mãn điều kiện $0 < \alpha < 1/\lambda_{\max}$ với λ_{\max} là giá trị riêng lớn nhất của ma trận tương quan mẫu

$$R = E(\mathbf{x}\mathbf{x}') = \frac{1}{N} \sum_{i=1}^N [\mathbf{x}^k(\mathbf{x}^k)']. \quad (7.3b)$$

Chú ý rằng, cũng có thể dùng hàm ngưỡng đối xứng *hardlims* làm hàm kích hoạt của mạng perceptron; tuy nhiên khi đó cần lưu ý khi dùng quy tắc học Rosenblatt.

Độc giả có thể viết công thức tổng quát cho mạng perceptron và ước lượng khả năng phân biệt của mạng như là bài tập.

7.2.2. Mạng ADALINE

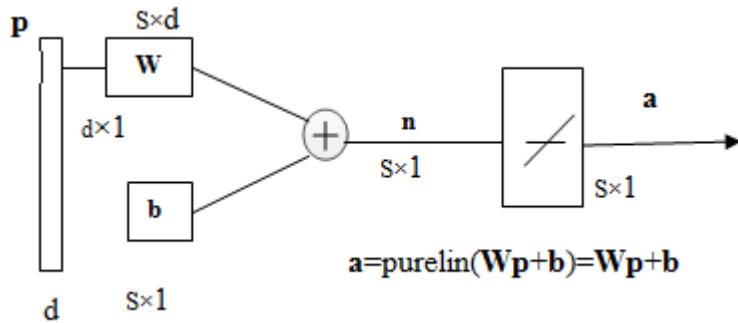
Mạng ADALINE có kiến trúc như perceptron nhưng dùng hàm kích hoạt tuyến tính như trong hình 7.7.

Phương pháp LMS thường được dùng để huấn luyện mạng và có thể dùng thuật toán Widrow-Hoff hoặc tìm trực tiếp cực trị sai số trung bình phương như sau.

Ta trở lại với tập mẫu quan sát $D = \{\mathbf{p}_i, t_i\}_{i=1}^N$ trong đó \mathbf{p}_i và t_i tương ứng là mẫu và tín hiệu ra thứ i tương ứng. Không giảm tổng quát, ta chỉ xét một noron, khi có nhiều noron thì có thể xem là nhiều noron được huấn luyện độc lập.

Ta dùng các ký hiệu: $\mathbf{x} = \begin{pmatrix} \mathbf{w}^T \\ b \end{pmatrix}$ và $\mathbf{z} = \begin{pmatrix} \mathbf{p} \\ 1 \end{pmatrix}$, khi đó đầu ra a của mạng sẽ là:

$$a = w^T p + b = x^T z. \quad (7.4a)$$



Hình 7.7. Một mạng ADALINE gồm S neuron và p tín hiệu vào

Ở đây xem khuynh hướng là thành phần có đầu vào bằng 1.

Biểu diễn sai số trung bình phương là một hàm F của x :

$$F(x) = E(e^2) = E(t-a)^2 = E(t - x^T z)^2 \quad (7.4b)$$

trong đó $E(\cdot)$ là ký kỳ vọng của biến ngẫu nhiên tương ứng. Ta xấp $F(x)$ bằng trung bình mẫu:

$$F(x) = \frac{1}{n} \sum_{i=1}^n (t_i - a_i)^2. \quad (7.4c)$$

Khai triển công thức (7.4b):

$$E(t-a)^2 = E(t^2 - 2tx^T z + x^T zz^T x) = E(t^2) - 2x^T E(tz) + x^T E(zz^T) x. \quad (7.4d)$$

Đặt $c = E(t^2)$; $h = E(tz)$ cho tương quan chập của tín hiệu vào và tín hiệu đích kết hợp và $R = E(zz^T)$ cho ma trận tương quan của tín hiệu vào, ta có:

$$F(x) = c - 2x^T h + x^T Rx. \quad (7.4f)$$

F là một dạng bậc 2 :

$$F(x) = c + d^T x + \frac{1}{2} x^T A x \quad (7.4g)$$

trong đó: $d = -2h$ và $A = 2R$.

Ma trận Hessian của F là $2R$ và luôn xác định dương hoặc không âm nên F luôn có cực tiểu toàn cục. Ta có gradient của F là:

$$\nabla F(x) = \nabla (c + d^T x + \frac{1}{2} x^T A x) = d + Ax = -2h + 2Rx \quad (7.4h)$$

Điểm dừng của F là điểm mà gradient triệt tiêu:

$$-2\mathbf{h} + 2R\mathbf{x} = 0. \quad (7.4i)$$

Khi R xác định dương thì F có duy nhất cực tiểu toàn cục :

$$\mathbf{x}^* = R^{-1}\mathbf{h} \quad (7.4h)$$

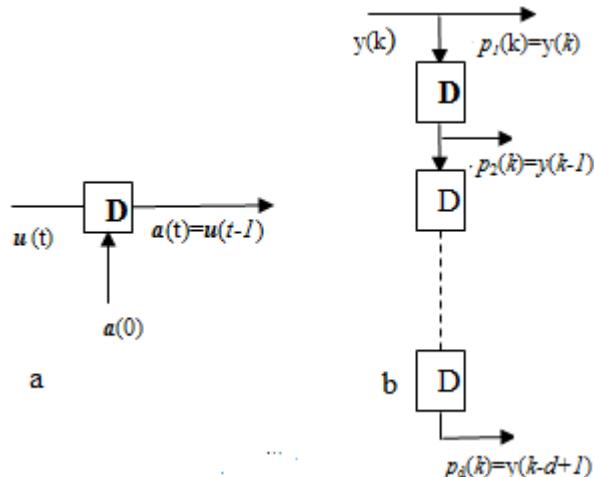
Khi không gian mẫu có số chiều lớn thì khó tìm gần đúng ma trận nghịch đảo của R , lúc đó phương pháp Widrow-hof hiệu quả hơn.

Bộ lọc thích nghi

Bộ lọc thích nghi (Adaptive filtering) là một ứng dụng của mạng ADALINE nhòe kết hợp với bộ trễ và đường trễ lặp.

Bộ trễ được minh họa trong hình 7.8a, trong đó t là biến thời gian, đầu ra $a(t)$ được tính từ $u(t)$ từ công thức $a(t) = u(t-1)$ và khởi tạo bằng $a(0)$.

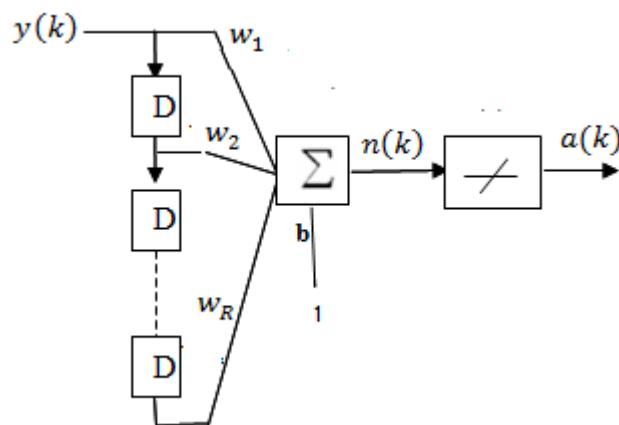
Đường trễ lặp được minh họa trong hình 7.8b, có tín hiệu vào $y(t)$, tại thời điểm k cho tín hiệu ra là vectơ d chiều $(y(k), \dots, y(k-d+1))^T$.



Hình 7.8. a) Bộ trễ b) Đường trễ lặp

Bộ lọc thích nghi cho phép hồi quy trễ một hàm số được minh họa trong hình 7.9, trong đó có $R-1$ bộ trễ và đầu ra tại thời điểm k là:

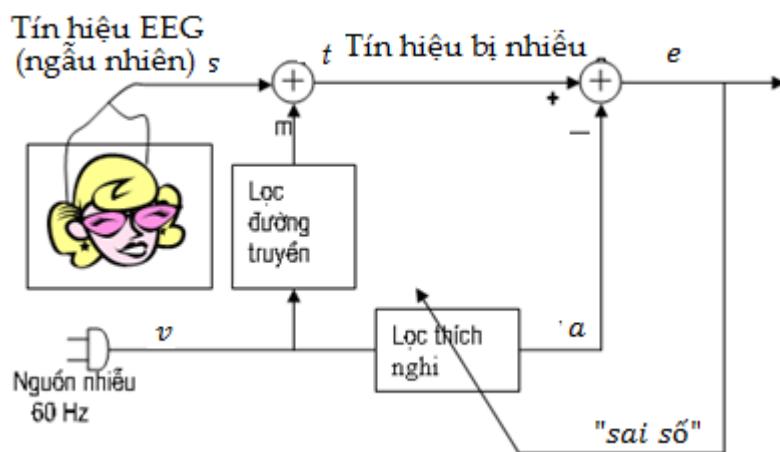
$$a(k) = \text{purelin}(\mathbf{W}\mathbf{p} + b) = \sum_{i=1}^R w_i y(k-i+1) + b. \quad (7.4l)$$



Hình 7.9. Bộ lọc thích nghi có R-1 bộ trễ

Ví dụ ứng dụng

Bộ lọc thích nghi có nhiều ứng dụng, ta xét một ví dụ sử dụng nó để khử nhiễu. Ta xét bộ đo điện não đồ (EEG) có tín hiệu s bị nhiễu bởi bởi m sinh từ nguồn nhiễu v có tần số 60 Hz nên có tín hiệu ra t . Tín hiệu v đã được lọc đường truyền nên cho m có biên độ giảm 10 lần và nâng pha thêm lượng $\pi/2$. Người ta dùng bộ lọc thích nghi được cài đặt để cực tiểu "sai số" e như trong hình 7.10, trong đó nó chỉ dựa trên nguồn nhiễu v để ước lượng được đầu ra a gần với m nhất.



Hình 7.10. Hệ khử nhiễu EEG sử dụng mạng ADALINE

Giả sử EEG s có phân bố đều trên đoạn $[-0,2; 0,2]$ và nguồn nhiễu $v(k)$ cho bởi

$$v(k) = 1,2 \sin \frac{2k\pi}{3}, \quad (7.5a)$$

$$\text{khi đó } m(k) = 0,12 \sin \left(\frac{2k\pi}{3} + \frac{\pi}{2} \right). \quad (7.5b)$$

Ta phân tích toán học cho hệ này. Xét ma trận tương quan mẫu:

$$R = E[zz^T]; h = E[tz], \quad (7.6a)$$

Trong trường hợp này tín hiệu vào của bộ lọc được cho bởi giá trị hiện tại và giá trị trước của nguồn nhiễu:

$$\mathbf{z} = \begin{bmatrix} v(k) \\ v(k-1) \end{bmatrix}, \quad (7.6b)$$

còn giá trị đích là tổng của tín hiệu s và tín hiệu nhiễu m tại thời điểm đang xét:

$$t(k) = s(k) + m(k) \quad (7.6c)$$

Bây giờ ta biểu diễn cụ thể R và h :

$$\begin{aligned} R &= E \begin{bmatrix} v(k) \\ v(k-1) \end{bmatrix} \begin{bmatrix} v(k) & v(k-1) \end{bmatrix} \\ &= \begin{bmatrix} E(v^2(k)) & E(v(k))v(k-1) \\ E(v(k-1))v(k) & E(v^2(k-1)) \end{bmatrix} \end{aligned} \quad (7.6d)$$

$$\text{và } h = E[tz] = \begin{bmatrix} E(s(k) + m(k))v(k) \\ E(s(k) + m(k))v(k-1) \end{bmatrix} \quad (7.6e)$$

Bây giờ ta tính các phân tử của R .

$$E(v^2(k)) = (1,2)^2 \frac{1}{3} \sum_{k=1}^3 (\sin \frac{2k\pi}{3})^2 = (1,2)^2 0,5 = 0,72 \quad (7.7a)$$

$$E(v^2(k-1)) = E(v^2(k)) = 0,72 \quad (7.7b)$$

$$\begin{aligned} E(v(k))v(k-1) &= \frac{1}{3} \sum_{k=1}^3 1,2(\sin \frac{2k\pi}{3})(1,2 \sin \frac{2(k-1)\pi}{3}) \\ &= (1,2)^2 0,5 \cos(\frac{2\pi}{3}) = -0,36 \end{aligned} \quad (7.7c)$$

$$\text{Vậy } R = \begin{bmatrix} 0,72 & -0,36 \\ -0,36 & 0,72 \end{bmatrix}. \quad (7.7d)$$

Tương tự ta tính h . Thành phần thứ nhất của (7.6e):

$$E[(s(k) + m(k))v(k)] = E[s(k)v(k)] + E[m(k)v(k)] \quad (7.7e)$$

Do $s(k), v(k)$ độc lập và s có phân bố đều nên số hạng thứ nhất bằng không.

$$E(m(k)v(k)) = \frac{1}{3} \sum_{k=1}^3 0,12 \sin(\frac{2k\pi}{3} + \frac{\pi}{2})(1,2 \sin \frac{2k\pi}{3}) = 0.$$

Như vậy thành phần thứ nhất của h bằng không.

Thành phần thứ hai của (7.6e):

$$E([s(k) + m(k)]v(k-1)) = E(s(k)v(k-1)) + E(m(k)v(k-1)). \quad (7.7f)$$

Lập luận như trước, ta có số hạng thứ nhất của (7.7f) triệt tiêu còn số hạng thứ hai:

$$E(m(k)v(k-1)) = \frac{1}{3} \sum_{k=1}^3 0,12 \sin\left(\frac{2k\pi}{3} + \frac{\pi}{2}\right) (1,2 \sin \frac{2(k-1)\pi}{3}) = -0,0624 = h_2.$$

Từ đó ta có $\mathbf{h} = \begin{bmatrix} 0 \\ -0,0624 \end{bmatrix}$. (7.7h)

Chú ý tới biểu thức (7.4h), ta có các trọng số cực tiêu sai số trung bình phương là :

$$\mathbf{x}^* = \mathbf{R}^{-1}\mathbf{h} = \begin{bmatrix} 0,72 & -0,36 \\ -0,36 & 0,72 \end{bmatrix}^{-1} \begin{bmatrix} 0 \\ -0,0624 \end{bmatrix} = \begin{bmatrix} -0,0578 \\ -0,1156 \end{bmatrix}$$

Để hiểu rõ hơn, ta trở lại với biểu thức (7.4f):

$$F(\mathbf{x}) = c - 2\mathbf{x}^T \mathbf{h} + \mathbf{x}^T \mathbf{R} \mathbf{x}, \quad (7.8a)$$

trong đó ta đã biết \mathbf{x}^* , \mathbf{R} và \mathbf{h} , nay cần tìm c .

$$c = E[t^2(k)] = E[s(k) + m(k)]^2 = E[s^2(k)] + 2E[s(k)m(k)] + E[m^2(k)].$$

Số hạng thứ hai triệt tiêu vì $s(k)$ và $m(k)$ độc lập và trung bình bằng không. Số hạng thứ nhất được tính như sau:

$$E[s^2(k)] = \frac{1}{0,4} \int_{-0,2}^{0,2} s^2 dS = \frac{1}{3 \cdot 0,4} S^3 \Big|_{-0,2}^{0,2} = 0,0133. \quad (7.8b)$$

Giá trị bình phương trung bình của nhiễu lọc là:

$$E[m^2(k)] = \frac{1}{3} \sum_{k=1}^3 \left\{ 0,12 \sin\left(\frac{2\pi}{3} + \frac{\pi}{2}\right) \right\}^2 = 0,0072. \quad (7.8c)$$

$$\text{Vậy } c = 0,0133 + 0,0072 = 0,0205 \quad (7.8d)$$

Thay các số hạng đã tính được vào biểu thức (7.8a) ta có:

$$F(\mathbf{x}^*) = 0,025 - 2 \cdot 0,0072 + 0,0072 = 0,0133. \quad (7.8e)$$

Như vậy lỗi bình phương trung bình giống như giá trị bình phương trung bình của tín hiệu EEG.

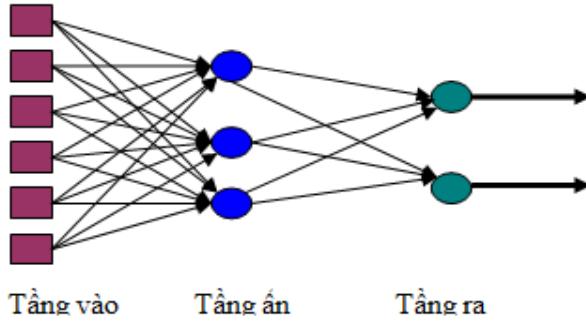
7.3. PERCEPTRON NHIỀU TẦNG (MẠNG MLP)

Perceptrons nhiều tầng thường gọi là mạng MLP (viết tắt của MultiLayer Perceptrons) là loại mạng thông dụng nhất để hồi quy hàm nhiều biến.

7.3.1. Kiến trúc mạng

Mạng MLP là mạng truyền tải nhiều tầng bao gồm các nút nhận tín hiệu vào bằng số, có hai hoặc nhiều tầng nơron, hàm tổng hợp trong các nơron đều có dạng:

$s = n = \sum_{i=1}^k w_i x_i + \theta$, các hàm chuyển có thể có dạng khác nhau. Một mạng có d tín hiệu vào và M tầng noron thứ i có s^i noron ta nói mạng có kiến trúc $d - S^1 - S^2 \dots S^M$. Trong trường hợp này ta gọi là mạng $(M+1)$ tầng hoặc nói rõ mạng M tầng noron cũng được. Mặc dù có thể có nhiều tầng noron nhưng trong mạng MLP, ta sẽ gọi tầng tín hiệu vào là *tầng vào*, tầng noron cuối cùng là *tầng ra*, còn các tầng noron ở giữa đều gộp chung gọi là *tầng ẩn*. Hình 7.11 mô tả mạng noron có kiến trúc 6-3-2 tức là có 6 nút vào, 2 tầng ẩn với 3 noron mỗi tầng và 2 noron tầng ra.



Hình 7.11. Kiến trúc mạng noron truyền tới nhiều tầng

Mạng MLP được dùng rộng rãi để *xấp xỉ/ hồi quy* hàm nhiều biến và phân lớp mẫu, khi dùng để xấp xỉ hàm thì hàm kích hoạt thường là hàm log_sig hoặc tanh hyperbolic. Một số điểm sau cần lưu ý khi thiết kế mạng này:

- *Tầng vào*. Nếu hàm đang xét có n biến thì tầng này có $n+1$ nút trong đó nút đầu ứng với giá trị $x_0 = -1$ và trọng số là ngưỡng θ , mỗi nút còn lại ứng với một biến của đối.
- *Tầng ẩn*. Mạng MLP ba tầng có thể xấp xỉ một hàm liên tục với sai số bé tùy ý nếu có đủ dữ liệu huấn luyện và số noron tầng ẩn phù hợp. Tuy nhiên, Việc chọn cấu trúc tầng ẩn thích hợp nhất đến nay vẫn là bài toán mở. Lưu ý rằng nếu số trọng số kết nối quá ít so với dữ liệu quan sát thì sai số lớn, còn nếu quá nhiều thì dẫn tới phù hợp trội.
- *Tầng ra*. Mỗi một noron ở tầng ra sẽ ứng với một hàm, tức là nếu hàm cần xấp xỉ có giá trị ra là vector M chiều thì có M noron ở tầng ra.

7.3.2. Thuật toán huấn luyện lan truyền ngược (BP)

Sau khi đã chọn kiến trúc mạng, người ta thường dùng phương pháp truyền ngược (Back-Propagation, viết tắt là BP) sai số để xác định các trọng số kết nối cho mạng MLP nhờ thuật toán Gradient cực tiểu hóa sai số trung bình phương.

Xét một MLP có kiến trúc $d - S^1 - S^2 \dots S^M$ và tập mẫu quan sát được $D = \{\mathbf{t}_i, \mathbf{t}_i\}_{i=1}^N$. Để tiện trình bày, trong mục này ta ký hiệu:

$$\mathbf{a}^0 = \mathbf{p}, \mathbf{a} = \mathbf{a}^M, \quad (7.9a)$$

$$\mathbf{a}^{m+1} = \mathbf{f}^{m+1}(\mathbf{W}^{m+1}\mathbf{a}^m + \mathbf{b}^{m+1}), \quad (7.9b)$$

\mathbf{W}^m là ma trận cấp $S^m \times S^{m-1}$ là ma trận trọng số kết nối của các nơron ở tầng thứ m và $w_{i,j}^m$ là trọng số kết nối thứ j của nơron thứ i trong này.

Ta dùng thuật toán tương tự của Widrow-Hoff, dùng thuật toán gradient cực tiểu tổng bình phương sai số hay kỳ vọng mẫu $F(\mathbf{x})$ của biến ngẫu nhiên \mathbf{x} :

$$F(\mathbf{x}) = E(\mathbf{e}^T \mathbf{e}) = E((\mathbf{t}-\mathbf{a})^T (\mathbf{t}-\mathbf{a})), \quad (7.10a)$$

$$\text{nếu } S^M=1 \text{ thì } F(\mathbf{x}) = E[(\mathbf{t}-\mathbf{a})^2]. \quad (7.10b)$$

Trong lần lặp thứ k , kỳ vọng này được xấp xỉ bởi $\hat{F}(\mathbf{x})$:

$$\hat{F}(\mathbf{x}) = (\mathbf{t}(k)-\mathbf{a}(k))^T (\mathbf{t}(k)-\mathbf{a}(k)) = \sum_{i=1}^{S^M} (t_i(k) - a_i(k))^2 = \mathbf{e}^T(k) \mathbf{e}(k), \quad (7.10c)$$

trong đó $\mathbf{t}(k)$, $\mathbf{a}(k)$ và $\mathbf{e}(k)$ được tính nhờ vecto tín hiệu vào $\mathbf{p}(k)$ lấy ngẫu nhiêu hoặc tuân tự từ tập mẫu D . Các trọng số kết nối ở mỗi tầng nơron m được điều chỉnh theo công thức :

$$w_{i,j}^m(k+1) = w_{i,j}^m(k) - \alpha \frac{\partial \hat{F}}{\partial w_{i,j}^m}, \quad (7.10a)$$

$$\text{và } b_i^m(k+1) = b_i^m(k) - \alpha \frac{\partial \hat{F}}{\partial b_i}, \quad (7.11b)$$

trong đó α là tốc độ học. Các đạo hàm riêng được tính nhờ công thức đạo hàm hàm hợp của hàm f tùy ý :

$$\frac{df(n(w))}{dw} = \frac{df(n)}{dn} \frac{dn}{dw}, \quad (7.12a)$$

$$\text{ta có: } \frac{\partial \hat{F}}{\partial w_{i,j}^m} = \frac{\partial \hat{F}}{\partial n_i^m} \frac{\partial n_i^m}{\partial w_{i,j}^m} \quad (7.12b)$$

$$\frac{\partial \hat{F}}{\partial b_i^m} = \frac{\partial \hat{F}}{\partial n_i^m} \frac{\partial n_i^m}{\partial b_i^m}. \quad (7.12c)$$

Bởi vì $n_i^m = \sum_{j=1}^{S^{m-1}} w_{i,j}^m a_j^{m-1} + b_i^m$ nên:

$$\frac{\partial n_i^m}{\partial w_{i,j}} = a_j^{m-1} \quad \text{và} \quad \frac{\partial n_i^m}{\partial b_i^m} = 1. \quad (7.12d)$$

Nếu ta đặt $S_i^m = \frac{\partial \hat{F}}{\partial n_i^m}$ (7.12e)

(độ nhạy cảm của \hat{F} đối với sự thay đổi của tín hiệu tổng hợp của noron i ở tầng m) thì các biểu thức trong (7.12d) có biểu diễn đơn giản là:

$$\frac{\partial F}{\partial w_{i,j}^m} = S_i^m a_j^{m-1}; \quad \frac{\partial \hat{F}}{\partial b_i^m} = S_i^m. \quad (7.12f)$$

Như vậy quy tắc hiệu chỉnh trọng số ở theo các công thức (7.10a) và (7.10b) trở thành :

$$\begin{cases} w_{i,j}^m(k+1) = w_{i,j}^m(k) - \alpha S_i^m a_j^{m-1} \\ b_j^m(k+1) = b_j^m(k) - \alpha S_i^m. \end{cases} \quad (7.12g)$$

Biểu diễn dưới dạng ma trận công thức này là:

$$\begin{cases} \mathbf{W}^m(k+1) = \mathbf{W}^m(k) - \alpha \mathbf{s}^m (\mathbf{a}^{m-1})^T \\ \mathbf{b}^m(k+1) = \mathbf{b}^m(k) - \alpha \mathbf{s}^m, \end{cases} \quad (7.12h)$$

trong đó $\mathbf{s}^m = \frac{\partial \hat{F}}{\partial n^m} = \begin{bmatrix} \frac{\partial \hat{F}}{\partial n_1^m} \\ \frac{\partial \hat{F}}{\partial n_2^m} \\ \vdots \\ \frac{\partial \hat{F}}{\partial n_{s^m}^m} \end{bmatrix}$ (7.12i)

Hiệu chỉnh lan truyền ngược

Để áp dụng công thức hiệu chỉnh (7.12h), công việc còn lại là ước lượng các s^m . Quá trình tính sẽ bắt đầu từ tầng ra, lùi dần về tầng noron thứ nhất, trong đó s^m được tính từ s^m nên có tên gọi *lan truyền ngược*.

Để đưa ra công thức đệ quy, ta dùng ma trận Jacobi cấp $S^{m+1} \times S^m$:

$$\frac{\partial n^{m+1}}{\partial n^m} = \left[\frac{\partial n_i^{m+1}}{\partial n_j^m} \right]_{S^{m+1} \times S^m} = \begin{bmatrix} \frac{\partial n_1^{m+1}}{\partial n_1^m} & \frac{\partial n_1^{m+1}}{\partial n_2^m} & \cdots & \frac{\partial n_1^{m+1}}{\partial n_{S^m}^m} \\ \frac{\partial n_2^{m+1}}{\partial n_1^m} & \frac{\partial n_2^{m+1}}{\partial n_2^m} & \cdots & \frac{\partial n_2^{m+1}}{\partial n_{S^m}^m} \\ \cdots & \cdots & \cdots & \cdots \\ \frac{\partial n_{S^m}^{m+1}}{\partial n_1^m} & \frac{\partial n_{S^m}^{m+1}}{\partial n_2^m} & \cdots & \frac{\partial n_{S^m}^{m+1}}{\partial n_{S^m}^m} \end{bmatrix}. \quad (7.13a)$$

Để tìm biểu diễn cho ma trận này, ta xét phần tử (i,j) của ma trận :

$$\frac{\partial n_i^{m+1}}{\partial n_j^m} = \frac{\partial (\sum_{j=1}^{S^m} w_{i,j}^{m+1} a_j^m + b_i^{m+1})}{\partial n_j^m} = w_{i,j}^{m+1} \frac{\partial a_j^m}{\partial n_j^m} \quad (7.13b)$$

$$= w_{i,j}^{m+1} \frac{\partial f^m(n_i^m)}{\partial n_i^m} = w_{i,j}^{m+1} \dot{f}^m(n_j^m).$$

$$\text{trong đó } \dot{f}^m(n_j^m) = \frac{\partial f^m(n_i^m)}{\partial n_i^m}. \quad (7.13c)$$

Như vậy ma trận Jacobi có thể viết là:

$$\frac{\partial \mathbf{n}^{m+1}}{\partial \mathbf{n}^m} = \mathbf{W}^{m+1} \dot{\mathbf{F}}^m(\mathbf{n}^m) \quad (7.13d)$$

trong đó:

$$\dot{\mathbf{F}}^m = \begin{pmatrix} \dot{f}^m(n_1^m) & 0 & \dots & 0 \\ 0 & \dot{f}^m(n_2^m) & \dots & \dots \\ \dots & \dots & \dots & \dots \\ 0 & 0 & 0 & \dot{f}^m(n_{S^m}^m) \end{pmatrix}. \quad (7.13e)$$

Bây giờ dạng đệ quy cho các s^m được viết theo quan hệ đệ quy nhò dùng quy tắc dây chuyền dạng ma trận:

$$\mathbf{s}^m = \frac{\partial \hat{F}}{\partial \mathbf{n}^m} = \left(\frac{\partial \mathbf{n}^{m+1}}{\partial \mathbf{n}^m} \right)^T \frac{\partial \hat{F}}{\partial \mathbf{n}^{m+1}} = \dot{\mathbf{F}}^m(\mathbf{n}^m) (\mathbf{W}^{m+1})^T \frac{\partial \hat{F}}{\partial \mathbf{n}^{m+1}} \quad (7.13f)$$

$$= \dot{\mathbf{F}}^m(\mathbf{n}^m) \mathbf{W}^{m+1} \mathbf{s}^{m+1}$$

Bây giờ ta thấy rõ sự lan truyền ngược khi tính s^m từ tầng ra tới tầng đầu:

$$\mathbf{s}^M \rightarrow \mathbf{s}^{M-1} \rightarrow \dots \mathbf{s}^2 \rightarrow \mathbf{s}^1 \quad (7.13g)$$

Để hoàn thiện thuật toán truyền ngược, ta cần tính s^M cho công thức đê quy (7.13f):

$$S_i^M = \frac{\partial \hat{F}}{\partial n_i^M} = \frac{\partial (\mathbf{t} - \mathbf{a})^T (\mathbf{t} - \mathbf{a})}{\partial n_i^M} = \frac{\partial \sum_{j=1}^{S^M} (t_j - a_j)^2}{\partial n_j^M} = -2(t_i - a_i) \frac{\partial a_i}{\partial n_i^M}. \quad (7.14a)$$

$$\text{Bởi vì } \frac{\partial a_i}{\partial n_i^M} = \frac{\partial a_i^M}{\partial n_i^M} = \frac{\partial f^M(n_i^M)}{\partial n_i^M} = \dot{f}(n_i^M) \quad (7.14b)$$

$$\text{nên: } S_i^M = -2(t_i - a_i) \dot{f}(n_i^M) \quad (7.14c)$$

Biểu diễn dưới dạng ma trận:

$$\mathbf{s}^M = -2 \dot{\mathbf{F}}(\mathbf{n}^M)(\mathbf{t} - \mathbf{a}) \quad (7.14e)$$

Điều kiện dừng của thuật toán thường là khi các trọng số và khuynh hướng/giá trị ngưỡng thay đổi không đáng kể, tức là nhỏ hơn ϵ đủ bé cho trước. Các trọng số ban đầu và khuynh hướng được khởi tạo tùy ý. Thủ tục của thuật toán BP được mô tả trong bảng 7.1.

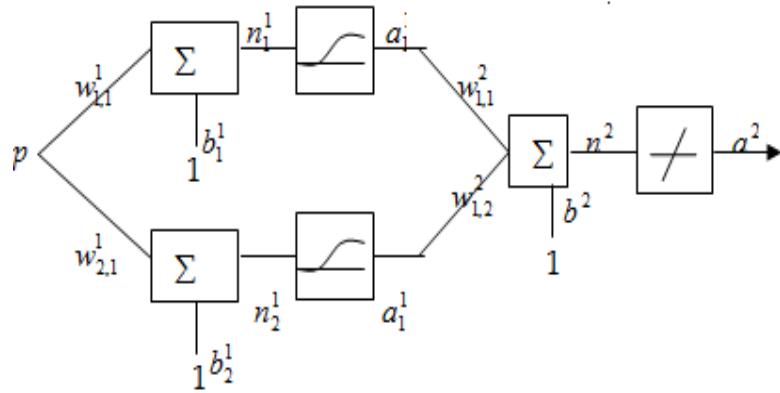
Nhược điểm quan trọng của thuật toán BP là thường chỉ cho lời giải gần đúng của cực trị địa phương và mất nhiều thời gian huấn luyện. Để khắc phục nhược điểm thứ nhất, ta có thể khởi tạo ngẫu nhiên nhiều bộ giá trị ban đầu cho các trọng số và giá trị khuynh hướng, sau khi huấn luyện thì chọn lời giải cho sai số trung bình phương nhỏ nhất.

Bảng 7.1. Thuật toán BP huấn luyện mạng MLP

<p>Thực hiện lặp đến khi thỏa mãn điều kiện dừng:</p> <p>Bước 0. $\mathbf{p} \leftarrow \mathbf{p}_k //$ Lấy ngẫu nhiên hoặc tuần tự trong D</p> <p>Bước 1. Đưa đầu vào \mathbf{p} qua mạng, rồi tính:</p> <ol style="list-style-type: none"> 1.1. $\mathbf{a}^0 = \mathbf{p},$ 1.2. $\mathbf{a}^{m+1} = \dot{\mathbf{f}}^{m+1}(\mathbf{W}^{m+1}\mathbf{a}^m + \mathbf{b}^{m+1}) \forall m = 0, 1, \dots, M-1,$ 1.3. $\mathbf{a} = \mathbf{a}^M$ <p>Bước 2. // truyền ngược độ nhạy cảm</p> <ol style="list-style-type: none"> 2.1. $\mathbf{s}^M = -2 \dot{\mathbf{F}}(\mathbf{n}^M)(\mathbf{t} - \mathbf{a})$ 2.2. $\mathbf{s}^m = \dot{\mathbf{F}}^m(\mathbf{n}^m)(\mathbf{W}^{m+1})^T \mathbf{s}^{m+1} \forall m = M-1, \dots, 2, 1$ <p>Bước 3. // Cập nhật theo công thức đê quy</p> <ol style="list-style-type: none"> 3.1. $\mathbf{W}^{m+1}(k+1) = \mathbf{W}^{m+1}(k) - \alpha \mathbf{s}^m (\mathbf{a}^{m+1})^T$ 3.2. $\mathbf{b}^{m+1}(k+1) = \mathbf{b}^{m+1}(k) - \alpha s^m$

Ví dụ

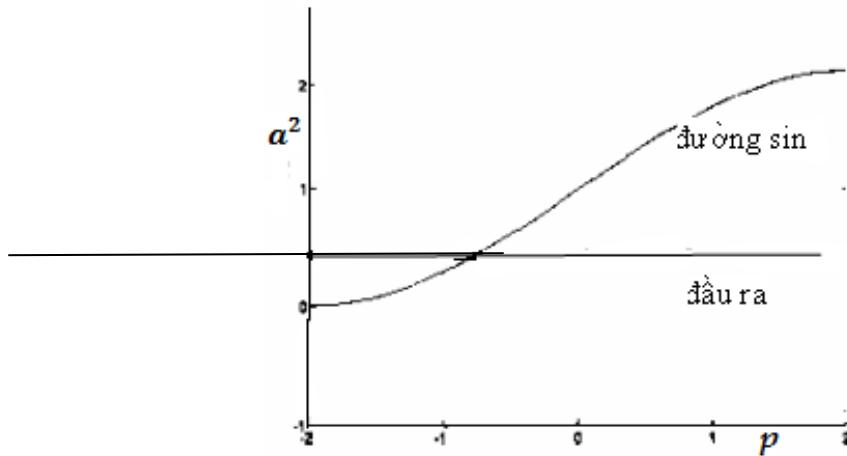
Xét hàm một biến $g(p) = 1 + \sin(\frac{\pi}{4} p)$ với $-2 \leq p \leq 2$. ta xây dựng mạng MLP có kiến trúc 1-2-1 với $f^1 = \log sig(n) = \frac{1}{1+e^{-n}}$ và $f^2 = purelin(s) = s$ như trong hình 7.12.



Hình 7.12. Ví dụ mạng xấp xỉ hàm $g(p) = 1 + \sin(\frac{\pi}{4}p)$, $-2 \leq p \leq 2$

$$\text{Giả sử } W^1(0) = \begin{bmatrix} -0.27 \\ -0.41 \end{bmatrix}; b^1(0) = \begin{bmatrix} -0.48 \\ -0.13 \end{bmatrix}; W^2(0) = (0.09 \quad -0.17), b^2(0) = [0.48]$$

Xấp xỉ ban đầu được mô tả trong hình 7.13



Hình 7.13. Đầu ra của mạng với trọng số khởi tạo

Quá trình thực hiện thuật toán như sau.

Bước 1: $p=1; a^0 = p = 1$

Đầu ra tầng thứ nhất là:

$$a^1 = f^1(W^1 a^0 + b^1) = \log sig\left(\begin{bmatrix} -0.27 \\ -0.41 \end{bmatrix}[1] + \begin{bmatrix} -0.48 \\ -0.13 \end{bmatrix}\right) = \log sig\left(\begin{bmatrix} -0.75 \\ -0.54 \end{bmatrix}\right)$$

$$= \begin{bmatrix} \frac{1}{1+e^{-0.75}} \\ \frac{1}{1+e^{-0.54}} \end{bmatrix} = \begin{bmatrix} 0.321 \\ 0.368 \end{bmatrix}$$

Đầu ra tầng hai là :

$$a^2 = f^2(W^2 a^1 + b^2) = \text{purelin}([0.09 \quad -0.17] \begin{bmatrix} 0.321 \\ 0.368 \end{bmatrix} + [0.48]) = [0.446]$$

Tính sai số:

$$e = t-a = \{1 + \sin(\frac{\pi}{4} p)\} - a^2 = \{1 + \sin(\frac{\pi}{4} 1)\} - 0.446 = 1.261$$

$$\dot{f^1}(n) = \frac{d}{dn} \left(\frac{1}{1+e^{-n}} \right) = \frac{e^n}{(1+e^{-n})^2} = (1 - \frac{1}{1+e^{-n}}) \left(\frac{1}{1+e^{-n}} \right) = (1-a^1)(a^1).$$

Đối với tầng thứ hai ta có :

$$\dot{f^2}(n) = \frac{d}{dn}(n) = 1.$$

$$\text{Vậy: } s^2 = -2 \dot{F^2}(n^2)(t-a) = -2[\dot{f^2}(n^2)](1,261) = -2[1](1,261) = -2,522.$$

Tính lan truyền ngược cho tầng thứ nhất:

$$\begin{aligned} s^1 &= F^1(n^1)(W^2)^T s^2 = \begin{bmatrix} (1-a_1^1)(a_1^1) & 0 \\ 0 & (1-a_2^1)(a_2^1) \end{bmatrix} \begin{bmatrix} 0.09 \\ -0.17 \end{bmatrix} [-2,522] \\ &= \begin{bmatrix} (1-0.321)(0.321) & 0 \\ 0 & (1-0.368)(0.368) \end{bmatrix} \begin{bmatrix} 0.09 \\ -0.17 \end{bmatrix} [-2,522] \\ &= \begin{bmatrix} 0.218 & 0 \\ 0 & 0.233 \end{bmatrix} \begin{bmatrix} -0.227 \\ 0.429 \end{bmatrix} = \begin{bmatrix} -0.0495 \\ 0.0997 \end{bmatrix} \end{aligned}$$

Chọn $\alpha=0,1$ ta có:

$$W^2(1) = W^2(0) - \alpha S^2(a^1)^T = [0.09 \quad -0.17] - 0.1[-2.522] \begin{bmatrix} 0.321 & 0.368 \end{bmatrix} = [0.171 \quad -0.0772]$$

$$b^2(1) = b^2(0) - \alpha S^2 = [0.48] - 0.1[-2.522] = [0.732]$$

$$W^1(1) = W^1(0) - \alpha S^1(a^0)^T = \begin{bmatrix} -0.27 \\ -0.41 \end{bmatrix} - 0.1 \begin{bmatrix} -0.0495 \\ 0.0997 \end{bmatrix} [1] = \begin{bmatrix} -0.265 \\ -0.420 \end{bmatrix}$$

$$b^1(1) = b^1(0) - \alpha S^1 = \begin{bmatrix} -0.48 \\ -0.13 \end{bmatrix} - 0.1 \begin{bmatrix} -0.0495 \\ 0.0997 \end{bmatrix} = \begin{bmatrix} -0.475 \\ -0.140 \end{bmatrix}$$

Như vậy bước lặp thứ nhất của thuật toán được hoàn thành.

7.4. MẠNG HÀM CƠ SỞ BÁN KÍNH (MẠNG RBF)

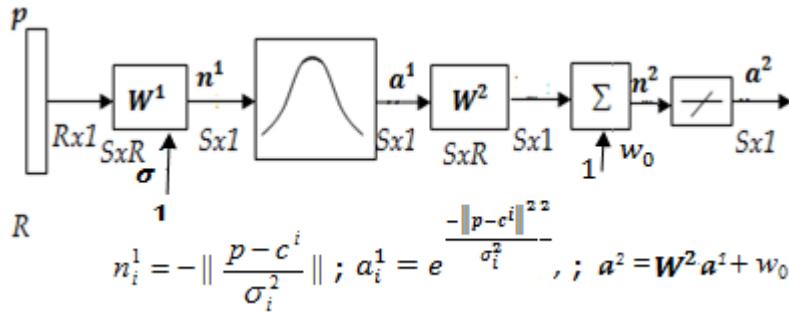
Như đã nói ở trên, mạng MLP là công cụ hữu hiệu để hồi quy hàm nhiều biến. Tuy nhiên, nhược điểm của nó là khó tìm được cấu trúc mạng phù hợp, thời gian huấn luyện lâu và thường chỉ cho cực trị địa phương của sai số trung bình phương. Mạng RBF (Radial Basis Function) là tiếp cận hiệu quả cho hồi quy hàm số với mục đích nội suy. Mặc dù không dùng được cho mục đích ngoại suy nhưng mạng RBF có ưu điểm là thời gian huấn luyện nhanh và các thuật toán luôn cho lời giải hội tụ tới cực tiểu toàn cục duy nhất.

7.4.1. Kiến trúc mạng RBF

Kỹ thuật hồi quy RBF do Powell đề xuất (1987), được Broomhead và Lowe (1988) áp dụng cho mạng nơron để hồi quy hàm nhiều biến. Mạng RBF xấp xỉ hàm thực n biến $f : D \subset \mathbb{R}^n \rightarrow \mathbb{R}^m$ là một mạng 3 tầng truyền tải như mô tả trong hình 7.13. Tầng vào gồm n nút cho vectơ tín hiệu vào $x \in \mathbb{R}^n$, tầng ẩn là tầng hàm cơ sở bán kính gồm M nơron, dạng thông dụng của đầu ra tầng 1 là :

$$n_i^1 = -\left\| \frac{p - c^i}{\sigma_i^2} \right\|^2 \text{ và } a_i^1 = e^{-\frac{\|p - c^i\|^2}{\sigma_i^2}},$$

và đầu ra tầng hai là: $\mathbf{a}^2 = \mathbf{W}^2 \mathbf{a}^1$, hay: $a_j^2 = \sum_{k=1}^M w_{j,k}^2 a_k^1 + w_0$.



Hình 7.14. Mạng RBF tổng quát

Các hàm $\varphi_k(x) = e^{-\frac{(x - c^k)^2}{\sigma_k^2}}$ là các hàm cơ sở bán kính dạng Gauss, c^k gọi là tâm của φ_k , tham số σ_k (còn gọi là tham số độ rộng của hàm cơ sở) dùng để điều khiển độ rộng miền ảnh hưởng của hàm cơ sở φ_k (khi $\|x - c^k\| > 3\sigma_k$ thì $\varphi_k(x)$ gần triệt tiêu) còn $w_{i,k}^2$ gọi là trọng số của φ_k nơron thứ i ở tầng ra . Đầu ra thứ j của nơron thứ k ở tầng ra được viết lại là:

$$a_j^2(x) = \varphi^j(x) = \sum_{k=1}^M w_{j,k}^2 \varphi_k(x) + w_0 \quad (7.15a)$$

7.4.2. Huấn luyện mạng RBF

Không giảm tổng quát, ta xét tầng ra chỉ có một nơron, tập dữ liệu quan sát được $D = \{x^k, y^k\}_{k=1}^N$. Ký hiệu $\mathbf{W}^2 = \mathbf{w}^T = (w_1, \dots, w_k)$, các w_i được gọi là trọng số tầng ra. Khi đó đầu ra của mạng là:

$$a^2(\mathbf{x}) = \varphi(\mathbf{x}) = \sum_{k=1}^M w_k \varphi_k(\mathbf{x}) + w_0 \quad (7.15b)$$

Việc huấn luyện mạng bao gồm xác định tâm các tâm c^k , tham số độ rộng σ_k của mỗi hàm bán kính φ_k , các trọng số $\{w_i\}_{i=0}^M$. Hai hướng tiếp cận sau thường được dùng để xác định chúng.

1) Tìm cực tiểu sai số tổng các bình phương (sum-of-squares error: SSE):

$$E = \sum_{k=1}^N (a^2(x^k) - y^k)^2 \quad (7.16a).$$

2) Giải đúng hoặc gần đúng hệ phương trình:

$$\varphi(\mathbf{x}^i) = \sum_{k=1}^M w_k \varphi_k(\mathbf{x}^i) + w_0 = y^i \quad \forall i = 1, 2, \dots, N \quad (7.16b)$$

7.4.2.1. Các thuật toán dựa trên tìm cực tiểu SSE

Các thuật toán này được phân ba loại: một pha, hai pha và 3 pha. Trong đó việc xác định trọng số tầng ra được thực hiện nhờ thuật toán gradient hoặc một biến thể của nó.

a) Các thuật toán một pha

Trong các thuật toán một pha, người ta xác định trước tâm và các tham số độ rộng còn trọng số tầng ra được xác định nhờ thuật toán gradient. Diễn hình là các *thuật toán đào tạo nhanh* (Quick Training viết tắt là QT), trong đó người ta chọn trước các tâm cho hàm bán kính, còn độ rộng có thể lấy theo các đề xuất của Looney hoặc Haykin:

- Theo Looney: $\sigma_k = \frac{1}{\sqrt{2}(2N)^{k/n}}$ (7.17a)

- Theo Haykin $\sigma = \frac{D_{max}}{\sqrt{2N}}$, (7.17b)

trong đó $D_{max} = \max \{d(\mathbf{x}^i, \mathbf{x}^k) : i, k \in \{1, \dots, N\}\}$

Trong trường hợp số mốc không nhiều, các điểm $\{x^k\}_{k=1}^N$ thường được chọn làm các tâm cho các hàm bán kính ($M=N$) để áp dụng các thuật toán QT.

Lưu ý rằng nếu độ rộng của các hàm bán kính được chọn bé thì các điểm xa tâm ít chịu ảnh hưởng của hàm cơ sở nên chất lượng xấp xỉ của hàm hồi quy thấp mặc dù thuật toán hội tụ nhanh.

b) Các thuật toán hai pha

Trong các thuật toán này, các tâm và độ rộng các hàm bán kính được xác định ở pha thứ nhất, còn pha thứ hai xác định các trọng số tầng ra.

Chẳng hạn, dùng thuật toán phân cụm ngưỡng để phân cụm tập $\{x^k\}_{k=1}^N$ và lấy điểm trung bình của mỗi cụm làm tâm, các độ rộng xác định để hàm bán kính tương ứng xấp xỉ đủ tốt dữ liệu quan sát trên cụm này.

c) Các thuật toán ba pha

Thuật toán này còn gọi là thuật toán *đào tạo đầy đủ*. Trong đó người ta dùng một thuật toán hai pha để tìm tâm và các tham số đủ tốt, sau đó để điều chỉnh các độ rộng bán kính và trọng số tầng ra nhò dùng một biến thể của phương pháp gradient.

Dễ thấy rằng phương pháp này đòi hỏi nhiều thời gian huấn luyện nhưng chất lượng mạng tốt hơn các phương pháp trên. Độc giả có thể tham khảo mô tả chi tiết của một thuật toán loại này trong [15]

Chú ý. Trong các thuật toán kể trên, thay vì tìm cực tiểu SSE bằng phương pháp gradient, người ta có thể tìm trọng số tầng ra bằng phương pháp giả nghịch đảo để giải hệ (7.16b). Tuy nhiên, khi số mốc lớn thì lời giải tìm được cho sai số lớn nên không đáng tin cậy.

Dưới đây giới thiệu phương pháp hai pha huấn luyện lắp mạng RBF nội suy để sử dụng, thời gian huấn luyện nhanh và cho chất lượng mạng khá tốt khi khoảng cách giữa các mốc nội suy phân bố gần với lưỡi đều. Khi các mốc nội suy cách đều, ta có thể xác định trước các độ rộng bán kính và thuật toán hai pha trở thành thuật toán một pha. Thuật toán này kết hợp với hồi quy k-k-NN trong chương 8 cho ta một phương pháp hiệu quả để xây dựng mạng RBF hồi quy.

7.4.2.2. Phương pháp huấn luyện lắp mạng nội suy

Ta trở lại bài toán nội suy với các mốc nội suy $\{x^k\}_{k=1}^N$ cho bởi tập dữ liệu quan sát $D = \{x^k, y^k\}_{k=1}^N$ nêu trên. Các mốc nội suy được dùng làm tâm cho các hàm bán kính.

Phân tích toán học

Thuật toán được xây dựng dựa trên nhận xét hệ phương trình tuyến tính (7.16b) với ẩn là các trọng số tầng ra thì các hệ số của chúng có đặc điểm:

$$\varphi_k(x^i) = 1 \text{ vì } i = k, \quad (7.18a)$$

$$\lim_{\sigma_k \rightarrow 0} \varphi_k(x^i) = 0 \quad \forall i \neq k, \quad (7.18b)$$

hơn nữa, $\varphi_k(x^i)$ là hàm đơn điệu giảm của σ_k với mọi $i \neq k$.

Đặc điểm này cho phép ta chọn được các σ_k thích hợp để hệ phương trình tuyến tính (7.16b) có dạng đường chéo trội và trọng số tầng ra được tìm nhò phép lắp đơn. Cụ thể như sau.

Ký hiệu I là ma trận đơn vị cấp N ; $\mathbf{w} = \begin{pmatrix} \mathbf{w}_1 \\ \dots \\ \mathbf{w}_N \end{pmatrix}$, $\mathbf{z} = \begin{pmatrix} \mathbf{z}_1 \\ \dots \\ \mathbf{z}_N \end{pmatrix}$ là các vec tơ trong không gian N -chiều R^N trong đó:

$$z_k = y_k w_0, \quad \forall k \leq N \quad (7.19a)$$

$$\text{và đặt } \Phi = (\varphi_k(\mathbf{x}^i))_{N \times N}, \Psi = \mathbf{I} - \Phi = (\psi_{k,j})_{N \times N}. \quad (7.19b)$$

$$\text{Ta có } \psi_{k,j} = \begin{cases} 0; & \text{khi } k = j \\ -e^{-||\mathbf{x}^i - \mathbf{x}^k||^2 / \sigma_k^2}; & \text{khi } k \neq j \end{cases} \quad (7.19c)$$

hệ phương trình nội suy (7.16b) tương đương với hệ :

$$\mathbf{w} = \Psi \mathbf{w} + \mathbf{z} \quad (7.20a)$$

Nếu $\|\Psi\| < 1$ thì với các tham số σ_k đã chọn và w_0 tùy ý, (7.9) luôn có duy nhất nghiệm \mathbf{W} . Về sau ta sẽ lấy w_0 là trung bình cộng của các giá trị y^k :

$$w_0 = \frac{1}{N} \sum_{k=1}^N y^k \quad (7.20b)$$

Bây giờ với mỗi $k \leq N$, ta có hàm q_k của σ_k xác định như sau:

$$q_k = \sum_{j=1}^N |\psi_{k,j}|. \quad (7.20c)$$

Khi đó $\|\Psi\| < q$ nếu $q_k \leq q \forall k \leq N$. Ta sẽ xây dựng thuật toán để $q_k \in [\alpha q, q] \forall k \leq N$.

Mô tả thuật toán

Với sai số ϵ và các hằng số dương $q, \alpha < 1$ cho trước, thuật toán chúng ta gồm 2 pha để xác định các tham số σ_k và \mathbf{w} . Trong pha thứ nhất, ta sẽ xác định các σ_k để $q_k \in [\alpha q, q]$ với mọi k để cho chuẩn của ma trận Ψ tương ứng với chuẩn vectơ cho bởi $\|u\|_* = \sum_{j=1}^N |u_j|$ thuộc đoạn này. Pha sau tìm nghiệm gần đúng \mathbf{w}^* của (7.20a) bằng phương pháp lặp đơn. Thuật toán được đặc tả trong hình 7.14.

```

Procedure Huấn luyện mạng RBF
Begin
    for k=1 to N do
        Xác định các  $\sigma_k$  để  $q_k \in [\alpha q, q]$ ;
    End;
    Tìm  $\mathbf{W}$  bằng phương pháp lặp đơn;
end

```

Hình 7.14. Đặc tả thủ tục lắp huấn luyện mạng.

Pha thứ nhất: (Xác định các độ rộng hàm bán kính)

Bởi vì $\varphi_k(x^i)$ là hàm đơn điệu giảm của σ_k nên q_k là hàm đơn điệu tăng của σ_k . Dựa trên đặc tính này ta xây dựng thuật toán lặp xác định các tham số độ rộng.

Dùng hằng số điều chỉnh chính ban đầu $0 < \beta < 1$ tùy ý, với mỗi $k \leq N$, khởi tạo $\sigma_k = \sigma_0$, (có thể lấy là $\frac{1}{\sqrt{2} \cdot N}$). Tính σ_k theo công thức (7.20c) và kiểm tra điều kiện $q_k \in [\alpha q, q]$, nếu $q_k > q$ thì giảm σ_k , ngược lại tăng nó lên.

Với mọi $k=1$ tới N , thuật toán được mô tả trong bảng 7.2.

Bảng 7.2. Pha thứ nhất của thuật toán lặp

Bước 1. khởi tạo $\sigma \leftarrow \sigma_0$; //khởi tạo σ_k

$$\text{Tính } p = \sum_{j=1}^N e^{-\frac{\|x^j - x^k\|^2}{\sigma}}; // \text{tính } q_k$$

Bước 2 .Nếu $p \in [\alpha q, q]$ thì $\sigma_k \leftarrow \sigma$ và trở về bước 1 với $k := k+1$;

Bước 3. Nếu $p > q$ thì thực hiện //điều chỉnh tham số bán kính để $q_k \in [\alpha q, q]$

3.1. $\sigma^1 \leftarrow \beta\sigma$; //giảm tham số bán kính

$$\text{Tính } p = \sum_{j=1}^N e^{-\frac{\|x^j - x^k\|^2}{\sigma^1}}; // \text{tính lại } q_k$$

3.2. Nếu $p \in [\alpha q, q]$ thì $\sigma_k \leftarrow \sigma^1$ và trở về bước 1 với $k := k+1$;

3.3. Nếu $p > q$ thì $\sigma \leftarrow \sigma^1$ và trở lại 3.1;

3.4.Nếu $p < \alpha q$ thực hiện thủ tục lặp// Thủ tục chia đôi để tìm $\sigma_k \in [\sigma, \sigma^1]$

$$3.4.1. \sigma^0 \leftarrow \frac{\sigma + \sigma^1}{2};$$

$$\text{Tính } p = \sum_{j=1}^N e^{-\frac{\|x^j - x^k\|^2}{\sigma^0}};$$

3.4.2. Nếu $p \in [\alpha q, q]$ thì $\sigma_k \leftarrow \sigma^0$ và trở về bước 1 với $k := k+1$;

3.4.3. Nếu $p > q$ thì $\sigma \leftarrow \sigma^0$ và trở lại 3.4.1;

3.4.4. Nếu $p < \alpha q$ thì $\sigma^1 \leftarrow \sigma^0$ và trở lại 3.4.1;

Bước 4. Nếu $p < \alpha q$ thì thực hiện //điều chỉnh bán kính để $q_k \in [\alpha q, q]$

4.1. $\sigma^1 \leftarrow \frac{\sigma}{\beta}$; //tăng tham số bán kính

$$\text{Tính } p = \sum_{j=1}^N e^{-\frac{\|x^j - x^k\|^2}{\sigma^1}}; // \text{tính lại } q_k$$

4.2. Nếu $p \in [\alpha q, q]$ thì $\sigma_k \leftarrow \sigma^1$ và trở về bước 1 với $k := k+1$;

4.3. Nếu $p < \alpha q$ thì $\sigma \leftarrow \sigma^1$ và trở lại 4.1;

4.4. Nếu $p > q$ thực hiện thủ tục lặp// Thủ tục chia đôi để tìm $\sigma_k \in [\sigma, \sigma^1]$

$$4.4.1. \sigma^0 \leftarrow \frac{\sigma + \sigma^1}{2};$$

$$\text{Tính } p = \sum_{j=1}^N e^{-\frac{\|x^j - x^k\|^2}{\sigma^0}};$$

4.4.2. Nếu $p \in [\alpha q, q]$ thì $\sigma_k \leftarrow \sigma^0$ và trở về bước 1 với $k := k+1$;

4.4.3. Nếu $p > q$ thì $\sigma^1 \leftarrow \sigma^0$ và trở lại 4.4.1;

4.4.4. Nếu $p < \alpha q$ thì $\sigma \leftarrow \sigma^0$ và trở lại 4.4.1;

Pha thứ hai. (Xác định trọng số tầng ra)

Nghiệm gần đúng \mathbf{w}^* cho trọng số tầng ra \mathbf{w} của hệ (7.20a) được tìm bằng phương pháp lặp đơn như mô tả trong bảng 7.3.

Bảng 7.3. Thủ tục lặp đơn tìm trọng số tầng ra

Bước 1. Khởi tạo $\mathbf{w}^0 = \mathbf{z}$;

Bước 2. Tính $\mathbf{w}^1 = \Psi \mathbf{w}^0 + \mathbf{z}$;

Bước 3. Nếu điều kiện kết thúc chưa đúng thì $\mathbf{w}^0 \leftarrow \mathbf{w}^1$ và trở lại bước 2 ;

Bước 4. $\mathbf{w}^* \approx \mathbf{w}^1$.

Điều kiện kết thúc

Ta ký hiệu chuẩn $\|\mathbf{u}\|_* = \sum_{j=1}^N |u_j|$ cho mỗi vecto N -chiều \mathbf{u} . Khi đó có thể chọn một trong hai điều kiện sau để kết thúc thuật toán.

$$a) \quad \frac{q}{1-q} \left\| W^1 - W^0 \right\|_* \leq \varepsilon \quad (7.21a)$$

$$b) \quad t \geq \frac{\ln \frac{\varepsilon(1-q)}{\|Z\|_*}}{\ln q} = \frac{\ln \varepsilon - \ln \|Z\|_* + \ln(1-q)}{\ln q}, \quad (7.21b)$$

trong đó t là số lần lặp.

Tính hội tụ và sai số

Thuật toán lặp này luôn hội tụ và ta có đánh giá sai số của lời giải \mathbf{w}^1 với nghiệm đúng \mathbf{w}^* :

$$\|\mathbf{w}^1 - \mathbf{w}^*\|_* \leq \frac{q}{q-1} \|\mathbf{w}^1 - \mathbf{w}^0\|_* . \quad (7.22)$$

Thực nghiệm cho thấy thuật toán huấn luyện rất nhanh, sai số huấn luyện nhỏ và khi các mốc nội suy có phân bố khoảng cách gần với cách đều thì tính tổng hóa tốt hơn hẳn các phương pháp khác hiện hành.

Mạng nội suy với mốc cách đều.

Ta xét trường hợp hợp mốc nội suy cách đều trong không gian n-chiều và biến x_k có bước h_k tương ứng. Ta dùng đa chỉ số để biểu diễn các mốc nội suy như sau :

$$\mathbf{x}^{i_1, \dots, i_n} = (x_1^{i_1}, \dots, x_n^{i_n}); x_k^{i_k} = x_k^0 + i_k \times h_k; k = 1, 2, \dots, n, \quad (7.23a)$$

trong đó i_k nhận giá trị từ 1 đến N_k .

Ký hiệu chuẩn Mahalanobis:

$$\|x\|_A = \sqrt{x^T A x}, \quad (7.23b)$$

trong đó A là ma trận

$$A = \begin{pmatrix} \frac{1}{h_1^2} & 0 & \dots & 0 \\ 0 & \frac{1}{h_2^2} & \dots & 0 \\ \dots & \dots & \ddots & \dots \\ 0 & 0 & \dots & \frac{1}{h_n^2} \end{pmatrix}. \quad (7.23c)$$

Biểu thức (7.15b) bây giờ thành:

$$a^2(\mathbf{x}) = \varphi(\mathbf{x}) = \sum_{i_1, \dots, i_n=1}^{N_1, \dots, N_n} w_{i_1, \dots, i_n} \varphi_{i_1, \dots, i_n}(\mathbf{x}) + w_0, \quad (7.24a)$$

$$\text{trong đó } \varphi_{i_1, \dots, i_n}(\mathbf{x}) = \exp \left(- \frac{\|\mathbf{x} - \mathbf{x}^{i_1, \dots, i_n}\|_A^2}{\sigma_{i_1, \dots, i_n}^2} \right).$$

Lúc này ta chọn trước tham số độ rộng:

$$\sigma_{i_1, \dots, i_n} = \sqrt{\frac{1}{\ln\left(\frac{6}{\pi\sqrt{1+q}-1}\right)}} \quad (7.24b)$$

thì $q_{i_1, \dots, i_n} \leq q$. Khi đó thuật toán hai pha ở trên trở thành thuật toán một pha. Thực nghiệm cho thấy thuật toán này có thời gian huấn luyện nhanh và chất lượng mạng tốt hơn hẳn các thuật toán huấn luyện nhanh đã được đề xuất.

Chú ý. Thay cho việc sử dụng chuẩn Mahalanobis cho bởi biểu thức (7.32b), ta có thể dùng phép đổi biến:

$$\bar{x}_k = \frac{(x_i - x_k^0)}{h_k} \quad \forall K = 1, \dots, N, \quad (7.25)$$

thì $\bar{x}_k \in [0, N_k]$ $\forall K = 1, \dots, N$ Euclide mà kết quả không thay đổi.

KẾT LUẬN

Mạng nơron nhân tạo là tiếp cận mô phỏng kiến trúc, tính năng và cơ chế xử lý của hệ thần kinh sinh học. và có ứng dụng rộng rãi. Những cách kết nối đa dạng và các luật học khác nhau cho ta các mạng nơron có tính năng ứng dụng khác nhau. Các cơ chế xử lý phi tuyến, song song, đáp ứng theo mẫu đào tạo...cho phép ta tạo nên các *chương trình* có tính năng vượt trội so với cách tiếp cận truyền thống. Các chương trình này có thể cài đặt cứng hóa như là các *máy*.

Perceptron của Rosenblatt là kiểu mạng được đề xuất sớm nhất nhưng đến nay vẫn còn được sử dụng. Luật học của mạng đơn giản, dễ cài đặt, tuy nhiên chỉ dùng được khi các lớp tách được tuyến tính. Mạng ADALINE với luật học Widrow-Hoff cho phép giải các bài toán nhận dạng mẫu với hàm phân biệt tuyến tính hoặc hồi quy tuyến tính nhiều chiều, khi kết nối với các thành phần khác, nó có nhiều ứng dụng. Một tín hiệu và đường trẽ lặp kết hợp với mạng ADALINE cho ta một bộ hồi quy trung bình trượt, chúng có thể dùng làm bộ lọc tín hiệu như là ví dụ ứng dụng.

Mạng MLP là kiểu mạng đang được sử dụng rộng rãi nhất hiện nay, đặc biệt cho các bài toán hồi quy nhiều biến. Nhược điểm của mạng là thời gian huấn luyện lâu, thường chỉ cho lời giải là cực trị địa phương của SSE, bài toán tối ưu kiến trúc tầng ẩn tới nay vẫn là bài toán mở.

Mạng nơron RBF có ưu điểm là huấn luyện nhanh và SSE có duy nhất một cực tiểu toàn cục nên dễ xây dựng thuật toán huấn luyện cho lời giải gần đúng. Thuật toán lặp huấn luyện mạng nội suy dễ cài đặt và có ưu điểm nổi trội so với các mạng RBF nội suy khác. Tuy có nhiều ưu điểm nhưng phạm vi ứng dụng của mạng RBF hạn chế hơn mạng MLP vì nó không dùng để ngoại suy được.

BÀI TẬP

1. Cho tập mẫu : { $\mathbf{p}_1=\begin{pmatrix} 1 \\ 2 \end{pmatrix}; t_1=1\}$, { $\mathbf{p}_2=\begin{pmatrix} -1 \\ 2 \end{pmatrix}; t_2=0\}$, { $\mathbf{p}_3=\begin{pmatrix} 0 \\ -1 \end{pmatrix}; t_3=0\}$.

a) Xây dựng mạng perceptron phân biệt tập mẫu trên với giá trị khởi tạo: $\mathbf{w}^T = (1,0 \ -0,8)$, giá trị ngưỡng luôn lấy $b=0$.

b) Xây dựng perceptron với giá trị khởi tạo như câu a) nhưng giá trị ngưỡng b được điều chỉnh từng bước theo luật học của Rosenblatt, cho nhận xét so sánh kết quả.

2. Xây dựng mạng perceptron phân biệt bốn lớp dựa trên tập mẫu :

$$\text{lớp 1: } \left\{ \mathbf{p}_1 = \begin{bmatrix} 1 \\ 1 \end{bmatrix}, \mathbf{p}_2 = \begin{bmatrix} 1 \\ 2 \end{bmatrix} \right\}, \text{lớp 2: } \left\{ \mathbf{p}_3 = \begin{bmatrix} 2 \\ -1 \end{bmatrix}, \mathbf{p}_4 = \begin{bmatrix} 2 \\ 0 \end{bmatrix} \right\};$$

$$\text{lớp 3: } \left\{ \mathbf{p}_5 = \begin{bmatrix} -1 \\ 2 \end{bmatrix}, \mathbf{p}_6 = \begin{bmatrix} -2 \\ 1 \end{bmatrix} \right\}, \text{lớp 4: } \left\{ \mathbf{p}_7 = \begin{bmatrix} -1 \\ -1 \end{bmatrix}, \mathbf{p}_8 = \begin{bmatrix} -2 \\ -2 \end{bmatrix} \right\}.$$

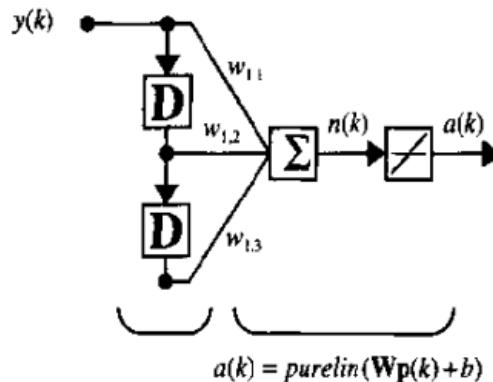
3. Cho bộ lọc thích nghi như trong hình 7.15 , trong đó:

$$w_{1,1} = 2, w_{1,2} = -1, w_{1,3} = 3, \text{ đầu vào } y(k) = \begin{cases} 5 & k = 0 \\ -4 & k = 1 \\ 0 & k \notin \{0, 1\} \end{cases}$$

Hãy trả lời các câu hỏi sau:

a) Tính đầu ra $a(k)$ với $k \geq 0$.

b) $y(0)$ có đóng góp trong những đầu ra nào?



Hình 7.15. Một bộ lọc thích nghi

5. Thiết kế kiến trúc một mạng truyền tới 2 tầng noron để nội suy hàm nhiều biến

$f(x) = \begin{bmatrix} f_1(x) \\ f_2(x) \\ f_3(x) \end{bmatrix}$ có miền xác định là R^5 và nhận giá trị trong R^3 khi tập quan sát có 100

đối tượng dữ liệu, tức là $D = \{\mathbf{x}^k, \mathbf{y}^k\}_{k=1}^{100}$.

6. Thiết kế kiến trúc mạng noron RBF cho hàm $f: R^5 \rightarrow R^4$ khi tập dữ liệu quan sát có 1000 đối tượng, tức là $D = \{\mathbf{x}^k, \mathbf{y}^k\}_{k=1}^{1000}$.

7. Viết công thức điều chỉnh trọng số cho mạng MLP có hai tầng noron và hàm chuyển log-sigmoid

8. Chỉ ra rằng mạng MLP nhiều tầng với hàm chuyển tuyến tính tương đương với mạng tuyến tính một tầng.

Chương 8

CÁC MÔ HÌNH HỌC ĐỊA PHƯƠNG

Trong đa số các phương pháp học, xấp xỉ cho hàm đích được tìm dựa trên toàn bộ tập dữ liệu quan sát được trong thời gian *học/huấn luyện*, sau đó nhãn của đối tượng mới được xác định nhờ hàm xấp xỉ này. Khác với các phương pháp học này, trong các phương pháp học địa phương, người ta thường dựa trên thông tin của các mẫu huấn luyện được lưu trữ gần với mẫu mới để tìm nhãn cho mẫu mới đang xét. Quy tắc phân lớp k -NN trong mục 5.3 và các kỹ thuật phi tham số giới thiệu trong mục 6.1.2 thuộc loại phương pháp học địa phương.

Chương này khởi đầu bằng giới thiệu hai phương pháp địa phương cho bài toán hồi quy: k -láng giềng gần nhất (k -NN) và hồi quy trọng số địa phương, tiếp theo là hai kiểu mạng neural RBF: mạng RBF hồi quy và mạng RBF nội suy địa phương, sau đó giới thiệu phương pháp lập luận dựa trên tình huống.

8.1. HỌC K-LÁNG GIỀNG GẦN NHẤT (K-NN)

Với mỗi điểm $\mathbf{x} \in \mathcal{R}^n$ chưa biết giá trị hàm số, k -NN là phương pháp đơn giản tìm giá trị xấp xỉ dựa trên giá trị của hàm tại k điểm quan sát được gần nó.

8.1.1. Lược đồ tổng quát

Ta trở lại với bài toán *hồi quy/nội suy* hàm số. Xét hàm giá trị thực $y = f(\mathbf{x})$ xác định trong không gian \mathcal{R}^n và tập dữ liệu quan sát được $D = \{\mathbf{x}^k, y^k\}_{k=1}^N$ thỏa mãn:

$$y^i = f(\mathbf{x}^i) + \varepsilon_i \quad \forall i = 1, \dots, N, \tag{8.1a}$$

trong đó ε_i là các đại lượng ngẫu nhiên độc lập cùng phân phối có kỳ vọng bằng không. Ta cần xác định hàm hồi quy $g(\mathbf{x})$ để xấp xỉ $f(\mathbf{x})$ với mọi $\mathbf{x} \in \mathcal{R}^n$ đủ tốt theo một tiêu chuẩn nào đó. Với bài toán nội suy, ta cần có:

$$g(\mathbf{x}^i) = y^i \quad \forall i = 1, \dots, N. \tag{8.1b}$$

Trong phương pháp k -NN, ta không xác định được dạng tương minh của hàm *hồi quy/nội suy* $g(\mathbf{x})$ mà sẽ xác định giá trị cho các đối tượng mới \mathbf{x} dựa vào k đối tượng trong tập $D_x = \{\mathbf{z}^k\}_{k=1}^N$ gần với \mathbf{x} nhất.

Với mỗi $\mathbf{x} \in \mathbb{R}^n$, ta ký hiệu tập k đối tượng trong D_x gần nó nhất là $\{\mathbf{z}^i\}_{i=1}^k$ thì $g(\mathbf{x})$ thường được xác định dưới dạng:

$$g(\mathbf{x}) = \sum_{i=1}^k \rho_i(\mathbf{x}) y(\mathbf{z}^i), \quad (8.2a)$$

trong đó $\rho_i(\mathbf{x})$ là các trọng số phụ thuộc \mathbf{x} , được chuẩn hóa, tức là: $\sum_{i=1}^k \rho_i(\mathbf{x}) = 1$. Một cách đơn giản có thể xác định $g(\mathbf{x})$ bởi trọng số đều:

$$g(\mathbf{x}) = \frac{1}{k} \sum_{i=1}^k y(\mathbf{z}^i). \quad (8.2b)$$

Với bài toán có nhiều trăng và k lớn thì biểu thức (8.2b) cho phép khử được nhiễu. Tuy nhiên, nếu \mathbf{x} gần với các đối tượng đã quan sát được trong D_x thì sai số có thể lớn và không thích hợp cho bài toán nội suy. Thuật toán sau có thể khắc phục được nhược điểm này.

8.1.2. Thuật toán nghịch đảo khoảng cách

Thuật toán này phát triển dựa trên lập luận rằng các trọng số $\rho_i(\mathbf{x})$ trong biểu thức (8.2a) ứng với các đối tượng \mathbf{z}^i càng gần với \mathbf{x} càng phải có giá trị lớn để đảm bảo tính liên tục của hàm nội suy tại các mốc quan sát được.

Thuật toán thực hiện như sau. Với mỗi đối tượng \mathbf{x} , ta dùng khoảng cách Euclide:

$$\forall \mathbf{x}, \mathbf{y} \in \mathbb{R}^n \quad d(\mathbf{x}, \mathbf{y}) = \sqrt{\sum_{i=1}^n |x_i - y_i|^2}, \quad (8.2c)$$

để xác định k đối tượng $\{\mathbf{z}^i\}_{i=1}^k \in D_x$ gần \mathbf{x} nhất. Giá trị $g(\mathbf{x})$ được tính theo công thức (8.2a) với các trọng số $\rho_i(\mathbf{x})$ là:

$$\rho_i(\mathbf{x}) = \frac{1/d(\mathbf{x}, \mathbf{z}^i)}{\sum_{j=1}^k 1/d(\mathbf{x}, \mathbf{z}^j)}. \quad (8.2d)$$

Thuật toán được mô tả trong hình 8.1.

Procedure Thuật toán nghịch đảo khoảng cách tính $g(\mathbf{x})$

Begin

1. Xác định $\{\mathbf{z}^i\}_{i=1}^k \in D_x$ gần \mathbf{x} nhất;
2. Tính các trọng số $\rho_i(\mathbf{x})$; // theo (8.2d)
3. Tính $g(\mathbf{x})$; // theo (8.2a);

End

Hình 8.1. Thuật toán nghịch đảo khoảng cách tính $g(\mathbf{x})$

Dễ dàng chứng minh được hàm nội suy $g(\mathbf{x})$ nay liên tục tại tập điểm quan sát được D_x , vì vậy là một hàm liên tục.

Nhận xét.

- Thuật toán này đơn giản, dễ dùng nên thường được ứng dụng cho các bài toán không đòi hỏi xác định trước hàm nội suy, đặc biệt khi các điểm cần tính không nhiều.
- Mặc dù trên đây chỉ xét các đối tượng có đặc trưng thực nhưng người đọc có thể mở rộng áp dụng cho các đối tượng có đặc trưng khác, khi trên chúng xác định một khoảng cách để áp dụng công thức (8.2d).

Tuy nhiên, thuật toán có một số nhược điểm sau:

- Giá trị hàm *hồi quy/nội suy* phụ thuộc vào giá trị k và khó xác định k tối ưu.
- Hàm hồi quy không xác định trước trong giai đoạn học được nên không áp dụng được đối với các bài toán cần xác định nhanh giá trị $g(\mathbf{x})$ khi chạy ứng dụng.
- Như đã nói ở trên, hàm hồi quy là hàm liên tục. Vì vậy khi dữ liệu huấn luyện có nhiều trăng thì kết quả hồi quy không tốt nếu so với phương pháp hồi quy tuyến tính địa phương dưới đây.

8.2. HỒI QUY TRỌNG SỐ ĐỊA PHƯƠNG

Hồi quy tuyến tính địa phương tổng quát hóa của phương pháp k -NN bằng việc tìm một hàm xấp xỉ f trên một miền địa phương bao quanh \mathbf{x} , nhờ đó tính được giá trị hàm hồi quy tại đó.

8.2.1. Lược đồ tổng quát

Trong phương pháp này, người ta xấp xỉ hàm mục tiêu trong miền lân cận của \mathbf{x} gần nó nhờ dùng một hàm tuyến tính, hàm bậc hai, mạng nơron MLP hay một số dạng hàm khác, dựa trên các mốc trong D_x .

Xét tập mẫu $D = \{\mathbf{x}^k, y^k\}_{k=1}^N$ thỏa mãn (8.1a) và điểm \mathbf{x} trong \mathcal{R}^n cần tính giá trị gần đúng $f(\mathbf{x})$ bởi giá trị hàm hồi quy $g(\mathbf{x})$. Trong mục 2.4 ở chương 2, đã giới thiệu phương pháp tìm hàm hồi quy nhờ cực tiểu tổng bình phương sai số:

$$E = \sum_{i=1}^N (g(\mathbf{x}^i) - y^i)^2. \quad (8.3a)$$

Trong thực tế, giá trị hàm số ở những điểm xa với điểm \mathbf{x} được xét thì ít ảnh hưởng tới giá trị của hàm tại điểm này nên người ta có thể thêm vào trọng số xác định bởi hàm đơn điệu giảm $K(d(\mathbf{x}^i, \mathbf{x}))$

$$E = \sum_{i=1}^N (g(\mathbf{x}^i) - y^i)^2 K(d(\mathbf{x}^i, \mathbf{x})), \quad (8.3b)$$

trong đó $\lim_{t \rightarrow \infty} k(t) = 0$. (8.3c)

Điều kiện (8.3c) thể hiện tính ảnh hưởng *địa phương* của các điểm quan sát trong hàm đánh giá E ở (8.3b), khoảng cách Euclidean thường được dùng trong biểu thức tính hàm $K(d(\mathbf{x}^i, \mathbf{x}))$.

Trong các ứng dụng, thay cho đòi hỏi điều kiện (8.3c), với số tự nhiên k chọn trước, người ta chỉ xét k điểm quan sát $\{\mathbf{z}^i\}_{i=1}^k$ gần \mathbf{x} nhất để tính giá trị hồi quy hàm số tại \mathbf{x} và hàm E trong biểu thức (8.3b) trở thành:

$$E = \sum_{i=1}^k (g(\mathbf{z}^i) - y(\mathbf{z}^i))^2 K(d(\mathbf{z}^i, \mathbf{x})), \quad (8.4a)$$

hoặc đơn giản hơn

$$E = \sum_{i=1}^k (g(\mathbf{z}^i) - y(\mathbf{z}^i))^2 \quad (8.4b)$$

khi lấy $K(d(\mathbf{z}^i, \mathbf{x})) = 1$.

Cực tiểu của E có thể tìm nhờ thuật toán gradient hoặc giải trực tiếp hệ phương trình tìm điểm dừng của nó.

Từ lược đồ trên, trên ta thấy cụm từ “hồi quy trọng số địa phương” được hiểu như sau: gọi là địa phương bởi vì hàm này được xấp xỉ chỉ dựa trên dữ liệu gần điểm truy vấn được xét, gọi là trọng số bởi vì sự đóng góp của mỗi mẫu huấn luyện được gán trọng số phụ thuộc vào khoảng cách của nó đến điểm truy vấn, còn hồi quy là khái niệm được dùng rộng rãi trong học thống kê cho bài toán xấp xỉ các hàm nhận giá trị thực.

Như đã biết, hàm hồi quy có thể là một hàm phụ thuộc tham số tùy ý, trong đó thông dụng nhất là hàm tuyến tính.

8.2.2. Hồi quy tuyến tính địa phương

Trong trường hợp hàm hồi quy g là hàm tuyến tính:

$$g(\mathbf{x}) = w_0 + w_1 x_1 + \cdots + w_n x_n \quad (8.5)$$

trong đó $\mathbf{x} = (x_0, \dots, x_n)^T$. Khi đó các biểu thức (8.4a-b) sẽ là:

$$E = \sum_{i=1}^k \left(\sum_{i=1}^n w_i x_i + w_0 - y(\mathbf{z}^i) \right)^2 K(d(\mathbf{z}^i, \mathbf{x})), \quad (8.6a)$$

$$E = \sum_{i=1}^k \left(\sum_{i=1}^n w_i x_i + w_0 - y(\mathbf{z}^i) \right)^2. \quad (8.6b)$$

Các hệ số w_i cực tiểu E theo công thức (8.6a-b) có thể tìm nhờ giải trực tiếp hệ phương trình $\frac{\partial E}{w_m} = 0$ hoặc phương pháp gradient đã biết.

Vì g là hàm tuyến tính nên dễ dàng tính được gradient(E) và khi giải trực tiếp thì các w_i tương ứng là nghiệm của hệ:

$$\sum_{i=1}^k (\sum_{j=1}^n (w_j x_j^i + w_0) - y(\mathbf{z}^i)) K(d(\mathbf{z}^i, \mathbf{x})) x_m^i = 0 \quad \forall m = 0, 1, \dots, n \quad (8.7a)$$

$$\sum_{i=1}^k (\sum_{j=1}^n (w_j x_j^i + w_0)) - y(\mathbf{z}^i) x_m^i = 0 \quad \forall m = 0, 1, \dots, n. \quad (8.7b)$$

Đổi thứ tự lấy tổng ta được các hệ phương trình tuyến tính tương đương:

$$\sum_{j=1}^n (\sum_{i=1}^k (x_j^i K(d(\mathbf{z}^i, \mathbf{x})) x_m^i) w_j + n w_0) = \sum_{i=1}^k y(\mathbf{z}^i) \quad \forall i = 0, 1, \dots, n \quad (8.7c)$$

$$\sum_{j=1}^n (\sum_{i=1}^k x_j^i x_m^i) w_j + n w_0 = \sum_{i=1}^k y(\mathbf{z}^i) \quad \forall i = 0, 1, \dots, n \quad (8.7d)$$

Khi n lớn, nghiệm của các hệ trên có thể không ổn định do tích lũy sai số trong tính toán, ta có thể dùng phương pháp gradient. Thuật toán gradient để tìm cực tiểu E được mô tả trong bảng 8.1, điều kiện dừng của nó như đã xét trong chương 2.

Bảng 8.1. Thuật toán gradient tìm giá trị hàm hồi quy $g(\mathbf{x})$

Bước 1. Khởi tạo các giá trị $w_i, i = 0, \dots, n$ tùy ý;

Bước 2. Tìm ka điểm $\{\mathbf{z}^i\}_{i=1}^k$ trong D_x gần \mathbf{x} nhất

Bước 3. Thực hiện lặp

3.1. Tính các giá trị $g(\mathbf{z}^i) \forall i = 1, \dots, k$

3.2. $w_i \leftarrow w_i + \Delta w_i \quad \forall i // \Delta w_i$ lấy theo công thức (8.8a/b)

3.3. Kiểm tra điều kiện dừng.

$$\Delta w_i = \alpha \sum_{i=1}^k (g(\mathbf{z}^i) - y(\mathbf{z}^i)) K(d(\mathbf{z}^i, \mathbf{x})), \quad (8.8a)$$

$$\Delta w_i = \alpha \sum_{i=1}^k (g(\mathbf{z}^i) - y(\mathbf{z}^i)), \quad (8.8b)$$

trong đó $\alpha \in (0, 1)$ là tốc độ học được chọn thích hợp.

Nhận xét

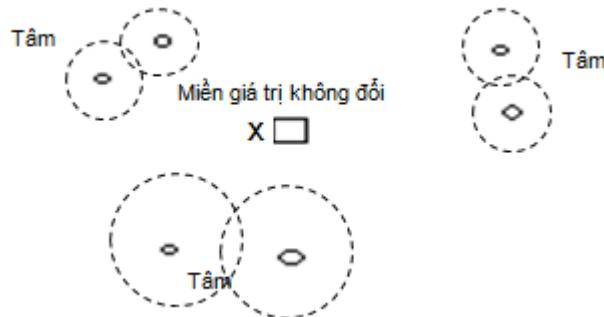
Cũng như k-NN ở mục trước, phương pháp này và thuộc loại học dựa trên mẫu (Instance based learning) và có chung các đặc điểm khi sử dụng, chẳng hạn: thời gian huấn luyện ít nhưng không tính trước được hàm hồi quy. Vì vậy hạn chế phạm vi áp dụng.

Tuy nhiên phương pháp này là công cụ tốt để tạo dữ liệu cho các phương pháp khác. Mạng nơron RBF hồi quy là một ví dụ ứng dụng của phương pháp này.

8.3. MẠNG NORON RBF HỒI QUY

8.3.1. Đặt vấn đề

Bởi vì các hàm RBF gần như triệt tiêu tại những điểm xa tâm nên các mạng noron RBF cũng thuộc loại học địa phương. Các mạng RBF nội suy ở chương trước có nhiều ưu điểm nhưng khi dùng cho các bài toán xấp xỉ thì chất lượng thấp. Hình 8.2 chỉ ra trường hợp khi các điểm mốc quan sát được có từng cặp một gần nhau thì những điểm xa tâm có giá trị đều ra bằng w_0 không đổi, vì vậy mạng này chỉ đáp ứng được điều kiện tương thích (8.1b) ở các điểm dữ liệu cho bởi tập D mà thôi.



Hình 8.2. Khi tâm phân bố không tốt, các điểm x xa tâm có giá trị không đổi

Để khắc phục hạn chế này ta sử dụng phương pháp hồi quy tuyến tính địa phương để tạo ra tập dữ liệu mới có mốc cách đều và sử dụng thuật toán lặp một pha đã biết để huấn luyện mạng noron RBF nội suy.

8.3.2. Xây dựng mạng noron RBF hồi quy

Giả sử tập dữ liệu D_x nằm trong hình hộp n -chiều $B = \prod_{i=1}^n [l_i, b_i]$, mạng noron RBF hồi quy được xây dựng như sau.

Tùy theo mật độ phân bố của dữ liệu trên miền giá trị của mỗi thuộc tính, ta chọn các bước h_i thích hợp cho mỗi thuộc tính x_i và tạo ra một lưới điểm mốc cách đều $\{\mathbf{z}^i\}_{i=1}^M$ trên B (Thực nghiệm cho thấy nên chọn các bước sao cho M lớn hơn N).

Với một số tự nhiên k chọn trước, dùng hồi quy tuyến tính địa phương để xấp xỉ hàm hồi quy $g(\mathbf{x})$ tại mọi điểm \mathbf{x} trên lưới này. Sau đó sử dụng tập dữ liệu $\{(\mathbf{z}^i, g(\mathbf{z}^i))\}_{i=1}^M$ vừa tính được để xây dựng mạng hồi quy nhờ áp dụng thuật toán huấn luyện một pha trong mục 7.4.2 chương trước.

Thuật toán xây dựng mạng RBF hồi quy được mô tả trong bảng 8.2. Thực nghiệm trên dữ liệu thực và dữ liệu mô phỏng cho thấy mạng này có chất lượng tốt.

Bảng 8.2. Thuật toán xây dựng mạng nơron RBF hồi quy

Bước 1. Tạo lưới cách đều $\{\mathbf{z}^i\}_{i=1}^M \subset B \in \mathbb{R}^k$, chọn $k > n$;

Bước 2. Tính $g(\mathbf{z}^i) \forall i = 1, \dots, M$; // Dùng phương pháp hồi quy tuyến tính địa phương

Bước 3. Huấn luyện mạng hồi quy; // Dùng thuật toán huấn luyện mạng nội suy với mốc cách đều

8.4. MẠNG RBF ĐỊA PHƯƠNG

Trong nhiều bài toán, tập dữ liệu quan sát được D rất lớn hoặc thường xuyên có bổ sung dữ liệu mới thì có thể dùng các mạng RBF địa phương nhò dùng kỹ thuật cây k-d sau đây để xây dựng mạng. Để dễ theo dõi, trong mục này chỉ giới thiệu xây dựng mạng nội suy, người đọc có thể ứng dụng kỹ thuật cây k-d để xây dựng mạng hồi quy địa phương như là bài tập.

8.4.1. Kiến trúc và thủ tục xây dựng mạng

Giả sử tập mẫu $D = \{\mathbf{x}^k, y^k\}_{k=1}^N$ nằm trong miền đóng giới nội $B = \prod_{i=1}^n [a_i, b_i]$ và

N lớn. Ta định trước số M cho cận trên của số điểm trong mỗi cụm con và chia miền B thành các hình hộp n chiều $\{B_j\}_{j=1}^L$ sao cho số mốc trong mỗi cụm nhỏ hơn M nhò thuật toán phân cụm nhò cây k-d (mô tả trong mục 8.4.2 ở dưới). Sau đó sử dụng thuật toán huấn luyện lặp đã nói ở mục 7.4 chương trước để huấn luyện các mạng RBF cho mỗi miền con B_j .

Như vậy Mạng RBF địa phương được xây dựng nhò kết hợp giữa thủ tục cây k-d với thủ tục xây dựng các mạng RBF con.

Khi có một dữ liệu mới, ta cần định vị nó thuộc miền B_q nào trong tập $\{B_j\}_{j=1}^L$ thì chỉ cần huấn luyện lại mạng RBF trên miền đó. Cây k-d giúp ta dễ dàng xây dựng bộ định vị này. Nếu số điểm dữ liệu mẫu trong B_q lớn hơn M , ta chia B_q thành hai miền con theo thủ tục cây k-d và huấn luyện lại các mạng con RBF trên miền này nhò dùng

độ rộng bán kính và trọng số tầng ra đang có làm giá trị khởi tạo. Còn khi số điểm dữ liệu mẫu trong B_q không lớn hơn M thì bổ sung dữ liệu mới và huấn luyện lại.

Nhờ số điểm dữ liệu trên mỗi miền con ít nên thời gian huấn luyện mạng RBF địa phương nhanh hơn thuật toán hai pha. Hơn nữa, chất lượng tổng quát hóa của mạng mới cũng tốt hơn. Đặc biệt, khi có dữ liệu quan sát bổ sung thì thời gian huấn luyện lại mạng con là không đáng kể so với mạng đầy đủ huấn luyện bằng phương pháp lặp. Thủ tục xây dựng mạng được đặc tả trong hình 8.3.

Procedure Xây dựng mạng RBF địa phương;

Begin

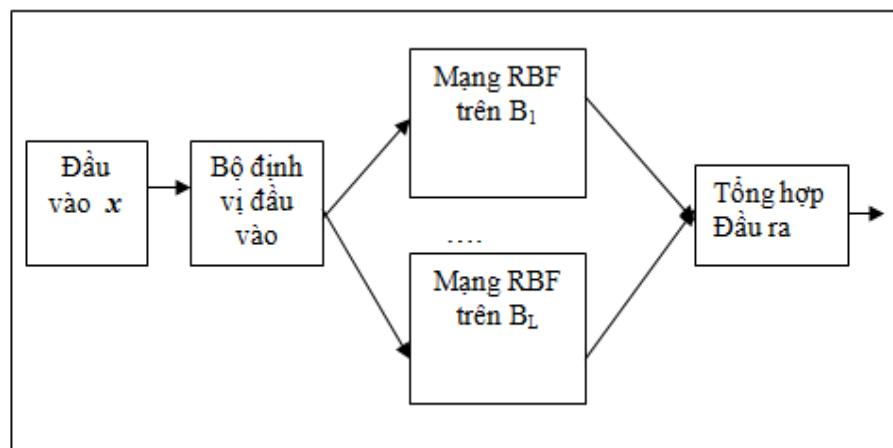
1. Phân B thành các miền con B_1, \dots, B_k ; // sử dụng thuật toán phân cụm cây $k-d$ để số mốc trong mỗi cụm con không vượt quá M .
2. Xây dựng bộ định vị đầu vào cho các mạng RBF con;
3. Huấn luyện các mạng RBF con; // Dùng thuật toán huấn luyện lặp
4. Kết nối bộ định vị đầu vào với các mạng con để được mạng RBF địa phương.

End

Hình 8.3. Thủ tục xây dựng mạng RBF nội suy địa phương

Sau đó xây dựng thủ tục để định vị mỗi điểm mới x trong B thuộc miền con B_j nào trong tập $\{B_j\}_{j=1}^L$.

Kiến trúc mạng được mô tả trong hình 8.4, trong pha huấn luyện, thành phần *tổng hợp đầu ra* không hoạt động, mỗi mạng con được huấn luyện độc lập.



Hình 8.4. Mô hình kiến trúc mạng RBF địa phương

Trong khi nội suy, với mỗi giá trị vào x , bộ định vị sẽ xác định mang con tương ứng, còn tất cả các mạng con khác có giá trị vào rỗng, do đó giá trị đầu ra của tất cả các mạng con đều là rỗng ngoại trừ mạng con có đầu vào x . Sau đó thành phần *tổng hợp đầu ra* là tổng đầu ra của các mạng RBF con.

Với cấu trúc mạng RBF địa phương mới này, hàm nội suy khả vi liên tục trên từng miền con nhưng không liên tục trên toàn miền B được xét. Khi M nhỏ thì thời gian huấn luyện và huấn luyện lại cũng như sai số giảm. Tuy nhiên, nếu số mạng con RBF nhiều thì sẽ làm cho mạng phức tạp hơn. Đặc điểm này chính là điều không mong muốn trong giải tích số. Như thế, việc lựa chọn M phải thích hợp để cân bằng giữa các điều kiện đó.

8.4.2. Thuật toán phân cụm nhờ cây k-d

Cây K-d được Bentley đề xuất để tổ chức dữ liệu, các biến thể của nó là công cụ hiệu quả để phân cụm nhanh dữ liệu. Thuật toán sau đây sửa đổi nhỏ kỹ thuật *k-d* thông dụng để phân hình hộp n -chiều B chứa N đối tượng dữ liệu thành các hình hộp con B_1, \dots, B_k sao cho mỗi hình hộp B_j chứa không quá M đối tượng dữ liệu.

Thuật toán có thể mô tả bởi quá trình xây dựng một cây nhị phân n -chiều có gốc là hình hộp B chứa N đối tượng dữ liệu, mỗi nút ở độ sâu l là hình hộp n -chiều có được nhờ một nhát cắt trực giao với cạnh lớn nhất của hình hộp cha chia thành hai hình hộp con để chúng chứa số lượng dữ liệu xấp xỉ nhau. Như vậy một hình hộp con chứa những điểm dữ liệu ứng với giá thuộc tính có cạnh chia nhỏ hơn điểm cắt và hình hộp còn lại có giá trị lớn hơn. Thủ tục chia đôi hình hộp n -chiều $B_j = \prod_{i=1}^n [a_i^j, b_i^j]$ thành hai hình hộp con được đặc tả trong hình 8.5.

Procedure chia đôi hình hộp n -chiều $B_j = \prod_{i=1}^n [a_i^j, b_i^j]$

Begin

Bước 1. Chọn i sao cho cạnh $[a_i^j, b_i^j]$ của B_j là lớn nhất; // Chọn phương cho siêu mặt cắt

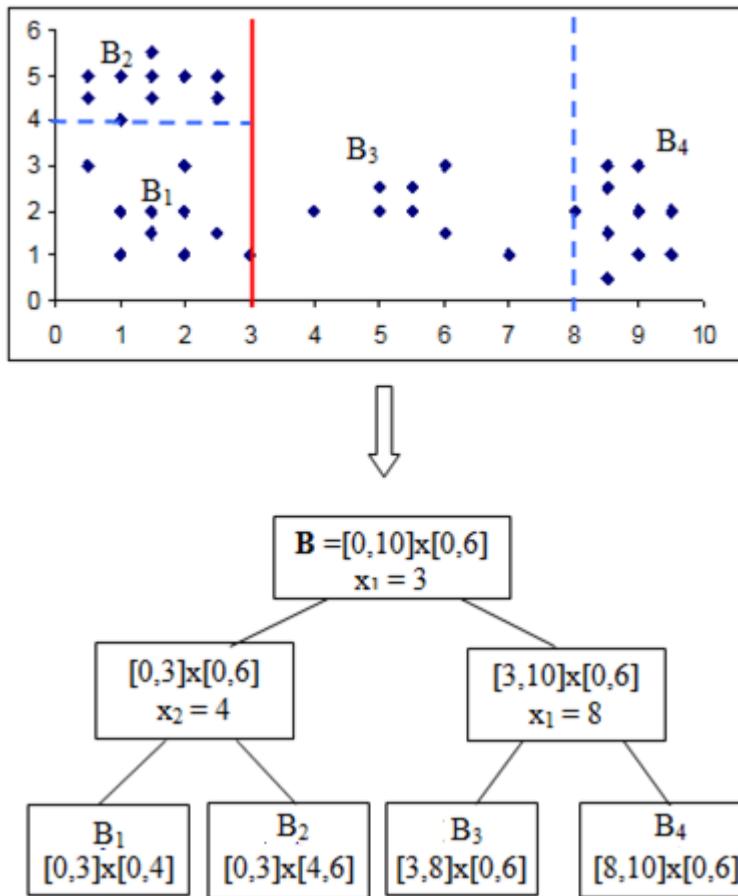
Bước 2. Chia $B_j = B^1 \cup B^2$; // Chia cạnh $[a_i^j, b_i^j]$ bởi nhát cắt trực giao và qua ít nhất một điểm dữ liệu sao cho hai hình hộp nhận được từ nó chứa số đối tượng dữ liệu bằng nhau hoặc lệch 1, với quy ước các điểm dữ liệu thuộc nhát cắt có thể đồng thời tính là thuộc hai hình hộp

End

$$\text{Hình 8.5. Thủ tục chia đôi hình hộp } n\text{-chiều } B_j = \prod_{i=1}^n [a_i^j, b_i^j]$$

Thủ tục chia đôi kết thúc khi số lượng đối tượng trong mỗi hình hộp con không vượt quá M chọn trước. Như vậy số mốc nội suy trong mỗi hình hộp con ở độ sâu d được xem là bằng nhau và xấp xỉ bằng $\frac{N}{2^d}$ do có những mốc có thể được tính vào nhiều hình hộp. Khi thủ tục kết thúc thì số lượng dữ liệu nằm trong mỗi miền con thuộc khoảng $\left[\frac{M}{2}, M\right]$ và có khoảng từ $\frac{N}{M}$ đến $2\frac{N}{M}$ miền con. Ngoài ra, có thể chọn trước số miền con làm điều kiện dừng chia.

Hình 8.6 mô tả kết quả chia hình hộp 2 chiều $[0,10] \times [0,6]$ chứa 38 dữ liệu ($N=38$) với $M=10$. Kết quả có 4 hình hộp con đều chứa 10 dữ liệu, trong đó các điểm $(1,4)$ và $(8,2)$ được tính đồng thời ở hai hình hộp chứa nó còn điểm $(3,1)$ chỉ thuộc B_3 .



Hình 8.6. Cây k-d mô tả tập dữ liệu trong không gian 2 chiều, với $N=38$, $M=10$.

Với cấu trúc cây nhị phân này, khi có điểm dữ liệu x mới, ta dễ dàng xác định được nó thuộc hình hộp con nào ở lá bằng cách tìm kiếm từ gốc đến nhánh so sánh giá trị thuộc tính ở cạnh cắt của hình hộp tương ứng với giá trị ở điểm cắt.

Chú ý. Việc chọn cạnh $[a_i^j, b_i^j]$ trong bước 1 của thủ tục chia đôi hình hộp B_j có thể thay bằng cách chọn tuân tự theo các biến như thủ tục cây k-d thông thường.

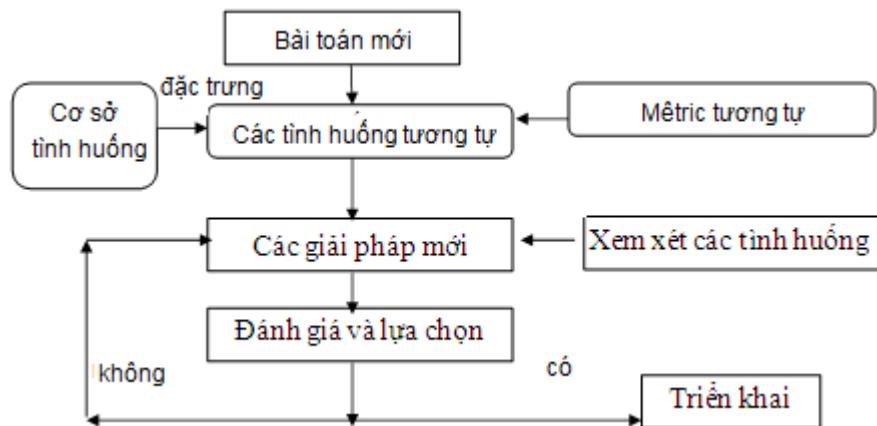
Trên đây, giới thiệu các phương pháp địa phương giải bài toán hồi quy. Phương pháp dựa vào tiếp cận *trợ quyết định* để tìm lời giải cho các bài toán có cấu trúc xâu, không thể giải bằng thuật toán chặt chẽ được.

8.5. LẬP LUẬN DỰA TRÊN TÌNH HUỐNG (CBR)

Các phương pháp k -NN và hồi quy trọng số địa phương có chung 3 đặc tính chính.

- 1) Chúng là các phương pháp học lười, không đưa ra hàm số cụ thể để khái quát hóa dữ liệu huấn luyện mà chỉ thực hiện thủ tục xác định giá trị hàm khi có mẫu truy vấn mới.
- 2) Giá trị hàm tại các mẫu truy vấn mới được xác định bằng việc phân tích các mẫu *gần/tương tự* nó nhất mà bỏ qua các mẫu khác xa với mẫu đang xét.
- 3) Các mẫu được đặc trưng bởi các điểm trong một không gian Euclidean n -chiều.

Lập luận dựa trên tình huống (Case based reasoning: CBR) là một kiểu học dựa trên 2 nguyên tắc đầu tiên nhưng không dựa trên nguyên tắc thứ ba. Trong phương pháp CBR, các mẫu có thể được đặc trưng bởi các thuộc tính có giá trị định danh và dùng metric thích hợp (chẳng hạn các metric đã được giới thiệu trong mục 4.2.4.1) để xác định tính tương tự giữa các tình huống cần xử lý. Thoạt tiên, CBR được dùng trong các hệ hỗ trợ quyết định, trong đó mỗi khi có một tình huống mới cần tìm giải pháp, người ta tìm các tình huống tương tự với nó trong dữ liệu lưu trữ và tham khảo giải pháp đã có để xây dựng lời giải mới. Lược đồ chung của phương pháp CBR được mô tả trong hình 8.7.



Hình 8.7. Sơ đồ cấu trúc của hệ CBR

CBR đã được áp dụng thiết kế các *thiết bị/máy móc* dựa trên sử dụng thư viện lưu trữ các thiết kế trước đó, hoặc suy luận về các trường hợp hợp lệ mới dựa trên các quyết định trước đó. CBR cũng được dùng để giải quyết các bài toán lập kế hoạch và lập lịch bằng việc sử dụng lại và kết hợp các phần lời giải trước cho các bài toán tương tự. Ngày nay, CBR còn được áp dụng cho các hệ suy luận để xây dựng các hệ thiết kế để xây dựng các hệ như thiết kế các hệ điều khiển tự động.

Ví dụ. Để dự báo các hiện tượng thời tiết bất thường có thể gặp khi bay dựa trên các số liệu từ các trạm quan trắc, người ta so sánh các số liệu này với các số liệu của các ngày lân cận trong những năm trước đó để tìm ra những ngày tương tự nhất với nó và tham khảo các hiện tượng đã có. Các hiện tượng này được thông báo cho các phi công để chuẩn bị trước giải pháp xử lý mà không bị bất ngờ khi gặp phải nó khi bay. Hệ thống CADET (Sycara et al. 1992) sử dụng suy luận dựa trên tình huống để giúp thiết kế hệ điều khiển vòi nước.

Tóm lại, CBR là một phương pháp học dựa trên mẫu, ở đó các mẫu (các tình huống) có thể được đặc tả bằng nhiều quan hệ, khi gặp tình huống mới, người ta kết hợp suy luận dựa trên cơ sở tri thức và các giải pháp đã áp dụng trong các tình huống tương tự để xây dựng lời giải. Một vấn đề cần nghiên cứu hiện nay trong CBR là phát triển các phương pháp thích hợp để cải tiến việc lưu trữ và truy vấn các tình huống.

KẾT LUẬN

Trong chương này ta đã xét ba phương pháp học lười hay học dựa trên mẫu: hồi quy k-NN, hồi quy trọng số địa phương và CBR. Các mạng nơron RBF được xem là cầu nối giữa mạng MLP và các phương pháp hồi quy dựa trên mẫu. Những phương pháp học địa phương có các điểm chính sau.

- Các phương pháp học dựa trên mẫu khác với các cách tiếp cận phép xấp xỉ hàm khác ở chỗ chúng chỉ xử lý các mẫu huấn luyện khi chúng phải gán nhãn cho một mẫu truy vấn mới. Kết quả là chúng không xây dựng giả thuyết tường minh cho hàm mục tiêu trên toàn bộ không gian mẫu, độc lập với mẫu truy vấn đó. Thay vào đó, chúng chỉ tạo ra một phép xấp xỉ địa phương cho hàm mục tiêu đối với mỗi mẫu truy vấn.
- Ưu điểm của các phương pháp dựa trên mẫu là cho phép tìm nhãn của mẫu mới khi hàm mục tiêu phức tạp nhò dùng các phép xấp xỉ địa phương đơn giản hơn dựa trên thông tin trong tập mẫu huấn được lưu trữ. Các khó khăn chính trong cách tiếp cận này là: 1) Xác định một độ đo khoảng cách phù hợp để tìm ra các mẫu liên quan (đặc biệt, khi các mẫu được miêu tả bởi các mô tả tượng

trung phức tạp); 2) Xử lý các ảnh hưởng bất lợi của các đặc trưng ít liên quan đối với độ đo khoảng cách đã có.

- k-NN là một thuật toán dựa trên mẫu dùng cho việc xấp xỉ các hàm mục tiêu nhận giá trị *thực/rồi rạc* và các mẫu được tương ứng với các điểm trong không gian Euclide n -chiều. Giá trị hàm mục tiêu cho một mẫu truy vấn mới được ước lượng từ các giá trị đã biết của k mẫu huấn luyện gần nhất.
- Các phương pháp hồi quy trọng số địa phương là tổng quát hóa của k -NN, trong đó, khi cần gán nhãn cho mẫu mới, người ta xây dựng một phép xấp xỉ địa phương hiển của hàm mục tiêu cho mẫu này. Phép xấp xỉ địa phương cho hàm mục tiêu có thể thực hiện nhờ dùng nhiều dạng hàm như các hàm hằng số, hàm tuyến tính hoặc hàm bậc hai hay dựa vào các hàm nhân có ảnh hưởng địa phương trong không gian mẫu.
- Các mạng nơron RBF là một loại mạng nơron nhân tạo được xây dựng từ các hàm nhân ảnh hưởng địa phương. Các phương pháp này có thể được xem như là tổng hợp các hướng tiếp cận dựa trên mẫu (ảnh hưởng được khoanh vùng theo không gian của mỗi hàm nhân) và các hướng tiếp cận mạng nơron. Mạng nơron RBF địa phương cho phép xây dựng hàm nội suy với tập mẫu lớn và thường có bổ sung các mẫu quan sát mới để huấn luyện lại mạng. Mạng nơron RBF xấp xỉ dùng phương pháp hồi quy trọng số địa phương để xây dựng một tập mẫu có mốc cách đều rồi dùng nó để xây dựng mạng RBF bằng phương pháp lắp.
- Lập luận dựa trên tình huống là một cách tiếp cận dựa trên mẫu, trong đó các mẫu được miêu tả bởi các đặc tả logic phức tạp hơn các điểm trong không gian Euclidean. Dựa vào các mô tả phức tạp của các mẫu đã cho, các phương pháp đa dạng được đề xuất để xây dựng hàm đích từ cho các mẫu mới, chúng được dùng để tham khảo cho xác định lời giải cuối cùng. Các phương pháp lập luận dựa trên tình huống đã và đang được dùng trong nhiều ứng dụng như tạo mô hình lập luận hợp lý và hướng dẫn tìm kiếm trong các bài toán thiết kế, lập kế hoạch phức tạp.

Một nhược điểm của các hướng tiếp cận dựa trên mẫu là khi phải tìm nhãn cho nhiều mẫu mới thì có thể mất nhiều thời gian, hạn chế phạm vi áp dụng. Nguyên nhân của hạn chế này là do hầu hết các tính toán được thực hiện ở lúc tìm nhãn cho các mẫu kể từ khi bắt gặp mẫu mới đầu tiên mà không thể thực hiện tìm hàm đích trước. Bởi vậy, các kỹ thuật sắp xếp các mẫu huấn luyện thuận tiện cho quá trình tính toán là một bài toán thực tế rất ý nghĩa để giảm thời gian truy vấn.

Nhược điểm thứ hai của các hướng tiếp cận dựa trên mẫu (đặc biệt là cách tiếp cận láng giềng gần nhất) là chúng dùng metric dựa trên tất cả các thuộc tính của

không gian mẫu để tìm ra các mẫu huấn luyện tương tự được lưu trữ trong bộ nhớ. Khi khái niệm mục tiêu chỉ dựa trên một vài trong nhiều thuộc tính của mẫu thì các mẫu thật sự tương tự với mẫu mới có thể không phải là các mẫu gần nhau theo khoảng cách được xét.

BÀI TẬP

1. Xây dựng ví dụ để so sánh các phương pháp K-NN, hồi quy trọng số địa cho hàm $f = \sin(2\pi x_1) + \cos(2\pi x_2)$, dựa trên tập mẫu với x có phân bố đều trên miền $[0,1] \times [0,1]$ gồm 40 mẫu đào tạo và 10 mẫu kiểm tra.
2. Tạo ra tập mẫu quan sát được gồm 100 mẫu với nhiễu của hàm số có phân bố chuẩn $N(0,0.2)$ cho hàm $f = 3\sin(\pi x_1 x_2) + 2\sin(\pi x_1 + x_2)$, trong đó x có phân bố đều trên miền $[0,1] \times [0,1]$.
 - a) Dùng 90 mẫu để xây dựng mạng nơron RBF hồi quy cho hàm này.
 - b) Cũng dùng dữ liệu huấn luyện ở câu a) để ước lượng giá trị hàm f tại 10 mẫu còn lại và so sánh chất lượng xấp xỉ với phương pháp trên.
3. Chỉ ra cách dùng phương pháp k -NN cho hàm số mà tập mẫu có 5 thuộc tính định danh và 2 thuộc tính thực.

Chương 9

HỌC TĂNG CƯỜNG

Khác với các phương pháp học trước đây, trong học tăng cường bộ học là tác tử (agent) ra quyết định (decision-making), nó thực hiện các tác động trong môi trường và nhận các khoản thưởng (hoặc phạt) cho các tác động của mình trong quá trình giải quyết bài toán. Sau các thực nghiệm kiểu *thử-sai*, nó học được chính sách tốt nhất, tức là chuỗi tác động tốt nhất để tối ưu hóa giá trị tích lũy thưởng.

9.1. TÁC TỬ VÀ CÁC BÀI TOÁN HỌC

Trong chương 1, ta đã làm quen với ví dụ về robot tìm chu trình Hamilton tối ưu, trước khi mô tả tác tử và phát biểu các bài toán học cho nó, ta xét thêm các ví dụ về robot tìm đường đi và hệ chơi cờ.

9.1.1. Một số ví dụ

Ví dụ 1: *Tìm đường đi của robot*

Xét một robot có trang bị cảm biến (sensor) để quan sát trạng thái của môi trường và chọn đường đi. Tại thời điểm ban đầu, nó ở vị trí A và muốn tới điểm B theo con đường ngắn nhất. Mỗi ngã rẽ, điểm A hoặc điểm đích B được xem là một trạng thái của môi trường mà robot dùng làm mốc. Như vậy robot xuất phát ở trạng thái $s_0 = A$ và ở mỗi trạng thái s_i nó chọn đường đi như là tác động a tương ứng sao cho đến được trạng thái đích $s_{k+1} = B$, trong đó k là số lần chuyển trạng thái chưa xác định trước mà tùy theo đường đi được chọn. Ký hiệu độ dài quãng đường (chi phí) từ s_i tới s_{i+1} là d_i , khi đó độ dài đường đi (chi phí) tích hợp là:

$$C = \sum_{i=0}^k d_i. \quad (9.1a)$$

Robot cần tìm một chính sách chọn đường đi ở mỗi chẽ rẽ sao cho C nhỏ nhất. Nếu thay $r_i = m - d_i$ như là lượng thưởng (m là hằng số chọn trước tùy ý) thì ta có bài toán tương đương là tìm cực đại hàm giá trị tích lũy thưởng V :

$$V = \sum_{i=0}^k r_i. \quad (9.1b)$$

Trong trường hợp này, lượng thưởng r_i (chi phí d_i) là đơn định đối với mỗi trạng thái s_i và tác động a_i . Để đơn giản về sau ta chỉ xét bài toán với các tác động có thưởng và tìm cực đại giá trị tích lũy thưởng.

Nếu ta xét mục đích của robot là đi sạc pin khi được báo sạc cạn thì B là điểm sạc pin và robot cần phải đến trước khi pin cạn. Giả sử thời gian còn lại cho robot di chuyển là T nhưng robot không đo được lượng pin tiêu hao trên quãng đường. Khi đó nếu robot đến được đích thì mọi tác động sẽ được thưởng cho giá trị là 100, ngược lại lượng thưởng bằng không. Lúc này lượng thưởng cho mỗi tác động ở trạng thái s_i không được biết ngay mà phải chờ đến khi kết thúc nó tìm được B hay nằm giữa đường mới xác định được. Hơn nữa mỗi đoạn đường s_i tới s_{i+1} có thể thuộc đường đến đích hoặc không nên lượng thưởng cho tác động chọn nó nhận giá trị ngẫu nhiên trong tập $\{0,100\}$. Về sau sẽ giả thiết rằng ước lượng được các xác suất có điều kiện $P(r_{i+1}/s_i, a_i)$ cho mọi cặp trạng thái s_i và tác động a_i nhờ thống kê. Khi đó ta cần cực đại kỳ vọng của hàm tích lũy:

$$E(V) = E(\sum_{i=0}^k r_i). \quad (9.1c)$$

Trong hai trường hợp trên, robot có trạng thái đích B gọi là *trạng thái hút* (asorbing state)/ *kết thúc* và nó dừng làm việc ở đó sau một số lần chuyển trạng thái. Nếu robot điều khiển xe tải đi nhận hàng, phân phối và nhận luân phiên không nghỉ, còn các giá trị thưởng là đơn định thì hàm tích lũy ở (9.1b) được thay bởi:

$$V = \sum_{i=0}^{\infty} \gamma^i r_i, \quad (9.1d)$$

trong đó $\gamma \in [0,1]$ là hằng số chiết khấu. Khi $\gamma < 1$ có nghĩa rằng các khoản thưởng muộn càng ít ý nghĩa, khi $\gamma = 0$ thì ta chỉ quan tâm tới khoản thưởng ban đầu, còn khi lấy $\gamma = 1$ thì các giá trị thưởng ở mọi thời điểm đều quan trọng như nhau. Bạn đọc có thể mở rộng (9.1.c) cho trường hợp ngẫu nhiên.

Ví dụ 2: Tác tử chơi cờ

Bây giờ ta xét tác tử là chương trình chơi cờ ở bên A và có đối thủ bên B (có thể là một chương trình khác). Mỗi trạng thái bàn cờ mà A chọn nước đi (tác động) sẽ là một trạng thái. Nhiệm vụ của tác tử là chọn nước đi (tác động) để bên A thắng. Nếu bên A thắng thì mỗi nước đi được thưởng giá trị 100, thua thì lượng thưởng là -100, còn hòa thì giá trị thưởng bằng không. Trong trường hợp này không chỉ lượng thưởng có tính ngẫu nhiên, không biết trước mà trạng thái bàn cờ sau mỗi nước đi cũng có tính ngẫu nhiên vì còn tùy thuộc vào nước đi của bên B. Ta giả thiết biết được các xác suất chuyển trạng thái $P(s_{i+1}/s_i, a_i)$ cho mọi cặp trạng thái s_i và tác động a_i . Trường hợp hòa thì số lần chuyển trạng thái có thể xem là vô hạn. Tương tự công thức (9.1c), hàm thưởng tích lũy có dạng tổng quát là:

$$V = E(\sum_{i=0}^{\infty} \gamma^i r_i), \quad (9.1e)$$

Bây giờ ta có thể giới thiệu mô hình cho tác tử và một số khái niệm liên quan.

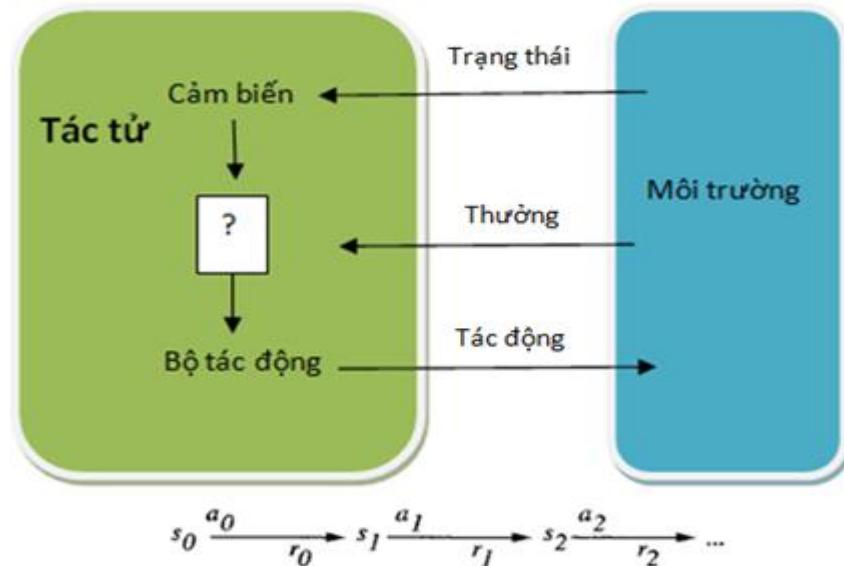
9.1.2. Tác tử

Không chú ý tới cấu trúc vật lý của phần cứng, tác tử là một bộ ra quyết định được đặt trong một môi trường. Trong robot tìm đường đi, tác tử là robot và môi trường là hệ thống đường đi bao gồm các vị trí liên quan, còn trong chơi cờ thì tác tử là bộ chơi cờ còn môi trường là bàn cờ.

Môi trường được đặc trưng bởi một tập trạng thái $S = \{s^i\}_{i \in I}$, trong chuỗi thời gian $t = 1, 2, \dots$ robot có thể quan sát được trạng thái tương ứng của môi trường. Trong các ví dụ trên, trạng thái của môi trường là vị trí của robot hoặc thế cờ tại thời điểm đang xét.

Ở mỗi trạng thái $s \in S$, có một tập tác động $\mathcal{A}(s)$ mà tác tử có thể chọn. Nếu thời điểm t tác tử ở trạng thái s_t và thực hiện tác động $a_t \in \mathcal{A}(s_t)$ thì nó chuyển sang trạng thái $s_{t+1} = \delta(s_t, a_t)$ và nhận giá trị thưởng $tích thời r_t = r(s_t, a_t)$, trong đó δ và r tương ứng là các hàm chuyển trạng thái và hàm giá trị thưởng xác định trên $\cup_{s \in S} \{s\} \times \mathcal{A}(s)$ đơn định hoặc ngẫu nhiên.

Tác tử sẽ bắt đầu ở trạng thái s_0 và tuân tự ở các trạng thái thực hiện chuỗi tác động $\{a_t : t = 0, 1, 2, \dots\}$, tương ứng chuyển sang các trạng thái $\{s_{t+1} : t = 0, 1, 2, \dots\}$ và nhận các giá trị thưởng $\{r_t : t = 0, 1, 2, \dots\}$ hữu hạn hay vô hạn tùy theo bài toán. Mặc dù ta gọi các giá trị thưởng r_t là *thưởng tích thời* nhưng có thể nó chưa xác định được ngay sau khi thực hiện tác động như trong ví dụ chơi cờ hay sạc pin. Mô hình một tác tử được mô tả trong hình 9.1.



Hình 9.1. Một tác tử tương tác với môi trường: bắt đầu ở trạng thái s_0 , thực hiện chuỗi tác động, chuyển trạng thái và nhận thưởng

Các quá trình chuyển trạng thái và nhận thưởng này là quá trình Markov, trong đó kết quả ở thời điểm t chỉ phụ thuộc vào trạng thái và tác động hiện thời mà không phụ thuộc vào các trạng thái và tác động ở những thời điểm trước đó

Ký hiệu $\mathcal{A} = \cup_{s \in S} \mathcal{A}(s)$ và $\pi: S \rightarrow \mathcal{A}$ là hàm xác định *chính sách* (policy) chọn tác động $a = \pi(s)$ của tác tử khi ở trạng thái s . Mục đích của bộ học là tìm ra chính sách chọn tác động tối ưu π^* sao cho nếu xem tác tử khởi đầu từ s_t và chọn tác động theo chính sách này thì giá trị tích lũy thưởng đạt cực đại với mọi $s_t \in S$.

Khác với các bài toán học có giám sát trước đây, tác tử sẽ thực hiện các thực nghiệm hoặc thí nghiệm mô phỏng để dựa vào đó mà tìm chính sách tối ưu. Quá trình học tăng cường thường có các đặc điểm sau.

Thưởng trễ (delayed reward). Tác tử cần học một hàm mục tiêu π ánh xạ từ trạng thái hiện thời s vào tác động tối ưu $a = \pi(s)$. Theo cách tiếp cận trong học có giám sát trước đây thì khi học hàm π ta thường giả thiết mỗi mẫu đào tạo là cặp $(s, \pi(s))$. Tuy nhiên, trong học tăng cường thì thông tin như thế không có sẵn mà thay vào đó bộ học chỉ nhận được một chuỗi giá trị thưởng tức thời tương ứng với chuỗi tác động khi tác tử thực hiện. Thậm chí các khoản thưởng này chỉ xác định được khi kết thúc thử nghiệm như trong trường hợp chơi cờ hoặc đi sạc pin. Trong các chương trước, chúng ta học hàm mục tiêu π dựa trên các dữ liệu quan sát được, trong đó mỗi ví dụ huấn luyện là một cặp mẫu $(s, \pi(s))$.

Khám phá (exploration). Trong học tăng cường, tác tử ảnh hưởng tới phân bố của các ví dụ huấn luyện bằng dãy tác động mà nó chọn. Điều này đặt ra một câu hỏi là chính sách thí nghiệm nào đem lại nhiều hiệu quả nhất. Bộ học cần sự cân bằng giữa khám phá các trạng thái và tác động chưa biết hay khai thác các trạng thái và tác động mà nó đã được học và sẽ đem lại giá trị thưởng cao hơn.

Các trạng thái quan sát được một phần. Mặc dù có thể cho rằng các bộ cảm biến của tác tử có thể nhận biết toàn bộ trạng thái của môi trường trong mỗi thời điểm, nhưng trong nhiều trường hợp thực tế, các bộ cảm biến chỉ cung cấp các thông tin cục bộ. Chẳng hạn robot với camera phía trước không thể thấy được những gì ở sau nó hoặc bị khuất khi có chướng ngại vật. Trong các trường hợp như vậy, tác tử cần phải kết hợp với những quan sát trước đó với những dữ liệu hiện thời khi lựa chọn tác động.

Học cả đời. Không giống với việc xấp xỉ hàm riêng lẻ, tác tử thường phải học nhiều việc trong cùng một môi trường, sử dụng cùng một cảm biến. Chẳng hạn, một robot di động cần phải học làm thế nào để có thể tự nạp điện, làm thế nào để đi qua các hành lang hẹp, làm thế nào để lấy kết quả từ máy in laser. Điều này đặt ra khả năng sử dụng những kinh nghiệm và hiểu biết từ trước để giảm bớt sự phức tạp mẫu khi học các nhiệm vụ mới.

9.1.3. Các bài toán học

Trong phần này chúng ta thiết lập bài toán tìm *chính sách/chiến lược điều khiển* tối ưu cho bài toán học tăng cường một cách rõ ràng hơn dưới các dạng: đơn định và ngẫu nhiên. Các bài toán được thiết lập tổng quát dựa trên quá trình quyết định Markov (Markov decision process: MDP).

Bài toán đơn định

Xét một thế giới gồm tác tử trong một môi trường có tập hữu hạn trạng thái S , tập các tác động có thể của tác tử $\mathcal{A} = \cup_{s \in S} \mathcal{A}(s)$, trong đó các giá trị thưởng tức thời $r_t = r(s_t, a_t)$ và trạng thái chuyển $s_{t+1} = \delta(s_t, a_t)$ là đơn định với mọi cặp (s_t, a_t) . Giả sử tại thời điểm t tác tử ở trạng thái s_t và thực hiện chiến lược điều khiển $\pi: S \rightarrow \mathcal{A}$ thì thu được giá trị tích lũy thưởng:

$$V^\pi(s_t) = \sum_{i=0}^{\infty} \gamma^i r_{t+i}, \quad (9.2a)$$

trong đó $\gamma \in [0, 1]$ là giá trị chiết khấu. Đại lượng $V^\pi(s_t)$ được gọi là *giá trị tích lũy thưởng* đạt được từ chiến lược điều khiển/chính sách π và trạng thái ban đầu s_t . Lưu ý rằng để (9.2a) có nghĩa trong trường hợp có trạng thái hút s_f , ta quy ước ở trạng thái này chỉ có tác động giữ nguyên trạng thái và giá trị thưởng tức thời bằng không (hoặc bằng m nếu là chuyển từ bài toán cực tiểu chi phí với hằng số m). Công thức (9.2a) cũng trùng với (9.1d) khi xem s_t là trạng thái khởi đầu s_0 .

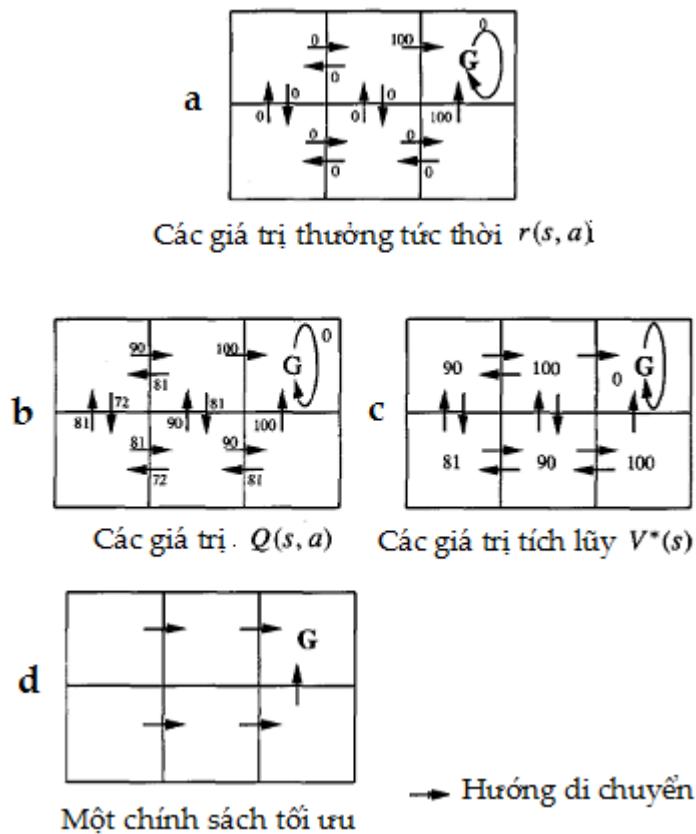
Nhiệm vụ của bài toán học tăng cường đơn định là tìm *chiến lược điều khiển* (chính sách) tối ưu $\pi^*: S \rightarrow \mathcal{A}$ sao cho với mọi trạng thái $s_t \in S$ ở thời điểm t và thực hiện tác động theo chính sách này thì ta đều thu được giá trị tích lũy thưởng cực đại, tức là:

$$\pi^* = \arg \max_\pi V^\pi(s_t), \forall s_t \quad (9.2b)$$

Để đơn giản, ta sẽ dùng ký hiệu $V^*(s)$ thay cho ký hiệu giá trị tích lũy thưởng $V^{\pi^*}(s)$ của chính sách tối ưu. $V^*(s)$ cho giá trị thưởng tích lũy có chiết khấu cực đại mà tác tử tích lũy được khi khởi đầu ở trạng thái s .

Hình 9.2a mô tả thế giới có một môi trường chia ô đơn giản. Sáu ô lưỡi trong biểu đồ mô tả sáu trạng thái hay vị trí của tác tử, ô có nhãn G là trạng thái hút. Mỗi mũi tên trong biểu đồ biểu thị một tác động mà tác tử có thể thực hiện để di chuyển từ trạng thái này sang trạng thái khác. Các tác động có thể là sang trái, sang phải và lên, xuống trong phạm vi của sáu ô này. Chữ số kết hợp với mỗi mũi tên mô tả giá trị thưởng tức thời $r(s, a)$ tác tử nhận được khi nó thực hiện các tác động tương ứng. Lưu ý rằng giá trị thưởng tức thời trong môi trường đặc biệt này được đặt bằng 0 đối với mọi tác động chuyển trạng thái, ngoại trừ những trạng thái được gán nhãn G. Để dễ dàng thấy rằng trong trường hợp này, trạng thái G là trạng thái đích bởi vì đó là cách

duy nhất tác tử có thể nhận được giá trị thưởng là chuyển đến trạng thái đó. Các giá trị tích lũy thường ứng với giá trị chiết khấu $\gamma = 0,9$ được cho trong hình 9.2c.



Hình 9.2. Một thế giới đơn định đơn giản. Mỗi ô biểu thị một trạng thái, giá trị thưởng bằng 100 nếu chuyển vào trạng thái G và bằng 0 trong các tường hợp khác. Các giá trị $V^*(s)$ và $Q(s, a)$ với chiết khấu $\gamma = 0,9$, chỉ ra một chính sách (chiến lược điều khiển) tối ưu

Một chính sách tối ưu cho bài toán này được chỉ ra trong hình 9.2d, ở đó chỉ rõ một tác động mà tác tử sẽ lựa chọn trong mỗi trạng thái môi trường. Chính sách tối ưu hướng tác tử theo con đường ngắn nhất để đạt đến trạng thái G.

Chú ý rằng V^* được định nghĩa là tổng của các giá trị thưởng có chiết khấu trong tương lai vô hạn. Trong môi trường đặc biệt này, chỉ một lần tác tử chuyển tới trạng thái hút G, tương lai vô hạn của nó chính là duy trì trong trạng thái này và nhận các giá trị thưởng bằng không.

Bài toán ngẫu nhiên

Trong bài toán ngẫu nhiên, với mỗi trạng thái $s_t \in S$ và tác động $a \in \mathcal{A}(s_t)$, giá trị thưởng $r(s_t, a)$ và trạng thái kế tiếp $s_{t+1} = \delta(s_t, a)$ không đơn định mà có tính ngẫu nhiên. Phân bố xác suất của chúng có thể đã biết hoặc sẽ được xác định qua thực nghiệm. Tương tự như trường hợp đơn định, với mỗi chiến lược điều khiển $\pi: S \rightarrow \mathcal{A}$,

hàm mục tiêu lúc này là kỳ vọng của giá trị thưởng tích lũy như trong (9.1f) với trạng thái ban đầu s_t :

$$V^\pi(s_t) = E(\sum_{i=0}^{\infty} \gamma^i r_{t+i}). \quad (9.3a)$$

Như vậy, yêu cầu của bài toán học tăng cường là tìm chiến lược điều khiển tối ưu $\pi^*: S \rightarrow \mathcal{A}$ sao cho:

$$\pi^* = \arg \max_\pi V^\pi(s_t) \quad (9.3b)$$

với mọi trạng thái $s_t \in S$ ở thời điểm t .

Chú ý. Trong các bài toán học tăng cường, các tác động và hàm chuyển trạng thái là yếu tố nội tại quyết định bản chất bài toán, còn giá trị thưởng tức thời có vai trò là yếu tố bình luận, đánh giá tác động và bộ học có thể xây dựng hàm thưởng phù hợp với mục đích hoạt động của tác tử. Chẳng hạn, giá trị thưởng tức thời và do đó giá trị tích lũy thường ở cả hai bài toán ở mục 9.1.1 đều do bộ học quyết định để đạt mục đích tìm đường đi tối ưu hay chiến lược cho bộ chơi cờ thắng.

9.2. HỌC Q (Q learning)

9.2.1. Học Q trong các bài toán đơn định

Trong học tăng cường, chúng ta không có được các mẫu dạng (s, a) nên rất khó học trực tiếp chiến lược điều khiển tối ưu $\pi^*: S \rightarrow \mathcal{A}$. Thay vào đó, bộ học chỉ dựa vào thông tin từ các chuỗi giá trị thưởng tức thời $r(s_i, a_i)$ trên thực nghiệm.

Giả sử đã tìm được chiến lược điều khiển tối ưu $\pi^*: S \rightarrow \mathcal{A}$. Khi đó với mỗi trạng thái ban đầu $s_t \in S$, ta nhận được chuỗi tác động tối ưu và giá trị tích lũy cực đại $V^*(s_t)$ theo công thức (9.2a). Biểu thức tính $V^*(s_t)$ được biểu diễn lại là:

$$\begin{aligned} V^*(s_t) &= \max_{a_t} \{r(s_t, a_t) + \gamma \sum_{i=0}^{\infty} \gamma^i s_{t+1+i}\} \\ &= \max_{a_t} \{r(s_t, a_t) + \gamma V^*(\delta(s_t, a_t))\}. \end{aligned} \quad (9.3c)$$

Từ đó ta có:

$$\pi^*(s) = \arg \max_a \{r(s, a) + \gamma V^*(\delta(s, a))\}. \quad (9.3d)$$

Như vậy, tác động tối ưu ở trạng thái s là tác động a làm cực đại tổng giá trị thưởng tức thời $r(s, a)$ với giá trị V^* của trạng thái kế tiếp có chiết khấu γ . Tác tử có thể học được chính sách tối ưu thông qua học V^* nhờ được cho toàn bộ thông tin về giá trị hàm thưởng r và hàm chuyển trạng thái δ . Tuy nhiên trong nhiều bài toán thực tế, các hàm r và δ của môi trường chưa biết trước mà cần xác định qua thực nghiệm. Thay cho V^* ta sẽ dùng một hàm có thể ước lượng để xác định π^* .

9.2.1.1. Hàm Q

Chúng ta định nghĩa hàm $Q(s,a)$ là giá trị thưởng tức thời nhận được ở trạng thái ban đầu s với tác động a cộng với giá trị tích lũy thưởng tối ưu sau đó (có chiết khấu γ):

$$Q(s,a) \equiv r(s,a) + \gamma V^*(\delta(s,a)) \quad (9.4)$$

Lưu ý rằng $Q(s,a)$ là lượng đúng được cung cấp bởi biểu thức trong (9.3d) để chọn tác động tối ưu a ở trạng thái s . Do đó ta có thể viết lại (9.3d) theo $Q(s,a)$ như sau:

$$\pi^*(s) = \arg \max_a Q(s,a) \quad (9.5)$$

Việc viết lại này rất quan trọng, vì nó chỉ ra rằng nếu tác tử học theo hàm Q thay vì V^* thì nó có thể chọn được các tác động tối ưu kể cả khi nó không biết hàm r và δ . Công thức (9.5) chỉ ra rằng bộ học chỉ cần quan tâm tới mỗi tác động a ứng với trạng thái hiện tại s và lựa chọn tác động làm $Q(s,a)$ đạt cực đại.

Về sau ta sẽ thấy tác tử có thể chọn một chuỗi tác động tối ưu toàn cục bằng cách tác động lặp lại các giá trị cục bộ của Q đối với trạng thái hiện thời. Có nghĩa là tác tử có thể chọn các tác động tối ưu mà không phải quan tâm tới các trạng thái dẫn đến kết quả gì từ tác động.

Hình 9.2b chỉ ra giá trị $Q(s,a)$ đối với mỗi trạng thái s và tác động a trong một thế giới lưới đơn giản, các giá trị $V^*(s)$ được cho ở hình 9.2c, còn một chiến lược tối ưu được cho ở hình 9.2d.

9.2.1.2. Một thuật toán học Q

Công thức (9.5) ở trên cho ta thấy việc học hàm Q sẽ cho phép học được chiến lược tối ưu π^* . Vậy làm thế nào để học hàm Q ? Watkins (1989) đề xuất thuật toán lặp để ước lượng gần đúng Q . Để hiểu rõ thuật toán, ta chú ý tới mối quan hệ gần gũi giữa Q và V^* :

$$V^*(s) = \max_{a'} Q(s,a').$$

Quan hệ này cho phép viết lại công thức (9.4) như sau :

$$Q(s,a) = r(s,a) + \gamma \max_{a'} Q(\delta(s,a), a') \quad (9.6)$$

Cách xác định Q theo đê quy này là cơ sở của thuật toán xấp xỉ lặp Q . Để mô tả thuật toán, ta dùng ký hiệu \hat{Q} để để biểu diễn ước lượng hiện thời cho hàm Q trong quá trình học. Trong thuật toán này, bộ học đưa ra giả thuyết \hat{Q} của nó bằng một bảng lớn với mỗi phần tử biểu thị cho một cặp trạng thái-tác động. Mỗi phần tử ứng với mỗi cặp (s,a) của bảng lưu trữ giá trị $\hat{Q}(s,a)$ của giả thuyết hiện thời cho giá

trí hàm học chưa biết $Q(s,a)$. Bảng này có thể được khởi tạo ngẫu nhiên, dễ hiểu nhất là khởi tạo bằng 0 cho mọi cặp trạng thái-tác động. Các bước trong thuật toán sẽ được mô tả cho ba trường hợp: 1) tổng quát, 2) môi trường có trạng thái hút và mọi chuỗi tác động đều dẫn tới trạng thái này, 3) tồn tại một chuỗi trạng thái-tác động vô hạn sao cho mỗi trạng thái $s \in S$ và tác động $a \in \mathcal{A}$ được lặp lại vô hạn lần khi độ dài chuỗi ra vô hạn.

Thuật toán cho trường hợp tổng quát

Bộ học xác định một phương thức lấy ngẫu nhiên các trạng thái $s \in S$ và $a \in \mathcal{A}$ sao cho mỗi cặp (s, a) tùy ý đều có xác suất được chọn khác không. Trong trường hợp đó, bộ học thực hiện lặp việc chọn ngẫu nhiên cặp tác động (s, a) và quan sát giá trị kết quả thưởng $r = (s, a)$. Sau đó cập nhật lại bảng lưu trữ của theo quy tắc:

$$\hat{Q}(s, a) \leftarrow r + \gamma \max_{a'} \hat{Q}(s', a') \quad (9.7)$$

Chú ý rằng quy tắc huấn luyện này sử dụng các giá trị \hat{Q} hiện thời của tác tử đổi với trạng thái mới s' để cải thiện ước lượng của $\hat{Q}(s, a)$ đổi với trạng thái s trước đó. Trong thuật toán này, mặc dù biểu thức tính $Q(s, a)$ (9.6) mô tả Q qua các hàm $\delta(s, a)$ và $r(s, a)$, nhưng bộ học không cần thiết phải biết các hàm tổng quát này khi áp dụng luật huấn luyện theo công thức (9.7). Thay vào đó, nó thực hiện tác động trong môi trường rồi quan sát trạng thái tiếp theo s' và giá trị thưởng. Vì thế, có thể xem là thực hiện thủ tục lấy mẫu cho các hàm này ở những giá trị hiện thời của s và a .

Thuật toán học Q trên đây đối với các quá trình quyết định Markov đơn định được miêu tả chính xác hơn ở bảng 9.1.

Bảng 9.1. Thuật toán học Q đơn định với hệ số chiết khấu γ

Bước 1. Khởi tạo $\hat{Q}(s, a)$ bằng 0 cho mỗi cặp s, a

Bước 2. Thực hiện lặp:

- 2.1. Lấy ngẫu nhiên $s \in S$;
- 2.2. Lấy ngẫu nhiên $a \in \mathcal{A}(s)$
- 2.3. Xác định $r(s, a)$ và trạng thái kế tiếp s' ;
- 2.4. Cập nhật bảng $\hat{Q}(s, a)$ theo quy tắc (9.7)

Điều kiện dừng.

Về lý thuyết, với một số điều kiện sẽ nói về sau, khi bước 2 thực hiện vô hạn lần thì $\hat{Q}(s, a)$ hội tụ đến $Q(s, a)$. Tuy nhiên, trong thực hành thì thuật toán chỉ thực hiện

được một số hữu hạn bước. Vì vậy thuật toán dừng mỗi khi các cặp (s, a) đã được áp dụng khắp lượt mà bảng $\hat{Q}(s, a)$ không đổi hoặc độ lệch hiệu chỉnh lớn nhất nhỏ hơn ngưỡng ϵ cho trước.

Mặt khác, trong thuật toán trên, việc lấy ngẫu nhiên cặp (s, a) có thể dẫn đến hiện tượng bảng $\hat{Q}(s, a)$ không được điều chỉnh trong nhiều lần lặp ở bước 2. Vì vậy với các trường hợp đặc biệt, ta có các thuật toán phù hợp hơn

Thuật toán cho môi trường có trạng thái hút

Ta xét trường hợp thế giới mà môi trường có trạng thái hút và mọi trạng thái khởi tạo đều dẫn tới trạng thái này. Khi đó thuật toán sẽ thực hiện lặp các bước 2.3 và 2.4 ở trên khi thay s bởi trạng thái kế tiếp s' . Thuật toán được mô tả chi tiết trong bảng 9.2.

Bảng 9.2. Thuật toán học Q đơn định khi có trạng thái hút

Bước 1. Khởi tạo $\hat{Q}(s, a)$ bằng 0 cho mỗi cặp s, a

Bước 2. Thực hiện lặp:

- 2.1. Lấy ngẫu nhiên $s \in S$;
- 2.2. Thực hiện lặp đến khi s là trạng thái hút
 - 2.2.1. Lấy ngẫu nhiên $a \in \mathcal{A}(s)$;
 - 2.2.2. Xác định $r(s, a)$ và trạng thái kế tiếp s' ;
 - 2.2.3. Cập nhật bảng $\hat{Q}(s, a)$ theo quy tắc (9.7);
 - 2.2.4. $s \leftarrow s'$

Trường hợp này hàm \hat{Q} sẽ hội tụ tới Q sau hữu hạn bước. Trong thực hành, thuật toán dừng khi $\hat{Q}(s, a)$ không đổi hoặc độ lệch hiệu chỉnh lớn nhất nhỏ hơn ngưỡng ϵ cho trước.

Thuật toán cho trường hợp có chuỗi lặp vô hạn lần

Trường hợp không có trạng thái hút và tồn tại một chuỗi trạng thái-tác động vô hạn sao cho mỗi trạng thái $s \in S$ và tác động $a \in \mathcal{A}$ được lặp lại vô hạn lần. Khi độ dài chuỗi ra vô hạn, ta có thể áp dụng thuật toán sau.

Sau khi khởi tạo bảng giá trị $\hat{Q}(s, a)$, bộ học thực hiện lặp việc lấy ngẫu nhiên trạng thái ban đầu $s \in S$ và thực hiện chuỗi tác động để nhận được chuỗi trạng thái kết thúc ở trạng thái hút. Sau mỗi tác động, bảng $\hat{Q}(s, a)$ được cập nhật và thực hiện

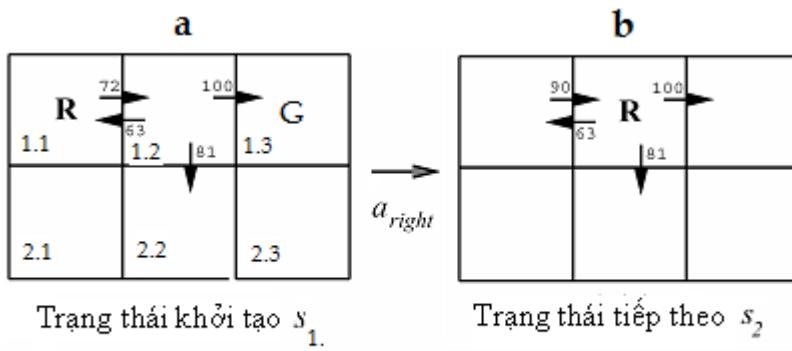
như ở các bước 2.2.2-2.2.4 ở trên. Thủ tục lặp xây dựng ngẫu nhiên chuỗi trạng thái thực hiện cho đến khi mỗi trạng thái của S đều thuộc ít nhất một chuỗi mà $\hat{Q}(s, a)$ không đổi hoặc độ lệch hiệu chỉnh lớn nhất nhỏ hơn ngưỡng ϵ cho trước. Chi tiết thuật toán được mô tả trong bảng 9.3.

Bảng 9.3. Thuật toán học Q đơn định cho trường hợp có chuỗi lặp vô hạn lần

- | |
|--|
| <p>Bước 1. Khởi tạo $\hat{Q}(s, a)$ bằng 0 cho mỗi cặp s, a</p> <p>Bước 2. Lấy ngẫu nhiên $s \in S$;</p> <p>Bước 3. Thực hiện lặp đến khi dừng:</p> <ol style="list-style-type: none"> 3.1. Lấy ngẫu nhiên $a \in \mathcal{A}(s)$; 3.2. Xác định $r(s, a)$ và trạng thái kế tiếp s'; 3.3. Cập nhật bảng $\hat{Q}(s, a)$ theo quy tắc (9.7); 3.4. $s \leftarrow s'$ |
|--|

Ví dụ minh họa

Trong hoạt động của thuật toán học Q ở trên, điểm then chốt là thủ tục cập nhật bảng giá trị của \hat{Q} sau mỗi tác động được chọn. Để minh họa, ta xét bài toán đơn giản với các giá trị thường tức thời cho trong hình 9.2a. Tại thời điểm đang xét tác tử ở ô 1.1 và giá trị liên quan của ước lượng \hat{Q} được biểu diễn trong hình 9.3a, với hệ số chiết khấu $\gamma = 0.9$. Lúc này giá trị ước lượng $\hat{Q}(s_1, a_{right}) = 72$. Tác tử thực hiện tác động a_{right} đi sang phải như mô tả trong hình 9.3 và nhận được giá trị thường $r = 0$.



$$\hat{Q}(s, a) \leftarrow r + \gamma \max_{a'} \hat{Q}(s', a') = 0 + 0.9 \max \{63, 81, 100\} = 90$$

Hình 9.3. a) Tác tử ở trạng thái s_1 ở ô 1.1 và các ước lượng \hat{Q} hiện thời. b) Cập nhật \hat{Q} sau khi tác tử thực hiện tác động sang phải.

Sau đó nó thực hiện cập nhật giá trị $\hat{Q}(s_1, a_{right})$ theo quy tắc (9.7) dựa trên ước lượng đã có của \hat{Q} cho trạng thái mới s_2 trong ô 1.2:

$$\hat{Q}(s, a) \leftarrow r + \gamma \max_{a'} \hat{Q}(s', a') = 0 + 0,9 \max_{a'} \hat{Q}(s', a') \xrightarrow{?} 90.$$

Theo quy tắc huấn luyện này, một ước lượng \hat{Q} mới cho tác động chuyển trạng thái sẽ là tổng của giá trị thưởng thu được (0/100) cộng với giá trị \hat{Q} cao nhất liên quan với trạng thái kết quả (100) được chiết khấu với hệ số γ (0.9). Sau khi điều chỉnh \hat{Q} ở ô 1.1, tác tử ở trạng thái mới s_2 , nó chọn tiếp tác động và cập nhật tăng cường \hat{Q} cho đến khi nó ở trạng thái hút (ô 1.3) thì kết thúc một bước lặp 3 trong bảng 9.2. Nếu điều kiện dừng chưa thỏa mãn thì nó chọn ngẫu nhiên trạng thái khởi tạo mới và thực hiện các tác động cho tới khi đạt tới được trạng thái đích mong muốn và kết thúc lần lặp mới.

Hình 9.2d mô tả một chiến lược điều khiển tối ưu cho bài toán đang xét với các giá trị cập nhật cuối cùng của $\hat{Q} = Q$ cho trong hình 9.2c. Người đọc có thể tìm ra chính sách tối ưu khác nhau sửa đổi cách chọn tác động (chiến lược thực nghiệm) cho mỗi trạng thái. Trước khi thảo luận về chiến lược thực nghiệm, ta giới thiệu một chứng minh tính hội tụ cho thuật toán học Q tổng quát trong bảng 9.1.

Sự hội tụ

Để chứng minh tính hội tụ cho các thuật toán trên, ta giả thiết hệ thống thỏa mãn ba điều kiện:

- Trước hết ta giả thiết rằng hệ thống là một MDP đơn định.
- Thứ hai: hệ số chiết khấu $\gamma < 1$ và các giá trị thưởng ở mỗi bước bị chặn, tức là tồn tại hằng số nào đó c sao cho với mọi trạng thái s và tác động a ta có: $|r(s, a)| < c$.
- Thứ ba: thuật toán không dừng và theo thời gian tác tử chọn thăm mỗi cặp trạng thái- tác động (s, a) nhiều vô hạn lần.

Điều kiện thứ ba đòi hỏi rằng theo thời gian, mọi tác động $a \in \mathcal{A}$ ở trạng thái s đều được tác tử lặp lại việc thực hiện nó với tần suất khác không khi độ dài của dãy tác động dần ra vô hạn.

Chú ý. Với mọi chiến lược điều khiển π và trạng thái ban đầu $s \in S$, giá trị tích lũy thưởng $V^\pi(s)$ xác định theo công thức (9.2a) khi $s = s_t$, ta có đánh giá:

$$|V^\pi(s)| = \left| \sum_{i=0}^{\infty} \gamma^i r_i \right| \leq c \sum_{i=0}^{\infty} \gamma^i = \frac{c}{1-\gamma}.$$

Như vậy $V^\pi(s)$ hội tụ tuyệt đối và bị chặn với mọi π và s và do đó $V^*(s)$ cũng bị chặn với cùng hằng số. Từ công thức (9.4) ta suy ra $Q(s, a)$ cũng bị chặn với mọi cặp (s, a) . Tương tự ta cũng có các ước lượng của $\hat{Q}(s, a)$ bị chặn.

Bây giờ ta ký hiệu $\hat{Q}_n(s, a)$ là giá trị tương ứng ở bảng \hat{Q} khi nó được cập nhật lần thứ n . Khi đó ta có định lý sau.

Định lý 9.1. *Nếu thế giới của tác tử thỏa mãn ba điều kiện nêu trên thì $\hat{Q}_n(s, a)$ sẽ hội tụ về $Q(s, a)$ khi $n \rightarrow \infty$ với mọi cặp giá trị s, a .*

Chứng minh. Bởi vì mỗi cặp trạng thái-tác động (s, a) nhiều vô hạn lần nên ta xét các khoảng mà mỗi cặp đều xảy ra ít nhất một lần. Giả sử \hat{Q}_n là bảng ước lượng Q sau n kỳ cập nhật như thế và Δ_n là cực đại lỗi trong \hat{Q}_n :

$$\Delta_n = \max_{s, a} |\hat{Q}_n(s, a) - Q(s, a)|.$$

Về sau ta dùng s' để ký hiệu $\delta(s, a)$. Bây giờ, đối với mỗi phần tử trong bảng \hat{Q}_n được cập nhật lần thứ $n+1$, ước lượng lỗi của nó trong bảng \hat{Q}_{n+1} là:

$$\begin{aligned} |\hat{Q}_{n+1} - Q(s, a)| &= |(r(s, a) + \gamma \max_{a'} \hat{Q}_n(s', a')) - (r(s, a) + \gamma \max_{a'} Q(s', a'))| \\ &= \gamma |(\max_{a'} \hat{Q}_n(s', a')) - (\max_{a'} Q(s', a'))| \\ &\leq \gamma \max_{a'} |\hat{Q}_n(s', a') - Q(s', a')| \\ &\leq \gamma \max_{s'', a''} |\hat{Q}_n(s'', a'') - Q(s'', a'')| \end{aligned}$$

Như vậy $|\hat{Q}_{n+1} - Q(s, a)| \leq \gamma \Delta_n$, suy ra $\Delta_{n+1} \leq \gamma \Delta_n$. Suy luận đệ quy ta có

$$\Delta_n \leq \gamma^n \Delta_0$$

Bởi vì $Q(s, a)$ và $\hat{Q}(s, a)$ bị chặn nên Δ_0 bị chặn và $\lim_{n \rightarrow \infty} \Delta_n = 0$, đpcm

Chiến lược thí nghiệm

Bây giờ ta thảo luận về chiến lược chọn trạng thái khởi tạo và tác động của tác tử trong các phiên bản của thuật toán mà ở trên chưa chỉ rõ.

Một chiến lược tự nhiên là bộ học chọn trạng thái s theo phân bố đều, các tác động thì chọn tác động a có $\hat{Q}(s, a)$ lớn nhất, nhờ đó khai thác tính xấp xỉ \hat{Q} hiện thời của nó. Tuy nhiên, chiến lược này làm cho tác tử làm giảm tính khám phá và không đảm bảo điều kiện “mỗi cặp trạng thái-tác động được lặp vô hạn lần” của định lý hội tụ. Vì lý do này, người ta thường dùng cách tiếp cận ngẫu nhiên để chọn tác động thuật toán học Q . Những tác động có giá trị \hat{Q} cao sẽ được gán cho một xác suất cao hơn, nhưng mọi tác động đều được gán một xác suất khác 0. Một trong các cách gán xác suất là:

$$P(a_i | s) = \frac{k^{\hat{Q}(s, a_i)}}{\sum_j k^{\hat{Q}(s, a_j)}} \quad (9.8)$$

trong đó $P(a_i | s)$ là xác suất lựa chọn tác động a khi tác tử ở trạng thái (s , $k > 1$ (có thể chọn $k = e$, là hằng số) thể hiện mức độ ưu tiên lựa chọn các tác động. Theo cách đó, tác tử có thể sẽ lựa chọn cả những tác động mà hiện tại chưa có giá trị \hat{Q} lớn. Trong một số trường hợp k được tổng hợp từ nhiều kết quả khác nhau mà tác tử đã thu được từ các trạng thái trước của quá trình học và chính sách tác động sẽ theo đó mà từng bước được thiết lập.

9.2.2. Học Q trong các bài toán ngẫu nhiên

Ở trên chúng ta đã đề cập đến thuật toán học Q trong môi trường đơn định. Trong phần này chúng ta sẽ đề cập tới trường hợp ngẫu nhiên (không đơn định), trong đó hàm thưởng $r(s, a)$ và hàm chuyển trạng thái $\delta(s, a)$ không phụ thuộc vào các trạng thái và tác động trước đó. Nói cách khác hệ thống là quá trình quyết định Markov không đơn định.

Để phát triển thuật toán học Q ở trên cho trường hợp MDP không đơn định, ta xem lại hành trình để dẫn tới thuật toán trong hợp đơn định và sửa đổi những chỗ cần thiết. Trở lại với hàm mục tiêu của trường hợp ngẫu nhiên, giá trị tích lũy thưởng V^π cho một chiến lược điều khiển π xác định theo biểu thức (9.3a)

$V^\pi(s_t) \equiv E\left[\sum_{i=0}^{\infty} \gamma^i r_{t+i}\right]$, trong đó, dãy giá trị thưởng r_{t+i} được sinh ra bởi chính sách π với trạng thái khởi đầu là s_0 . Chiến lược điều khiển tối ưu được biểu thị bởi biểu thức (9.3b) $\pi^* = \arg \max_\pi V^\pi(s_t), \forall s_t$. Ta cần tổng quát hóa định nghĩa trước đây về Q từ công thức (9.4) qua giá trị kỳ vọng của nó.

$$\begin{aligned} Q(s, a) &\equiv E[r(s, a) + \gamma V^*(\delta(s, a))] \\ &= E[r(s, a)] + \gamma E[V^*(\delta(s, a))] \\ &= E[r(s, a)] + \gamma \sum_{s'} P(s' | s, a) V^*(s') \end{aligned} \quad (9.9a)$$

trong đó $P(s' | s, a)$ là xác suất trạng thái tiếp theo là s' nếu thực hiện tác động a tại trạng thái s . Như vậy ta có thể biểu diễn Q một cách đệ quy nhờ tổng quát hóa công thức (9.6)

$$Q(s, a) = E(r(s, a)) + \gamma \sum_{s'} P(s' | s, a) \max_{a'} Q(s', a'). \quad (9.9b)$$

Để ước lượng Q , ta không dùng được công thức (9.7) vì nó không hội tụ trong trường hợp không đơn định. Chẳng hạn, xét hàm thưởng không đơn định $r(s, a)$ sinh ra các giá trị khác nhau mỗi khi thực hiện lặp lại việc chuyển trạng thái $\langle s, a \rangle$. Khi

đó, luật huấn luyện sẽ lặp lại việc thay đổi giá trị của $\hat{Q}(s, a)$, ngay cả khi ta khởi tạo bảng giá trị \hat{Q} bằng đúng giá trị đúng của Q . Trong trường hợp này, luật huấn luyện sẽ không hội tụ. Khó khăn này được khắc phục như sau. Ta dùng ký hiệu \hat{Q}_n để biểu thị cho ước lượng của tác tử tại bước lặp thứ n của thuật toán, khi đó luật huấn luyện hiệu chỉnh sau đây đủ đảm bảo cho \hat{Q} hội tụ về giá trị Q :

$$\hat{Q}_n(s, a) \leftarrow (1 - \alpha_n)\hat{Q}_{n-1}(s, a) + \alpha_n[r + \gamma \max_{a'} \hat{Q}_{n-1}(s', a')] \quad (9.10a)$$

với:

$$\alpha_n = \frac{1}{1 + \text{visits}_n(s, a)} \quad (9.10b)$$

trong đó s và a là trạng thái và tác động đã được cập nhật cho đến bước lặp n , và $\text{visits}_n(s, a)$ là tổng số lần cặp trạng thái-tác động này được thăm và thực hiện tính tới hết bước lặp thứ n .

Tư tưởng chính trong công thức huấn luyện hiệu chỉnh là hiệu chỉnh giá trị \hat{Q} một cách chậm hơn so với trường hợp hệ đơn định. Chú ý là nếu ta đã đặt $\alpha_n = 1$ trong công thức (9.10), ta sẽ thu được luật huấn luyện cho trường hợp hệ đơn định. Với α có giá trị nhỏ, số hạng này sẽ trở thành lấy trung bình với giá trị $\hat{Q}(s, a)$ hiện tại để sinh ra một giá trị cập nhật mới. Chú ý là giá trị của α_n trong công thức (9.10b) sẽ giảm khi n giảm, và như vậy mức độ hiệu chỉnh của quá trình huấn luyện là nhỏ. Bằng việc giảm α_n ở một mức độ phù hợp trong quá trình huấn luyện, ta có thể đạt được sự hội tụ về giá trị Q . Việc lựa chọn α_n như ở trên là một trong nhiều cách để đảm bảo điều kiện cho sự hội tụ, dựa trên định lý của Watkins và Dayan (1992).

Sự hội tụ của thuật toán học Q cho MDP không đơn định.

Xét một bộ học thực hiện học Q trong một MDP không đơn định với các giá trị thưởng bị chặn: $(\forall s, a) | r(s, a)| \leq c$. Bộ học Q sử dụng luật học (9.10a) với bảng giá trị $\hat{Q}(s, a)$ được khởi tạo bằng những giá trị hữu hạn tùy ý và sử dụng hệ số chiết khấu γ thỏa mãn $0 < \gamma < 1$. Gọi $n(i, s, a)$ là bước lặp ứng với lần thứ i tác động a tại trạng thái s được thực hiện thì khẳng định sau đúng.

Định lý 9.2. Nếu mỗi cặp trạng thái – tác động được thăm vô hạn lần, $0 \leq \alpha < 1$ và

$$\sum_{i=1}^{\infty} \alpha_{n(i, s, a)} = \infty, \quad \sum_{i=1}^{\infty} [\alpha_{n(i, s, a)}]^2 < \infty$$

thì với mọi cặp s và a , ta có $\hat{Q}_n(s, a) \rightarrow Q(s, a)$ khi $n \rightarrow \infty$, với xác suất bằng 1.

Trong thực tế thuật toán học Q trong học tăng cường thường phải lặp lại nhiều nghìn lần lặp huấn luyện mới hội tụ. Để làm ví dụ ứng dụng, dưới đây dưới thiệu phương pháp tối ưu đàm kiến giải các bài toán tối ưu tổ hợp.

9.3. PHƯƠNG PHÁP TỐI ƯU ĐÀM KIẾN (ACO)

Như đã trình bày ở trên, phương pháp họ Q dựa trên ý tưởng thực nghiệm mô phỏng hệ thống nhiều lần để quan sát hệ thống và giá trị thường tức thời được xem là bình luận được lượng hóa cho mỗi tác động, giá trị tích lũy thường biểu thị bình luận tổng hợp cho mỗi thí nghiệm. Trong mỗi thí nghiệm, các giá trị thường xa với thời điểm quan sát hiện thời sẽ chịu lượng chiết khấu lớn giải theo hàm mũ ($\gamma < 1$) theo khoảng tới thời điểm hiện tại.

Trong tự nhiên, trên đường đi, mỗi con kiến tiết ra một hóa chất gọi là vết mùi (pheromone trail) và theo vết mùi của các con kiến khác để xác định đường đi, lối rẽ có vết mùi càng cao thì khả năng kiến chọn càng lớn. Phương pháp tối ưu đàm kiến (ant colony optimization: ACO) do Dorigo đề xuất (1991) mô phỏng cách tìm đường đi của đàm kiến để giải các bài toán tối ưu tổ hợp là ứng dụng của ý tưởng học tăng cường này. Thuật toán tìm đường đi của robot trong chương 1 là một thể hiện ứng dụng của ACO. Trước hết ta phát biểu bài toán tối ưu tổ hợp tổng quát

9.3.1. Phát biểu bài toán tối ưu tổ hợp tổng quát

Xét bài toán cực tiểu hóa (S, f, Ω) , trong đó S là tập hữu hạn trạng thái, f là hàm mục tiêu xác định trên S còn Ω là các ràng buộc để xác định S qua các thành phần của tập hữu hạn C và các liên kết của tập này. Các tập S, C và Ω có các đặc tính sau.

- 1) $C = \{c_1, \dots, c_n\}$ là tập hữu hạn gồm n thành phần. Ta ký hiệu X là tập các dãy trong C độ dài không quá h : $X = \{<u_0, \dots, u_h> / u_i \in C \quad \forall i \leq k \leq h\}$.
- 2) Tồn tại tập con X^* của X và ánh xạ φ từ X^* lên S sao cho $\varphi^{-1}(s)$ không rỗng với mọi $s \in S$. Trong đó tập X^* có thể xây dựng được từ tập con C_0 nào đó của C nhờ mở rộng tuần tự theo đặc tính 3 dưới đây.
- 3) Từ C_0 mở rộng được thành X^* theo thủ tục tuần tự:
 - i) $x_0 = <u_0>$ là mở rộng được với mọi $u_1 \in C_0$.
 - ii) Nếu $x_k = <u_1, \dots, u_k>$ là mở rộng được thì tồn tại Ω xác định được tập con $J(x_k)$ của C sao cho với mọi $u_{k+1} \in J(x_k)$ thì $x_{k+1} = <u_0, \dots, u_k, u_{k+1}>$ là mở rộng được hoặc $x_k \in X^*$ khi $J(x_k)$ là rỗng.
 - iii) Với mọi $u_0 \in C_0$, thủ tục mở rộng nêu trên xây dựng được mọi phần tử của X^* .

Ví dụ: Bài toán người chào hàng

Bài toán người chèo hàng (TPS) được phát biểu như sau. Với n thành phố đã cho, người chèo hàng cần tìm một chu trình có đường đi ngắn nhất qua mỗi thành phố đúng một lần.

Để đơn giản, ta xét bài toán trên đồ thị vô hướng $G = (V, E)$, trong đó tập đỉnh V (n đỉnh) ký hiệu tập các thành phố và E là tập các cạnh (i, j) biểu thị đường đi nối các đỉnh $i, j \in V$ và có độ dài $d_{i,j}$ tương ứng ($d_{i,j}$ là khoảng cách từ thành phố i tới thành phố j). Trong trường hợp này S là các chu trình trên đồ thị đầy, f là độ dài đường đi, Ω là ràng buộc các chu trình qua mọi đỉnh (và mỗi đỉnh một lần) còn C_0 là tập các đỉnh của đồ thị.

Các bài toán tối ưu tổ hợp như trên thường thuộc loại NP-khó, phương pháp ACO để xuất xây dựng đồ thị cấu trúc với tập đỉnh V mà mỗi đỉnh của nó tương ứng với mỗi thành phần của C và có thể dùng các thuật toán theo lược đồ dưới đây để giải.

9.3.2. Thuật toán tổng quát

Nói chung, đối với các bài toán thuộc loại NP-khó thường ta có các phương pháp heuristic để tìm lời giải đủ tốt cho bài toán. Các thuật toán ACO kết hợp thông tin heuristic này với phương pháp học tăng cường nhòm phỏng hành vi của đàn kiến để tìm lời giải tốt hơn.

Đồ thị cấu trúc

Giả sử với mỗi cạnh nối các đỉnh $i, j \in C$ có trọng số heuristic $h_{i,j}$ để định hướng chọn thành phần mở rộng là j khi thành phần cuối của x_k là i theo thủ tục nêu trên ($h_{i,j} > 0 \forall (i, j)$). Đàn kiến m con sẽ xây dựng lời giải trên đồ thị có trọng số $G = (V, E, H, \tau)$, trong đó V là tập đỉnh tương ứng với tập thành phần C đã nêu ở trên (về sau ta vẫn dùng ký hiệu C thay cho V để chỉ tập đỉnh), E là tập các cạnh, H là vectơ các trọng số heuristic của cạnh tương ứng (trong bài toán TPS nó là vectơ mà thành phần là nghịch đảo độ dài của cạnh tương ứng) còn τ vectơ vết mìt tích luỹ được ban đầu được khởi tạo bằng τ_0 cho mọi thành phần đều bằng $\tau_0 > 0$. Để dễ trình bày chúng tôi xét vết mìt τ để ở các cạnh tuy rằng nó có thể để ở các đỉnh của đồ thị (xem [11]). Đồ thị G sẽ gọi là đồ thị cấu trúc của bài toán.

Với điều kiện kết thúc đã chọn (có thể dùng số lần lặp N_c định trước) thuật toán được mô tả hình thức như trong hình 9.4.

Xây dựng lời giải trong mỗi bước

Sau khi khởi tạo các tham số và lượng mùi ban đầu, các con kiến thực hiện lặp thủ tục xây dựng lời giải. Trong mỗi lần lặp t , mỗi con kiến h chọn ngẫu nhiên một đỉnh xuất phát trong C_0 và kết hợp thông tin heuristic với thông tin mùi để xây dựng lời giải ngẫu nhiên theo thủ tục mở rộng tuần tự nêu trên với xác suất chọn đỉnh tiếp theo như sau.

Procedure of ACO algorithms.

Begin

Initialize //khởi tạo m con kiến

Repeat

 Construct solutions// mỗi con kiến xây dựng lời giải,

 Update trail//cập nhật mùi

until End condition//điều kiện kết thúc

End

Hình 9.4. Đặc tả thuật toán ACO

Các thủ tục xây dựng lời giải và cập nhật mùi thực hiện như sau.

Quy tắc chuyển trạng thái. Giả sử kiến s đã xây dựng $x_k = \langle u_0, \dots, u_k \rangle$, nó chọn đỉnh y thuộc $J(x_k)$ để $x_{k+1} = \langle u_0, \dots, u, y \rangle$ với xác suất $P(y/\tau, x_k)$:

$$P(y/\tau, x_k) = \begin{cases} \tau_{u,y}^\alpha h_{u,y} & y \in J(x_k) \\ 0 & \text{ngc li} \end{cases} \quad (9.11)$$

trong đó $\alpha \in (0,1)$ là tham số chọn trước.

Quá trình này tiếp tục cho tới khi mỗi con kiến r đều tìm được một lời giải chấp nhận được $x(r) \in X^*$ và do đó $s(r) = \varphi(x(r)) \in S$. Khi đó ta sẽ nói các cạnh (u_i, u_{i+1}) thuộc $X(r)$. Để tiện trình bày, về sau ta sẽ xem $x(r)$ và $s(r)$ như nhau và không phân biệt X^* với S . Ký hiệu $w(t)$ là lời giải tốt nhất các con kiến tìm được cho tới lần lặp này và $w^i(t)$ là lời giải tốt nhất trong bước lặp, nếu $w^i(t)$ không tốt hơn $w(t-1)$ ta sẽ có $w(t) := w(t-1)$. Ta sẽ quan tâm tới các lời giải gần đúng $w^i(t)$ này.

Do giả thiết (3-iii) của bài toán và để tiện trình bày, về sau ta sẽ không phân biệt mỗi $x \in X^*$ với trạng thái $s \in S$ tương ứng.

Cập nhật mùi

Quy tắc cập nhật mùi có vai trò quan trọng, quyết định chất lượng thuật toán. Có nhiều quy tắc cập nhật mùi, sau đây là một số quy tắc được sử dụng phổ biến nhất hiện nay. Giả sử g là một hàm giá trị thực xác định trên S sao cho $0 < g(s) < \infty$ $\forall s \in S$ và $g(s) > g(s')$ nếu $f(s) < f(s')$ (trong bài toán TSP $g(s)$ là nghịch đảo độ dài đường

đi tương ứng), khi đó ở mỗi cuối mỗi bước lặp cường độ vết mùi sẽ thay đổi theo một trong các quy tắc sau đây.

Quy tắc ACS: Quy tắc này bao gồm cả cập nhật địa phương và toàn cục.

Cập nhật mùi địa phương. Nếu con kiến h thăm cạnh (i,j) , tức là $(i,j) \in s(h)$ thì cạnh này sẽ thay đổi mùi theo công thức:

$$\tau_{i,j} \leftarrow (1 - \rho)\tau_{i,j} + \rho\tau_1 \quad (9.12a)$$

trong đó ở đây $\tau_1 > 0$, $\rho \in (0,1)$ là tham số (có thể lấy $\tau_1 = \tau_0$).

Cập nhật mùi toàn cục. Cập nhật mùi toàn cục chỉ cho các cạnh thuộc $w(t)$:

$$\tau_{i,j} \leftarrow (1 - \rho)\tau_{i,j} + \rho W(t) \quad \forall (i,j) \in w(t) \quad (9.12b)$$

trong đó $W(t) = f(w(t))$.

Quy tắc MMAS. Sau khi mỗi con kiến đều xây dựng xong lời giải ở mỗi bước lặp, vết mùi được thay đổi theo công thức:

$$\tau_{i,j} \leftarrow (1 - \rho)\tau_{i,j} + \Delta_{i,j} \quad (13a)$$

$$\text{trong đó } \Delta_{i,j} = \begin{cases} \rho g(W(t)): & (i,j) \in w(t) \\ \max\{\tau_1 - (1 - \tau_{1,j}), 0\}: & (i,j) \notin w(t) \end{cases} \quad (9.13b)$$

Quy tắc SMMAS. Vết mùi được thay đổi theo công thức:

$$\tau_{i,j} \leftarrow (1 - \rho)\tau_{i,j} + \Delta_{i,j}; \quad \Delta_{i,j} = \begin{cases} \rho\tau_{max} & (i,j) \in w(t) \\ \rho\tau_{min} & (i,j) \notin w(t) \end{cases} \quad (9.14)$$

trong đó $\tau_{max} > \tau_{min}$ là hai hằng số chọn trước. Tỷ lệ $\frac{\tau_{max}}{\tau_{min}}$ phản ánh chiến lược khám phá hay tìm kiếm tăng cường và có ảnh hưởng quan trọng tới chất lượng của thuật toán.

Các quy tắc cập nhật mùi trên là quy tắc G-best, nếu trong các công thức (9.12a) đến (9.14) thay $w(t)$ bởi $w^i(t)$ thì ta nói là quy tắc i-best, thường thì cập nhật i-best tốt hơn G-best.

Người ta đã chứng minh lời giải gần đúng $w(t)$ hội tụ theo xác suất tới lời giải tối ưu của bài toán. Trong các quy tắc cập nhật mùi ở trên, quy tắc SMMAS đơn giản, dễ sử dụng và hiệu quả hơn.

Nhận xét. Phương pháp ACO không đặt vấn đề tìm chiến lược tối ưu như bài toán học tăng cường mà chỉ xây dựng quá trình quyết định Markov tối ưu với trạng thái ban đầu thuộc tập C_0 là lời giải cho bài toán. Việc chọn định kế tiếp trong thủ tục tuần tự được xem là tác động. Vết mùi cập nhật trên mỗi cạnh tương ứng với giá trị tích lũy thưởng, việc thưởng, các đại lượng $\Delta_{i,j}$ là lượng thưởng *tức thời*. Khác với học Q ở chỗ lượng chiết khấu $1 - \rho$ áp dụng cho các giá trị thưởng *tức thời* trước đó,

càng xa hiện tại thì số lần chiết khấu càng tăng, việc thưởng cho mỗi lần lặp trong các quy tắc cập nhật mùi thực hiện cho mọi cạnh (tác động) sau khi đã được bình luận (đánh giá). Như vậy, về bản chất ACO là một vận dụng của phương pháp học Q cho bài toán chỉ tìm một quá trình quyết định Markov tối ưu. Các kết quả thực nghiệm cho thấy phương pháp này hiệu quả hơn giải thuật di truyền.

KẾT LUẬN

Học tăng cường là bài toán học tìm chính sách điều khiển tối ưu của một tác tử tự trị. Trong đó giả thiết rằng thông tin học biểu diễn dưới dạng giá trị thưởng thực được cho đổi với mỗi phép chuyển trạng thái-tác động. Mục đích của tác tử là học một chính sách tác động làm cực đại giá trị thưởng tích lũy mà nó nhận được từ bất cứ trạng thái khởi đầu nào.

Các thuật toán học tăng cường được trình bày trong chương này làm việc với quá trình quyết định Markov (MPS), trong đó kết quả của mỗi tác động ở bất cứ trạng thái nào chỉ phụ thuộc vào tác động và trạng thái đang xét mà không phụ thuộc vào các tác động và trạng thái trước đó. Các MPS thường gặp trong các bài toán ở phạm vi rộng, bao gồm những bài toán điều khiển robot, tự động hóa nhà máy, lập lịch.

Học Q là một dạng của học tăng cường, trong đó tác tử học một hàm đánh giá các trạng thái và tác động. Đặc biệt, hàm đánh giá $Q(s,a)$ được định nghĩa như là kỳ vọng tích lũy thưởng có chiết khấu mà tác tử có thể nhận được khi thực hiện tác động a ở trạng thái s . Các thuật toán học Q có ưu điểm là nó có thể sử dụng ngay cả khi bộ học không có tri thức trước về việc các tác động ảnh hưởng tới môi trường ra sao.

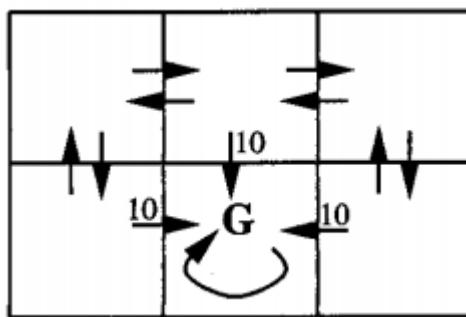
Các ước lượng $\hat{Q}(s,a)$ cho hàm Q được biểu diễn dưới dạng bảng tra cứu với các thành phần riêng biệt cho mỗi cặp (s,a) . Với một số giả thiết trong học Q, ước lượng $\hat{Q}(s,a)$ hội tụ đến hàm đúng $Q(s,a)$ cả trong trường hợp các MDP đơn định và ngẫu nhiên. Trong ứng dụng, học Q có thể đòi hỏi nhiều nghìn lần lặp huấn luyện cho một bài toán cõi tầm thường.

Trong học Q, cơ chế chuyển tác động-trạng thái đóng vai trò bản chất nội tại của thuật toán, giá trị thưởng tức thời và tích lũy thưởng có chiết khấu đóng vai trò như lượng hóa sự bình luận hiệu quả của các tác động và các quá trình MDP, với mỗi bài toán, người thiết kế bộ học có thể thay đổi thích hợp các giá trị này.

Phương pháp ACO là một tiếp cận vận dụng phương pháp học Q cho trường hợp chỉ cần tìm một xích Markov có trạng thái khởi đầu thuộc tập C_0 . Thực nghiệm cho thấy hiệu quả nổi trội của nó so với các phương pháp khác như mô phỏng luyen kim hay thuật toán di truyền.

BÀI TẬP

- Giải thích các số liệu trong hình 9.2b và tìm một chính sách tối ưu thứ hai cho bài toán được mô tả trong hình 9.2
- Đưa ra các chuỗi tác động-trạng thái trong thuật toán học Q để giải thích làm thế nào có được các số liệu trong bản 9.3a?
- Xét thế giới lưới đơn định với trạng thái hút G cho trong hình bên dưới. Trong đó các phép chuyển trạng thái có nhãn nhận giá trị thưởng tức thời bằng 10, các phép chuyển không có nhãn nhận giá trị thưởng bằng không.



- Tính giá trị V^* cho mỗi trạng thái trong thế giới lưới này. Với $\gamma = 0,8$ tính $Q(s, a)$ cho mỗi tác động và cuối cùng chỉ ra một chiến lược điều khiển tối ưu.
 - Gợi ý một thay đổi hàm thưởng $r(s, a)$ sao cho nó làm thay đổi $Q(s, a)$ nhưng không thay đổi chính sách tối ưu.
 - Gợi ý một thay đổi cho $r(s, a)$ sao cho nó làm thay đổi $Q(s, a)$ nhưng không thay đổi $V^*(s)$.
 - Xét thuật toán học Q với bảng \hat{Q} được khởi tạo bằng không, giả sử tác tử khởi đầu ở trạng thái bên trái-dưới và đi theo chiều kim đồng hồ tới trạng thái hút. Thực hiện hiệu chỉnh bảng \hat{Q} cho mỗi tác động ở giai đoạn này. Hãy mô tả tiếp thay đổi của bảng \hat{Q} nếu tác tử thực hiện lại lần hai cùng quỹ đạo.
- Lưu ý rằng trong nhiều MDP, có thể tìm được hai chính sách điều khiển π_1 và π_2 sao cho khi tác tử khởi đầu từ trạng thái s_1 thì $V^{\pi_1}(s_1) > V^{\pi_2}(s_1)$ còn khi khởi đầu từ trạng thái s_2 thì $V^{\pi_2}(s_2) > V^{\pi_1}(s_2)$. Giải thích tại sao luôn tồn tại một chính sách là cực đại $V^\pi(s)$ đối với mọi trạng thái ban đầu s (tức là chính sách tối ưu π^*).
 - Chỉ rõ thuật toán ACO giải bài toán TSP (tìm chu trình Hamilton có độ dài ngắn nhất) với quy tắc cập nhật mì SMMAS và chỉ ra các yếu tố tương đồng với phương pháp học Q trong cơ chế vết mì và giá trị thưởng.

Chương 10

KẾT HỢP CÁC BỘ HỌC

Đã có nhiều thuật toán học khác nhau được giới thiệu trong các chương trước. Tuy nhiên, khi áp dụng thì không có thuật toán nào tạo nên được một bộ học thực sự chính xác hơn hẳn các phương pháp khác. Một cách tiếp cận có hiệu quả để tăng độ chính xác là kết hợp các bộ học nhận dạng với nhau để được một bộ tốt hơn hay còn gọi là học tập thể.

Chương này giới thiệu các kỹ thuật thông dụng trong kết hợp các bộ học: bỏ phiếu, tạo các bộ học cơ sở bằng cách nhặt dữ liệu theo gói (bagging) và nhặt định hướng (boosting), rùng ngẫu nhiên, kiến trúc bậc thang.

10.1. LÝ DO NÊN HỌC TẬP THỂ

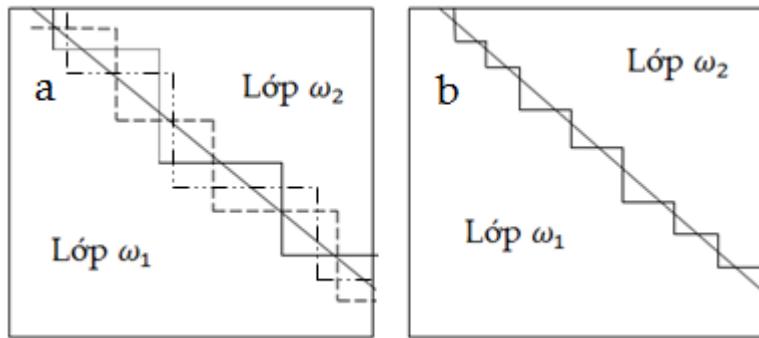
Với mỗi bài toán phân lớp hoặc hồi quy cụ thể, người ta thường có nhiều thuật toán học để khi xây dựng bộ học. Cùng một thuật toán, có thể chọn các tham số khác nhau hoặc sử dụng tập dữ liệu huấn luyện khác nhau nên cho các bộ nhận dạng khác nhau.

Ví dụ. Cùng dùng mạng MLP nhưng ta chọn số tầng ẩn/nơron tầng ẩn hay số đặc trưng khác nhau thì kết quả huấn luyện cho các bộ nhận dạng có chất lượng khác nhau. Cùng là phương pháp học k-NN nhưng với mỗi k cho ta một bộ nhận dạng và rất khó khẳng định được tham số k nào là tối ưu.

Những thuật toán cho cùng lớp bài toán thường tuân theo luật “*không có bữa trưa miễn phí (no free lunch theory)*”, tức là không có thuật toán tốt hơn hẳn các thuật toán mà mỗi thuật toán có ưu /nhược điểm riêng, khi thực hiện nhận dạng thì mỗi bộ huấn luyện theo thuật toán tương ứng có những lớp mẫu nhận dạng tốt và tồi khác nhau. Kết hợp hợp lý các bộ nhận dạng có thể cho ta bộ nhận dạng mới có nhiều ưu điểm hơn và ta gọi là học tập thể (ensemble learning).

Một bộ phân lớp được kết hợp từ 3 bộ phân lớp cơ sở khác nhau theo cách bỏ phiếu được minh họa trong hình 10.1. Hãy tưởng tượng hai lớp ω_1 và ω_2 có biên là đường thẳng nhưng hai bộ phân lớp cơ sở tạo nên các biên tương ứng là đường

liên và các đường đứt như trong hình 10.1a, còn hình 10.1b chỉ ra rằng bộ phân lớp kết hợp theo hình thức bỏ phiếu cho ta biên quyết định gần với biên thực hơn.



Hình 10.1. a) Biên quyết định của mỗi bộ phân lớp. b) Biên kết hợp ba bộ

Như vậy, mỗi cách học cho ta một bộ nhận dạng cơ sở, nhờ kết hợp các bộ nhận dạng thành phần có được mà ta có một bộ nhận dạng tốt hơn. Các bộ nhận dạng cơ sở này thường được xây dựng theo các tiếp cận sau đây:

- 1) *Dùng các thuật toán huấn luyện khác nhau.* Các thuật toán này sử dụng các giả thiết khác nhau về dữ liệu, các bộ học có thể phụ thuộc tham số hoặc không. Khi kết hợp các bộ học, ta được giải phóng khỏi các giả thiết áp đặt này.
- 2) *Mỗi bộ học dùng cách chọn đặc trưng khác nhau.* Chẳng hạn chúng ta dùng thuật toán SVM để phân biệt chữ viết tay nhưng cách chọn đặc trưng có thể là nội dung ảnh hay qua phép biến đổi nào đó.
- 3) *Có thể sử dụng cùng một thuật toán nhưng có tham số khác nhau.* Chẳng hạn đều sử dụng thuật toán k-láng giềng gần nhất nhưng với k khác nhau.
- 4) *Cùng một thuật toán nhưng sử dụng các tập dữ liệu huấn luyện khác nhau.*

Thông thường thì các bộ nhận dạng được xây dựng theo hai cách cách tiếp cận đều có thời gian chạy khác nhau và bộ nhận dạng chính xác hơn thường đòi hỏi thời gian xử lý nhiều hơn.

Khi có các bộ nhận dạng cơ sở, bộ nhận dạng tập thể được kết hợp theo các kiểu tôpô đa dạng để cho ta những bộ mới tốt hơn các bộ thành phần. Trong đó phương thức kết hợp đơn giản và dễ dàng nhất là phương pháp bỏ phiếu.

10.2. PHƯƠNG PHÁP BỎ PHIẾU

Một cách đơn giản để kết hợp các bộ học cơ sở là dùng phương pháp bỏ phiếu nhờ kiến trúc song song, đâu ra được quyết định nhờ kết quả tổng hợp có trọng số của các bộ nhận dạng thành phần. Đối với đối tượng x cần gán nhãn, nếu

mỗi bộ học cơ sở C_i cho quyết định \mathbf{d}_i với trọng số ý kiến w_i tương ứng thì đầu ra của bộ kết hợp đối với mẫu này được tính theo công thức:

$$\mathbf{d}(\mathbf{x}) = \sum_{i=1}^L w_i \mathbf{d}_i(\mathbf{x}) \text{ cho bài toán hồi quy,} \quad (10.1a)$$

và theo đa số có trọng số của tập $\{w_i \mathbf{d}_i(\mathbf{x})\}_{i=1}^L$ cho bài toán phân lớp, (10.1b)

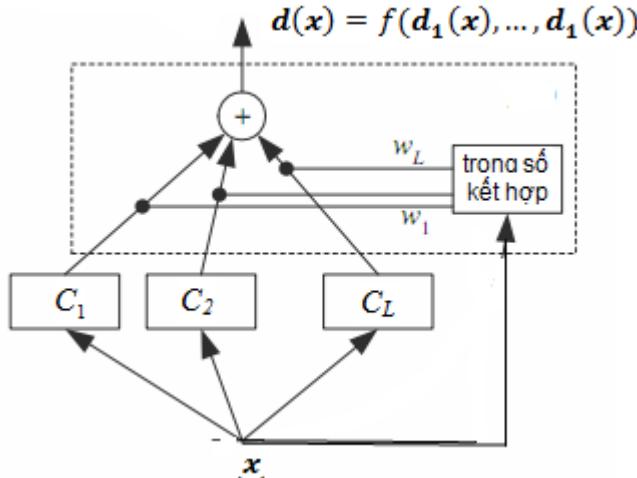
trong đó các trọng số thỏa mãn:

$$\sum_{i=1}^L w_i = 1 \quad (10.1c)$$

Các trọng số có thể chọn bằng nhau. Tổng quát hơn, ta có thể quyết định bằng một hàm tổng hợp phi tuyến f nào đó:

$$\mathbf{d}(\mathbf{x}) = f(\mathbf{d}_1(\mathbf{x}), \dots, \mathbf{d}_L(\mathbf{x})) \quad (10.2)$$

Sơ đồ quyết định tổng quát của quyết định theo hình thức bỏ phiếu được mô tả trong hình 10.2.

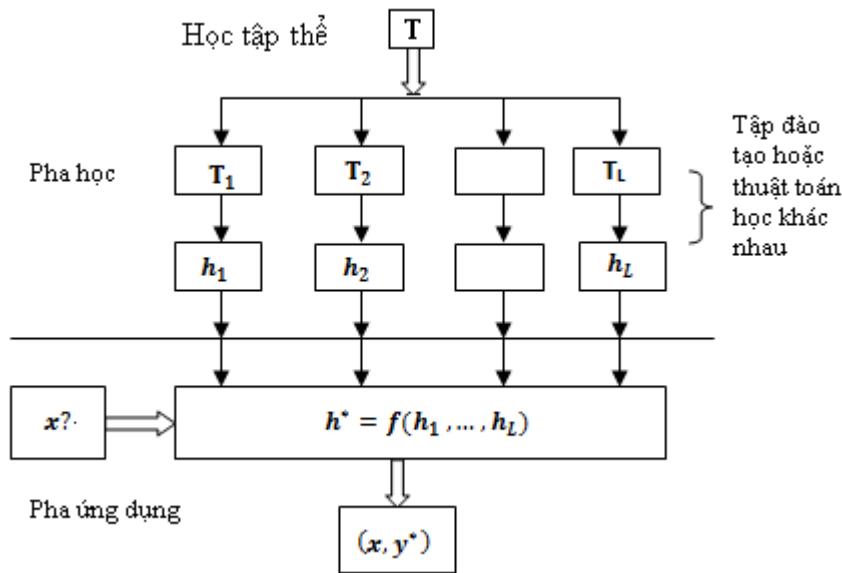


Hình 10.2 . Sơ đồ kết hợp các bộ nhận dạng nhờ bỏ phiếu

Việc huấn luyện các bộ thành phần của bộ học tập thể này có thể sử dụng một trong các phương thức sau:

- L thuật toán huấn luyện khác nhau.
- Một thuật toán nhưng L tập dữ liệu đào tạo hay tham số khác nhau.
- Một thuật toán nhưng dùng tập dữ liệu với tập đặc trưng khác nhau
- Kết hợp các phương thức trên.

Việc học tập thể T bao gồm các quá trình huấn luyện T_i cho bộ học C_i để cho giả thuyết h_i tương ứng và chúng được kết hợp thành giả thuyết h^* . Khi ứng dụng nhận dạng mẫu x , giả thuyết h^* sẽ cho ta nhãn $y^* = h^*(x)$ như minh họa trong hình 10.3.



Hình 10.3. Sơ đồ học tập thể của các bộ học $T_1 \ T_2 \ T_l \ h_1 \ h_2 \ h_L \ x? \ (x, y^*) \ h^* = f(h_1, \dots, h_L)$

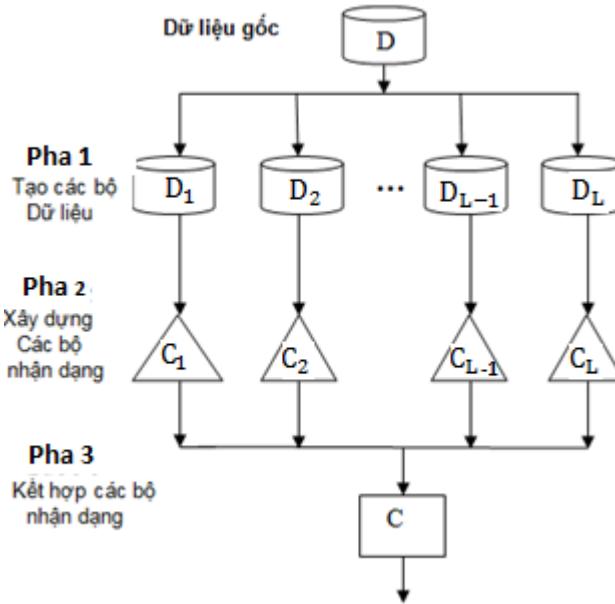
10.3. KỸ THUẬT TẠO VÀ KẾT HỢP BỘ NHẬN DẠNG CƠ SỞ

Mục này giới thiệu ba kỹ thuật thông dụng để tạo và kết hợp các bộ học cơ sở nhằm dùng một thuật toán học với cách tạo và dùng dữ liệu khác nhau: nhặt theo gói (bagging), nhặt định hướng bao gồm boosting và adaboost, rủi ro ngẫu nhiên.

10.3.1. Nhặt theo gói

Nhặt theo gói (Bagging) là phương pháp học tập thể đơn giản và thông dụng nhất, mặt khác, nó giúp giảm phương sai khi dùng cho bài toán hồi quy. Phương pháp này dùng cùng một thuật toán để xây dựng các bộ nhận dạng cơ sở bằng cách lấy ngẫu nhiên từ tập dữ liệu đào tạo một tập dữ liệu cho một bộ học, sau đó kết nối song song các bộ nhận dạng có được và quyết định với trọng số đều. Cụ thể như sau.

Giả sử ta có tập dữ liệu đào tạo $D = \langle \mathbf{x}^k, y^k \rangle_{k=1}^N$ gồm N mẫu đã gán nhãn. Từ một thuật toán cơ sở (chẳng hạn ID3, K-NN hay mạng nơron...) ta huấn luyện L bộ học trên L tập con dữ liệu được lấy ngẫu nhiên của D . Để có L tập dữ liệu này, ta chọn trước một số $M < N$ và với mỗi $i = 1, 2, \dots, L$, lấy ngẫu nhiên phân bố đều từ D tập D_i gồm M đối tượng làm dữ liệu đào tạo. Sau khi huấn luyện bằng thuật toán đã chọn trên các tập dữ liệu $\langle \mathbf{x}_i^k, y_i^k \rangle_{k=1}^N$ này ta có các bộ nhận dạng $g_i \in \mathcal{G}$. Các bộ nhận dạng này được kết hợp theo hình thức bỏ phiếu với trọng số bằng nhau, kiến trúc của bộ học này được mô tả trong hình 10.4.



Hình 10.4. Kiến trúc của hệ học theo phương pháp Bagging

Thuật toán xây dựng bộ nhận dạng được mô tả trong bảng 10.1.

Bảng 10.1. Thuật toán Bagging

Bước 1. Lặp với $i=1, 2, \dots, K$:

- 1.1. $D_i \leftarrow M$ mẫu đào tạo lấy ngẫu nhiên từ D ;
- 1.2. $h_i \leftarrow$ Học từ D_i theo thuật toán được chọn;

Bước 2. Kết hợp các h_i theo hình thức bỏ phiếu với trọng số đều.

Kỹ thuật này có thể áp dụng cho cả bài toán phân lớp và hồi quy. Dietterich và Bakiri (1991) thử nghiệm phân lớp cho các tập dữ liệu thực và dữ liệu chuẩn trên UCI với 100 bộ phân lớp cơ sở đã giảm 10% lỗi cho dữ liệu thực và 6% đến 42% lỗi cho dữ liệu trên UCI.

10.3.2. Nhặt định hướng

Mục đích của phương pháp nhặt định hướng là để cải tiến độ chính xác của thuật toán phân lớp đã cho. Trong phương pháp này người ta xây dựng các *bộ học yếu*, trong đó tập dữ liệu huấn luyện dùng cho bộ học sau được lấy định hướng dựa trên lỗi của các bộ học trước.

Dưới đây giới thiệu hai kỹ thuật thông dụng của cách tiếp cận này là *boosting* và *adaboost*

Boosting

Đối với bài toán hai lớp, phương pháp này tạo ra 3 bộ phân lớp yếu theo quá trình sau đây.

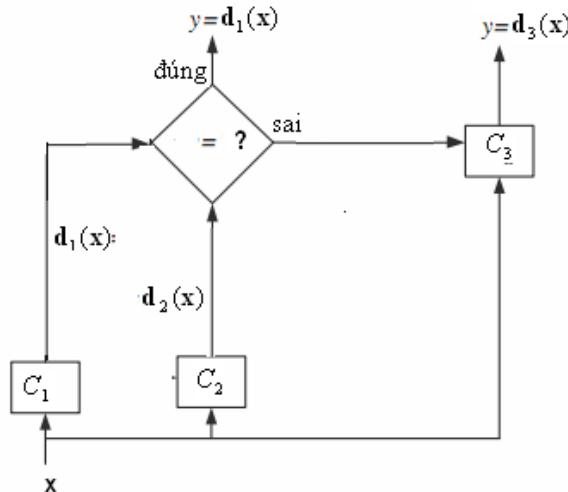
Trước tiên người ta lấy ngẫu nhiên tập D_1 gồm $n_1 (< N)$ mẫu trong tập dữ liệu huấn luyện D . Sau đó huấn luyện bộ phân lớp C_1 từ tập dữ liệu D_1 . C_1 là bộ học yếu, tức là tỷ lệ lỗi cao (nếu độ chính xác cao thì mức độ cải tiến sẽ thấp, tuy nhiên nó có thể có tỷ lệ lỗi thấp trong tập dữ liệu huấn luyện).

Tập dữ liệu huấn luyện D_2 được xây dựng dựa trên thông tin định hướng của bộ phân lớp C_1 , trong đó một nửa được C_1 phân lớp đúng còn một nửa phân lớp sai. Cụ thể như sau: lấy ngẫu nhiên, $r \in (0,1)$ nếu $r < 1/2$ thì lấy từng mẫu một trong D chưa thuộc D_2 cho C_1 đoán nhận cho đến lúc sai thì bổ sung vào D_2 , ngược lại nếu $r \geq 1/2$ thì lấy từng mẫu một còn trong D chưa thuộc D_2 cho C_1 đoán nhận cho đến lúc đúng thì bổ sung vào D_2 . Thủ tục lặp theo cách này cho đến khi D_2 có n_2 dữ liệu gồm một nửa C_1 nhận dạng đúng và một nửa nhận dạng sai. Huấn luyện bằng tập dữ liệu D_2 ta được bộ phân lớp C_2 .

Tiếp theo ta xây dựng tập dữ liệu huấn luyện D_3 gồm các dữ liệu mà C_1 và C_2 không thống nhất. Ta lần lượt lấy mẫu ngẫu nhiên từ D , nếu C_1 và C_2 phân loại không thống nhất thì bổ sung nó vào D_3 , ngược lại thì mẫu này bị bỏ qua. Theo cách này ta xây dựng được D_3 và dùng để huấn luyện bộ phân lớp C_3 .

Bây giờ ba bộ này được kết hợp như sau. Nếu mẫu x mà kết quả đoán nhận của C_1 và C_2 thống nhất thì nhận nhãn chung này, ngược lại, nếu C_1 và C_2 không thống nhất thì dùng nhãn đoán nhận của C_3 .

Ký hiệu $d_i(x)$ là kết quả đoán nhận của bộ học C_i đối với mẫu x thì nhãn y của nó được đoán nhận theo sơ đồ ở hình 10.5.



Hình 10.5. Sơ đồ quyết định sử dụng boosting

Freund và Schapire (1996) đề xuất phương pháp nhặt định hướng thích nghi (Adaboost) cho phép thiết kế tiếp tục các bộ học cho đến khi chất lượng được cải tiến thực sự.

Adaboost

Trong Adaboost, mỗi lần xây dựng tập huấn luyện D_i thì các mẫu \mathbf{x}^j trong D được lấy với xác suất p_j^i . Ban đầu các xác suất này như nhau: $p_j^1 = \frac{1}{N} \forall j = 1, \dots, N$ để xây dựng D_1 và dùng nó để huấn luyện bộ phân lớp C_1 . Sau mỗi bước lặp các xác suất này tăng dần với các mẫu phân lớp sai và giảm dần với mẫu phân lớp đúng. Giả sử ở bước lặp thứ k , ta đã chọn tập đào tạo D_k với phân bố xác suất $P_j^k \forall j$ trong D và huấn luyện được bộ phân lớp C_k , ta tăng xác suất chọn $P_j^{k+1} \forall j$ đối với các mẫu trong D mà C_k phân lớp sai và giảm xác suất chọn đối với các mẫu phân lớp đúng.

Đến nay có nhiều biến thể của Adaboost được sử dụng, một thể hiện của nó được mô tả trong bảng 10.2.

Bảng 10.2. Một biến thể của Adaboost

Bước 1. Khởi tạo $D, k_{\max}, k = 1, p_j^1 = \frac{1}{N} \forall j = 1, \dots, N;$

Bước 2. Lặp đến khi $k = k_{\max}$:

2.1. Tạo ngẫu nhiên D_k từ D theo phân bố $P_j^k \forall j$;

2.2. Huấn luyện C_k nhờ D_k ;

2.3. $E_k \leftarrow$ Dán giá tỷ lệ lỗi đào tạo của C_k trên D ;

2.4. $\alpha_k \leftarrow \frac{1}{2} \ln[(1 - E_k) / E_k];$

2.5. $p_j^{k+1} \leftarrow \frac{p_j^k}{z_k} \times \begin{cases} e^{-\alpha_k} & \mathbf{d}_k(\mathbf{x}^j) = \mathbf{y}^j \\ e^{\alpha_k} & \mathbf{d}_k(\mathbf{x}^j) \neq \mathbf{y}^j \end{cases}$

2.6. $k = k + 1$;

Bước 3. Trả về các $C_k, \alpha_k \forall k = 1, \dots, k_{\max}$

Trong mô tả trên của thuật toán, z_k là các hằng số để chuẩn hóa phân bố xác suất, \mathbf{y}^j là nhãn đúng của \mathbf{x}^j , $\mathbf{d}_i(\mathbf{x})$ là kết quả đoán nhận của bộ học C_i đối với mẫu \mathbf{x} . Kết quả phân lớp cuối cùng của mẫu \mathbf{x} sẽ là:

$$g(\mathbf{x}) = \left[\sum_{k=1}^{k_{\max}} \alpha_k \mathbf{d}_k(\mathbf{x}) \right] \quad (10.3)$$

trong đó tổng sẽ lấy theo từng nhãn lớp, còn ngoặc vuông được hiểu là lấy nhãn lớp có trọng số tổng hợp lớn nhất.

10.3.3. Rừng ngẫu nhiên

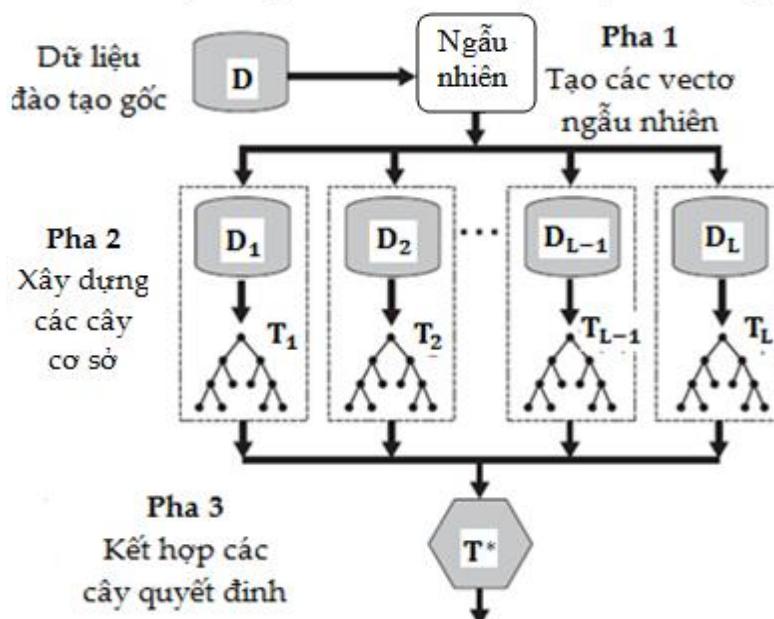
Rừng ngẫu nhiên là phương pháp học tập thể rất thích hợp cho xử lý dữ liệu có số chiều cao, vì vậy đang thu hút nhiều người quan tâm nghiên cứu, áp dụng. Định nghĩa sau đây giải thích tên gọi của phương pháp này.

Định nghĩa 10.1. Rừng ngẫu nhiên là một bộ nhận dạng bao gồm một tập bộ phân lớp cơ sở dạng cây quyết định được kết hợp theo phương thức bỏ phiếu. Các bộ cơ sở được xây dựng từ các tập con dữ liệu với đặc trưng khác nhau được lấy ngẫu nhiên từ tập dữ liệu đào tạo.

Định nghĩa cho thấy thủ tục xây dựng rừng ngẫu nhiên gồm ba pha: tạo dữ liệu (tạo vectơ ngẫu nhiên), xây dựng các cây cơ sở, kết hợp các cây cơ sở theo phương thức bỏ phiếu.

Điểm mới nhất ở đây là pha tạo dữ liệu. Giả sử $D = \{(\mathbf{x}^k, y^k)\}_{k=1}^N, \mathbf{x}^k \in \mathcal{R}^n$ là tập dữ liệu đào tạo với số chiều n lớn, pha tạo vectơ ngẫu nhiên thực hiện như sau.

Chọn trước các số tự nhiên $M (< N)$ và $m (< n)$. Để có mỗi tập dữ liệu $D_k (\subset \mathcal{R}^m)$ cho xây dựng cây quyết định $T_k (k \leq L)$, ta chọn ngẫu nhiên m đặc trưng trong số n đặc trưng của D và lấy ngẫu nhiên M đối tượng từ D rồi chiếu nó lên các đặc trưng được chọn này. Việc chọn đặc trưng và lấy dữ liệu từ D gọi là tạo vectơ ngẫu nhiên. Quá trình xây dựng rừng ngẫu nhiên được mô tả trong hình 10.6.



Hình 10.6. Sơ đồ xây dựng rừng ngẫu nhiên

Chọn số đặc trưng m

Rõ ràng nhiên thường áp dụng cho các bài toán phân lớp hoặc hồi quy có số chiều n lớn, khi đó số đặc trưng m cho mỗi tập dữ liệu cây được chọn nhỏ hơn n nhiều. Breiman gợi ý chọn m như sau:

- Đối với bài toán phân lớp, $m = \lfloor \sqrt{n} \rfloor$, trong đó $\lfloor a \rfloor$ ký hiệu phần nguyên của a .
- Đối với bài toán hồi quy, $m = \lfloor n/3 \rfloor$

Hiện nay việc chọn số đặc trưng m , số đối tượng dữ liệu M cho mỗi tập D_k và cách chọn ngẫu nhiên để lấy các đặc trưng thế nào là tốt cho từng lớp bài toán vẫn đang là chủ đề mở và được nhiều người nghiên cứu. Các cây quyết định được kết hợp song song theo phương pháp bỏ phiếu với nhau ra có trọng số đều cho bởi các công thức (10.1a-b). Thuật toán xây dựng rỗng ngẫu nhiên được mô tả trong bảng 10.3.

Bảng 10.3. Thuật toán xây dựng rỗng ngẫu nhiên cho phân lớp hoặc hồi quy

Bước 1. Với $k=1$, đến L thực hiện:

- 1.1. Lấy ngẫu nhiên m đặc trưng $\{A_{i1}^k, \dots, A_{im}^k\}$ của D ;
- 1.2. Lấy ngẫu nhiên tập R_k gồm M dữ liệu trong D ;
- 1.3. $D_k =$ Hình chiếu của R_k lên các đặc trưng $\{A_{i1}^k, \dots, A_{im}^k\}$;
- 1.4. Xây dựng cây T_k từ tập D_k ; có bộ nhận dạng c_k

Bước 2. Kết hợp bỏ phiếu trọn số đều $\{C_k\}_{k=1}^L$

Đầu ra của hệ cho đối tượng x sẽ là:

$$\bullet \text{ Đối với bài toán hồi quy: } C(x) = \frac{1}{L} \sum_{k=1}^L C_k(x) \quad (10.4a)$$

$$\bullet \text{ Đối với bài toán phân lớp: } C(x) = \left[\frac{1}{L} \sum_{k=1}^L C_k(x) \right], \quad (10.4b)$$

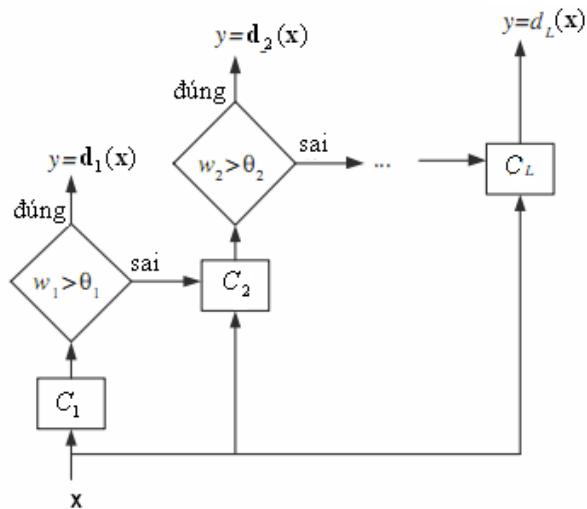
ngoặc vuông chỉ quyết định theo đa số.

10.4. KIẾN TRÚC BẬC THANG

Trong mục 10.3, ta đã xét các bộ phân lớp cơ sở tạo nên nhò cùng một thuật toán huấn luyện nhưng dùng các tập dữ liệu khác nhau để huấn luyện. Trong thực tế, có thể dùng các thuật toán và cách trích chọn đặc trưng khác nhau để xây dựng các bộ nhận dạng cơ sở. Khi đó các bộ phân lớp này thường có hiệu quả và thời

gian chạy khác nhau. Thông thường thì bộ phân lớp có độ chính xác cao sẽ tồn tại nhiều thời gian chạy hơn. Trong trường hợp đó, kiến trúc bậc thang là phương pháp hiệu quả để kết hợp các bộ phân lớp này.

Ta xét L bộ phân lớp, được sắp theo thứ tự tăng dần về độ chính xác: C_1, C_2, \dots, C_L . Mỗi bộ phân lớp C_i có hàm đánh giá độ tin cậy w_i và ngưỡng tin cậy θ_i sao cho với mẫu x mà $w_i > \theta_i$ thì nhãn $d_i(x)$ hầu chắc chắn đúng (độ chính xác cao). Khi đó các bộ phân lớp này có thể kết hợp theo kiến trúc bậc thang như trong hình 10.7. Các mẫu được đưa tuần tự cho từng bộ phân lớp bắt đầu từ C_1 , nếu bộ phân lớp C_k đang xét dự đoán nhãn x với độ tin cậy lớn hơn ngưỡng θ_k thì lấy nhãn $d_k(x)$ cho nó, ngược lại thì đưa cho bộ C_{k+1} nhận, cuối cùng thì dành cho bộ C_L . Bộ phân lớp kết hợp theo kiến trúc này thường có độ chính xác cao hơn các bộ thành phần và thời gian chạy trung bình để xử lý một mẫu cũng ít hơn.



Hình 10.7. Kiến trúc bậc thang của L bộ phân lớp

KẾT LUẬN

Trong học có giám sát để xây dựng bộ phân lớp hoặc hồi quy, nếu dùng một thuật toán và dùng một tập dữ liệu đào tạo thì ta chỉ được một bộ nhận dạng yếu, tức là sai số lớn. Học tập thể là một phương pháp để tăng độ chính xác của bộ nhận dạng. Trong cách tiếp cận này, người ta xây dựng các bộ nhận dạng cơ sở theo các phương thức: 1) dùng các thuật toán huấn luyện khác nhau. 2) dùng một thuật toán nhưng sử dụng các tập dữ liệu đào tạo hay tham số khác nhau, 3) dùng một thuật toán nhưng dùng tập dữ liệu với tập đặc trưng khác nhau. 4) kết hợp các phương thức trên.

Bỏ phiếu là phương pháp kết hợp các bộ nhận dạng đơn giản và thông dụng nhất. Trong sơ đồ này, các bộ nhận dạng cơ sở được kết nối song song và đầu ra tổng

hợp thường dùng nhất là dạng tuyến tính, có thể lấy trọng số đều. Phương pháp bỏ phiếu không chỉ tăng độ chính xác mà trong các bài toán hồi quy, nó còn cho kết quả có phương sai thấp hơn các bộ thành phần.

Bagging là phương pháp đơn giản nhất để xây dựng bộ học tập thế, trong đó các bộ nhận dạng thành phần được xây dựng nhờ dùng cùng một thuật toán nhưng dùng các tập dữ liệu đào tạo khác nhau được lấy ngẫu nhiên có hoàn lại các tập dữ liệu gốc.

Các kỹ thuật nhặt theo hướng bao gồm boosting và adaboost khác với bagging ở chỗ phân bố xác suất lấy dữ liệu thay đổi theo thứ tự lấy tập dữ liệu đào tạo của bộ nhận dạng cơ sở. Phương thức kết hợp các bộ cơ sở trong kỹ thuật boosting không theo kiến trúc song song, nhò đó tiết kiệm thời gian chạy hơn.

Rừng ngẫu nhiên là phương pháp thích hợp cho các bài toán phân lớp hoặc hồi quy với dữ liệu có số chiều cao. Trong đó mỗi bộ nhận dạng cơ sở là một cây quyết định, được xây dựng từ tập dữ liệu có số chiều nhỏ nhò chiều một tập dữ liệu lấy ngẫu nhiên từ tập dữ liệu đào tạo gốc lên tập đặc trưng được chọn ngẫu nhiên.

Khi các bộ phân lớp được xây dựng nhò dùng các thuật toán khác nhau, có độ chính xác và thời gian chạy khác nhau thì có thể dùng kiến trúc bậc thang để tăng độ chính xác và giảm thời gian chạy trung bình so với bộ nhận dạng thành phần có độ chính xác cao nhất nhưng thời gian chạy cũng lâu nhất.

BÀI TẬP

1. Giả sử mỗi bộ phân lớp cơ sở đều có xác suất đúng $p > 1/2$, hãy ước lượng xác suất đúng khi bỏ phiếu theo đa số của L bộ phân lớp.
2. Hãy đề xuất một lược đồ boosting cho bài toán nhiều lớp.
3. Tìm một lược đồ điều chỉnh xác suất chọn tập đào tạo theo adaboost khác với lược đồ ở mục 10.3.
4. Hãy đề xuất một cách chọn đặc trưng để tạo các tập dữ liệu cho cây quyết định của phương pháp rừng ngẫu nhiên. Giải thích vì sao cách chọn này tốt hơn chọn phân bố đều.
5. Tại sao trong kiến trúc bậc thang ta xếp các bộ phân lớp tăng dần về độ chính xác?

TÀI LIỆU THAM KHẢO

1. E. Alpaydin, *Introduction to Machine Learning*, Massachusetts Institute of Technology, Second Edition, 2010
2. T. Mitchell, *Machine learning*, McGraw-Hill, 1997
3. N.J Nielsson, *Introduction to Machine Learning, Robotic Laboratory*, Stanford University, 1997
4. K.P. Murphy, *Machine Learning*, MIT Press, 2012
5. T. Hastie, R.Tibshirani and J. Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction* (2nd edition), Springer, 2009
6. C.M. Bishop, *Pattern Recognition and Machine Learning*, Springer, 2006
7. G. James, D. Witten, T. Hastie and R.t Tibshirani, *An Introduction to Statistical Learning with Applications in R*, Springer 2013
8. R.O. Duda et al, *Pattern Clasification*, John Wiley& Sons, Inc, 2001
9. Hoàng Xuân Huân, *Giáo trình nhận dạng mẫu*, NXB ĐHQG HN, 2012
10. J. P. Masques de Sá, *Pattern Recognition - Concepts, Methods and applications*. Springer, 2001
11. S. Theodoridis, *Pattern Recognition*, Elssevier Academic Press (4th edition), 2009
12. A.R. Webb and K.D. Copsey, *Statistical Pattern Recognition*, John Wiley& Sons, Inc. (3rd edition), 2011
13. H.B. Demuth and M. Beale, *Neural Network Design*, PWS Publishing company, 1996
14. S. Haykin, *Neural Network and Machine Learning* ,Pearson International Edition (3rd edition), 2009
15. C. G. Looney, *Pattern Recognition using Neural Networks: Theory and Algorithms for Engineers and Scientists*, Oxford University Press, New York,1997.
16. M. Dorigo and T.Stutzle: *Ant Colony Optimization*. MIT Press, Cambridge, 2004.

BẢNG CHỮ VIẾT TẮT

ACO	Tối ưu đòn kiến
AIS	Hệ miễn dịch nhân tạo
BP	Lan truyền ngược
CART	Cây phân lớp và hồi quy
CRB	Lập luận dựa trên tình huống
DNA	Deoxyribonucleic acid
EEG	Điện não đồ
FPR	Tỷ lệ chấp nhận sai
FNR	Tỷ lệ bác bỏ sai
GA	Thuật toán di truyền
K-NN	K lảng giềng gần nhất
ĐTCT	Đặc trưng chi tiết
MAP	Xác suất hậu nghiệm cực đại
MDP	Quá trình quyết định Markov
ML	Có khả năng nhất
MLP	Perceptron nhiều tầng
PAC	Gần đúng xác suất
PCA	Phân tích thành phần chính
RBF	Hàm cơ sở bán kính
RNA	Ribonucleic acid
SSE	Tổng bình phương sai số
SVM	Máy vectơ tựa
TSP	Bài toán người chào hàng
VC	Vapnik-Chervonekis