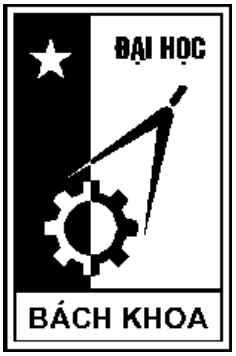
TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI

VIỆN TOÁN ỨNG DỤNG VÀ TIN HỌC



MÔ HÌNH PHÂN LOẠI
TRONG HỆ HỖ TRỢ QUYẾT ĐỊNH

ĐỒ ÁN III

Chuyên ngành : TOÁN TIN

Chuyên sâu : Tin học

Giảng viên hướng dẫn: TS. NGUYỄN THỊ THANH HUYỀN

Sinh viên thực hiện: VŨ MẠNH QUANG

Lớp: Toán Tin K61

Hà Nội – 2020

NHẬN XÉT CỦA GIẢNG VIÊN HƯỚNG DẪN

1. Mục đích và nội dung của đồ án:

2. Kết quả đạt được:

3. Ý thức làm việc của sinh viên:

Hà Nội, ngày tháng năm

Giảng viên hướng dẫn

(Ký và ghi rõ họ tên)

Nội dung

Lời nói đầu

Ngày nay, việc đưa ra các quyết định đã được hỗ trợ rất nhiều bởi các phần mềm, các hệ thống hỗ trợ quyết định. Trong hệ thống hỗ trợ ra quyết định, mô hình là một thành phần quan trọng, kết quả đưa ra của mô hình giúp cho người ra quyết định có những quyết định tốt và chính xác hơn. Có nhiều loại mô hình được sử dụng như mô hình dự đoán, mô hình nhận dạng và học tập, mô hình tối ưu hóa, mô hình phân tích rủi ro, mô hình dòng chờ. Mô hình phân loại thuộc lớp mô hình dự đoán, mô hình nhận dạng và học tập.

Mô hình phân loại được sử dụng rất phổ biến trong các hệ thống hỗ trợ ra quyết định, chúng rút ra các kết luận từ dữ liệu quan sát và phân chúng vào những lớp mục tiêu đã biết, người ra quyết định dựa vào các kết quả phân loại đó để đưa ra các quyết định có lợi. Trong đồ án này, em tập trung nghiên cứu về một số mô hình phân loại phổ biến.

Bố cục đồ án gồm ba chương chính như sau:

- Chương 1: Hệ hỗ trợ quyết định
- Chương 2: Mô hình phân loại
- Chương 3: Cài đặt thử nghiệm mô hình hồi quy logistic

Để hoàn thành được đồ án: “Mô hình phân loại trong hệ hỗ trợ quyết định”, lời đầu tiên em xin chân thành cảm ơn giảng viên hướng dẫn TS. Nguyễn Thị Thanh Huyền đã tận tình hướng dẫn em trong suốt thời gian thực hiện.

Em cũng xin chân thành cảm ơn các thầy cô trong viện Toán ứng dụng và Tin học Trường Đại học Bách Khoa Hà Nội đã truyền đạt những kiến thức, kỹ năng, kinh nghiệm nền tảng để giúp em hoàn thành đồ án môn học này

Tuy đã có những cố gắng nhất định, tìm hiểu và tiếp cận với đề tài nhưng do trình độ và thời gian hạn chế nên đồ án này không tránh khỏi thiếu sót. Rất mong nhận được những nhận xét góp ý và sửa sai của thầy giáo hướng dẫn, các thầy cô và các bạn để quyển đồ án này hoàn thiện hơn

Em xin chân thành cảm ơn!

Hà Nội, ngày tháng năm

Sinh viên thực hiện

Vũ Mạnh Quang

Chương 1: Hệ hỗ trợ quyết định

1.1. Khái niệm hệ hỗ trợ quyết đinh

Ngày này, môi trường hoạt động của các tổ chức ngày càng trở nên phức tạp, tạo ra các cơ hội kèm theo đó là các vấn đề phát sinh chẳng hạn như toàn cầu hóa. Một số yếu tố môi trường kinh doanh như:

- Thị trường cạnh tranh ngày càng mạnh mẽ, mở rộng toàn cầu kèm đó là sự nở rộ của internet và sự hỗ trợ của công nghệ thông tin
- Nhu cầu của người tiêu dùng: mong muốn chất lượng, sự đa dạng, tốc độ giao hàng, khách hàng ít trung thành
- Công nghệ: nhiều đổi mới, sản phẩm mới, dịch vụ mới, nhanh chóng lỗi thời, mạng xã hội phát triển và sự quá tải thông tin
- Xã hội: quy định của chính phủ, lực lượng lao động đa dạng, an ninh và khủng bố,…

Các tổ chức cần kịp thời phản ứng, dự đoán, thích ứng và chủ động với các yếu tố môi trường. Người quản lý cần có các hành động như lập kế hoạch chiến lược, sử dụng mô hình kinh doanh mới sáng tạo, cải thiện hệ thống thông tin doanh nghiệp, dịch vụ khách hàng, … tiến tới ra quyết định quản lý. Quản lý có thể xem là ra quyết định, ra quyết định là việc lựa chọn giải pháp tốt nhất từ hai hay nhiều lựa chọn.

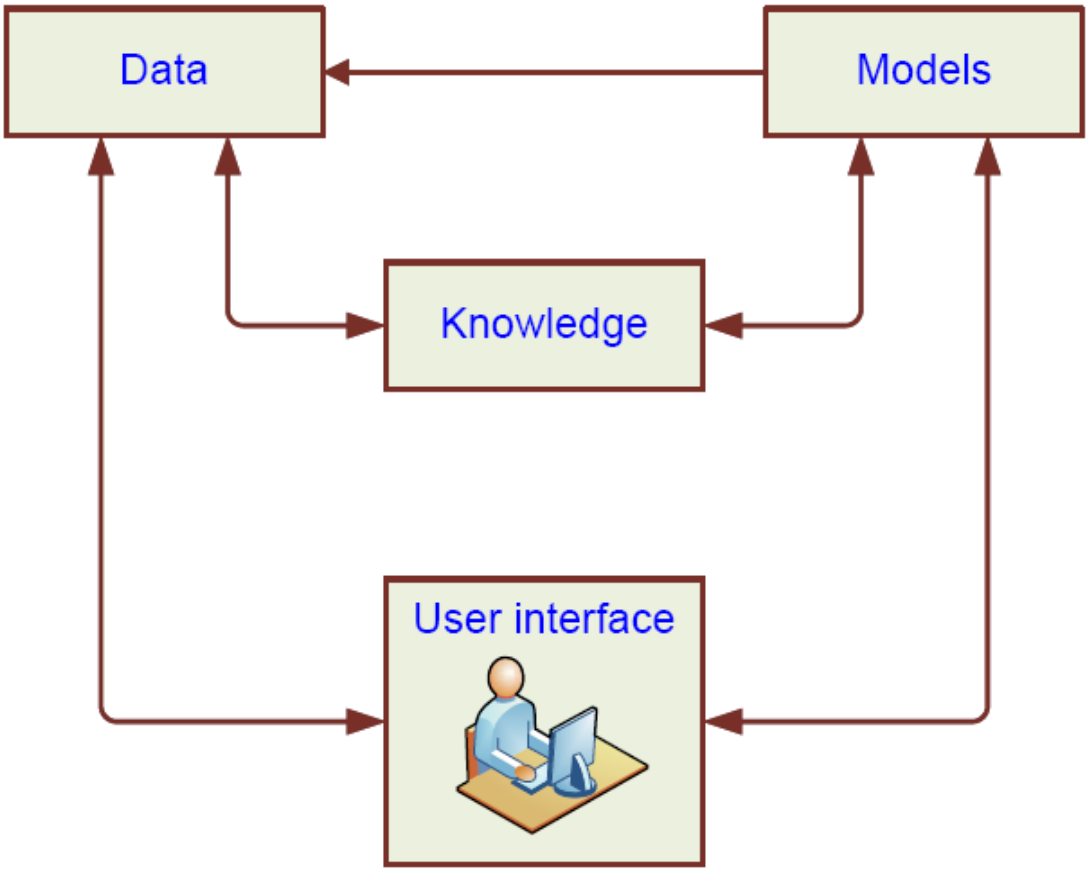
Nhà quản lý thường đưa ra quyết định theo quy trình: xác định vấn đề => xây dựng mô hình mô tả vấn đề trong thế giới thực => xác định các giải pháp cho vấn đề được mô hình hóa và đánh giá các giải pháp => so sánh, lựa chọn và đề xuất giải pháp tiềm năng cho vấn đề. Trên thực tế, việc ra quyết định là một vấn đề khó: công nghệ, hệ thống thông tin, công cụ tìm kiếm tiên tiến và toàn cầu hóa dẫn đến ngày càng có nhiều lựa chọn trong việc sử dụng công cụ để đưa ra quyết định; các quy định của chính phủ yêu cầu cần phải tuân thủ, bất ổn chính trị và khủng bố, cạnh tranh và thay đổi nhu cầu của người tiêu dùng gây ra nhiều yếu tố bất ổn, tạo ra sự khó khăn trong việc dự đoán hậu quả và tương lai; một số yếu tố khác như cần đưa ra quyết định

nhanh chóng, những thay đổi thường xuyên và không thể đoán trước dẫn đến việc học tập thử và sai trở nên khó khăn, và chi phí cho nhưng sai lầm.

Khái niệm cổ điển của hệ hỗ trợ quyết định:

- Năm 1971, Gorry and Scott-Morton lần đầu tiên đưa ra định nghĩa về hệ thống hỗ trợ quyết định: hệ thống dựa trên sự tương tác với máy tính, hỗ trợ những người ra quyết định sử dụng dữ liệu và mô hình để giải quyết các vấn đề phi cấu trúc.
- Một định nghĩa khác được cung cấp bởi Keen and Scott-Morton năm 1978: hệ thống hỗ trợ quyết định kết hợp các nguồn lực trí tuệ của cá nhân với khả năng của máy tính để cải thiện chất lượng của các quyết định. Nó là một hỗ trợ dựa trên hệ thống máy tính cho những người quản lý ra quyết định đối phó với vấn đề bán cấu trúc.
- Trên thực tế, DSS (hệ thống hỗ trợ quyết định) có thể xem như là một thuật ngữ rộng, tuy nhiên một số người xem DSS hẹp hơn, ứng dụng hỗ trợ quyết định cụ thể. Thuật ngữ DSS có thể được sử dụng như một thuật ngữ mô tả bất kỳ hệ thống máy tính nào hỗ trợ ra quyết định trong một tổ chức.

DSS là một ứng dụng cụ thể. Theo nghĩa hẹp, DSS đề cập đến một quá trình xây dựng các ứng dụng tùy biến cho các vấn đề phi cấu trúc hoặc bán cấu trúc. Các thành phần của kiến trúc DSS: dữ liệu, mô hình, kiến thức / thông tin, người dùng, giao diện. DSS thường được tạo ra bằng cách kết hợp các thành phần này.



Hình 1.1: Kiến trúc của một hệ thống hỗ trợ quyết định

1.2. Ra quyết định

1.2.1. Định nghĩa ra quyết định

Ra quyết định là một quá trình lựa chọn giữa hai hoặc nhiều hành động dùng cho mục đích đạt được một hoặc nhiều mục tiêu. Theo Simon (1977), việc ra quyết định quản lý đồng nghĩa với toàn bộ quy trình quản lý. Hãy xem xét chức năng quản lý quan trọng của kế hoạch. Lập kế hoạch liên quan đến một loạt các quyết định: Nên làm gì? Khi nào? Ở đâu? Tại sao? Làm sao? Bởi ai? Người quản lý đặt mục tiêu, hoặc kế hoạch; do đó, lập kế hoạch ngụ ý ra quyết định. Các chức năng quản lý khác, như tổ chức và kiểm soát, cũng liên quan đến việc ra quyết định.

Một vấn đề xảy ra khi một hệ thống không đạt được mục tiêu đề ra, không mang lại kết quả dự đoán mong muốn hay không hoạt động theo đúng kế hoạch. Vấn đề là sự khác biệt giữa kết quả mong muốn và thực tế đạt được. Giải quyết vấn đề

cũng liên quan đến việc xác định những cơ hội mới. Ra quyết định là một phần của chủ đề rộng hơn thường được gọi là giải quyết vấn đề.

1.2.2. Đặc điểm của việc ra quyết định

Việc ra quyết định có thể bị ảnh hưởng bởi một số đặc điểm sau:

- Ra quyết định theo nhóm: Để đưa đến một quyết định nào đấy thì phải có sự bàn bạc kĩ lưỡng của một nhóm các chuyên gia phụ trách các vấn đề,… và quyết định được đưa ra của ban lãnh đạo dựa trên cơ sở tiếp thu, phân tích, đánh giá các ý kiến của các chuyên gia đấy.
- Người ra quyết định quan tâm đến việc đánh giá các kịch bản giả định Nếu–Thì: Nếu tôi làm như thế thì sẽ thế nào, nếu tôi không làm thế thì sẽ thế nào?
- Thử nghiệm trên các hệ thống thực để xem xét tính hiệu quả (ví dụ: phát triển lịch biểu, thử nó và xem cách nó hoạt động tốt hay không tốt)
- Thay đổi trong môi trường ra quyết định có thể diễn ra liên tục và quá trình ra quyết định luôn phải cân nhắc những thay đổi đó (ví dụ: giao hàng vào khoảng thời gian nghỉ lễ có thể tăng, đòi hỏi một cái nhìn khác về vấn đề)
- Thay đổi trong môi trường ra quyết định có thể ảnh hưởng đến chất lượng quyết định bằng cách gây áp lực thời gian cho người ra quyết định.
- Thu thập thông tin và phân tích một vấn đề cần có thời gian và có thể tốn kém. Nó rất khó để xác định khi nào nên dừng lại và đưa ra quyết định. Một quyết định nhanh chóng có thể thiếu chính xác
- Có thể không có đủ thông tin để đưa ra quyết định thông minh
- Quá nhiều thông tin có thể có sẵn (nghĩa là quá tải thông tin)
- Quyết định tốt hơn có thể cần đánh đổi độ chính xác và thời gian
- Quyết định nhanh chóng có thế có hại do không kịp cân nhắc vấn đề kĩ lưỡng
- Một số vấn đề chịu ảnh hưởng từ quyết định nhanh chóng: vấn đề nguồn nhân lực (27%), ngân sách/tài chính (24%), cơ cấu tổ chức (22%), chất lượng/năng suất (20%), lựa chọn và cài đặt các giải pháp công nghệ thông tin (17%), cải tiến quy trình (17%).

Để xác định cách những người ra quyết định đưa ra quyết định của mình, trước tiên chúng ta phải hiểu quá trình và các vấn đề quan trọng liên quan đến việc ra quyết định. Sau đó chúng ta có thể tìm hiểu phương pháp thích hợp để hỗ trợ người ra quyết định. Sau đó chúng ta mới có thể phát triển DSS để giúp những người ra quyết định.

1.2.3. Phong cách ra quyết định

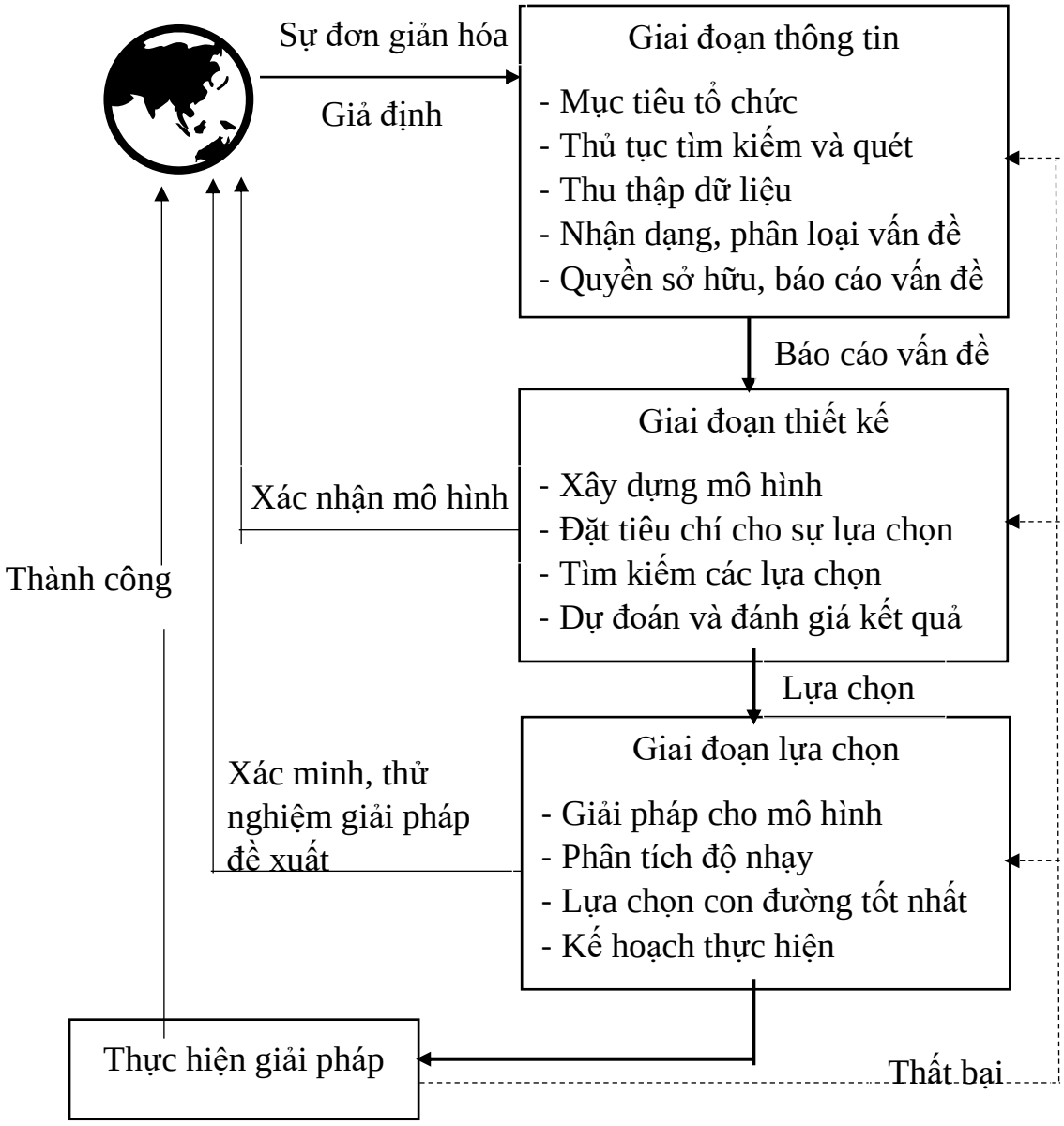
Phong cách ra quyết định là các mà những người ra quyết định suy nghĩ và phản ứng vấn đề. Điều này bao gồm cách họ nhìn nhận vấn đề, phản ứng nhận thức của họ, và giá trị, niềm tin khác nhau như thế nào từ cá nhân này đến cá nhân khác và từ tình huống đến tình hình. Kết quả là, mọi người đưa ra quyết định theo những cách khác nhau. Các nhà nghiên cứu đã xác định một số phong cách ra quyết định, chúng bao gồm:

- Phân tích và heuristic
- Độc đoán và dân chủ
- Tư vấn (cá nhân hoặc nhóm)

Tất nhiên, có thể có nhiều kết hợp và biến thể của phong cách chẳng hạn như một người có thể phân tích, chuyên quyền hoặc tư vấn (với các cá nhân) và heuristic. Để một hệ thống máy tính hỗ trợ thành công cho người quản lý, nó phải phù hợp với tình huống quyết định cũng như phong cách quyết định. Do đó, hệ thống phải linh hoạt và thích ứng với những người dùng khác nhau. Nếu một DSS là để hỗ trợ các phong cách, kỹ năng khác nhau, và kiến thức, nó không nên cố gắng thực thi một quy trình cụ thể. Thay vào đó, nó sẽ giúp người ra quyết định sử dụng và phát triển phong cách, kỹ năng và kiến thức của riêng họ.

1.3. Quá trình ra quyết định

Simon (1977) nói rằng quá trình ra quyết định bao gồm ba giai đoạn chính: thông tin, thiết kế, và lựa chọn. Sau đó ông đã thêm một giai đoạn thứ tư: thực hiện.



Hình 1.2: Quá trình ra quyết định / mô hình hóa

1.3.1. Giai đoạn thông tin

Quá trình ra quyết định bắt đầu với giai đoạn thông tin. trong giai đoạn này, người ra quyết định xem xét thực tế và xác định vấn đề. Thông tin trong quá trình ra quyết định liên quan đến việc xem xét chi tiết môi trường liên tục hay không liên tục. Nó bao gồm một số hoạt động xác định các vấn đề, hoặc những cơ hội. Giai đoạn thông tin bắt đầu bằng việc xác định các mục tiêu của tổ chức liên quan đến một vấn đề quan tâm (ví dụ: quản lý hàng tồn kho, lựa chọn công việc) và xác định xem chúng

có được đáp ứng hay không. Vấn đề xảy ra vì không hài lòng với hiện trạng. Sự không hài lòng là kết quả của sự khác biệt giữa những gì mọi người mong muốn và những gì đang xảy ra. Trong giai đoạn đầu tiên này, một người ra quyết định cố gắng để xác định xem một vấn đề có tồn tại hay không, xác định các triệu chứng, xác định cường độ và xác định rõ ràng vấn đề.

Thực tế vấn đề thường phức tạp bởi nhiều yếu tố liên quan đến nhau, đôi khi rất khó để phân biệt giữa các triệu chứng và các vấn đề thực sự. Sự tồn tại của một vấn đề có thể được xác định bằng cách theo dõi và phân tích mức năng suất của tổ chức. Đo lường năng suất và xây dựng của một mô hình được dựa trên dữ liệu thực. Việc thu thập dữ liệu và ước tính dữ liệu trong tương lai là một trong những bước khó khăn nhất trong phân tích. Sau đây là một số vấn đề có thể phát sinh trong quá trình thu thập và ước tính dữ liệu và do đó, người ra quyết định có thể có nhiều phiền phức:

- Dữ liệu không có sẵn
- Lấy dữ liệu có thể tốn kém
- Dữ liệu có thể không đủ chính xác
- Ước tính dữ liệu thường chủ quan
- Dữ liệu có thể không an toàn
- Có thể có quá nhiều dữ liệu (quá tải thông tin)
- Dữ liệu phụ thuộc thời gian

Khi điều tra sơ bộ được hoàn thành, có thể xác định xem một vấn đề có thực sự tồn tại, nó nằm ở đâu và tầm quan trọng của nó. Vấn đề thường được phân loại theo cấu trúc, nhiều vấn đề phức tạp có thể được chia thành các bài toán con, giải quyết các bài toán con đơn giản có thể giúp giải quyết một vấn đề phức tạp. Giai đoạn thông tin kết thúc với một tuyên bố vấn đề chính thức.

1.3.2. Giai đoạn thiết kế

Trong giai đoạn thiết kế, một mô hình đại diện cho hệ thống được xây dựng. Điều này được thực hiện bằng cách đưa ra các giả định đơn giản hóa thực tế và viết

ra các mối quan hệ giữa tất cả các biến. Giai đoạn thiết kế bao gồm tìm kiếm hoặc phát triển và phân tích tiến trình của hành động, chúng bao gồm sự hiểu biết vấn đề và nghiên cứu các giải pháp có thể thực hiện được. Một mô hình của vấn đề ra quyết định được xây dựng, kiểm tra và xác nhận. Trước tiên chúng ta hay xác định một mô hình.

Một đặc điểm chính của DSS là thường bao gồm ít nhất một mô hình. Ý tưởng cơ bản là thực hiện phân tích DSS trên một mô hình thực tế chứ không phải trên hệ thống thực. Một mô hình là một đại diện đơn giản hoặc trừu tượng của thực tế. Nó thường được đơn giản hóa vì thực tế quá phức tạp để mô tả chính xác và phần lớn sự phức tạp thực sự không liên quan trong việc giải quyết một vấn đề cụ thể.

Sự phức tạp của các mối quan hệ trong nhiều hệ thống tổ chức được mô tả bằng toán học. Hầu hết các phân tích DSS được thực hiện bằng số với toán học hoặc các mô hình định lượng khác.

Mô hình hóa: khái niệm hóa một vấn đề và trừu tượng nó thành dạng định lượng hoặc định tính (sử dụng ký hiệu/biến). Trừu tượng hóa: đưa ra các giả định để đơn giản hóa, mô hình hóa như một sự kết hợp giữa khoa học và nghệ thuật.

Nguyên tắc lựa chọn là một tiêu chí mô tả khả năng chấp nhận một cách tiếp cận giải pháp, trong một mô hình, nó là một biến kết quả. Chọn một nguyên tắc lựa chọn phản ánh mục tiêu ra quyết định. Điều quan trọng là phải nhận ra sự khác biệt giữa tiêu chí và ràng buộc, tiêu chí không phải là một ràng buộc.

Mô hình chuẩn (tối ưu hóa) là mô hình trong đó lựa chọn được cho là tốt nhất trong tất cả các lựa chọn có thể thay thế. Lý thuyết quyết định chuẩn được dựa trên các giả định sau đây của người ra quyết định:

- Con người luôn muốn tối đa hóa mục tiêu.
- Đối với một tình huống ra quyết định, tất cả các lựa chọn của hành động và kết quả là được biết đến.
- Người ra quyết định có một thứ bậc hoặc sự ưu tiên cho phép họ xếp hạng mức độ mong muốn của tất cả các kết quả.

Mô hình heuristic (suboptimization) là mô hình trong đó lựa chọn tốt nhất của một tập con các lựa chọn có thể thay thế. Thông thường, không thể tối ưu hóa các vấn đề thực tế (kích thước/độ phức tạp). Suboptimization cũng có thể giúp giảm các giả định không thực tế trong các mô hình, giúp đạt được một giải pháp đủ tốt nhanh hơn. Đủ tốt, hoặc thỏa mãn: một cái gì đó ít hơn những gì tốt nhất, một hình thức của suboptimization, tìm kiếm để đạt được một mức hiệu suất mong muốn nhưng không phải là tốt nhất, giúp tiết kiệm thời gian.

Mô hình mô tả mô tả mọi thứ chúng là hay chúng được cho rằng (dựa trên toán học). Mô hình mô tả là cực kỳ hữu ích, chúng không cung cấp giải pháp nhưng thông tin dẫn đến giải pháp. Mô phỏng - phương pháp mô hình mô tả phổ biến nhất (mô tả toán học của các hệ thống trong môi trường máy tính). Việc đo lường/xếp hạng kết quả sử dụng nguyên tắc lựa chọn. Rủi ro có thể xảy ra do thiếu kiến thức chính xác và có thể đo lường bởi xác suất. Kịch bản (trường hợp nếu-thì) một tuyên bố của các giả định về hoạt động của môi trường (biến) của một hệ thống cụ thể tại một thời điểm nhất định. Kịch bản có thể xảy ra: tốt nhất, tệ nhất, có thể tốt, trung bình.

1.3.3. Giai đoạn lựa chọn

Lựa chọn là hành động quan trọng của việc ra quyết định. Giai đoạn lựa chọn là giai đoạn quyết định thực tế và cam kết tuân theo một số hành động nhất định được đưa ra. Ranh giới giữa các giai đoạn thiết kế và lựa chọn thường không rõ ràng vì một số hoạt động nhất định có thể được thực hiện trong cả hai (chồng chéo một phần) và bởi vì người ra quyết định có thể trở lại thường xuyên từ các hoạt động lựa chọn đến các hoạt động thiết kế (ví dụ: tạo ra các lựa chọn mới trong khi thực hiện đánh giá những cái hiện có). Giai đoạn lựa chọn bao gồm tìm kiếm, đánh giá và đề xuất một giải pháp thích hợp cho mô hình. Một giải pháp cho một mô hình là một tập hợp các giá trị cụ thể cho các biến quyết định trong một lựa chọn. Việc giải quyết một mô hình không giống như giải quyết vấn đề mà mô hình đại diện. Giải pháp cho mô hình mang lại một giải pháp được đề xuất cho vấn đề. Vấn đề chỉ xem xét được giải quyết nếu giải pháp được đề xuất được thực hiện thành công. Giải quyết một mô hình ra

quyết định liên quan đến việc tìm kiếm một hướng đi phù hợp của hành động. Phương pháp tìm kiếm bao gồm các kỹ thuật phân tích (giải bằng một công thức), các thuật toán (các thủ tục từng bước), heuristic (quy tắc tự đặt ra) và tìm kiếm mù (tìm kiếm thực sự ngẫu nhiên). Một số hoạt động bổ sung như phân tích độ nhạy, phân tích nếu-thì, tìm kiếm mục tiêu.

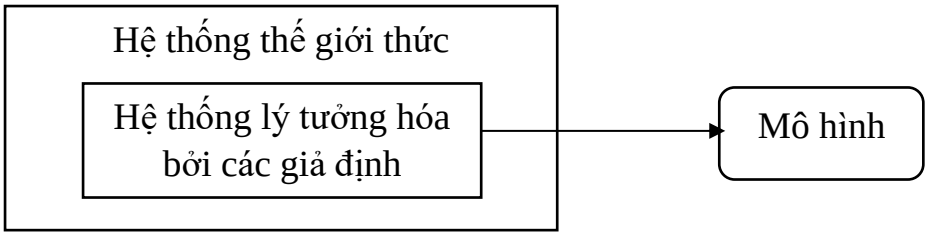
1.3.4. Giai đoạn thực hiện

Trên thực tế, việc thực hiện một giải pháp được đề xuất cho một vấn đề là sự khởi đầu của một trật tự mới của sự vật hoặc là bước đầu của sự thay đổi. Và thay đổi phải được quản lý. Sự mong đợi của người dùng phải được quản lý như một phần của sự thay đổi quản lý. Giải pháp cho một vấn đề = thay đổi. Việc thực hiện liên quan đến việc đưa ra một giải pháp được đề xuất làm việc.

1.4. Mô hình toán học để ra quyết định

1.4.1. Cấu trúc mô hình toán học

Nói chung, một mô hình là một sự trừu tượng có chọn lọc của một hệ thống thực.



Hình 1.3: Mô hình là một sự trừu tượng hóa có chọn lọc của thực tế

Theo đặc điểm, các mô hình có thể được chia thành: *biểu tượng, tương tự, tượng trưng*. Một mô hình mang tính *biểu tượng* là một đại diện vật chất của một hệ thống thực, có hành vi được bắt chước cho mục đích phân tích. Một mô hình thu nhỏ của một khu phố mới là một ví dụ về mô hình mang tính biểu tượng. Một mô hình *tương tự* cũng là một đại diện vật chất, mặc dù nó bắt chước hành vi thực tế bằng cách tương tự. Một đường hầm gió được xây dựng để điều tra các đặc tính khí động học của xe cơ giới là một ví dụ của một mô hình tương tự nhằm thể hiện sự tiến bộ thực tế của một chiếc xe trên đường. Một mô hình *tượng trưng*, chẳng hạn như một

mô hình toán học, là một đại diện trừu tượng của một hệ thống thực sự. Nó được dự định để mô tả hành vi của hệ thống thông qua một loạt các biến tượng trưng, tham số và các mối quan hệ toán học.

Một sự khác biệt khác có liên quan đến bản chất xác suất của mô hình có thể là *ngẫu nhiên* và *xác định*. Trong mô hình *ngẫu nhiên*, một số thông tin đầu vào biểu thị ngẫu nhiên các sự kiện và do đó được đặc trưng bởi một phân phối xác suất. Một mô hình được gọi là *xác định* khi tất cả dữ liệu đầu vào được cho là được biết đến trước đó và chắc chắn. Vì giả định này hiếm khi được thực hiện trong các hệ thống thực, người ta sử dụng các mô hình xác định khi có vấn đề là đủ phức tạp và bất kỳ yếu tố ngẫu nhiên có sự giới hạn.

Một sự khác biệt nữa liên quan đến thời gian trong mô hình toán học, có thể là *tĩnh* hoặc *động*. Các mô hình *tĩnh* xem xét một hệ thống nhất định và việc ra quyết định liên quan quá trình trong một giai đoạn thời gian duy nhất. Các mô hình *động* xem xét một hệ thống nhất định thông qua một số các giai đoạn thời gian, tương ứng với một chuỗi các quyết định.

1.4.2. Các lớp mô hình

Có một số lớp mô hình toán học để ra quyết định, trong đó lần lượt có thể được giải quyết bằng một số kỹ thuật giải pháp thay thế. Mỗi lớp mô hình phù hợp để đại diện cho một số loại quy trình ra quyết định. Các loại chính của các mô hình toán học cho ra quyết định, bao gồm:

- Mô hình dự đoán: là một mô hình được sử dụng khá thường xuyên trong một hệ hỗ trợ quyết định, yêu cầu dữ liệu đầu vào liên quan tới các sự kiện trong tương lai. Dự đoán cho phép thông tin đầu vào được đưa vào các quy trình ra quyết định khác nhau, nghiên cứu và phát triển, quản trị và kiểm soát, tiếp thị, sản xuất và hậu cần. Về cơ bản, tất cả các chức năng bộ phận của một doanh nghiệp sử dụng một số thông tin dự đoán để phát triển việc ra quyết định, mặc dù họ theo đuổi các mục tiêu khác nhau. Các mô hình dự đoán có thể được chia thành hai loại chính. Mục đích của các mô hình giải thích là chức năng xác định một mối quan hệ có thể có giữa một

biến phụ thuộc và một tập hợp các thuộc tính độc lập. Các mô hình hồi quy thuộc về loại này cũng như các mô hình phân loại. Kết quả của các mô hình chuỗi thời gian là đặc trưng xác định bất kỳ mẫu thời gian nào được biểu thị bằng một chuỗi các quan sát thời gian được gọi là cùng một biến số.

- Mô hình nhận dạng và học tập: Theo nghĩa rộng, mục đích của nhận dạng và học tập là để hiểu các cơ chế điều chỉnh sự phát triển của trí thông minh, được hiểu là khả năng rút ra kiến thức từ kinh nghiệm trong quá khứ để áp dụng nó trong tương lai. Các mô hình toán học cho việc học có hai mục tiêu chính. Mục đích của các mô hình diễn giải là xác định các mẫu thông thường trong dữ liệu và thể hiện chúng thông qua các quy tắc và tiêu chí dễ hiểu. Dựa trên sự tồn tại hay không thuộc tính mục tiêu, quá trình học tập có thể được giám sát hoặc không được giám sát. Trong trường hợp đầu tiên, thuộc tính đích thể hiện cho mỗi bản ghi lớp thành viên hoặc số lượng có thể đo được. Các mô hình phân loại và hồi quy thuộc về thể loại này. Trong trường hợp thứ hai, không có thuộc tính đích nào tồn tại và do đó, mục đích của phân tích là xác định tính thường xuyên, tương đồng và khác biệt trong dữ liệu, cũng có thể rút ra các quy tắc kết hợp. Ngoài ra, người ta có thể xác định các nhóm bản ghi, được gọi là các cụm, được đặc trưng bởi sự giống nhau trong mỗi cụm và bởi sự khác biệt giữa các yếu tố của các cụm khác nhau.

- Mô hình tối ưu hóa: các mô hình tối ưu hóa phát sinh một cách tự nhiên trong các quy trình ra quyết định, trong đó một tập hợp các nguồn lực hạn chế phải được phân bổ theo cách hiệu quả nhất cho các thực thể khác nhau. Những nguồn lực này có thể là nhân sự, quy trình sản xuất, nguyên liệu thô, linh kiện hoặc yếu tố tài chính. Các mô hình tối ưu hóa đại diện cho một nhóm các vấn đề tối ưu hóa xuất phát khi mục tiêu của quá trình ra quyết định là một hàm của các biến quyết định và các tiêu chí mô tả các quyết định khả thi có thể được biểu thị bằng một tập hợp các đẳng thức và bất đẳng thức toán học trong các biến quyết định.

- Mô hình quản lý dự án: Các mô hình toán học để ra quyết định đóng một vai trò quan trọng trong phương pháp quản lý dự án. Các mô hình này cho phép xác định thời gian thực hiện dự án tổng thể, giả định kiến thức xác định về thời lượng của từng

hoạt động. Mặt khác, các mô hình ngẫu nhiên, thường được quy vào các kỹ thuật đánh giá và xem xét dự án (PERT).

- Mô hình phân tích rủi ro: chủ yếu dựa trên định lý Bayes, lớp mô hình này thường được sử dụng để đánh giá rủi ro các lựa chọn của người ra quyết định, được sử dụng thành công trong một số ứng dụng lĩnh vực, như đầu tư công nghệ, thiết kế sản phẩm mới, nghiên cứu và phát triển, và đầu tư tài chính và bất động sản.

- Mô hình dòng chờ: Mục đích của lý thuyết dòng chờ là điều tra các hiện tượng tắc nghẽn xảy ra khi nhu cầu và cung cấp dịch vụ là ngẫu nhiên. Các mô hình dòng chờ cho phép đánh giá hiệu năng của một hệ thống một khi cấu trúc của nó đã được xác định, và do đó phần lớn hữu ích trong giai đoạn thiết kế hệ thống.

Chương 2: Một số mô hình phân loại

Các mô hình phân loại là các phương pháp học có giám sát để dự đoán giá trị của thuộc tính mục tiêu phân loại, không giống như các mô hình hồi quy xử lý các thuộc tính số. Bắt đầu từ một tập hợp các quan sát trong quá khứ có lớp mục tiêu được biết đến, các mô hình phân loại được sử dụng để tạo ra một tập hợp các quy tắc cho phép dự đoán lớp mục tiêu của các dữ liệu trong tương lai. Phân loại giữ một vị trí nổi bật trong lý thuyết học tập do ý nghĩa lý thuyết của nó và vô số ứng dụng mà nó có. Mặt khác, các cơ hội được phân loại mở rộng sang một số lĩnh vực ứng dụng khác nhau: lựa chọn khách hàng mục tiêu cho chiến dịch tiếp thị, phát hiện gian lận, nhận dạng hình ảnh, chẩn đoán bệnh, lập danh mục văn bản và nhận dạng email spam là một vài ví dụ về vấn đề có thể xử lý với mô hình phân loại.

2.1. Vấn đề phân loại

Trong một vấn đề phân loại, chúng ta có một tập dữ liệu D chứa m các quan sát được mô tả theo n thuộc tính giải thích và thuộc tính mục tiêu phân loại. Các thuộc tính giải thích, còn được gọi là các biến dự đoán, có thể là một phần phân loại và một phần số. Thuộc tính đích cũng được gọi là một lớp hoặc nhãn, trong khi các quan sát cũng được gọi là các ví dụ hoặc quan sát (dữ liệu). Đối với các mô hình phân loại, biến mục tiêu có số lượng giá trị hữu hạn. Cụ thể, chúng tôi có một vấn đề phân loại nhị phân nếu các thể hiện chỉ thuộc về hai lớp và phân loại đa lớp hoặc đa danh mục nếu có nhiều hơn hai lớp.

Mục đích của mô hình phân loại là xác định các mối quan hệ giữa các biến giải thích, chúng mô tả các dữ liệu thuộc cùng một lớp. Các mối quan hệ như vậy sau đó được chuyển thành các quy tắc phân loại được sử dụng để dự đoán lớp các dữ liệu mà chỉ các giá trị của các thuộc tính giải thích được biết. Các quy tắc có thể có các hình thức khác nhau tùy thuộc vào loại mô hình được sử dụng.

Theo quan điểm toán học, trong một bài toán phân loại các dữ liệu đã biết được đưa ra, bao gồm các cặp (x_i, y_i), i ϵ M, trong đó $x_i \epsilon R^n$ là vectơ các giá trị được lấy bởi các thuộc tính dự đoán cho dữ liệu thứ i và $y_i \epsilon H = \{v_1, v_2, \dots, v_H\}$ biểu thị lớp mục tiêu tương ứng. Trong bài toán phân loại nhị phân, người ta có H = 2, và hai

lớp có thể được ký hiệu như H = (0, 1) hoặc H = (-1, 1). Các mô hình phân loại phù hợp cho cả giải thích và dự đoán, chúng thuộc các phương pháp học tập có giám sát. Các mô hình đơn giản thường mang lại các quy tắc phân loại trực quan có thể dễ dàng giải thích, trong khi các mô hình tiên tiến hơn tạo ra các quy tắc ít dễ hiểu hơn mặc dù chúng thường mang lại độ chính xác cao hơn.

Một phần dữ liệu trong bộ dữ liệu D được sử dụng để đào tạo một mô hình phân loại, nghĩa là để có được mối quan hệ giữa biến mục tiêu và các biến giải thích. Những dữ liệu còn lại được sử dụng sau này để đánh giá sự phù hợp của mô hình được đào tạo và để chọn mô hình tốt nhất trong số các mô hình được phát triển. Sự phát triển của một mô hình phân loại bao gồm ba giai đoạn chính:

- Giai đoạn đào tạo: trong giai đoạn đào tạo, thuật toán phân loại được sử dụng cho các dữ liệu thuộc tập con T của tập dữ liệu D, được gọi là tập huấn luyện, để lấy các quy tắc phân loại cho phép lớp mục tiêu tương ứng y gắn với mỗi quan sát x.
- Giai đoạn kiểm tra: trong giai đoạn kiểm tra, các quy tắc được tạo ra trong giai đoạn được sử dụng để phân loại các dữ liệu của D không có trong tập dữ liệu huấn luyện, trong đó giá trị của lớp mục tiêu đã được biết đến. Để đánh giá độ chính xác của mô hình phân loại, lớp mục tiêu thực tế của từng dữ liệu trong tập kiểm tra được so sánh với lớp được dự đoán bởi trình phân loại. Để tránh đánh giá quá cao độ chính xác của mô hình, tập huấn luyện và tập kiểm tra phải rời rạc.
- Giai đoạn dự đoán: giai đoạn dự đoán đại diện cho việc sử dụng thực tế của mô hình phân loại để gán lớp mục tiêu cho các dữ liệu mới sẽ được ghi lại trong tương lai. Một dự đoán có được bằng cách áp dụng các quy tắc được tạo ra trong giai đoạn đào tạo cho các biến giải thích mô tả dữ liệu mới.

2.2. Một số mô hình phân loại

2.2.1. Hồi quy logistic

Hồi quy logistic là một mô hình hồi quy thường được sử dụng trong phân loại nhị phân. Hồi quy logistic được đặt tên cho hàm được sử dụng cốt lõi, hàm logistic, ở đây sẽ dùng bộ phân loại sigmoid để đưa ra quyết định.

Với một quan sát đầu vào x, chúng ta sẽ biểu thị bằng một vectơ các đặc trưng $[x_1, x_2, \dots, x_n]$. Đầu ra phân loại y có thể là 1 (có nghĩa quan sát là một thành viên của lớp) hoặc 0 (quan sát không phải là thành viên của lớp). Chúng ta muốn biết xác suất P(y = 1 | x) rằng quan sát này là một thành viên của lớp, và P(y = 0|x) là xác suất quan sát này không là thành viên của lớp.

Hồi quy logistic giải quyết nhiệm vụ này bằng cách học, từ một tập huấn luyện, một vectơ trọng số và một hệ số bias. Mỗi trọng số w_i là một số thực và được liên kết với một trong các đặc trưng đầu vào x_i. Trọng số w_i biểu thị mức độ quan trọng của tính năng đầu vào đối với quyết định phân loại.

Để đưa ra quyết định về một dữ liệu thử nghiệm sau khi chúng ta đã học được các trọng số trong việc đào tạo, bộ phân loại đầu tiên nhân mỗi x_i với trọng số của nó, lấy tổng của chúng với hệ số bias b

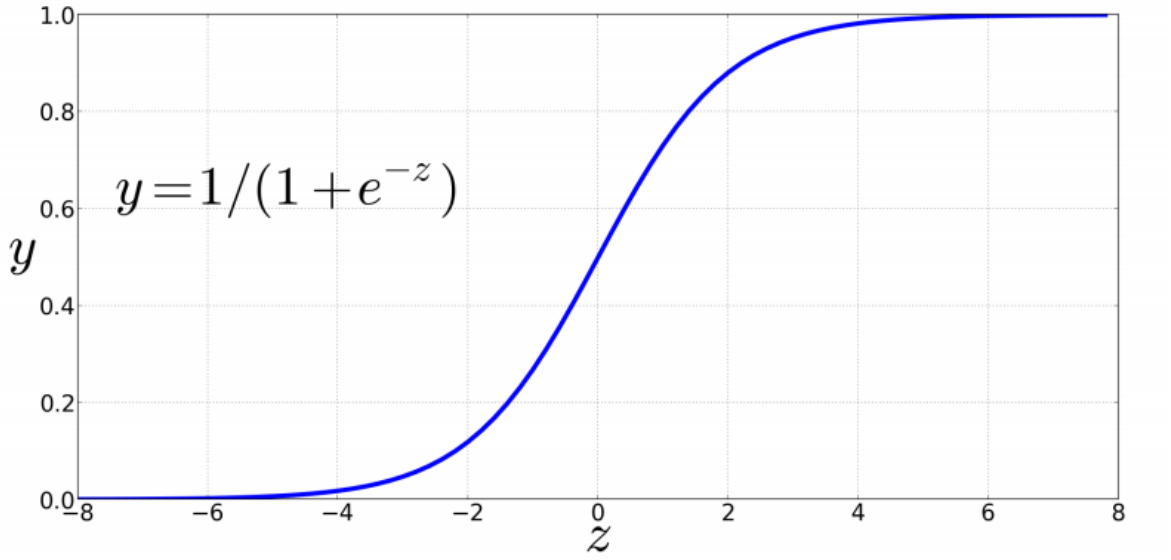
$$z = \sum_{i=1}^{n} w_i x_i + b$$

Đặt w_0 = b, vectơ w trở thành $[w_0, w_1, w_2, \dots, w_n]$ và mở rộng vectơ x thành $[1, x_1, x_2, \dots, x_n]$, z trở thành tích của vectơ trọng số w và vectơ đặc trưng x

$$z = \sum_{i=0}^{n} w_i x_i = w^T \cdot x$$

Để tạo ra xác suất, chúng ta chuyển z qua hàm sigmoid $\sigma(z)$, hàm sigmoid lấy một số có giá trị thực và ánh xạ nó vào phạm vi [0, 1], đó là những gì chúng ta muốn cho một xác suất

$$y = \sigma(z) = \frac{1}{1 + e^{-z}}$$



Hình 2.1: Đồ thị hàm sigmoid

Nếu áp dụng sigmoid cho tổng các đặc trưng có trọng số, chúng ta sẽ có một số từ 0 đến 1. Để xác định nó, chúng ta chỉ cần đảm bảo rằng hai trường hợp, P(y = 1) và P(y = 0), tổng bằng 1. Chúng ta có thể làm điều này như sau:

$$P(y = 1|x) = \sigma(w^T \cdot x) = \frac{1}{1 + e^{-w \cdot x}}$$

$$P(y = 0|x) = 1 - \sigma(w^T \cdot x) = 1 - \frac{1}{1 + e^{-w^T \cdot x}} = \frac{e^{-w^T \cdot x}}{1 + e^{-w^T \cdot x}}$$

Bây giờ ta có một xác suất P(y = 1 | x) của một quan sát x, làm cách nào để đưa ra quyết định. Chúng ta sẽ đưa ra một ngưỡng quyết định ε (có thể là 0.5, hoặc khác đối với từng bài toán cụ thể) và nói quan sát x thuộc lớp y nếu xác suất P(y =1|x) lớn hơn ε và ngược lại.

$$\hat{y} = \begin{cases} 1 \ \ nếu \ P(y = 1|X) \geq \varepsilon \\ 0 \ \ nếu \ P(y = 1|X) < \varepsilon \end{cases}$$

Các tham số của mô hình, các trọng số w, được học như thế nào? Hồi quy logistic là một ví dụ của phân loại có giám sát, trong đó chúng ta biết nhãn y (0 hoặc 1) cho mỗi quan sát x. Những gì hệ thống tạo ra thông qua phương trình trên là $\hat{y}$,

ước tính của hệ thống của y thật. Chúng ta muốn tìm hiểu các trọng số (có nghĩa là w) làm $\hat{y}$ cho mỗi quan sát đào tạo càng gần với y thực sự. Chúng ta thường nói về khoảng cách giữa dự đoán của hệ thống và thực tế, chúng ta gọi khoảng cách này là hàm mất mát hoặc hàm chi phí.

$$L(\hat{y}, y) = mức\ độ\ \hat{y}\ khác\ y$$

Chúng ta muốn tìm các trọng số tối đa hóa xác suất sự chính xác của nhãn P(y | x). Vì chỉ có hai kết quả riêng biệt (1 hoặc 0), đây là phân phối Bernoulli và chúng ta có thể biểu thị xác suất P(y | x) mà trình phân loại tạo ra cho một quan sát như sau:

$$P(y \mid x) = \sigma^{y}(1-\sigma)^{1-y}$$

Hàm mất mát cross-entropy:

$$L(w) = -\frac{1}{m}\sum_{i=1}^{m} y^{(i)} \log\big(\sigma(z^{(i)})\big) + (1-y^{(i)}) \log\big(1-\sigma(z^{(i)})\big)$$

Với m là kích thước tập dữ liệu, $y^{(i)}$ là lớp tương ứng của dữ liệu thứ i (0 hoặc 1), $z^{(i)} = w^{T} \cdot x^{(i)}$ là tương ứng khi tính với mô hình cho dữ liệu thứ i. Việc ta cần làm là tối thiểu hóa hàm mất mát $L(w)$.

Để tối ưu hàm mất mát L(w) trên, ta sử dụng Gradient Descent để thực hiện. Ở đây, đạo hàm của hàm log trên có thể được tính:

$$\frac{\partial L(w)}{\partial w_j} = \frac{\partial L(w)}{\partial p} \cdot \frac{\partial p}{\partial z} \cdot \frac{\partial z}{\partial w_j} = \frac{1}{m}\sum_{i=1}^{m} \big(\sigma_j^{(i)} - y_j^{(i)}\big)\, x_j^{(i)} = \frac{1}{m}\big(\sigma_j - y_j\big) X_j$$

Ta sẽ cập nhật trọng số sau mỗi vòng lặp cho hồi quy logistic

$$w = w - \eta \frac{1}{m}(\sigma - y) X$$

2.2.2. Cây quyết định

Cây quyết định có lẽ là phương pháp học tập được biết đến nhiều nhất và sử dụng rộng rãi nhất trong các ứng dụng khai thác dữ liệu. Cây quyết định có khái niệm

đơn giản, dễ sử dụng, tốc độ tính toán khá cao và mạnh mẽ với các quy tắc dễ hiểu nó tạo ra.

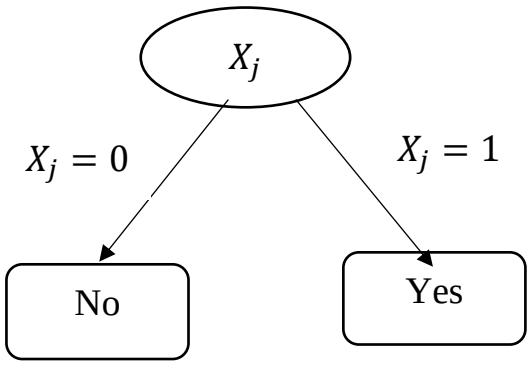
Sự phát triển của cây phân loại tương ứng với giai đoạn huấn luyện của mô hình và được điều chỉnh bởi một thủ tục đệ quy có tính chất heuristic, dựa trên chia để trị từ trên xuống. Các dữ liệu của tập huấn luyện ban đầu chứa trong nút gốc của cây được chia thành các tập con rời rạc được đặt tạm thời trong hai hoặc nhiều nút con cháu (phân nhánh). Tại mỗi nút được xác định theo cách này, kiểm tra được áp dụng để xác minh xem các điều kiện dừng phát triển của nút có được thỏa mãn hay không. Nếu ít nhất một trong những điều kiện này được đáp ứng, không có phân chia nào nữa được thực hiện và nút trở thành một lá của cây. Nếu không, việc phân chia các dữ liệu có trong nút được thực hiện. Kết thúc thủ tục khi không có nút cây nào có thể được chia nhỏ hơn nữa hoặc mỗi nút lá được gắn nhãn với giá trị của lớp mà phần lớn các dữ liệu trong nút thuộc về một tiêu chí được gọi là biểu quyết đa số.

Việc phân chia các dữ liệu trong mỗi nút được thực hiện bằng quy tắc chia tách, cũng được gọi là quy tắc tách, được chọn dựa trên một hàm đánh giá cụ thể. Bằng cách thay đổi quy tắc chia tách được sử dụng, có thể thu được các phiên bản khác nhau của cây phân loại. Hầu hết các tiêu chí đánh giá được đề xuất đều có chung mục tiêu là tối đa hóa tính đồng nhất của lớp mục tiêu cho các quan sát được đặt trong mỗi nút.

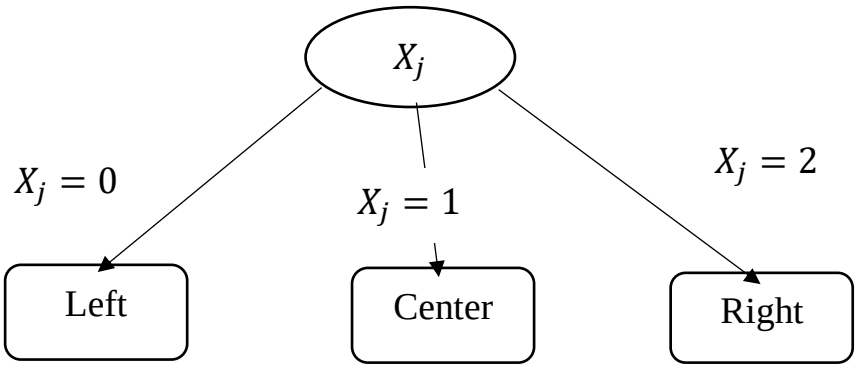
Tập hợp các quy tắc phân tách có thể được tìm thấy dọc theo đường dẫn kết nối gốc cây với một nút lá tạo thành quy tắc phân loại. Trong giai đoạn dự đoán, để gán lớp mục tiêu cho một dữ liệu mới, một đường dẫn được theo dõi từ nút gốc đến nút lá bằng cách tuân theo chuỗi quy tắc được áp dụng cho các giá trị của các thuộc tính của dữ liệu mới. Lớp mục tiêu dự đoán sau đó trùng khớp với lớp mà nút lá đạt được đã được dán nhãn trong giai đoạn phát triển.

Bắt đầu từ một tập dữ liệu huấn luyện, xây dựng một số lượng lớn các cây phân loại riêng biệt. Có thể chỉ ra rằng vấn đề xác định cây tối ưu là NP-Khó. Kết quả là, các phương pháp để phát triển cây phân loại là heuristic.

• **Quy tắc chia tách**: có thể được chia thành cây nhị phân và cây tổng quát dựa trên số lượng con cháu tối đa mà mỗi nút được phép tạo cùng với phân chia đơn biến và đa biến. Một cây được gọi là nhị phân nếu mỗi nút có nhiều nhất hai nhánh. Một cây được gọi là tổng quát nếu mỗi nút có số lượng nhánh là tùy ý.



Hình 2.2: Phân chia nhị phân



Hình 2.3: Phân chia tổng quát

• **Tiêu chí chọn thuộc tính chia tách:** sự chia tách được thực hiện trên các nút theo thuộc tính giải thích nào hiệu quả nhất dựa trên tiêu chí phân tách đặt ra (các hàm đánh giá, cũng cấp thước đo về sự không đồng nhất của các giá trị lớp tại mỗi nút). Trong số các chỉ số không đồng nhất của một nút thì phổ biến nhất là chỉ số entropy, chỉ số gini,…

• Chỉ số Entropy: p_i là xác suất của một dữ liệu tùy ý trong nút D thuộc lớp đối tượng i, chỉ số entropy tại một nút đại diện cho độ hỗn loạn thông tin tại nút đó: $E(D) = -\sum_{i=1}^{m} p_i \log_2 p_i$, tiêu chí lựa chọn thuộc tính chia tách ở đây được tính theo mức

giảm entropy hay còn gọi là mức tăng thông tin thu được: $Gain(A) = E(D) - E_A(D) = E(D) - \sum_{j=1}^{v} \frac{|D_j|}{|D|} \times E(D_j)$ (A là thuộc tính chia tách, v là số phân vùng sau khi chia tách). Mức giảm entropy càng lớn càng tốt.

• Chỉ số gini: cũng như entropy, chỉ số gini đại diện cho mức độ pha tạp thông tin trong dữ liệu: $gini(D) = 1 - \sum_{i=1}^{n} p_i^2$ với p_i là tần số tương đối của lớp i trong D. Nếu tập dữ liệu D được chia tách thành hai tập con D_i và D_2, $gini_A(D) = \frac{|D_1|}{|D|} gini(D_1) + \frac{|D_2|}{|D|} gini(D_2)$ là chỉ số gini khi chia tách D dựa trên thuộc tính A. Độ giảm sự pha tạp thông tin theo thuộc tính A: $\Delta gini(A) = gini(D) - gini_A(D)$. Thuộc tính cung cấp $gini_{split}(A)$ nhỏ nhất (hoặc mức giảm sự pha tạp thông tin lớn nhất) được chọn để phân chia nút.

• Ví dụ về chọn thuộc tính phân tách:

age	income	student	credit_rating	buys
<=30	high	no	fair	no
<=30	high	no	excellent	no
30…40	high	no	fair	yes
>40	medium	no	fair	yes
>40	low	yes	fair	yes
>40	low	yes	excellent	no
30…40	low	yes	excellent	yes
<=30	medium	no	fair	no
<=30	low	yes	fair	yes
>40	medium	yes	fair	yes
<=30	medium	yes	excellent	yes
30…40	medium	no	excellent	yes
30…40	high	yes	fair	yes
>40	medium	no	excellent	no

Với entropy, mức tăng thông tin: Gain(age) = 0.246, Gain(income) = 0.029, Gain(student) = 0.151, Gain(credit_rating) = 0.048, như vậy ta sẽ chọn thuộc tính tuổi làm thuộc tính chia tách tại nút gốc.

Với gini, mức giảm sự pha tạp: $\Delta gini(age)$ = 0.116, $\Delta gini(income)$ = 0.019, $\Delta gini(student)$ = 0.092, $\Delta gini(credit\_rating)$ = 0.03, như vậy thuộc tính được chọn để chia tách vẫn là tuổi.

- **Tiêu chí dừng và quy tắc cắt tỉa:** là một bộ quy tắc được sử dụng tại mỗi nút trong quá trình phát triển cây để xác định xem bó có phù hợp để tạo ra nhiều nhánh hơn không, hay nút hiện tại sẽ là nút lá. Một số tiêu chí dừng như kích thước nút (nút chứa số lượng dữ liệu đặt một ngưỡng nào đó), độ tinh khiết (tỷ lệ dữ liệu thuộc cùng một lớp), cải thiện (mức giảm độ hỗn loạn hay pha tạp của dữ liệu), độ sâu tối đa của cây. Một cây có quá nhiều phân nhánh thường đạt được sự quá phù hợp với tập dữ liệu huấn luyện nhưng gây ra lỗi nhiều hơn với bộ dữ liệu kiểm tra và dữ liệu dự đoán trong tương lai. Hiện tượng này thường được gọi là quá mức, một cây có quá nhiều phân nhánh tạo ra nhiều quy tắc, từ đó giảm lỗi ở tập huấn luyện, tuy nhiên chúng cũng làm tăng lỗi tổng quát hóa. Các tiêu chí dừng có thể giúp ngăn chặn vấn đề này bằng cách đưa ra một số ngưỡng cho các tiêu chí dừng được gọi là quy tắc cắt tỉa trước, nó tỉa một cây bằng cách hạn chế sự phát triển của nó và thường được dùng, tuy nhiên việc chọn được ngưỡng phù hợp là khó và cần nhiều thử nghiệm. Có những kỹ thuật được cắt tỉa sau, được áp dụng sau khi đã xây dựng hoàn chỉnh một cây để giảm số lượng phân nhánh và hy vọng cải thiện kết quả mô hình, phương pháp này sử dụng bộ dữ liệu độc lập với tập huấn luyện để xác định cây được cắt tỉa tốt nhất.

2.2.3. Phương pháp Bayes

Phương pháp Bayes thuộc họ mô hình phân loại xác suất. Họ tính toán rõ ràng xác suất P(y | x) rằng một quan sát đã cho thuộc về một lớp mục tiêu cụ thể bằng định lý Bayes, dựa trên một xác suất P (y) và xác suất có điều kiện P (x | y) đã biết.

Không giống những phương pháp khác không dựa trên các giả định về xác suất, phân loại Bayes yêu cầu người dùng ước tính xác suất P(x | y) mà một quan sát cụ thể có thể xảy ra, miễn là nó thuộc về một lớp cụ thể. Do đó, giai đoạn học tập của phân loại Bayes có thể được xác định bằng phân tích sơ bộ các dữ liệu trong tập huấn luyện, rút ra ước tính xác giá trị xác suất cần thiết để thực hiện nhiệm vụ phân loại.

Chúng ta hãy xem xét một quan sát chung x của tập huấn luyện, có biến mục tiêu y có thể lấy các giá trị riêng biệt H được ký hiệu là $H = \{v_1, v_2, \ldots, v_H\}$. Định lý Bayes được sử dụng để tính xác suất P (y | x), nghĩa là xác suất nhận lớp mục tiêu y đã cho của dữ liệu x:

$$P(y|x) = \frac{P(x|y).P(y)}{P(x)}$$

Để phân loại một dữ liệu mới x, trình phân loại Bayes áp dụng một nguyên tắc được gọi là ước lượng hậu nghiệm cực đại (maximum a posteriori) bao gồm tính toán xác suất P(y | x) và gán dữ liệu x cho lớp mang lại giá trị P(y | x) tối đa:

$$y_{MAP} = \arg\max_{y \in H} P(y\,|x) = \arg\max_{y \in H} \frac{P(x|y).P(y)}{P(x)}$$

Do mẫu số P(x) độc lập với y, như vậy để tối đa hóa xác suất, ta chỉ cần tối đa hóa tử số. Do đó dữ liệu x được gán cho lớp v_h khi và chỉ khi

$P(x \mid y = v_h)P(y = v_h) \geq P(x \mid y = v_l)P(y = v_l)$ $với$ $l = 1, 2, \ldots, H.$

Xác suất P(y) có thể được ước tính bằng cách sử dụng tần số m_h mà mỗi giá trị của lớp mục tiêu v_h xuất hiện trong tập dữ liệu D, nghĩa là

$$P(y = v_h) = \frac{m_h}{m}$$

Với một mẫu đủ lớn, các ước tính xác suất này sẽ khá chính xác.

Thật không may, một ước tính mẫu tương tự của các xác suất có điều kiện P (x | y) không thể có được trong thực tế do độ phức tạp tính toán và số lượng lớn các quan sát mẫu mà nó sẽ yêu cầu. Để thấy điều này, hãy xem xét tình huống giả định

sau: nếu tập dữ liệu bao gồm 50 thuộc tính phân loại nhị phân, sẽ có $2^{50} \approx 10^{15}$ kết hợp các giá trị thuộc tính. Để có được ước tính đáng tin cậy, cần phải có ít nhất 10 quan sát cho mỗi kết hợp. Do đó, tập dữ liệu D phải chứa ít nhất 1016 quan sát. Để khắc phục khó khăn tính toán được mô tả, có thể đưa ra giả thuyết đơn giản hóa dẫn đến phân loại Naïve Bayes.

Naïve Bayes: các trình phân loại Naïve Bayes dựa trên giả định rằng các biến giải thích là độc lập có điều kiện với lớp mục tiêu. Giả thuyết này cho phép chúng ta biểu diễn xác suất P(x | y):

$$P(x \mid y) = P(x_1 \mid y) \times P(x_2 \mid y) \times \ldots \times P(x_n \mid y) = \prod_{j=1}^{n} P\left(x_j \mid y\right).$$

Các xác suất $P\left(x_j \mid y\right)$, j ∈ N, có thể được ước tính bằng cách sử dụng các dữ liệu từ tập huấn luyện, tùy thuộc vào bản chất của thuộc tính được xem xét.

Thuộc tính số rời rạc hoặc phân loại: với thuộc tính a_j có thể lấy các giá trị $\{r_{j1}, r_{j2}, \ldots, r_{jk}\}$, xác suất $P\left(x_j \mid y\right) = P\left(x_j = r_{jk} \middle| y = v_h\right)$ được đánh giá là tỷ lệ giữa số s_{jhk} của các trường hợp của lớp v_h mà thuộc tính a_j lấy giá trị r_{jk} và tổng số các trường hợp của lớp v_h trong tập dữ liệu D, đó là

$$P\left(x_j \mid y\right) = P\left(x_j = r_{jk} \middle| y = v_h\right) = \frac{s_{jhk}}{m_h}$$

Thuộc tính số: Đối với thuộc tính số a_j, xác suất $P\left(x_j \mid y\right)$ được ước tính giả sử rằng các dữ liệu tuân theo phân phối đã cho. Ví dụ, người ta có thể xem xét hàm mật độ Gaussian, trong đó

$$P\left(x_j \mid y = v_h\right) = \frac{1}{\sqrt{2\pi\sigma_{jh}}} e^{-\frac{\left(x_j - \mu_{jh}\right)^2}{2\sigma_{jh}^2}}$$

Trong đó μ_{jh} và σ_{jh} lần lượt biểu thị độ lệch trung bình và độ lệch chuẩn của biến x_j cho các dữ liệu của lớp v_h và có thể được ước tính trên cơ sở các dữ liệu có trong D.

Giả định đơn giản hóa tính độc lập có điều kiện của các thuộc tính, giúp dễ dàng tính toán xác suất có điều kiện, bằng chứng thực nghiệm cho thấy các trình phân loại Bayes thường có thể đạt được các mức chính xác không thấp hơn các trình phân loại được cung cấp bởi các phương pháp khác như cây quyết định, thậm chí bằng các phương pháp phân loại phức tạp hơn.

Mô tả việc sử dụng trình phân loại Naïve Bayes, ta xem xét lại ví dụ trong mục 2.2.3 liên quan tới việc mua máy tính. Ta tính được:

Age:

P (age <=30 | buys = ‘yes’) = $\frac{2}{9}$, P (age <=30 | buys = ‘no’) = $\frac{3}{5}$

P (30< age <= 40 | buys = ‘yes’) =$\frac{4}{9}$, P (30< age <= 40 | buys = ‘no’) =$\frac{1}{5}$

P (age > 40 | buys = ‘yes’) =$\frac{3}{9}$, P (age > 40 | buys = ‘no’) =$\frac{1}{5}$

Income:

P (income = ‘low’ | buys = ‘yes’) = $\frac{3}{9}$, P (income = ‘low’ | buys = ‘no’) = $\frac{1}{5}$

P (income=‘medium’|buys =‘yes’) =$\frac{4}{9}$, P (income =‘medium’|buys=‘no’) =$\frac{2}{5}$

P (income = ‘high’ | buys = ‘yes’) =$\frac{2}{9}$, P(income = ‘high’ | buys = ‘no’) =$\frac{2}{5}$

Student:

P (student = ‘yes’ | buys = ‘yes’) = $\frac{6}{9}$, P (student = ‘yes’ | buys = ‘no’) = $\frac{1}{5}$

P (student = ‘no’ | buys = ‘yes’) =$\frac{3}{9}$, P (student = ‘no’ | buys = ‘no’) =$\frac{4}{5}$

Credit_rating:

P (credit_rating = ‘excellent’ | buys = ‘yes’) = $\frac{3}{9}$,

P (credit_rating = ‘fair’ | buys = ‘yes’) =$\frac{6}{9}$,

P (credit_rating = ‘excellent’ | buys = ‘no’) =$\frac{3}{5}$,

P (credit_rating = ‘fair’ | buys = ‘no’) = $\frac{2}{5}$

Khi xác suất có điều kiện của từng thuộc tính được cung cấp cho lớp mục tiêu đã được ước tính, giả sử rằng ta muốn dự đoán một quan sát mới được biểu thị bằng vectơ x = (<=30, ‘low’, ‘no’, ‘fair’)

$$P(x \mid yes) = \frac{2}{9} \cdot \frac{3}{9} \cdot \frac{3}{9} \cdot \frac{6}{9} = \frac{4}{243}$$

$$P(x \mid no) = \frac{3}{5} \cdot \frac{1}{5} \cdot \frac{4}{5} \cdot \frac{2}{5} = \frac{24}{625}$$

Tần số tương đối của hai lớp được cho

P(buys = ‘yes’) = $\frac{9}{14}$, P(buys = ‘no’) = $\frac{5}{14}$.

Như vậy chúng ta có

P(buys = ‘yes’| x) = P(x | yes).P(buys = ‘yes’) = $\frac{4}{243} \cdot \frac{9}{14} = \frac{2}{189}$

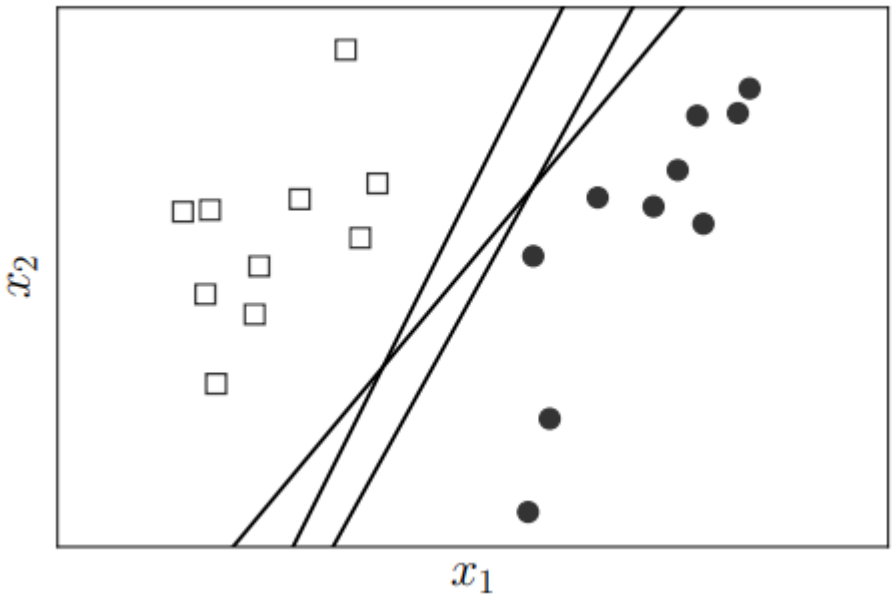
P(buys = ‘no’| x) = P(x | no).P(buys = ‘no’) = $\frac{24}{625} \cdot \frac{5}{14} = \frac{12}{875}$

Quan sát mới x sẽ được gắn với giá trị lớp ‘no’, vì lấy xác suất tối đa.

2.2.4. Máy vectơ hỗ trợ (SVM)

Máy vectơ hỗ trợ là một họ các phương pháp phân tách để phân loại và hồi quy được phát triển trong lý thuyết học thống kê. Chúng đã được chứng minh là đạt được hiệu suất tốt hơn về độ chính xác so với các phân loại khác trong một số lĩnh vực ứng dụng và có thể mở rộng hiệu quả cho các vấn đề lớn. Một đặc trưng quan trọng hơn nữa liên quan đến việc hiểu rõ các quy tắc phân loại được tạo ra. Các máy vectơ hỗ trợ xác định một tập hợp các dữ liệu, được gọi là vectơ hỗ trợ, là các quan sát đại diện nhất cho mỗi lớp mục tiêu. Theo một cách nào đó, chúng đóng vai trò

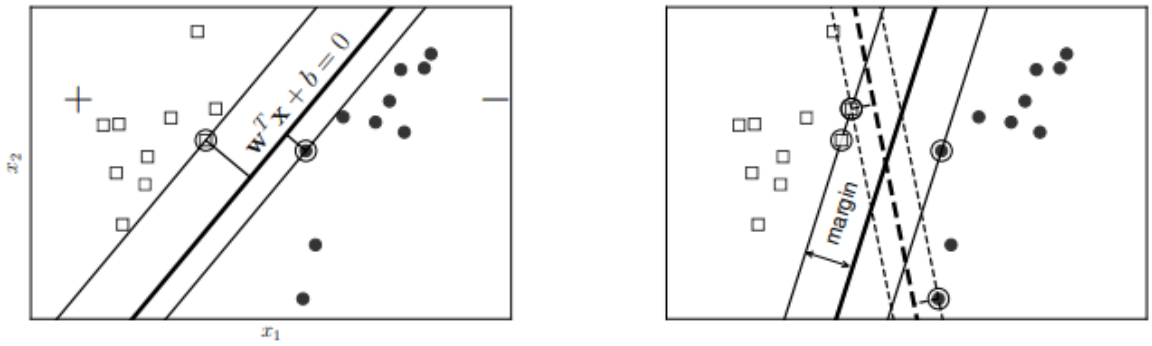
quan trọng hơn các dữ liệu khác, vì chúng xác định vị trí của bề mặt phân tách được tạo bởi bộ phân loại trong không gian thuộc tính.



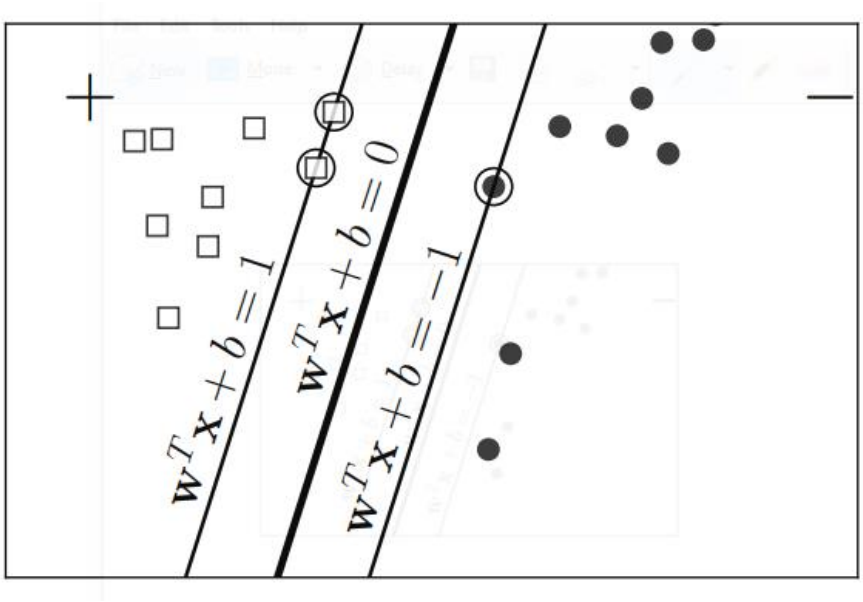
Hình 2.4: Hai lớp dữ liệu vuông và tròn là tách biệt tuyến tính

Giả sử có hai lớp dữ liệu được mô tả bởi các vectơ đặc trưng trong không gian nhiều chiều, hơn nữa hai lớp dữ liệu này là tách biệt tuyến tính tức là tồn tại một siêu phẳng phân chia chính xác hai lớp đó. Việc cần làm là tìm một siêu phẳng sao cho tất cả các điểm thuộc một lớp nằm về cùng một phía của siêu phẳng đó và ngược phía với toàn bộ các điểm thuộc lớp còn lại. Như vậy sẽ có vô số nghiệm thỏa mãn như hình 2.4. Câu hỏi đặt ra là trong vô số các mặt phân chia đó, đâu là mặt tốt nhất.

Để trả lời câu hỏi này, chúng ta cần tìm một tiêu chuẩn để đo sự lệch về mỗi lớp của đường phân chia. Gọi khoảng cách nhỏ nhất từ một điểm thuộc một lớp tới đường phân chia là *lề* (margin). Ta cần tìm một đường phân chia sao cho lề của hai lớp là như nhau đối với đường phân chia đó. Hơn nữa, độ rộng của lề càng lớn thì khả năng xảy ra phân loại lỗi càng thấp. Bài toán tối ưu trong SVM chính là bài toán đi tìm đường phân chia sao cho lề rộng nhất.



Hình 2.5: Ý tưởng SVM. Lề của hai lớp phải bằng nhau và lớn nhất có thể



Hình 2.6: Giả sử mặt phân chia có phương trình $w^T x + b = 0$. Không mất tính tổng quát, bằng cách nhân các hệ số w và b với các hằng số phù hợp, ta có thể giả sử rằng điểm gần nhất của lớp vuông tới mặt này thoả mãn $w^T x + b = 1$. Khi đó, điểm gần nhất của lớp tròn thoả mãn $w^T x + b = -1$.

Giả sử các điểm vuông có nhãn là 1, các điểm tròn có nhãn là -1 và siêu phẳng $w^T x + b = 0$ là mặt phân chia hai lớp (Hình 2.6). Ngoài ra, lớp hình vuông nằm về phía dương, lớp hình tròn nằm về phía âm của mặt phân chia. Nếu xảy ra điều ngược lại, ta chỉ cần đổi dấu của w và b. Bài toán tối ưu trong SVM sẽ là bài toán đi tìm các tham số mô hình w và b.

Với cặp dữ liệu (x_n, y_n) bất kỳ, khoảng cách từ x_n tới mặt phân chia là $\frac{y_n(w^T x_n+b)}{\|w\|^2}$. Điều này xảy ra ta đã giả sử y_n cùng dấu với phía của x_n. Từ đó suy ra y_n cùng dấu với $(w^T x_n + b)$ và tử số luôn là một đại lượng không âm. Với mặt phân chia này, lề được tính là khoảng cách gần nhất từ một điểm (trong cả hai lớp, vì cuối cùng lề của hai lớp bằng nhau) tới mặt phân chia.

$$l\hat{e} = \min_n \frac{y_n(w^T x_n + b)}{\|w\|^2}$$

Bài toán tối ưu của SVM đi tìm w và b sao cho lề đạt giá trị lớn nhất:

$$(w, b) = \arg\max_{w,b} \left\{ \min_n \frac{y_n(w^T x_n + b)}{\|w\|^2} \right\} = \arg\max_{w,b} \left\{ \frac{1}{\|w\|^2} \min_n y_n(w^T x_n + b) \right\} \quad (*)$$

Với mọi n ta luôn có

$$y_n(w^T x_n + b) \geq 1$$

Bài toán tối ưu (*) được đưa về bài toán tối ưu có ràng buộc dạng:

$$(w, b) = \arg\max_{w,b} \frac{1}{\|w\|^2}$$

thỏa mãn: $y_n(w^T x_n + b) \geq 1, \forall n = 1, 2, \dots, N$

Bằng một biến đổi đơn giản, ta có thể tiếp tục đưa bài toán này về dạng

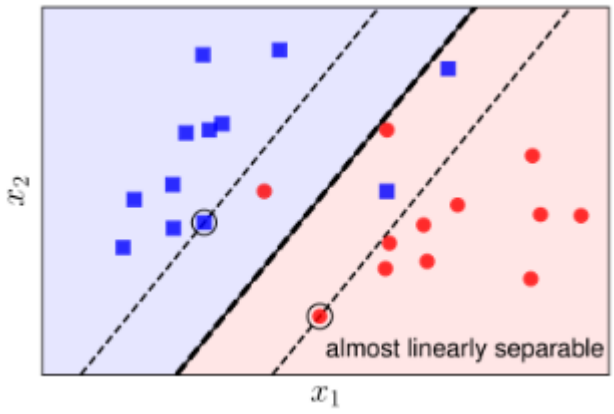
$$(w, b) = \arg\min_{w,b} \frac{1}{2} \|w\|_2^2$$

thỏa mãn: $1 - y_n(w^T x_n + b) \leq 0, \forall n = 1, 2, \dots, N$

Sau khi giải bài toán, ta tìm được mặt phẳng phân chia $w^T x + b = 0$, nhãn của một điểm bất kỳ sẽ được xác định đơn giản bằng

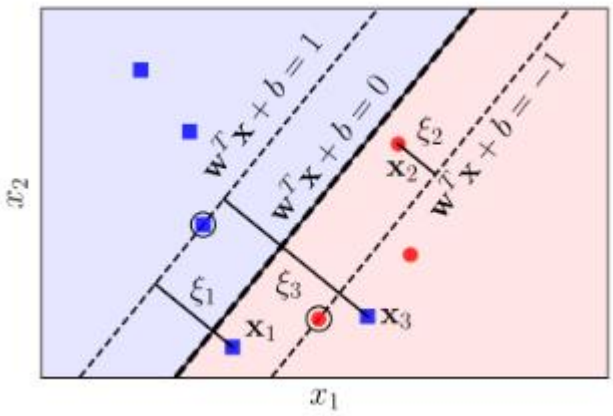
$$class(x) = sgn(w^T x + b)$$

Tuy nhiên nhiều khả năng là các điểm của bộ dữ liệu không tách biệt tuyến tính mà chỉ gần tách biệt tuyến tính, như hình 2.7. Trong trường hợp này, dễ nhận thấy SVM thậm chí không làm việc, tuy nhiên nếu ta chịu hy sinh một số điểm, ta vẫn có thể tạo được một đường phân chia khá tốt như trong hình, các đường hỗ trợ nét đứt mảnh vẫn giúp tạo được một lề đủ lớn và phân lớp chính xác hầu hết các điểm dữ liệu.



Hình 2.7: Dữ liệu gần tách biệt tuyến tính

Như đã đề cập ở trên, để có được đường phân chia như hình, chúng ta cần phải hy sinh một vài điểm dữ liệu, tất nhiên chúng ta phải hạn chế sự hy sinh này. Vậy hàm mục tiêu là một sự kết hợp để tối đa lề và tối thiểu sự hy sinh.



Hình 2.8: Các Slack Variable

Với mỗi điểm x_n trong tập toàn bộ dữ liệu huấn luyện, ta *giới thiệu* thêm một biến đo *sự hy sinh* ξ_n tương ứng. Biến này còn được gọi là *slack variable*. Với những điểm x_n nằm trong *vùng an toàn*, $\xi_n = 0$. Với mỗi điểm nằm trong *vùng không an toàn* như x_1 *hay* x_3, ta có $\xi_n > 0$. Nhận thấy rằng $y_i = \pm 1$ là nhãn của x_i trong vùng không an toàn thì $\xi_i = |w^T x_i + b - y_i|$

Hàm mục tiêu của bài toán tối ưu sẽ thêm một số hạng nữa giúp tối thiểu sự hi sinh $\frac{1}{2}\|w\|_2^2 + C\sum_{n=1}^{N}\xi_n$ với C là một hằng số dùng để điều chỉnh tầm quan trọng giữa lề và sự hy sinh. Điều kiện ràng buộc cũng thay đổi một chút, ta sẽ có ràng buộc $y_n(w^T x_n + b) \geq 1 - \xi_n \Leftrightarrow 1 - \xi_n - y_n(w^T x_n + b) \leq 0$ và ràng buộc phụ $\xi_n \geq 0\ \forall n = 1, 2, \dots, N$.

Tóm lại bài toán tối ưu cho trường hợp này là:

$$(w, b, \xi) = \arg\min_{w,b,\xi} \frac{1}{2}\|w\|_2^2 + C\sum_{n=1}^{N}\xi_n$$

thỏa mãn: $1 - \xi_n - y_n(w^T x_n + b) \leq 0, \forall n = 1, 2, \dots, N$

$$-\xi_n \leq 0\ \forall n = 1, 2, \dots, N$$

Bài toán có thể được giải quyết thông qua tính đối ngẫu Lagrangian.

Hàm Lagrangian cho bài toán là:

$$L(w, b, \xi, \lambda, \mu) = \frac{1}{2}\|w\|_2^2 + C\sum_{n=1}^{N}\xi_n + \sum_{n=1}^{N}\lambda(1 - \xi_n - y_n(w^T x_n + b)) - \sum_{n=1}^{N}\mu_n\xi_n$$

Với $\lambda = [\lambda_1, \lambda_2, \dots, \lambda_N]^T \geq 0\ và\ \mu = [\mu_1, \mu_2, \dots, \mu_N]^T \geq 0$ là các biến đối ngẫu Lagrange. Để tìm giải pháp tối ưu, (w, b, ξ) thỏa mãn các đạo hàm của Lagrangian bằng 0.

$$\frac{\partial L}{\partial w} = 0 \Leftrightarrow w = \sum_{n=1}^{N}\lambda_n y_n x_n$$

$$\frac{\partial L}{\partial b} = 0 \Leftrightarrow \sum_{n=1}^{N} \lambda_n y_n = 0$$

$$\frac{\partial L}{\partial \xi_n} = 0 \Leftrightarrow \lambda_n = C - \mu_n$$

Thay các biểu thức này vào Lagrangian ta sẽ thu được hàm đối ngẫu:

$$L(w, b, \xi, \lambda, \mu) = \sum_{n=1}^{N} \lambda_n - \frac{1}{2} \sum_{n=1}^{N} \sum_{m=1}^{N} \lambda_n \lambda_m y_n y_m x_n^T x_m$$

Hàm này không phụ thuộc vào μ, tuy nhiên cần lưu ý ràng buộc $0 \leq \lambda_n \leq C$

Ta giải được λ, từ đó quay lại tìm nghiệm (w, b, ξ) của bài toán gốc, để làm điều này, ta xét hệ điều kiện KKT

$$1 - \xi_n - y_n(w^T x_n + b) \leq 0$$

$$-\xi_n \leq 0$$

$$\lambda_n \geq 0$$

$$\mu_n \geq 0$$

$$\lambda\big(1 - \xi_n - y_n(w^T x_n + b)\big) = 0$$

$$\mu_n \xi_n = 0$$

$$w = \sum_{n=1}^{N} \lambda_n y_n x_n$$

$$\sum_{n=1}^{N} \lambda_n y_n = 0$$

$$\lambda_n = C - \mu_n$$

Ta có một vài quan sát như sau:

- Nếu $\lambda_n = 0$ thì suy ra $\mu_n = C \neq 0$, suy ra $\xi_n = 0$, như vậy không có sự hy sinh nào xảy ra, như vậy x_n nằm trong vùng an toàn.
- Nếu $\lambda_n > 0$, ta có:

$$y_n(w^T x_n + b) = 1 + \xi_n$$

  - Nếu $0 < \lambda_n < C$, suy ra $\mu_n \neq 0$, như vậy $\xi_n = 0$, và $y_n(w^T x_n + b) = 1$, những điểm x_n nằm chính xác trên biên.
  - Nếu $\lambda_n = C$, suy ra $\mu_n = 0$, như vậy ξ_n có thể nhận bất kỳ giá trị nào không âm, nếu $\xi_n \leq 1$, x_n sẽ được phân lớp đúng, và ngược lại.
  - λ_n không thể lớn hơn C vì khi đó$\mu_n < 0$, mâu thuẫn.

Ngoài ra, những điểm tương ứng $0 < \lambda_n < C$ sẽ là các vectơ hỗ trợ, mặc dù những điểm này có thể không nằm trên biên, chúng vẫn là các vectơ hỗ trợ vì có đóng góp trong việc tính toán w thông qua $w = \sum_{n=1}^{N} \lambda_n y_n x_n$.

Đặt $M = \{n: 0 < \lambda_n < C\}$ $và$ $S = \{m: 0 < \lambda_m \leq C\}$. Tức M là tập hợp các chỉ số của các điểm nằm chính xác trên biên – hỗ trợ cho việc tính b, S là tập hợp các chỉ số của các vectơ hỗ trợ trực tiếp cho việc tính w. Các hệ số w, b có thế được xác định bởi:

$$w = \sum_{m \in S} \lambda_m y_m x_m$$

$$b = \frac{1}{N_M} \sum_{n \in M} (y_n - w^T x_n) = \frac{1}{N_M} \sum_{n \in M} (y_n - \sum_{m \in S} \lambda_m y_m x_m^T x_n)$$

Mục đích cuối cũng là xác định nhãn cho một điểm dữ liệu liệu mới, nên ta quan tâm hơn tới cách xác định giá trị của biểu thức sau với điểm dữ liệu x bất kỳ:

$$w^T x + b = \sum_{m \in S} \lambda_m y_m x_m^T x + \frac{1}{N_M} \sum_{n \in M} (y_n - \sum_{m \in S} \lambda_m y_m x_m^T x_n)$$

Với dữ liệu thực tế, rất khó để có dữ liệu gần phân biệt tuyến tính, vì vậy các hàm phân tách tuyến tính khó có thể thực hiện phân loại chính xác. Trong trường hợp này, ý tưởng cơ bản là tìm một phép biến đổi sao cho dữ liệu ban đầu là không phân

biệt tuyến tính được ánh xạ sang không gian mới, ở không gian mới, dữ liệu trở nên phân biệt tuyến tính.

Giả sử rằng có thể tìm được hàm số ϕ(x) sao cho sau khi được biến đổi sang không gian mới, mỗi điểm dữ liệu x trở thành ϕ(x) và trong không gian mới này, dữ liệu trở nên gần phân biệt tuyến tính, lúc này hy vọng nghiệm của bài toán giới thiệu trước đó sẽ cho chúng ta một bộ phân lớp tốt.

Trong không gian mới, bài toán đối ngầu lề mềm sẽ trở thành:

$$\lambda = \arg\max_{\lambda} \sum_{n=1}^{N} \lambda_n - \frac{1}{2} \sum_{n=1}^{N} \sum_{m=1}^{N} \lambda_n \lambda_m y_n y_m \phi(x_n)^T \phi(x_m)$$

Thỏa mãn: $$\sum_{n=1}^{N} \lambda_n y_n = 0$$

$$0 \leq \lambda_n \leq C \ \forall n = 1, 2, \dots, N$$

Và nhãn của một điểm dữ liệu mới được xác định bởi dấu của biểu thức:

$$w^T \phi(x) + b = \sum_{m \in S} \lambda_m y_m \phi(\mathrm{x_m})^T \phi(x) + \frac{1}{N_M} \sum_{n \in M} (y_n - \sum_{m \in S} \lambda_m y_m \phi(x_m)^T \phi(x_n)$$

Tuy nhiên, việc tính toán trực tiếp ϕ(x) cho mỗi điểm dữ liệu có thể sẽ tốn rất nhiều bộ nhớ và thời gian vì số chiều của ϕ(x) thường là rất lớn, có thể là vô hạn. Thêm nữa để tìm nhãn của một điểm dữ liệu mới x, ta lại phải tìm biến đổi của nó ϕ(x) trong không gian rồi mới lấy tích vô hướng của nó với tất cả các $\phi(\mathrm{x_m})$. Để tránh điều này xảy ra ta sử dụng một kỹ thuật gọi là kernel trick. Chúng ta chỉ cần tính được $\phi(\mathrm{x})^T \phi(\mathrm{z})$ dựa trên hai điểm x và z, thay vì trực tiếp tính tọa độ một điểm trong không gian mới, ta đi tính tích vô hướng giữa hai điểm trong không gian mới. Những phương pháp dựa trên kỹ thuật này được gọi chung là kernel method.

Lúc này, bằng các định nghĩa hàm kernel $k(x, z) = \phi(\mathrm{x})^T \phi(\mathrm{z})$, ta có thể viết bài toán trên lại như sau:

$$\lambda = \arg\max_{\lambda} \sum_{n=1}^{N} \lambda_n - \frac{1}{2} \sum_{n=1}^{N} \sum_{m=1}^{N} \lambda_n \lambda_m y_n y_m \mathrm{k}(x_n, x_m)$$

Thỏa mãn: $$\sum_{n=1}^{N} \lambda_n y_n = 0$$

$$0 \leq \lambda_n \leq C \ \forall n = 1, 2, \dots, N$$

Và

$$\sum_{m \in S} \lambda_m y_m \mathrm{k}(\mathrm{x_m}, x) + \frac{1}{N_M} \sum_{n \in M} (y_n - \sum_{m \in S} \lambda_m y_m \mathrm{k}(\mathrm{x_m}, x_n))$$

Một số hàm kernel thông dụng:

- Linear: $k(x, z) = x^T z$
- Polynomial: $k(x, z) = (r + \gamma x^T z)^d$ với d là một số dương để chỉ bậc của đa thức
- Radial basic funtion: $k(x, z) = \exp(-\gamma \|x - z\|_2^2), \ \ \gamma > 0$
- Sigmoid: $k(x, z) = \tanh(\gamma x^T z + r)$ (hàm tanh có thể được biểu diễn bằng hàm sigmoid như sau $\tanh(x) = 2\sigma(2x) - 1$)

Chương 3: Cài đặt thử nghiệm mô hình hồi quy Logistic

3.1. Xây dựng mô hình hồi quy logistic

Như đã giới thiệu ở chương 2, mô hình hồi quy logistic thường được sử dụng trong phân loại nhị phân, sử dụng hàm sigmoid để đưa ra xác suất. Bằng cách học từ một tập huấn luyện một vectơ trọng số và một hệ số bias.

Hàm mất mát cross-entropy:

$$L(w) = -\frac{1}{m}\sum_{i=1}^{m} y^{(i)} \log\left(\sigma(z^{(i)})\right) + (1 - y^{(i)}) \log\left(1 - \sigma(z^{(i)})\right)$$

Việc học tập các trọng số thực hiện bằng cách tối ưu hàm mất mát với Gradient Descent.

$$w = w - \eta \cdot grad\left(L(w)\right) = w - \eta \frac{1}{m}(\sigma - y)X$$

có $\sigma = \frac{1}{1+e^{-z}}$ và $z = w^T x + b$

Mã nguồn cho mô hình hồi quy logistic được xây dựng với ngôn ngữ lập trình python:

```
import numpy as np

#hàm sigmoid

def sigmoid(z):

    s = 1 / (1 + np.exp(-z))

    return s


# khởi tạo vectơ trọng số và hệ số bias

def initialize_with_zeros(dim):

    w = np.zeros(dim)

    b = 0
```

```
    return w, b


# Tính toán hàm mất mát và Gradient
def propagate(w, b, x, y):
    # số bản ghi dữ liệu
    m = x.shape[0]
    # Tính toán hàm kích hoạt sigmoid
    A = sigmoid(np.dot(x, w) + b)
    # Tính toán hàm mất mát
    loss = (-1 / m) * (np.dot(y, np.log(A).T) + np.dot((1 - y), np.log(1 - A).T))
    loss = np.squeeze(loss)
    # Tính toán Gradident
    dw = (1 / m) * np.dot((A - y).T, x)
    db = (1 / m) * np.sum(A - y)
    grad = {'dw': dw, 'db': db}
    return grad, loss


# Hàm cập nhật tham số, tối ưu hóa bằng cách chạy thuật toán Gradient Descent
def optimize(w, b, x, y, number_of_iterations, learning_rate, print_loss = False):
    # danh sách lưu trữ lịch sử hàm mất mát
    loss_history = []
    # lặp lại và tối ưu hóa các tham số
```

```
for i in range(number_of_iterations):
    # tính hàm mất mát và gradient
    grad, loss = propagate(w, b, x, y)
    # lấy giá trị
    dw = grad['dw']
    db = grad['db']
    #khi tham số không thay đổi quá nhiều thì ta dừng vòng lặp
    if (np.linalg.norm(dw)**2 + np.linalg.norm(db)**2) < 1e-6:
        break;
    # cập nhật tham số
    w -= learning_rate * dw
    b -= learning_rate * db
    # mất mát sau mỗi 500 lần lặp
    if i % 500 == 0:
        loss_history.append(loss)
    # in ra mất mát sau mỗi 500 lần lặp
    if print_loss and i % 500 == 0:
        print('Loss after {0}: {1}'.format(i, loss))
# lưu lại tham số đã cập nhật và gradient
params = {'w': w, 'b': b}
grad = {'dw': dw, 'db': db}
return params, grad, loss_history
```

```
# hàm dự đoán
def predict(w, b, x):
    # Lấy số bản ghi dữ liệu đầu vào
    m = x.shape[0]
    # tạo vectơ lưu kết quả
    y_prediction = np.zeros(m)
    w = w.reshape(x.shape[1], 1)
    # Tính toán xác suất
    A = sigmoid(np.dot(x, w) + b)
    # Chuyển xác suất sang 0 và 1
    for i in range(A.shape[0]):
        if A[i, 0] >= 0.5:
            y_prediction[i] = 1
        else:
            y_prediction[i] = 0
    return y_prediction

# Tạo mô hình
def model(x_train, y_train, x_test, y_test, number_of_iterations = 5000, learning_rate = 0.05, print_loss = False):
    # khởi tạo tham số (trọng số w và hệ số bias b)
```

```
    w, b = initialize_with_zeros(x_train.shape[1])
    # Tối ưu hóa với Gradient Descent
    params, grad, loss_history = optimize(w, b, x_train, y_train, number_of_iterations, learning_rate, print_loss)
    # lấy trọng số đã tính
    w = params['w']
    b = params['b']
    # dữ đoán cho tập train và test
    y_prediction_train = predict(w, b, x_train)
    y_prediction_test = predict(w, b, x_test)
    #một số thông số mô hình
    d = {'y_prediction_train' : y_prediction_train,
        'y_prediction_test': y_prediction_test,
        'w' : w,
        'b' : b,
        'learning_rate' : learning_rate,
        'number_of_iterations': number_of_iterations}
    return d
```

3.2. Bài toán thử nghiệm

Trong thực tế hoạt động, các công ty, hệ thống thường xảy ra tình trạng mất mát khách hàng. Có nhiều nguyên nhân dẫn đến sự ra đi của khách hàng. Điều này khiến các nhà quản lí luôn phải tìm ra kế hoạch để giữ chân khách hàng . Nhưng vấn đề là có quá nhiều khách hàng, vậy làm sao để nhà quản lí biết cần quan tâm khách

hàng nào hơn. Giải pháp: cần có một hệ thống có thể gợi ý cho nhà quản lí về số người cũng như khả năng họ sẽ rời khỏi hệ thống là bao nhiêu. Để làm được điều này, ta cần phải xây dựng được một mô hình giúp dự đoán khả năng rời đi của khách hàng.

Dữ liệu thử nghiệm được lấy từ một nhà khai thác viễn thông không dây về sự rời đi của khác hàng sử dụng dịch vụ từ trang Kaggle (https://www.kaggle.com/mahreen/sato2015), bộ dữ liệu gồm 14 trường thuộc tính với hai nhãn mục tiêu đại diện cho sự rời bỏ và còn hoạt động của khách hàng.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N
1	network_i	Aggregate	Aggregate	Aggregate	Aggregate	Aggregate	Aggregate	aug_user_	sep_user_	aug_fav_a	aug_fav_a	sep_fav_a	sep_fav_a	Class
2	1914	1592.72	23.26	2.5	11.6113	25523	99000	0	0	0	0	1	0	Churned
3	2073	1404.15	174.45	27.5	2531.725	14584	77299	0	0	1	0	0	1	Churned
4	3139	85.5504	14.34	5	29133.06	477	4194	0	0	0	0	0	0	Churned
5	139	2315.229	19.25	52.5	267441.3	50316	52400	0	0	0	0	0	1	Active
6	143	973.9664	21.86	22.5	920871.1	4032	15476	1	1	1	0	0	1	Active
7	174	457.6752	96.93	12.5	191.5703	708	22437	1	1	0	1	1	0	Churned
8	431	32.68	15.45	0	6.3379	0	336	0	0	1	0	1	0	Churned
9	443	1166.4	0	28.75	32954801	2400	1306	1	1	0	0	0	1	Active
10	619	212.41	80.86	2.5	38.6113	3972	2339	0	0	0	0	1	0	Active
11	924	58.89	11.95	23.75	1044714	0	645	1	1	0	0	0	0	Churned
12	941	69.998	9.56	2.5	28.8105	192	3438	0	0	0	0	0	0	Churned
13	1036	3078.39	0	0	103607.3	0	33766	0	0	0	0	0	1	Active
14	1061	423.0128	4.79	2.5	381433.8	9552	10846	0	0	0	0	0	1	Active
15	1142	817.08	10.5	25	69186.74	15228	7656	1	1	0	0	0	1	Active
16	1207	1753.16	33.38	126.25	1064996	10542	63070	1	1	0	0	0	1	Churned
17	1239	686.298	22.74	176.25	1322.06	6948	13334	1	1	0	0	0	0	Active
18	1251	514.884	45.88	3.75	6.0098	708	38807	0	0	0	0	1	0	Churned
19	1265	359.54	1.75	6.25	57359.46	4980	8960	1	1	0	0	0	1	Active
20	1326	3658.94	31.35	41.25	22332.03	13920	171215	1	1	0	0	1	0	Active
21	1355	875.4	0	0	144766.5	0	0	0	0	0	0	0	1	Active
22	1370	983.35	89.58	21.25	112779.6	11688	48900	1	1	0	1	0	1	Churned

Hình 3.1: dữ liệu thử nghiệm cho sự rời đi của khách hàng

Ta sẽ đi sử dụng mô hình hồi quy logistic để giải quyết bài toán phân loại nhị phân này.

3.3. Kết quả

Áp dụng mô hình hồi quy logistic đã xây dựng vào bài toán trên, ta được kết quả khi xây dựng mô hình như sau:

```
print('Trọng số w: ', model['w'], '\nHệ số bias b:', model['b'])
report =  metrics.classification_report(Y_test,model['y_prediction_test'],digits=4)

print(report)
matrix = metrics.confusion_matrix(Y_test,model['y_prediction_test'])
print(matrix)
```

```
Trọng số w:  [ -1.37603423 -10.14624261   3.34188163   0.91382048  -5.8590799
  -1.65798446   4.03245069  -0.08473571  -0.12097297   0.13143698
   0.18265305  -0.32344066  -1.36910606]
Hệ số bias b: 1.4921503299007532
              precision    recall  f1-score   support

           0     0.6763    0.7550    0.7135       249
           1     0.7448    0.6642    0.7022       268

    accuracy                         0.7079       517
   macro avg     0.7105    0.7096    0.7078       517
weighted avg     0.7118    0.7079    0.7076       517

[[188  61]
 [ 90 178]]
```

Hình 3.2: Kết quả khi sử dụng mô hình xây dựng ở mục 3.1

Mô hình đã xây dựng cho độ chính xác 70,79% và với các hệ số như trên, ta thử sử dụng thư viện có sẵn của python để kiểm tra.

Sử dụng mô hình hồi quy logistic trong thư viện sklearn của python ta được kết quả

```
model1 = LogisticRegression()
model1.fit(X_train,Y_train)
print('Trọng số w: ', model1.coef_, '\nHệ số bias b:', model1.intercept_)
y_pred = model1.predict(X_test)
report =  metrics.classification_report(Y_test,y_pred,digits=4)

print(report)
matrix = metrics.confusion_matrix(Y_test,y_pred)
print(matrix)
```

```
Trọng số w:  [[-1.27532917 -2.85916916  1.71300104  0.1360953  -2.21087343 -0.61322196
   1.16228276 -0.13879367 -0.17184651  0.15453264  0.20369242 -0.24714426
  -1.46288833]]
Hệ số bias b: [1.42000719]
              precision    recall  f1-score   support

           0     0.6690    0.7791    0.7199       249
           1     0.7577    0.6418    0.6949       268

    accuracy                         0.7079       517
   macro avg     0.7133    0.7105    0.7074       517
weighted avg     0.7150    0.7079    0.7069       517

[[194  55]
 [ 96 172]]
```

Hình 3.3: Kết quả khi sử dụng mô hình trong thư viện sklearn có sẵn

Như đã thấy, mặc dù có một chút khác biệt trong hệ số và khả năng dự đoán của hai mô hình, tuy nhiên hai mô hình cho ra độ chính xác khá tương đồng với nhau. Mặc dù khi xây dựng mô hình giải quyết các bài toán người ta thường dùng thư viện có sẵn, tuy nhiên việc xây dựng lại mô hình giúp ta có cái nhìn chi tiết hơn, giúp ta hiểu rõ hơn mô hình ta xây dựng lên.

Kết luận

Trong đồ án này, em đã trình bày tổng quan về hệ hỗ trợ quyết định, về khái niệm, ra quyết định, quá trình ra quyết định cũng như mô hình trong hệ hỗ trợ ra quyết định. Em cũng đã trình bày về vấn đề phân loại và một số mô hình phân loại như hồi quy logistic, cây quyết định, phân loại Bayes, máy vectơ hỗ trợ, về xây dựng các mô hình dựa. Phần cuối, em có thực hiện cài đặt mô hình hồi quy logistic và sử dụng nó để thử nghiệm với bài toán đặt ra là xây dựng mô hình dự đoán khả năng rời đi của khách hàng viễn thông với bộ dữ liệu lấy trên Kaggle. Kết quả thử nghiệm cho thấy việc cài đặt lại mô hình và sử dụng mô hình có sẵn trong thư viện sklearn của python khá là tương đồng về độ chính xác mặc dù có chút khác biệt về trọng số tìm được. Hướng mở rộng nghiên cứu, phát triển: cài đặt các mô hình phân loại khác và áp dụng chúng vào các bài toán thực tế trong việc xây dựng hệ hỗ trợ quyết định cũng như các vấn đề khác liên quan.

Tài liệu tham khảo

[1] Vũ Hữu Tiệp, *Machine learning cơ bản*, Nhà xuất bản Khoa học và Kĩ thuật, Hà Nội, 2018.

[2] Carlo Vercellis, Business Intelligence: *Data Mining and Optimization for Decision Making*, John Wiley & Sons, 2009.

[3] David Hosmer, Stanley Lemeshow, Rodney Sturdivant, *Applied Logistic Regression*, Wiley, 2013.

[4] Joseph M**.** Hilbe, *Logistic regression model*, CRC Press, 2009.

[5] Krzysztof Grąbczewski, *Meta-Learning in Decision Tree Induction*, Springer International Publishing, 2014.

[6] Ramesh Sharda • Dursun Delen • Efraim Turban, *Business Intelligence and Analytics: Systems for Decision Support*, Pearson Education Limited, 2014.

[7] Suykens • Johan A.K., *Regularization, Optimization, Kernels, and Support Vector Machines*, CRC Press, 2014.