TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT VĨNH LONG

KHOA CÔNG NGHỆ THÔNG TIN

TÀI LIỆU GIẢNG DẠY

KHAI PHÁ DỮ LIỆU

(DATA MINING)

Mã HP: TH1346

Số tín chỉ: 3

Vĩnh Long, năm 2022

MỤC LỤC

CHƯƠNG 1: TỔNG QUAN VỀ KHAI PHÁ DỮ LIỆU

1.1 Các khái niệm cơ bản

1.1.1 Khai phá dữ liệu (Data mining)

Công nghệ thông tin, Internet, Intranet, kho dữ liệu, cùng với công nghệ lưu trữ tiên tiến hiện nay đã tạo điều kiện cho các doanh nghiệp, các tổ chức thu thập và sở hữu được khối lượng thông tin khổng lồ. Để khai thác hiệu quả nguồn thông tin từ các cơ sở dữ liệu lớn đó để hỗ trợ cho tiến trình ra quyết định, bên cạnh các phương pháp khai thác thông tin truyền thống, các nhà nghiên cứu đã phát triển các phương pháp, kỹ thuật và phần mềm mới để hỗ trợ tiến trình khám phá, phân tích và tổng hợp thông tin.

Theo đánh giá của IBM, các phương pháp khai thác thông tin truyền thống chỉ thu được khoảng 80% thông tin từ cơ sở dữ liệu, phần còn lại bao gồm các thông tin mang tính khái quát, thông tin có quy luật vẫn đang còn tiềm ẩn bên trong dữ liệu. Lượng thông tin này tuy nhỏ nhưng là thông tin cốt lõi và cần thiết cho tiến trình ra quyết định.

Khai phá dữ liệu là tiến trình khám phá tri thức tiềm ẩn trong cơ sở dữ liệu. Cụ thể hơn, đó là tiến trình trích lọc, sản sinh những tri thức hoặc các mẫu tiềm ẩn, chưa biết nhưng hữu ích từ các cơ sở dữ liệu lớn.

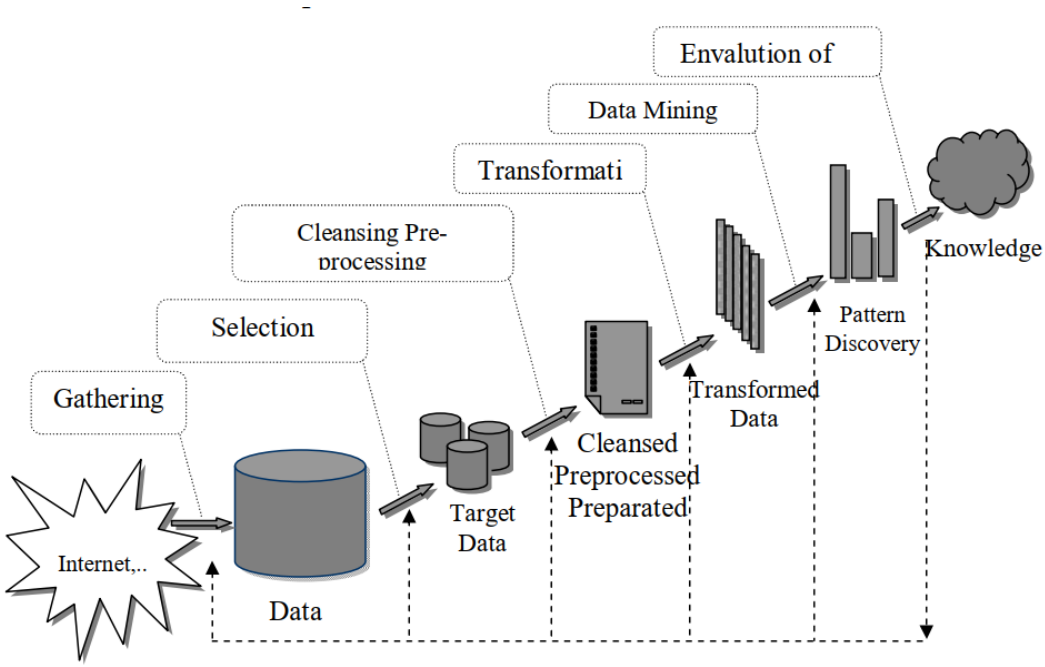
Khai phá dữ liệu là tiến trình khái quát các sự kiện rời rạc trong dữ liệu thành các tri thức mang tính khái quát, tính quy luật hỗ trợ tích cực cho các tiến trình ra quyết định.

Nguồn dữ liệu phục vụ cho khai phá dữ liệu có thể là các cơ sở dữ liệu lớn hay các kho dữ liệu có hoặc không có cấu trúc. Nói như vậy không có nghĩa là khai phá dữ liệu không thể thực hiện ở các cơ sở dữ liệu nhỏ. Khai phá dữ liệu chỉ thực sự phát huy tác dụng trên các cơ sở dữ liệu lớn, nơi mà khả năng diễn dịch và trực giác của con người cũng như các kỹ thuật truyền thống không thể thực hiện nổi hoặc nếu thực hiện được thì hiệu quả không cao.

1.1.2 Lịch sử phát triển khai phá dữ liệu

- **Thập niên 1960:** Xuất hiện cơ sở dữ liệu theo mô hình mạng và mô hình phân cấp.
- **Thập niên 1970:** Thiết lập nền tảng lý thuyết cho cơ sở dữ liệu quan hệ, các hệ quản trị cơ sở dữ liệu quan hệ.
- **Thập niên 1980:** Hoàn thiện lý thuyết về cơ sở dữ liệu quan hệ và các hệ quản trị
- cơ sở dữ liệu quan hệ, xuất hiện các hệ quản trị cơ sở dữ liệu cao cấp (hướng đối tượng, suy diễn,...) và hệ quản trị cơ sở dữ liệu hướng ứng dụng trong lĩnh vực không gian, khoa học, công nghiệp, nông nghiệp, địa lý, ...
- **Thập niên từ 1990 đến 2000:** Phát triển khai phá dữ liệu và kho dữ liệu, cơ sở dữ liệu đa phương tiện và cơ sở dữ liệu web.
- Khai phá dữ liệu là một công đoạn trong tiến trình khám phá tri thức từ cơ sở dữ liệu (Knowledge Discovery in Database - KDD). Khai phá dữ liệu mang tính trực giác, cho phép thu được những hiểu biết rõ ràng và sâu sắc hơn, vượt xa kho dữ liệu.
- Khai phá dữ liệu giúp phát hiện những xu thế phát triển từ những thông tin quá khứ, cũng như cho phép đề xuất các dự báo mang tính thống kê, gom cụm và phân loại dữ liệu.

1.1.3 Các bước chính trong khám phá tri thức và khai phá dữ liệu



Hình 1.1 – Quá trình khám phá tri thức

Quá trình khám phá tri thức từ CSDL là một quá trình có sử dụng nhiều phương pháp và công cụ tin học nhưng vẫn là một quá trình mà trong đó con người là trung tâm. Do đó, nó không phải là một hệ thống phân tích tự động mà là một hệ thống bao gồm nhiều hoạt động tương tác thường xuyên giữa con người và CSDL, tất nhiên là với sự hỗ trợ của các công cụ tin học. Người sử dụng hệ thống ở đây phải là người có kiến thức cơ bản về lĩnh vực cần phát hiện tri thức để có thể chọn được đúng các tập con dữ liệu, các lớp mẫu phù hợp và đạt tiêu chuẩn quan tâm so với mục đích. Tri thức mà ta nói ở đây là các tri thức rút ra từ các CSDL, thường để phục vụ cho việc giải quyết một loạt nhiệm vụ nhất định trong một lĩnh vực nhất định. Do đó, quá trình phát hiện tri thức cũng mang tính chất hướng nhiệm vụ, không phải là phát hiện mọi tri thức bất kỳ mà là phát hiện tri thức nhằm giải quyết tốt nhiệm vụ đề ra.

– Gom dữ liệu (Gathering): Tập hợp dữ liệu là bước đầu tiên trong quá trình khai phá dữ liệu. Đây là bước được khai thác trong một CSDL, một kho dữ liệu và thậm chí các dữ liệu từ các nguồn ứng dụng Web.

- Trích lọc dữ liệu (Selection): Ở giai đoạn này lựa chọn những dữ liệu phù hợp với nhiệm vụ phân tích trích rút từ CSDL.
- Làm sạch, tiền xử lý và chuẩn bị trước dữ liệu (Cleansing, Pre-processing and Preparation) Giai đoạn thứ ba này là giai đoạn hay bị sao lãng, nhưng thực tế nó là một bước rất quan trọng trong quá trình khai phá dữ liệu. Một số lỗi thường mắc phải trong khi gom dữ liệu là tính không đủ chặt chẻ, logic. Vì vậy, dữ liệu thường chứa các giá trị vô nghĩa và không có khả năng kết nối dữ liệu, ví dụ: điểm = -1. Giai đoạn này sẽ tiến hành xử lý những dạng dữ liệu không chặt chẽ nói trên. Những dữ liệu dạng này được xem như thông tin dư thừa, không có giá trị. Bởi vậy, đây là một quá trình rất quan trọng vì dữ liệu này nếu không được "làm sạch" sẽ gây nên những kết quả sai lệch nghiêm trọng.
- Chuyển đổi dữ liệu (Transformation): Tiếp theo là giai đoạn chuyển đổi dữ liệu, dữ liệu được chuyển đổi hay được hợp nhất về dạng thích hợp cho việc khai phá.
- Khai phá dữ liệu (Data Mining): Đây là một tiến trình cốt yếu. Ở giai đoạn này nhiều thuật toán khác nhau đã được sử dụng một cách phù hợp để trích xuất thông tin có ích hoặc cá mẫu điển hình trong dữ liệu
- Đánh giá kết quả mẫu (Evaluation of Result): Đây là giai đoạn cuối trong quá trình khai phá dữ liệu. Ở giai đoạn này, các mẫu dữ liệu được chiết xuất, không phải bất cứ mẫu dữ liệu nào cũng đều hữu ích, đôi khi nó còn bị sai lệch. Vì vậy, cần phải ưu tiên những tiêu chuẩn đánh giá để chiết xuất ra các tri thức cần thiết.
- Từ quá trình khám phá tri thức trên chúng ta thấy được sự khác biệt giữa khám phá tri thức và khai phá dữ liệu. Trong khi khám phá tri thức là nói đến quá trình tổng thể phát hiện tri thức hữu ích từ dữ liệu. Còn KHAI PHÁ DỮ LIỆU chỉ là một bước trong quá trình khám phá tri thức, các công việc chủ yếu là xác định được bài toán khai phá, tiến hành lựa chọn phương pháp KHAI PHÁ DỮ LIỆU phù hợp với dữ liệu có được và tách ra các tri thức cần thiết.

1.1.4 Các dạng dữ liệu có thể khai phá dữ liệu

- Cơ sở dữ liệu quan hệ (relational databases): là những CSDL được tổ chức theo mô hình quan hệ. Hiện nay, các hệ quản trị CSDL đều hỗ trợ mô hình này như: MS Access, MS SQL Server, Oracle, IBM DB2,...
- Cơ sở dữ liệu đa chiều (multidimention structures, data warehouse, data mart): còn được gọi là nhà kho dữ liệu, trong đó dữ liệu được chọn từ nhiều nguồn khác nhau và chứa những đặc tính lịch sử thông qua thuộc tính thời gian tường minh hoặc ngầm định.
- Cơ sở dữ liệu giao tác (transaction databases): là loại dữ liệu được sử dụng nhiều trong siêu thị, thương mại, ngân hàng,...
- Cơ sở dữ liệu quan hệ – hướng đối tượng (object relational databases): mô hình CSDL này là lai giữa mô hình hướng đối tượng và mô hình CSDL quan hệ.
- Cơ sở dữ liệu không gian và thời gian (spatial, temporal, and time – series data): chứa những thông tin về không gian địa lý hoặc thông tin theo thời gian.
- Cơ sở dữ liệu đa phương tiện (Multimedia database): là loại dữ liệu có nhiều trên mạng, bao gồm các loại như âm thanh, hình ảnh, video, văn bản và nhiều kiểu dữ liệu định dạng khác.

1.1.5 Các lĩnh vực liên quan đến khai phá dữ liệu

- Sử dụng dữ liệu để xây dựng các mô hình dự báo:
  - Khả năng dự báo tiềm ẩn trong dữ liệu.
  - Gợi ý về các chiều và các nhóm dữ liệu có khả năng chứa các tri thức hữu ích.
- Tạo tóm tắt và báo cáo rõ ràng:
  - Tự động tìm những phân đoạn trong dữ liệu.
  - Tìm ra những phân đoạn mà nhà phân tích chưa biết hoặc có hiểu biết nhưng chưa rõ ràng.
- Cung cấp cơ chế hỗ trợ ra quyết định:
  - Dự báo.

- Mô hình hóa.

1.2 Các phương pháp, kỹ thuật chính trong khai phá dữ liệu

Các kỹ thuật khai phá dữ liệu được có thể chia làm 2 nhóm chính:

- Kỹ thuật khai phá dữ liệu mô tả: có nhiệm vụ mô tả về các tính chất hoặc các đặc tính chung của dữ liệu trong Cơ sở dữ liệu hiện có. Nhóm kỹ thuật này gồm các phương pháp: phân nhóm (Clustering), tổng hợp hóa (Summerization), phát hiện sự biến đổi và độ lệch (Change and deviation detection), phân tích luật kết hợp (Association Rules), ...
- Kỹ thuật khai phá dữ liệu dự đoán: có nhiệm vụ đưa ra các dự đoán dựa vào các suy diễn trên dữ liệu hiện thời. Nhóm kỹ thuật này gồm các phương pháp: phân lớp (Classification), hồi quy (Regression), ...

1.2.1 Phân lớp và dự đoán (Classification & Prediction)

Là đặt các mẫu vào các lớp được xác định trước. Nhiệm vụ chính là tìm các hàm ánh xạ các mẫu dữ liệu một cách chính xác vào trong các lớp. Ví dụ một ngân hàng muốn phân loại các khách hành của họ vào trong hai nhóm có nợ hay không nợ, từ đó giúp họ ra quyết định cho vay hay không cho vay. Quá trình phân lớp dữ liệu thường gồm 2 bước: xây dựng mô hình và sử dụng mô hình để phân lớp dữ liệu.

- Bước 1: một mô hình sẽ được xây dựng dựa trên việc phân tích các mẫu dữ liệu sẵn có. Mỗi mẫu tương ứng với một lớp, được quyết định bởi một thuộc tính gọi là thuộc tính lớp. Các mẫu dữ liệu này còn được gọi là tập dữ liệu huấn luyện (training data set). Các nhãn lớp của tập dữ liệu huấn luyện đều phải được xác định trước khi xây dựng mô hình, vì vậy phương pháp này còn được gọi là học có giám sát (supervised learning) khác với phân nhóm dữ liệu là học không có giám sát (unsupervised learning)
- Bước 2: sử dụng mô hình để phân lớp dữ liệu. Trước hết chúng ta phải tính độ chính xác của mô hình. Nếu độ chính xác là chấp nhận được, mô hình sẽ được sử dụng để dự đoán nhãn lớp cho các mẫu dữ liệu khác trong tương lai.

Trong kỹ thuật phân lớp chúng ta có thể sử dụng các phương pháp như: Cây quyết định (Decision Tree), K-Láng giềng gần nhất (k-Nearest Neighbor), Mạng Nơron (Neural networks), Giải thuật di truyền (Genetic algorithms), Mạng Bayesian (Bayesian networks), Tập mờ và tập thô (Rough and Fuzzy Sets).

1.2.2 Phân cụm (Clustering)

Mục tiêu chính của việc phân nhóm dữ liệu là nhóm các đối tượng tương tự nhau trong tập dữ liệu vào các nhóm sao cho mức độ tương tự giữa các đối tượng trong cùng một nhóm là lớn nhất và mức độ tương tự giữa các đối tượng nằm trong các nhóm khác nhau là nhỏ nhất. Các nhóm có thể tách nhau hoặc phân cấp gối lên nhau và số lượng các nhóm là chưa biết trước. Một đối tượng có thể vừa thuộc nhóm này, nhưng cũng có thể vừa thuộc nhóm khác. Không giống như phân lớp dữ liệu, phân nhóm dữ liệu không đòi hỏi phải định nghĩa trước các mẫu dữ liệu huấn luyện. Vì thế, có thể coi phân nhóm dữ liệu là một cách học bằng quan sát (learning by observation), trong khi phân lớp dữ liệu là học bằng ví dụ (learning by example). Trong phương pháp này bạn sẽ không thể biết kết quả các nhóm thu được sẽ như thế nào khi bắt đầu quá trình. Vì vậy, thông thường cần có một chuyên gia về lĩnh vực đó để đánh giá các nhóm thu được. Phân nhóm còn được gọi là học không có giám sát (unsupervised learning). Phân nhóm dữ liệu được sử dụng nhiều trong các ứng dụng về phân đoạn thị trường, phân đoạn khách hàng, nhận dạng mẫu, phân loại trang Web, … Ngoài ra phân nhóm dữ liệu còn có thể được sử dụng như một bước tiền xử lý cho các thuật toán khai phá dữ liệu khác.

1.2.3 Luật kết hợp (Assoiation Rule)

Luật kết hợp là dạng luật biểu diễn tri thức ở dạng tương đối đơn giản. Mục tiêu của phương pháp này là phát hiện và đưa ra các mối liên hệ giữa các giá trị dữ liệu trong CSDL. Mẫu đầu ra của giải thuật KPDL là tập luật kết hợp tìm được.

Tuy luật kết hợp là một dạng luật khá đơn giản nhưng lại mang rất nhiều ý nghĩa. Thông tin mà dạng luật này đem lại rất có lợi trong các hệ hỗ trợ ra quyết định. Tìm kiếm được

những luật kết hợp đặc trưng và mang nhiều thông tin từ CSDL tác nghiệp là một trong những hướng tiếp cận chính của lĩnh vực khai phá dữ liệu.

1.2.4 Hồi qui và dự báo (Regression)

Hồi quy là việc học một hàm ánh xạ từ một mẫu dữ liệu thành một biến dự đoán có giá trị thực. Nhiệm vụ của hồi quy tương tự như phân lớp, điểm khác nhau chính là ở chỗ thuộc tính để dự báo là liên tục chứ không rời rạc. Việc dự báo các giá trị số thường được làm bởi các phương pháp thống kê cổ điển chẳng hạn như hồi quy tuyến tính. Tuy nhiên phương pháp mô hình hóa cũng có thể được sử dụng như cây quyết định.

1.2.5 Chuỗi thời gian (sequential/temporal patterns)

Tương tự như khai thác luật kết hợp nhưng có thêm tính thứ tự và tính thời gian. Một luật mô tả mẫu tuần tự có dạng tiêu biểu X → Y phản ánh sự xuất hiện của biến cố X sẽ dẫn đến việc xuất hiện kế tiếp biến cố Y. Hướng tiếp cận này có tính dự báo cao.

1.2.6 Mô tả khái niệm, tổng hợp (concept description & summarization)

Là công việc liên quan đến các phương pháp tìm kiếm một mô tả tập con dữ liệu. Kỹ thuật mô tả khái niệm và tổng hợp hóa thường áp dụng trong việc phân tích dữ liệu có tính thăm dò và báo cáo tự động. Nhiệm vụ chính là sản sinh ra các mô tả đặc trưng cho một lớp. Mô tả loại này là một kiểu tổng hợp, tóm tắt các đặc tính chung của tất cả hay hầu hết các mục của một lớp. Các mô tả đặc trưng thể hiện theo luật có dạng sau: “Nếu một mục thuộc về lớp đã chỉ trong tiền đề thì mục đó có tất cả các thuộc tính đã nêu trong kết luận”.

1.3 Ứng dụng của khai phá dữ liệu

– Khai phá dữ liệu có nhiều ứng dụng trong thực tế, một số ứng dụng điển hình như: Bảo hiểm, tài chính và thị trường chứng khoán: phân tích tình hình tài chính và dự

báo giá của các loại cổ phiếu trong thị trường chứng khoán. Danh mục vốn và giá, lãi suất, dữ liệu thẻ tín dụng, phát hiện gian lận,…

- Điều trị y học và chăm sóc y tế: một số thông tin về chẩn đoán bệnh lưu trong các hệ thống quản lý bệnh viện. Phân tích mối liên hệ giữa triệu chứng bệnh, chẩn đoán và phương pháp điều trị (chế độ dinh dưỡng, thuốc,…).
- Sản xuất và chế biến: qui trình, phương pháp chế biến và xử lý xử cố Text mining & Web mining: phân lớp văn bản và các trang web, tóm tắt văn bản,...
- Lĩnh vực khoa học: quan sát thiên văn, dữ liệu gene, dữ liệu sinh vật học, tìm kiếm, so sánh các hệ gene và thông tin di truyền, mối liên hệ gene và các bệnh di truyền,…
- Lĩnh vực khác: viễn thông, môi trường, thể thao, âm nhạc, giáo dục,…

1.4 Những thách thức trong ứng dụng và kỹ thuật khai phá dữ liệu

1.4.1 Các vấn đề về cơ sở dữ liệu

Đầu vào chủ yếu của một hệ thống khai thác tri thức là các dữ liệu thô trong cơ sở phát sinh trong khai phá dữ liệu chính là từ đây. Do các dữ liệu trong thực tế thường động, không đầy đủ, lớn và bị nhiễu. Trong những trường hợp khác, người ta không biết cơ sở dữ liệu có chứa các thông tin cần thiết cho việc khai thác hay không và làm thế nào để giải quyết với sự dư thừa những thông tin không thích hợp này.

- **Dữ liệu lớn**: Cho đến nay, các cơ sở dữ liệu với hàng trăm trường và bảng, hàng triệu bản ghi và với kích thước đến gigabytes đã là chuyện bình thường. Hiện nay đã bắt đầu xuất hiện hiện các cơ sở dữ liệu có kích thước tới terabytes. Các phương pháp giải quyết hiện nay là đưa ra một ngưỡng cho cơ sở dữ liệu, lấy mẫu, các phương pháp xấp xỉ, xử lý song song (Agrawal et al, Holsheimer et al).
- **Kích thước lớn**: không chỉ có số lượng bản ghi lớn mà số các trường trong cơ sở dữ liệu cũng nhiều. Vì vậy mà kích thước của bài toán trở nên lớn hơn. Một tập dữ liệu có kích thước lớn sinh ra vấn đề làm tăng không gian tìm kiếm mô hình suy diễn. Hơn nữa, nó cũng làm tăng khả năng một giải thuật khai phá dữ liệu có thể tìm thấy các mẫu giả. Biện pháp khắc phục là làm giảm kích thước tác động của bài toán và sử dụng các tri thức biết trước để xác định các biến không phù hợp.

- **Dữ liệu động**: Đặc điểm cơ bản của hầu hết các cơ sở dữ liệu là nội dung của chúng thay đổi liên tục. Dữ liệu có thể thay đổi theo thời gian và việc khai phá dữ liệu cũng bị ảnh hưởng bởi thời điểm quan sát dữ liệu. Ví dụ trong cơ sở dữ liệu về tình trạng bệnh nhân, một số giá trị dữ liệu là hằng số, một số khác lại thay đổi liên tục theo thời gian (ví dụ cân nặng và chiều cao), một số khác lại thay đổi tùy thuộc vào tình huống và chỉ có giá trị được quan sát mới nhất là đủ (ví dụ nhịp đập của mạch). Vậy thay đổi dữ liệu nhanh chóng có thể làm cho các mẫu khai thác được trước đó mất giá trị. Hơn nữa, các biến trong cơ sở dữ liệu của ứng dụng đã cho cũng có thể bị thay đổi, bị xóa hoặc là tăng lên theo thời gian. Vấn đề này được giải quyết bằng các giải pháp tăng trưởng để nâng cấp các mẫu và coi những thay đổi như là cơ hội để khai thác bằng cách sử dụng nó để tìm kiếm các mẫu bị thay đổi.
- **Các trường không phù hợp**: Một đặc điểm quan trọng khác là tính không thích hợp của dữ liệu, nghĩa là mục dữ liệu trở thành không thích hợp với trọng tâm hiện tại của việc khai thác. Một khía cạnh khác đôi khi cũng liên quan đến độ phù hợp là tính ứng dụng của một thuộc tính đối với một tập con của cơ sở dữ liệu. Ví dụ trường số tài khoản Nostro không áp dụng cho các tác nhân.
- **Các giá trị bị thiếu**: Sự có mặt hay vắng mặt của giá trị các thuộc tính dữ liệu phù hợp có thể ảnh hưởng đến việc khai phá dữ liệu. Trong hệ thống tương tác, sự thiếu vắng dữ liệu quan trọng có thể dẫn đến việc yêu cầu cho giá trị của nó hoặc kiểm tra để xác định giá trị của nó. Hoặc có thể sự vắng mặt của dữ liệu được coi như một điều kiện, thuộc tính bị mất có thể được coi như một giá trị trung gian và là giá trị không biết.
- **Các trường bị thiếu**: Một quan sát không đầy đủ cơ sở dữ liệu có thể làm cho các dữ liệu có giá trị bị xem như có lỗi. Việc quan sát cơ sở dữ liệu phải phát hiện được toàn bộ các thuộc tính có thể dùng để giải thuật khai phá dữ liệu có thể áp dụng nhằm giải quyết bài toán. Giả sử ta có các thuộc tính để phân biệt các tình huống đáng quan tâm. Nếu chúng không làm được điều đó thì có nghĩa là đã có lỗi trong dữ liệu. Đối với một hệ thống học để chuẩn đoán bệnh sốt rét từ một cơ sở dữ liệu bệnh nhân thì trường hợp các bản ghi của bệnh nhân có triệu chứng giống nhau

nhưng lại có các chẩn đoán khác nhau là do trong dữ liệu đã bị lỗi. Đây cũng là vấn đề thường xảy ra trong cơ sở dữ liệu kinh doanh. Các thuộc tính quan trọng có thể sẽ bị thiếu nếu dữ liệu không được chuẩn bị cho việc khai phá dữ liệu.

- **Độ nhiễu và không chắc chắn**: Đối với các thuộc tính đã thích hợp, độ nghiêm trọng của lỗi phụ thuộc vào kiểu dữ liệu của các giá trị cho phép. Các giá trị của các thuộc tính khác nhau có thể là các số thực, số nguyên, chuỗi và có thể thuộc vào tập các giá trị định danh. Các giá trị định danh này có thể sắp xếp theo thứ tự từng phần hoặc đầy đủ, thậm chí có thể có cấu trúc ngữ nghĩa.

  Một yếu tố khác của độ không chắc chắn chính là tính kế thừa hoặc độ chính xác mà dữ liệu cần có, nói cách khác là độ nhiễu crên các phép đo và phân tích có ưu tiên, mô hình thống kê mô tả tính ngẫu nhiên được tạo ra và được sử dụng để định nghĩa độ mong muốn và độ dung sai của dữ liệu. Thường thì các mô hình thống kê được áp dụng theo cách đặc biệt để xác định một cách chủ quan các thuộc tính để đạt được các thống kê và đánh giá khả năng chấp nhận của các (hay tổ hợp các) giá trị thuộc tính. Đặc biệt là với dữ liệu kiểu số, sự đúng đắn của dữ liệu có thể là một yếu tố trong việc khai phá. Ví dụ như trong việc đo nhiệt độ cơ thể, ta thường cho phép chênh lệch 0.1 độ. Nhưng việc phân tích theo xu hướng nhạy cảm nhiệt độ của cơ thể lại yêu cầu độ chính xác cao hơn. Để một hệ thống khai thác có thể liên hệ đến xu hướng này để chuẩn đoán thì lại cần có một độ nhiễu trong dữ liệu đầu vào.
- **Mối quan hệ phức tạp giữa các trường**: các thuộc tính hoặc các giá trị có cấu trúc phân cấp, các mối quan hệ giữa các thuộc tính và các phương tiện phức tạp để diễn tả tri thức về nội dung của cơ sở dữ liệu yêu cầu các giải thuật phải có khả năng sử dụng một cách hiệu quả các thông tin này. Ban đầu, kỹ thuật khai phá dữ liệu chỉ được phát triển cho các bản ghi có giá trị thuộc tính đơn giản. Tuy nhiên, ngày nay người ta đang tìm cách phát triển các kỹ thuật nhằm rút ra mối quan hệ giữa các biến này.

1.4.2 Một số vấn đề khác

- **"Quá phù hợp" (Overfitting):** Khi một giải thuật tìm kiếm các tham số tốt nhất cho đó sử dụng một tập dữ liệu hữu hạn, nó có thể sẽ bị tình trạng "quá độ" dữ liệu (nghĩa là tìm kiếm quá mức cần thiết gây ra hiện tượng chỉ phù hợp với các dữ liệu đó mà không có khả năng đáp ứng cho các dữ liệu lạ), làm cho mô hình hoạt động rất kém đối với các dữ liệu thử. Các giải pháp khắc phục bao gồm đánh giá chéo (cross-validation), thực hiện theo nguyên tắc nào đó hoặc sử dụng các biện pháp thống kê khác.
- **Đánh giá tầm quan trọng thống kê:** Vấn đề (liên quan đến overfitting) xảy ra khi một hệ thống tìm kiếm qua nhiều mô hình. Ví dụ như nếu một hệ thống kiểm tra N mô hình ở mức độ quan trọng 0,001 thì với dữ liệu ngẫu nhiên trung bình sẽ có N/1000 mô hình được chấp nhận là quan trọng. Để xử lý vấn đề này, ta có thể sử dụng phương pháp điều chỉnh thống kê trong kiểm tra như một hàm tìm kiếm, ví dụ như điều chỉnh Bonferroni đối với các kiểm tra độc lập.
- **Khả năng biểu đạt của mẫu:** Trong rất nhiều ứng dụng, điều quan trọng là những điều khai thác được phải cáng dễ hiểu với con người càng tốt. Vì vậy, các giải pháp thường bao gồm việc diễn tả dưới dạng đồ họa, xây dựng cấu trúc luật với các đồ thị có hướng (Gaines), biểu diễn bằng ngôn ngữ tự nhiên (Matheus et al.) và các kỹ thuật khác nhằm biểu diễn tri thức và dữ liệu.
- **Sự tương tác với người sử dụng và các tri thức sẵn có:** rất nhiều công cụ và phương phá khai phá dữ liệu không thực sự tương tác với người dùng và không dễ dàng kết hợp cùng với các tri thức đã biết trước đó. Việc sử dụng tri thức miền là rất quan trọng trong khai phá dữ liệu. Đã có nhiều biện pháp nhằm khắc phục vấn đề này như sử dụng cơ sở dữ liệu suy diễn để phát hiện tri thức, những tri thức này sau đó được sử dụng để hướng dẫn cho việc tìm kiếm khai phá dữ liệu hoặc sử dụng sự phân bố và xác suất dữ liệu trước đó như một dạng mã hóa tri thức có sẵn.

CHƯƠNG 2: CÁC VẤN ĐỀ TIỀN XỬ LÝ DỮ LIỆU

2.1 Dữ liệu

2.1.1 Tập dữ liệu

- Một tập dữ liệu (dataset) là một tập hợp các đối tượng (object) và các thuộc tính của chúng.
- Mỗi thuộc tính (attribute) mô tả một đặc điểm của một đối tượng.

2.1.2 Các kiểu tập dữ liệu

- Bản ghi (record): Các bản ghi trong cở sở dữ liệu quan hệ. Ma trận dữ liệu.
- Biểu diễn văn bản. Hay dữ liệu giao dịch.,,
- Đồ thị (graph): World wide web. Mạng thông tin, hoặc mạng xã hội
- Dữ liệu có trật tự: Dữ liệu không gian (ví dụ: bản đồ); Dữ liệu thời gian (ví dụ: time-series data); Dữ liệu chuỗi (ví dụ: chuỗi giao dịch).

2.1.3 Các kiểu giá trị thuộc tính

- Kiểu định danh/chuỗi (norminal): không có thứ tự. Ví dụ: Các thuộc tính như Name, Profession, ...
- Kiểu nhị phân (binary): là một trường hợp đặc biệt của kiểu định danh. Tập các giá trị chỉ gồm có 2 giá trị (Y/N, 0/1, T/F).
- Kiểu có thứ tự (ordinal): Integer, Real, -lấy giá trị từ một tập có thứ tự giá trị. Ví dụ: Các thuộc tính lấy giá trị số như: Age, Height,... Hay lấy một tập xác định, thuộc tính Income lấy giá trị từ tập {low, medium, high}.
- Kiểu thuộc tính rời rạc (discrete-valued attributes): có thể là tập các giá trị của một tập hữu hạn. Bao gồm thuộc tính có kiểu giá trị là các số nguyên, nhị phân.
- Kiểu thuộc tính liên tục (continuous – valued attributes): Các giá trị là số thực.

2.1.4 Các đặc tính mô tả của dữ liệu

- Giúp hiểu rõ về dữ liệu có được: chiều hướng chính/trung tâm, sự biến thiên, sự phân bố.
- Sự phân bố của dữ liệu (data dispersion):
  - Giá trị cực tiểu/cực đại (min/max).
  - Giá trị xuất hiện nhiều nhất (mode).
  - Giá trị trung bình (mean).
  - Giá trị trung vị (median).
  - Sự biến thiên (variance) và độ lệch chuẩn (standard deviation).
  - Các ngoại lai (outliers).

2.2 Tiền xử lý dữ liệu

Quá trình tiền xử lý dữ liệu, đầu tiên phải nắm được dạng dữ liệu, thuộc tính, mô tả của dữ liệu thao tác. Sau đó tiếp hành 4 giai đoạn chính: làm sạch, tích hợp, biến đổi, thu giảm dữ liệu.

2.2.1 Làm sạch dữ liệu

Đối với dữ liệu thu thập được, cần xác định các vấn đề ảnh hưởng là cho nó không sạch. Bởi vì, dữ liệu không sạch (có Chứa lỗi, nhiễu, không đầy đủ, có mâu thuẫn) thì các tri thức khám phá được sẽ bị ảnh hưởng và không đáng tin cậy, sẽ dẫn đến các quyết định không chính xác. Do đó, cần gán các giá trị thuộc tính còn thiếu; sửa chữa các dữ liệu nhiễu/lỗi; xác định hoặc loại bỏ các ngoại lai (outliers); giải quyết các mâu thuẫn dữ liệu.

- *Các vấn đề của dữ liệu*
  - Trên thực thế dữ liệu thu có thể chứa nhiễu, lỗi, không hoàn chỉnh, có mâu thuẫn.
  - Không hoàn chỉnh (incomplete): Thiếu các giá trị thuộc tính hoặc thiếu một số thuộc tính. Ví dụ: salary = <undefined>.
  - Nhiễu/lỗi (noise/error): Chứa đựng những lỗi hoặc các mang các giá trị bất thường. Ví dụ: salary = “-525” , giá trị của thuộc tính không thể là một số âm.

- Mâu thuẫn (inconsistent): Chứa đựng các mâu thuẫn (không thống nhất). Ví dụ: salary = “abc” , không phù hợp với kiểu dữ liệu số của thuộc tính salary.

– *Nguồn gốc/lý do của dữ liệu không sạch*
  - Không hoàn chỉnh (incomplete): Do giá trị thuộc tính không có (not available) tại thời điểm được thu thập. Hoặc các vấn gây ra bởi phần cứng, phần mềm, hoặc người thu thập dữ liệu.
  - Nhiễu/lỗi (noise/error): Do việc thu thập dữ liệu, hoăc việc nhập dữ liệu, hoặc việc truyền dữ liệu.
  - Mâu thuẫn (inconsistent): Do dữ liệu được thu thập có nguồn gốc khác nhau. Hoặc vi phạm các ràng buộc (điều kiện) đối với các thuộc tính.

– *Giải pháp khi thiếu giá trị của thuộc tính*
  - Bỏ qua các bản ghi có các thuộc tính thiếu giá trị. Thường áp dụng trong các bài toán phân lớp. Hoặc khi tỷ lệ % các giá trị thiếu đối với các thuộc tính quá lớn.
  - Một số người sẽ đảm nhiệm việc kiểm tra và gán các giá trị thuộc tính còn thiếu, nhưng đòi hỏi chi phí cao và rất tẻ nhạt.
  - Gán giá trị tự động bởi máy tính:

    + Gán giá trị mặc định

    + Gán giá trị trung bình của thuộc tính đó.

    + Gán giá trị có thể xảy ra nhất – dựa theo phương pháp xác suất.

– Giải pháp khi dữ liệu chứa nhiều lỗi
  - Phân khoảng (binning): Sắp xếp dữ liệu và phân chia thành các khoảng (bins) có tần số xuất hiện giá trị như nhau. Sau đó, mỗi khoảng dữ liệu có thể được biểu diễn bằng trung bình, trung vị, hoặc các giới hạn … của các giá trị trong khoảng đó.
  - Hồi quy (regression): Gắn dữ liệu với một hàm hồi quy.
  - Phân cụm (clustering): Phát hiện và loại bỏ các ngoại lai (sau khi đư xác định các cụm).

- Kết hợp giữa máy tính và kiểm tra của con người: Máy tính sẽ tự động phát hiện ra các giá trị nghi ngờ. Các giá trị này sẽ được con người kiểm tra lại.

a) Thiếu giá trị

Hãy xem xét một kho dữ liệu bán hàng và quản lý khách hàng. Trong đó có thể có một hoặc nhiều giá trị mà khó có thể thu thập được ví dụ như thu nhập của khách hàng. Vậy làm cách nào để chúng ta có được các thông tin đó, hãy xem xét các phương pháp sau.

- Bỏ qua các bộ: Điều này thường được thực hiện khi thông tin nhãn dữ liệu bị mất. Phương pháp này không phải lúc nào cũng hiệu quả trừ khi các bộ có chứa một số thuộc tính không thực sự quan trọng.

- Điền vào các giá trị thiếu bằng tay: Phương pháp này thường tốn thời gian và có thể không khả thi cho một tập dữ liệu nguồn lớn với nhiều giá trị bị thiếu.

- Sử dụng các giá trị quy ước để điền vào cho giá trị thiệu: Thay thế các giá trị thuộc tính thiếu bởi cùng một hằng số quy ước, chẳng hạn như một nhãn ghi giá trị "Không biết" hoặc "∞". Tuy vậy điều này cũng có thể khiến cho chương trình khai phá dữ liệu hiểu nhầm trong một số trường hợp và đưa ra các kết luận không hợp lý.

- Sử dụng các thuộc tính có nghĩa là để điền vào cho giá trị thiếu: Ví dụ, ta biết thu nhập bình quân đầu người của một khu vực là 800.000đ, giá trị này có thể được dùng thể thay thế cho giá trị thu nhập bị thiếu của khách hàng trong khu vực đó.

- Sử dụng các giá trị của các bộ cùng thể loại để thay thế cho giá trị thiếu: Ví dụ, nếu khách hàng A thuộc cùng nhóm phân loại theo rủi ro tín dụng với một khách hàng B khác trong khi đó khách hàng này có thông tin thu nhập bình quân. Ta có thể sử dụng giá trị đó để điền vào cho giá trị thu nhập bình quân của khách hàng A .

- Sử dụng giá trị có tỉ lệ xuất hiện cao để điền vào cho các giá trị thiếu.: Điều này có thể xác định bằng phương pháp hồi quy, các công cụ suy luận dựa trên lý thuyết Bayersian hay cây quyết định

b) Dữ liệu nhiễu

Nhiễu dữ liệu là một lỗi ngẫu nhiên hay do biến động của các biến trong quá trình thực hiện, hoặc sự ghi chép nhầm lẫn ko được kiểm soát... Ví dụ cho thuộc tính như giá cá, làm cách nào để có thể làm mịn thuộc tính này để loại bỏ dữ liệu nhiễu. Hãy xem xét các kỹ thuật làm mịn sau:

Mảng lưu giá các mặt hàng: 4, 8, 15, 21, 21, 24, 25, 28, 34

Phân thành các bin:

Bin 1: 4, 8, 15

Bin 2: 21, 21, 24

Bin 3: 25, 28, 34

Làm mịn sử dụng phương pháp trung vị:

Bin 1: 9, 9, 9

Bin 2: 22, 22, 22

Bin 3: 29, 29, 29

Làm mịn biên

Bin 1: 4, 4, 15

Bin 2: 21, 21, 24

Bin 3: 25, 25, 35

a. Binning: Làm mịn một giá trị dữ liệu được xác định thông qua các giá trị xung quanh nó. Ví dụ, các giá trị giá cả được sắp xếp trước sau đó phân thành các dải khác nhau có cùng kích thước 3 (tức mỗi "Bin" chứa 3 giá trị).

- Khi làm mịn trung vị trong mỗi bin, các giá trị sẽ được thay thế bằng giá trị trung bình các giá trị có trong bin.

- Làm mịn biên: các giá trị nhỏ nhất và lớn nhất được xác định và dùng làm danh giới của bin. Các giá trị còn lại của bin sẽ được thay thế bằng một trong hai giá trị trên tùy thuộc vào độ lệch giữa giá trị ban đầu với các giá trị biên đó.

Ví dụ, bin 1 có các giá trị 4, 8, 15 với giá trị trung bình là 9. Do vậy nếu làm mịn trung vị các giá trị ban đầu sẽ được thay thế bằng 9. Còn nếu làm mịn biên giá trị 8 ở gần giá trị 4 hơn nên nó được thay thế bằng 4.

b. Hồi quy:

Phương pháp thường dùng là hồi quy tuyến tính, để tìm ra được một mối quan hệ tốt nhất giữa hai thuộc tính (hoặc các biến), từ đó một thuộc tính có thể dùng để dự đoán thuộc tính khác. Hồi quy tuyến tính đa điểm là một sự mở rộng của phương pháp trên, trong đó có nhiều hơn hai thuộc tính được xem xét, và các dữ liệu tính ra thuộc về một miền đa chiều.

c. Nhóm cụm: Các giá trị tương tự nhau được tổ chức thành các nhóm hay "cụm" trực quan. Các giá trị rơi ra bên ngoài các nhóm này sẽ được xem xét để làm mịn để đưa chúng

2.2.2 Tích hợp dữ liệu (data integration)

- Tích hợp dữ liệu là quá trình trộn dữ liệu từ các nguồn khác nhau vào một kho dữ liệu có sẵn cho quá trình khai phá dữ liệu.
- Khi tích hợp cần xác định thực thể từ nhiều nguồn dữ liệu để tránh dư thừa dữ liệu. Ví dụ: Bill Clinton ≡ B.Clinton.
- Việc dư thừa dữ liệu là thường xuyên xảy ra, khi tích hợp nhiều nguồn. Bởi cùng một thuộc tính (hay cùng một đối tượng) có thể mang các tên khác nhau trong các nguồn (cơ sở dữ liệu) khác nhau. Hay các dữ liệu suy ra được như một thuộc tính trong một bảng có thể được suy ra từ các thuộc tính trong bảng khác. Hay sự trùng lắp các dữ liệu. Các thuộc tính dư thừa có thể bị phát hiện bằng phân tích tương quan giữa chúng.

- Phát hiện và xử lý các mâu thuẫn đối với giá trị dữ liệu: Đối với cùng một thực thể trên thực tế, nhưng các giá trị thuộc tính từ nhiều nguồn khác nhau lại khác nhau. Có thể cách biểu diễn khác nhau, hay mức đánh giá, độ do khác nhau.
- Yêu cầu chung đối với quá trình tích hợp là giảm thiểu (tránh được là tốt nhất) các dư thừa và các mâu thuẫn. Giúp cải thiện tốc độ của quá trình khai phá dữ liệu và nâng cao chất lượng của các kết quả tri thức thu được.

2.2.3 Biến đổi dữ liệu (data transformation)

- Biến đổi dữ liệu là việc chuyển toàn bộ tập giá trị của một thuộc tính sang một tập các giá trị thay thế, sao cho mỗi giá trị cũ tương ứng với một trong các giá trị mới.
- Các phương pháp biến đổi dữ liệu:
  - Làm trơn (smoothing): Loại bỏ nhiễu/lỗi khỏi dữ liệu.
  - Kết hợp (aggregation): Sự tóm tắt dữ liệu, xây dựng các khối dữ liệu.
  - Khái quát hóa (generalization): Xây dựng các phân cấp khái niệm.
  - Chuẩn hóa (normalization): Đưa các giá trị về một khoảng được chỉ định.
  - Chuẩn hóa min-max, giá trị mới nằm khoảng [new_min$_i$, new_max$_i$]

$$v^{new} = \frac{v^{old} - min_i}{max_i - min_i}(new\_max_i - new\_min_i) + new\_min_i$$

  - Chuẩn hóa z-score, với μ_i, σ_i: giá trị trung bình và độ lệch chuẩn của thuộc tính i

$$v^{new} = \frac{v^{old} - \mu_i}{\sigma_i}$$

  - Chuẩn hóa bởi thang chia 10, với j là giá trị số nguyên nhỏ nhất sao cho: $\max(\{v^{new}\}) < 1$

$$v^{new} = \frac{v^{old}}{10^j}$$

- Xây dựng các thuộc tính mới dựa trên các thuộc tính ban đầu.

2.2.4 Thu giảm dữ liệu (data reduction)

- Một kho dữ liệu lớn có thể chứ lượng dữ liệu lên đến terabytes sẽ làm cho quá trình khai phá dữ liệu chạy rất mất thời gian, do đó nên thu giảm dữ liệu
- Việc thu giảm dữ liệu sẽ thu được một biểu diễn thu gọn, mà nó vẫn sinh ra cùng (hoặc xấp xỉ) các kết quả khai phá như tập dữ liệu ban đầu.
- Các chiến lược thu giảm:
  - Giảm số chiều (dimensionality reduction), loại bỏ bớt các thuộc tính không (ít) quan trọng.
  - Giảm lượng dữ liệu (data/numberosity reduction)

    + Kết hợp khối dữ liệu.

    + Nén dữ liệu.

    + Hồi quy.

    + Rời rạc hóa.

CHƯƠNG 3: CÁC PHƯƠNG PHÁP PHÂN LỚP

3.1 Khái niệm cơ bản

3.1.1 Phân lớp

Quá trình phân lớp thực hiện nhiệm vụ xây dựng mô hình các công cụ phân lớp giúp cho việc gán nhưn phân loại cho các dữ liệu. Ví dụ như "An toàn" hoặc "Rủi ro" cho các yêu của vay vốn; "Có" hoặc "Không" cho các thông tin thị trường. Các nhãn dùng phân loại được biểu diễn bằng các giá trị rời rạc trong đó việc sắp xếp trùng là không có ý nghĩa.

Phân lớp dữ liệu gồm hai quá trình. Trong quá trình thứ nhất một công cụ phân lớp sẽ được xây dựng để xem xét nguồn dữ liệu. Đây là quá trình học, trong đó một thuật toán phân lớp được xây dựng bằng cách phân tích hoặc "học" từ tập dữ liệu huấn luyện được xây dựng sẵn bao gồm nhiều bộ dữ liệu. Một bộ dữ liệu X biểu diễn bằng một vector n chiều, $X = (x_1, x_2,\ldots, x_n)$, đây là các giá trị cụ thể của một tập n thuộc tính của nguồn dữ liệu $\{A_1, A_2, \ldots, A_n\}$. Mỗi bộ được giả sử rằng nó thuộc về một lớp định nghĩa trước với các nhãn xác định.

Quá trình đầu tiên của phân lớp có thể được xem như việc xác định ánh dạ hoặc hàm y = f(X), hàm này có thể dự đoán nhãn y cho bộ X. Nghĩa là với mỗi lớp dữ liệu chúng ta cần học (xây dựng) một ánh xạ hoặc một hàm tương ứng

Trong bước thứ hai, mô hình thu được sẽ được sử dụng để phân lớp. Để đảm bảo tính khách quan nên áp dụng mô hình này trên một tập kiểm thử hơn là làm trên tập dữ liệu huấn luyện ban đầu. Tính chính xác của mô hình phân lớp trên tập dữ liệu kiểm thử là số phần trăm các bộ dữ liệu kiểm tra được đánh nhãn đúng bằng cách so sánh chúng với các mẫu trong bộ dữ liệu huấn luyện Nếu như độ chính xác của mô hình dự đoán là chấp nhận được thì chúng ta có thể sử dụng nó cho các bộ dữ liệu với thông tin nhãn phân lớp chưa xác định.

Kỹ thuật phân lớp dữ liệu được tiến hành bao gồm 2 bước:

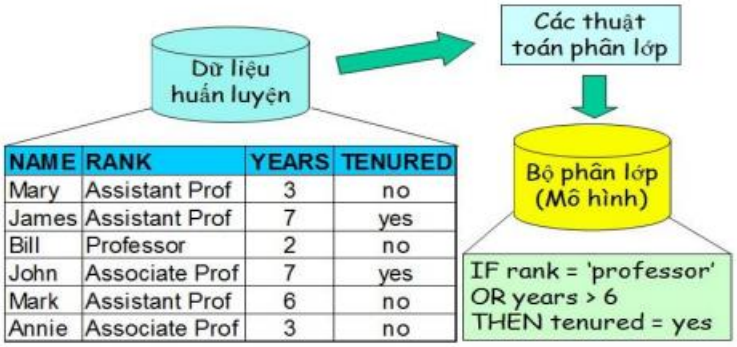
Bước 1: Xây dựng mô hình từ tập huấn luyện

- Mỗi bộ/mẫu dữ liệu được phân vào một lớp được xác định trước.

- Lớp của một bộ/mẫu dữ liệu được xác định bởi thuộc tính gán nhãn lớp

- Tập các bộ/mẫu dữ liệu huấn luyện - tập huấn luyện - được dùng để xây dựng mô hình.

- Mô hình được biểu diễn bởi các luật phân lớp, các cây quyết định hoặc các công thức toán học.



NAME	RANK	YEARS	TENURED
Mary	Assistant Prof	3	no
James	Assistant Prof	7	yes
Bill	Professor	2	no
John	Associate Prof	7	yes
Mark	Assistant Prof	6	no
Annie	Associate Prof	3	no

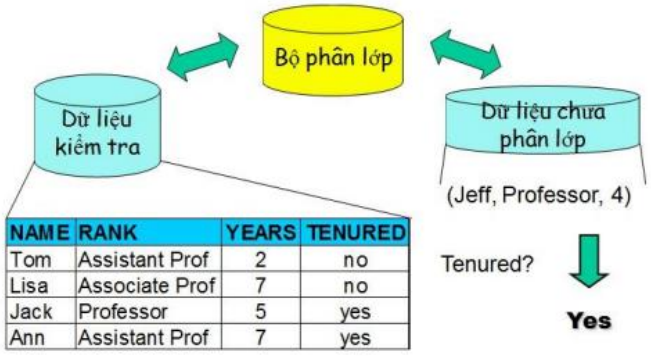
Hình 3.1 – Ví dụ xây dựng mô hình

Bước 2: Sử dụng mô hình – kiểm tra tính đúng đắn của mô hình và dùng nó để phân lớp

- Phân lớp cho những đối tượng mới hoặc chưa được phân lớp

- Đánh giá độ chính xác của mô hình

- Lớp biết trước của một mẫu/bộ dữ liệu đem kiểm tra được so sánh với kết quả thu được từ mô hình.
- Tỉ lệ chính xác bằng phần trăm các mẫu/bộ dữ liệu được phân lớp đúng bởi mô hình trong số các lần kiểm tra.



Hình 3.2 – Sử dụng mô hình

3.1.2 Dự đoán

Dự đoán dữ liệu là một quá trình gồm hai bước, nó gần giống với quá trình phân lớp. Tuy nhiên để dự đoán, chúng ta bỏ qua khái niệm nhãn phân lớp bởi vì các giá trị được dự đoán là liên tục (được sắp xếp) hơn là các giá trị phân loại. Ví dụ thay vì phân loại xem một khoản vay có là an toàn hay rủi ro thì chúng ta sẽ dự đoán xem tổng số tiền cho vay của một khoản vay là bao nhiêu thì khoản vay đó là an toàn.

Có thể xem xét việc dự đoán cũng là một hàm y = f(X), trong đó X là dữ liệu đầu vào, và đầu ra là một giá trị y liên tục hoặc sắp xếp được. Việc dự đoán và phân lớp có một vài điểm khác nhau khi sử dụng các phương pháp xây dựng mô hình. Giống với phân lớp, tập dữ liệu huấn luyện sử dụng để xây dựng mô hình dự đoán không được dùng để đánh giá tính chính xác. Tính chính xác của mô hình dự đoán được đánh giá dựa trên việc tính độ lệch giá các giá trị dự đoán với các giá trị thực sự nhận được của mỗi bộ kiểm tra X.

3.2 Cây quyết định

3.2.1 Sơ lược về cây quyết định

Cuối những năm 70 đầu những năm 80, J.Ross Quinlan đã phát triển một thuật toán sinh cây quyết định. Đây là một tiếp cận tham lam, trong đó nó xác định một cây quyết định được xây dựng từ trên xuống một cách đệ quy theo hướng chia để trị. Hầu hết các

thuật toán sinh cây quyết định đều dựa trên tiếp cận topdown trình bày sau đây, trong đó nó bắt đầu từ một tập các bộ huấn luyện và các nhãn phân lớp của chúng. Tập huấn luyện được chia nhỏ một các đệ quy thành các tập con trong quá trình cây được xây dựng.

Cây quyết định là một mô tả tri thức dạng đơn giản nhằm phân các đối tượng dữ liệu thành một số lớp nhất định. Các nút của cây được gán nhãn là tên của các thuộc tính, các cạnh được gán các giá trị có thể của các thuộc tính, các lá mô tả các lớp khác nhau. Các đối tượng được phân lớp theo các đường đi trên cây, qua các cạnh tương ứng với giá trị của thuộc tính của đối tượng tới lá.

Quá trình xây dựng cây quyết định là quá trình phát hiện ra các luật phân chia tập dữ liệu đã cho thành các lớp đã được định nghĩa trước. Trong thực tế, tập các cây quyết định có thể có đối với bài toán này rất lớn và rất khó có thể duyệt hết được một cách tường tận.

3.2.2 Định nghĩa cây quyết định

Một cây quyết định là một cấu trúc hình cây, trong đó:

- Mỗi đỉnh trong (đỉnh có thể khai triển được) biểu thị cho một phép thử đối với một thuộc tính.

- Mỗi nhánh biểu thị cho một kết quả của phép thử.

- Các đỉnh lá (các đỉnh không khai triển được) biểu thị các lớp hoặc các phân bổ lớp.

- Đỉnh trên cùng trong một cây được gọi là gốc.

3.2.3 Thuật toán ID3

Được phát biểu bởi Quinlan (trường đại học Syney, Australia) và được công bố vào cuối thập niên 70 của thế kỷ 20. Sau đó, thuật toán ID3 được giới thiệu và trình bày trong mục Induction on Decision Trees, Machine learning năm 1986. Quinlan đã khắc phục được hạn chế của thuật toán CLS. ID3 cho cây kết quả tối ưu hơn thuật toán CLS. Khi áp dụng thuật toán ID3 cho cùng một tập dữ liệu đầu vào và thử nhiều lần thì cho

cùng một kết quả bởi vì thuộc tính ứng viên ở mỗi bước trong quá trình xây dựng cây được lựa chọn trước. Tuy nhiên thuật toán này cũng chưa giải quyết được về vấn đề thuộc tính số, liên tục, số lượng các thuộc tính còn bị hạn chế và ID3 làm việc không hiệu quả với dữ liệu bị nhiễu hoặc bị thiếu.

Giải thuật quy nạp cây ID3 (gọi tắt là ID3) là một giải thuật học đơn giản nhưng tỏ ra thành công trong nhiều lĩnh vực. ID3 biểu diễn các khái niệm (concept) ở dạng cây quyết định (decision tree). Biểu diễn này cho phép chúng ta xác định phân loại đối tượng bằng cách kiểm tra các giá trị của nó trên một số thuộc tính nào đó. Như vậy, nhiệm vụ của giải thuật ID3 là học cây quyết định từ tập dữ liệu rèn luyện (training data). Hay nói khác hơn, giải thuật có:

- Đầu vào: Một tập hợp các ví dụ. Mỗi ví dụ bao gồm các thuộc tính mô tả một tình huống, hay một đối tượng nào đó, và một giá trị phân loại của nó.
- Đầu ra: Cây quyết định có khả năng phân loại đúng đắn các ví dụ trong tập dữ liệu rèn luyện, và hy vọng là phân loại đúng cho cả các ví dụ chưa gặp trong tương lai.

Thuật toán ID3 xây dựng cây quyết định dựa vào sự phân lớp các đối tượng (mẫu huấn luyện) bằng cách kiểm tra giá trị các thuộc tính. ID3 xây dựng cây quyết định từ trên xuống (top -down) bắt đầu từ một tập các đối tượng và các thuộc tính của nó. Tại mỗi nút của cây, tiến hành việc kiểm tra các thuộc tính để tìm ra thuộc tính tốt nhất được sử dụng để phân chia tập các đối tượng mẫu, theo các giá trị của thuộc tính được chọn để mở rộng. Quá trình này được thực hiện một cách đệ quy cho đến khi mọi đối tượng của phân vùng đều thuộc cùng một lớp; lớp đó trở thành nút lá của cây. Để làm được việc này thuật toán ID3 có sử dụng tới hai hàm Entropy và Gain.

ID3 được xem là một cải tiến của CLS. Tuy nhiên thuật toán ID3 không có khả năng xử lý đối với những dữ liệu có chứa thuộc tính số - thuộc tính liên tục (numeric attribute) và khó khăn trong việc xử lý các dữ liệu thiếu (missing data) và dữ liệu nhiễu (noisy data).

Trong ID3, tổng có trọng số của entropy tại các leaf-node sau khi xây dựng decision tree được coi là hàm mất mát của decision tree đó. Các trọng số ở đây tỉ lệ với số điểm dữ liệu được phân vào mỗi node. Công việc của ID3 là tìm các cách phân chia hợp lý (thứ tự chọn thuộc tính hợp lý) sao cho hàm mất mát cuối cùng đạt giá trị càng nhỏ càng tốt. Như đã đề cập, việc này đạt được bằng cách chọn ra thuộc tính sao cho nếu dùng thuộc tính đó để phân chia, entropy tại mỗi bước giảm đi một lượng lớn nhất. Bài toán xây dựng một decision tree bằng ID3 có thể chia thành các bài toán nhỏ, trong mỗi bài toán, ta chỉ cần chọn ra thuộc tính giúp cho việc phân chia đạt kết quả tốt nhất. Mỗi bài toán nhỏ này tương ứng với việc phân chia dữ liệu trong một non-leaf node. Chúng ta sẽ xây dựng phương pháp tính toán dựa trên mỗi node này.

Xét một bài toán với C class khác nhau. Giả sử ta đang làm việc với một *non-leaf node* với các điểm dữ liệu tạo thành một tập S với số phần tử là |S|=N. Giả sử thêm rằng trong số N điểm dữ liệu này, N_c, c = 1, 2, …, C điểm thuộc vào class c. Xác suất để mỗi điểm dữ liệu rơi vào một class c được xấp xỉ bằng $\frac{N_c}{N}$ (maximum likelihood estimation). Như vậy, entropy tại node này được tính bởi:

$$H(S) = -\sum_{c=1}^{C} \frac{N_c}{N} \log\left(\frac{N_c}{N}\right)$$

Tiếp theo, giả sử thuộc tính được chọn là x. Dựa trên x, các điểm dữ liệu trong S được phân ra thành K child node S_1, S_2, …, S_K với số điểm trong mỗi child node lần lượt là m_1, m_2, …, m_K. Ta định nghĩa:

$$H(x,S) = \sum_{k=1}^{K} \frac{M_K}{N} \mathrm{H}(S_K)$$

là tổng có trọng số entroy của mỗi child node–được tính tương tự như H(S). Việc lấy trọng số này là quan trọng vì các node thường có số lượng điểm khác nhau.

Tiếp theo, ta định nghĩa information gain dựa trên thuộc tính x:

$$G(x,S) = H(S) - H(x,S)$$

Trong ID3, tại mỗi node, thuộc tính được chọn được xác định dựa trên:

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

tức thuộc tính khiến cho information gain đạt giá trị lớn nhất.

3.2.4 Entropy

Khái niệm entropy của một tập S được định nghĩa trong Lý thuyết thông tin là số lượng mong đợi các bít cần thiết để mã hóa thông tin về lớp của một thành viên rút ra một cách ngẫu nhiên từ tập S. Trong trường hợp tối ưu, mã có độ dài ngắn nhất. Theo lý thuyết thông tin, mã có độ dài tối ưu là mã gán $-\log_2 p$ bits cho thông điệp có xác suất là p.

Trong trường hợp S là tập ví dụ, thì thành viên của S là một ví dụ, mỗi ví dụ thuộc

một lớp hay có một giá trị phân loại.

- Entropy có giá trị nằm trong khoảng [0..1],
- Entropy(S) = 0 ↔ tập ví dụ S chỉ toàn ví dụ thuộc cùng một loại, hay S là thuần nhất.
- Entropy(S) = 1 ↔ tập ví dụ S có các ví dụ thuộc các loại khác nhau với độ pha trộn là cao nhất.
- 0 < Entropy(S) < 1 ↔ tập ví dụ S có số lượng ví dụ thuộc các loại khác nhau là không bằng nhau.

Để đơn giản ta xét trường hợp các ví dụ của S chỉ thuộc loại âm (-) hoặc dương (+)

- p_+ là phần các ví dụ dương trong tập S.
- p_- là phần các ví dụ âm trong tập S.

Khi đó, entropy đo độ pha trộn của tập S theo công thức sau:

$$Entropy(S) = -p_+ log_2 p_+ - p_- log_2 p_-$$

Một cách tổng quát hơn, nếu các ví dụ của tập S thuộc nhiều hơn hai loại, giả sử là có c giá trị phân loại thì công thức entropy tổng quát là:

$$Entorpy(S) = \sum_{i=1}^{c} -p_i log_2 p_i$$

Trong đó: p_i là tỷ lệ mẫu thuộc lớp i trên tập hợp S các mẫu kiểm tra.

3.2.5 Ví dụ:

Để mọi thứ được rõ ràng hơn, chúng ta cùng xem ví dụ với dữ liệu huấn luyện được cho trong Bảng dưới đây. Đây là một bảng dữ liệu được sử dụng rất nhiều trong các bài giảng về decision tree. Bảng dữ liệu này mô tả mối quan hệ giữa thời tiết trong 14 ngày (bốn cột đầu, không tính cột id) và việc một đội bóng có chơi bóng hay không (cột cuối cùng). Nói cách khác, ta phải dự đoán giá trị ở cột cuối cùng nếu biết giá trị của bốn cột còn lại.

id	outlook	temperature	humidity	wind	play
1	sunny	hot	high	weak	no
2	sunny	hot	high	strong	no
3	overcast	hot	high	weak	yes
4	rainy	mild	high	weak	yes
5	rainy	cool	normal	weak	yes
6	rainy	cool	normal	strong	no
7	overcast	cool	normal	strong	yes
8	sunny	mild	high	weak	no
9	sunny	cool	normal	weak	yes
10	rainy	mild	normal	weak	yes
11	sunny	mild	normal	strong	yes
12	overcast	mild	high	strong	yes
13	overcast	hot	normal	weak	yes
14	rainy	mild	high	strong	no

Bảng 3.1 Bảng dữ liệu thời tiết

Có bốn thuộc tính thời tiết:

1. Outlook nhận một trong ba giá trị: sunny, overcast, rainy.
2. Temperature nhận một trong ba giá trị: hot, cool, mild.
3. Humidity nhận một trong hai giá trị: high, normal.

4. Wind nhận một trong hai giá trị: weak, strong.

(Tổng cộng có $3 \times 3 \times 2 \times 2 = 36$ loại thời tiết khác nhau, trong đó 14 loại được thể hiện trong bảng.)

Đây có thể được coi là một bài toán dự đoán liệu đội bóng có chơi bóng không dựa trên các quan sát thời tiết. Ở đây, các quan sát đều ở dạng categorical. Cách dự đoán dưới đây tương đối đơn giản và khá chính xác, có thể không phải là cách ra quyết định tốt nhất:

- Nếu outlook = sunny và humidity = high thì play = no.
- Nếu outlook = rainy và windy = true thì play = no.
- Nếu outlook = overcast thì play = yes.
- Ngoài ra, nếu humidity = normal thì play = yes.
- Ngoài ra, play = yes.

Chúng ta sẽ cùng tìm thứ tự các thuộc tính bằng thuật toán ID3.

Trong 14 giá trị đầu ra ở Bảng trên, có năm giá trị bằng no và chín giá trị bằng yes. Entroy tại root node của bài toán là:

$$H(S) = -\frac{5}{14}\log\left(\frac{5}{14}\right) - \frac{9}{14}\log\left(\frac{9}{14}\right) \approx 0.65$$

Tiếp theo, chúng ta tính tổng có trọng số entropy của các child node nếu chọn một trong các thuộc tính outlook, temperature, humidity, wind, play để phân chia dữ liệu.

Xét thuộc tính outlook. Thuộc tính này có thể nhận một trong ba giá trị sunny, overcast, rainy. Mỗi một giá trị sẽ tương ứng với một child node. Gọi tập hợp các điểm trong mỗi child node này lần lượt là S_s, S_o, S_r với tương ứng m_s, m_o, m_r phần tử. Sắp xếp lại Bảng ban đầu theo thuộc tính outlook ta đạt được ba Bảng nhỏ sau đây.

id	outlook	temperature	humidity	wind	play
1	sunny	hot	high	weak	no
2	sunny	hot	high	strong	no
8	sunny	mild	high	weak	no
9	sunny	cool	normal	weak	yes
11	sunny	mild	normal	strong	yes

id	outlook	temperature	humidity	wind	play
3	overcast	hot	high	weak	yes
7	overcast	cool	normal	strong	yes
12	overcast	mild	high	strong	yes
13	overcast	hot	normal	weak	yes

id	outlook	temperature	humidity	wind	play
4	rainy	mild	high	weak	yes
5	rainy	cool	normal	weak	yes
6	rainy	cool	normal	strong	no
10	rainy	mild	normal	weak	yes
14	rainy	mild	high	strong	no

Bảng 3.2 Bảng sau khi xếp theo giá trị outlook

Quan sát nhanh ta thấy rằng *child node* ứng với *outlook = overcast* sẽ có entropy bằng 0 vì tất cả $m_o = 4$ output đều là *yes*. Hai *child node* còn lại với $m_s = mr_r = 5$ có entropy khá cao vì tần suất output bằng *yes* hoặc *no* là xấp xỉ nhau. Tuy nhiên, hai *child node* này có thể được phân chia tiếp dựa trên hai thuộc tính *humidity* và *wind*.

Bạn đọc có thể kiểm tra được rằng

$$H(\mathcal{S}_s) = -\frac{2}{5}\log\left(\frac{2}{5}\right) - \frac{3}{5}\log\left(\frac{3}{5}\right) \approx 0.673$$

$$H(\mathcal{S}_o) = 0$$

$$H(\mathcal{S}_r) = -\frac{3}{5}\log\left(\frac{2}{5}\right) - \frac{3}{5}\log\left(\frac{3}{5}\right) \approx 0.673$$

$$H(outlook, \mathcal{S}) = \frac{5}{14}H(\mathcal{S}_s) + \frac{4}{14}H(\mathcal{S}_o) + \frac{5}{14}H(\mathcal{S}_r) \approx 0.48$$

Xét thuộc tính temperature, ta có phân chia như các Bảng dưới đây.

id	outlook	temperature	humidity	wind	play
1	sunny	hot	high	weak	no
2	sunny	hot	high	strong	no
3	overcast	hot	high	weak	yes
13	overcast	hot	normal	weak	yes

id	outlook	temperature	humidity	wind	play
4	rainy	mild	high	weak	yes
8	sunny	mild	high	weak	no
10	rainy	mild	normal	weak	yes
11	sunny	mild	normal	strong	yes
12	overcast	mild	high	strong	yes
14	rainy	mild	high	strong	no

id	outlook	temperature	humidity	wind	play
5	rainy	cool	normal	weak	yes
6	rainy	cool	normal	strong	no
7	overcast	cool	normal	strong	yes
9	sunny	cool	normal	weak	yes

Bảng 3.3 Bảng thuộc tính temperature

Gọi S_h, S_m, S_c là ba tập con tương ứng với temperature bằng hot, mild, cool. Bạn đọc có thể tính được

$$H(\mathcal{S}_h) = -\frac{2}{4}\log\left(\frac{2}{4}\right) - \frac{2}{4}\log\left(\frac{2}{4}\right) \approx 0.693$$

$$H(\mathcal{S}_m) = -\frac{4}{6}\log\left(\frac{4}{6}\right) - \frac{2}{6}\log\left(\frac{2}{6}\right) \approx 0.637$$

$$H(\mathcal{S}_c) = -\frac{3}{4}\log\left(\frac{3}{4}\right) - \frac{1}{4}\log\left(\frac{1}{4}\right) \approx 0.562$$

$$H(temperature, \mathcal{S}) = \frac{4}{14}H(\mathcal{S}_h) + \frac{6}{14}H(\mathcal{S}_m) + \frac{4}{14}H(\mathcal{S}_c) \approx 0.631$$

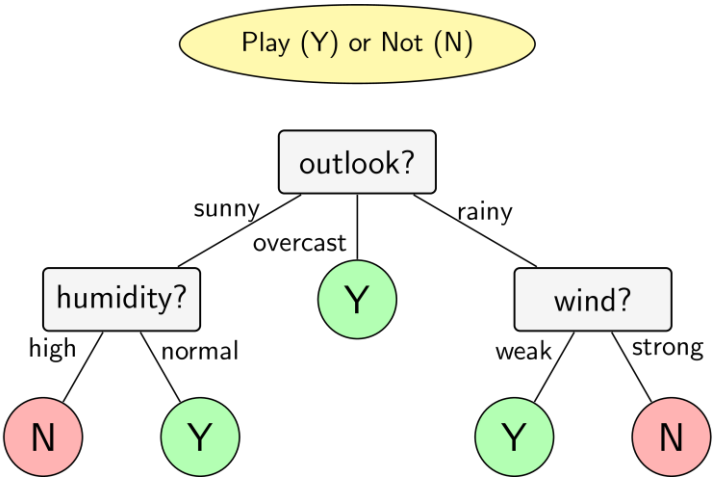
Việc tính toán với hai thuộc tính còn lại được dành cho bạn đọc. Nếu các kết quả là giống nhau, chúng sẽ bằng:

$$H(humidity, \mathcal{S}) \approx 0.547, \quad H(wind, \mathcal{S}) \approx 0.618$$

Như vậy, thuộc tính cần chọn ở bước đầu tiên là outlook vì $H(outlook, S)$ đạt giá trị nhỏ nhất (information gain là lớn nhất).

Sau bước phân chia đầu tiên này, ta nhận được ba child node với các phần tử như trong ba Bảng phân chia theo outlook. Child node thứ hai không cần phân chia tiếp vì nó đã tinh khiết. Với child node thứ nhất, ứng với outlook = sunny, kết quả tính được bằng ID3 sẽ cho chúng ta thuộc tính humidity vì tổng trọng số của entropy sau bước này sẽ bằng 0 với output bằng yes khi và chỉ khi humidity = normal. Tương tự, child node ứng với outlook = wind sẽ được tiếp tục phân chia bởi thuộc tính wind với output bằng yes khi và chỉ khi wind = weak.

Như vậy, cây quyết định cho bài toán này dựa trên ID3 sẽ có dạng như trong Hình 3.3.



Hình 3.3: Decision tree cho bài toán ví dụ sử dụng thuật toán ID3.

3.2.6 Ưu điểm của cây quyết định

– Cây quyết định dễ hiểu. Người ta có thể hiểu mô hình cây quyết định sau khi được giải thích ngắn.

- Việc chuẩn bị dữ liệu cho một cây quyết định là cơ bản hoặc không cần thiết. Các kỹ thuật khác thường đòi hỏi chuẩn hóa dữ liệu, cần tạo các biến phụ (dummy variable) và loại bỏ các giá trị rỗng.
- Cây quyết định có thể xử lý cả dữ liệu có giá trị bằng số và dữ liệu có giá trị là tên thể loại. Các kỹ thuật khác thường chuyên để phân tích các bộ dữ liệu chỉ gồm một loại biến. Chẳng hạn, các luật quan hệ chỉ có thể dùng cho các biến tên, trong khi mạng nơ-ron chỉ có thể dùng cho các biến có giá trị bằng số.
- Cây quyết định là một mô hình hộp trắng. Nếu có thể quan sát một tình huống cho trước trong một mô hình, thì có thể dễ dàng giải thích điều kiện đó bằng logic Boolean. Mạng nơ-ron là một ví dụ về mô hình hộp đen, do lời giải thích cho kết quả quá phức tạp để có thể hiểu được.
- Có thể thẩm định một mô hình bằng các kiểm tra thống kê. Điều này làm cho ta có thể tin tưởng vào mô hình.
- Cây quyết định có thể xử lý tốt một lượng dữ liệu lớn trong thời gian ngắn. Có thể dùng máy tính cá nhân để phân tích các lượng dữ liệu lớn trong một thời gian đủ ngắn để cho phép các nhà chiến lược đưa ra quyết định dựa trên phân tích của cây quyết định.

3.3 Phương pháp phân lớp Naïve Bayes

3.3.1 Định lí Bayes

Trong học máy, phân loại Naive Bayes là một thành viên trong nhóm các phân loại có xác suất dựa trên việc áp dụng định lý Bayes khai thác mạnh giả định độc lập giữa các hàm, hay đặc trưng. Mô hình Naive Bayes cũng được biết đến với nhiều tên khác nhau (Simple Bayes hay independence Bayes hay phân loại Bayes). Phân loại Naive Bayes được đánh giá cao khả năng mở rộng, đòi hỏi một số thông số tuyến tính trong số lượng các biến (các tính năng/ tố dự báo) trong nhiều lĩnh vực khác nhau.

Một phân loại Naive Bayes dựa trên ý tưởng nó là một lớp được dự đoán bằng các giá trị của đặc trưng cho các thành viên của lớp đó. Các đối tượng là một nhóm (group)

trong các lớp nếu chúng có cùng các đặc trưng chung. Có thể có nhiều lớp rời rạc hoặc lớp nhị phân.

Các luật Bayes dựa trên xác suất để dự đoán chúng về các lớp có sẵn dựa trên các đặc trưng được trích rút. Trong phân loại Bayes, việc học được coi như xây dựng một mô hình xác suất của các đặc trưng và sử dụng mô hình này để dự đoán phân loại cho một ví dụ mới. Biến chưa biết hay còn gọi là biến ẩn là một biến xác suất chưa được quan sát trước đó. Phân loại Bayes sử dụng mô hình xác suất trong đó phân loại là một biến ẩn có liên quan tới các biến đã được quan sát. Quá trình phân loại lúc này trở thành suy diễn trên mô hình xác suất.

Trường hợp đơn giản nhất của phân loại Naive Bayes là tạo ra các giả thiết độc lập về các đặc trưng đầu vào và độc lập có điều kiện với mỗi một lớp đã cho. Sự độc lập của phân loại Naive Bayes chính là thể hiện của mô hình mạng tin cậy (belief network) trong trường hợp đặc biệt, và phân loại là chỉ dựa trên một nút cha duy nhất của mỗi một đặc trưng đầu vào. Mạng tin cậy này đề cập tới xác suất phân tán P(Y) đối với mỗi một đặc trưng đích Y và $P(X_i|Y)$ đối với mỗi một đặc trưng đầu vào X_i. Với mỗi một

đối tượng, dự đoán bằng cách tính toán dựa trên các xác suất điều kiện của các đặc trưng quan sát được cho mỗi đặc trưng đầu vào.

Định lý Bayes: Giả sử A và B là hai sự kiện đã xảy ra. Xác suất có điều kiện A khi biết trước điều kiện B được cho bởi:

P(A|B) = P(B|A).P(A)/P(B)

- P(A): Xác suất của sự kiện A xảy ra.

- P(B): Xác suất của sự kiện B xảy ra.

- P(B|A): Xác suất (có điều kiện) của sự kiện B xảy ra, nếu biết rằng sự kiện A đã xảy ra.

- P(A|B): Xác suất (có điều kiện) của sự kiện A xảy ra, nếu biết rằng sự kiện B đã xảy ra

3.3.2 Mô hình xác xuất

Một cách trừu tượng, mô hình xác suất cho phân loại là một mô hình điều kiện $\rho(C|F_1$,.., $F_n)$

Trên một lớp biến C với số lượng nhỏ các đầu ra hoặc các lớp. Điều kiện trên một vài biến đặc trưng F_1 đến F_2. Vấn đề chính trong bài toán này là nếu số đặc trưng n là lớp hoặc một đặc trưng có thể có số lượng lớn các giá trị, thì một mô hình được tạo ra dựa trên các bảng xác suất là phù hợp trong điều kiện này. Lý thuyết Bayes có thể viết thành:

$$\rho(C|F_1, \dots, F_n) = \frac{\rho(C)\rho(|F_1, \dots, F_n|C)}{\rho(F_1, \dots, F_n|C)}$$

Một cách mô tả đơn giản cho công thức trên như sau:

$$\text{Hậu nghiệm} = \frac{nghiệm\ trước \times khả\ năng}{bằng\ chứng}$$

Trên thực tế, chỉ cần quan tâm tới số các phân mảnh (fraction), bởi có một số đặc trưng không phụ thuộc vào C và các giá trị F_i đã cho, mô hình $\rho(C|F_1,.., F_n)$ có thể được viết lại như sau, sử dụng luật xích để lặp lại định nghĩa của xác suất điều kiện:

$\rho(C,F1,..., Fn) = \rho(C)\ \rho(F1,..., Fn|C)$

$= \rho(C)\ \rho(F1|C)\ \rho(F2,..., Fn|C, F1)$

$= \rho(C)\ \rho(F1|C)\ \rho(F2|C, F1)\ \rho(F3,..., Fn|C, F1,F2)$

$= \rho(C)\ \rho(F1|C)\ \rho(F2|C, F1) \dots \rho(Fn|C, F1,F2, F3, \dots Fn\text{-}1)$

Giả thiết của xác suất điều kiện: giả thiết rằng mỗi đặc trưng F_i là độc lập có điều kiện với các đặc trưng khắc F_j với $j \neq i$, trong lớp đã cho C. Điều đó có nghĩa rằng:

$\rho(F_i|C, F_j) = \rho(F_i|C)$,

$\rho(F_i|C, F_j, F_k) = \rho(F_i|C)$,

$\rho(F_i|C, F_j, F_k, F_l) = \rho(F_i|C)$,

Với mọi trường hợp i≠ j, k, l. Từ đó, mô hình kết hợp được biểu diễn bởi
$\rho(C| F_1,..., F_n) \alpha \rho(C, F_1,..., F_n)$

$\alpha\rho(C) \rho(F_1|C) \rho(F_2|C) \rho(F_3|C)$

$\alpha\rho(C) \prod_{i=1}^{n} \rho(F_2|C)$

Có nghĩa rằng dưới giả thiết độc lập trên, phân tán có điều kiện trên các lớp biến C là:

$\rho(C| F_{1...} F_n)=\rho(C) \prod_{i=1}^{n} \rho(F_i|C)$

Với $Z = \rho(F_1,..., F_n)$ được gọi là nhân tố độc lập trên $F_1,..., F_n$ và là một hằng nếu các giá trị của các biến đặc trưng là đã biết.

Xây dựng phân lớp từ mô hình xác suất

Phân lớp Bayes kết hợp với luật quyết định tạo ra phân loại Naïve Bayes. Một luật thông thường đưa ra giả thuyết về khả năng nhất hay còn được xem như là cực đại hóa xác suất hậu nghiệm (maximum a posteriori). Bộ phân loại Bayes là một hàm phân loại được định nghĩa:

$$classify(f_1, \dots, f_n) = argmaxp(C = c) \coprod_{i=1}^{n} p(F_i = f_i|C = c)$$

3.3.3 Bộ phân loại Naïve Bayes

Naive Bayes là phương pháp phân loại dựa vào xác suất được sử dụng rộng rãi trong lĩnh vực máy học và nhiều lĩnh vực khác như trong các công cụ tìm kiếm, các bộ lọc mail.

Mục đích chính là làm sao tính được xác suất $Pr(C_j, d')$, xác suất đểtài liệu d'nằm trong lớp C_j. Theo luật Bayes, tài liệu d' sẽ được gán vào lớp C_j nào có xác suất $Pr(C_j, d')$ cao nhất.

Công thức để tính $Pr(C_j, d')$ như sau:

$$H_{BAYES(d')} = argmax\left[\frac{\Pr(C_j) \times \coprod_{i=1}^{|d'|} \Pr(w_i|C_j)}{\sum \Pr(c') \times \coprod_{i=1}^{|d'|} \Pr(w_i|C')}\right]$$

- TF(w_i, d') là số lần xuất hiện của từ w_i trong tài liệu d'

- |d'| là số lượng các từ trong tài liệu d'

- w_i là một từ trong không gian đặc trưng F với số chiều là |F|

- Pr(C_j) được tính dựa trên tỷ lệ phần trăm của số tài liệu mỗi lớp tương ứng

$$\Pr(C_j) = \frac{||C_J||}{C} = \frac{||C_J||}{\sum_{C' \in C} ||C'||}$$

trong tập dữ liệu huấn luyện.

$$\Pr(w_i|C_j) = \frac{1 + TF(w_i, c_j)}{|F| + \sum_{W' \in |F|} TF(w', c_j)} = \frac{||C_J||}{\sum_{C' \in C} ||C'||}$$

Ngoài ra còn có các phương pháp NB khác có thể kể ra như ML Naive Bayes, MAP Naive Bayes, Expected Naive Bayes. Nói chung, Naïve Bayes là một công cụ rất hiệu quả trong một số trường hợp.

Thuật toán Naive Bayes dựa trên nguyên lý Bayes được phát biểu như sau:

$$P(Y|X) = \frac{P(XY)}{P(X)} = \frac{P(X|Y)P(Y)}{P(X)}$$

Áp dụng trong bài toán phân loại, các dữ kiện gồm có:

- D: tập dữ liệu huấn luyện đã được vector dạng $\vec{x} = (x_1, x_2, \dots, x_n)$

- C_i: phân lớp im với i = {1, 2,.., m}

- Các thuộc tính độc lập điều kiện đôi một với nhau.

Theo định lý Bayes:

$$P(X|C_i) = \frac{P(X|C_i)P(C_i)}{P(X)}$$

Theo tính chất độc lập điều kiện:

$$P(X|C_i) = \prod_{k=1}^{n} P(x_k|C_i)$$

Trong đó:

- $P(C_i|X)$: là xác suất thuộc phân lớp i khi biết trước mẫu X

- $P(C_i)$: Xác suất phân lớp i

- $P(x_k|C_i)$: Xác suất thuộc tính thứ k mang giá trị xk khi biết X thuộc phân lớp i.

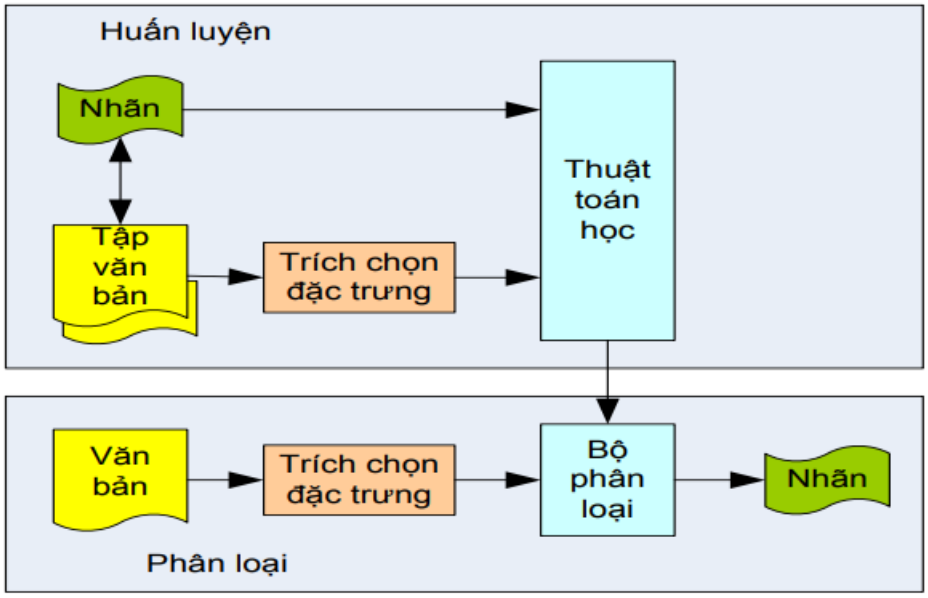
Các bước thực hiện thuật toán Naive Bayes

Bước 1: Huấn luyện Naive Bayes (dựa vào tập dữ liệu), tính $P(C_i)$ và $P(x_k|C_i)$

Bước 2: Phân lớp X^{new}($x_1,x_2,...,x_n$), ta cần tính xác suất thuộc từng phân lớp khi đã biết trước X^{new}.X^{new}được gán vào lớp có xác suất lớn nhất theo công thức

$$(P_{C_i \in c}^{max}(C_i) \prod_{k=1}^{n} P(x_k|C_i))$$

Mô hình tổng quát việc phân loại:



Hình 3.4 - Mô tả bước xây dựng bộ phân lớp

3.4 Sơ lược các mô hình phân lớp và gắn nhãn hiện đại

3.4.1 Maximum entropy

Mô hình Entropy cực đại là mô hình dựa trên xác suất có điều kiện cho phép tích hợp các thuộc tính đa dạng từ dữ liệu mẫu nhằm hỗ trợ quá trình phân lớp.

Tư tưởng chủ đạo của nguyên lý Entropy cực đại rất đơn giản: ta phải xác định môt phân phối mô hình sao cho phân phối đó tuân theo mọi giả thiết đã quan sát từ thực nghiệm, ngoài ra không cho thêm bất kì giả thiết nào khác. Điều này có nghĩa là phân phối mô hình phải thoả mãn các ràng buộc quan sát từ thực nghiệm và phải gần nhất với phân phối đều.

Entropy là độ đo về tính đồng đều hay tính không chắc chắn của một phân phối xác suất. Một phân phối xác suất có Entropy càng cao thì phân phối của nó càng đều. Độ đo Entropy điều kiện của một phân phối xác suất trên một chuỗi các trạng thái với điều kiện biết từ một chuỗi dữ liệu quan sát được tính như sau:

$$H(p) = -\sum_{x,y} \tilde{p}(x)p(y/x)logp(y/x)$$

a) Xây dựng mô hình:

Xem xét bài toán phân lớp, với Y là tập các lớp, X là tập các thông tin ngữ cảnh, là những thông tin quan trọng cần cho việc phân lớp văn bản vào lớp Y một cách chính xác.

Nhiệm vụ trong bài toán phân lớp là xây dựng một mô hình thống kê mà dự đoán chính xác lớp của văn bản bất kì. Mô hình như vậy chính là phương pháp ước lượng xác suất có điều kiện p(y|x).

Mô hình Entropy cực đại cung cấp một phương pháp đơn giản để ước lượng xác suất có điều kiện p(y|x) thông qua việc thống kê các thuộc tính quan trọng quan sát được từ tập dữ liệu huấn luyện.

b) Tập dữ liệu huấn luyện:

Để làm bài toán phân lớp trước tiên phải xây dựng tập dữ liệu huấn luyện T = {(x_1, y_1), …, (x_N, y_N)} trong đó {x_1, …, x_N} là tập các thông tin ngữ cảnh đã được gán nhãn tương ứng là tập các lớp {y_1, …, y_N}.

Với một cặp (x_i, y_i), phân phối xác suất thực nghiệm của nó được tính bởi:

$\tilde{p} = (x_i, y_i) = \frac{1}{N} \times$ số lần xuất hiện của (x_i, y_i) trong tập dữ liệu mẫu. Thông thường thì mỗi cặp (x_i, y_i) không thể không xuất hiện trong tập mẫu mà nó sẽ xuất hiện ít nhất một lần.

c) Những thống kê, đặc trưng và ràng buộc:

Mục đích cả chúng ta là xây dựng một mô hình thống kê của bài toán mà nó phát sinh xác suất $\tilde{p}(x, y)$ mẫu huấn luyện. Khối kiến trúc của mô hình này sẽ là một tập các thống kê của mẫu huấn luyện. Ví dụ khi xét bài toán phân loại bộ phim. Bộ phim được xếp vào một trong ba loại: good, not good, normal. Quan sát từ tập dữ liệu mẫu là 74 câu nhận xét đã được gán nhãn, ta có nhận xét như sau: nếu nhận xét có từ "failure" thì xác suất nhận xét đó thuộc loại "not good" là 80%. Đây chính là một thống kê.

Để biểu diễn sự kiện đó chúng ta có thể sử dụng hàm để biểu diễn như sau:

$$f(x, y) = \begin{cases} 1 & if\ y = "failure" \\ 0 \end{cases}$$

Giá trị kỳ vọng của f liên quan tới phân phối thực nghiệm $\tilde{p}(x, y)$ chính là thống kê mà chúng ta đã nhắc tới. Chúng ta biểu diễn giá trị kỳ vọng này bởi:

$\tilde{E}(f) = \sum \tilde{p}(x, y).f(x, y)$ với mọi cặp (x, y) (1)

Chúng ta có thể biểu diễn bất kỳ thống kê nào của mẫu huấn luyện như giá trị kỳ vọng của hàm nhị phân(f) thích hợp. Chúng ta gọi hàm đó là hàm đặc trưng hay đặc trưng (Như vậy với các phân phối xác suất, chúng ta sẽ dùng ký hiệu và sử dụng hàm f(x, y) để biểu diễn giá trị của f với mỗi cặp (x, y) riêng biệt cũng như toàn bộ hàm f)

Cần phân biệt rõ ràng 2 khái niệm về đặc trưng và ràng buộc: một đặc trưng là một hàm nhận giá trị nhị phân của cặp (x, y); một ràng buộc là một phương trình giữa giá trị kỳ

vọng của hàm đặc trưng trong mô hình và giá trị kỳ vọng của nó trong dữ liệu huấn luyện.

d) Nguyên lý Maximum Entropy:

Giả thiết rằng chúng ta có n hàm đặc trưng f_i, nó quyết định những thống kê mà chúng ta cảm thấy là quan trọng trong quá trình mô hình hóa. Chúng ta muốn mô hình của chúng ta phù hợp với những thống kê đó. Vì vậy, chúng ta sẽ muốn p hợp lệ trong tập con **C** của **P** được định nghĩa bởi:

$$C = \left\{ p \in P \middle| E(f_i) = \tilde{E}(f_i) \qquad for\ i \in \{1,2,\dots,n\} \right\} (4)$$

Trong số các mô hình p € C, triết lý cực đại Entropy yêu cầu rằng chúng ta lựa chọn phân phối mà ngang bằng nhau nhất.

Trong phạm vi toán học ngang bằng nhau của phân phối có điều kiện p(y|x) được cung cấp bởi Entropy có điều kiện:

$$H(p) = -\sum_{x,y} \tilde{p}(x).p(y|x).\log\big(p(y|x)\big) \ (5)$$

Entropy là bị chặn dưới bởi 0, Entropy của mô hình không có sự không chắc chắn nào, và chặn trên bởi log|**Y**|, Entropy của phân phối ngang bằng nhau trên toàn bộ các giá trị có thể |Y| của y. Với định nghĩa này, chúng ta đã sẵn sàng để biểu diễn nguyên lý của cực đại Entropy:

Để lựa chọn mô hình từ một tập C các phân phối xác suất được chấp nhận, lựa chọn mô hình $p^* \in$ C với cực đại Entropy H(p):

$$p^* = argmaxH(p) \qquad với\ p \in C \qquad (6)$$

Điều đó thể hiện rằng p^* luôn luôn xác định; vì vậy, luôn luôn tồn tại một mô hình duy nhất p^* với cực đại Entropy trong bất kỳ tập ràng buộc C nào.

3.4.2 SVM

Máy vectơ hỗ trợ (SVM - viết tắt tên tiếng Anh support vector machine) là một khái niệm trong thống kê và khoa học máy tính cho một tập hợp các phương pháp học có giám sát liên quan đến nhau để phân loại và phân tích hồi quy. SVM dạng chuẩn nhận dữ liệu vào và phân loại chúng vào hai lớp khác nhau. Do đó SVM là một thuật toán phân loại nhị phân. Với một bộ các ví dụ luyện tập thuộc hai thể loại cho trước, thuật toán luyện tập SVM xây dựng một mô hình SVM để phân loại các ví dụ khác vào hai thể loại đó. Một mô hình SVM là một cách biểu diễn các điểm trong không gian và lựa chọn ranh giới giữa hai thể loại sao cho khoảng cách từ các ví dụ luyện tập tới ranh giới là xa nhất có thể.

Phân loại thống kê là một nhiệm vụ phổ biến trong học máy. Trong mô hình học có giám sát, thuật toán được cho trước một số điểm dữ liệu cùng với nhãn của chúng thuộc một trong hai lớp cho trước. Mục tiêu của thuật toán là xác định xem một điểm dữ liệu mới sẽ được thuộc về lớp nào. Mỗi điểm dữ liệu được biểu diễn dưới dạng một vector p-chiều, và ta muốn biết liệu có thể chia tách hai lớp dữ liệu bằng một siêu phẳng $p - 1$ chiều. Đây gọi là phân loại tuyến tính. Có nhiều siêu phẳng có thể phân loại được dữ liệu. Một lựa chọn hợp lý trong chúng là siêu phẳng có lề lớn nhất giữa hai lớp.

a) Lịch sử:

Thuật toán Support Vector Machines (SVM) ban đầu tìm ra bởi Vladimir N.Vapnik và dạng chuẩn hiện nay sử dụng lề mềm được tìm ra bởi Vapnik và Corinna Cortes năm 1995.

b) Định nghĩa

Là phương pháp dựa trên nền tảng của lý thuyết thống kê nên có một nền tảng toán học chặt chẽ để đảm bảo rằng kết quả tìm được là chính xác.

Là thuật toán học giám sát (supervied learning) được sử dụng cho phân lớp dữ liệu.

Là một phương pháp thử nghiệm, đưa ra 1 trong những phương pháp mạnh và chính xác nhất trong số các thuật toán nổi tiếng về phân lớp dữ liệu.

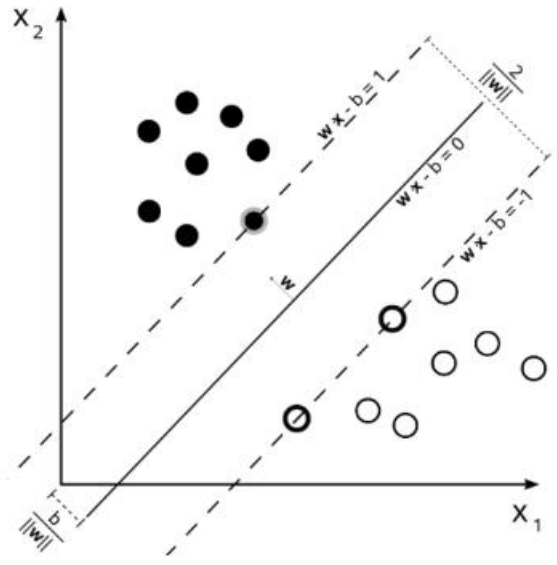
SVM là một phương pháp có tính tổng quát cao nên có thể được áp dụng cho nhiều loại bài toán nhận dạng và phân loại

c) Ý tưởng

Cho trước một tập huấn luyện, được biểu diễn trong không gian vector, trong đó mỗi tài liệu là một điểm, phương pháp này tìm ra một siêu phẳng f quyết định tốt nhất có thể chia các điểm trên không gian này thành hai lớp riêng biệt tương ứng là lớp "+" và lớp "-". Chất lượng của siêu phẳng này được quyết định bởi khoảng cách (gọi là biên) của điểm dữ liệu gần nhất của mỗi lớp đến mặt phẳng này. Khi đó, khoảng cách biên càng lớn thì mặt phẳng quyết định càng tốt, đồng thời việc phân loại càng chính xác.

Ý tưởng của nó là ánh xạ (tuyến tính hoặc phi tuyến) dữ liệu vào không gian các vector đặc trưng (space of feature vectors) mà ở đó một siêu phẳng tối ưu được tìm ra để tách dữ liệu thuộc hai lớp khác nhau.

Mục đích của phương pháp SVM là tìm được khoảng cách biên lớn nhất.



Hình 3.5 – Ý tưởng thuật toán Support Vector Machines

Đường tô đậm là siêu phẳng tốt nhất và các điểm được bao bởi hình chữ nhật là những điểm gần siêu phẳng nhất, chúng được gọi là các vector hỗ trợ (support vector). Các đường nét đứt mà các support vector nằm trên đó được gọi là lề (margin).

d) Cơ sở lý thuyết

SVM thực chất là một bài toán tối ưu, mục tiêu của thuật toán này là tìm được một không gian F và siêu phẳng quyết định f trên F sao cho sai số phân loại là thấp nhất.

Cho tập mẫu D = {(x_1, y_1), (x_2, y_2), …, (x_l, y_l)} với xi ∈ R_n, thuộc vào hai lớp nhãn yi ∈ {-1,1} là tập nhãn lớp tương ứng của các xi (-1 biểu thị lớp I, 1 biểu thị lớp II).

Ta có, phương trình siêu phẳng chứa vector $\vec{x}_i$ trong không gian:

$$\vec{x}_i.\vec{w}_i + b = 0$$

Đặt $f(\vec{x}_i) = sign(\vec{x}_i.\vec{w}_i + b) = \begin{cases} +1, \vec{x}_i.\vec{w}_i + b > 0 \\ -1, \vec{x}_i.\vec{w}_i + b < 0 \end{cases}$

Như vậy, f($\vec{x}_i$) biểu diễn sự phân lớp của $\vec{x}_i$ vào hai lớp như nêu trên.

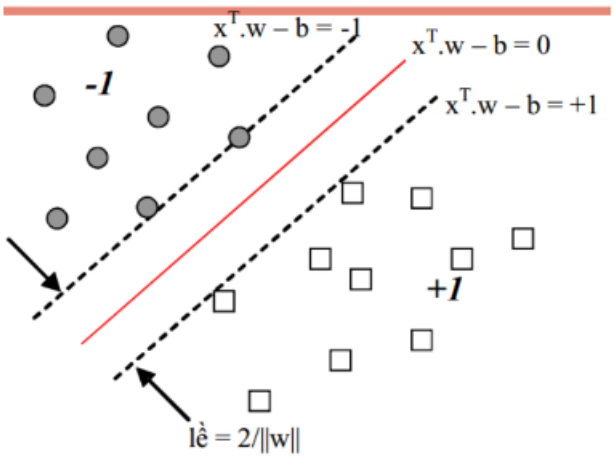
Ta nói y_i = +1 nếu thuộc lớp I và y_i = -1 nếu $\vec{x}_i$ thuộc lớp II.

e) Bài toán phân lớp

*** Bài toán phân 2 lớp**

Bài toán đặt ra là: Xác định hàm phân lớp lớp để phân lớp các mẫu trong tương lai, nghĩa là với một mẫu dữ liệu mới xi thì cần phải xác định x_i được phân lớp +1 hay lớp -1.

Trường hợp 1: Tập D có thể phân chia tuyến tính được mà không có nhiễu (tất cả các điểm được gán nhãn +1 thuộc về phía dương của siêu phẳng, tất cả các điểm được gán nhãn -1 thuộc về phía âm của siêu phẳng)



Hình 3.6 – Trường hợp 1 (bài toán phân 2 lớp)

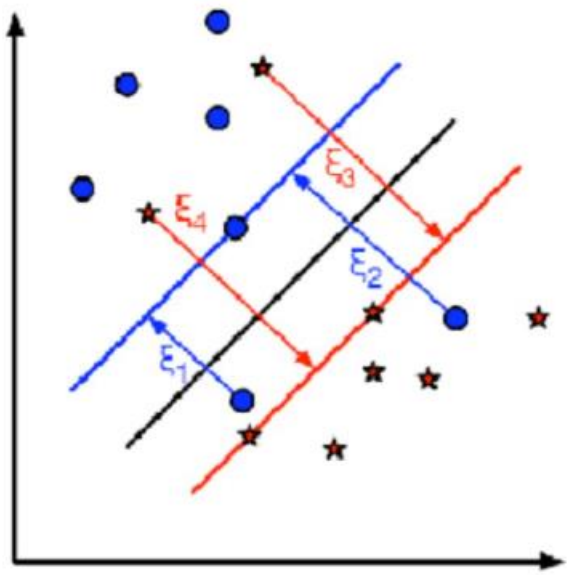
Ta sẽ tìm siêu phẳng tách với w ∈ R^n là vector trọng số, b ∈ R^n là hệ số tự do, sao cho:

Đặt $f(\vec{x}_i) = sign(\vec{x}_i.\vec{w} + b) = \begin{cases} +1, \vec{x}_i.\vec{w} + b > 0 \\ -1, \vec{x}_i.\vec{w} + b < 0 \end{cases} \forall (x_i, y_i) \in D$

Lúc này ta cần giải toán tối ưu:

$$\begin{cases} Min(L(w)) = \frac{1}{2}||w||^2 \\ y_i(x_i.w^T + b) \geq 1, i = 1, \dots, l \end{cases}$$

Trường hợp 2: Tập dữ liệu D có thể phân chia tuyến tính được nhưng có nhiễu. Trong trường hợp này, hầu hết các điểm đều được phân chia đúng bởi siêu phẳng. Tuy nhiên có 1 số điểm bị nhiễu, nghĩa là: điểm có nhãn dương nhưng lại thuộc phía âm của siêu phẳng, điểm có nhãn âm nhưng lại thuộc phía dương của siêu phẳng.



Hình 3.7 – Trường hợp 2 (bài toán phân 2 lớp)

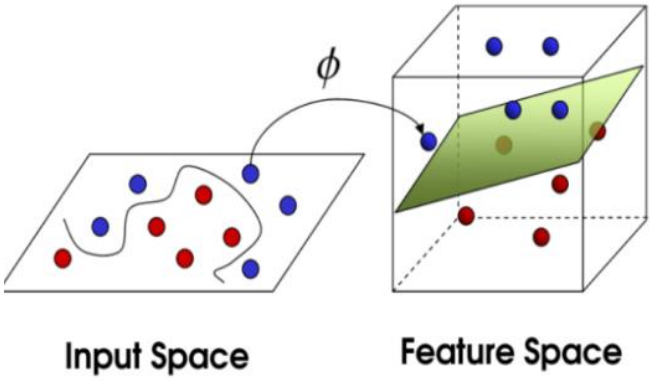
Trong trường hợp này, ta sử dụng 1 biến mềm $\varepsilon_i \geq 0$ sao cho: $\vec{y}_i.(\vec{x}_i.\vec{w}+b) \geq 1-\varepsilon_i$, $i = 1,\dots,l$

Bài toán tối ưu trở thành:

$$\begin{cases} Min\big(L(w,\varepsilon)\big) = \dfrac{1}{2}||w||^2 + C\sum_{i=1}^{l}\varepsilon_i \\ y_i(x_i.w^T+b) \geq 1-\ \varepsilon_i, i = 1,\dots,l;\ \varepsilon_i \geq 0 \end{cases}$$

Trong đó C là tham số xác định trước, định nghĩa giá trị ràng buộc, C càng lớn thì mức độ phạm vi đối với những lỗi thực nghiệm (là lỗi xảy ra lúc huấn luyện, tính bằng thương số của số phần tử lỗi và tổng số phần tử huấn luyện) càng cao.

Trường hợp 3: Ta dữ liệu D không thể phân chia tuyến tính được, ta sẽ ánh xạ các vector dữ liệu x từ không gian n chiều vào một không gian m chiều (m > n), sao cho trong không gian m chiều, D có thể phân chia tuyến tính được.



Hình 3.8 – Trường hợp 3 (bài toán phân 2 lớp)

Gọi $\emptyset$ là ánh xạ phi tuyến từ không gian R^n vào không giam R^m

$$\emptyset: R^n \rightarrow R^m$$

Bài toán tối ưu trở thành:

$$\begin{cases} Min\big(L(w,\varepsilon)\big) = \frac{1}{2}||w||^2 + C\sum_{i=1}^{l}\varepsilon_i \\ y_i(\emptyset(x_i).w^T + b) \geq 1 - \varepsilon_i, i = 1,\dots,l;\ \varepsilon_i \geq 0 \end{cases}$$

*** Bài toán phân đa lớp**

Để phân đa lớp thì kỹ thuật SVM sẽ chia không gian dữ liệu thành 2 phần và tiếp tục với không gian đã được phân chia. Khi đó hàm quyết định phân dữ liệu vào lớp thứ I sẽ là:

$$f_i(x) = w_i^T(x) + b_i$$

Những phần tử x là support vector nếu thỏa điều kiện:

$$\mathrm{f}_i(\mathrm{x}) = \begin{cases} 1, \in i \\ -1 \notin i \end{cases}$$

Giả sử bài toán phân loại k lớp ($k \geq 2$), ta sẽ tiến hành $k(k-1)/2$ lần phân lớp nhị phân sử dụng phương pháp SVM. Mỗi lớp sẽ tiến hành phân tách với k-1 lớp còn lại để xác định $k-1$ hàm phân tách (chiến lược "một-đối-một" (one-against-one).

Kỹ thuật phân đa lớp bằng phương pháp hiện vẫn đang được tiếp tục nghiên cứu và phát triển.

f) Các bước chính của phân lớp SVM

Step 1: Tiền xử lý dữ liệu: Phương pháp SVM yêu cầu được diễn tả như các vector của các số thực. Như vậy nếu đầu vào chưa phải là số thực thì ta cần tìm cách chuyển chúng về dạng số SVM. Tránh các số quá lớn, thường nên co giãn dữ liệu để chuyển đoạn [-1,1] hoặc [0,1].

Step 2: Chọn hàm hạt nhân: cần chọn hàm hạt nhân phù hợp tương ứng cho từng bài toán cụ thể để đạt được độ chính xác cao trong quá trình học tập.

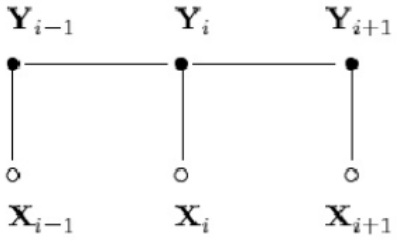
Step 3: Thực hiện việc kiểm tra chéo để xác định các tham số cho ứng dụng.

Step 4: Sử dụng các tham số cho việc huấn luyện tập mẫu.

Step 5: Kiểm thử tập dữ liệu Test.

3.4.3 Conditional random fields

Conditional random fields – CRF là mô hình chuỗi các xác suất có điều kiện, huấnluyện để tối đa hóa xác suất điều kiện. Nó là một framework cho phép xây dựng nhữngmô hình xác suất để phân đoạn và gán nhãn chuỗi dữ liệu. Cũng giống như trường ngẫu nhiên Markov (Markov random field), CRF là một mô hình đồ thị vô hướng mà mỗi đỉnh biểu diễn cho một biến ngẫu nhiên (random variable) mà có phân phối (distribution) được suy ra, và mỗi cung (edge) biểu diễn mối quan hệ phụ thuộc giữa hai biến ngẫu nhiên



Hình 3.9 – Cấu trúc chuỗi (chain-structured) của đồ thị CRFs

X là một biến ngẫu nhiên trên chuỗi dữ liệu cần được gán nhãn và Y là biến ngẫu nhiên trên chuỗi nhãn (hoặc trạng thái) tương ứng. Ví dụ X là chuỗi các từ quan sát (observation) thông qua các câu bằng ngôn ngữ tự nhiên, Y là chuỗi các nhãn từ loại được gán cho những câu trong tập X (các nhãn này được quy định sẵn trong tập các nhãn từ loại). Một linear-chain (chuỗi tuyến tính) CRF với các tham số được cho bởi công thức [2]:

$$P_\lambda = \frac{1}{Z_x} \exp\left(\sum_{t=1}^{T} \sum_{k} \lambda_k f_k(y_{t-1}, y_t, x, t)\right.$$

Với Z_x là một thừa số chuẩn hóa nhằm để đảm bảo tổng các xác suất của chuỗi trạng thái bằng 1

$$Z(x) = \sum_{y} exp\left\{\sum_{k=1}^{K} \lambda_k f_k(y_t, y_t - 1, x_t)\right\}$$

$f_k(y_t - 1, y_t, x, t)$ là một hàm đặc trưng (feature function), thường có giá trị nhị phân (binary-valued), nhưng cũng có thể là giá trị thực (real-valued). Và là một trọng số học (learned weight) kết hợp với đặc trưng f_k. Những hàm đặc trưng có thể đo bất kỳ trạng thái chuyển dịch (state transition) nào, y_t-1 → y_t, và chuỗi quan sát x, tập trung vào thời điểm hiện tại t. Ví dụ, một hàm đặc trưng có thể có giá trị 1 khi y_t-1 là trạng thái TITLE, y_t là trạng thái AUTHOR và x_t là một từ xuất hiện trong tập từ vựng chứa tên người.

Người ta thường huấn luyện CRFs bằng cách làm cực đại hóa hàm likelihood theo dữ liệu huấn luyện sử dụng các kĩ thuật tối ưu như L-BFGS. Việc lập luận (dựa trên mô hình đã học) là tìm ra chuỗi nhãn tương ứng của một chuỗi quan sát đầu vào. Đối với CRFs, người ta thường sử dụng thuật toán qui hoạch động điển hình là Viterbi (là thuật toán lập trình động nhằm tìm ra chuỗi khả năng (most likely) của các trạng thái ẩn) để thực hiện lập luận với dữ liệu mới

CHƯƠNG 4: CÁC PHƯƠNG PHÁP PHÂN CỤM

4.1 K – means

4.1.1 Khái niệm

K-means là một thuật toán phân cụm đơn giản thuộc loại học không giám sát (tức là dữ liệu không có nhãn) và được sử dụng để giải quyết bài toán phân cụm. Ý tưởng của thuật toán phân cụm k-means là phân chia 1 bộ dữ liệu thành các cụm khác nhau. Trong đó số lượng cụm được cho trước là k. Công việc phân cụm được xác lập dựa trên nguyên lý: Các điểm dữ liệu trong cùng 1 cụm thì phải có cùng 1 số tính chất nhất định. Tức là giữa các điểm trong cùng 1 cụm phải có sự liên quan lẫn nhau. Đối với máy tính thì các điểm trong 1 cụm đó sẽ là các điểm dữ liệu gần nhau.

Thuật toán phân cụm K-means là một phương pháp được sử dụng trong phân tích tính chất cụm của dữ liệu. Nó đặc biệt được sử dụng nhiều trong khai phá dữ liệu và thống kê. Nó phân vùng dữ liệu thành k cụm khác nhau. Giải thuật này giúp chúng ta xác định được dữ liệu của chúng ta nó thực sử thuộc về nhóm nào.

Hình 4.1 – Mô tả thuật toán phân cụm K-means

4.1.2 Các bước của thuật toán K-means

Bước 1: Chọn ngẫu nhiên K tâm (centroid) cho K cụm (cluster). Mỗi cụm được đại diện bằng các tâm của cụm.

Bước 2: Tính khoảng cách giữa các đối tượng (objects) đến K tâm (thường dùng khoảng cách Euclidean)

Bước 3: Nhóm các đối tượng vào nhóm gần nhất

Bước 4: Xác định lại tâm mới cho các nhóm

Bước 5: Thực hiện lại bước 2 cho đến khi không có sự thay đổi nhóm nào của các đối tượng

4.1.3 Một số lưu ý

a) Lựa chọn số cụm:

Chỉ việc lựa chọn số cụm k đã có thể tách thành 1 bài toán riêng. Không có 1 con số k nào là hợp lý cho tất cả các bài toán. Bạn có thể đọc hiểu tập dữ liệu của mình để xác định xem trong đó có thể có bao nhiêu cụm? Nhưng không phải lúc nào bạn cũng có thể làm thế. Cách làm duy nhất là bạn hãy thử với từng giá trị k=1,2,3,4,5,... để xem kết quả phân cụm thay đổi như thế nào. Một số nghiên cứu cho thấy việc thay đổi k sẽ có hiệu quả nhưng sẽ dừng lại ở 1 con số nào đó. Như vậy bạn hoàn toàn có thể thử xem dữ liệu của mình tốt với giá trị k nào đó.

b) Khởi tạo K vị trí ban đầu:

Bằng cách nào đó, hãy có gắng khởi tạo k tâm cụm này phân bố đồng đều trên không gian của bộ dữ liệu. Điều đó có thể làm khi bạn có thể xác định được không gian và tính chất của dữ liệu. Nhưng ít nhất, các tâm cụm mà bạn khởi tạo cũng đừng quá gần nhau, cũng đừng trùng nhau.

c) Về vấn đề tính dừng (hội tụ)

Đối với những trường hợp dữ liệu phức tạp, thuật toán k-means sẽ rất lâu hoặc không bao giờ hội tụ. Tức là sẽ không bao giờ xác định được tâm cụm cố định để kết thúc bài toán. Hoặc là phải chạy qua rất nhiều bước lặp. Trong những trường hợp như vậy, thay vì phải tìm được k tâm cụm cố định thì ta sẽ dừng bài toán khi sự thay đổi ở một con số chấp nhận được. Tức là giữa hai lần cập nhật tâm cụm thì chênh lệch vị trí giữa tâm cũ và mới nhỏ hơn một số delta cho phép nào đó.

4.1.4 Ví dụ:

Giả sử ta có 4 loại thuốc A, B, C, D, mỗi loại thuộc được biểu diễn bởi 2 đặc trưng X và Y như sau. Mục đích của ta là nhóm các thuốc đã cho vào 2 nhóm (K=2) dựa vào các đặc trưng của chúng

Object	Feature 1 (X): weight index	Feature 2 (Y): pH
Medicine A	1	1
Medicine B	2	1
Medicine C	3	3
Medicine D	4	4

Bảng 4.1 Thông tin 4 loại thuốc ABCD

Bước 1. Khởi tạo tâm (centroid) cho 2 nhóm. Giả sử ta chọn A là tâm của nhóm thứ nhất (tọa độ tâm nhóm thứ nhất c1(1,1)) và B là tâm của nhóm thứ 2 (tạo độ tâm nhóm thứ hai c2 (2,1))

Bước 2. Tính khoảng cách từ các đối tượng đến tâm của các nhóm (Khoảng cách Euclidean)

$$D^0 = \begin{bmatrix} 0 & 1 & 3.61 & 5 \\ 1 & 0 & 2.83 & 4.24 \end{bmatrix} \begin{matrix} c_1 = (1,1)\ group - 1 \\ c_2 = (2,1)\ group - 2 \end{matrix}$$

$$\begin{matrix} A & B & C & D \end{matrix}$$

$$\begin{bmatrix} 0 & 1 & 3.61 & 5 \\ 1 & 0 & 2.83 & 4.24 \end{bmatrix} \begin{matrix} X \\ Y \end{matrix}$$

Mỗi cột trong ma trận khoảng cách (D) là một đối tượng (cột thứ nhất tương ứng với đối tượng A, cột thứ 2 tương ứng với đối tượng B,…). Hàng thứ nhất trong ma trận khoảng cách biểu diễn khoảng cách giữa các đối tượng đến tâm của nhóm thứ nhất (c1) và hàng thứ 2 trong ma trận khoảng cách biểu diễn khoảng cách của các đối tượng đến tâm của nhóm thứ 2 (c2)

Ví dụ, khoảng cách từ loại thuốc C=(4,3) đến tâm c1(1,1) là 3.61 và đến tâm c2(2,1) là 2.83 được tính như sau:

$$c_1 = (1,1)\sqrt{(4-1)^2 + (3-1)^2} = 3.61$$

$$c_2 = (2,1)\sqrt{(4-2)^2 + (3-1)^2} = 2.83$$

Bước 3. Nhóm các đối tượng vào nhóm gần nhất

$$G^0 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 \end{bmatrix} \begin{matrix} group-1 \\ group-2 \end{matrix}$$

$$\begin{matrix} A & B & C & D \end{matrix}$$

Ta thấy rằng nhóm 1 sau vòng lặp thứ nhất gồm có 1 đối tượng A và nhóm 2 gồm các đối tượng còn lại B, C, D

Bước 4. Tính lại tọa độ các tâm cho các nhóm mới dựa vào tọa độ của các đối tượng trong nhóm. Nhóm 1 chỉ có 1 đối tượng A nên tâm nhóm 1 vẫn không đổi, c1(1,1). Tâm nhóm 2 được tính như sau:

$$c_2 = \left(\frac{2+4+5}{3}, \frac{1+3+4}{3}\right) = \left(\frac{11}{3}, \frac{8}{3}\right)$$

Bước 5. Tính lại khoảng cách từ các đối tượng đến tâm mới

$$D^1 = \begin{bmatrix} 0 & 1 & 3.61 & 5 \\ 3.14 & 2.36 & 0.47 & 1.89 \end{bmatrix} \quad \begin{matrix} c_1 = (1{,}1)\ group-1 \\ c_2 = \left(\frac{11}{3}, \frac{8}{3}\right) group-2 \end{matrix}$$

$$\begin{matrix} A & B & C & D \end{matrix}$$

$$\begin{bmatrix} 1 & 2 & 4 & 5 \\ 1 & 1 & 3 & 4. \end{bmatrix} \begin{matrix} X \\ Y \end{matrix}$$

Bước 6. Nhóm các đối tượng vào nhóm

$$G^1 = \begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \end{bmatrix} \begin{matrix} group-1 \\ group-2 \end{matrix}$$

$$\begin{matrix} A & B & C & D \end{matrix}$$

Bước 7. Tính lại tâm cho nhóm mới

$$c_1 = \left(\frac{1+2}{2}, \frac{1+1}{2}\right) = \left(1\frac{1}{2}, 1\right)$$

$$c_2 = \left(\frac{4+5}{2}, \frac{3+4}{2}\right) = \left(4\frac{1}{2}, 3\frac{1}{2}\right)$$

Bước 8. Tính lại khoảng cách từ các đối tượng đến tâm mới

$$D^2 = \begin{bmatrix} 0.5 & 0.5 & 3.20 & 4.61 \\ 4.30 & 3.54 & 0.71 & 0.71 \end{bmatrix} \quad \begin{matrix} c_1 = \left(1\frac{1}{2}, 1\right)\ group-1 \\ c_2 = \left(4\frac{1}{2}, 3\frac{1}{2}\right) group-2 \end{matrix}$$

$$\begin{matrix} A & B & C & D \end{matrix}$$

$$\begin{bmatrix} 1 & 2 & 4 & 5 \\ 1 & 1 & 3 & 4. \end{bmatrix} \begin{matrix} X \\ Y \end{matrix}$$

Bước 9. Nhóm các đối tượng vào nhóm

$$G^2 = \begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \end{bmatrix} \begin{matrix} group-1 \\ group-2 \end{matrix}$$

$$\begin{matrix} A & B & C & D \end{matrix}$$

Ta thấy $G^2 = G^1$ (Không có sự thay đổi nhóm nào của các đối tượng) nên thuật toán dừng và kết quả phân nhóm như sau:

Object	Feature 1 (X): weight index	Feature 2 (Y): pH	Group
Medicine A	1	1	1
Medicine B	2	1	1
Medicine C	4	3	2
Medicine D	5	4	2

4.2 K-Nearest neighbors

4.2.1 Giới thiệu

Thuật toán K-láng giềng gần nhất (KNN) là một loại thuật toán máy học có giám sát có thể được sử dụng cho cả phân loại cũng như các bài toán dự đoán hồi quy. Tuy nhiên, nó chủ yếu được sử dụng để phân loại các vấn đề dự đoán trong công nghiệp. Hai thuộc tính sau sẽ xác định tốt KNN:

- Lazy learning algorithm - KNN là một thuật toán lười học vì nó không có giai đoạn huấn luyện chuyên biệt và sử dụng tất cả dữ liệu để huấn luyện trong khi phân loại.
- Non-parametric learning algorithm - KNN cũng là một thuật toán học phi tham số vì nó không giả định bất cứ điều gì về dữ liệu bên dưới.

4.2.2 Quy trình hoạt động của thuật toán KNN

Thuật toán K-láng giềng gần nhất (KNN) sử dụng 'tính năng tương tự' để dự đoán giá trị của các điểm dữ liệu mới, điều này có nghĩa là điểm dữ liệu mới sẽ được chỉ định một giá trị dựa trên mức độ phù hợp chặt chẽ của nó với các điểm trong tập huấn luyện. Chúng tôi có thể hiểu cách hoạt động của nó với sự trợ giúp của các bước sau:

Bước 1: Để thực hiện bất kỳ thuật toán nào, chúng ta cần tập dữ liệu. Vì vậy trong bước đầu tiên của KNN, chúng ta phải tải dữ liệu huấn luyện cũng như kiểm tra.

Bước 2: Tiếp theo, chúng ta cần chọn giá trị của K tức là các điểm dữ liệu gần nhất. K có thể là bất kỳ số nguyên nào.

Bước 3: Đối với mỗi điểm trong dữ liệu kiểm tra, hãy làm như sau:

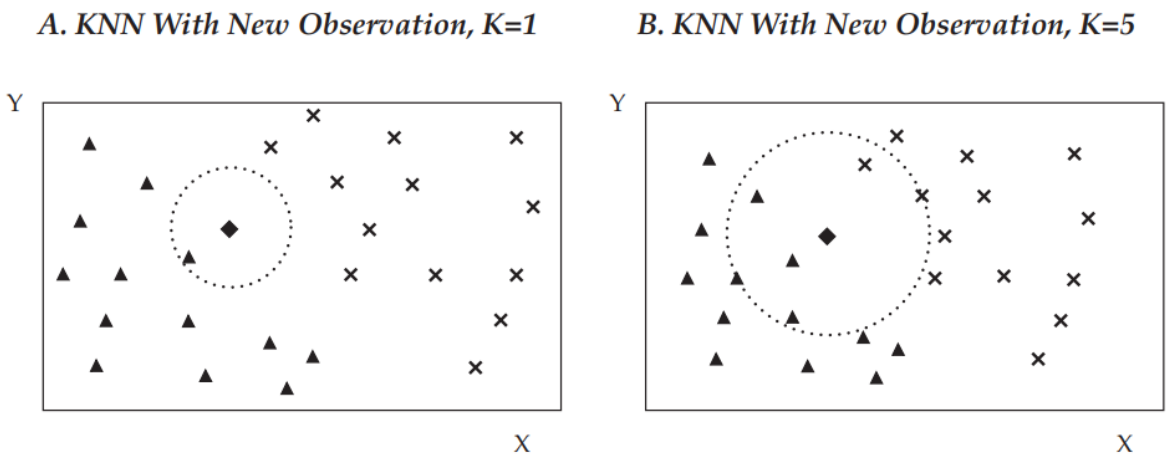
- Tính toán khoảng cách giữa dữ liệu kiểm tra và mỗi hàng dữ liệu đào tạo với sự trợ giúp của bất kỳ phương pháp nào cụ thể là: Khoảng cách Euclidean, Manhattan hoặc Hamming. Phương pháp thông dụng nhất để tính khoảng cách là Euclidean.
- Bây giờ, dựa trên giá trị khoảng cách, hãy sắp xếp chúng theo thứ tự tăng dần.
- Tiếp theo, nó sẽ chọn K hàng trên cùng từ mảng đã sắp xếp.
- Bây giờ, nó sẽ chỉ định một lớp cho điểm kiểm tra dựa trên lớp thường xuyên nhất của các hàng này.

Bước 4: Kết thúc

Ví dụ: Hình thoi trong Hình 1 đang cần được phân loại thuộc hình chữ thập hoặc hình tam giác.

- Nếu K = 1, hình thoi sẽ được phân loại vào cùng loại với điểm tài liệu gần nhất của nó (tức là hình tam giác trong bảng bên trái – bảng A).
- Bảng bên phải (bảng B) thể hiện trường hợp K = 5, thuật toán sẽ xem xét 5 điểm dữ liệu gần hình thoi nhất, đó là 3 hình tam giác và 2 hình chữ thập.

Qui tắc quyết định là chọn phân loại có số lượng lớn nhất trong 5 điểm dữ liệu được xem xét. Vì vậy, trong trường hợp này, hình thoi cũng được xếp vào phân loại tam giác.



Hình 4.2 – Mô tả thuật toán KNN

4.2.3 Ưu và nhược điểm của KNN

a) Ưu điểm

- Là một thuật toán rất đơn giản để hiểu và giải thích.
- Hữu ích cho dữ liệu phi tuyến vì không có giả định về dữ liệu trong thuật toán này.
- Thuật toán linh hoạt vì chúng ta có thể sử dụng nó để phân loại cũng như hồi quy.
- Độ chính xác tương đối cao nhưng có nhiều mô hình học có giám sát tốt hơn KNN.

b) Nhược điểm

- Là một thuật toán hơi tốn kém vì nó lưu trữ tất cả các dữ liệu huấn luyện.
- Yêu cầu bộ nhớ lưu trữ cao so với các thuật toán học có giám sát khác.
- Dự đoán chậm trong trường hợp N lớn.
- Rất nhạy cảm với quy mô dữ liệu cũng như các tính năng không liên quan.

4.3 Phân cụm đa cấp

Thuật toán phân cụm K-means cho thấy cần phải cấu hình trước số lượng cụm cần phân chia. Ngược lại, phương pháp phân cụm phân cấp (Hierachical Clustering) không yêu cầu khai báo trước số lượng cụm. Thay vào đó, thuật toán chỉ yêu cầu xác định trước thước đo về sự khác biệt giữa các cụm (không giao nhau), dựa trên sự khác biệt từng cặp giữa các quan sát trong hai cụm. Theo phương pháp này, chúng tạo ra những biểu

diễn phân cấp trong đó các cụm ở mỗi cấp của hệ thống phân cấp được tạo bằng cách hợp nhất các cụm ở cấp độ thấp hơn bên dưới. Ở cấp thấp nhất, mỗi cụm chứa một quan sát. Ở cấp cao nhất, chỉ có một cụm chứa tất cả dữ liệu.

Các chiến lược phân cụm phân cấp chia thành hai mô hình cơ bản: Hợp nhất (agglomerative) và phân chia (divisive). Trước khi tìm hiểu về hai chiến lược này, tôi khuyến nghị bạn đọc ôn tập lại kiến thức cây quyết định để nắm rõ các thành phần trong cây quyết định. Trục hoành thể hiện index của các quan sát trong nhóm được phân vào một cụm, trong khi tục tung là gía trị thước đo sự khác biệt giữa các cụm.Một cụm được đại diện bởi một node mà toàn bộ các quan sát khác nếu thuộc cụm thì đều liên kết tới node đó. Như vậy chúng ta có thể nhận thấy rằng các cụm có sự phân cấp dựa vào level của node. Khi kẻ một đường thẳng nằm ngang cắt toàn bộ các đường thẳng thẳng đứng ta sẽ thu được các cụm tương ứng với các node nằm gần nhất bên dưới đường thẳng. Bất kì hai cụm nào trong số chúng sẽ không chồng lấn nhau.

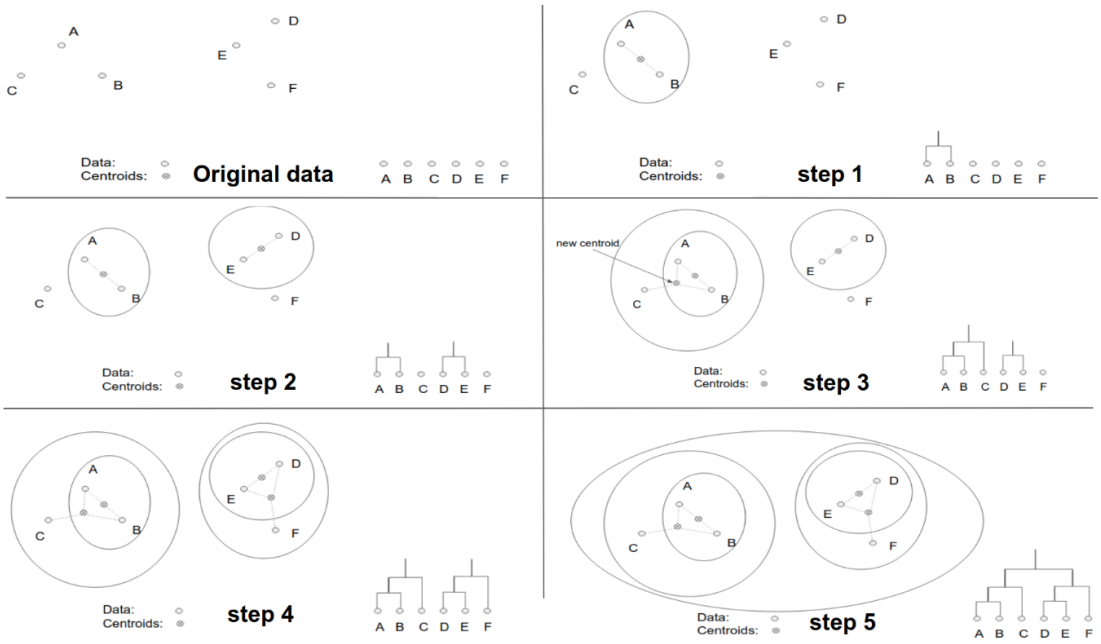
Thuật toán phân cụm phân cấp được xây dựng trên bộ dữ liệu có kích thước thì sẽ trải qua tổng cộng bước phân chia. Có hai chiến lược phân chia chính phụ thuộc vào chiều di chuyển trên biểu đồ dendrogram mà chúng ta sẽ tìm hiểu bên dưới:

- Chiến lược hợp nhất: Chiến lược này sẽ đi theo chiều bottum-up (từ dưới lên trên). Quá trình phân cụm bắt đầu ở dưới cùng tại các node lá (còn gọi là leaf node hoặc termial node). Ban dầu mỗi quan sát sẽ được xem là một cụm tách biệt được thể hiện bởi một node lá. Ở mỗi level chúng ta sẽ tìm cách hợp một cặp cụm thành một cụm duy nhất nhằm tạo ra một cụm mới ở level cao hơn tiếp theo. Cụm mới này tương ứng với các node quyết định (non-leaf node). Như vậy sau khi hợp cụm thì số lượng cụm ít hơn. Một cặp được chọn để hợp nhất sẽ là những cụm trung gian không giao nhau.
- Chiến lược phân chia: Chiến lược này sẽ thực hiện theo chiều top-down. Tức là phân chia bắt đầu từ node gốc của đồ thị. Node gốc bao gồm toàn bộ các quan sát, tại mỗi level chúng ta phân chia một cách đệ qui các cụm đang tồn tại tại level đó thành hai cụm mới. Phép phân chia được tiến hành sao cho tạo thành

hai cụm mới mà sự tách biệt giữa chúng là lớn nhất. Sự tách biệt này sẽ được đo lường thông qua một thước đo khoảng cách mà ta sẽ tìm hiểu kĩ hơn bên dưới.

4.3.1 Chiến lược hợp nhất

Chiến lược hợp nhất sẽ bắt đầu biểu diễn mỗi quan sát là một cụm đơn lẻ. Giả định chúng ta có N quan sát, thuật toán cần thực hiện N – 1 bước để hợp nhất hai nhóm có khoảng cách gần nhất lại với nhau và đồng thời giảm số lượng cụm trước khi chúng đạt được tới node gốc gồm toàn bộ các quan sát.



Hình 4.3: Hình minh hoạ các bước được thực hiện trên thuật toán phân cụm phân cấp sử dụng chiến lược hợp nhất đối với 6 điểm dữ liệu {A, B, C, D, E, F}. Chấm tròn thể hiện cho các điểm dữ liệu, chấm tròn có dấu x ở giữa là tâm của các cụm. Các đường elipse bao ngoài thể hiện cho các điểm được phân về cùng một cụm. Ở bên phải dưới cùng của mỗi hình là đồ thị dendrogram thể hiện sự gộp nhóm.

Bộ dữ liệu ở hình 2 bao gồm 6 điểm nên sẽ trải qua 5 bước dữ liệu để nhóm dữ liệu. Thứ tự nhóm sẽ như sau:

Bước 1: Dựa trên khoảng cách gần nhất giữa các điểm chúng ta sẽ nhóm 2 điểm {A, B} thành 1 cụm. Khi đó điểm đại diện cho một cụm {A, B} sẽ là trung bình cộng giữa hai điểm A và B, được thể hiện bằng dấu ⊗ giữa A và B trên hình.

Bước 2: Lựa chọn ngẫu nhiên một điểm chưa được gộp cụm, chẳng hạn điểm D. Đo khoảng cách tới các điểm còn lại và với tâm cụm {A, B} ta sẽ thu được khoảng cách d(D, E} là nhỏ nhất. Như vậy ta sẽ thu được một cụm {D, E}.

Bước 3: Xuất phát từ điểm C, ta đo khoảng cách tới các tâm cụm {A, B} và {D, E} và tới điểm F. Khoảng cách gần nhất là d(C,{A, B}) nên ta nhóm C vào cụm {A,B} để thu được cụm mới .

Bước 4: Xuất phát từ F ta đo khoảng cách tới các tâm cụm {A, B, C} và {D, E}. Điểm F gần cụm {D, E} hơn nên sẽ được gộp vào thành cụm {D, E, F}.

Bước 5: Gộp cả 2 cụm {A, B, C} và {D, E, F} ta thu được cụm cuối cùng là node gốc bao trùm toàn bộ dữ liệu.

4.3.2 Chiến lược phân chia (divisive)

Chiến lược phân chia chưa được nghiên cứu và phát triển rộng rãi trong các bài toán phân cụm như hợp nhất. Trong sklearn cũng chưa có module phát triển cho phương pháp này. Nó được giới thiệu lần đầu trong một tài liệu của Gersho và Grey, 1992 về kĩ thuật nén. Chiến lược phân chia sẽ bắt đầu từ một cụm gồm toàn bộ các quan sát bên trong cụm và sau đó phân chia đệ qui những cụm đang tồn tại thành hai cụm con tại mỗi bước theo hướng top-down.

Đầu tiên thuật toán sẽ chọn ra một điểm từ toàn bộ tập dữ liệu S sao cho điểm này thoả mãn điều kiện trung bình khoảng cách từ điểm đó tới toàn bộ những điểm còn lại là nhỏ nhất. Chúng ta đưa điểm này vào tập S_1, tập còn lại gồm N – 1 điểm là tập S_2. Tiếp theo ta sẽ thực hiện các lượt phân chia sao cho mỗi một lượt lựa chọn ra một điểm x_1 từ tập S_2 đưa sang S_1. Điểm này cần thoả mãn hai điều kiện:

Trung bình khoảng cách từ điểm đó tới toàn bộ các điểm còn lại trong S_1 phải là nhỏ

nhất. Điều đó có nghĩa là x_1 là điểm tách biệt nhất so với phần còn lại của S_1.

$$x_i = argmax \frac{1}{|S_1| - 1} \sum_{j=1, j \neq i}^{|S_1|} d(x_i, x_j)$$

Khoảng cách tối thiểu từ x_i tới các điểm trong S_2 phải lớn hơn khoảng cách tối thiểu tới các điểm trong S_1. Điều này nhằm mục đích khiến cho điểm x_i phải gần với cụm S_2 hơn cụm S_1.

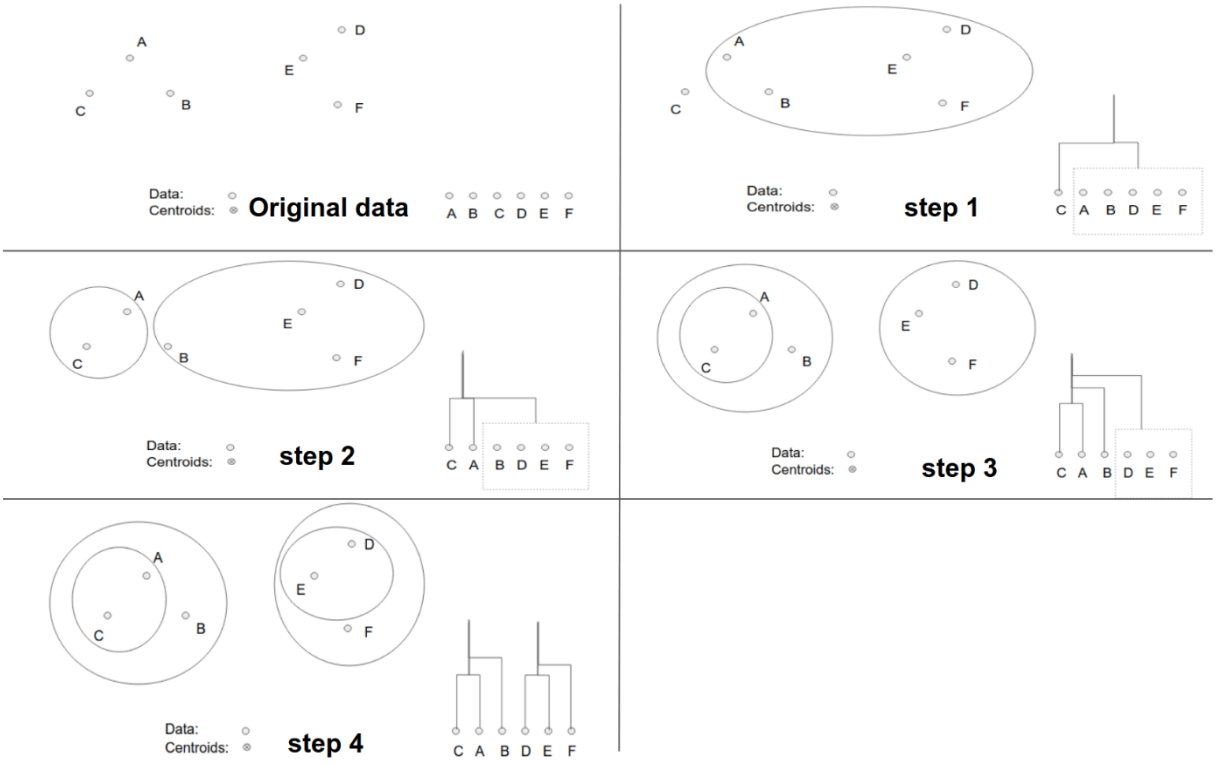
$$d(x_i, S_1) \geq d(x_i, S_2)$$

Trong đó:

$$d(x_i, S_k) = min_{x_j, x_j \in S_k} d(x_i, x_j)$$

Quá trình chuyển cụm sẽ kết thúc khi không còn điểm nào thoả mãn hai điều kiện trên. Khi đó chúng ta lại thực hiện đệ qui lại quá trình trên trên từng tập S_1 và S_2.

Chúng ta cùng diễn giải lại quá trình này thông qua hình minh hoạ bên dưới:



Hình 4.4: Hình minh hoạ phương pháp phân chia trong thuật toán phân cụm phân cấp. Ở bước 1 chúng ta sẽ lựa chọn ra điểm C là điểm đầu tiên thuộc cụm mới dựa trên khoảng cách so với các điểm còn lại là xa nhất. Sau bước 1 ta thu được tập S_1 = {C} và S_2 = {A,B,D,E,F}. Tại bước 2 lựa chọn trong số các điểm thuộc S_2 ra điểm mà có khoảng cách xa nhất so với những điểm còn lại sao cho điểm này gần với C hơn so với các điểm thuộc tập S_2, đó chính là điểm A. Di chuyển điểm này sang S_1. Bước 3 chúng ta lại tiếp tục thực hiện như vậy và lựa chọn được điểm B để đưa sang S_1. Ở bước thứ 4 ta sẽ dừng quá trình chuyển cụm cho các điểm thuộc S_2 vì thuật toán đã đạt sự hội tụ về hai cụm. Khi đó ta lại tiếp tục tiến hành đệ qui thuật toán trên từng cụm con.

CHƯƠNG 5: KHO DỮ LIỆU VÀ PHÂN TÍCH KẾT HỢP

5.1 Kho dữ liệu (Data Warehouse)

5.1.1 Khái niệm

Data Warehouse tạm dịch là kho dữ liệu thường được viết tắt là DW hay DWH. Về cơ bản có thể hiểu DW là một tập hợp các dữ liệu, thông tin có chung một chủ đề, được tổng hợp từ nhiều nguồn khác nhau trong nhiều mốc thời gian và không chỉnh sửa. Được dùng cho việc hỗ trợ ra quyết định, phân tích dữ liệu và lập báo cáo trong công tác quản lý.

Hiện nay khái niệm kho dữ liệu được phát triển rộng hơn, nó mô tả tập hợp các công nghệ, phương pháp, kỹ thuật có thể kết hợp với nhau thực hiện các chức năng tích hợp, lưu trữ, xử lý và phân tích dữ liệu để cung cấp thông tin cho người sử dụng. Một kho dữ liệu thường có dung lượng lên đến hàng trăm GB thậm chí tính bằng đơn vị TB.

Quá trình tập hợp và thao tác trên các dữ liệu này có những đặc điểm sau (ACID):

- Atomicity (Tính nguyên tử): dữ liệu được tập hợp từ nhiều nguồn khác nhau → khi tập hợp phải thực hiện làm sạch, sắp xếp, rút gọn dữ liệu.
- Consistency (Tính nhất quán): chỉ lấy những dữ liệu có ích (các dữ liệu có cùng chủ đề).
- Isolation (Tính cô lập): Các dữ liệu truy suất không bị ảnh hưởng bởi các dữ liệu khác hoặc tác động lên nhau.
- Durable (Tính bền vững): Dữ liệu không thể tạo thêm, xóa hay sửa đổi.

5.1.2 Mục đích của kho dữ liệu

– Hỗ trợ để các nhân viên của tổ chức thực hiện tốt, hiệu quả công việc của mình, như có những quyết định hợp lý, nhanh và bán được nhiều hàng hơn, năng suất cao hơn, thu được lợi nhuận cao hơn, …

– Giúp cho tổ chức, xác định, quản lý và điều hành các dự án, các nghiệp vụ một cách hiệu quả và chính xác.

– Tích hợp dữ liệu và các siêu dữ liệu từ nhiều nguồn khác nhau

5.1.3 Mục tiêu của kho dữ liệu

Một Data Warehouse phải đảm bảo được các mục tiêu sau:

a) Truy cập dễ dàng

Thông tin lưu trữ trong DW phải trực quan và dễ hiểu với người dùng. Dữ liệu nên được trình bày thông qua các tên gọi quen thuộc và gần gũi với nghiệp vụ của người dùng.

Tốc độ truy cập data warehouse phải nhanh. Do phải xử lý một số lượng bản ghi lớn cùng một lúc nên đây là một trong những yêu cầu cần phải có của một DW

b) Thông tin nhất quán

Dữ liệu trong một DW thường đến từ nhiều nguồn khác nhau. Do vậy trước khi được đưa vào DW dữ liệu cần phải được làm sạch và đảm bảo về chất lượng. Việc làm sạch sẽ giúp cho việc đồng nhất dữ liệu trở nên dễ dàng

Một nguyên tắc được đặt ra cho quá trình này là:

– Nếu dữ liệu có cùng tên thì bắt buộc phải chỉ đến cùng một địa chỉ.
– Nếu dữ liệu chỉ đến các thực thể khác nhau thì phải được đặt tên khác nhau

c) Thích nghi với sự thay đổi

DW cần phải được thiết kế để xử lý những thay đổi có thể xảy ra. vì thay đổi là diều không thể tránh khỏi cho bất cứ ứng dụng nào. Nói vậy có nghĩa là khi có thay đổi mới dữ liệu cũtrong DW vẫn phải đảm bảo tính đúng đắn.

d) Hỗ trợ ra quyết định

Đây là mục tiêu quan trọng nhất của doanh nghiệp khi xây dựng DW. Những người quản lý doanh nghiệp muốn dưa vào thông tin để từ đó đưa ra những chiến lược góp phần đem lại kết quả kinh doanh tốt nhất

d) Bảo mật

Dữ liệu trong DW đến từ nhiều nguồn khác nhau. Vì vậy việc đảm bảo thông tin không bị lộ ra ngoài là một điều vô cùng quan trọng.

5.1.4 Các chức năng chính

- Phân hệ tích hợp dữ liệu
- Phân hệ phân tích dữ liệu
- Phân hệ giám sát hệ thống
- Phân hệ sao lưu và phục hồi hệ thống
- Phân hệ bảo mật dữ liệu

5.1.5 Lợi ích

a) Đối với người khai thác

Cung cấp công cụ phân tích, khai thác dữ liệu nhanh gọn, đầy đủ và chính xác, dễ dàng đưa ra các chính sách mới.

Giúp người sử dụng khai thác dữ liệu theo chủ đề với các nguồn và khoảng thời gian khác nhau oDữ liệu được xử lý nhanh chóng

Dễ dàng tạo ra các báo cáo đơn giản phù hợp với nhiều trình độ khai thác

b) Đối với người quản trị hệ thống

Hỗ trợ xây dựng một kho dữ liệu lớn

Thiết kế mềm dẻo giúp dễ dàng tích hợp dữ liệu tác nghiệp mới và tạo ra các báo cáo mới theo yêu cầu người khai thác.

5.1.6 Đặc tính của kho dữ liệu

- Tính tích hợp (Integration): Dữ liệu tập hợp từ nhiều nguồn khác nhau. Điều này sẽ dẫn đến việc quá trình tập hợp phải thực hiện việc làm sạch, xắp xếp, rút gọn dữ liệu.
- Dữ liệu gắn thời gian và có tính lịch sử. Các dữ liệu đến từ quá trình kinh doanh của công ty có thể có từ nhiều năm trước.

- Dữ liệu có tính ổn định (nonvolatility): Khi một Transaction hoàn chỉnh, dữ liệu không thể tạo thêm hay sửa đổi.
- Dữ liệu không biến động
- Dữ liệu tổng hợp

Dữ liệu tổng hợp nhanh (lightly summarized data) là dấu hiệu xác nhận chất lượng của một kho dữ liệu. Tất cả các yếu tố của công việc kinh doanh (phòng ban, lĩnh vực hoạt động, chức năng hoạt động, ...) có những yêu cầu thông tin khác nhau, vì thế việc thiết kế kho dữ liệu phải có kết quả cung cấp dữ liệu tuỳ biến, tổng hợp nhanh cho mỗi yếu tố doanh nghiệp (xem thêmphần kho dữ liệu thông minh bên dưới). Mỗi yếu tố của công việc kinh doanh có thể có truy cập đến dữ liệu chi tiết và tổng hợp, nhưng sẽ không có nhiều hơn tổng số dữ liệu được lưu trữ trong chi tiết hiện hành.

Dữ liệu tổng hợp chất lượng cao (hightly summarized data) là căn bản cho việc tiến hành công việc kinh doanh. Dữ liệu tổng hợp chất lượng cao có thể đến từ dữ liệu tổng hợp nhanh được dùng cho các yếu tố công việc kinh doanh hoặc từ chi tiết hiện hành. Số lượng dữ liệu ở mức độ này có ít hơn ở các mức độ khác, nó mô tả một tập hợp được chọn lọc cung cấp một sự sự đa dạng rộng lớn cho các nhu cầu và các sự quan tâm. Thêm vào đó để truy cập đến dữ liệu tổng hợp chất lượng cao, việc tiến hành nói chung cũng cần có khả năng tăng mức độ cập nhật chi tiếtthông qua tiến trình khoan đi xuống (drill down)

5.1.7 Cấu trúc dữ liệu cho kho dữ liệu

Vì dữ liệu trong kho dữ liệu rất lớp và không có những thao tác như sửa đổi hay tạomới nên nó được tối ưu cho việc phân tích và báo cáo

Các thao tác với dữ liệu của khodữ liệu dựa trên cơ sở là Mô hình dữ liệu đa chiều (multidimensional data model), được mô hình vào đối tượng gọi là data cube.

Data cube là nơi trung tâm của vấn đề cần phân tích, nó bao gồm một hay nhiều tập dữ kiện (fact) và các dữ kiện được tạo ra từ nhiều chiều dữ kiện khác nhau (dimention).

Ví dụ: Một thống kê doanh số bán hàng dựa trên ba yếu tố là: địa điêm, thời gian và chủng loại hàng. Data cube là vấn đề "Thống kê bán hàng" với ba chiều là ba yếu tố: địa điểm, thời gian và chủng loại hàng. Bảng fact là bảng tổng hợp dữ liệu của mối liên quan của doanh số với 3 yếu tố. trong SQL).

5.1.8 Kiến trúc của một hệ thống kho dữ liệu

Kiến trúc kho dữ liệu mô tả các cấu kiện, công cụ và dịchvụ của kho dữ liệu, cũng như quan hệ và sự phát triển của chúng. Mục đích của việc chuẩn hoákiến trúc kho dữ liệu là tích hợp các hệ thông tin cấp dưới để phục vụ các hệ thông tin cấp trên và ngược lại. Kiến trúc này cung cấp một cơ chế tổ chức dữ liệu, cải thiện việc chia sẻ thông tin giữa các cơ quan và về lâu dài có khả năng tái sử dụng dữ liệu cũng như phát triển các dự án kho dữ liệu tiếp theo được nhanh hơn

Bao gồm ba tầng:

- Tầng đáy: Là nơi cung cấp dịch vụ lấy dữ liệu từ nhiều nguồn khác sau đó chuẩn hóa, làm sạch và lưu trữ dữ liệu đã tập tung.
- Tầng giữa: cung cấp các dịch vụ để thực hiện các thao tác với kho dữ liệu gọi là dịch vụ OLAP (OLAP server). Có thể cài đặt bằng Relational OLAP, Multidimensional OLAP hay kết hợp cả hai mô hình trên Hybrid OLAP.
- Tầng trên cùng: nơi chứa các câu truy vấn, báo cáo, phân tích

5.1.9 Mối quan hệ giữa kho dữ liệu và khai phá dữ liệu

Cả hai đều có thể đứng độc lập với nhau, tuy nhiên khi kết hợp được kho dữ liệu với khai phá dữ liệu thì lợi ích rất lớn vì các lý do như:

- Dữ liệu của kho dữ liệu rất phù hợp cho việc khai phá dữ liệu (Data Mining) do đã được tập hợp và làm sạch.
- Cơ sở hạ tầng của kho dữ liệu hỗ trợ rất tốt cho các việc như xuất, nhập cũng như các thao tác cơ bản trên dữ liệu.
- OLAP cung cấp các tập lệnh rất hữu hiệu trong phân tích dữ liệu

5.1.10 Các lĩnh vực ứng dụng

Có thể đưa kho dữ liệu vào ba hướng ứng dụng chính cần đến trí tuệ kinh doanh (Business Intelligence)

- Xử lý thông tin như tạo ra các báo cáo và trả lời các câu hỏi định trước.
- Phân tích và tổng hợp dữ liệu, kết quả được thể hiện bằng các báo cáo và bảng biểu.
- Dùng cho các dự án có mục đích kế hoạch hoá như khai phá dữ liệu

Các lĩnh vực hiện tại có ứng dụng kho dữ liệu bao gồm:

- Thương mại điện tử.
- Kế hoạch hoá nguồn lực doanh nghiệp (ERP -EnterpriseResource Planning).
- Quản lý quan hệ khách hàng (CRM -Customer Relationship Management)
- Chăm sóc sức khỏe.
- Viễn thông.

5.2 Mẫu phổ biến và luật kết hợp

5.2.1 Mẫu phổ biến

a) Tập mục

Gọi $I = \{x_1, x_2, \dots, x_n\}$ là tập n mục (item). Một tập $X \subseteq I$ được gọi là một tập mục (itemset). Nếu X có k mục (tức $|X| = k$) thì X được gọi là k–itemset.

Ví dụ:

Hình 5.1 Ví dụ về mẫu phổ biến và luật kết hợp

Tập tất cả các mặt hàng thực phẩm trong siêu thị: I = {sữa, trứng, đường, bánh mỳ, mật ong, mứt, bơ, thịt bò, giá, … }.

Tập tất cả các bộ phim: I = {pearl harbor, fast and furious 7, fifty shades of grey, spectre, … }.

b) Giao dịch

Ký hiệu D = {T_1, T_2, …, T_m} là cơ sở dữ liệu gồm m giao dịch (transaction). Mỗi giao dịch $T_i \in D$ là một tập mục, tức $T_i \subseteq I$.

VD: Cơ sở giao dịch

Tập tất cả các mục I = {A, B, C, D, E}. Cơ sở dữ liệu giao dịch D = {T_1, T_2, T_3, T_4, T_5, T_6} trong đó:

T_1 = {A, B, D, E}	T_4 = {A, B, C, E}
T_2 = {B, C, E}	T_5 = {A, B, C, D, E}
T_3 = {A, B, D, E}	T_6 = {B, C, D}

Tập mục I là các sản phẩm trong siệu thị, Cơ sở giao dịch là những đơn mua của khách hàng.

T_1 = {sữa, trứng, đường, bánh mỳ}

T_2 = {sữa, mật ong, mứt, bơ}

T_3 = {trứng, mì tôm, thịt bò, giá}

c) Mẫu phổ biến

Frequent patterns – mẫu phổ biến được biết đến như: các tập mục – itemsets, dãy con – subsequence, hoặc cấu trúc con – substructures, là những mẫu xuất hiện phổ biến trong một tập dữ liệu.

Cho tập mục X ($\subseteq$ I)

Độ hỗ trợ của X, kí hiệu là sup(X, D), là số lượng giao dịch trong D chứa tập X:

sup(X, D) = |{T|T $\subseteq$ D và X $\subseteq$ T}|

Độ hỗ trợ tương đối của X, kí hiệu là rsup(X, D) là số phần trăm các giao dịch trong D chứa X:

rsup(X, D) = sup(X, D)/|D|

Tập mục X được gọi là tập phổ biến trong cơ sở giao dịch D nếu sup(X, D) $\geq$ minsup, với minsup là một ngưỡng độ hỗ trợ tối thiểu (minimum support threshold) do người dùng định nghĩa.

F là kí hiệu của tất cả các tập phổ biến

$F^{(k)}$ là kí hiệu của tập các tập phổ biến có độ dài k

VD: Các tập phổ biến (với minsup = 3) từ cơ sở dữ liệu D (tức số lần xuất hiện của tập trong 6 giao dịch $\geq$ 3) D = {T_1, T_2, T_3, T_4, T_5, T_6} trong đó:

T_1 = {A, B, D, E}	T_4 = {A, B, C, E}
T_2 = {B, C, E}	T_5 = {A, B, C, D, E}

T_3 = {A, B, D, E}	T_6 = {B, C, D}

Ta có tập các tập phổ biến là:

F = {A, B, C, D, E, AB, AD, AE, BC, BD, BE, CE, DE, ABD, ABE, ADE, BCE, BDE, ABDE}

$F^{(1)}$ = {A, B, C, D, E}

$F^{(2)}$ = {AB, AD, AE, BC, BD, BE, CE, DE}

$F^{(3)}$ = {ABE, ABD, ADE, BCE, BDE}

$F^{(4)}$ = {ABDE}

5.2.2 Luật kết hợp

Luật kết hợp là mối quan hệ giữa các tập thuộc tính trong cơ sở dữ liệu. Luật kết hợp là phương tiện hữu ích để khám phá các mối liên kết trong dữ liệu.

Một luật kết hợp là một mệnh đề kéo theo có dạng X → Y, trong đó X, Y ⊆ I, thỏa mãn điều kiện X giao Y = rỗng. Các tập hợp X và Y được gọi là các tập hợp thuộc tính (itemset). Tập X gọi là nguyên nhân, tập Y gọi là hệ quả. Có 2 độ đo quan trọng đối với luật kết hợp: Độ hỗ trợ (support) và độ tin cậy (confidence), được định nghĩa như phần dưới đây.

a) Độ hỗ trợ

Độ hỗ trợ của một luật kết hợp X → Y là tỷ lệ giữa số lượng các bản ghi chứa tập hợp X → Y, so với tổng số các bản ghi trong $\mathbb{D}$ - Ký hiệu sup(X → Y).

$$\sup(X \to Y, \mathbb{D}) = \sup(X \cup Y, \mathbb{D})$$

Độ hỗ trợ tương đối của luật X→Y trong cở sở dữ liệu $\mathbb{D}$ kí hiệu là rsup(X→Y, $\mathbb{D}$) là số phần trăm các giao dịch trong $\mathbb{D}$ chứa cả X và Y.

$$\text{rsup}(X \to Y, \mathbb{D}) = \frac{\sup(X \cup Y, \mathbb{D})}{|\mathbb{D}|}$$

Nếu độ hỗ trợ của một kết kết hợp X → Y là 30% thì có nghĩa là 30% tổng số bản ghi chứa X hợp Y. Như vậy độ hỗ trợ mang ý nghĩa thống kê của luật.

b) Độ tin cậy

Độ tin cậy (confidence) của luật X → Y trong $\mathbb{D}$, ký hiệu conf (X → Y , $\mathbb{D}$), là tỉ lệ giữa số giao dịch chứa cả X và Y trên số giao dịch chỉ chứa X.

$$\text{conf}(X \rightarrow Y, \mathbb{D}) = \frac{\text{sup}\ (X \cup Y, \mathbb{D})}{\text{sup}\ (X, \mathbb{D})}$$

Ký hiệu độ tin cậy của một luật r là conf(r). Ta có $0 \leq \text{conf(r)} \leq 1$

Độ hỗ trợ và độ tin cậy có xác xuất như sau:

- Độ hỗ trợ là xác xuất trong giao dịch chứa cả X và Y.
- Độ tin cậy là xác xuất có điều kiện mà một giao dịch trong $\mathbb{D}$ chứa Y trong khi đã chứa X (bản chất vẫn là mức độ in cậy của luật).

$$\text{Supp}(X \rightarrow Y) = \text{P}(X \cup Y)$$

$$\text{Conf}(X \rightarrow Y) = \text{P}(X/Y) = supp(X \cup Y)/supp(X)$$

Kết luận:

Luật X → Y được gọi là phổ biến nếu sup(X→Y, $\mathbb{D}$) ≥ minsup (minsup do người dùng định nghĩa)

Luật X→ Y được gọi là mạnh nếu độ tin cậy của nó lớn hơn hoặc bằng một ngưỡng minconf do người dùng định nghĩa: conf (X→Y) ≥ minconf

Ví dụ: Cơ sở giao dịch D = {T_1, T_2, T_3, T_4, T_5, T_6} trong đó:

T_1 = {A, B, D, E}	T_4 = {A, B, C, E}
T_2 = {B, C, E}	T_5 = {A, B, C, D, E}

T_3 = {A, B, D, E}	T_6 = {B, C, D}

- Xét luật {B, C} → {E} hay BC→E
  - sup(BC→E, D) = sup(BCE, D) = 3 (số lần xuất hiện của bộ ba BCE trong các giao dịch thuộc D)
  - conf(BC→E, D) = sup(BCE, D)/sup(BC, D) = 3/4 (=75%)
- Xét luật {A, D} → {B, E} hay AB → DE
  - Độ hỗ trợ sup(AD→BE, D) = sup(ABDE, D) = 3
  - Độ tin cậy conf(AD→BE, D) = sup(ABDE, D)/sup(AD, D) = 3/3 (=100%)

  Tức ta có thể đưa ra kết luận là nếu trong giao dịch có chứa AD thì chắc chắn sẽ chứa BE

c) Tính chất

– Tính chất 1: Giả sử A,B ⊆ I là hai tập hợp với A⊆B thì sup(A) ≥ sup(B). Như vậy, những bản ghi nào chứa tập hợp B thì cũng chứa tập hợp A

– Tính chất 2: Giả sử A, B là hai tập hợp, A,B ⊆ I, nếu B là tập phổ biến và A⊆B thì A cũng là tập phổ biến. Vì nếu B là tập phổ biến thì sup(B) ≥ minsup, mọi tập hợp A là con của tập hợp B đều là tập phổ biến trong cơ sở dữ liệu D vì sup(A) ≥ sup(B) (Tính chất 1)

– Tính chất 3: Giả sử A, B là hai tập hợp, A ⊆ B và A là tập hợp không thường xuyên thì B cũng là tập hợp không thường xuyên (Tính chất 1) (Tức nếu A là tập hợp không phổ biến thì mọi tập cha của nó cũng không biến)

5.3 Thuật toán Apriori

Apriori là thuật toán khả sinh được đề xuất bởi R. Agrawal và R. Srikant vào năm 1993 để khai thác các tập item đối với các luật kết hợp kiểu bool. Tên của thuật toán dựa trên việc thuật toán sử dụng tri thức trước (prior knowledge) của các thuộc tính tập item phổ biến, chúng ta sẽ thấy sau đây. Apriori dùng cách tiếp cận lặp được biết đến như tìm kiếm level-wise, với các tập k item được dùng để thăm dò các tập (k+1) item. Đầu tiên,

tập các tập 1 item phổ biến được tìm thấy bằng cách quét cơ sở dữ liệu để đếm số lượng từng item, và thu thập những item thỏa mãn độ hỗ trợ tối thiểu. Tập kết quả đặt là L_1. Tiếp theo, L_1 được dùng để tìm L_2, tập các tập 2 item phổ biến, nó được dùng để tìm L_3, và cứ thế tiếp tục, cho tới khi tập k item phổ biến không thể tìm thấy. Việc tìm kiếm cho mỗi L_k đòi hỏi một lần quét toàn bộ cơ sở dữ liệu.

Apriori dùng cách tiếp cận lặp được biết đến như tìm kiếm level-wise, với các tập k item được dùng để thăm dò các tập (k+1) item.

1. Đầu tiên, tập (frequent 1- itemsets) phổ biến 1 được tìm thấy ký hiệu là C_1
2. Bước tiếp theo là tính support có nghĩa là sự xuất hiện của các item trong cơ sở dữ liệu. Điều này đòi hỏi phải duyệt qua toàn bộ cơ sở dữ liệu.
3. Sau đó, bước cắt tỉa được thực hiện trên C1 trong đó những item được so sánh với thông số minimum support. Những item thỏa điều kiện minimum support mới được xem xét cho tiến trình tiếp theo ký hiệu là L_1.
4. Sau đó, bước phát sinh các bộ ứng viên được thực hiện trong đó tập phổ biến 2 được tạo ra ký hiệu là C_2.
5. Một lần nữa, cở sở dữ liệu được duyệt để tính toán support của 2 tập phổ biến. Theo minimum support, các bộ ứng viên tạo ra được kiểm tra và chỉ những tập phổ biến nào thỏa điều kiện minimum support thì tiếp tục được sử dụng tạo ra bô ứng viên tập phổ biến 3.

Bước trên tiếp tục cho đến khi không có tập phổ biến hoặc bộ ứng viên có thể được tạo ra.

5.4 Ví dụ Thuật toán Apriori

Bảng 1 biểu diễn một giao dịch cơ sở dữ liệu có 4 giao dịch. TID là một nhận dạng duy nhất cho mỗi giao dịch.

TID	Items

T001	A, C, D
T002	B, C, E
T003	A, B, C, E
T004	B, E

Bảng 1

Bước 1: K = 1. Tạo bảng chứa số support của từng mục có trong tập dữ liệu xác định số lượng sự xuất hiện cho một item cụ thể - Được gọi là C1 (tập ứng cử viên), được thể hiện trong Bảng 2

Itemset	Support
{A}	2
{B}	3
{C}	3
{D}	1
{E}	3

Bảng 2: C_1

Bước 2: So sánh số support của tập các ứng cử viên với số lượng hỗ trợ tối thiểu – minimum support (ở đây minimum support = 2 nếu Support của tập ứng cử viên nhỏ hơn minimum support sẽ xóa các tập đó). Điều này cung cấp cho chúng ta mục L_1 được thể hiện trong Bảng 3.

Itemset	Support
{A}	2
{B}	3
{C}	3
{E}	3

Bảng 3: L_1

Bước 3: K = 2 Tạo tập ứng viên C_2 bằng L_1 (đây được gọi là bước kết hợp). Duyệt qua các tập cha của C_2, nếu tập cha nào không đạt chuẩn thường xuyên thì tập con đó sẽ bị xóa. Bây giờ tính độ thường xuyên của các tập con mới được tạo. Bảng 4 cho thấy tất cả khả năng kết hợp mà có thể tạo ra từ Bảng 3 tập phổ biến.

Itemset	Support
{A, B}	1
{A, C}	2
{A, E}	1
{B, C}	2
{B, E}	3
{C, E}	2

Bảng 4: C_2

Bước 5: Tiếp tục kiểm tra độ thường xuyên của các tập trong C_2, nếu tập nào không thỏa mãn minimum support thì xóa đi. Ta sẽ nhận được kết quả là tập L_2. Sau khi cắt tỉa chúng ta nhận được kết quả như sau:

Itemset	Support
{A, C}	2
{B, C}	2
{B, E}	3
{C, E}	2

Bảng 5: L_2

Bước 6: Các quá trình tương tự được tiếp tục cho đến khi không có tập phổ biến hoặc bộ ứng viên có thể tạo ra. Tiến trình được mô tả trong Bảng 6 và Bảng 7.

Itemset	Support
{A, B, C}	1
{A, B, E}	1
{B, C, E}	2

Bảng 6:C_3

Itemset	Support
{B, C, E}	2

Bảng 7: L_3 (Kết quả cuối cùng)

Mã giả thuật toán:

Apriori_Algorithm()

{

C_k: Candidate itemset of size k

L_k : frequent itemset of size k

L_1 = {frequent items};

for (k = 1; L_k!=0; k++)

{

C_{k+1} = candidates generated from L_k;

foreach transaction t in database do

increment the count of all candidates in C_{k+1}

that are contained in t

L_{k+1} = candidates in C_{k+1} with min_support

}

return $\cup k L_k$;

}

Hạn chế của thuật toán Apriori:

- Số lượng lớn tập phổ biến được tạo ra làm gia tăng sự phức tạp không gian.
- Quá nhiều lần duyệt cơ sở dữ liệu được yêu cầu vì số lượng lớn tập phổ biến được tạo.
- Khi số lần duyệt cơ sở dữ liệu nhiều làm gia tăng sự phức tạp thời gian khi cơ sở dữ liệu gia tăng.

5.5 Sơ lược các phương pháp khác

Khai phá luật kết hợp là việc phát hiện ra các luật kết hợp thỏa mãn các ngưỡng độ hỗ trợ (minsup) và ngưỡng độ tin cậy (minconf) cho trước. Bài toán khai phá luật kết hợp gồm hai pha:

- Pha 1: Khai phá tất cả các tập phổ biến (FI) trong CSDL D với ngưỡng độ hộ trợ tối thiểu minsup (thường có độ tính toán cao và chiếm phần lớn thời gian trong khai phá luật kết hợp)
- Pha 2: Sinh ra tất cả các luật mạnh từ các tập phổ biến khai phá được từ pha trước với ngưỡng tiến cậy tối thiểu mà minconf.

5.4.1 Thuật toán 1 – Thuật toán cơ bản

Đầu vào: I, D, minsup, minconf

Đầu ra: Các luật kết hợp thỏa mãn ngưỡng độ hỗ trợ minsup, ngưỡng độ tin cậy minconf.
Thuật toán:

1. Tìm tất cả các tập hợp các tính chất có độ hỗ trợ ≥ minsup.
2. Từ các tập hợp mới tìm ra, tạo ra các luật kết hợp có độ tin cậy ≥ minconf.

5.4.2 Thuật toán 2- Tìm luật kết hợp khi đã biết các tập hợp thường xuyên

Đầu vào: I, D, minsup, minconf, tập phổ biến S

Đầu ra: Các luật kết hợp thỏa mãn ngưỡng độ hỗ trợ minsup, ngưỡng độ tin cậy minconf.

Thuật toán:

1. Lấy ra một tập phổ biến s⊆S, và một tập con X ⊆ s.
2. Xét luật kết hợp có dạng X → (sX), đánh giá độ tin cậy của nó xem có nhỏ hơn minconf hay không. Thực chất, tập hợp S mà ta xét đóng vai trò của tập hợp giao S = X hợp Y, và do X giao (S – X) = rỗng, nên coi như Y = S – X.

Các thuật toán xoay quanh khai phá luật kết hợp chủ yếu nêu ra các giải pháp để đẩy nhanh việc thực hiện Pha 1 là tìm tất cả các tập phổ biến.