#HCMUS #DataMining

Thông tin kỳ thi

Thời gian: 9h55 - 29/06/2024

Phòng thi: E403 (NVC)

Overview

Đây là note của toàn bộ môn học. Chuẩn bị cho kỳ thi cuối kỳ sắp tới.

Note sẽ bám theo sách **Data Mining The Textbook** gồm các nội dung sau:

1. An Introduce to Data Mining
2. Data Preparation
3. Similarity & Distances
4. Association Pattern Mining
5. Cluster Analysis
6. Outlier Analysis
7. Data Classification

1. Introduce to Data Mining

#DataMining/AssociationPatternMining
#DataMining/DataClustering
#DataMining/OutlierDetection
#DataMining/DataClassification
#DataMining/DataCollection
#DataMining/DataPreprocessing
#DataMining/DataPreprocessing/FeatureExtraction
#DataMining/StreamingData

1.1. Introduce

Data mining là quá trình collect, clean, process, analyzing và gain useful insights từ data của nhiều domain khác nhau.

Do đó, 'data mining' là cách gọi tổng quát để miêu tả các khía cạnh khác của data processing.

Các nhà phân tích dữ liệu sẽ sử dụng hệ thống xử lý, trong đó, các dữ liệu thô sẽ được:

- Thu thập
- Làm sạch
- Chuyển đổi thành một định dạng tiêu chuẩn hóa

Dữ liệu có thể có nhiều định dạng hoặc kiểu khác nhau.

- Định lượng (quantitative)
- Định tính (categorical || quanlitative)
- Văn bản (text)
- Không gian (spatial), thời gian (temporal)
- Biểu đồ (graph-oriented)

Hầu hết dữ liệu trong thực tế là multidimensional data. Các loại dữ liệu có cấu trúc phức tạp dần dần có tỷ lệ ngày càng tăng

1.2. The Data Mining Process

Data Mining Process là một pipeline chứa nhiều phases khác nhau như:

1. **Data Collection**: là quá trình yêu cầu sử dụng nhiều phương pháp khác nhau để collect dữ liệu. Đây là một quá trình rất quan trọng ảnh hưởng trực tiếp đến DMP (stand for Data Mining Process). Sau quá trình collection, data sẽ được lưu trữ ở database hoặc data warehouse (dành cho processing)
2. **Feature extraction và data cleaning**: Sau khi dữ liệu đã được collect, chúng thường không phù hợp để processing ngay. Trong nhiều trường hợp, data có nhiều type khác nhau sẽ mix lại với nhau, transform chúng, biến data thành dạng suitable type, chẳng hạn như multidimensional, time series, semistructured format. Thường quá trình feature extraction sẽ song song với data cleaning. Đầu ra cuối cùng sẽ là structured data set, có thể được sử dụng bởi computer program. Sau đấy, dữ liệu sẽ lại một lần nữa được lưu trữ ở database.
3. **Analytical processing and algorithms**: Đây là final part của DMP. Part này giúp đưa ra các useful insights hoặc các dự đoán có lợi

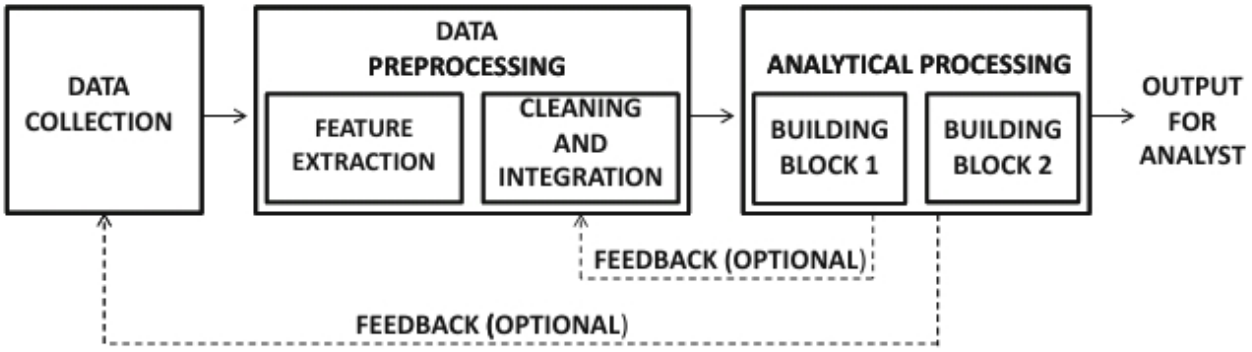


Figure 1.1: The data processing pipeline

1.2.1. The Data Preprocessing Phase

Đây là giai đoạn quan trọng nhất trong quá trình xử lý dữ liệu

Bao gồm các bước:

1. Feature Extraction
2. Data Cleaning
3. Feature selection and transformation

1.2.2. The Analytical Phase

1.3. The Basic Data Types

Có hai loại chính:

1.3.1. Dữ liệu định hướng không phụ thuộc

Là dạng dữ liệu đơn giản nhất và thường được gọi là dữ liệu nhiều chiều

Table 1.1: An example of a multidimensional data set

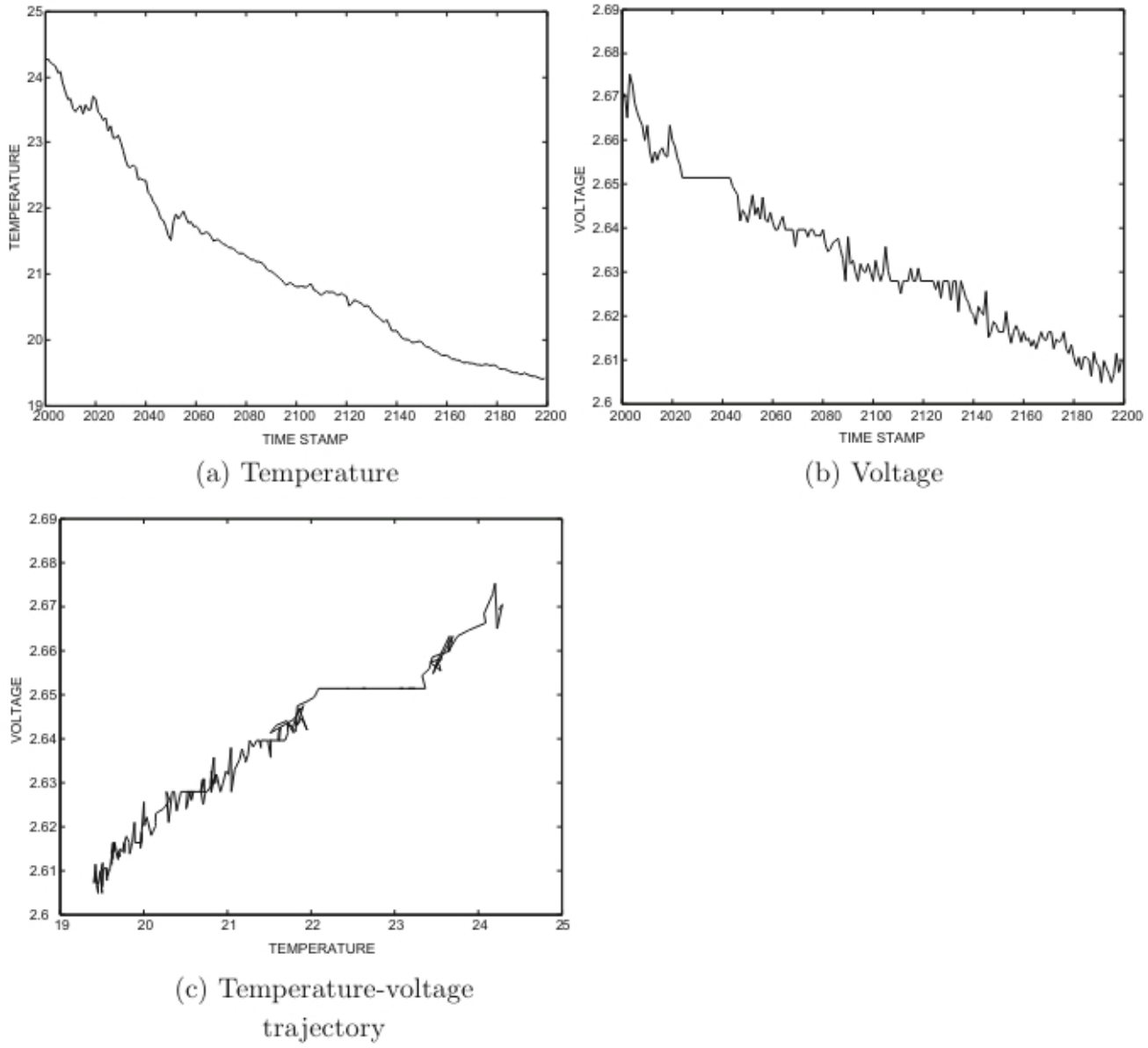
Name	Age	Gender	Race	ZIP code
John S.	45	M	African American	05139
Manyona L.	31	F	Native American	10598
Sayani A.	11	F	East Indian	10547
Jack M.	56	M	Caucasian	10562
Wei L.	63	M	Asian	90210

Có hai loại phụ thuộc:

1. Phụ thuộc ngầm: các phụ thuộc giữa các mục dữ liệu không được chỉ định rõ ràng nhưng chúng "thường" tồn tại trong lĩnh vực đó. Ví dụ: nhiệt độ đo được từ cảm biến ở các thời điểm khác nhau, nếu ở hai thời điểm gần nhau mà nhiệt độ chêch lệch lớn thì đấy là dấu hiệu của sự không bình thường.
2. Phụ thuộc tường minh: Thường ám chỉ đến dữ liệu đồ thị (graph) hoặc mạng (network data) trong đó các cạnh được sử dụng để chỉ định mối quan hệ rõ ràng.

Dữ liệu định hướng phụ thuộc có thể được phân thành các loại sau:

- Time-Series data: Dữ liệu chuỗi thời gian chứa các giá trị thường được tạo ra bởi việc đo liên tục trong thời gian
- Discrete Sequences and Strings: Dãy rời rạc có thể được coi là biến thể của dữ liệu chuỗi thời gian. Giống như dữ liệu chuỗi thời gian, contextual attribute (thuộc tính ngữ cảnh) là một time stamp hoặc position index. Behavioral attribute (thuộc tính hành vi) là một catogorical value. Do đó, dữ liệu dãy rời rạc được định nghĩa tương tự dữ liệu chuỗi thời gian.
- Spatial Data: bao gồm nhiều attribute không tuân theo không gian (nonspatial attributes) (ví dụ: nhiệt độ, áp suất...) được đo dựa trên không gian của chúng. Ví dụ: nhiệt độ bề mặt biển.
- Network and Graph Data:



(a) Temperature

(b) Voltage

(c) Temperature-voltage trajectory

Figure 1.2: Mapping of multivariate time series to trajectory data

1.3.2. Dữ liệu định hướng phụ thuộc

Được phân thành các loại như:

- Quantitative Multidimensional Data (dữ liệu nhiều chiều định lượng được)
- Categorical and Mixed Attribute Data (dữ liệu phân loại và dữ liệu thuộc tính trộn lẫn)
- Binary and Set Data (dữ liệu nhị phân và dữ liệu tập hợp)
- Text Data (dữ liệu văn bản)

Thông tin về sự phụ thuộc có từ trước có ảnh hưởng rất lớn đến DMP

1.4. The Major Building Blocks: A Bird’s Eye View

Association Pattern Mining
Data Clustering
Outlier Detection
Data Classification

Các mục này sẽ được đề cập rõ hơn sau

1.4.5. Impact of Complex Data Types on Problem Definitions

Các loại dữ liệu cụ thể sẽ gây ảnh hưởng lớn đến các bài toàn đặt ra, đặc biệt là các loại dữ liệu phức tạp

Ví du:

1. Nếu dữ liệu đầu vào của bài toán Association Pattern Mining là time-series thì thay vì chăm chăm nghiên cứu sự giống hay khác nhau thì ta nên xem xét tính tuần hoàn của dữ liệu dựa trên mốc thời gian.
2. Hoặc với bài toàn Data Clustering mà dữ liệu là time-series thì không thể sử dụng khoảng cách Euclidean để tính khoảng cách được, cần phải tìm một công thức khác

1.5. Scalability Issues and the Streaming Scenario

Với lượng dữ liệu ngày càng lớn thì chúng ta có 2 tình huống mở rộng như sau.

- Dữ liệu được chứa trên một hoặc nhiều máy, nhưng quá nhiều để xử lý một cách hiệu quả.
- Dữ liệu được sinh ra liên tục theo thời gian và với lượng lớn, việc lưu trữ toàn bộ không thực tế. Đây là tình huống dòng dữ liệu (streaming data).

Hai thử thách với dòng dữ liệu lớn:

1. One-pass constraint: Một số thuật toán cần phải duyệt qua toàn bộ lượng dữ liệu trong một lần. Tuy nhiên, lượng dữ liệu này phải tùy thuộc vào dung lượng lưu trữ có sẵn tại thời điểm đó.
2. Concept drift: data distribution có thể thay đổi liên tục theo thời gian. Phân phối của dữ liệu bán hàng của một công ty ở một tiếng trước chắc chắn sẽ khác so với một tiếng sau đó. Điều này dẫn đến việc kết quả đầu ra sẽ thay đổi.

1.6. Some Application Scenarios

1. **Store Product Placement**
   Dựa vào bộ sản phẩn trước đó của người mua. Người bán muốn biết cách đặt sản phẩm lên kệ như thế nào để tăng khả năng các món hàng được mua cùng nhau sẽ đặt các kệ liền kề nhau
2. **Product Recommendation**
   Dựa vào một số ít mặt hàng đã được chọn của khách hàng, sử dụng thuật toán để đưa ra recommend các mặt hàng tiếp theo
3. **Web Log Anomalies**
   Dựa vào một set các web logs có sẵn. Xác định xem đâu là các đoạn log bất thường
4. **Medical ECG Diagnosis**
   Xem xét một chuỗi thời gian ECG được thu thập từ các bệnh nhân khác nhau. Xác định chuỗi bất thường trong đó

2. Data Preparation

#DataMining/DataPreprocessing
#DataMining/DataPreprocessing/FeatureSelection
#DataMining/DataPreprocessing/FeatureExtraction
#DataMining/DataPreprocessing/DataCleaning

#DataMining/DataPreprocessing/Normalization
#DataMining/DataPreprocessing/Reduction

2.1. Introduction

- Định dạng của dữ liệu thực tế rất đa dạng.
- Trong dữ liệu có thể có nhiều giá trị bị thiếu, không đồng nhất hoặc có lỗi sai.
  --> Có nhiều thử thách khi muốn sử dụng dữ liệu hiệu quả.

Các bước chuẩn bị dữ liệu bao gồm:

1. Feature Extraction and portability (trích xuất đặc trưng và khả năng biến đổi kiểu dũ liệu)
2. Data Cleaning
3. Data Reduction, selection, and transformation

2.2. Feature extraction and portability

2.2.1. Feature Extraction

Trong một số trường hợp, trích xuất đặc trưng có quan hệ mật thiết với khái niệm về khả năng biến đổi kiểu dữ liệu. Với khả năng biến đổi kiểu dữ liệu, các đặc trưng bậc thấp của một kiểu dữ liệu có thể được biến đổi thành các đặc trưng bậc cao hơn của một kiểu dữ liệu khác

1. **Sensor data**: thường được collect dưới dạng khối lượng lớn các tín hiệu low-level. Các tính hiệu low-level này sẽ được chuyển đổi thành higher-level features bằng các thuật toán như wavelet hoặc Fourier transforms
2. **Image data**: Ở raw data, hình ảnh được lưu trữ ở dạng các pixel. Do ảnh là dữ liệu thường có nhiều chiều nên việc trích xuất đặc trưng tùy thuộc vào mức độ khác nhau của dự án
3. **Web logs**: thường biểu diễn ở dạng text string nên khá dễ để convert sang các thuộc tính categorical hay numeric
4. **Network traffic**: trong nhiều ứng dụng phát hiện xâm nhập, các đặc điểm liên quan đến network packets thường được sử dụng để phân tích hành vi, chẳng hạn như số byte transfer, network protocol (giao thức mạng) sử dụng, ...

5. **Document data**: dữ liệu ở dạng raw thường ko có cấu trúc. Chúng chứa nhiều mối quan hệ giữa các thực thể khác nhau. Một số cách tiếp cận là remove stop words, sử dụng bag-of-words, ...

Tùy thuộc vào mức độ và kiểu dữ liệu của ứng dụng mà chọn cách tiếp cận feature extraction khác nhau

2.2.2 Data Type Portability

- Khả năng biến đổi kiểu dữ liệu có vai trò rất quan trọng do dữ liệu thường không thuần nhất mà chứa nhiều kiểu khác nhau.
- Với các dữ liệu không thuần nhất như vậy, chúng ta sẽ có các vấn đề sau.
  - Cần thiết kế thuật toán cho một tổ hợp kiểu dữ liệu bất kì.
  - Khó sử dụng các công cụ xử lý có sẵn.
- Trong một số trường hợp, việc biến đổi kiểu dữ liệu dẫn đến việc mất độ chính xác và tính biểu đạt.

Bảng các thuật toán để biến đổi dữ liệu

Table 2.1: Portability of different data types

Source data type	Destination data type	Methods
Numeric	Categorical	Discretization
Categorical	Numeric	Binarization
Text	Numeric	Latent semantic analysis (LSA)
Time series	Discrete sequence	SAX
Time series	Numeric multidimensional	DWT, DFT
Discrete sequence	Numeric multidimensional	DWT, DFT
Spatial	Numeric multidimensional	2-d DWT
Graphs	Numeric multidimensional	MDS, spectral
Any type	Graphs	Similarity graph (Restricted applicability)

1. Discretization: rời rạc hóa

2. Binarization: nhị phân hóa

3. LSA: *đề cập sau*

4. SAX, DWT (Discrete Wavelet Transform), DFT (Discrete Fourier Transform): link chatgpt

2.3. Data Cleaning

- Xử lý dữ liệu bị thiếu
  - Loại bỏ
  - Sử dụng phương pháp ước lượng hoặc điền khuyết
  - Dùng các thuật toán được thiết kế để hoạt động với dữ liệu bị thiếu
- Xử lý dữ liệu sai
  - Phát hiện sự không đồng nhất
  - Sử dụng kiến thức chuyên môn
  - Các phương pháp tập trung dữ liệu. Ví dụ: dùng các đặc tính thống kê của dữ liệu để lọc ngoại lai
- Scale và chuẩn hóa dữ liệu
  - Standardization
  - Min-max scaling
  - ...

2.4. Data Reduction and Transformation

2.4.1. Data Sampling

Lợi thế của lấy mẫu dữ liệu là sự đơn giản, trực quan và dễ thực hiện

Có 2 cách thức lấy mẫu dữ liệu

1. **Lấy mẫu cho dữ liệu tĩnh (static data)**

   Với lấy mẫu cho dữ liệu tĩnh. Khi toàn bộ dữ liệu đã có sẵn và từ đó số điểm dữ liệu gốc được biết thì việc lấy mẫu đơn giản hơn.

   Với cách lấy mẫu không chệch (unbiased sampling)
   một tỉ lệ dữ liệu được định trước và giữ nguyên trong suốt quá trình phân tích. Có hai cách như sau:

   - **Lấy mẫu không hoàn lại (without replacement):** Các bản ghi `n * f` được chọn ngẫu nhiên từ tập dữ liệu *D* chứa các bản ghi *n*, trong đó *f* biểu thị tỷ lệ các điểm dữ liệu sẽ được đưa vào mẫu. Phương pháp này ngăn không cho cùng một bản ghi được đưa vào nhiều lần trừ khi tập dữ liệu gốc chứa các bản sao.

- **Lấy mẫu có hoàn lại (with replacement):** Các bản ghi được lấy mẫu tuần tự và độc lập với tập dữ liệu *D* chứa các bản ghi *n* với tổng số `n * f` lần. Phương pháp này cho phép đưa cùng một bản ghi vào mẫu nhiều lần.

Ngoài kiểu unbiased sampling, chúng ta còn một số loại lấy mẫu khác:

- **Lấy mẫu chệch (Biased Sampling)**: Do một số phần dữ liệu quan trọng hơn, được đánh giá cao hơn trong phần lấy mẫu dựa vào tầm quan trọng của chúng trong quá trình phân tích
- **Lấy mẫu phân tầng (Stratified Sampling)**: Dữ liệu trước tiên sẽ chia thành các nhóm, sau đó lấy mẫu một số điểm dữ liệu xác định trước từ mỗi tầng.

2. **Lấy mẫu reservoir cho dòng dữ liệu**

Các dòng dữ liệu thường ko kích thước cố đinh mà liên tục có các điểm dữ liệu mới. Với cách lấy mẫu này, một mẫu với *k* điểm cho trước được duy trì một cách linh động từ dòng dữ liệu.

Do kích thước rất lớn của một dòng dữ liệu, chúng ta cần các bước xử lý để duy trì tập mẫu *k* điểm với mỗi điểm dữ liệu mới từ dòng.

Với mỗi điểm dữ liệu mới, chúng ta có 2 quyết định sau:

- Luật lấy mẫu nào để quyết định xem điểm dữ liệu mới có được cho vào mẫu hay không
- Luật nào để quyết định xem một điểm dữ liệu cũ trong mẫu có bị bỏ ra để có chỗ cho điểm dữ liệu mới

Với một reservoir kích thước *k* điểm dữ liệu, chúng ta sẽ lấy *k* điểm đầu tiên trong dòng dữ liệu để khởi tạo reservoir.

Sau đó, với điểm dữ liệu thứ *n* từ dòng, chúng ta có 2 quyết định điều khiển sau:

- Cho điểm thử n vào reservoir với xác xuất *k/n*.
- Nếu điểm dữ liệu mới được cho vào thì loại bỏ một trong *k* điểm dữ liệu cũ một cách ngẫu nhiên.

2.4.2. Feature Selection

Một cách khác để rút gọn dữ liệu là loại bỏ đi các đặc trưng không quan trọng.
Chỉ lựa chọn các subset of features từ dữ liệu để sử dụng. Cách chọn subset tùy theo riêng từng application

Có 2 phương pháp chính trong việc feature selection:

1. Unsupervised feature selection: *đề cập ở chương Data Clustering*
2. Supervised //: *đề cập ở chương Data Classification*

2.4.3. Data reduction with axis rotation (Giảm chiều bằng phép xoay trục)

Trong dataset thực tế thường tồn tại các tương quan giữa các feature khác nhau và chúng thường không chặt chẽ và xác định một cách thủ công.

Từ các ràng buộc và tương quan trên, một số thông tin từ một chiều có thể dùng để dự đoán thong tin của các chiều khác.

1. **PCA (Principal Component Analysis)**: Mục tiêu của PCA là xoay dữ liệu về một hệ trục sao cho lượng phương sai lớn nhất có thể được biểu diễn bởi một số chiều nhỏ nhất. Phương sai của một tập dữ liệu theo một hướng cụ thể có thể được thể hiện thông qua ma trận phương sai của dữ liệu. Có thể chứng minh được ma trận phương sai là đối xứng, nửa xác định dương. Từ đó, ma trận này chéo hóa được và các trị riêng của ma trận phương sai biểu diễn phương sai của dữ liệu dọc theo vectors riêng tương ứng. Do đó các vectors riêng với trị riêng lớn sẽ thể hiện phương sai lớn hơn và được gọi là các principal component, các trục chính mới chúng ta dùng để biểu diễn dữ liệu.
2. **SVD (Singular value decomposition)**: SVD có quan hệ gần với PCA. SVD có 2 bộ vector cơ sở thay vì 1 như PCA. SVD cho cùng vector cơ sở với PCA nêu các thuộc tính của dữ liệu có trung bình là 0
3. **LSA (Latent Semantic Analysis)**: là một ứng dụng của SVD với dữ liệu văn bản. Với dữ liệu văn bản, mỗi dòng của ma trận dữ liệu ứng với mỗi văn bản trong dữ liệu và chứa tần suất xuất hiện của mỗi từ của văn bản đó. Do đây là ma trận thưa nên trung bình của mỗi cột rất gần 0, điều này dẫn đến kết quả khá gần với PCA, mặc

dù không sử dụng mean centering. Tính thưa của ma trận cũng dẫn đến số chiều nội tại thấp, điều này cũng dẫn đến việc giảm số chiều bằng LSA có thể rất mạnh.

Ngoài giảm chiều dữ liệu và nén dữ liệu thì PCA và SVD còn các ứng dụng khác như: Khử nhiễu, điền khuyết, giải hệ tuyến tính, nghịch đảo ma trận, ...

Ví dụ và cách tính chi tiết của PCA, SVD, LSA, tham khảo link gpt

2.4.3. Data reduction with type transformation

Với các phương pháp này thì việc rút giảm dữ liệu đi kèm với biến đổi kiểu dữ liệu. Thông thường thì dữ liệu sẽ được biến đổi từ một kiểu phức tạp về một kiểu ít phức tạp hơn.

Ta sẽ tìm hiểu hai phương pháp là

1. Time-series to multidimensional using Haar Wavelet Transform:
   Chúng ta có thể dùng kĩ thuật wavelet để khai triển một time series thành các vector cơ sở wavelet có trọng số. Mỗi trọng số này thể hiện độ biến thiên của time series giữa 2 nửa của một khoảng thời gian. Trong ứng dụng giảm số chiều, các hệ số lớn (sau khi chuẩn hóa) sẽ được giữ lại.
2. Weighted Graphs to multidimensional using multidimensional scaling and Spectral methods
   Xem thêm tại link chatgpt

3. Similarity and Distances

#DataMining/Similarity
#DataMining/Distances

3.1. Introduction

Trong nhiều ứng dụng khai phá dữ liệu, chúng ta cần xác định các đối tượng dữ liệu tương đồng, hoặc không tương đồng.

Việc lựa chọn hàm tương đồng (hoặc hàm khoảng cách, tùy thuộc ứng dụng) rất quan trọng trong thiết kế của các thuật toán.

3.2. Quantitative Data

#DataMining/Distances/Euclidean
#DataMining/Distances/Manhattan
#DataMining/Distances/LpNorm

Công thức khoảng cách phổ biến nhất với dữ liệu định lượng là $L_p - Norm$

$$Dist(\overline{X}, \overline{Y}) = \left(\sum_{i=1}^{d} |x_i - y_i|^p \right)^{1/p}$$

2 giá trị *p* hay dùng nhất là *p = 1* (Euclidean) và *p = 2* (Manhattan)

3.2.1. Impact of Domain-Specific Relevance

#DataMining/Distances/Minkowski

Do ảnh hướng về tầm quan trọng của feature này so với feature kia. Công thức $L_p - Norm$ sẽ được thêm trọng số phía trước và trở thành công thức *Minkowski*

$$Dist(\overline{X}, \overline{Y}) = \left(\sum_{i=1}^{d} a_i . |x_i - y_i|^p \right)^{1/p}$$

3.2.2. Impact of High Dimensionality

Rất nhiều ứng dụng khai phá dữ liệu bị mất tính hiệu quả khi số chiều của dữ liệu tăng cao. Hiện tượng này được gọi là “curse of dimensionality”.

3.2.3. Impact of Locally Irrelevant Features

...

3.2.4. Impact of Different $L_p - Norm$

Với các *p* lớn thì ảnh hưởng của các thuộc tính không quan trọng thường được nhấn mạnh.

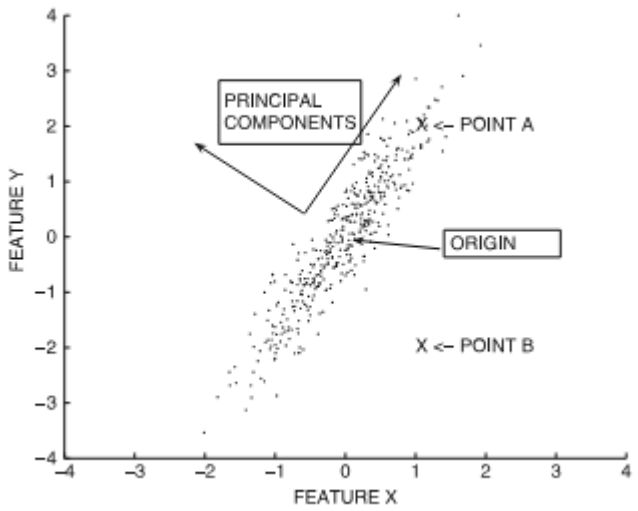
3.2.5. Match-Based Similarity Computation

...

3.2.6. Impact of Data Distribution

#DataMining/Distances/Mahalanobis

Trong nhiều ứng dụng, việc tính khoảng cách còn phụ thuộc vào phân phối của dữ liệu.

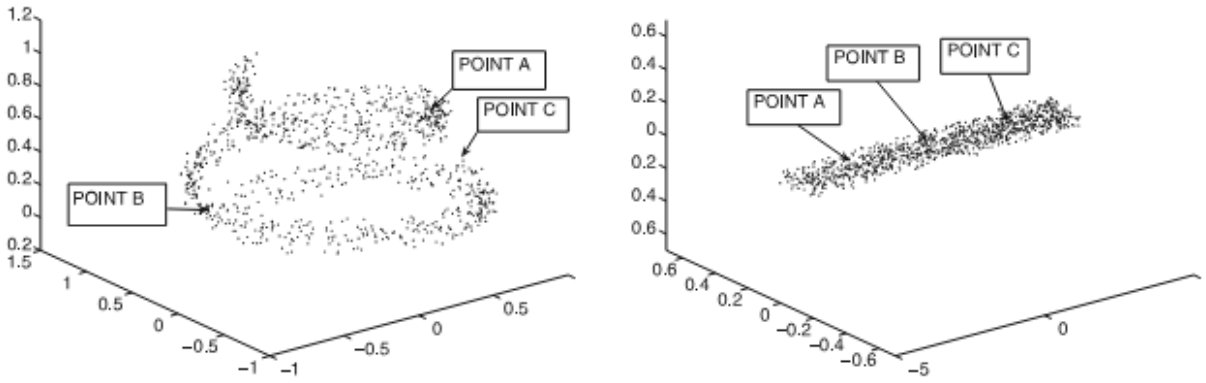


Như trong hình thì đường nối tâm O đến điểm A nằm theo hướng có phương sai cao, còn đường nối tâm O đến điểm B thì có dữ liệu thưa và nằm theo hướng có phương sai thấp. Theo cách nhìn này, có thể đánh giá đoạn OB dài hơn OA. Khoảng cách Mahalanobis cũng được dựa trên nguyên lý này.

$$\mathrm{Maha}(\overline{X}, \overline{Y}) = \sqrt{(\overline{X} - \overline{Y})\Sigma^{-1}(\overline{X} - \overline{Y})^{T}}$$

3.2.7 Nonlinear Distributions: ISOMAP

#DataMining/Distances/ISOMAP



(a) A and C seem close (original data)

(b) A and C are actually far away (*ISOMAP* embedding)

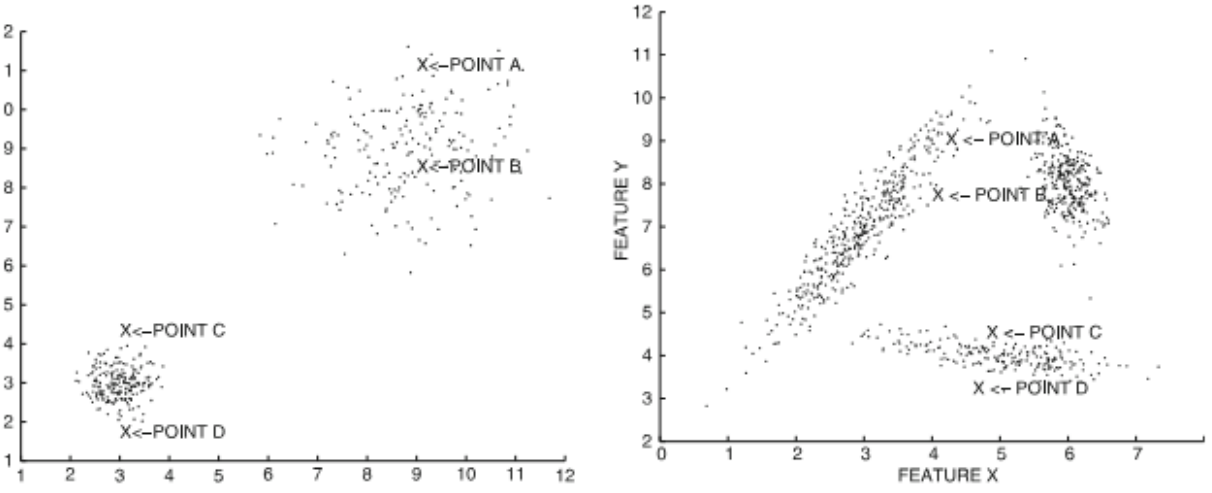
Figure 3.5: Impact of *ISOMAP* embedding on distances

Ảnh hưởng của ISOMAP đã giúp ta thấy được khoảng cách của điểm A và C là xa nhất, so với dùng khoảng cách truyền thống thì A, C là khoảng cách gần nhất

Cách tính gồm 2 bước:

1. Tính *k*-nearest neighbors của từng điểm. Xây dựng đồ thị trọng số *G* với từng node biểu thị cho các điểm dữ liệu và cạnh biểu thị khoảng cách của *k*-nearest neighbors này.
2. Đối với bất kỳ cặp điểm X, Y nào, Dist(X,Y) là đường đi ngắn nhất giữa các nút tương ứng trong đồ thị *G*

3.2.8. Impact of Local Data Distribution



(a) local density variation

(b) local orientation variation

Figure 3.6: Impact of local distributions on distance computations

Phân phối dữ liệu có thể thay đổi đáng kể theo từng cục bộ, dẫn tới việc tính toán có thể thay đổi

Như hình vẽ 3.6a, khoảng cách giữa (A,B) và (C,D) được cho là bằng nhau (theo Euclidean) nhưng do có sự khác biệt về mật độ phân phối nên (C,D) nên được cho là lớn hơn (A,B). Điều này chứng tỏ là (C,D) nên được cho là ở xa hơn trong bối cảnh địa phương của chúng. Vấn đề này thường gặp trong các phương pháp dựa trên khoảng cách như phát hiện điểm ngoại lai và một trong những pp nổi tiếng đó là LOF (Local Outlier Factor)

Đối với hình 3.6b, khoảng cách giữa (A, B) và (C, D) là giống nhau khi sử dụng metric Euclidean. Tuy nhiên, các cụm địa phương trong mỗi vùng có định hướng rất khác nhau. Trục có phương sai cao của cụm dữ liệu liên quan đến (A, B) thẳng hàng với đường từ A đến B, nhưng điều này không đúng với (C, D). Do đó, khoảng cách nội tại giữa C và D lớn hơn so với A và B. Ví dụ, nếu khoảng cách Mahalanobis địa phương được tính toán sử dụng thống kê hiệp phương sai cụm liên quan, thì khoảng cách giữa C và D sẽ lớn hơn khoảng cách giữa A và B.

3.3. Categorical Data

Một cách đơn giản để tính khoảng cách giữa các dữ liệu định tính là sử dụng kĩ thuật nhị phân hóa (one hot encoding)

Với dữ liệu định tính thì chúng ta thường làm việc với sự tương đồng hơn khoảng cách.

Với bản ghi X và Y. Sự tương đồng đơn giản nhất giữa 2 bản ghi này là

$$\mathrm{Sim}(X, Y) = \sum_{i=1}^{d} S(x_i, y_i)$$

trong đó, $S(x_i, y_i)$ là hàm similarity

Hàm similarity đơn giản nhất là set $S(x_i, y_i)$ bằng 1 nếu $x_i = y_i$ hoặc bằng 0 nếu ngược lại

Với trường hợp là categorical data, aggregate statistical properties (tính chất thống kê gộp chung) của tập dữ liệu nên được sử dụng khi tính similarity

Ý tưởng quan trọng ở đây là các cặp giống nhau với một giá trị hiếm nên có trọng số cao hơn so với các giá trị phổ biến. Từ đó, chúng ta có khái niệm "*Tần suất xuất hiện ngược*" (inverse occurence frequency)

$$S(x_i, y_i) = \begin{cases} 1/p_k(x_i)^2 & \text{if } x_i = y_i \\ 0 & \text{otherwise} \end{cases}$$

Với $p_k(x)$ là tỉ lệ records tại attribute thứ *k* có giá trị *x* trong tập dữ liệu. Nói cách khác, $p_k(x)$ là tần suất xuất hiện của giá trị *x* cho thuộc tính thứ *k*

Một công thức khác cũng có tư tưởng giống với công thức trên là công thức *Goodall*

$$S(x_i, y_i) = \begin{cases} 1 - p_k(x_i)^2 & \text{if } x_i = y_i \\ 0 & \text{otherwise} \end{cases}$$

3.4. Mixed Quantitative and Categorical Data

Với các dữ liệu trộn định tính và định lượng thì ta dùng các trọng số lần lượt cho các thành phần

Cụ thể với $X = (X_n, X_c), Y = (Y_n, Y_c)$
Chúng ta có thể tính độ tương đồng giữa hai bản ghi này với công thức

$$Sim(X, Y) = \lambda \cdot NumSim(X_n, Y_n) + (1 - \lambda) \cdot CatSim(X_c, Y_c)$$

Bên cạnh đó, để có thể so sánh các giá trị tương đồng của các thuộc tính định lượng và các thuộc tính định tính thì chúng ta cần chuẩn hóa. Một cách để chuẩn hóa là sử dụng độ lệch chuẩn của các giá trị tường đồng trên 2 miền thành phần.

$$Sim(X, Y) = \lambda \cdot NumSim(X_n, Y_n)/\sigma_n + (1 - \lambda) \cdot CatSim(X_c, Y_c)/\sigma_c$$

3.5. Text Document Data

Văn bản có thể được xem là dữ liệu định lương đa chiều với tần suất xuất hiện của mỗi từ là một thuộc tính.

Trường hợp này thì các khoảng cách sử dụng chuẩn Lp không thích hợp với các văn bản có độ dài khác nhau.

Một cách để tránh trở ngại với độ dài văn bản là dùng độ đo cosine để tính góc giữa các văn bản.

Với 2 bản ghi $X = (x_1, \dots, x_d), Y = (y_1, \dots, y_d)$, ta có

$$\text{cosine}(X, Y) = \frac{\sum x_i \cdot y_i}{\sqrt{\sum x_i^2} \cdot \sqrt{\sum y_i^2}}$$

Xét đến tấn suất xuất hiện ngược của document data, nếu hai đoạn document match với nhau những uncommon word, thì IDF (*inverse document frequency*) id_i sẽ lớn

$$\text{idf}_i = \log(n / n_i)$$

Đây là một hàm nghịch biến với n_i là số lượng document chứa từ thứ *i*

Normalized Frequency (tần suất chuẩn hóa) $h(x_i)$ cho word thứ *i* bằng $h(x_i) = f(x_i) \cdot \text{idf}_i$ với $f(x_i)$ là damping function (hàm giảm tần suất), thường bằng $\sqrt{x_i}$, $\log(x_i)$, hoặc x_i

Từ đó suy ra công thức cosine sau khi sử dụng normalized frequency là

$$\text{cosine}(X, Y) = \frac{\sum h(x_i) \cdot h(y_i)}{\sqrt{\sum h(x_i)^2} \cdot \sqrt{\sum h(y_i)^2}}$$

Phương pháp này đảm bảo rằng các từ ít gặp và có ý nghĩa hơn sẽ có trọng số cao hơn trong việc tính toán độ tương đồng giữa các tài liệu, cải thiện độ chính xác của phép đo cosine ban đầu.

Một công thức có mục tiêu tương tụ nhưng ít phổ biến hơn là *Jaccard coefficient* , thường được sử dụng cho sparse binary data sets

$$J(\overline{X}, \overline{Y}) = \frac{\sum_{i=1}^{d} h(x_i) \cdot h(y_i)}{\sum_{i=1}^{d} h(x_i)^2 + \sum_{i=1}^{d} h(y_i)^2 - \sum_{i=1}^{d} h(x_i) \cdot h(y_i)}.$$

3.6. Binary and Set Data

Dữ liệu nhị phân là một dạng đặc biệt của dữ liệu dạng tập hợp, trong đó giá trị 1 biểu thị sự có mặt của một phần tử trong tập hợp, còn giá trị 0 biểu thị sự vắng mặt của phần tử đó

Dữ liệu nhị phân thường gặp trong các lĩnh vực như giỏ hàng (marketbasket), nơi mà các giao dịch chứa thông tin về việc một mặt hàng có xuất hiện trong giao dịch hay không

Dữ liệu này có thể được coi là một trường hợp đặc biệt của dữ liệu văn bản

$$J(X, Y) = \frac{\sum_{i=1}^{d} x_i \cdot y_i}{\sum_{i=1}^{d} x_i^2 + \sum_{i=1}^{d} y_i^2 - \sum_{i=1}^{d} x_i \cdot y_i}$$

Điều này tương đương với công thức:

$$J(X, Y) = \frac{|S_X \cap S_Y|}{|S_X \cup S_Y|}$$

Công thức này chính là chỉ số Jaccard (Jaccard Index), một phép đo phổ biến cho độ tương đồng giữa hai tập hợp.

3.7. Time-Series Similarity Measures

Cách biểu diễn theo dãy rời rạc có thể xem là dạng rời rạc của cách biểu diễn theo chuỗi thời gian liên tục.

Việc thiết kế độ đo tương đồng cho time-series phụ thuộc vào từng ứng dụng. Một số yếu tố ảnh hưởng như sau:

1. Scaling và Translation của Thuộc tính Hành Vi (Behavioral attribute scaling and translation)
2. Translation Thuộc Tính Ngữ Cảnh Theo Thời Gian (Temporal (contextual) attribute translation)
3. Scaling Thuộc Tính Ngữ Cảnh Theo Thời Gian (Temporal (contextual) attribute scaling)
4. Không Liên Tục Trong Việc So tương đồng (Noncontiguity in matching)

3.7.1. Impact of Behavioral Attribute Normalization

1. **Dịch Chuyển Thuộc Tính Hành Vi (Behavioral Attribute Translation)**: Thuộc tính hành vi được chuẩn hóa trung bình trong quá trình tiền xử lý. Điều này có nghĩa là giá trị trung bình của thuộc tính sẽ được dịch chuyển về 0.

2. **Biến Đổi Tỷ Lệ Thuộc Tính Hành Vi (Behavioral Attribute Scaling)**: Độ lệch chuẩn của thuộc tính hành vi được chuẩn hóa về 1 đơn vị. Điều này có nghĩa là các giá trị thuộc tính sẽ được chia cho độ lệch chuẩn của chúng để có được một thang đo thống nhất.

Lưu ý rằng không phải lúc nào các vấn đề chuẩn hóa này cũng liên quan đến mọi ứng dụng. Một số ứng dụng có thể chỉ cần dịch chuyển, chỉ cần biến đổi tỷ lệ, hoặc không cần cả hai. Các ứng dụng khác có thể cần cả hai bước chuẩn hóa. Thực tế, việc chọn sai phương pháp chuẩn hóa có thể gây hại cho khả năng diễn giải kết quả. Do đó, người phân tích cần cẩn thận lựa chọn phương pháp chuẩn hóa phù hợp dựa trên nhu cầu cụ thể của ứng dụng.

3.7.2. $\mathrm{L_p - Norm}$

Phép đo này coi một chuỗi thời gian như một điểm dữ liệu đa chiều, trong đó mỗi dấu thời gian là một chiều.

Các điểm chính trong đoạn văn bao gồm:

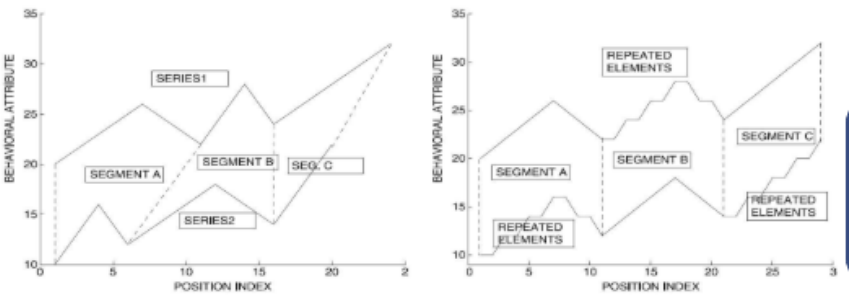
1. **Áp dụng chuẩn $\mathrm{L_p}$ cho chuỗi thời gian**:
   - Chuẩn $\mathrm{L_p}$ có thể áp dụng cho các chuỗi thời gian bằng cách coi mỗi dấu thời gian là một chiều dữ liệu.
2. **Áp dụng chuẩn $\mathrm{L_p}$ cho biến đổi wavelet của chuỗi thời gian**:
   - Chuẩn $\mathrm{L_p}$ cũng có thể áp dụng cho các biến đổi wavelet của chuỗi thời gian.
   - Khi $\mathrm{p = 2}$ và nếu giữ lại hầu hết các hệ số wavelet lớn, việc tính toán khoảng cách sẽ chính xác với đại diện wavelet.
   - Nếu không loại bỏ bất kỳ hệ số wavelet nào, thì khoảng cách giữa hai đại diện sẽ giống nhau. Điều này là do biến đổi wavelet có thể được coi là một sự xoay trục hệ tọa độ, trong đó mỗi chiều đại diện cho một dấu thời gian. Các phép đo Euclidean không thay đổi theo sự xoay trục.
3. **Hạn chế của chuẩn $\mathrm{L_p}$**:
   - Chuẩn $\mathrm{L_p}$ được thiết kế cho các chuỗi thời gian có chiều dài bằng nhau.
   - Nó không thể giải quyết các biến dạng về thuộc tính ngữ cảnh (temporal contextual attributes), chẳng hạn như sự co giãn hoặc dịch chuyển trong trục thời gian.

Chuẩn L_p là một công cụ hữu ích để đo khoảng cách giữa các chuỗi thời gian có chiều dài bằng nhau, đặc biệt khi áp dụng với biến đổi wavelet. Tuy nhiên, nó gặp khó khăn khi phải xử lý các biến dạng ngữ cảnh theo thời gian.

3.7.3. Dynamic Time Warping Distance

Khoảng cách biến đổi thời gian động (Dynamic Time Warping - DTW)

- DTW kéo dãn chuỗi thời gian dọc theo chiều thời gian một cách linh động tại mỗi vị trí.
- Như trong hình sau thì đoạn A, B, C có hình dạng giống nhau, nhưng mỗi đoạn cần được kéo dãn để so khớp với nhau.



(a) Original series (b) Warped series

Figure 3.8: Illustration of dynamic time warping by repeating elements

1. **Định Nghĩa và Ứng Dụng của DTW**:
   - DTW kéo dài chuỗi dọc theo trục thời gian một cách động (dynamic) để tạo ra sự khớp tốt hơn giữa các chuỗi. Phương pháp này xuất phát từ lĩnh vực nhận dạng giọng nói để điều chỉnh các tốc độ nói khác nhau.
   - DTW có thể được sử dụng cho cả dữ liệu chuỗi thời gian và chuỗi tuần tự, vì nó chỉ giải quyết vấn đề biến đổi thuộc tính ngữ cảnh (contextual attribute scaling) và không liên quan đến bản chất của thuộc tính hành vi (behavioral attribute).
2. **Ưu Điểm của DTW so với Lp-norm**:
   - Trong khi Lp-norm chỉ có thể áp dụng cho các chuỗi có độ dài bằng nhau, DTW cho phép đo khoảng cách giữa hai chuỗi có độ dài khác nhau.
   - Lp-norm sử dụng ánh xạ một-một giữa các dấu thời gian của hai chuỗi, còn DTW cho phép ánh xạ nhiều-một để giải quyết việc biến đổi thời gian.
3. **Quy Trình Tính Toán DTW**:
   - DTW cho phép lặp lại một số phần tử trong các đoạn được chọn một cách cẩn thận của một trong hai chuỗi để tạo ra hai chuỗi có độ dài bằng nhau, tạo ra ánh xạ một-một giữa chúng.

- Khoảng cách có thể được đo trên các chuỗi đã được biến đổi sử dụng bất kỳ phương pháp đo khoảng cách nào như Lp-norm.

4. **Công Thức Tính Toán DTW**:
   - $\mathrm{DTW}(i,j)$ được xác định tối ưu bằng cách sử dụng lập trình động:

$$\mathrm{DTW}(i,j) = \mathrm{distance}(x_i, y_j) + \min \begin{cases} \mathrm{DTW}(i, j-1) & \text{repeat } x_i \\ \mathrm{DTW}(i-1, j) & \text{repeat } y_j \\ \mathrm{DTW}(i-1, j-1) & \text{repeat neither} \end{cases}$$

5. **Triển Khai Thuật Toán**:
   - Thuật toán được khởi tạo với $\mathrm{DTW}(0,0) = 0, \mathrm{DTW}(0,j) = \infty$ cho $j \in \{1 \ldots n\}$, và $\mathrm{DTW}(i,0) = \infty$ cho $i \in \{1 \ldots m\}$.
   - Sau đó, tính toán giá trị $\mathrm{DTW}(i,j)$ bằng cách thực hiện lặp đi lặp lại công thức trên với các giá trị chỉ số i và j tăng dần từ 1 đến m và từ 1 đến n.
6. **Tối Ưu Hóa và Ràng Buộc Thực Tiễn**:
   - Đường đi tối ưu có thể được hiểu như một đường đi tối ưu qua các giá trị khác nhau của i và j trong lưới $m \times n$.
   - Một ràng buộc thông thường là ràng buộc cửa sổ, yêu cầu rằng DTW(i, j) chỉ được tính khi $|i - j| \leq w$. Điều này giúp giảm bớt số lượng giá trị cần tính toán trong quá trình đệ quy lập trình động.
7. **Mở Rộng DTW cho Nhiều Thuộc Tính Hành Vi**:
   - Khoảng cách DTW có thể dễ dàng mở rộng cho nhiều thuộc tính hành vi nếu giả định rằng các thuộc tính hành vi khác nhau có cùng sự biến dạng thời gian. Trong trường hợp này, công thức đệ quy không thay đổi, chỉ khác ở chỗ khoảng cách $\mathrm{distance}(x_i, y_j)$ được tính bằng phương pháp đo khoảng cách dựa trên vector.

DTW là một công cụ mạnh mẽ để đo khoảng cách giữa các chuỗi thời gian, đặc biệt là khi các chuỗi có độ dài khác nhau và cần xử lý các biến dạng ngữ cảnh.

3.7.4. Window-Based Methods

Ý tưởng ở đây là nếu hai chuỗi có nhiều đoạn liên tiếp giống nhau, chúng nên được coi là tương tự. Đối với các chuỗi thời gian dài, việc so

khớp toàn cầu trở nên khó khăn. Do đó, lựa chọn hợp lý là sử dụng các cửa sổ để đo lường độ tương đồng theo từng đoạn.

$$\mathrm{Sim}(X, Y) = \sum_{i=1}^{r} \mathrm{Match}(Xi, Yi)$$

Các phương pháp dựa trên cửa sổ nhằm giải quyết vấn đề dữ liệu bị mất bằng cách phân chia chuỗi thời gian thành các cửa sổ không chồng lấp và đo lường độ tương đồng của từng cặp cửa sổ.

Độ tương đồng tổng thể giữa hai chuỗi được tính bằng tổng các giá trị độ tương đồng của từng cặp cửa sổ.

Việc xác định giá trị $\mathrm{Match}(Xi, Yi)$ và phân chia chuỗi thành các cửa sổ tối ưu là những thách thức lớn trong phương pháp này.

3.8. Discrete Sequence Similarity Measures

Các độ đo tương đồng dãy rời rạc dựa trên cùng nguyên lý với các độ đo chuỗi thời gian.

Khi có ánh xạ song ánh giữa 2 dãy rời rạc tại các vị trí, nhiều độ đo khoảng cách định tính cho dữ liệu đa chiều có thể được áp dụng.

Tuy nhiên, trong các ứng dụng thực tế với dãy rời rạc thì các song ánh như vậy thường không tồn tại.

3.8.1. Edit Distance

Khoảng cách chỉnh sửa (còn gọi là khoảng cách Levenshtein) đo lường khoảng cách giữa hai chuỗi bằng cách tính chi phí tối thiểu để chuyển đổi một chuỗi thành chuỗi khác thông qua một loạt các phép biến đổi (chèn, xóa, thay thế).

Thay thế thường có chi phí cao hơn chèn hoặc xóa, trong khi chèn và xóa thường có chi phí bằng nhau.

$$\mathrm{Edit}(i, j) = \min \begin{cases} \mathrm{Edit}(i-1, j) + \text{Deletion Cost} \\ \mathrm{Edit}(i, j-1) + \text{Insertion Cost} \\ \mathrm{Edit}(i-1, j-1) + I_{ij} \cdot \text{Replacement Cost} \end{cases}$$

Ví dụ: Để chuyển đổi chuỗi "abababababab" thành "bababababa", có thể thực hiện 10 phép thay thế hoặc 1 phép xóa và 1 phép chèn.

Phương pháp này sử dụng lập trình động để tìm chi phí tối ưu và có thể mở rộng cho dữ liệu số với các phép biến đổi đặc thù cho chuỗi thời gian.

3.8.2. Longest Common Subsequence (LCSS)

Chuỗi con chung dài nhất

$$LCSS(i,j) = \max \begin{cases} LCSS(i-1,j-1)+1 & \text{only if } x_i = y_i \\ LCSS(i-1,j) & \text{otherwise (no match on } x_i) \\ LCSS(i,j-1) & \text{otherwise (no match on } y_i) \end{cases}$$

LCSS đo lường mức độ tương tự giữa hai chuỗi bằng cách tìm chuỗi con dài nhất mà cả hai chuỗi cùng chứa theo thứ tự tương tự. Phương pháp này sử dụng lập trình động để tìm giá trị tối ưu và có thể mở rộng cho chuỗi thời gian liên tục bằng cách phân loại giá trị chuỗi thời gian.

3.9. Graph Similarity Measures

Với các đồ thị thì sự tương đồng có thể được đo với nhiều cách khác nhau, tùy thuộc được đo giữa 2 đồ thị hay 2 node trong 1 đồ thị.

Các đồ thị ở đây được giả sử là vô hướng

3.9.1. Similarity between Two Nodes in a Single Graph

Trong một số lĩnh vực, như mạng thư tịch (bibliographic networks) thì các
cạnh được gắn trọng số (weights) và hàm tương đồng được dùng.

Trong một số lĩnh vực khác, như mạng giao thông thì các cạnh được gắn chi phí (costs) và hàm khoảng cách được dùng.

Thông thường thì việc chuyển đổi giữa hàm tương đồng và khoảng cách có thể thực hiện bằng các hàm kernel. Ví dụ: Kernel nhiệt $K(x) = e^{-x^2/t^2}$

Tiêu chí Đo Lường Độ Tương Tự

- Đường Ngắn Nhất: Nút được kết nối qua các đường ngắn nên được coi là tương tự nhau hơn.
- Độ Kết Nối: Nút được kết nối qua nhiều đường nên được coi là tương tự hơn.

Tóm tắt: Độ tương tự giữa các nút trong một mạng không hướng có thể được đo lường dựa trên nguyên tắc đồng hình, với các tiêu chí như khoảng cách đường ngắn nhất và số lượng đường kết nối giữa các nút. Việc sử dụng các hàm kernel heuristic cho phép chuyển đổi giữa chi phí và trọng số, giúp phù hợp với các ứng dụng cụ thể.

3.9.1.1. Biện Pháp Đo Khoảng Cách Cấu Trúc (Structural Distance-Based Measure)

Mục Tiêu: Đo khoảng cách từ một nút nguồn *s* đến bất kỳ nút nào khác trong mạng.

Thuật toán sử dụng là thuật toán Dijkstra, đặc trưng là kiểm tra mỗi nút và cạnh liên kết đúng một lần

Đặc Điểm: Biện pháp này chỉ tập trung vào khoảng cách cấu trúc mà không tính đến số lượng đường đi giữa hai nút.

3.9.1.2. Tương Tự Dựa trên Bước Ngẫu Nhiên (Random Walk-Based Similarity)

Hạn Chế của Đo Khoảng Cách Cấu Trúc: Không hiệu quả khi các cặp nút có số lượng đường đi giữa chúng khác nhau.

Nguyên Lý Bước Ngẫu Nhiên: - **Bước Ngẫu Nhiên Khởi Điểm từ *s***: Bắt đầu từ nút nguồn *s* và chuyển đến nút láng giềng với xác suất tỉ lệ với trọng số w_{ij}. - **Xác Suất Khởi Động Lại**: Tại bất kỳ nút nào, có thể quay lại nút nguồn *s* với một xác suất gọi là xác suất khởi động lại. - **Phân Phối Xác Suất**: Kết quả là một phân phối xác suất thiên về nút nguồn *s*, các nút giống *s* sẽ có xác suất được ghé thăm cao hơn. - **Trực Giác**: Nếu bạn bị lạc trong một mạng đường và di chuyển ngẫu nhiên, bạn sẽ có khả năng cao đến được vị trí gần và có nhiều đường dẫn đến đó.

3.9.2. Similarity Between Two Graphs

Nhiều biện pháp, như khoảng cách chỉnh sửa đồ thị và sự tương tự dựa trên cấu trúc con, đã được đề xuất để giải quyết trường hợp khó khăn này. Ý tưởng cốt lõi trong mỗi phương pháp này là:

1. **Khoảng cách đồ thị con chung lớn nhất (Maximum common subgraph distance)**: Khi hai đồ thị chứa một đồ thị con lớn chung, chúng thường được coi là tương tự hơn
2. **Sự tương tự dựa trên cấu trúc con (Substructure-based similarity)**: Mặc dù rất khó để khớp hai đồ thị lớn, nhưng dễ dàng hơn nhiều để khớp các cấu trúc con nhỏ hơn. Ý tưởng cốt lõi là đếm các cấu trúc con thường xuyên xuất hiện giữa hai đồ thị và báo cáo nó như một biện pháp đo lường sự tương tự
3. **Khoảng cách chỉnh sửa đồ thị (Graph-edit distance)**: Tương tự edit distance chuỗi đã được định nghĩa
4. **Graph kernels**: Kernel đường đi ngắn nhất và Kernel bước ngẫu nhiên

Các phương pháp này khá phức tạp và yêu cầu nền tảng sâu rộng hơn về lĩnh vực đồ thị. Do đó, việc thảo luận về các biện pháp này được hoãn lại cho chương 17 của cuốn sách này.

3.10. Supervised Similarity Functions

Trong các phần trước đã thảo luận về các phương pháp đo sự tương tự không yêu cầu hiểu biết về ý đồ của người dùng. Tuy nhiên, trong thực tế, sự liên quan của một đặc trưng hay sự lựa chọn của hàm khoảng cách phụ thuộc rất nhiều vào lĩnh vực cụ thể

Ví dụ, trong tập dữ liệu hình ảnh, nên trọng số đặc trưng màu sắc hay đặc trưng kết cấu nhiều hơn?

Phản hồi từ người dùng có thể được biểu diễn dưới dạng các cặp đối tượng:

$$S = (O_i, O_j) : O_i \text{is similar to } O_j$$

$$D = (O_i, O_j) : O_i \text{is dissimilar to } O_j$$

4. Association Pattern Mining

#DataMining/AssociationPatternMining

4.1. Introduction

Bài toán khai phá mẫu liên hệ cổ điển được định nghĩa với dữ liệu siêu thị
chứa các tập hạng mục mà khách hàng mua (được gọi là giao dịch)

Mục tiêu của bài toán là xác định các liên hệ giữa các nhóm hạng mục được
mua bởi khách hàng.

4.2. The Frequent Pattern Mining Model

Giả định CSDL *T* chứa *n* giao dịch $\mathrm{T}_1, \mathrm{T}_2, \ldots, \mathrm{T}_\mathrm{n}$

Mỗi giao dịch T_i được biểu diễn bằng một bản ghi đa chiều với độ dài $\mathrm{d} = |\mathrm{U}|$, trong đó U là tập hợp các mục (items).

Định nghĩa **Itemset**: Itemset là một tập hợp các mục. K-itemset là itemset chứa đúng k mục.

Định nghĩa **Support**: Tỷ lệ giao dịch trong $\mathrm{T}_1 \ldots \mathrm{T}_\mathrm{n}$ mà một itemset xuất hiện như một tập con cung cấp một định lượng rõ ràng về tần suất của nó. Tần suất này còn được gọi là độ hỗ trợ (support).

Các mục có mối quan hệ thường xuất hiện cùng nhau trong các giao dịch, và các itemset như vậy sẽ có độ hỗ trợ cao.

Định nghĩa **Frequent Itemset Mining**: Xác định tất cả các itemset I xuất hiện ít nhất trong một tỷ lệ giao dịch tối thiểu minsup trong T.

Property (Support Monotonicity Property) : Support của mọi subset *J* thuộc *I* luôn lớn hơn bằng support của itemset *I*

$$\mathrm{sup}(\mathrm{J}) \geq \mathrm{sup}(\mathrm{I}) \quad \forall \mathrm{J} \subset \mathrm{I}$$

4.3. Association Rule Generation Framework

Định nghĩa **Confidence**: Confidence của luật $\mathrm{X} \Rightarrow \mathrm{Y}$ là xác suất có điều kiện của một transaction chứa tập Y với điều kiện đã chứa tập X trước đó và công thức là:

$$\text{conf}(X \Rightarrow Y) = \frac{\text{sup}(X \cup Y)}{\text{sup}(X)}$$

Định nghĩa **Association Rules**: Với X, Y là hai tập items được coi là association rule có minimum support là minsup và minimum confidence là minconf nếu thỏa hai điều kiện

1. Support của itemset $X \cup Y$ lớn hơn hoặc bằng minsup
2. Confidence của luật $X \Rightarrow Y$ lớn hơn hoặc bằng minconf

Cách xây dựng:

- **Giai đoạn 1**: Tìm tất cả các itemset thường xuyên, tức là các itemset có độ hỗ trợ ít nhất là minsup.
- **Giai đoạn 2**:
  - Giả sử có một tập hợp các itemset thường xuyên F.
  - Với mỗi itemset I trong F, phân chia I thành tất cả các kết hợp có thể của các tập X và Y = I − X, sao cho I = X ∪ Y.
  - Tính toán độ tin cậy của mỗi luật X ⇒ Y và giữ lại nếu nó thỏa mãn yêu cầu về độ tin cậy tối thiểu.

Property **(Confidence Monotonicity)**: X_1, X_2, I là các itemsets thỏa $X_1 \subset X_2 \subset I$. Suy ra:

$$\text{conf}(X_2 \Rightarrow I - X_2) \geq \text{conf}(X_1 \Rightarrow I - X_1)$$

4.4. Frequent Itemset Mining Algorithms

4.4.1. Brute Force Algorithms

#DataMining/AssociationPatternMining/BruteForce

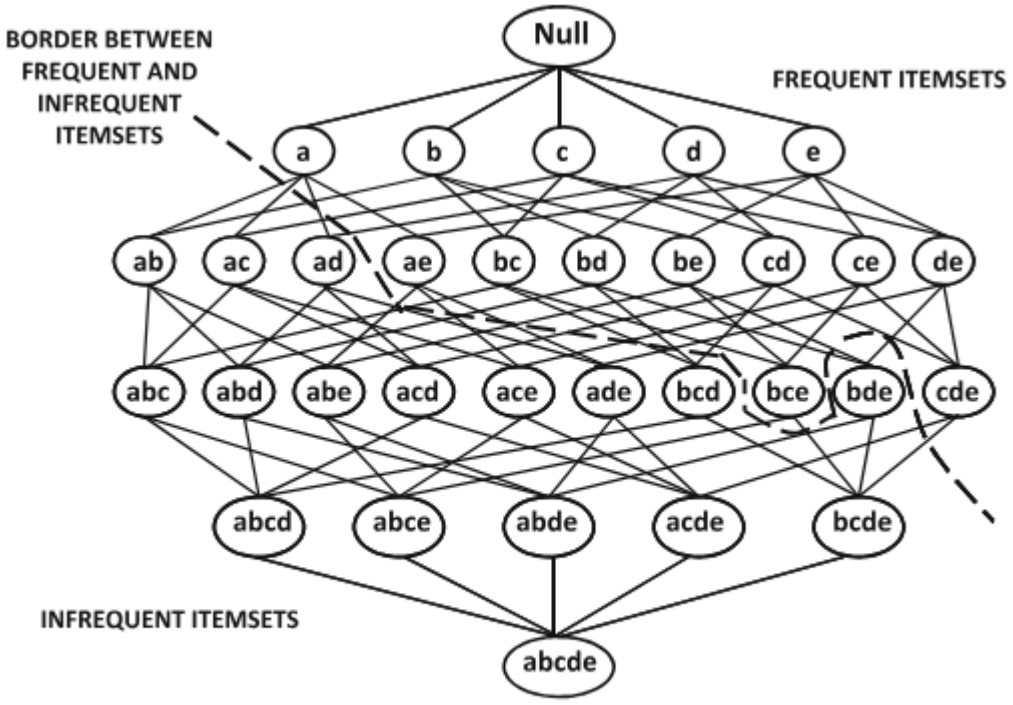


Figure 4.1: The itemset lattice

Nếu ta có một items U, thì số lượng distinct subsets sẽ là $2^{|U|} - 1$, trừ empty set và được biểu diễn như hình 4.1

Phương pháp tính support của một tập *I* bằng cách đếm có phải là tập con của một giao dịch $T_i \in T$ (T là CSDL giao dịch) không là cách không thực tế vì khi $|U| = 1000 \implies \text{no. subset} = 2^{1000} > 10^{300}$ (vượt quá khả năng tính toán của máy tính). Đây còn được gọi là vét cạn

Do đó, ta sẽ tận dụng Downward Closure Property

- Quan sát rằng không có mẫu $K + 1$ nào là frequent nếu không có mẫu K nào frequent
- Tức là tính support các mẫu 1 hạng mục → 2 hạng mục → … → *I* hạng mục với không *I*-mẫu nào là thường xuyên và kết thúc thuật toán tại đó.

4.4.2. Apriori Algorithm

#DataMining/AssociationPatternMining/Apriori

Thuật toán Apriori cũng áp dụng tính chất Downward Closure

Mô tả thuật toán:

```
Algorithm Apriori(Transactions: T, Minimum Support: minsup)
begin
    k = 1;
    F1 = { All Frequent 1-itemsets };
    while Fk is not empty do begin
        Generate Ck+1 by joining itemset-pairs in Fk;
        Prune itemsets from Ck+1 that violate downward closure;
        Determine Fk+1 by support counting on (Ck+1, T) and
retaining
        itemsets from Ck+1 with support at least minsup;
        k = k + 1;
    end;
    return(Uk i=1Fi);
end
```

Truyền vào thụât toán là bộ T và giá trị minsup
Thuật toán bắt đầu với tập F_1 chứa toàn bộ tập chứa 1 item, $k = 1$.
Lặp cho tới khi F_k rỗng thì dừng.

- Tạo tập C_{k+1} bằng cách join từng đôi tập trong F_k
- Cắt nhánh của C_{k+1} nếu vi phạm Downward Closure Property
- Xác định lại F_{k+1} bằng cách đếm support của (C_{k+1}, T) và so sánh support value với minsup
- k += 1
  Return $\cup_{i=1}^{k} F_i$

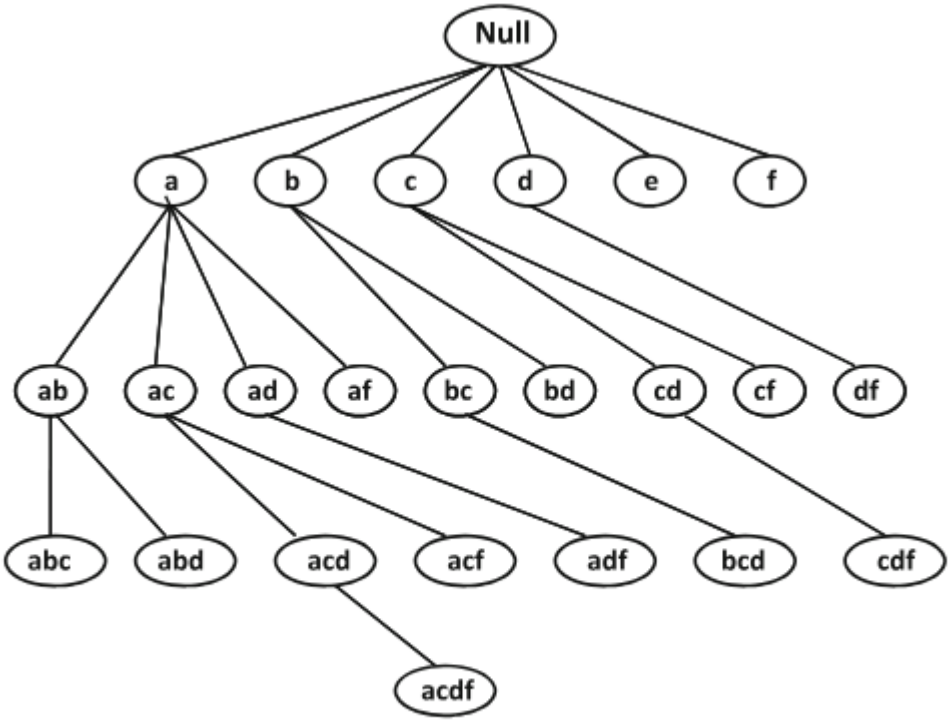


Figure 4.3: The lexicographic or enumeration tree of frequent itemsets

6. Cluster Analysis

#DataMining/DataClustering

6.1. Introduction

Ứng dụng:

1. Data summarization
2. Customer segmentation
3. Social network analysis
4. Relationship to other data mining problem

6.2. Feature Selection for Clustering

#DataMining/DataPreprocessing/FeatureSelection

Mục tiêu chính của việc chọn lọc đặc trưng là loại bỏ các đặc trưng nhiễu

Việc chọn lọc đặc trưng thường khó hơn với các bài toán không giám sát như gom cục do thiếu yếu tố đánh giá như label của dữ liệu

Có hai mô hình cho việc feature selection:

1. Filter model:
2. Wrapper model

6.2.1. Filter model

Trong các mô hình lọc, một số tiêu chí cụ thể được sử dụng để đánh giá ảnh hưởng của các đặc trưng hoặc tập con đặc trưng đến xu hướng phân cụm của tập dữ liệu. Dưới đây là một số tiêu chí thường được sử dụng:

1. Độ Mạnh Của Thuật Ngữ (Term Strength)

- **Mô tả**: Phù hợp cho các miền dữ liệu thưa thớt như dữ liệu văn bản. Trong các miền này, việc quan tâm đến sự hiện diện hay vắng mặt của các giá trị khác 0 trên các thuộc tính (từ ngữ) quan trọng hơn khoảng cách giữa chúng.
- **Cách tính**: Sử dụng hàm similarity thay vì hàm distance. Cặp tài liệu được lấy mẫu và sắp xếp ngẫu nhiên. Độ mạnh của thuật ngữ được định nghĩa là tỷ lệ các cặp tài liệu tương tự (có độ tương tự lớn hơn β), trong đó thuật ngữ xuất hiện ở cả hai tài liệu, với điều kiện nó xuất hiện ở tài liệu đầu tiên. Công thức là:

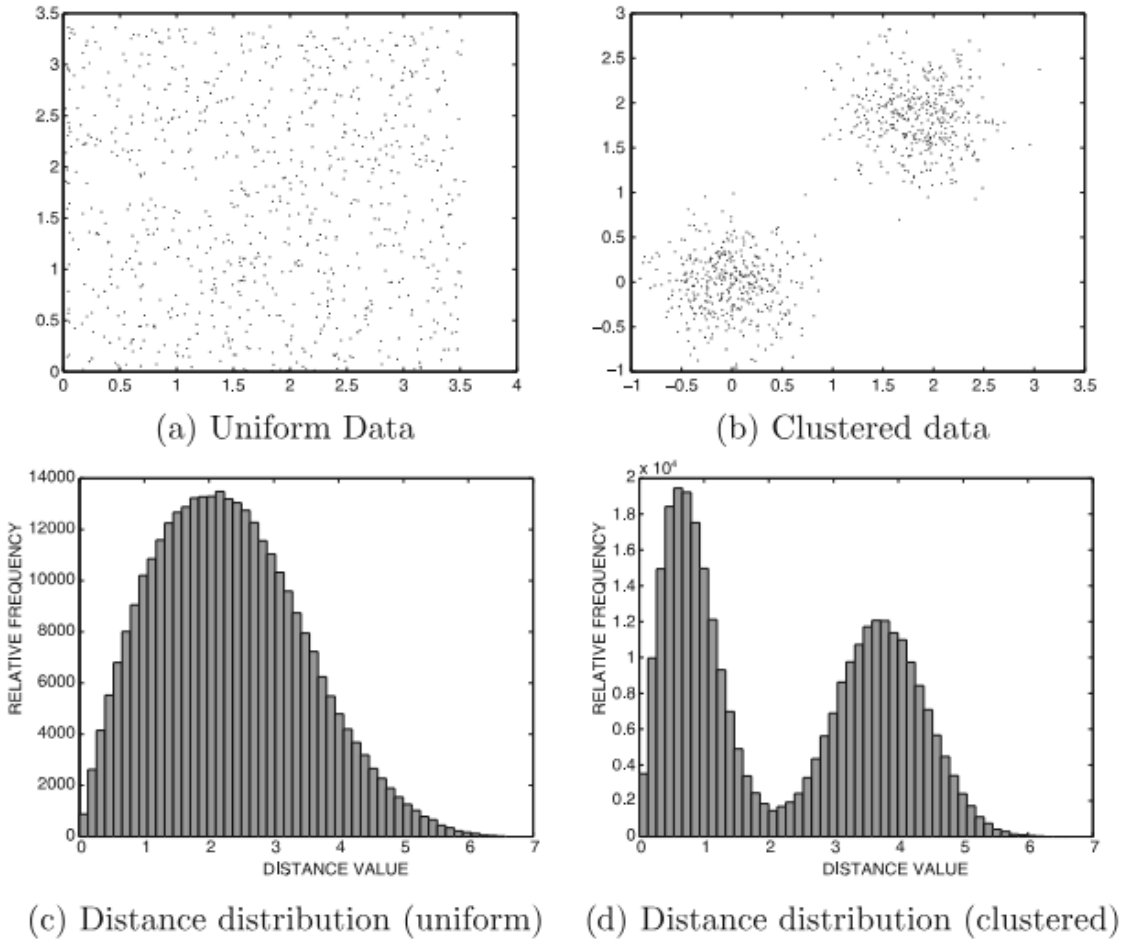
$$\text{Term Strength} = P(t \in X_2 \mid t \in X_1)$$

2. Sự Phụ Thuộc Dự Báo Của Thuộc Tính (Predictive Attribute Dependence)

- **Mô tả**: Đặc trưng liên quan sẽ dẫn đến các cụm tốt hơn đặc trưng không liên quan. Khi một thuộc tính có liên quan, các thuộc tính khác có thể được sử dụng để dự đoán giá trị của nó.
- **Cách tính**: Sử dụng thuật toán phân loại hoặc hồi quy để đánh giá tính dự đoán của thuộc tính.
    1. Sử dụng thuật toán phân loại trên tất cả các thuộc tính, ngoại trừ thuộc tính cần đánh giá, để dự đoán giá trị của thuộc tính đó.
    2. Báo cáo độ chính xác của phân loại như là mức độ liên quan của thuộc tính đó.

3. Entropy

- **Mô tả**: Dữ liệu có sự phân cụm cao sẽ phản ánh đặc tính phân cụm của nó trên các phân phối khoảng cách nền tảng.



(a) Uniform Data (b) Clustered data

(c) Distance distribution (uniform) (d) Distance distribution (clustered)

Figure 6.1: Impact of clustered data on distance distribution entropy

- **Cách tính**:
  - Phân chia dữ liệu thành các vùng lưới đa chiều.
  - Tính xác suất phân phối và entropy dựa trên các giá trị này:

$$E = -\sum_{i=1}^{m} [p_i \log(p_i) + (1 - p_i) \log(1 - p_i)]$$

  - Dữ liệu phân bố đều có entropy cao, trong khi dữ liệu phân cụm có entropy thấp.
- **Cách tiếp cận khác**: Tính entropy dựa trên phân phối khoảng cách điểm-điểm 1 chiều của một mẫu dữ liệu.

4. Thống Kê Hopkins (Hopkins Statistic)

- **Mô tả**: Đo lường xu hướng phân cụm của tập dữ liệu, và có thể áp dụng cho tập con các thuộc tính.
- **Cách tính**:

- Tạo một mẫu dữ liệu tổng hợp S và một mẫu dữ liệu từ tập dữ liệu gốc R.
- Tính khoảng cách từ các điểm trong R đến các láng giềng gần nhất trong dữ liệu gốc D (α_i) và từ các điểm trong S đến các láng giềng gần nhất trong D (β_i).
- Thống kê Hopkins được định nghĩa như sau:

$$H = \frac{\sum_{i=1}^{r} \beta_i}{\sum_{i=1}^{r} (\alpha_i + \beta_i)}$$

- Giá trị của H nằm trong khoảng (0, 1). Dữ liệu phân bố đều có $H \approx 0.5$, trong khi dữ liệu phân cụm có H gần bằng 1.

Các tiêu chí trên có thể kết hợp với các phương pháp tìm kiếm tham lam để phát hiện tập hợp các đặc trưng liên quan cho việc phân cụm dữ liệu.

6.2.2. Wrapper model

Ý tưởng ở đây là dùng một thuật toán gom nhóm với một tập các đặc trưng và sau đó đánh giá chất lượng với tiêu chí được chọn.

Tuy nhiên, nhược điểm lớn của phương pháp này là nó nhạy cảm với các tiêu chí đánh giá được chọn. Hơn nữa, phương pháp này có thể tốn kém về mặt tính toán.

Một cách khác đơn giản hơn là chọn các đặc trưng riêng lẻ với một tiêu chí chọn đặc trưng của các thuật toán phân loại

1. Sử dụng Clustering Algorithm trên subset features hiện tại F, để cố định label L cho các (cụm) điểm dữ liệu
2. Sử dụng bất kì tiêu chí giám sát nào để định lượng chất lượng của các đặc trưng riêng rẽ đối với label L. Sau đó, sort và chọn các đặc trưng hàng đầu dựa trên đánh giá trước đó.

Các tiêu chí có thể sử dụng như:

- Class-based entropy
- Fisher score

Ngoài ra, các mô hình Wrapper thường được kết hợp với mô hình Filter nhằm nâng cao kết quả.

Trong trường hợp này, các tập hợp con đặc trưng được xây dựng bằng cách sử dụng các mô hình filter. Sau đó, chất lượng của mỗi tập hợp con đặc trưng trên được đánh giá bằng thuật toán phân cụm. Việc đánh giá có thể được thực hiện bằng tiêu chí xác định độ hợp lệ của cụm (cluster validity criterion) hoặc bằng cách sử dụng thuật toán phân loại (classification algorithm) trên các nhãn cụm kết quả. Tập hợp con đặc trưng ứng viên tốt nhất được chọn

---

6.3. Representative-Based Algorithms (Thuật Toán Phân Cụm Dựa Trên Đại Diện)

Các thuật toán dựa theo đại diện dựa trực tiếp vào khái niệm khoảng cách (hoặc sự tương đồng) để gom nhóm các điểm dữ liệu.

Trong các thuật toán này, các nhóm được tạo trong một lần và không có quan hệ phân tầng với các nhóm

---

Thuật toán tập trung vào việc xác đinh các đại diện tốt nhất cho từng cụm. Ban đầu, các đại diện được chọn từ các điểm dữ liệu hoặc trung bình (hoặc hàm khác) của các điểm dữ liệu.

Sau khi các đại diện của cụm đã được xác định, ta sẽ đi tìm khoảng cách của các sample với đại diện gần nhất của chúng. Mục tiêu là minimize giá trị khoảng cách đó.

$$O = \sum_{i=1}^{n} \min_j Dist(X_i, Y_j)$$

trong đó, X là samples, Y là đại diện

Các bài toàn tối ưu này thường được giải bằng phương pháp lặp với 2 bước con bên trong như sau:

- (Assign step): Gán mỗi sample cho đại diện gần nhất của bằng hàm khoảng cách và đánh dấu chúng thuộc vê các cluster $C_1, \dots, C_k$
- (Optimize step): Xác định các đại diện Y_j cho từng cluster C_j bằng cách minimize khoảng cách giữa các X_i thuộc C_j với Y_j

6.3.1. k-Means Clustering

Trong thuật toán k-Means, hàm distance là Euclidean

$$Dist(X_i, Y_j) = ||(X_i - Y_j)||_2^2$$

Điểm khác biệt duy nhất giữa thuật toán tổng quát và k-Means là việc sử dụng hàm distance cụ thể và việc chọn đại diện là trung bình cục bộ của cụm

---

Biến thể k-Means với distance Mahalanobis

$$Dist(X_i, Y_j) = (X_i - Y_j)\Sigma_j^{-1}(X_i - Y_j)^T$$

Việc sử dụng khoảng cách Mahalanobis thường hữu ích khi các cụm có dạng elip kéo dài theo các hướng nhất định

Ưu điểm là nó cung cấp khả năng local density normalization (chuẩn hóa mật độ cục bộ), đặc biệt hữu ích với các dataset có local density khác nhau

---

Nhược điểm của k-Means

Thuật toán k-means không hoạt động tốt khi các cụm có hình dạng tùy ý.

Ví dụ, nếu cụm A có dạng phi lồi, k-means sẽ chia nó thành hai phần và cũng có thể gộp một phần này với cụm B. Những tình huống này thường xảy ra với k-means vì nó có xu hướng tìm các cụm hình cầu.

6.3.2. Kernel k-Means Algorithm

Thuật toán k-mean có thể được mở rộng để tìm các cụm với hình dạng bất kì bằng kĩ thuật kernel

Ý tưởng cơ bản là biến đổi ẩn dữ liệu sao cho các cụm với hình dạng bất kì được nối đến các cụm Euclid trong không gian mới.

6.3.3. k-Medoids Clustering

Thuật toán k-medoids mặc dù cũng dùng khái niệm đại diện và hàm mục tiêu, nhưng có cấu trúc thuật toán khác các phương pháp đã đề cập.

Khác biệt chính của thuật toán này là các đại diện luôn được chọn từ các phần tử trong cơ sở dữ liệu.

6.3.4. k-Medians Clustering

6.3.5. Practical and Implementation Issues

Với các thuật toán dựa theo đại diện này, chúng ta có một số vấn đề đáng quan tâm.

1. Tiêu chí khởi tạo: Khởi tạo ngẫu nhiên, Khởi tạo thông qua lấy mẫu hay khởi tạo bằng tâm của các mẫu ngẫu nhiên
2. Điểm ngoại lai: Các điểm ngoại lai thường có tác động xấu đến các thuật toán này. Điều này có thể xảy ra nếu quá trình khởi tạo chọn một điểm ngoại lai làm một trong những tâm ban đầu
3. Lựa chọn số lượng cụm k

6.4. Hierarchical Clustering Algorithms

Các thuật toán gom cụm phân tầng thường gom cụm dữ liệu với khoảng cách. Tuy nhiên, các hàm khoảng cách thường không bắt buộc phải có.

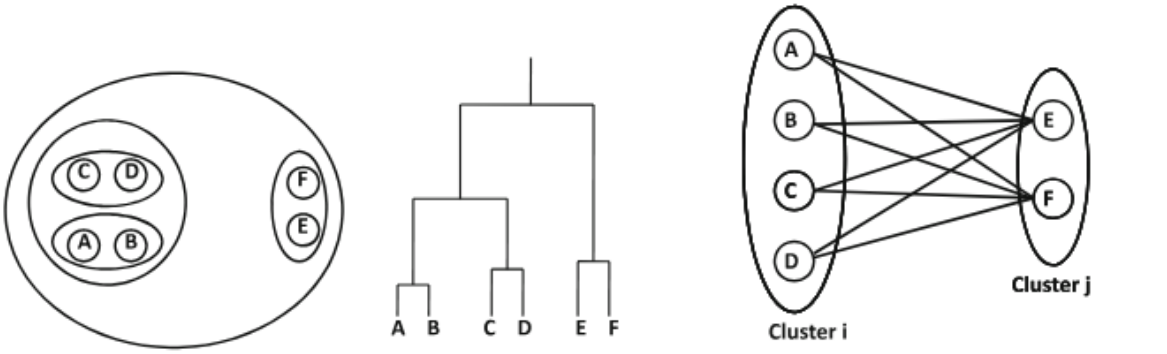
Một lý do chính để sử dụng các phương pháp phân tầng là các độ mịn gom nhóm khác nhau cho chúng ta thêm các hiểu biết cụ thể theo ứng dụng.

Có 2 loại thuật toán phân tầng chính dựa vào cách cây phân tầng được xây dựng thế nào.

1. PP từ dưới lên (agglomerative clustering)
2. PP này trên xuống (divisive clustering)

6.4.1. Agglomerative (Bottom-Up) Clustering

```
Algorithm AgglomerativeMerge(Data: D)
begin
    Initialize n × n distance matrix M using D;
    repeat
        Pick closest pair of clusters i and j using M;
        Merge clusters i and j;
        Delete rows/columns i and j from M and create a new row
and column for newly merged cluster;
        Update the entries of new row and column of M;
    until termination criterion;
    return current merged cluster set;
end
```



(a) Dendrogram (b) Group similarity computation

Figure 6.8: Illustration of hierarchical clustering steps

Lựa Chọn Tiêu Chí Gom Cụm

Những tiêu chí này sẽ ảnh hưởng đến cách tính toán và duy trì ma trận khoảng cách trong quá trình gom cụm.

- Best (single) linkage: là khoảng cách nhỏ nhất giữa các đối tượng trong hai cụm
- Worst (complete) linkage: là khoảng cách lớn nhất giữa các đối tượng trong hai cụm

- Group-average linkage: Là khoảng cách trung bình giữa các đối tượng trong hai cụm
- Closest centroid: là khoảng cách của đại diện hai cụm
- Các tiêu chí dựa theo phương sai: Tối thiểu hóa sự thay đổi trong hàm mục tiêu (ví dụ như phương sai của cụm) sau khi gom. (?...?)
- Ward's Method: là khoảng cách giữa các centroid và nhân với harmonic mean ($\frac{n}{\sum_{i=1}^{n} \frac{1}{x_i}}$) của số lượng samples trong hai cụm

6.4.2. Divisive (Top-Down) Clustering

Phương pháp phân cụm từ trên xuống, mặc dù thường là các phương pháp dựa trên khoảng cách, có thể được coi là các siêu thuật toán (meta-algorithm) tổng quát có thể sử dụng hầu như bất kỳ thuật toán phân cụm nào làm quy trình con

Nhờ cách tiếp cận từ trên xuống, kiểm soát tốt hơn được đạt được về cấu trúc tổng thể của cây về mặt độ sâu và sự cân bằng giữa các nhánh khác nhau.

Cách Tiếp Cận Chung Cho Phân Cụm Từ Trên Xuống

```
Algorithm GenericTopDownClustering(Data: D, Flat Algorithm: A)
begin
    Initialize tree T to root containing D;
    repeat
        Select a leaf node L in T based on pre-defined criterion;
        Use algorithm A to split L into L1 ...Lk;
        Add L1 ...Lk as children of L in T;
    until termination criterion;
end
```

Thuật toán khởi tạo cây tại nút gốc chứa tất cả các điểm dữ liệu. Trong mỗi lần lặp, tập dữ liệu tại một nút cụ thể của cây hiện tại được chia thành nhiều nút (cụm)

Lưu ý rằng thuật toán A có thể là bất kỳ thuật toán phân cụm nào, không chỉ là thuật toán dựa trên khoảng cách.

6.5. Probabilistic Model-Based Algorithms

Các thuật toán như chúng ta đã tìm hiểu mà mỗi điểm dữ liệu được gom xác định gán vào một cụm cụ thể gọi là hard clustering algorithm.

Trong khi đó, các thuật toán dựa theo mô hình xác suất là các soft clustering algorithm mà mỗi điểm dữ liệu được gán xác suất với nhiều (có thể tất cả) nhóm.

Kết quả của thuật toán soft có thể chuyển về kết quả hard bằng cách gán các điểm dữ liệu vào nhóm với xác suất cao nhất

---

Nguyên tắc chung của mô hình sinh hỗn hợp (mixture-based generative model) là giả định rằng dữ liệu được sinh ra từ một hỗn hợp của k phân phối với các phân phối xác suất $\mathrm{G}_1, \ldots, \mathrm{G}_\mathrm{k}$

Mỗi phân phối G_i đại diện cho một cụm và cũng được gọi là một thành phần hỗn hợp (mixture component)

Mỗi điểm dữ liệu X_i, nơi $\mathrm{i} \in \{1, \ldots, \mathrm{n}\}$, được sinh ra bởi mô hình hỗn hợp này như sau:

1. Chọn một thành phần hỗn hợp với xác suất trước $\alpha_\mathrm{i} = \mathrm{P}(\mathrm{G}_\mathrm{i})$, nơi $\mathrm{i} \in \{1, \ldots, \mathrm{k}\}$. Giả sử rằng thành phần thứ r được chọn.
2. Sinh ra một điểm dữ liệu từ G_r.

Mô hình sinh này sẽ được ký hiệu là M. Các xác suất trước α_i và các tham số của các phân phối khác nhau G_r không được biết trước.

Lựa chọn phân phối G_i là quan trọng vì nó phản ánh hiểu biết ban đầu của người dùng về phân phối và hình dạng của các cụm riêng lẻ (các thành phần hỗn hợp).

Các tham số của phân phối của mỗi thành phần hỗn hợp, chẳng hạn như trung bình và phương sai của nó, cần được ước lượng từ dữ liệu để dữ liệu tổng thể có xác suất cao nhất được sinh ra bởi mô hình. Điều này được thực hiện bằng thuật toán tối đa hóa kỳ vọng (Expectation-Maximization, EM).

Giả sử rằng hàm mật độ xác suất của thành phần hỗn hợp G_i được ký hiệu là $f_i(\cdot)$. Xác suất của điểm dữ liệu X_j được sinh ra từ model M

$$f*point(X_j|M) = \sum *i = 1^k \alpha_i \cdot f_i(X_j)$$

Sau đó, đối với một tập dữ liệu D chứa n điểm dữ liệu, ký hiệu là $X_1, \dots, X_n$, hàm mật độ xác suất của tập dữ liệu:

$$f*data(D|M) = \prod *j = 1^n f\_point(X_j|M)$$

$$\Longrightarrow L(D|M) = \log(\prod *j = 1^n f*point(X*j|M)) = \sum *j = 1^n \log(\sum \_i = 1^k \alpha_i f_i($$

Hàm log-likelihood fit này cần được tối đa hóa để xác định các tham số của mô hình.

Giả sử Θ là một vector, đại diện cho toàn bộ tập hợp các tham số mô tả tất cả các thành phần của mô hình hỗn hợp.

Ví dụ, trong trường hợp mô hình hỗn hợp Gaussian, Θ chứa tất cả các trung bình hỗn hợp thành phần, phương sai, ma trận hiệp phương sai và các xác suất sinh trước $\alpha_1, \dots, \alpha_k$.

Sau đó, thuật toán EM bắt đầu với một tập giá trị ban đầu của Θ (có thể tương ứng với sự gán ngẫu nhiên của các điểm dữ liệu vào các thành phần hỗn hợp), và tiến hành như sau:

1. **(Bước E)**: Với giá trị hiện tại của các tham số trong Θ, ước lượng xác suất hậu nghiệm $P(G_i|X_j, \Theta)$ của thành phần G_i được chọn trong quá trình sinh, giả sử rằng chúng ta đã quan sát điểm dữ liệu X_j. Lượng $P(G_i|X_j, \Theta)$ cũng là soft cluster assignment mà chúng ta đang cố gắng ước lượng. Bước này được thực hiện cho mỗi điểm dữ liệu X_j và thành phần hỗn hợp G_i.
2. **(Bước M)**: Với các xác suất hiện tại của việc gán các điểm dữ liệu vào các cụm, sử dụng phương pháp xác suất cực đại để xác định các giá trị của tất cả các tham số trong Θ sao cho tối đa hóa hàm log-likelihood fit

Hai bước này được thực hiện lặp đi lặp lại để cải thiện xác suất cực đại. Thuật toán được coi là hội tụ khi hàm mục tiêu không cải thiện đáng kể

6.6. Grid-Based and Density-Based Algorithms

Một vấn đề quan trọng với các thuật toán dựa theo khoảng cách hoặc các phương pháp xác suất là hình dáng của các nhóm đã được quy định ngầm với hàm khoảng cách hoặc phân phối xác suất.

Đặc tính này sẽ không phù hợp với một số ứng dụng cần các nhóm có hình dạng bất kì.

Với các tình huống thế này thì các phương pháp dựa theo mật độ rất hữu ích.

Ý tưởng chính của phương pháp này là xác định các vùng dày đặc (mật độ cao) trong dữ liệu. Các vùng này sẽ là các “khối xây dựng” cho các cụm với hình dáng bất kì.

Tùy thuộc vào việc lựa chọn các “khối xây dựng” mà chúng ta có các biến thể.

6.6.1. Grid-Based Algorithms

Với các phương pháp này, dữ liệu được rời rạc hóa thành một số các khoảng (thường là cùng chiều rộng).

Các hypercube từ việc rời rạc hóa này chính là các “khối xây dựng” cho thuật toán

Ở đây chúng ta có một threshold mật độ được dùng để xác định xem tập con nào của các hypercube này dày đặc (có mật độ cao).

```
Algorithm GenericGrid(Data: D, Ranges: p, Density: τ )
begin
    Discretize each dimension of data D into p ranges
    Determine dense grid cells at density level τ ;
    Create graph in which dense grids are connected if they are
adjacent;
    Determine connected components of graph;
return points in each connected component as a cluster;
end
```

Figure 6.12: Generic grid-based algorithm

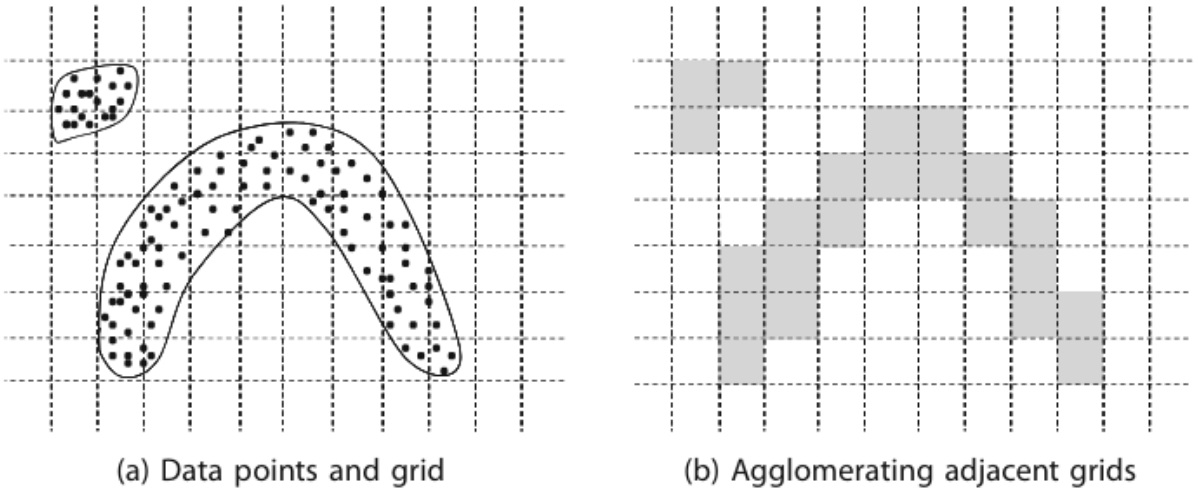


Figure 6.13: Agglomerating adjacent grids

6.6.2. Density-Based Algorithms

6.6.2.1. DBSCAN

```
Algorithm DBSCAN(Data: D, Radius: Eps, Density: τ )
begin
    Determine core, border and noise points of D at level (Eps,
τ);
    Create graph in which core points are connected
        if they are within Eps of one another;
    Determine connected components in graph;
    Assign each border point to connected component
        with which it is best connected;
    return points in each connected component as a cluster;
end
```

Nó được sử dụng để phân loại các điểm dữ liệu thành các nhóm có mật độ cao, phân biệt chúng với các vùng dữ liệu thưa thớt hoặc nhiễu.

Các khái niệm về điểm:

1. Core point (điểm lõm): Là điểm có ít nhất MinPts điểm trong bán kính eps.

2. Border point (điểm biên): Là điểm nằm trong bán kính eps của một điểm core point nhưng không đủ điểm để trở thành core point.
3. Noise point (điểm nhiễu): Là các điểm không phải core point hoặc border point.

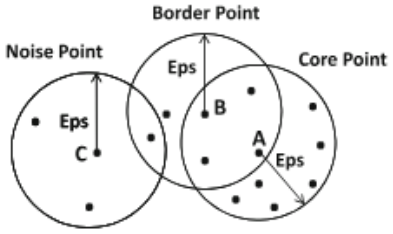


Figure 6.16: Examples of core, border, and noise points

Quá trình phân cụm:

- Bắt đầu từ một điểm bất kỳ chưa được ghép nhóm, DBSCAN tìm tất cả các điểm liên kết với nó thông qua các điểm core point.
- Từ các điểm này, nó mở rộng và tìm kiếm các điểm liên kết tiếp theo, và tiếp tục quá trình này cho đến khi không còn điểm nào có thể thêm vào cụm hiện tại.
- Các điểm còn lại sẽ được đánh dấu là điểm nhiễu.

6.6.2.2. DENCLUE

```
Algorithm DENCLUE(Data: D, Density: τ )
begin
    Determine density attractor of each data point in D with
        gradient-ascent of Equation 6.20;
    Create clusters of data points that converge to the same
        density attractor;
    Discard clusters whose density attractors have density less
        than τ and report as outliers;
    Merge clusters whose density attractors are connected with
        a path of density at least τ ;
    return points in each cluster;
end
```

Tương tự như DBSCAN, DENCLUE là một phương pháp phân cụm dựa trên mật độ.

Đây là một phương pháp phát triển từ DBSCAN nhằm giải quyết một số vấn đề mà DBSCAN gặp phải, như hiệu suất và khả năng xử lý dữ liệu lớn hơn.

Thuật toán sử dụng hàm nhân (kernel function) để ước lượng mật độ của không gian dữ liệu. Hàm mật độ này sẽ xác định mức độ tập trung của các điểm dữ liệu trong không gian.

$$f(X) = \frac{1}{n}\sum_{i=1}^{n} K(X - X_i)$$

Một hàm nhân phổ biến thường được sử dụng là hàm nhân Gaussian.

$$K(X - X_i) = (\frac{1}{h\sqrt{2\pi}})^d e^{-\frac{||X - X_i||^2}{2h^2}}$$

Thuật toán sẽ tập trung tìm attrator points (điểm thu hút) - là những điểm mật độ cục bộ cực đại

Sử dụng gradient ascent (dùng pp gradient để đi về cực đại, ngược với gradient descent) để di chuyển từ một điểm dữ liệu ban đầu đến điểm thu hút gần nhất. Quá trình này được lặp lại cho đến khi đạt tới một điểm thu hút (điểm cực đại của hàm mật độ).

Cách tính:

1. Xác định hướng gradient của hàm mật độ f

$$\nabla f = \frac{1}{n}\sum_{i=1}^{n} \nabla K(X - X_i)$$

2. Cập nhật vị trí: $X^{(t+1)} = X^{(t)} - \alpha \nabla f(X^{(t)})$
3. Lặp lại quá trình cho đến khi hội tụ tại đến một điểm (giá trị gradient xấp xỉ bằng 0, tức ko còn di chuyển đáng kể)

Tất cả các điểm dữ liệu dẫn đến cùng một điểm thu hút sẽ thuộc về cùng một cụm. Quá trình này được lặp lại cho tất cả các điểm dữ liệu trong không gian.

Các cụm được xác định dựa trên mật độ và khoảng cách giữa các điểm thu hút. Các điểm dữ liệu không dẫn đến bất kỳ điểm thu hút nào hoặc có mật độ dưới ngưỡng xác định sẽ được coi là nhiễu.

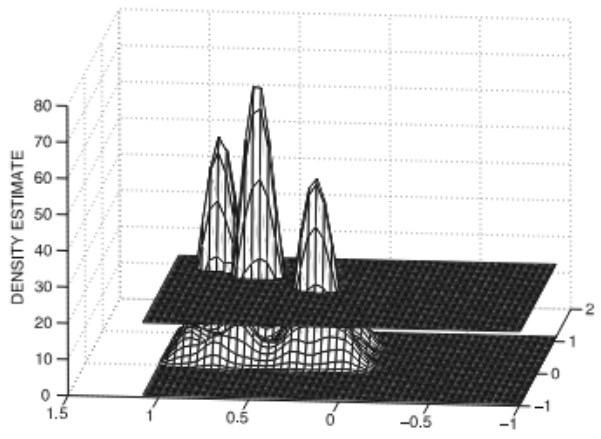


Figure 6.18: Density-based profile with lower density threshold

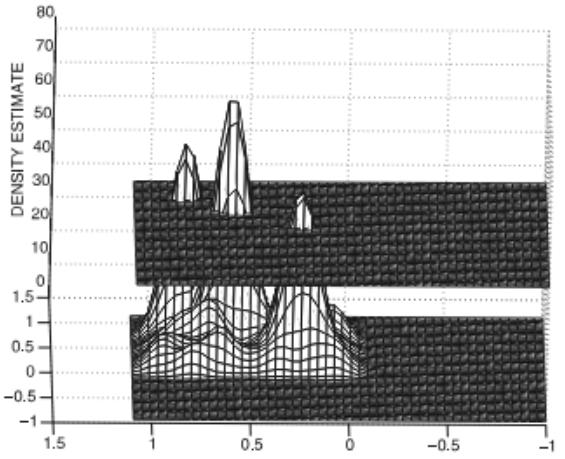


Figure 6.19: Density-based profile with higher density threshold

6.7. Graph-Based Algorithms

Các thuật toán dựa theo đồ thị cung cấp một meta-framework chung mà trong đó gần như tất cả kiểu dữ liệu đều có thể được gom nhóm.

Ý quan trọng cần để ý ở đây là gần như tất cả kiểu dữ liệu đều có thể được biến đổi thành đồ thị tương đồng để thực hiện phân tích.