WEB DATA MINING

UNIT-I
Contents

- The World Wide Web
- History of the Web and the Internet
- Data Mining
- Web Mining
- Basic concepts of Association Rules
- Apriori algorithm
  - Frequent item set generation
  - Association Rule Generation
- Different data formats for mining
- Mining with multiple minimum supports-Extended Model
- Mining Algorithm
- Rule Generation

The World Wide Web: Connecting Our Digital World

Exploring the history, impact, and technology that transformed how we access and share information globally.

Introduction

Largest Information Source

The World Wide Web has become the biggest and most widely accessible information repository in human history, impacting nearly every aspect of our daily lives.

Interconnected Documents

The Web consists of billions of interconnected documents called web pages, authored by millions of people worldwide and linked through hyperlinks.

Information at Your Fingertips

Everything is just a few clicks away from your home or office—a dramatic shift from asking experts or searching through physical books.

Before the Web, finding information meant consulting friends, experts, or spending time in libraries. Today, knowledge is instantly accessible and easily shareable with others around the globe.

The Web's Impact on Our Lives

Information Discovery

Easily find and access the world's largest information source with powerful search capabilities.

Knowledge Sharing

Share information and expertise with global audiences effortlessly through Web platforms.

E-Commerce

Buy almost anything from online stores without visiting physical shops—convenience redefined.

Global Communication

Express views, discuss ideas, and connect with people anywhere in the world instantly.

The Web as a Business Channel

Online Shopping

Purchase almost anything without visiting physical stores—from groceries to electronics.

Digital Commerce

The Web provides an important channel for conducting business transactions efficiently.

Global Marketplace

Connect buyers and sellers across borders, creating unprecedented business opportunities.

What is the World Wide Web?

Simple Definition

The Web is an Internet-based computer network that allows users of one computer to access information stored on another through the worldwide network called the Internet.

Official Definition

A wide-area hypermedia information retrieval initiative aiming to give universal access to a large universe of documents. Hypermedia means a nonlinear medium of information that includes text, images, audio, and video.

How the Web Works

Client-Server Model

Users rely on a **browser** (client program) to connect to remote servers where data is stored. Browsers send requests and interpret HTML documents, displaying text and graphics on your screen.

Hypertext Foundation

Web pages use **hyperlinks** to connect documents across the world. Invented by Ted Nelson in 1965, hypertext (and hypermedia) enables seamless navigation between related content.

The Power of Hypertext & Hypermedia

01

Hypertext Structure

The Web's operation depends on the structure of hypertext documents that allow authors to link their content to related documents anywhere in the world.

02

Following Hyperlinks

Users simply follow hyperlinks to view connected documents across different computers globally—making navigation seamless.

03

Hypermedia Evolution

Hypertext evolved into hypermedia, which includes images, audio, and video files—not just text.

Historical Note: The idea of hypertext was invented by Ted Nelson in 1965, who also created the well-known hypertext system Xanadu (http://xanadu.com/).

A Brief History: The Birth of the Web

March 1989

1

Tim Berners-Lee invents the World Wide Web at CERN in Switzerland and coins the term "World Wide Web"

2

1989-1990

Writes the first World Wide Web server (httpd) and the first client program—a browser and editor called "WorldWideWeb"

Foundation Created

3

Establishes the basic architecture for the distributed hypertext system we use today

Tim Berners-Lee, inventor of the World Wide Web

Learn more: https://www.w3.org/People/Berners-Lee/

The Original Vision: "Information Management"

Tim Berners-Lee's 1989 Proposal

Problem Identified

Discussed disadvantages of hierarchical information organization and outlined advantages of hypertext-based systems

Solution Proposed

A simple protocol to request information from remote systems and exchange it in a common format with hyperlinks connecting documents

Technical Vision

Methods for reading text and graphics using existing display technology—essentially a distributed hypertext system

In 1990, after re-circulating his proposal, Berners-Lee received support to begin work. His team introduced the server, browser, HTTP protocol, HTML markup language, and URL—laying the foundation for the Web's future.

The Browser Revolution: Mosaic & Netscape

Mosaic Arrives (February 1993)

Four years after the Web's discovery, Mosaic was released with versions for Macintosh and Windows—the first consistent, point-and-click graphical interface across major operating systems.

Netscape Released

Within months, the Netscape browser launched to the public, triggering the explosive growth of the Web and making it accessible to millions.

Internet Explorer (August 1995)

Microsoft entered the market with Internet Explorer, beginning the famous "browser wars" and further popularizing Web access.

World Wide Web

- The World Wide Web is officially defined as a **“wide-area hypermedia information retrieval initiative aiming to give universal access to a large universe of documents.”**
- In simple terms, the Web is an **Internet-based computer network** that enables users on one computer to access information stored on another computer via the Internet.
- The Web follows a standard **client–server model**:
  The **client** is the program used by the user to connect to remote machines.
  The **server** is the remote machine where the data is stored.
- Navigation on the Web is done using a **web browser** (client program), such as:
  Netscape
  Internet Explorer
  Firefox
  Chrome
- Web browsers work by:
  Sending **requests** to remote servers for information.
  Interpreting and displaying the information retrieved from those servers.

Hypertext and Web Operation

- The Web operates based on the structure of **hypertext documents**.
- **Hypertext** enables authors to link one document to other related documents across the world.
- These linked documents can reside on **any computer connected to the Internet**.
- Users can access the related documents simply by following **hyperlinks** (links within the page).

History of the Web and the Internet

•**1989:** Tim Berners-Lee created the Web at CERN; introduced **HTTP, HTML, URL**, first server & browser.

•**1993:** Mosaic browser released → first graphical, cross-platform Web browser.

•**1994:** Netscape founded; Web usage exploded.

•**1995:** Microsoft launched Internet Explorer → browser wars.

•**Internet origins:** Began as **ARPANET** (1969); **TCP/IP** developed by Cerf & Kahn (1973–82); Internet born in 1982.

•**Search engines:** Excite (1993), Yahoo! (1994), Google (1998), MSN/Bing (2005).

•**W3C (1994):** Sets standards for Web technologies.

What is Data Mining?

Data mining is also called **knowledge discovery in databases (KDD)**.

It is commonly defined as the process of discovering useful patterns or knowledge from data sources, e.g., databases, texts, images, the Web, etc.

Data mining is a multi-disciplinary field involving machine learning, statistics, databases, artificial intelligence, information retrieval, and visualization.

There are many data mining tasks. Some of the common ones are

1. Supervised learning (or classification),
2. Unsupervised learning (or clustering),
3. Association rule mining, and sequential pattern mining.

Data mining can be performed, in three main steps:

1.Pre-processing: The raw data is usually not suitable for mining due to various reasons. It may need to be cleaned to remove noises or abnormalities. The data may also be too large and/or involve many irrelevant attributes, which call for data reduction through sampling and attribute or feature selection.

2. **Data mining:** The processed data is then fed to a data mining algorithm which will produce patterns or knowledge.

3. **Post-processing:** In many applications, not all discovered patterns are useful. This step identifies those useful ones for applications. Various evaluation and visualization techniques are used to make the decision.

Applications:

Data mining has a wide range of applications across various industries, including

- Marketing,
- Finance,
- Healthcare, and
- Telecommunications.

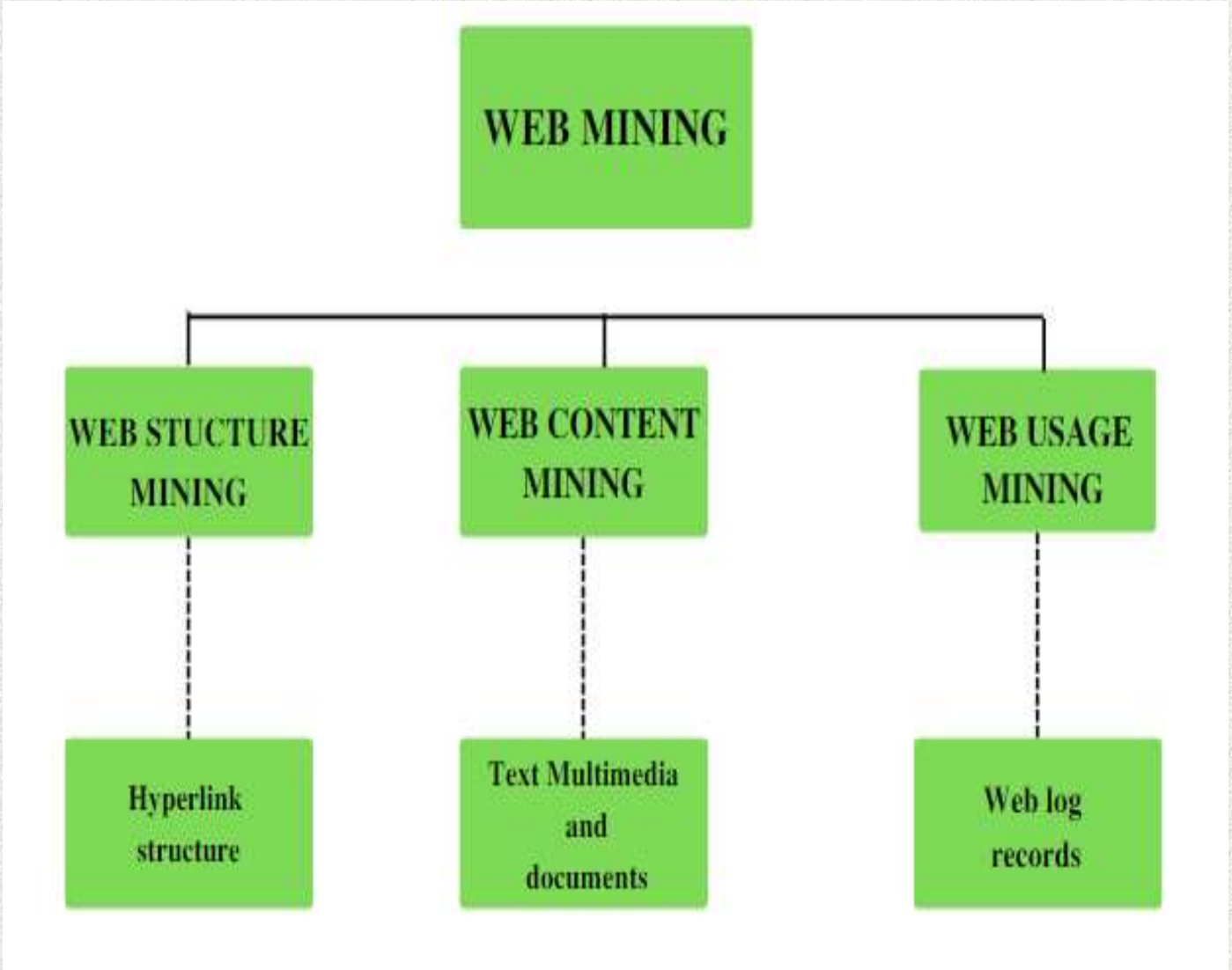
For example, in healthcare, it can be used to identify risk factors for diseases and develop personalized treatment plans.

What is Web Mining?

Web mining aims to discover useful information or knowledge from the Web hyperlink structure, page content, and usage data.

Web mining can be broadly divided into three different types of techniques of mining:

1. Web Structure Mining,
2. Web Content Mining, and
3. Web Usage Mining.


WEB MINING
WEB STUCTURE MINING
WEB CONTENT MINING
WEB USAGE MINING
Hyperlink structure
Text Multimedia and documents
Web log records

Web Structure Mining:

- Web structure mining is the application of discovering structure information from the web.
- The structure of the web graph consists of web pages as nodes, and hyperlinks as edges connecting related pages.
- Structure mining basically shows the structured summary of a particular website.
- For example, from the links, we can discover important Web pages, which is a key technology used in search engines.
- We can also discover communities of users who share common interests

Web Content Mining:

- Web content mining is the application of extracting useful information from the content of the web documents.
- Web content consist of several types of data – text, image, audio, video etc.
- Content data is the group of facts that a web page is designed. It can provide effective and interesting patterns about user needs.
- Text documents are related to text mining, machine learning and natural language processing.
- This mining is also known as text mining. This type of mining performs scanning and mining of the text, images and groups of web pages according to the content of the input.

Web Usage Mining:

- Web usage mining involves analyzing user behavior on the web, including clickstream data, search queries, and other interactions with web pages.
- Web usage mining can help identify user preferences, behavior patterns, and trends.
- This information can be used to personalize content, improve website design, and target advertising.
- Web usage mining can also be used for security purposes, such as detecting fraud and identifying potential security threats.

Applications of Web Mining:

The applications of web mining are wide-ranging and include:

E-commerce -

Web mining is used to analyze user behavior on e-commerce websites, including purchase history, search queries, and clickstream data. This information can be used to optimize website design, personalize product recommendations, and improve customer experience.

Search engine optimization:

Web mining can be used to analyze search engine queries and search engine results pages (SERPs). This information can be used to improve the visibility of websites in search engine results and increase traffic to the website.

Fraud detection:

Web mining can be used to detect fraudulent activity on websites. This information can be used to prevent financial fraud, identity theft, and other types of online fraud.

The process of web mining typically involves the following steps -

- **Data collection** -
  Web data is collected from various sources, including web pages, databases, and APIs.
- **Data pre-processing** -
  The collected data is pre-processed to remove irrelevant information, such as advertisements and duplicate content.
- **Data integration** -
  The pre-processed data is integrated and transformed into a structured format for analysis.
- **Pattern discovery** -
  Web mining techniques are applied to identify patterns, trends, and relationships.
- **Evaluation** -
  The discovered patterns are evaluated to determine their significance and usefulness.
- **Visualization** -
  The analysis results are visualized through graphs, charts, and other visualizations.

Difference Between Data Mining and Web Mining

Parameter	Data Mining	Web Mining
Definition	The process of discovering patterns in large datasets	The process of discovering patterns in web data
Data Source	Databases, data warehouses, and other data repositories	Web pages, weblogs, social media, and other web-related data sources
Data Characteristics	Structured, semi-structured, and unstructured data	Mostly unstructured data
Techniques	Clustering, classification, association rules, regression, etc.	Text mining, natural language processing, image analysis, link analysis, etc.
Applications	Marketing, finance, healthcare, etc.	E-commerce, social media, search engines, etc.
Challenges	Data quality, scalability, and privacy concerns	Data heterogeneity, ambiguity, and dynamic nature of the web

Association Rules and Sequential Patterns:

Association rule is a fundamental data mining task.

Proposed by Agrawal et al in 1993.

Its objective is to find all co-occurrence relationships, called **associations**, among data items.

The classic application of association rule mining is the **market basket data analysis,**

which aims to discover how items purchased by customers in a supermarket (or a store) are

associated.

Association rule mining

- Proposed by Agrawal et al in 1993.
- It is an important data mining model studied extensively by the database and data mining community.
- Assume all data are categorical.
- No good algorithm for numeric data.
- Initially used for Market Basket Analysis to find how items purchased by customers are related.

Bread $\rightarrow$ Milk [sup = 5%, conf = 100%]

The model: data

- $I = \{i_1, i_2, \ldots, i_m\}$: a set of *items*.
- Transaction t :
  - t a set of items, and $t \subseteq I$.
- Transaction Database T: a set of transactions $T = \{t_1, t_2, \ldots, t_n\}$.

Transaction data: supermarket data

- Market basket transactions:

  t1: {bread, cheese, milk}

  t2: {apple, eggs, salt, yogurt}

  … …

  tn: {biscuit, eggs, milk}

- Concepts:
  - An *item*: an item/article in a basket
  - *I*: the set of all items sold in the store
  - A *transaction*: items purchased in a basket; it may have TID (transaction ID)
  - A *transactional dataset*: A set of transactions

The model: rules

- A transaction *t* contains *X*, a set of items (itemset) in *I*, if $X \subseteq t$.
- An association rule is an implication of the form:

  $X \rightarrow Y$, where $X, Y \subset I$, *and* $X \cap Y = \varnothing$

- An itemset is a set of items.
  - E.g., X = {milk, bread, cereal} is an itemset.
- A *k*-itemset is an itemset with *k* items.
  - E.g., {milk, bread, cereal} is a 3-itemset

Rule strength measures :

- Support: The rule holds with support *sup* in *T* (the transaction data set) if sup% of transactions contain $X \cup Y$.
  - *sup* = Pr($X \cup Y$).
- Confidence: The rule holds in *T* with confidence *conf* if *conf*% of tranactions that contain *X* also contain *Y.*
  - *conf* = Pr($Y \mid X$)
- An association rule is a pattern that states when *X* occurs, *Y* occurs with certain probability.

Support and Confidence :

- Support count: The support count of an itemset *X*, denoted by *X.count*, in a data set *T* is the number of transactions in *T* that contain *X*. Assume *T* has *n* transactions.
- Then,

$$support = \frac{(X \cup Y).count}{n}$$

$$confidence = \frac{(X \cup Y).count}{X.count}$$

Goal and key features

- **Goal:** Find all rules that satisfy the user-specified *minimum support* (minsup) and *minimum confidence* (minconf).
- **Key Features**
  - Completeness: find all rules.
  - No target item(s) on the right-hand-side
  - Mining with data on hard disk (not in memory)

An example

- Transaction data
- Assume:

  minsup = 30%

  minconf = 80%
- An example frequent *itemset*:

  {Chicken, Clothes, Milk} [sup = 3/7]
- Association rules from the itemset:

  Clothes → Milk, Chicken [sup = 3/7, conf = 3/3]

  …

  Clothes, Chicken → Milk, [sup = 3/7, conf = 3/3]

t1: Beef, Chicken, Milk

t2: Beef, Cheese

t3: Cheese, Boots

t4: Beef, Chicken, Cheese

t5: Beef, Chicken, Clothes, Cheese, Milk

t6: Chicken, Clothes, Milk

t7: Chicken, Milk, Clothes

Transaction data representation

- A simplistic view of shopping baskets,
- Some important information not considered. E.g,
  - the quantity of each item purchased and
  - the price paid.

Many mining algorithms

- There are a large number of them!!
- They use different strategies and data structures.
- Their resulting sets of rules are all the same.
  - Given a transaction data set *T*, and a minimum support and a minimum confident, the set of association rules existing in *T* is uniquely determined.
- Any algorithm should find the same set of rules although their computational efficiencies and memory requirements may be different.
- We study only one: the Apriori Algorithm

Road map

- Basic concepts of Association Rules
- Apriori algorithm
- Different data formats for mining
- Mining with multiple minimum supports
- Mining class association rules
- Sequential pattern mining
- Summary

The Apriori algorithm :

- **The best known algorithm**
- **Two steps**:
  - Find all itemsets that have minimum support (*frequent itemsets*, also called large itemsets).
  - Use frequent itemsets to generate rules.

- E.g., a frequent itemset

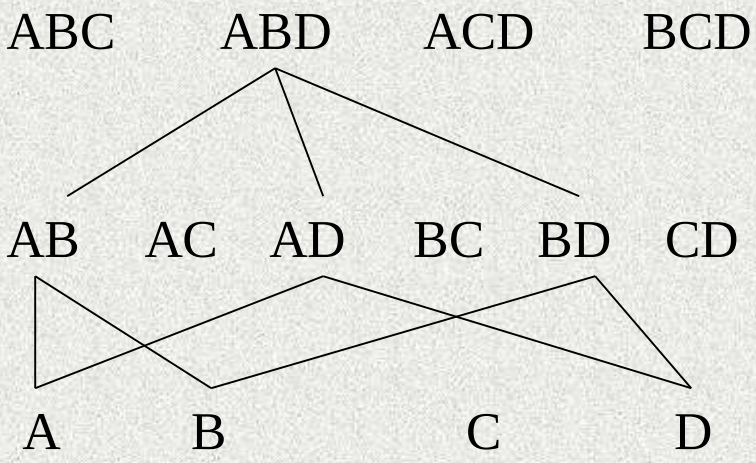
  {Chicken, Clothes, Milk} [sup = 3/7]

  and one rule from the frequent itemset

  Clothes $\rightarrow$ Milk, Chicken [sup = 3/7, conf = 3/3]

Step 1: Mining all frequent itemsets

- A frequent *itemset* is an itemset whose support is ≥ minsup.
- Key idea: The apriori property (downward closure property): any subsets of a frequent itemset are also frequent itemsets



The Algorithm

- Iterative algo. (also called level-wise search): Find all 1-item frequent itemsets; then all 2-item frequent itemsets, and so on.
  - In each iteration *k*, only consider itemsets that contain some *k*-1 frequent itemset.
- Find frequent itemsets of size 1: F_1
- From $k = 2$
  - C_k = candidates of size *k*: those itemsets of size *k* that could be frequent, given F_{k-1}
  - F_k = those itemsets that are actually frequent, $F_k \subseteq C_k$ (need to scan the database once).

Example –
Finding frequent itemsets

minsup=0.5

Dataset T

TID	Items
T100	1, 3, 4
T200	2, 3, 5
T300	1, 2, 3, 5
T400	2, 5

itemset:count

1. scan T ➔ C_1: {1}:2, {2}:3, {3}:3, {4}:1, {5}:3

   ➔ F_1: {1}:2, {2}:3, {3}:3, {5}:3

   ➔ C_2: {1,2}, {1,3}, {1,5}, {2,3}, {2,5}, {3,5}

2. scan T ➔ C_2: {1,2}:1, {1,3}:2, {1,5}:1, {2,3}:2, {2,5}:3, {3,5}:2

   ➔ F_2: **{1,3}**:2, **{2,3}**:2, **{2,5}**:3, **{3,5}**:2

   ➔ C_3: {2, 3,5}

3. scan T ➔ C_3: **{2, 3, 5}**:2 ➔ F_3: **{2, 3, 5}**

Details: ordering of items

- The items in *I* are sorted in lexicographic order (which is a total order).
- The order is used throughout the algorithm in each itemset.
- {*w*[1], *w*[2], …, *w*[*k*]} represents a *k*-itemset *w* consisting of items *w*[1], *w*[2], …, *w*[*k*], where *w*[1] < *w*[2] < … < *w*[*k*] according to the total order.

Details: the algorithm :

Algorithm Apriori(*T*)

$C_1 \leftarrow$ init-pass(T);

$F_1 \leftarrow \{f \mid f \in C_1, f.\text{count}/n \geq minsup\}$; // n: no. of transactions in T

for (k = 2; $F_{k-1} \neq \emptyset$; k++) **do**

$C_k \leftarrow$ candidate-gen(F_{k-1});

for each transaction $t \in T$ **do**

for each candidate $c \in C_k$ **do**

if c is contained in t **then**

$c.count$++;

end

end

$F_k \leftarrow \{c \in C_k \mid c.count/n \geq minsup\}$

end

return $F \leftarrow \bigcup_k F_k$;

Apriori candidate generation

- The candidate-gen function takes F_{k-1} and returns a superset (called the candidates) of the set of all frequent k-itemsets. It has two steps
  - *join* step: Generate all possible candidate itemsets C_k of length k
  - *prune* step: Remove those candidates in C_k that cannot be frequent.

Candidate-gen function

Function candidate-gen(F_{k-1})

$C_k \leftarrow \emptyset$;

forall $f_1, f_2 \in F_{k-1}$

with $f_1 = \{i_1, \dots, i_{k-2}, i_{k-1}\}$

and $f_2 = \{i_1, \dots, i_{k-2}, i'_{k-1}\}$

and $i_{k-1} < i'_{k-1}$ **do**

$c \leftarrow \{i_1, \dots, i_{k-1}, i'_{k-1}\}$; // join f_1 and f_2

$C_k \leftarrow C_k \cup \{c\}$;

for each (k-1)-subset s of c **do**

if ($s \notin F_{k-1}$) **then**

delete c from C_k; // prune

end

end

return C_k;

An example

- F_3 = {{1, 2, 3}, {1, 2, 4}, {1, 3, 4},
  {1, 3, 5}, {2, 3, 4}}

- After join
  - C_4 = {{1, 2, 3, 4}, {1, 3, 4, 5}}
- After pruning:
  - C_4 = {{1, 2, 3, 4}}
    because {1, 4, 5} is not in F_3 ({1, 3, 4, 5} is removed)

Step 2: Generating rules from frequent itemsets

- Frequent itemsets ≠ association rules
- One more step is needed to generate association rules
- For each frequent itemset *X*,

  For each proper nonempty subset *A* of *X*,
  - Let *B* = X - *A*
  - A $\rightarrow$ B is an association rule if
    - Confidence(A $\rightarrow$ B) ≥ minconf,

      support(A $\rightarrow$ B) = support(A$\cup$B) = support(X)

      confidence(A $\rightarrow$ B) = support(A $\cup$ B) / support(A)

Generating rules: an example

- Suppose {2,3,4} is frequent, with sup=50%
  - Proper nonempty subsets: {2,3}, {2,4}, {3,4}, {2}, {3}, {4}, with sup=50%, 50%, 75%, 75%, 75%, 75% respectively
  - These generate these association rules:
    - 2,3 $\rightarrow$ 4, confidence=100%
    - 2,4 $\rightarrow$ 3, confidence=100%
    - 3,4 $\rightarrow$ 2, confidence=67%
    - 2 $\rightarrow$ 3,4, confidence=67%
    - 3 $\rightarrow$ 2,4, confidence=67%
    - 4 $\rightarrow$ 2,3, confidence=67%
    - All rules have support = 50%

Generating rules: summary

- To recap, in order to obtain $A \rightarrow B$, we need to have support($A \cup B$) and support(A)
- All the required information for confidence computation has already been recorded in itemset generation. No need to see the data *T* any more.
- This step is not as time-consuming as frequent itemsets generation.

On Apriori Algorithm

Seems to be very expensive

- Level-wise search
- K = the size of the largest itemset
- It makes at most K passes over data
- In practice, K is bounded (10).
- The algorithm is very fast. Under some conditions, all rules can be found in linear time.
- Scale up to large data sets

More on association rule mining

- Clearly the space of all association rules is exponential, $O(2^m)$, where m is the number of items in *I*.
- The mining exploits sparseness of data, and high minimum support and high minimum confidence values.
- Still, it always produces a huge number of rules, thousands, tens of thousands, millions, ...

Road map

- Basic concepts of Association Rules
- Apriori algorithm
- Different data formats for mining
- Mining with multiple minimum supports
- Mining class association rules
- Sequential pattern mining
- Summary

Different data formats for mining :

- The data can be in transaction form or table form

  Transaction form: a, b

  a, c, d, e

  a, d, f

  Table form:

Attr1	Attr2	Attr3
a,	b,	d
b,	c,	e

- Table data need to be converted to transaction form for association mining

From a table to a set of transactions

Table form:

Attr1	Attr2	Attr3
a,	b,	d
b,	c,	e

⇒Transaction form:

(Attr1, a), (Attr2, b), (Attr3, d)

(Attr1, b), (Attr2, c), (Attr3, e)

Road map

- Basic concepts of Association Rules
- Apriori algorithm
- Different data formats for mining
- Mining with multiple minimum supports
- Mining class association rules
- Sequential pattern mining
- Summary

Problems with the association mining

- Single minsup: It assumes that all items in the data are of the same nature and/or have similar frequencies.
- Not true: In many applications, some items appear very frequently in the data, while others rarely appear.

  E.g., in a supermarket, people buy *food processor* and *cooking pan* much less frequently than they buy *bread* and *milk*.

Rare Item Problem :

- If the frequencies of items vary a great deal, we will encounter two problems
  - If minsup is set too high, those rules that involve rare items will not be found.
  - To find rules that involve both frequent and rare items, minsup has to be set very low. This may cause combinatorial explosion because those frequent items will be associated with one another in all possible ways.

Multiple minsups model

- The minimum support of a rule is expressed in terms of *minimum item supports* (MIS) of the items that appear in the rule.
- Each item can have a minimum item support.
- By providing different MIS values for different items, the user effectively expresses different support requirements for different rules.
- To prevent very frequent items and very rare items from appearing in the same itemsets, we introduce a **support difference constraint**.

$$max_{i \in s}\{sup\{i\} - min_{i \in s}\{sup(i)\} \leq \varphi,$$

Extended Model

- In the extended model, the minimum support of a rule is expressed in terms of minimum item supports (MIS) of the items that appear in the rule.
- That is, each item in the data can have a MIS value spe cified by the user.
- Let MIS(i) be the MIS value of item i. The *minsup* of a rule R is the lowest MIS value of the items in the rule.
- I.e., a rule R: $a_1, a_2, \ldots, a_k \rightarrow a_{k+1}, \ldots, a_r$ satisfies its minimum support if its actual support is $\geq$

  $\min(\text{MIS}(a_1), \text{MIS}(a_2), \ldots, \text{MIS}(a_r))$.

An Example1

- Consider the following items:

  bread, shoes, clothes

  The user-specified MIS values are as follows:

  MIS(*bread*) = 2% MIS(*shoes*) = 0.1%

  MIS(*clothes*) = 0.2%

  i)The following rule doesn't satisfy its minsup:

  clothes → *bread* [sup=0.15%,conf =70%]

  MIS({clothes,bread})=min(0.2%,2%)=0.2%.

Given support = **0.15%**, and **0.15% < 0.2%**, the itemset is **not frequent**,

so the rule **does not satisfy its minsup** (confidence 70% is irrelevant to the minsup test).

ii)The following rule satisfies its minsup:

clothes $\rightarrow$ *shoes* [sup=0.15%,conf =70%]

MIS({clothes,shoes}) = min(0.2%, 0.1%) = 0.1%

Given support = **0.15%**, and **0.15% ≥ 0.1%**, the itemset is **frequent**, so the rule **satisfies its minsup**

An Example 2

- Consider the following items:

  bread, shoes, clothes

  The user-specified MIS values are as follows:

  MIS(*bread*) = 2% MIS(*shoes*) = 0.1%

  MIS(*clothes*) = 0.2%

  The following rule doesn't satisfy its minsup:

  clothes → *bread* [sup=0.15%,conf =70%]

  The following rule satisfies its minsup:

  clothes → *shoes* [sup=0.15%,conf =70%]

Downward closure property

- In the new model, the property no longer holds (?)

E.g., Consider four items 1, 2, 3 and 4 in a database. Their minimum item supports are

MIS(1) = 10% MIS(2) = 20%

MIS(3) = 5% MIS(4) = 6%

{1, 2} with support 9% is infrequent, but {1, 2, 3} and {1, 2, 4} could be frequent.

To deal with the problem

- We sort all items in *I* according to their MIS values (make it a total order).
- The order is used throughout the algorithm in each itemset.
- Each itemset *w* is of the following form:

  {*w*[1], *w*[2], …, *w*[*k*]}, consisting of items,

  w[1], *w*[2], …, *w*[*k*],

  where MIS(*w*[1]) ≤ MIS(*w*[2]) ≤ … ≤ MIS(*w*[*k*]).

The MSapriori algorithm

Algorithm MSapriori(*T, MS, φ*) // φ is for support difference constraint

$M \leftarrow$ sort(I, MS);

$L \leftarrow$ init-pass(M, T);

$F_1 \leftarrow \{\{i\} \mid i \in L, i.count/n \geq \text{MIS}(i)\}$;

for (k = 2; $F_{k-1} \neq \emptyset$; k++) **do**

if k=2 **then**

$C_k \leftarrow$ level2-candidate-gen(L, φ)

else $C_k \leftarrow$ MScandidate-gen(F_{k-1}, φ);

end;

for each transaction $t \in T$ **do**

for each candidate $c \in C_k$ **do**

if c is contained in t **then**

$c.count$++;

if $c - \{c[1]\}$ is contained in t **then**

$c.tailCount$++

end

end

$F_k \leftarrow \{c \in C_k \mid c.count/n \geq MIS(c[1])\}$

end

return $F \leftarrow \bigcup_k F_k$;

Candidate itemset generation:

- Special treatments needed:
  - Sorting the items according to their MIS values
  - First pass over data (the first three lines)
  - Candidate generation at level-2

```
Function level2-candidate-gen(L, φ)
1  C2 ← ∅;                                   // initialize the set of candidates
2  for each item l in L in the same order do
3     if l.count/n ≥ MIS(l) then
4        for each item h in L that is after l do
5           if h.count/n ≥ MIS(l) and |sup(h) – sup(l)| ≤ φ then
6              C2 ← C2 ∪ {{l, h}};   // insert the candidate {l, h} into C2
```

  - Pruning step in level-*k* (*k* > 2) candidate generation.

First pass over data

- It makes a pass over the data to record the support count of each item.
- It then follows the sorted order to find the first item i in M that meets MIS(i).
  - i is inserted into L.
  - For each subsequent item j in M after i, if $j.count/n \geq$ MIS(i) then j is also inserted into L, where $j.count$ is the support count of j and n is the total number of transactions in T. Why?
- L is used by function level2-candidate-gen

First pass over data: an example

- Consider the four items 1, 2, 3 and 4 in a data set. Their minimum item supports are:

  MIS(1) = 10% MIS(2) = 20%

  MIS(3) = 5% MIS(4) = 6%

- Assume our data set has 100 transactions. The first pass gives us the following support counts:

  {3}.*count* = 6, {4}.*count* = 3,

  {1}.*count* = 9, {2}.*count* = 25.

- **Then** L = {3, 2}, and F_1 = {{3}, {2}}
- Item 4 is not in L because 4.*count*/*n* < MIS(3) (= 5%),
- {1} is not in F_1 because 1.*count*/*n* < MIS(1) (= 10%).

Rule generation

- The following two lines in MSapriori algorithm are important for rule generation, which are not needed for the Apriori algorithm

  if $c - \{c[1]\}$ is contained in t **then**

  $c.tailCount$++

- Many rules cannot be generated without them.
- Why?

On multiple minsup rule mining

- Multiple minsup model subsumes the single support model.
- It is a more realistic model for practical applications.
- The model enables us to found rare item rules yet without producing a huge number of meaningless rules with frequent items.
- By setting MIS values of some items to 100% (or more), we effectively instruct the algorithms not to generate rules only involving these items.

Road map

- Basic concepts of Association Rules
- Apriori algorithm
- Different data formats for mining
- Mining with multiple minimum supports
- Mining class association rules
- Sequential pattern mining
- Summary

Mining class association rules (CAR)

- Normal association rule mining does not have any target.
- It finds all possible rules that exist in data, i.e., any item can appear as a consequent or a condition of a rule.
- However, in some applications, the user is interested in some targets.
  - E.g, the user has a set of text documents from some known topics. He/she wants to find out what words are associated or correlated with each topic.

Problem definition

- Let T be a transaction data set consisting of n transactions.
- Each transaction is also labeled with a class y.
- Let I be the set of all items in T, Y be the set of all class labels and $I \cap Y = \emptyset$.
- A **class association rule** (**CAR**) is an implication of the form

  $X \rightarrow y$, where $X \subseteq I$, and $y \in Y$.
- The definitions of **support** and **confidence** are the same as those for normal association rules.

An example

- **A text document data set**

doc 1:	Student, Teach, School	: Education
doc 2:	Student, School	: Education
doc 3:	Teach, School, City, Game	: Education
doc 4:	Baseball, Basketball	: Sport
doc 5:	Basketball, Player, Spectator	: Sport
doc 6:	Baseball, Coach, Game, Team	: Sport
doc 7:	Basketball, Team, City, Game	: Sport

- Let *minsup* = 20% and *minconf* = 60%. The following are two examples of class association rules:

Student, School $\rightarrow$ Education	[sup= 2/7, conf = 2/2]
game $\rightarrow$ Sport	[sup= 2/7, conf = 2/3]

Mining algorithm

- Unlike normal association rules, CARs can be mined directly in one step.
- The key operation is to find all **ruleitems** that have support above *minsup*. A **ruleitem** is of the form:

  (*condset*, *y*)

  where **condset** is a set of items from *I* (*i.e., condset* $\subseteq I$), and $y \in Y$ is a class label.
- Each ruleitem basically represents a rule:

  $condset \rightarrow y$,
- The Apriori algorithm can be modified to generate CARs

Multiple minimum class supports

- The multiple minimum support idea can also be applied here.
- The user can specify different minimum supports to different classes, which effectively assign a different minimum support to rules of each class.
- For example, we have a data set with two classes, Yes and No. We may want
  - rules of class Yes to have the minimum support of 5% and
  - rules of class No to have the minimum support of 10%.
- By setting minimum class supports to 100% (or more for some classes), we tell the algorithm not to generate rules of those classes.
  - This is a very useful trick in applications.

ANURAG UNIVERSITY

Thank You