MARUDHAR KESARI JAIN COLLEGE FOR WOMEN

SUBJECT NAME: DATA MINING

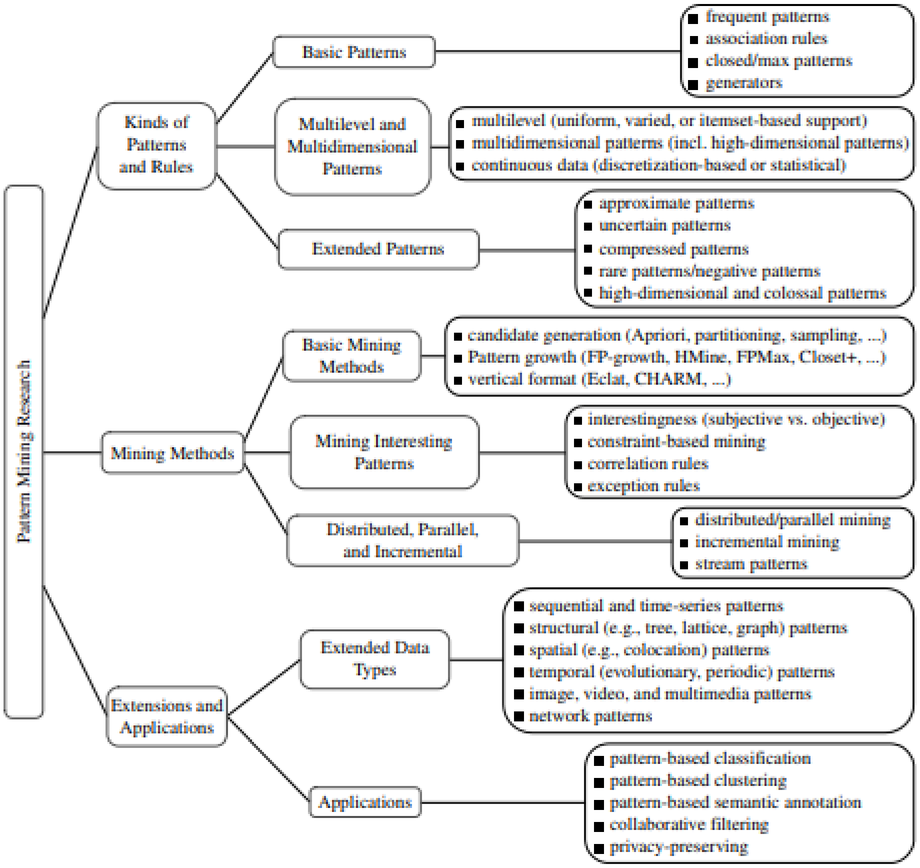
SUBJECT CODE: CECA54A

Unit-3: CONCEPTS OF PATTERN

Patterns– Basic concepts– Pattern Evaluation Methods–Pattern Mining: Pattern Mining in Multilevel– Multidimensional space–Constraint–Based Frequent Pattern Mining– Mining High Dimensional Data and Colossal patterns– Mining compressed or Approximate patterns– Pattern Exploration and Application. Classification–Decision tree Induction– Bayes Classification methods– Rule based Classification– Model Evaluation and selection– Techniques to Improve Classification Accuracy– Other Classification methods.

PATTERNS AND CLASSIFICATION

3.1 PATTERN MINING


Pattern Mining Research
Kinds of Patterns and Rules
Basic Patterns
■ frequent patterns
■ association rules
■ closed/max patterns
■ generators
Multilevel and Multidimensional Patterns
■ multilevel (uniform, varied, or itemset-based support)
■ multidimensional patterns (incl. high-dimensional patterns)
■ continuous data (discretization-based or statistical)
Extended Patterns
■ approximate patterns
■ uncertain patterns
■ compressed patterns
■ rare patterns/negative patterns
■ high-dimensional and colossal patterns
Mining Methods
Basic Mining Methods
■ candidate generation (Apriori, partitioning, sampling, ...)
■ Pattern growth (FP-growth, HMine, FPMax, Closet+, ...)
■ vertical format (Eclat, CHARM, ...)
Mining Interesting Patterns
■ interestingness (subjective vs. objective)
■ constraint-based mining
■ correlation rules
■ exception rules
Distributed, Parallel, and Incremental
■ distributed/parallel mining
■ incremental mining
■ stream patterns
Extensions and Applications
Extended Data Types
■ sequential and time-series patterns
■ structural (e.g., tree, lattice, graph) patterns
■ spatial (e.g., colocation) patterns
■ temporal (evolutionary, periodic) patterns
■ image, video, and multimedia patterns
■ network patterns
Applications
■ pattern-based classification
■ pattern-based clustering
■ pattern-based semantic annotation
■ collaborative filtering
■ privacy-preserving

Based on pattern diversity, pattern mining can be classified using the following criteria:

1. **Basic patterns: A** frequent pattern may have several alternative forms, including a simple frequent pattern, a closed pattern, or a max-pattern. To review, a **frequent pattern** is a pattern (or itemset) that satisfies a minimum support threshold. A pattern *p* is a **closed pattern** if there is no superpattern *p*0 with the same support as *p*. Pattern *p* is a **max-pattern** if there exists no frequent superpattern of *p*. Frequent patterns can also be mapped into **association rules**, or other kinds of rules based on interestingness measures. Sometimes we may also be interested in **infrequent** or **rare patterns** (i.e., patterns that occur rarely but are of critical importance, or **negative patterns** (i.e., patterns that reveal a negative correlation between items).

2.Based on the *abstraction* levels involved in a pattern: Patterns or association rules may have items or concepts residing at high, low, or multiple abstraction levels. For example, suppose that a set of association rules mined includes the following rules where *X* is a variable representing a customer:

(buys.X, “*computer*”/)   *buys.X*, “*printer* ”
(buys.X, “*laptop computer*”/)   *buys.X*, “*color laser printer* ”

In rules above the items bought are referenced at different abstraction levels (e.g., “*computer*” is a higher-level abstraction of “*laptop computer*,” and “*color laser printer*” is a lower-level abstraction of “*printer*”). We refer to the rule set mined as consisting of **multilevel association rules**. If, instead, the rules within a given set do not reference items or attributes at different abstraction levels, then the set contains **single-level association rules**.

3.Based on the *number of dimensions* involved in the rule or pattern: If the items or attributes in an association rule or pattern reference only one dimension, it is a **single-dimensional association rule/pattern**. For example, Rules and single-dimensional association rules because they each refer to only one dimension , buys. If a rule/pattern references two or more dimensions, such as *age*, *income*, and *buys*, then it is a **multidimensional association rule/pattern**. The following is an example of a multidimensional rule:

age.*X*, “20: : :29”/^*income*.(*X*, “52*K* : : :58*K*”/)   (*buys*.*X*, “*iPad* ”)

4.Based on the *types of values* handled in the rule or pattern: If a rule involves associations between the presence or absence of items, it is a **Boolean association rule**. For example, Rules are Boolean association rules obtained from market basket analysis.

If a rule describes associations between quantitative items or attributes, then it is a **quantitative association rule**. In these rules, quantitative values for items or attributes are partitioned into intervals. Rule can also be considered a quantitative association rule where the quantitative attributes *age* and *income* have been discretized.

5.Based on the *constraints* or *criteria* used to mine *selective patterns*: The patterns or rules to be discovered can be **constraint-based** (i.e., satisfying a set of userdefined constraints), **approximate**, **compressed**, **near-match** (i.e., those that tally the support count of the near or almost matching itemsets), **top-*k*** (i.e., the *k* most frequent itemsets for a user-specified value, *k*), **redundancy-aware top-*k*** (i.e., the top-*k* patterns with similar or redundant patterns excluded), and so on

6.Based on *kinds of data and features* to be mined: Given relational and data warehouse data, most people are interested in itemsets. Thus, frequent pattern mining in this context is essentially **frequent itemset mining**, that is, to mine frequent *sets of items*. However, in many other applications, patterns may involve sequences and structures. For example, by studying the order in which items are frequently purchased, we may find that customers tend to first buy a PC, followed by a digital camera, and then a memory card. This leads to **sequential patterns**, that is, frequent *subsequences* (which are often separated by some other events) in a *sequence* *of ordered events*.

7.Based on *application domain-specific semantics*: Both data and applications can be very diverse, and therefore the patterns to be mined can differ largely based on their domain-specific semantics. Various kinds of application data include spatial data, temporal data, spatiotemporal data, multimedia data (e.g., image, audio, and video data), text data, time-series data, DNA and biological sequences, software programs, chemical compound structures, web structures, sensor networks, social and information networks, biological networks, data streams, and so on. This diversity can lead to dramatically different pattern mining methodologies.

8.Based on *data analysis usages*: Frequent pattern mining often serves as an intermediate step for improved data understanding and more powerful data analysis. For example, it can be used as a feature extraction step for classification, which is often referred to as **pattern-based classification**. Similarly, **pattern-based clustering** has shown its strength at clustering high-dimensional data. For improved data understanding, patterns can be used for semantic annotation or contextual analysis. Pattern analysis can also be used in **recommender systems**, which recommend information items (e.g., books, movies, web pages) that are likely to be of interest to the user based on similar users' patterns. Different analysis tasks may require mining rather different kinds of patterns as well.

3.2 PATTERN EVALUATION METHODS

3.3 PATTERN MINING IN MULTILEVEL, MULTIDIMENSIONAL SPACE

Multilevel associations involve concepts at different abstraction levels. **Multidimensional associations** involve more than one dimension or predicate (e.g., rules that relate what a customer buys to his or her age). **Quantitative association** rules involve numeric attributes that have an implicit ordering among values (e.g., age). **Rare patterns** are patterns that suggest interesting although rare item combinations. **Negative patterns** show negative correlations between items.

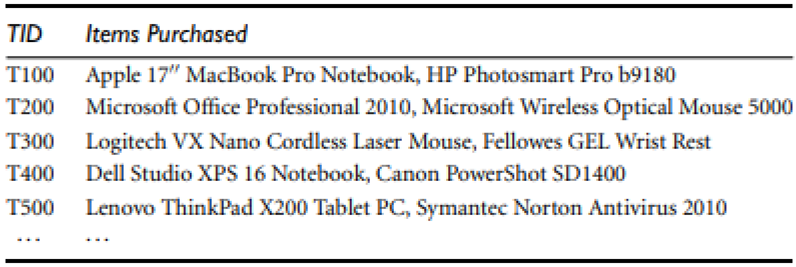
3.3.1 Mining Multilevel Associations

For many applications, strong associations discovered at high abstraction levels, though with high support, could be commonsense knowledge. We may want to drill down to find novel patterns at more detailed levels. On the other hand, there could be too many scattered patterns at low or primitive abstraction levels, some of which are just trivial specializations of patterns at higher levels. Therefore, it is interesting to examine how to develop effective methods for mining patterns at multiple abstraction levels, with sufficient flexibility for easy traversal among different abstraction spaces.

EXAMPLE:

Mining multilevel association rules. Suppose we are given the task-relevant set of transactional data in an AllElectronics store, showing the items purchased for each transaction. The concept hierarchy for the items is shown in Figure**. A concept hierarchy defines a sequence of mappings from a set of low-level concepts to a higher-level, more general concept set. Data can be generalized by replacing low-level concepts within the data by their corresponding higher-level concepts, or ancestors, from a concept hierarchy.**

Table 7.1 Task-Relevant Data, *D*



TID	*Items Purchased*
T100	Apple 17″ MacBook Pro Notebook, HP Photosmart Pro b9180
T200	Microsoft Office Professional 2010, Microsoft Wireless Optical Mouse 5000
T300	Logitech VX Nano Cordless Laser Mouse, Fellowes GEL Wrist Rest
T400	Dell Studio XPS 16 Notebook, Canon PowerShot SD1400
T500	Lenovo ThinkPad X200 Tablet PC, Symantec Norton Antivirus 2010
…	…

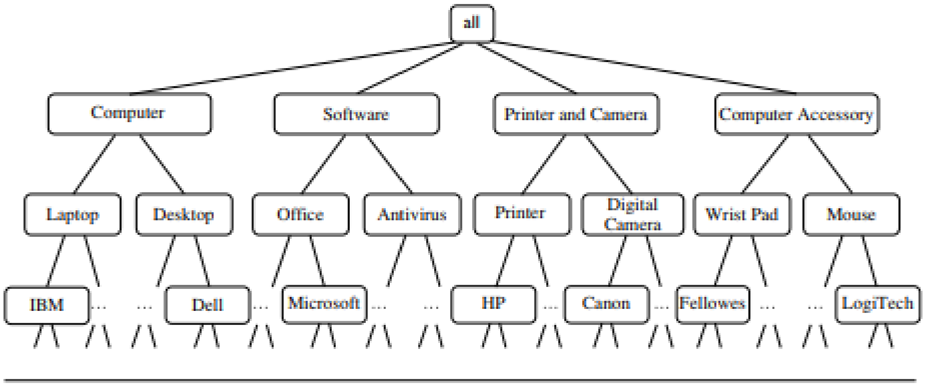
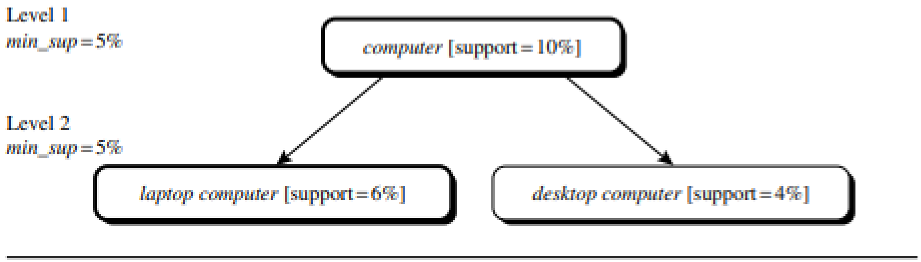


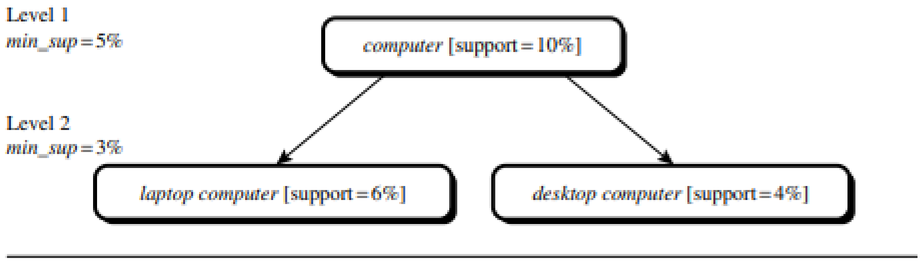
Figure 7.2 Concept hierarchy for *AllElectronics* computer items.

Using uniform minimum support for all levels: The same minimum support threshold is used when mining at each abstraction level. (e.g., for mining from “computer” downward to “laptop computer”). Both “computer” and “laptop computer” are found to be frequent, whereas “desktop computer” is not.



·e 7.3 Multilevel mining with uniform support.

Using reduced minimum support at lower levels (referred to as reduced support): Each abstraction level has its own minimum support threshold.



e 7.4 Multilevel mining with reduced support.

3.3.2 Mining Multidimensional Associations

association rules that imply a single predicate, that is, the predicate buys. For instance, in mining our AllElectronics database, we may discover the Boolean association rule

buys(X, “digital camera”) ⇒ buys(X, “HP printer”).

Following the terminology used in multidimensional databases, we refer to each distinct predicate in a rule as a dimension. Hence, we can refer to Rule as a single dimensional or intra dimensional association rule because it contains a single distinct predicate (e.g., buys) with multiple occurrences (i.e., the predicate occurs more than once within the rule). Such rules are commonly mined from transactional data.

Association rules that involve two or more dimensions or predicates can be referred to as multidimensional association rules. Rule contains three predicates (age, occupation, and buys), each of which occurs only once in the rule.

age(X, “20...29”) ∧ buys(X, “laptop”)⇒buys(X, “HP printer”).

3.3.3 Mining Quantitative Association Rules

Relational and data warehouse data often involve quantitative attributes or measures. We can discretize quantitative attributes into multiple intervals and then treat them as nominal data in association mining. However, such simple discretization may lead to the generation of an enormous number of rules, many of which may not be useful. Here we introduce three methods that can help overcome this difficulty to discover novel association relationships:

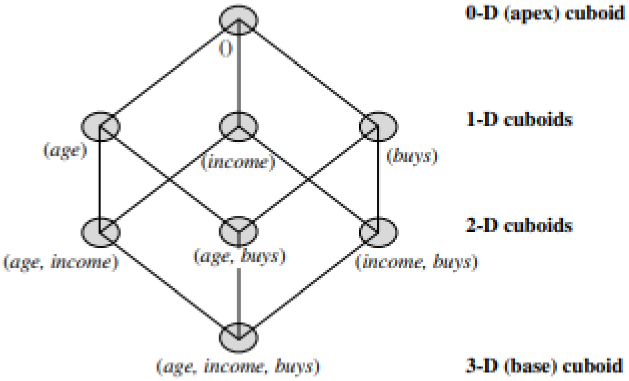
(1) a data cube method,

(2) a clustering-based method, and

(3) a statistical analysis method to uncover exceptional behaviors.

1. Data Cube–Based Mining of Quantitative Associations

The lattice of cuboids defining a data cube for the dimensions age, income, and buys. The cells of an n-dimensional cuboid can be used to store the support counts of the corresponding n-predicate sets. The base cuboid aggregates the task-relevant data by age, income, and buys; the 2-D cuboid, (age, income), aggregates by age and income, and so on; the 0-D (apex) cuboid contains the total number of transactions in the task-relevant data.



3.3.4 Mining Rare Patterns and Negative Patterns

All the methods presented so far in this chapter have been for mining frequent patterns. Sometimes, however, it is interesting to find patterns that are rare instead of frequent, or patterns that reflect a negative correlation between items. These patterns are respectively referred to as rare patterns and negative patterns. In this subsection, we consider various ways of defining rare patterns and negative patterns, which are also useful to mine

EXAMPLE: Rare patterns and negative patterns. In **jewelry sales data**, sales of diamond watches are rare; however, patterns involving the selling of diamond watches could be interesting. In supermarket data, if we find that customers frequently buy Coca-Cola Classic or Diet Coke but not both, then buying Coca-Cola Classic and buying Diet Coke together is considered a negative (correlated) pattern.

An **infrequent (or rare) pattern** is a pattern with a frequency support that is below (or far below) a user-specified minimum support threshold. However, since the occurrence frequencies of the majority of itemsets are usually below or even far below the minimum support threshold, it is desirable in practice for users to specify other conditions for rare patterns. For example, if we want to find patterns containing at least one item with a value that is over $500.

3.4 CONSTRAINT-BASED FREQUENT PATTERN MINING

A data mining process may uncover thousands of rules from a given data set, most of which end up being unrelated or uninteresting to users. Often, users have a good sense of which "direction" of mining may lead to interesting patterns and the "form" of the patterns or rules they want to find. They may also have a sense of "conditions" for the rules, which would eliminate the discovery of certain rules that they know would not be of interest. Thus, a good heuristic is to have the users specify such intuition or expectations as constraints to confine the search space. This strategy is known as constraint-based mining. The constraints can include the following:

Knowledge type constraints: These specify the type of knowledge to be mined, such as association, correlation, classification, or clustering.

Data constraints: These specify the set of task-relevant data.

Dimension/level constraints: These specify the desired dimensions (or attributes) of the data, the abstraction levels, or the level of the concept hierarchies to be used in mining.

Interestingness constraints: These specify thresholds on statistical measures of rule interestingness such as support, confidence, and correlation.

Rule constraints: These specify the form of, or conditions on, the rules to be mined. Such constraints may be expressed as metarules (rule templates), as the maximum or minimum number of predicates that can occur in the rule antecedent or consequent, or as relationships among attributes, attribute values, and/or aggregates.

3.4.1 Metarule-Guided Mining of Association Rules

Metarules allow users to specify the syntactic form of rules that they are interested in mining. The rule forms can be used as constraints to help improve the efficiency of the mining process. Metarules may be based on the analyst's experience, expectations, or intuition regarding the data or may be automatically generated based on the database schema.

EXAMPLE: Metarule-guided mining. Suppose that as a market analyst for AllElectronics you have access to the data describing customers (e.g., customer age, address, and credit rating) as well as the list of customer transactions. You are interested in finding associations between customer traits and the items that customers buy. However, rather than finding all of the association rules reflecting these relationships, you are interested only in determining which pairs of customer traits promote the sale of office software. A metarule can be used to specify this information describing the form of rules you are interested in finding. An example of such a metarule is

P1(X, Y) ∧ P2(X, W) ⇒ buys(X, "office software"),

where P1 and P2 are predicate variablesthat are instantiated to attributes from the given database during the mining process, X is a variable representing a customer, and Y and W take on values of the attributes assigned to P1 and P2, respectively. Typically, a user will specify a list of attributes to be considered for instantiation with P1 and P2. Otherwise, a default set may be used.

3.4.2 Constraint-Based Pattern Generation: Pruning Pattern Space and Pruning Data Space

Rule constraints specify expected set/subset relationships of the variables in the mined rules, constant initiation of variables, and constraints on aggregate functions and other forms of constraints. Users typically employ their knowledge of the application or data to specify rule constraints for the mining task. These rule constraints may be used together with, or as an alternative to, metarule-guided mining. In this section, we examine rule constraints as to how they can be used to make the mining process more efficient. Let's study an example where rule constraints are used to mine hybrid-dimensional association rules.

EXAMPLE: Constraints for mining association rules. Suppose that AllElectronics has a sales multidimensional database with the following interrelated relations:

item(item ID, item name, description, category, price)

sales(transaction ID, day, month, year, store ID, city)

trans item(item ID, transaction ID)

Here, the item table contains attributes item ID, item name, description, category, and price; the sales table contains attributes transaction ID day, month, year, store ID, and city; and the two tables are linked via the foreign key attributes, item ID and transaction ID, in the table trans item.

Suppose our association mining query is "Find the patterns or rules about the sales of which cheap items (where the sum of the prices is less than $10) may promote (i.e., appear in the same transaction) the sales of which expensive items (where the minimum price is $50), shown in the sales in Chicago in 2010."

3.5 MINING HIGH-DIMENSIONAL DATA AND COLOSSAL PATTERNS

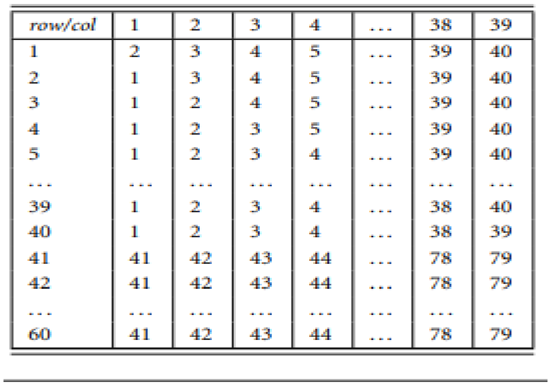
The frequent pattern mining methods presented so far handle large data sets having a small number of dimensions. However, some applications may need to mine highdimensional data (i.e., data with hundreds or thousands of dimensions). Can we use the methods studied so far to mine high-dimensional data? The answer is unfortunately negative because the search spaces of such typical methods grow exponentially with the number of dimensions.

Researchers have overcome this difficulty in two directions. One direction extends a pattern growth approach by further exploring the vertical data format to handle data sets with a large number of dimensions (also called features or items, e.g., genes) but a small number of rows (also called transactions or tuples, e.g., samples). This is useful in applications like the analysis of gene expressions in bioinformatics, for example, where we often need to analyze microarray data that contain a large number of genes (e.g., 10,000 to 100,000) but only a small number of samples (e.g., 100 to 1000). The other direction develops a new mining methodology, called Pattern-Fusion, which mines colossal patterns, that is, patterns of very long length.

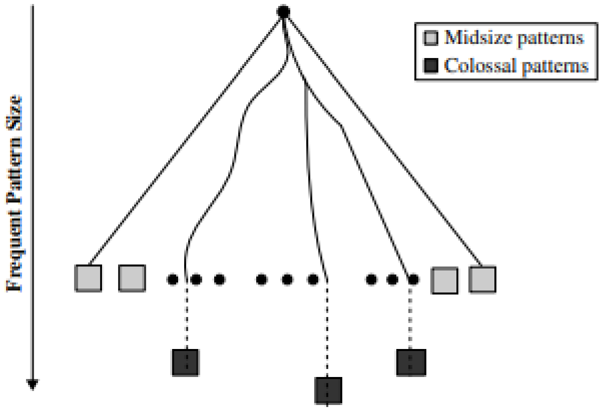
3.5.1 Mining Colossal Patterns by Pattern-Fusion

Although we have studied methods for mining frequent patterns in various situations, many applications have hidden patterns that are tough to mine, due mainly to their immense length or size. Consider bioinformatics, for example, where a common activity is DNA or microarray data analysis. This involves mapping and analyzing very long DNA and protein sequences. Researchers are more interested in finding large patterns (e.g., long sequences) than finding small ones since larger patterns usually carry more significant meaning. We call these large patterns colossal patterns, as distinguished from patterns with large support sets. Finding colossal patterns is challenging because incremental mining tends to get “trapped” by an explosive number of midsize patterns before it can even reach candidate patterns of large size.

EXAMPLE: The challenge of mining colossal patterns. Consider a 40 × 40 square table where each row contains the integers 1 through 40 in increasing order. Remove the integers on the diagonal, and this gives a 40 × 39 table. Add 20 identical rows to the bottom of the table, where each row contains the integers 41 through 79 in increasing order, resulting in a 60 × 39 table . We consider each row as a transaction and set the minimum support threshold at 20. The table has an exponential number (i.e., 40 20) of midsize closed/maximal frequent patterns of size 20, but only one that is colossal: $\alpha = (41,42,...,79)$ of size 39. None of the frequent pattern mining algorithms that we have introduced so far can complete execution in a reasonable amount of time.



row/col	1	2	3	4	...	38	39
1	2	3	4	5	...	39	40
2	1	3	4	5	...	39	40
3	1	2	4	5	...	39	40
4	1	2	3	5	...	39	40
5	1	2	3	4	...	39	40
...	...	...	...	...	...	...	...
39	1	2	3	4	...	38	40
40	1	2	3	4	...	38	39
41	41	42	43	44	...	78	79
42	41	42	43	44	...	78	79
...	...	...	...	...	...	...	...
60	41	42	43	44	...	78	79



3.6 MINING COMPRESSED OR APPROXIMATE PATTERNS:

A major challenge in frequent pattern mining is the huge number of discovered patterns.

Using a minimum support threshold to control the number of patterns found has limited effect. Too low a value can lead to the generation of an explosive number of output patterns, while too high a value can lead to the discovery of only commonsense patterns.

To reduce the huge set of frequent patterns generated in mining while maintaining high-quality patterns, we can instead mine a compressed or approximate set of frequent patterns. Top-k most frequent closed patterns were proposed to make the mining process concentrate on only the set of k most frequent patterns. Although interesting, they usually do not epitomize the k most representative patterns because of the uneven frequency distribution among itemsets. Constraint-based mining of frequent patterns incorporates user-specified constraints to filter out uninteresting patterns.

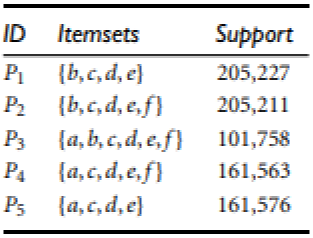
3.6.1 Mining Compressed Patterns by Pattern Clustering

Pattern compression can be achieved by pattern clustering. Clustering techniques are described in detail in Chapters 10 and 11. In this section, it is not necessary to know the fine details of clustering. Rather, you will learn how the concept of clustering can be applied to compress frequent patterns. Clustering is the automatic process of grouping like objects together, so that objects within a cluster are similar to one another and dissimilar to objects in other clusters. In this case, the objects are frequent patterns. The frequent patterns are clustered using a tightness measure called δ-cluster. A representative pattern is selected for each cluster, thereby offering a compressed version of the set of frequent patterns.

Before we begin, let's review some definitions. An itemset X is a closed frequent itemset in a data set D if X is frequent and there exists no proper super-itemset Y of X such that Y has the same support count as X in D. An itemset X is a maximal frequent itemset in data set D if X is frequent and there exists no super-itemset Y such that X ⊂ Y and Y is frequent in D.

EXAMPLE: Shortcomings of closed itemsets and maximal itemsets for compression. Table shows a subset of frequent itemsets on a large data set, where a, b, c, d, e, f represent individual items. There are no closed itemsets here; therefore, we cannot use closed frequent itemsets to compress the data. The only maximal frequent itemset is P3. However, we observe that itemsets P2, P3, and P4 are significantly different with respect to their support counts. If we were to use P3 to represent a compressed version of the data, we would lose this support count information entirely. From visual inspection, consider the two pairs (P1, P2) and (P4, P5). The patterns within each pair are very similar with respect to their support and expression. Therefore, intuitively, P2, P3, and P4, collectively, should serve as a better compressed version of the data.

Subset of Frequent Itemsets



ID	*Itemsets*	*Support*
P_1	$\{b, c, d, e\}$	205,227
P_2	$\{b, c, d, e, f\}$	205,211
P_3	$\{a, b, c, d, e, f\}$	101,758
P_4	$\{a, c, d, e, f\}$	161,563
P_5	$\{a, c, d, e\}$	161,576

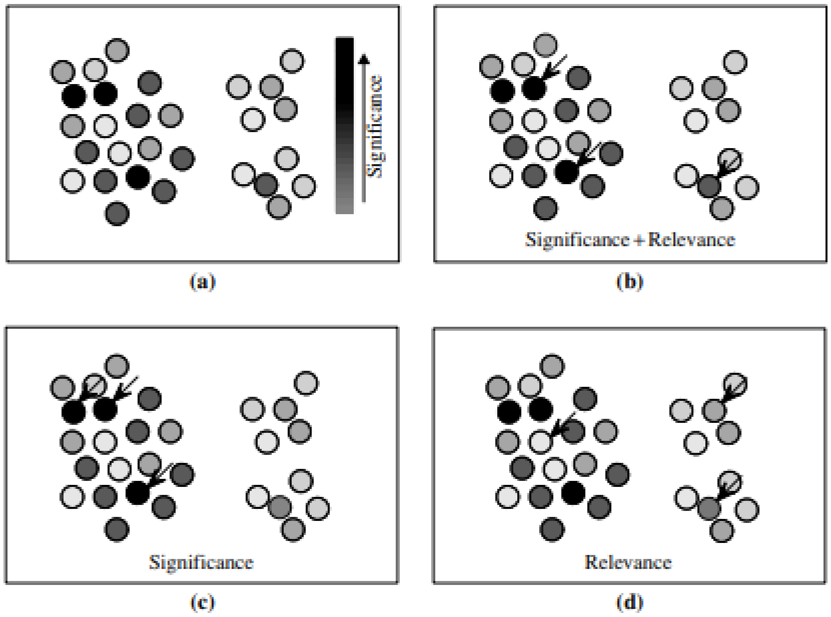
So, let's see if we can find a way of clustering frequent patterns as a means of obtaining a compressed representation of them. We will need to define a good similarity measure, cluster patterns according to this measure, and then select and output only a representative pattern for each cluster. Since the set of closed frequent patterns is a lossless compression over the original frequent patterns set, it is a good idea to discover representative patterns over the collection of closed patterns.

3.6.2 Extracting Redundancy-Aware Top-k Patterns

Mining the top-k most frequent patterns is a strategy for reducing the number of patterns returned during mining. However, in many cases, frequent patterns are not mutually independent but often clustered in small regions. This is somewhat like finding 20 population centers in the world, which may result in cities clustered in a small number of countries rather than evenly distributed across the globe. Instead, most users would prefer to derive the k most interesting patterns, which are not only significant, but also mutually independent and containing little redundancy. A small set of k representative patterns that have not only high significance but also low redundancy are called redundancy-aware top-k patterns.

EXAMPLE: Redundancy-aware top-k strategy versus other top-k strategies. Figure 7.11 illustrates the intuition behind redundancy-aware top-k patterns versus traditional top-k patterns and k-summarized patterns. Suppose we have the frequent patterns set shown in Figure 7.11(a), where each circle represents a pattern of which the significance is colored in grayscale. The distance between two circles reflects the redundancy of the two corresponding patterns: The closer the circles are, the more redundant the respective patterns are to one another. Let’s say we want to find three patterns that will best represent the given set, that is, k = 3. Which three should we choose?

Arrows are used to show the patterns chosen if using redundancy-aware top-k patterns (Figure b), traditional top-k patterns (Figure c), or k-summarized patterns (Figure d). In Figure (c), the traditional top-k strategy relies solely on significance: It selects the three most significant patterns to represent the set.



In Figure (d), the k-summarized pattern strategy selects patterns based solely on nonredundancy. It detects three clusters, and finds the most representative patterns to be the “centermost’” pattern from each cluster. The selected patterns are considered “summarized patterns” in the sense that they represent or "provide a summary" of the clusters they stand for. By contrast, in Figure (d) the redundancy-aware top-k patterns make a trade-off between significance and redundancy.

3.7 PATTERN EXPLORATION AND APPLICATION

The automated generation of semantic annotations for frequent patterns. These are dictionary-like annotations. They provide semantic information relating to patterns, based on the context and usage of the patterns, which aids in their understanding. Semantically similar patterns also form part of the annotation, providing a more direct connection between discovered patterns and any other patterns already known to the users.

3.7.1 Semantic Annotation of Frequent Patterns

Pattern mining typically generates a huge set of frequent patterns without providing enough information to interpret the meaning of the patterns. In the previous section, we introduced pattern processing techniques to shrink the size of the output set of frequent patterns such as by extracting redundancy-aware top-k patterns or compressing the pattern set. These, however, do not provide any semantic interpretation of the patterns. It would be helpful if we could also generate semantic annotations for the frequent patterns found, which would help us to better understand the patterns.

"What is an appropriate semantic annotation for a frequent pattern?" Think about what we find when we look up the meaning of terms in a dictionary. Suppose we are looking up the term pattern. A dictionary typically contains the following components to explain the term:

1. A set of definitions, such as "a decorative design, as for wallpaper, china, or textile fabrics, etc.; a natural or chance configuration"

2. Example sentences, such as "patterns of frost on the window; the behavior patterns of teenagers, . . .

3. Synonyms from a thesaurus, such as "model, archetype, design, exemplar, motif, ..."

3.7.2 Applications of Pattern Mining

Pattern mining is widely used for **noise filtering and data cleaning as preprocessing** in many data-intensive applications. We can use it to analyze microarray data, for instance, which typically consists of tens of thousands of dimensions (e.g., representing genes). Such data can be rather noisy. Frequent pattern data mining can help us distinguish between what is noise and what isn't. We may assume that items that occur frequently together are less likely to be random noise and should not be filtered out. On the other hand, those that occur very frequently (similar to stopwords in text documents) are likely indistinctive and may be filtered out. Frequent pattern mining can help in background information identification and noise reduction.

Pattern mining often helps in the **discovery of inherent structures and clusters hidden in the data**. Given the DBLP data set, for instance, frequent pattern mining can easily find interesting clusters like coauthor clusters (by examining authors who frequently collaborate) and conference clusters (by examining the sharing of many common authors and terms). Such structure or cluster discovery can be used as preprocessing for more sophisticated data mining.

Although there are numerous classification methods research has found that frequent patterns can be used as building blocks in the construction of highquality classification models, hence called **pattern-based classification.** The approach is successful because (1) the appearance of very infrequent item(s) or itemset(s) can be caused by random noise and may not be reliable for model construction, yet a relatively frequent pattern often carries more information gain for constructing more reliable models; (2) patterns in general (i.e., itemsets consisting of multiple attributes) usually carry more information gain than a single attribute (feature); and (3) the patterns so generated are often intuitively understandable and easy to explain. Recent research has reported several methods that mine interesting, frequent, and discriminative patterns and use them for effective classification.

Frequent patterns can also be used effectively for subspace **clustering in highdimensional space**. Clustering is challenging in high-dimensional space, where the distance between two objects is often difficult to measure. This is because such a distance is dominated by the different sets of dimensions in which the objects are residing.

Pattern analysis is useful in the analysis of **spatiotemporal data, time-series data, image data, video data, and multimedia data.** An area of spatiotemporal data analysis is the discovery of colocation patterns. These, for example, can help determine if a certain disease is geographically colocated with certain objects like a well, a hospital, or a river.

In **time-series data analysis**, researchers have discretized time-series values into multiple intervals (or levels) so that tiny fluctuations and value differences can be ignored. The data can then be summarized into sequential patterns, which can be indexed to facilitate similarity search or comparative analysis.

In **image analysis and pattern recognition**, researchers have also identified frequently occurring visual fragments as “visual words,” which can be used for effective clustering, classification, and comparative analysis.

Pattern mining has also been used for the **analysis of sequence or structural data** such as trees, graphs, subsequences, and networks. In software engineering, researchers have identified consecutive or gapped subsequences in program execution as sequential patterns that help identify software bugs. Copy-and-paste bugs in large software programs can be identified by extended sequential pattern analysis of source programs. Plagiarized software programs can be identified based on their essentially identical program flow/loop structures. Authors’ commonly used sentence substructures can be identified and used to distinguish articles written by different authors.

Frequent and discriminative patterns can be used as primitive **indexing structures** (known as graph indices) to help search large, complex, structured data sets and networks. These support a similarity search in graph-structured data such as chemical compound databases or XML-structured databases. Such patterns can also be used for data compression and summarization.

frequent patterns have been used in **recommender systems**, where people can find correlations, clusters of customer behaviors, and classification models based on commonly occurring or discriminative patterns.

Finally, studies on efficient computation methods in pattern mining mutually enhance many other studies on **scalable computation**. For example, the computation and materialization of iceberg cubes using the BUC and Star-Cubing algorithms respectively share many similarities to computing frequent patterns by the Apriori and FP-growth algorithms.