Extraction of textual information

University of Ghardaia

Dr. Degha Houssem Eddine
A doctor in computer science.

2024

1 *First Edition.*

Contents

CHAPTER 1

Module Information

1.1 Identification of the Faculty and the Program

- Faculty: Science of Technology
- Department: Mathematics and Computer Science
- Target audience: Master 2, Specialty: Intelligent Systems for Knowledge Extraction.

Master Title: Intelligent Systems for Knowledge Extraction

1.2 Course Identification

- Course title: Extraction of Textual Information.
- Credit: 5
- Coefficient: 3

1.3 Professor Name

Dr. Degha Houssem Eddine. A doctor in computer science, software engineer, artificial intelligence researcher, and data science expert.

1.4 Contact Details

By email at: `degha.houssem@outlook.com`, `hdegha@gmail.com`

1.5 Professor Availability

- Answer on the forum: Any question related to the course must be posted on the dedicated forum, so that all can benefit from my answer. I promise to answer the questions posted within 48 hours.

- By email: I agree to respond by email within 48 hours of receiving the message, except in the event of unforeseen circumstances. I draw your attention that the preferred communication channel is the forum; email is reserved for "emergencies" (in the event of a platform access problem) and must be used with discernment.

1.6 Learning Objectives

This course provides an overview of Information Mining technologies as an important field in the text mining process. It involves transforming unstructured or semi-structured collections of texts into an ordered collection of data.

During the courses, the students will learn:

- Architectures of Information Extraction Systems
- Knowledge-based Models vs. Probabilistic Models
- Recognition of Named Entities and Classification
- Coreference Resolution
- Recognition of Temporal Expressions and Normalization
- Pattern Extraction

Evaluation Method: Continuous assessment and examination

Recommended Prerequisites: Pattern Recognition 1, Data Analysis

References:

- *Mining the Social Web* by Russel, Matthew A., O'Reilly, 2011.
- *Named Entities: Recognition, Classification, and Use* by Sekine, Satoshi, John Benjamins Publishing Company, 2009.
- *The Text Mining Handbook: Advanced Approaches in Analyzing Unstructured Data* by Feldman, Ronen, Cambridge University Press, 2007.

CHAPTER 2

Intoroduction to Information Extraction Systems

2.1 Defining the Text Mining

Text Minng can be defined as process that user take to intensive knowledges by interacting with document collection using the approperate techniques and analysis tools.

The Text mining is analogouse to data mining in manner of extract useful information from data sources through the indetification and exploration of interesting patters. In Text Mining the Data source are document collections. in addition, they are usely unstructured textual data. in constrast. The data minin data are structured or semi-structurd in form of databases or data-sets.

In text mining systems, the preprocessing operation focus on the identification and extraction of representative feature for natural language documents. Further, they are resposible for transforming unstructured data stored in document collaction into explicity stuctured intermediate format. in contrast, for data mining the preprocessing focus on two critical taks: scrubbing and normalazing data.

the text mining exploit teachniques and methodologies from the areas of information retrival, information extraction and corpus-based computational linguistics.

2.2 The text mining Document Collection

The document collection in text mining can be any grouping of text-based documents. the most text mining solotion aime to discover patterns across. The text cam be a millions of strings. Text document collection can be static or dynamic. The dynamic term means that the text change over time to new or will updated text. This change create many new challanges for various componets of the text mining system.

one of the famouse data source of document collection is PubMed. its the National library of Medicine online repository of citation related information for biomedical research paper. it includes more then 12 million research paper on topics in the life sciences.

2.3 The Document

In Text mining the Document is a unit of discret textual data from the document collection. for example it can be an e-mail, research paper or news. the document can laso be a member of diffrent document collections, or diffrent subset of the same document collection. If we see it from the perspective of linguistic, we can consider the text as a structured objct. even the document includes a rish ammount of semantic and syntatical stucture. although this stucture is implicit and it hide a dgree of knowlegdes. Despite the case when we can have misleading lable in documents that bears as unstructured data. The documents can be sometime weakly studtured or semistructured. Ther are many examples of the semistructured such as : HTML document or an e-mail.

2.4 Document Features

The document includes defferent elements such as: words, phrases, semtemces, typographical elements and even layout artifacts that may exist. The preprocessing operation that support text mining attempt to transform those elements from an irregular and implicitly structure representaion into an explicitly structured representaion. The essential task of text mining systems is the identification of a simplified of documents features that can be used to represent a pacticular document as a whole. The set of features to represent docummnet called representational model. Usaly, the number of feature is large number. and this exceedingly large number of features affects almost every aspect of a text mining system approach, design and performence. The features can simply looks like the columns of an data-set.

Commonly Used Document Features

text mining algorithms operate on the feature-based representations of documents and not the underlying documents themselves, there is often a trade- off between two important goals. The first goal is to achieve the correct calibration of the volume and semantic level of features to portray the meaning of a document accurately, which tends to incline text mining preprocessing operations toward selecting or extracting relatively more features to represent documents. The second goal is to identify features in a way that is most computationally efficient and practical for pattern discovery, which is a process that emphasizes the streamlining of repre- sentative feature sets; such streamlining is sometimes supported by the validation, normalization, or cross-referencing of features against controlled vocabularies or external knowledge sources such as dictionaries, thesauri, ontologies, or knowledge bases to assist in generating smaller representative sets of more semantically rich features.

Although many potential features can be employed to represent documents,1 the following four types are most commonly used:

Characters

The individual component-level letters, numerals, special characters and spaces are the building blocks of higher-level semantic features such as words, terms, and concepts.

Acharacter-level representation can include the full set of all characters for a document or some filtered subset. Character-based representations without positional information (i.e., bag-of-characters approaches) are often ofvery limited utility in text mining applications. Character-based representations that include some level of positional information (e.g., bigrams or trigrams) are

somewhat more useful and common.

In general, however, character-based rep- resentations can often be unwieldy for some types of text processing techniques because the feature space for a document is fairly unoptimized. On the other hand, this feature space can in many ways be viewed as the most complete of any representation of a real-world text document.

Words

Specific words selected directly from a "native" document are at what might be described as the basic level of semantic richness. For this reason, word- level features are sometimes referred to as existing in the native feature space of a document.

In general, a single word-level feature should equate with, or have the value of, no more than one linguistic token. Phrases, multiword expressions, or even multiword hyphenates would not constitute single word-level features.

It is possible for a word-level representation of a document to include a feature for each word within that document – that is the "full text," where a document is represented by a complete and unabridged set of its word-level features. This can lead to some word-level representations of document collections having tens or even hundreds of thousands of unique words in its feature space.

However, most word-level document representations exhibit at least some minimal optimization and therefore consist of subsets of representative features filtered for items such as stop words, symbolic characters, and meaningless numerics.

Terms

Terms are single words and multiword phrases selected directly from the corpus of a native document by means of term-extraction methodologies. Term-level features, in the sense of this definition, can only be made up of specific words and expressions found within the native document for which they are meant to be generally representative. Hence, a term-based representation of a document is necessarily composed of a subset of the terms in that document. For example, if a document contained the sentence

President Abraham Lincoln experienced a career that took him from log cabin to White House,

a list of terms to represent the document could include single word forms such as "Lincoln," "took," "career," and "cabin" as well as multiword forms like "President Abraham Lincoln," "log cabin," and "White House."

Several of term-extraction methodologies can convert the raw text of a native document into a series of normalized terms – that is, sequences of one or more tokenized and lemmatized word forms associated with part-of-speech tags. Some- times an external lexicon is also used to provide a controlled vocabulary for term normalization.

Term-extraction methodologies employ various approaches for generating and filtering an abbreviated list of most meaningful candidate terms from among a set of normalized terms for

the representation of a document. This culling process results in a smaller but relatively more semantically rich document representation than found in word-level document representations.

Concepts

Concepts are features generated for a document by means of manual, statistical, rule-based, or hybrid categorization methodologies. Concept-level features can be manually generated for documents but are now more commonly extracted from documents using complex preprocessing routines that identify single words, multiword expressions, whole clauses, or even larger syntactical units that are then related to specific concept identifiers. For instance, a document collection that includes reviews of sports cars may not actually include the specific word "automotive" or the specific phrase "test drives," but the concepts "automotive" and "test drives" might nevertheless be found among the set of concepts used to to identify and represent the collection.

Many categorization methodologies involve a degree of cross-referencing against an external knowledge source; for some statistical methods, this source might simply be an annotated collection of training documents. For manual and rule-based categorization methods, the cross-referencing and validation of prospective concept-level features typically involve interaction with a "gold standard" such as a preexisting domain ontology, lexicon, or formal concept hierarchy – or even just the mind of a human domain expert. Unlike word- and term-level features, concept-level features can consist of words not specifically found in the native document.

Concept-level representations, however, are much better than any other featureset representation at handling synonymy and polysey and mare clearly best at relat- ing a given feature to its various hyponyms and hypernyms.

- Concept-based representations can be processed to support very sophisticated **concept hierarchies**, and **arguably provide the best representations for leveraging the domain knowledge afforded by ontologies and knowledge bases**.
- concept-level representations do have a few potential drawbacks. Possible disadvantages of using concept-level features to represent documents include **(a) the relative complexity of applying the heuristics, during preprocessing opera- tions, required to extract and validate concept-type features** and **(b) the domain-dependence of many concepts**

Concept-level document representations generated by categorization are often stored in vector formats. For instance, both CDM-based methodologies and Los Alamos II–type concept extraction approaches result in individual documents being stored as vectors.

Hybrid approaches to the generation of feature-based document representations can exist. hybrid approaches, however, need careful planning, testing, and optimization to avoid having dramatic – and extremely resourceintensive – growth in the feature dimensionality of individual document representations without proportionately increased levels of system effectiveness.

2.5 Domains and Background Knowledge

In text mining systems, concepts belong not only to the descriptive attributes of a particular document but generally also to domains. With respect to text mining, a domain has come to

be loosely defined as a specialized area of interest for which dedicated ontologies, lexicons, and taxonomies of information may be developed.

Text mining systems with some element of domain-specificity in their orientation – that is, most text mining systems designed for a practical purpose – can leverage information from formal external knowledge sources for these domains to greatly enhance elements of their preprocessing, knowledge discovery, and presentation- layer operations. Domain knowledge, perhaps more frequently referred to in the literature as

background knowledge, can be used in text mining preprocessing operations to enhance concept extraction and validation activities. Access to background knowl- edge – although not strictly necessary for the creation of concept hierarchies within the context of a single document or document collection – can play an important role in the development of more meaningful, consistent, and normalized concept hierarchies.

By relating features by way of lexicons and ontologies, advanced text mining systems can create fuller representations of document collections in preprocessing operations and support enhanced query and refinement functionalities.

2.6 The Search for Patterns and Trends Although

Although text mining preprocessing operations play the critical role of trans- forming unstructured content of a raw document collection into a more tractable concept-level data representation, the core functionality of a text mining system resides in the analysis of concept co-occurrence patterns across documents in a col- lection. Indeed, text mining systems rely on algorithmic and heuristic approaches to consider distributions, frequent sets, and various associations of concepts at an interdocument level in an effort to enable a user to discover the nature and relation- ships of concepts as reflected in the collection as a whole.

In another example, a potential relationship might be inferred between two proteins P1 and P2 by the pattern of (a) several articles mentioning the protein P1 in relation to the enzyme E1, (b) a few articles describing functional similarities between enzymes E1 and E2 without referring to any protein names, and (c) several articles linking enzyme E2 to protein P2. In all three of these examples, the information is not provided by any single document but rather from the totality of the collection. Text mining's methods of pattern analysis seek to discover co-occurrence relationships between concepts as reflected by the totality of the corpus at hand.

Text mining methods – often based on large-scale, brute-force search directed at large, high-dimensionality feature sets – generally produce very large numbers of patterns. This results in an overabundance problem with respect to identified patterns that is usually much more severe than that encountered in data mining applications aimed at structured data sources.

Amain operational task for text mining systems is to enable a user to limit pattern over-abundance by providing refinement capabilities that key on various specifiable measures of "interestingness" for search results. Such refinement capabilities prevent system users from getting overwhelmed by too many uninteresting results.

The problem of pattern overabundance can exist in all knowledge discovery activ- ities. It is simply heightenedwhen interacting with large collections oftext documents, and, therefore, text

mining operations must necessarily be conceived to provide not only relevant but also manageable result sets to a user.

Text mining also builds on various data mining approaches first specified in Lent, Agrawal, and Srikant (1997) to identify trends in data. In text mining, trend analysis relies on date-and-time stamping of documents within a collection so that comparisons can be made between a subset of documents relating to one period and a subset of documents relating to another.

Trend analysis across document subsets attempts to answer certain types of questions. For instance, in relation to a collection of news stories, Montes-y-Gomez, Gelbukh, and Lopez-Lopez (2001b) suggests that trend analysis concerns itself with questions such as the following:

- What is the general trend of the news topics between two periods (as represented by two different document subsets)?
- Are the news topics nearly the same or are they widely divergent across the two periods?
- Can emerging and disappearing topics be identified?
- Did any topics maintain the same level ofoccurrence during the two periods?

In these illustrative questions, individual "news topics" can be seen as specific concepts in the document collection. Different types of trend analytics attempt to com- pare the frequencies of such concepts (i.e., number of occurrences) in the documents that make up the two periods' respective document subcollections. Additional types of analysis, also derived from data mining, that can be used to support trend analysis are ephemeral association discovery and deviation detection.

2.7 The Importance of the Presentation Layer

Perhaps the key presentation layer functionality supported by text mining systems is browsing. Most contemporary text mining systems support browsing that is both dynamic and content-based, for the browsing is guided by the actual textual content of a particular document collection and not by anticipated or rigorously prespecified structures. Commonly, user browsing is facilitated by the graphical presentation of concept patterns in the form of a hierarchy to aid interactivity by organizing concepts for investigation.

Browsing is also navigational. Text mining systems confront a user with extremely large sets of concepts obtained from potentially vast collections of text documents. Consequently, text mining systems must enable a user to move across these concepts in such a way as to always be able to choose either a "big picture" view of the collection in toto or to drill down on specific – and perhaps very sparsely identified – concept relationships.

Visualization tools are often employed by text mining systems to facilitate navigation and exploration of concept patterns. These use various graphical approaches to express complex data relationships. In the past, visualization tools for text min- ing sometimes generated static maps or graphs that were essentially rigid snapshots of patterns or carefully generated reports displayed on the screen or printed by an attached printer. State-of-the-art text mining systems, however, now increasingly rely on highly interactive graphic representations of search results that permit a user to drag, pull, click, or otherwise directly interact with the graphical representation

of concept patterns.

Several additional types of functionality are commonly supported within the front ends of text mining systems. Because, in many respects, the presentation layer of a text mining system really serves as the front end for the execution of the system's core knowledge discovery algorithms, considerable attention has been focused on provid- ing users with friendlier and more powerful methods for executing these algorithms. Such methods can become powerful and complex enough to necessitate developing dedicated query languages to support the efficient parameterization and execution of specific types of pattern discovery queries.

Furthermore, text mining front endsmayoffer a user the ability to cluster concepts through a suite of clustering tools (discussed in Chapter V) in ways that make the most cognitive sense for a particular application or task. Text mining systems can also allow a user to create customized profiles for concepts or concept relationships to produce a richer knowledge environment for interactive exploration.

Finally, some text mining systems offer users the ability to manipulate, create, or concatenate refinement constraints to assist in producing more manageable and useful result sets for browsing. Like other aspects relating to the creation, shaping, and parameterization of queries, the use of such refinement constraints can be made much more user-friendly by incorporating graphical elements such as pull-downs, radio boxes, or context- or query-sensitive pick lists.

CHAPTER 3

Architectures of Information Extraction Systems

At an abstract level, a text mining system takes in input (raw documents) and generates various types of output (e.g., patterns, maps of connections, trends). Figure 3.1 illustrates this basic paradigm.

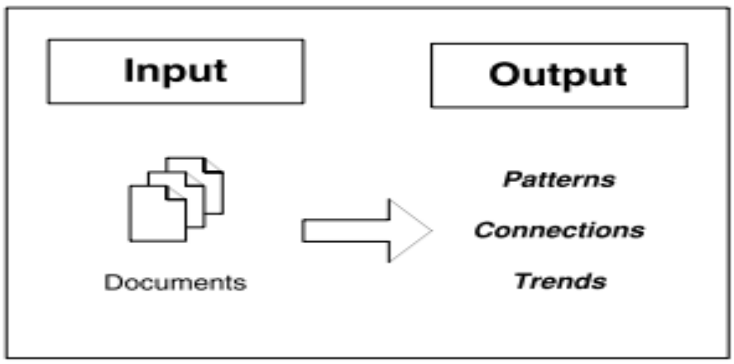

fig:arch1

Figure 3.1: Simple input–output model for text mining.

A human-centered view of knowledge discovery, however, yields a slightly more complex input–output paradigm for text mining (see Figure I.3). This paradigm is one in which a user is part of what might be seen as a prolonged interactive loop of querying, browsing, and refining, resulting in answer sets that, in turn, guide the user toward new iterative series of querying, browsing, and refining actions

3.1 Functional Architecture

On a functional level, text mining systems follow the general model provided by some classic data mining applications and are thus roughly divisible into four main areas:

- preprocessing tasks
- core mining operations
- presentation layer components and browsing functionality
- refinement techniques

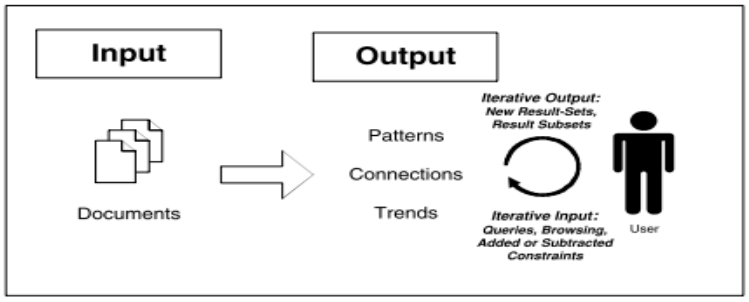

fig:arch2

Figure 3.2: Iterative loop for user input and output.

Preprocessing Tasks

Preprocessing Tasks include all those routines, processes, and methods required to prepare data for a text mining system's core knowledge discovery operations. These tasks typically center on data source preprocessing and categorization activities. Preprocessing tasks generally convert the information from each orig- inal data source into a canonical format before applying various types of feature extraction methods against these documents to create a new collection of doc- uments fully represented by concepts. Where possible, preprocessing tasks may also either extract or apply rules for creating document date stamps, or do both. Occasionally, preprocessing tasks may even include specially designed methods used in the initial fetching of appropriate "raw" data from disparate original data sources.

Core Mining Operations

Core Mining Operations are the heart of a text mining system and include pat- tern discovery, trend analysis, and incremental knowledge discovery algorithms. Among the commonly used patterns for knowledge discovery in textual data are distributions (and proportions), frequent and near frequent concept sets, and associations. Core mining operations can also concern themselves with compar- isons between – and the identification of levels of "interestingness" in – some of these patterns. Advanced or domain-oriented text mining systems, or both, can also augment the quality of their various operations by leveraging background knowledge sources. These core mining operations in a text mining system have also been referred to, collectively, as knowledge distillation processes.

Presentation Layer Components

Presentation Layer Components include GUIand pattern browsing functionality as well as access to the query language. Visualization tools and user-facing query editors and optimizers also fall under this architectural category. Presentation- layer components may include character-based or graphical tools for creating or modifying concept clusters as well as for creating annotated profiles for specific concepts or patterns.

Refinement Techniques

At their simplest, include methods that filter redundant information and cluster closely related data but may grow, in a given text mining system, to represent a full, comprehensive suite of

suppression, ordering, pruning, generalization, and clustering approaches aimed at discovery optimization. These techniques have also been described as postprocessing. Preprocessing

3.2 High-level text mining functional architecture

Preprocessing tasks and core mining operations are the two most critical areas for any text mining system and typically describe serial processes within a generalized view of text mining system architecture, as shown in Figure 3.3.

fig:arch3

Figure 3.3: High-level text mining functional architecture.

3.3 Generic Architecture for text mining systems

At a slightly more granular level of detail, one will often find that the processed document collection is, itself, frequently intermediated with respect to core mining operations by some form of flat, compressed or hierarchical representation, or both, of its data to better support various core mining operations such as hierarchical tree browsing. This is illustrated in Figure 3.4. The schematic in Figure 3.4 also factors in the typical positioning of refinement functionality. Further, it adds somewhat more detail with respect to relative functioning of core data mining algorithms.

3.4 Architecture for an advanced text mining system

Many text mining systems – and certainly those operating on highly domain-specific data sources, such as medicine, financial services, high tech, genomics, proteomics, and chemical compounds – can benefit significantly from access to special background or domain-specific data sources. See Figure I.6.

Background knowledge is often used for providing constraints to, or auxiliary information about, concepts found in the text mining collection's document collec- tion. The background knowledge for a text mining system can be created in various ways. Onecommonway is to run parsing routines against external knowledgesources, such as formal ontologies, after which unary or binary predicates for the concept- labeled documents in the text mining system's document collection are identified. These unary and binary predicates, which describe properties of the entities rep- resented by each concept deriving from the expert or "gold standard" information sources, are in turn put to use by a text mining system's query engine. In addition, such constraints can be used in a text mining system's front end to allow a user to either :

- create initial queries based around these constraints
- refine queries over time by adding, substracting, or concatenating constraints.

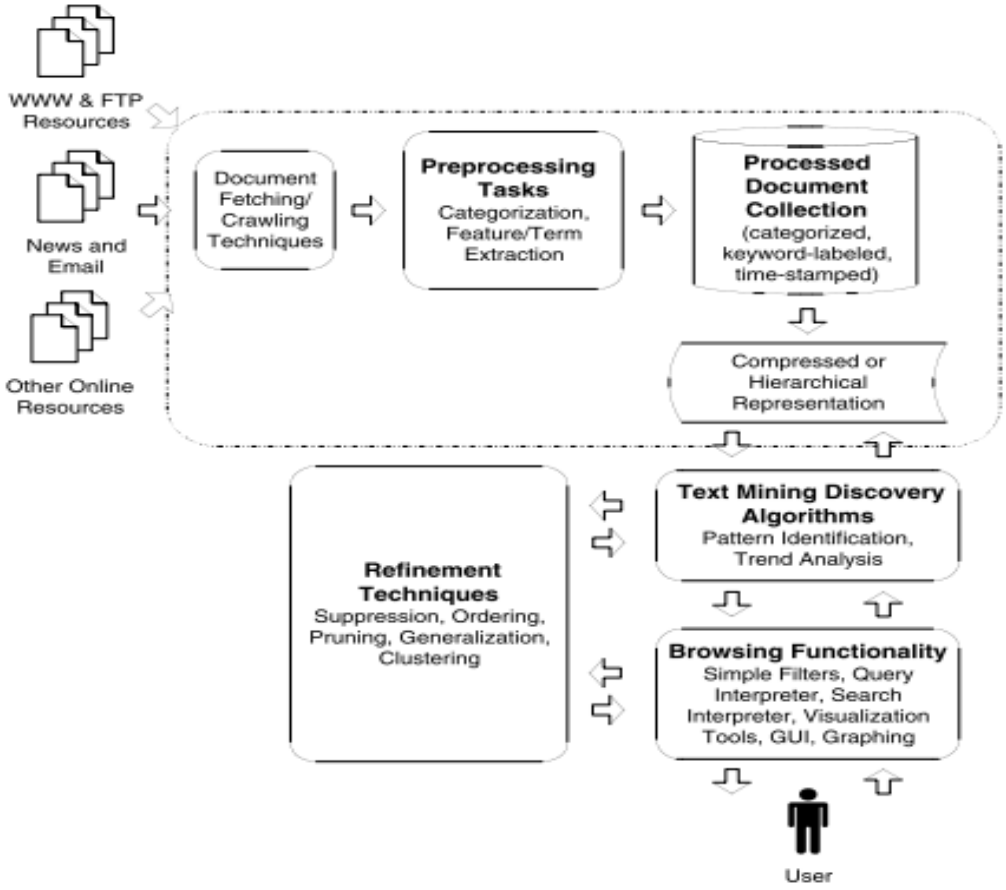

fig:arch4

Figure 3.4: System architecture for generic text mining system.

Commonly, background knowledge is preserved within a text mining system's architecture in a persistent store accessible by various elements of the sys- tem. This type of persistent store is sometimes loosely referred to as a system's knowledge base. The typical position of a knowledge base within the system archi- tecture of a text mining system can be seen in Figure 3.6.

These generalized architectures are meant to be more descriptive than prescriptive in that they represent some ofthe mostcommonframeworks found in the present generation of text mining systems. Good sense, however, should be the guide for prospective system architects of text mining applications, and thus significant vari- ation on the general themes that have been identified is possible. System architects and developers could include more of the filters typically found in a text mining system's browser or even within subroutines contained among the system's store of refinement techniques as "preset" options within search algorithms included in its main discovery algorithms. Likewise, it is conceivable that a particular text mining system's refinement techniques or main discovery algorithms might later find a very fruitful use for background knowledge.

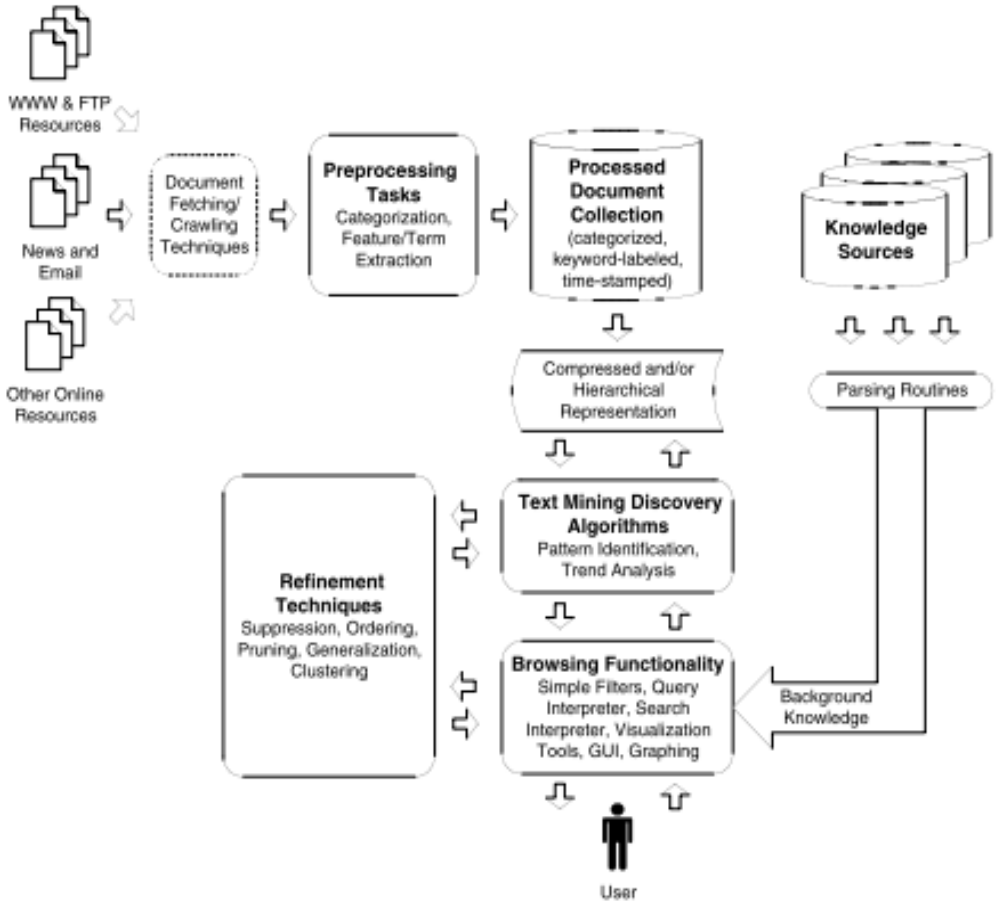

fig:arch5

Figure 3.5: System architecture for an advanced or domain-oriented text mining system.

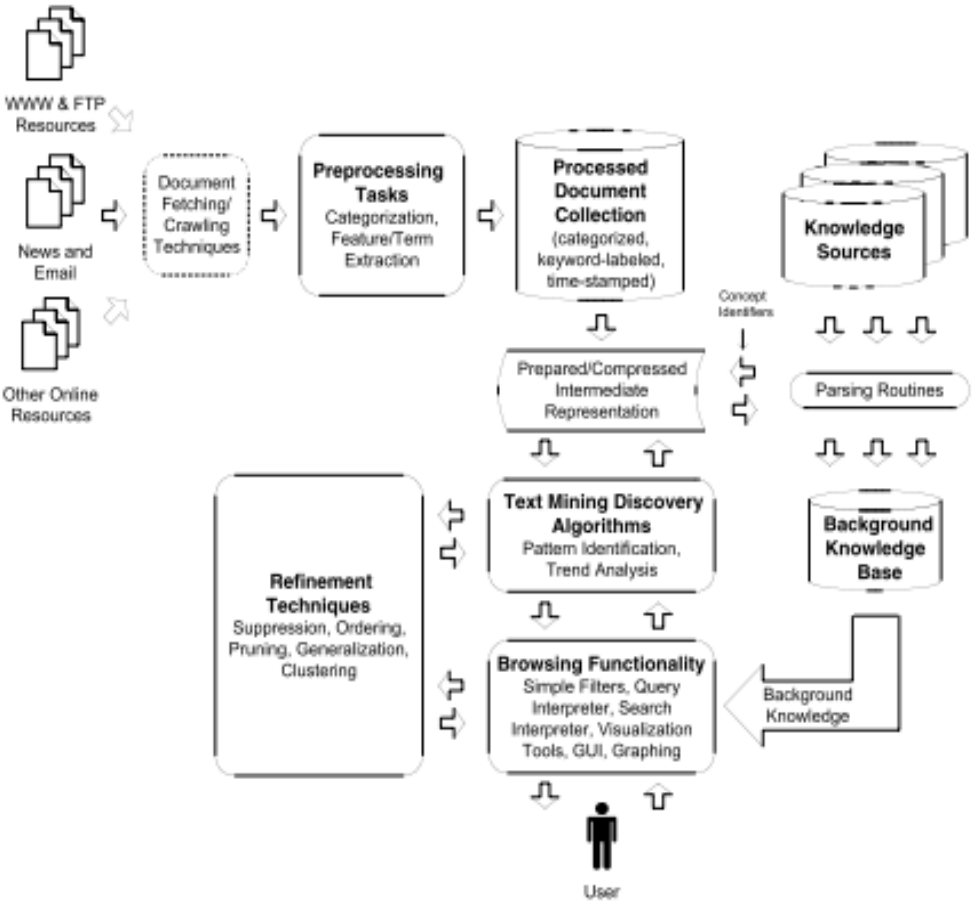

Figure 3.6: System architecture for an advanced text mining system with background knowledge base.

fig:arch6

CHAPTER 4

Text Mining Operations Core

4.1 Introduction

Core mining operations in text mining systems center on the algorithms that underlie the creation of queries for discovering patterns in document collections. This chapter describes Pattern-discovery algorithms from a high-level definitional perspective. In addition, we examine the incorporation of background knowledge into text mining query operations. Finally, we briefly treat the topic of text mining query languages.

4.2 Core Text Mining Operations Core

Core text mining operations consist of various mechanisms for discovering patterns of concept occurrence within a given document collection or subset of a document collection. The three most common types of patterns encountered in text mining are distributions (and proportions), frequent and near frequent sets, and associations. Typically, when they offer the capability of discovering more than one type of pattern. text mining systems afford users the ability to toggle between displays of the different types of patterns for a given concept or set of concepts. This allows the richest possible exploratory access to the underlying document collection data through a browser.

Distributions

This section defines of text mining's most commonly used distributions. We illustrate this in the context of a hypothetical text mining system that has a document collection **W** composed of documents containing news wire stories about world affairs that have all been preprocessed with concept labels.

Concept Selection

Whether as an initial step, to create a baseline, or to create more meaningful subdivisions of a single document collection for comparison purposes, text mining systems generally need to refer to some subcollection of a complete document collection. This activity is commonly referred to as **concept selection**. Given some collection of documents **D**, a text mining system will have a requirement to refer to some subcollection of **D** that is **labeled** by one or more given **concepts**. If **D** is a collection of documents and **K** is a set of concepts, **D/K** is the subset of documents in **D** labeled with **all of the concepts** in **K**.

When it is clear from the context, given a single concept **k**, rather than writing **D/k** we use the notation **D/k**.

For example, the collection **W** contains a subset of the World Affairs collection – namely those documents that are labeled with the concepts **Algeria, Palestine, Gazza**. **W/bush** contains the subset of documents that are labeled (at least) with **GAZZA**.

and **W/F3** contains those documents that are labeled with any terminal node under **F** (i.e., labeled with any F3 Places). **F3** is treated as a concept here when is being performed concept selection (rather than being viewed as the set of concepts under it, in which case it would have required all of its descendants to be present).

Concept Proportion

Text mining systems often need to identify or examine the proportion of a set of documents labeled with a particular concept. This analytic is commonly referred to as concept proportion. If **D** is a collection of documents and **K** is a set of concepts, ***f*(D,K)** is the fraction of documents in **D** labeled with all of the concepts in **K**, that is :

$$f(D, K) = \frac{|D/K|}{|D|} \tag{4.1}$$

Given one concept **k**, rather than writing **f(D,{K})**. We use the notation **f(D, k)**. When **D** is clear from context, we drop it and write ***f*(k)**.

Thus, for example, ***f*(W, {Algeria, Palestine, Gazza}** is the fraction of documents in the World Affairs collection labeled with Algeria, Palestine, Gazza; ***f(Gazza)*** is the proportion of the collection labeled with the concept Gazza; and ***f(F3)*** is the proportion labeled with any ***(F3)*** Places.

Conditional Concept Proportion

By employing definitions of selection and proportion, text mining systems can already begin identifying some useful quantities for analyzing a set of documents. For example, a text mining system might want to identify the proportion of those documents labeled with $\boldsymbol{K}_2$ that are also labeled by $\boldsymbol{K}_1$, which could be designated by expression ***f(D/*$\boldsymbol{K}_2$*,*$\boldsymbol{K}_1$*)***.

This type of proportion occurs regularly enough that it has received an explicit name and notation: ***conditional concept proportion***. If **D** is a collection of documents and $\boldsymbol{K}_1$ and $\boldsymbol{K}_2$ are sets of concepts, ***f (D,*** $\boldsymbol{K}_1$ ***|*** $\boldsymbol{K}_2$ ***)*** is the proportion of all those documents in **D** labeled with $\boldsymbol{K}_2$ that are also labeled with $\boldsymbol{K}_2$, that is, :

$$f(D, K_1|K_2) = f(D/K_2, K_1). \tag{4.2}$$

When D is clear from context, we will write this as ***f*** ***(***$\boldsymbol{K}_1$ ***/*** $\boldsymbol{K}_2$ ***)***.

Applying this definition, we find that f(Gazza | Palestine) would represent the proportion of all documents labeled by the concept ***Palestine*** that are also labeled by the concept ***Gazza***.

Concept Proportion Distribution

Commonly, a text mining system needs to analyze the distribution of concepts that are descendents of a particular node in a concept hierarchy. For example, a text mining system might need to allow the analysis of the distribution of concepts denoting finance topics – that is, descendents of the finance topics node in an example concept hierarchy.

To accomplish this, a text mining system could use the expression $P_K(x)$ to refer to such distributions – it will assign to any concept x in K a value between 0 and 1 – where the values are not required to add up to 1. This type of proportion can be referred to as a concept distribution.

One particularly important concept distribution for knowledge discovery operations is the concept proportion distribution, which gives the proportion of documents in some collection that are labeled with each of a number of selected concepts.

If D is a collection of documents and K is a set of concepts, $F_K(D, x)$ is the proportion of documents in D labeled with x for any x in K.

When D is clear from context, we will write this as $F_K(x)$. Note the distinction between $P_K(x)$ and $F_K(x)$.

- $P_K(x)$ refers generically to any function that is a concept distribution.
- $F_K(x)$ is a specific concept distribution defined by a particular concept-labeled set of documents.

For example $F_{topics}(R, x)$ would represent the proportions of documents in W labeled with keywords under the topics node in the concept hierarchy. In this expression, $topics$ is used as shorthand for referring to a set of concepts – namely, all those that occur under the topics node – instead of explicitly enumerating them all.

Also, note that $\mathbf{F}_{\{K\}}(D, k) = f(D, k)$ – that is, $\mathbf{F}_{\{K\}}$ subsumes the earlier defined f when it is applied to a single concept. Unlike f, however, $\mathbf{F}_{\{K\}}$ is restricted to refer only to the proportion of occurrences of individual concepts (those occurring in the set K). Thus f and F are not comparable.

Mathematically, $\mathbf{F}$ is not a true frequency distribution, for each document may be labeled by multiple items in the set $\mathbf{K}$. Thus, for example, a given document may be labeled by two (or more) **F3** Places because occurrences of concepts are not disjoint events. Therefore, the sum of values in $\mathbf{F}_{F3}$ may be greater than one. In the worst case, if all concepts in K label all documents, the sum of the values in a distribution $\mathbf{F}$ can be as large as $|\mathbf{K}|$.

Furthermore, because some documents may contain none of the concepts in a given $\mathbf{K}$, the sum of frequencies in $\mathbf{F}$ might also be smaller than one – in the worst case, zero. Nonetheless, the term "**distribution**" is used for $\mathbf{F}$, for many of the connotations this term suggests still hold true.

Conditional Concept Proportion Distribution

Just as was the case for concept proportions, text mining systems can also leverage conditional keyword-proportion distributions, which are probably one of the most used concept distributions in text mining systems.

If **D** is a collection of documents and **K** and **K'** are sets of concepts,
F_K (D, x | K') is the proportion of those documents in **D** labeled with all the concepts in **K'** that are also labeled with concept **x** (with x in **K**), that is:

$$F_K(D, x|K') = F_K(D/K|K', x). \tag{4.3}$$

We often write this as **F_K (x | K')** when **D** is clear from context.

Thus, for example, **F_{topics} (x | Algeria)** would assign any concept x under **topics** in the hierarchy with the proportion of documents labeled by **x** within the set of all documents labeled by the concept **Algeria**, and **F_{topics} (x | {Palestine, Gazza)}** is the similar distribution for those documents labeled with both the **Palestine** and **Gazza** concepts.

Average Concept Proportion

One of the baseline distributions text mining systems use to compare distributions is the average distribution over a set of sibling nodes in the hierarchy.

For example, when looking at the proportions of loan within South American countries such as ***f(W, loan | Argentina), f (W, loan | Brazil), and f (W, loan | Columbia))***, an end user may be interested in the average of all proportions of this form for all the South American countries – that is, the average of all proportions of the form ***f (W, loan |k)***, where ***k*** ranges over all South American countries.

Given a collection of documents **D**, a concept **k**, and an internal node in the hierarchy **n**, an average concept proportion, denoted by **a(D, k | n)**, is the average value of ***f(D, k| k')***, where ***k'*** ranges over all immediate children of ***n*** – that is :

$$a(D, k|n) = Avg_{\{k\ isachildofn\}} f(D, k|k'). \tag{4.4}$$

When **D** is clear from context, this will be written **a(k|n)**.

For example, **a(loan | South America)** is the average concept proportion of ***f(loan | k')*** as ***k'*** varies over each child of the node South America in the concept hierarchy. it is the average conditional keyword proportion for loan within South American countries. This quantity does not average the values weighted by the number of documents labeled by each child of ***n***. Instead, it equally represents each descendant of ***n*** and should be viewed as a summary of what a typical concept proportion is for a child of ***n***.

Average Concept Distribution

An end user may be interested in the distribution of averages for each economic topic within South American countries. This is just another keyword distribution referred to as an average concept distribution.

Given a collection of documents **D** and two internal nodes in the hierarchy **n** and **n'** , an average concept distribution, denoted by **A_n (Dx | n')**, is the distribution that, for any **x** that is a child of **n**, averages **x's** proportions over all children of n' – that is:

$$A_n(D, x|n') = Avg_{\{k'isachildofn'\}} F_n(D, x|k'). \tag{4.5}$$

When clear from context, this will be written A_n(x|n').

For example **A_{topics} (x|South America)**, which can be read as "the average distribution of topics within South American countries," gives the average proportion within all South American countries for any topic **x**. A very basic operation for text mining systems using concept-distributions is the display of conditional concept-proportion distributions. For example,

a user may be interested in seeing the proportion of documents labeled with each child of topics for all those documents labeled by the concept Argentina, that is, the proportion of Argentina documents that are labeled with each topic keyword.

This distribution would be designated by $\mathbf{F}_{topics}$**(W, x | Argentina)**, and a correlating graph could be generated, for instance, as a bar chart, which might display the fact that **12 articles** among all articles of Argentina are annotated with **sorghum**, **20 with corn**, **32 with grain**, and so on, providing a summary of the areas of economical activity of Argentina as reflected in the text collection. Conditional concept-proportion distributions can also be conditioned on sets of concepts.

In some sense, this type of operation can be viewed as a more refined form of traditional concept-based retrieval. For example, rather than simply requesting all documents labeled by Argentina or by both UK and USA, the user can see the documents at a higher level by requesting documents labeled by Argentina for example, and first seeing what proportions are labeled by concepts from some secondary set of concepts of interest with the user being able to access the documents through this more fine-grained grouping of Argentina-labeled documents.

4.3 Comparing with Average Distributions

Consider a conditional proportion of the form F_k(D,x|K)f the distribution over K of all documents labeled with some concept k (not necessarily in K). It is natural to expect that this distribution would be similar to other distributions of this form over conditioning events K' that are siblings of k. When they differ substantially it is a sign that the documents labeled with the conditioning concept k may be of interest.

To facilitate this kind of comparison of concept-labeled documents with the average of those labeled with the concept and its siblings, a user can specify two internal nodes of the hierarchy and compare individual distributions of concepts under one of the nodes conditioned on the concept set under the other node – that is, compute : D(F_n (x | k)||A_n (x | n')) for each k that is a child of n'.

4.4 Comparing Specific Distributions

- The preceding mechanism for comparing distributions with an average distribution is also useful for comparing conditional distributions of two specific nodes in the hierarchy. For example, one could measure the distance from the average topic distribution of Arab_League countries to the average topic distribution of G8 countries.
- An answer set could be returned from a query into a table with countries sorted in decreasing order of their contribution to the distance (second column) – namely
- d(A_{topics} (K | Arab_League) || A_{topics} (k | G8)).
- Additional columns could show, respectively, the percentage of the topic in the average topic distribution of the Arab_League countries (A_{topics} (x | G8)) and in the average topic distribution of the G8 countries (A_{topics} (x | G8)).
- One could also show the total number of articles in which the topic appears with any Arab_League country and any G8 country. This would reveal the topics with which

Arab_League countries are associated much more than G8 countries such as grain, wheat, and crude oil.

- Finally, one could show the comparison in the opposite direction, revealing the topics with which G8 countries are highly associated relative to the Arab_League.

4.5 Frequent and Near Frequent Sets

In addition to proportions and distributions, another basic type of pattern that can be derived from a document collection is a frequent concept set. This is defined as a set of concepts represented in the document collection with co-occurrences at or above a minimal support level (given as a threshold parameter s; i.e., all the concepts of the frequent concept set appear together in at least s documents). Although originally defined as an intermediate step in finding association rules, frequent concept sets contain a great deal of information of use in text mining.

- The search for frequent sets has been well treated in data mining literature, stemming from research centered on investigating market basket–type associations first published by Agrawal et al. in 1993.
- Essentially, a document can be viewed as a market basket of named entities. Discovery methods for frequent concept sets in text mining build on the Apriori algorithm of Agrawal et al. (1993) used in data mining for market basket association problems.

With respect to frequent sets in natural language application:

- support is the number (or percent) of documents containing the given rule – that is, the co-occurrence frequency.
- Confidence is the percentage of the time that the rule is true
- A frequent set in text mining can be seen directly as a query given by the conjunction of concepts of the frequent set.

- Frequent sets can be partially ordered by their generality and hold the simple but useful pruning property that each subset of a frequent set is a frequent set.
- The discovery of frequent sets can be useful both as a type of search for patterns in its own right and as a preparatory step in the discovery of associations.

4.6 Discovering Frequent Concept Sets

- As mentioned in the previous section, frequent sets are generated in relation to some support level. Because support (i.e., the frequency of co-occurrence) has been by convention often expressed as the variable σ , frequent sets are sometimes also referred to as **σ-covers**, or **σ-cover** sets.
- A simple algorithm for generating frequent sets relies on incremental building of the group of frequent sets from singleton **σ-covers**, to which additional elements that continue to satisfy the support constraint are progressively added.

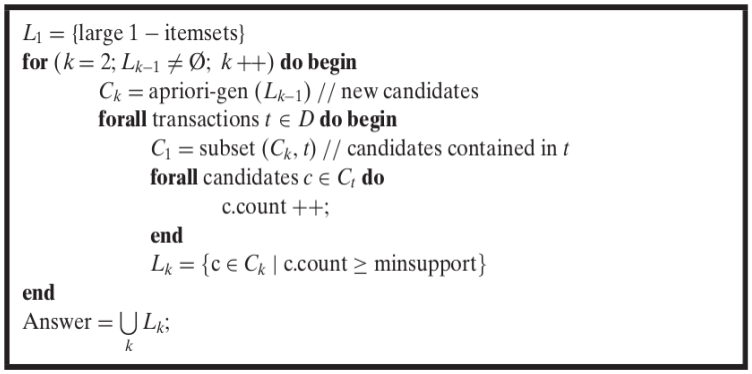

Figure 4.1: Algorithm for Frequent Set Generation.

fig:img1

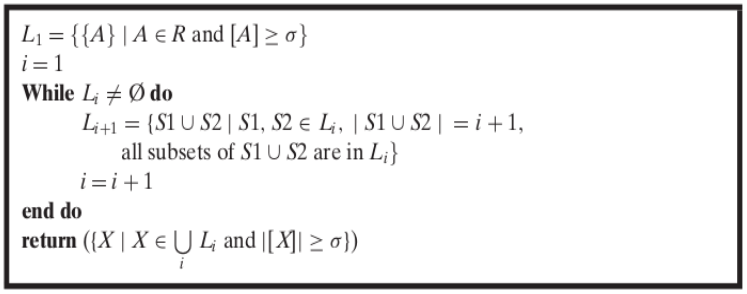

Figure 4.2: Algorithm for Frequent Set Generation.

fig:img1

4.7 Near Frequent Concept Sets

- Near frequent concept sets establish an undirected relation between two frequent sets of concepts. This relation can be quantified by measuring the degree of overlapping.
- for example, on the basis of the number of documents that include all the concepts of the two concept sets. This measure can be regarded as a distance function between the concept sets. Several distance functions can be introduced (e.g., based on the cosine of document vectors, Tanimoto distance, etc.).
- Directed relations between concept sets can also be identified. These are considered types of associations.
- A formal description of association rules was first presented in the same research on "**market basket**" problems that led to the identification of **frequent sets** in data mining. Subsequently, **associations** have been widely discussed in literature on knowledge discovery targeted at both structured and unstructured data.
- In text mining, **associations** specifically refer to the directed relations between concepts or sets of concepts.
- An association rule is generally an expression of the form : **A** $\rightarrow$ **B**, where **A** and **B** are sets of features.

- An association rule **A** $\rightarrow$ **B** indicates that transactions that involve **A** tend also to involve **B**.
- For example, from the original market-basket problem, an **association rule** might be **25** percent of the transactions that contain **pretzels** also contain **soda**; **8** percent of all transactions contain both items.
- In this example, **25** percent refers to the **confidence level** of the association rule, and **8** percent refers to the rule's level of **support**.
- With respect to concept sets, association rule **A** $\rightarrow$ **B**, relating two frequent concept sets **A** and **B**, can be quantified by these two basic measures of support and confidence.
- Confidence is the percentage of documents that include all the concepts in **B** within the subset of those documents that include all the concepts in **A**.
- Support is the percentage (or number) of documents that include all the concepts in **A** and **B**.

More precisely, we can describe association rules as follows:

- Let $\mathbf{r} = \{\mathbf{t}_1, \ldots, \mathbf{t}_n\}$ be a collection of documents, each labeled with some subset of concepts from the **m-concept** set $\mathbf{R} = \{\mathbf{I}_1, \mathbf{I}_2, \ldots, \mathbf{I}_m\}$.
- Given a concept **A** and document **t**, we write $\mathbf{t(A) = 1}$ if **A** is one of the concepts labeling **t**, and $\mathbf{t(A) = 0}$ otherwise.
- If **W** is a subset of the concepts in **R**, $\mathbf{t(W) = 1}$ represents the case that $\mathbf{t(A) = 1}$ for every concept $\mathbf{A} \in \mathbf{W}$
- Given a set **X** of concepts from **R**, define $\mathbf{(X) = i \mid t}_i\ \mathbf{(X) = 1}$;
- **(X)** is the set of all documents $\mathbf{t}_i$ that are labeled (at least) with all the concepts in **X**.
- Given some number σ (the support threshold), **X** is called a σ**-covering** if $\mathbf{v|(X)|} \geq \sigma$.
- **W** $\rightarrow$ **B** is an association rule over **r** if **W** $\subseteq$ **R** and **B** $\subseteq$ **R****W**. We refer to **W** as the left-hand side **(LHS)** of the association and **B** as the right-hand side **(RHS)**.
- Finally, we say that **r** satisfies **W** $\rightarrow$ **B** respect to $\mathbf{0 < \gamma \leq 1}$ (the confidence threshold) and σ (the support threshold). if **W** $\cup$ **B** is a σ**-covering**.
- For example: $|(\mathrm{W} \cup \mathrm{B})| \geq \sigma$ and $|(\mathrm{W} \cup \mathrm{B})|/|(\mathrm{W})| \geq \gamma$.
- Intuitively, this means that, of all documents labeled with the concepts in **W**, at least a proportion γ of them are also labeled with the concepts in **B**; further, this rule is based on at least σ documents labeled with all the concepts in both **W** and **B**.
- For example, a document collection has documents labeled with concepts in the following tuples: {x, y, z, w}, {x, w}, {x, y, p}, {x, y, t}.
- If $\gamma = 0.8$ and $\sigma = 0.5$, and x, y, w, x, w, and x, y are coverings, then y $\rightarrow$ x and w $\rightarrow$ x are the only associations.

CHAPTER 5

Knowledge-based Models vs. Probabilistic Models

5.1 Introduction to Models

Overview

Knowledge-based models and probabilistic models are fundamental approaches in information extraction. In this section, we will delve into the essence of these models and their significance in processing and extracting information from unstructured or semi-structured textual data.

Knowledge-based Models

Knowledge-based models rely on predefined rules and domain-specific knowledge to extract information. These models encode explicit knowledge about the relationships between entities and the structure of the information. They are particularly useful in scenarios where the extraction task can be well-defined through rules.

Probabilistic Models

On the other hand, probabilistic models operate on statistical principles. They leverage probabilities to make predictions about the likelihood of certain information being present in the text. These models are advantageous in situations where the extraction task involves uncertainty or ambiguity.

Challenges and Considerations

Each model type comes with its own set of challenges. Knowledge-based models may struggle with adapting to new domains, while probabilistic models may face difficulties in handling highly complex or rare patterns in the data. Striking a balance between the two approaches is often key to successful information extraction.

Evolution of Models

As technology advances, there is a continual evolution in the sophistication of knowledge-based and probabilistic models. Recent trends involve combining aspects of both approaches to create hybrid models that harness the strengths of each paradigm.

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

Nam dui ligula, fringilla a, euismod sodales, sollicitudin vel, wisi. Morbi auctor lorem non justo. Nam lacus libero, pretium at, lobortis vitae, ultricies et, tellus. Donec aliquet, tortor sed accumsan bibendum, erat ligula aliquet magna, vitae ornare odio metus a mi. Morbi ac orci et nisl hendrerit mollis. Suspendisse ut massa. Cras nec ante. Pellentesque a nulla. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Aliquam tincidunt urna. Nulla ullamcorper vestibulum turpis. Pellentesque cursus luctus mauris.

Nulla malesuada porttitor diam. Donec felis erat, congue non, volutpat at, tincidunt tristique, libero. Vivamus viverra fermentum felis. Donec nonummy pellentesque ante. Phasellus adipiscing semper elit. Proin fermentum massa ac quam. Sed diam turpis, molestie vitae, placerat a, molestie nec, leo. Maecenas lacinia. Nam ipsum ligula, eleifend at, accumsan nec, suscipit a, ipsum. Morbi blandit ligula feugiat magna. Nunc eleifend consequat lorem. Sed lacinia nulla vitae enim. Pellentesque tincidunt purus vel magna. Integer non enim. Praesent euismod nunc eu purus. Donec bibendum quam in tellus. Nullam cursus pulvinar lectus. Donec et mi. Nam vulputate metus eu enim. Vestibulum pellentesque felis eu massa.

5.2 Comparison and Contrast

Model Characteristics

To understand the differences and similarities between knowledge-based and probabilistic models, it's essential to delve into their fundamental characteristics. Table 5.1 provides a more detailed comparative overview.

Aspect	**Knowledge-based Models**	**Probabilistic Models**
Representation	Rule-based	Probability distributions
Adaptability	Domain-specific	Domain-agnostic
Interpretability	High, explicit rules	Variable, statistical inference
Handling Ambiguity	Limited, rigid rules	Effective, uncertainty management
Learning Approach	Deterministic	Stochastic

Table 5.1: Comparison of Knowledge-based and Probabilistic Models

tab:model_characteristics

Case Studies

Illustrating the concepts discussed, this section presents more in-depth case studies where knowledge-based and probabilistic models were employed for information extraction. Real-world examples will highlight the practical implications of choosing one model over the other.

Case Study	Scenario	Model Type	Outcome
Case 1	Financial News	Probabilistic	Accurate sentiment analysis
Case 2	Legal Documents	Knowledge-based	Precise named entity recognition
Case 3	Healthcare Records	Hybrid	Improved information retrieval

Table 5.2: In-depth Case Studies

:case_studies

Pros and Cons

Understanding the advantages and disadvantages of each model type is crucial for informed decision-making in the field of information extraction. Table 5.3 outlines an extended list of pros and cons for knowledge-based and probabilistic models.

Model Type	Pros	Cons
Knowledge-based Models	Explainability, Explicit rules	Limited adaptability, Rigid structure
Probabilistic Models	Adaptability, Effective in uncertainty	Lack of interpretability, Data-intensive

Table 5.3: Extended Pros and Cons of Knowledge-based and Probabilistic Models

tab:
os_and_cons

Quisque ullamcorper placerat ipsum. Cras nibh. Morbi vel justo vitae lacus tincidunt ultrices. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. In hac habitasse platea dictumst. Integer tempus convallis augue. Etiam facilisis. Nunc elementum fermentum wisi. Aenean placerat. Ut imperdiet, enim sed gravida sollicitudin, felis odio placerat quam, ac pulvinar elit purus eget enim. Nunc vitae tortor. Proin tempus nibh sit amet nisl. Vivamus quis tortor vitae risus porta vehicula.

Fusce mauris. Vestibulum luctus nibh at lectus. Sed bibendum, nulla a faucibus semper, leo velit ultricies tellus, ac venenatis arcu wisi vel nisl. Vestibulum diam. Aliquam pellentesque, augue quis sagittis posuere, turpis lacus congue quam, in hendrerit risus eros eget felis. Maecenas eget erat in sapien mattis porttitor. Vestibulum porttitor. Nulla facilisi. Sed a turpis eu lacus commodo facilisis. Morbi fringilla, wisi in dignissim interdum, justo lectus sagittis dui, et vehicula libero dui cursus dui. Mauris tempor ligula sed lacus. Duis cursus enim ut augue. Cras ac magna. Cras nulla. Nulla egestas. Curabitur a leo. Quisque egestas wisi eget nunc. Nam feugiat lacus vel est. Curabitur consectetuer.

Suspendisse vel felis. Ut lorem lorem, interdum eu, tincidunt sit amet, laoreet vitae, arcu. Aenean faucibus pede eu ante. Praesent enim elit, rutrum at, molestie non, nonummy vel, nisl. Ut lectus eros, malesuada sit amet, fermentum eu, sodales cursus, magna. Donec eu purus. Quisque vehicula, urna sed ultricies auctor, pede lorem egestas dui, et convallis elit erat sed nulla. Donec luctus. Curabitur et nunc. Aliquam dolor odio, commodo pretium, ultricies non, pharetra in, velit. Integer arcu est, nonummy in, fermentum faucibus, egestas vel, odio.

5.3 Applications

Text Summarization

One practical application of knowledge-based models is in text summarization. These models can identify key information and generate concise summaries based on predefined rules.

Sentiment Analysis

Probabilistic models excel in sentiment analysis tasks. By learning from large datasets, these models can accurately predict the sentiment expressed in textual content.

Question Answering Systems

Knowledge-based models play a crucial role in question answering systems. They can interpret complex queries using predefined rules to extract relevant information.

Named Entity Recognition

The application of probabilistic models in named entity recognition allows for flexible adaptation to different domains, enhancing the accuracy of entity identification.

Information Retrieval

Both model types contribute to information retrieval systems. Knowledge-based models provide structured retrieval based on explicit rules, while probabilistic models enhance relevance ranking through statistical methods.

CHAPTER 6

Recognition of Named Entities and Classification

6.1 Named Entity Recognition (NER)

Overview

Named Entity Recognition (NER) is a fundamental task in natural language processing (NLP) that focuses on identifying and classifying entities within text. These entities could be people, organizations, locations, dates, or numerical values like percentages. For example, in the sentence, *"Apple Inc. was founded by Steve Jobs in Cupertino in 1976,"* an NER system would classify **Apple Inc.** as an organization, **Steve Jobs** as a person, **Cupertino** as a location, and **1976** as a date.

NER transforms raw, unstructured data into structured information, aiding tasks like information retrieval, machine translation, and summarization.

Example: Given the text: *"Barack Obama was born on August 4, 1961, in Honolulu, Hawaii."*

- **Barack Obama**: Person
- **August 4, 1961**: Date
- **Honolulu**: Location
- **Hawaii**: Location

Techniques

NER systems typically rely on two major approaches: **rule-based** and **machine learning-based**.

- **Rule-Based Approaches:** These rely on predefined linguistic rules or patterns. For instance, capitalized words following titles like "Mr." or "Dr." may be tagged as persons.
- **Machine Learning Approaches:** These include models such as Conditional Random Fields (CRF), Support Vector Machines (SVM), and deep learning-based models such as Long Short-Term Memory (LSTM) networks and transformers (e.g., BERT).

Technique	Description	Examples
Rule-Based	Based on predefined rules and patterns	Capitalization patterns
Machine Learning	Learns from data, often using CRFs or decision trees	CRF, BERT
Deep Learning	Uses neural networks to capture complex patterns and context	LSTM, Transformer mo

Table 6.1: NER Techniques and Models

Role in Information Extraction

NER is crucial in **information extraction** (IE) pipelines. It helps categorize entities, a foundational step for downstream tasks such as relationship extraction, event detection, and knowledge base construction. For example, recognizing companies, products, and monetary amounts is invaluable in analyzing financial reports or news articles.

Challenges

NER systems face several challenges:

- **Ambiguity:** Some entities, like *Apple*, can refer to either a company or a fruit.
- **Context-Dependent Entities:** The meaning of entities often depends on context, such as *"Washington"* referring to either the person or the city.
- **Domain-Specific Language:** General models may perform poorly in specialized domains (e.g., medical texts), necessitating domain-specific adaptations.

Recent Developments

Recent advancements in NER include the use of deep learning models like transformers (BERT, GPT), which better handle contextual nuances and outperform traditional NER methods, particularly in complex sentences with multiple layers of meaning.

Case Study: In a medical corpus, applying a domain-adapted NER model might involve recognizing entities such as drug names, medical conditions, and dosages, improving the analysis of clinical reports.

6.2 Entity Classification

Introduction

Following NER, **entity classification** assigns a specific category to each recognized entity. For example, a person's name may be classified into subcategories such as *politician*, *athlete*, or *artist* based on the context.

Example: For the sentence *"Elon Musk is the CEO of Tesla and founded SpaceX,"* an entity classification system might assign:

- **Elon Musk**: Person (Businessman/Entrepreneur)
- **Tesla**: Organization (Automobile Manufacturer)
- **SpaceX**: Organization (Aerospace Manufacturer)

Methods

Entity classification can be performed using several approaches:

- **Rule-Based Systems:** Entities are matched against predefined categories using patterns or dictionaries.
- **Machine Learning:** Algorithms like Support Vector Machines (SVM) or neural networks can automatically classify entities based on features extracted from the text.

Role in Information Retrieval

Entity classification improves the accuracy of **information retrieval** systems by organizing data into meaningful categories. For instance, classifying legal entities in law documents helps legal professionals quickly find relevant cases or precedents.

Practical Considerations

When developing an entity classification system, consider:

- **Data Quality:** Poor quality training data can lead to incorrect classifications.
- **Feature Selection:** Choosing the right features (e.g., the surrounding words of an entity) is crucial for machine learning models.
- **Model Tuning:** Hyperparameter tuning significantly affects model accuracy.

Applications

Entity classification is widely used across domains:

- **Healthcare:** Classifying diseases, symptoms, and treatments.
- **Finance:** Labeling companies, stock symbols, and financial metrics.
- **Legal:** Classifying court rulings, laws, and legal terms.

Classification Metrics

Metric	Formula	Description
Accuracy	$\frac{TP+TN}{\text{Total}}$	Measures overall correctness
Precision	$\frac{TP}{TP+FP}$	Accuracy of positive predictions
Recall	$\frac{TP}{TP+FN}$	Ability to capture all relevant items
F1 Score	$\frac{2\times\text{Precision}\times\text{Recall}}{\text{Precision}+\text{Recall}}$	Balances precision and recall

Table 6.2: Classification Metrics

Confusion Matrix

	Predicted Positive	**Predicted Negative**
Actual Positive (AP)	True Positive (TP)	False Negative (FN)
Actual Negative (AN)	False Positive (FP)	True Negative (TN)

Table 6.3: Confusion Matrix

CHAPTER 7

Coreference Resolution

7.1 Introduction

Understanding Coreference

Coreference is a linguistic phenomenon where two or more expressions in a text refer to the same entity. In the context of information extraction, coreference resolution aims to identify and connect these referring expressions, enhancing the understanding of the text.

Importance in Information Extraction

Understanding coreference is crucial for constructing a coherent and accurate representation of information from textual data. Coreference resolution aids in creating a more structured and interconnected view of entities mentioned in a document.

7.2 Coreference Resolution Techniques

Rule-Based Approaches

Rule-based coreference resolution relies on predefined linguistic rules to identify and link expressions that refer to the same entity. These rules may involve syntactic, semantic, and contextual information.

Machine Learning Methods

Machine learning-based approaches leverage annotated data to train models that can automatically learn patterns and relationships between expressions. Algorithms such as Mention-Pair and Entity-Centric models fall into this category.

Hybrid Approaches

Hybrid approaches combine rule-based and machine learning methods to benefit from the strengths of both. This integration aims to improve robustness and adaptability across different domains.

Neural Coreference Resolution

Recent advancements in deep learning have led to the development of neural coreference resolution models, including neural mention-ranking models and end-to-end neural models. These models capture complex dependencies and contextual information for improved resolution accuracy.

7.3 Challenges and Solutions

Challenges in Coreference Resolution

Coreference resolution faces challenges such as ambiguity, varying contextual clues, and the need to handle different types of entities. Resolving coreference becomes more intricate in the presence of complex sentence structures and diverse writing styles.

Ambiguity Handling

Ambiguity is a common challenge in coreference resolution, especially when multiple entities share similar characteristics. Techniques such as using additional contextual information, leveraging external knowledge bases, and employing deep learning models contribute to better ambiguity handling.

Domain Adaptation

Coreference resolution models may struggle with domain-specific language and references. Domain adaptation techniques, including transfer learning and fine-tuning on domain-specific data, help enhance performance across diverse domains.

Evaluation Metrics

Measuring the effectiveness of coreference resolution systems requires appropriate evaluation metrics. Metrics like Precision, Recall, and the F1 score are commonly used to assess the system's ability to correctly identify and link coreferent expressions.

7.4 Tables and Graphs

Evaluation Metrics Table

Metric	Description
Precision	Ability to correctly identify coreferent expressions
Recall	Coverage of all actual coreferent expressions
F1 Score	Balance between precision and recall

Table 7.1: Coreference Resolution Evaluation Metrics

tab: coref_metrics

Case Studies Table

Domain	Application
News Articles	Improving coherence in news reports
Academic Literature	Enhancing the understanding of research papers
Conversational Data	Facilitating natural language understanding in chatbots

Table 7.2: Coreference Resolution Case Studies

tab:coref_case_ studies

CHAPTER 8

Recognition of Temporal Expressions and Normalization

8.1 Introduction

Temporal expressions in text play a crucial role in understanding the timeline of events and actions. Recognizing and normalizing these expressions are essential components of information extraction, contributing to a more structured representation of temporal information.

Temporal Expression Recognition

Temporal expression recognition involves identifying and extracting expressions in text that denote specific points or intervals in time. This includes dates, durations, and temporal relationships. Accurate recognition is vital for understanding the temporal context of information.

Normalization Process

The normalization process transforms recognized temporal expressions into a standardized format, facilitating consistent representation and comparison. Normalization may involve converting dates to a common calendar, expressing durations in a unified unit, and resolving relative temporal references.

Practical Applications

The recognition and normalization of temporal expressions find applications in various real-world scenarios, enhancing the extraction of temporal information from textual data.

8.2 Temporal Expression Recognition Techniques

Rule-Based Approaches

Rule-based approaches use predefined patterns and linguistic rules to identify and extract temporal expressions. These rules may consider language-specific date formats, temporal words, and contextual information.

Machine Learning Methods

Machine learning methods leverage annotated data to train models for automatic temporal expression recognition. Sequence labeling models, such as Conditional Random Fields (CRF) and Recurrent Neural Networks (RNNs), have shown effectiveness in capturing temporal patterns.

Hybrid Approaches

Hybrid approaches combine rule-based and machine learning methods to improve robustness and adaptability. These approaches benefit from the precision of rule-based systems and the generalization capabilities of machine learning models.

Challenges in Temporal Expression Recognition

Recognizing temporal expressions poses challenges, including language variations, contextual ambiguity, and the diversity of temporal formats. Addressing these challenges requires a combination of linguistic knowledge, contextual analysis, and machine learning techniques.

8.3 Normalization Techniques

Calendar Normalization

Calendar normalization involves converting recognized dates into a standardized calendar format. This ensures consistency and facilitates chronological ordering of events.

Duration Standardization

The standardization of durations involves expressing temporal intervals in a common unit, such as days, hours, or minutes. This simplifies comparisons and computations involving different time spans.

Relative Temporal References

Resolving relative temporal references involves establishing a clear timeline for events mentioned in text. This includes handling expressions like "yesterday," "tomorrow," or "next week" to determine their absolute temporal counterparts.

8.4 Practical Applications

Event Timeline Construction

Temporal expression recognition and normalization contribute to the construction of event timelines. This is particularly useful in scenarios such as historical document analysis, news event tracking, and project management.

Financial Forecasting

In financial texts, recognizing and normalizing temporal expressions are crucial for analyzing trends, predicting market movements, and understanding the temporal aspects of financial data.

Healthcare Record Analysis

In healthcare, temporal expression recognition facilitates the analysis of patient records, treatment timelines, and the coordination of medical events. Normalization ensures consistency in representing temporal information.

Legal Document Processing

Legal documents often contain time-sensitive information. Temporal expression recognition and normalization assist in organizing case timelines, court proceedings, and legal deadlines.

Natural Language Interfaces

Recognition and normalization of temporal expressions contribute to the development of natural language interfaces. Systems that understand and respond to temporal queries provide a more user-friendly experience.

8.5 Ontology Classification

Temporal Ontology Classification

Table 8.1 provides a classification of temporal concepts in ontologies.

Category	Temporal Concept
Instant	Point in time
Interval	Time duration
Time Point Set	Set of related time points
Time Interval Set	Set of related time intervals

Table 8.1: Temporal Ontology Classification

b:temporal_
ontology

CHAPTER 9

Pattern Extraction

9.1 Introduction to Pattern Extraction

Pattern extraction is a critical process in information extraction, where the goal is to identify regularities, structures, or trends from unstructured or semi-structured data. Its applications span various domains, including text mining, natural language processing (NLP), image recognition, and data mining.

Understanding the Concept

Pattern extraction involves discovering repeated structures within data, which can be leveraged to gain meaningful insights. For instance, in text mining, it identifies frequent terms or semantic patterns within large document corpora. In image recognition, it detects shapes and textures. Similarly, in time-series analysis, it identifies trends and periodicities for forecasting or anomaly detection.

Importance in Information Extraction

Pattern extraction is indispensable in processing and interpreting vast amounts of unstructured data. It enhances systems' ability to extract valuable information, which improves classification, clustering, and summarization. For example, NLP tasks such as named entity recognition (NER) rely on pattern extraction to identify entities like names or locations in texts.

9.2 Techniques of Pattern Extraction

Several techniques are available depending on the data type being analyzed. Below are some widely-used approaches:

- **Regular Expression Matching:** Utilizes predefined patterns to find specific structures, such as dates or email addresses, within text.
- **N-gram Modeling:** Extracts frequent sequences of words or characters for applications such as text classification.
- **Clustering and Classification Algorithms:** Machine learning techniques like K-means or decision trees help group or classify data based on patterns.
- **Frequent Pattern Mining:** Methods such as Apriori and FP-Growth discover frequent itemsets in datasets, particularly in transactional data.

9.3 Application Domains of Pattern Extraction

Pattern extraction finds usage across various fields:

- **Natural Language Processing (NLP):** Used in tasks like part-of-speech tagging and text classification.
- **Image and Video Processing:** Helps in recognizing textures, edges, and objects in visual data.
- **Bioinformatics:** Used to find motifs and conserved sequences in genomic data.
- **Market Basket Analysis:** Frequent pattern mining reveals purchasing trends and optimizes recommendation systems.

9.4 Examples of Pattern Extraction

Below are practical examples of how pattern extraction can be applied:

- **Customer Review Analysis:** Extracting frequent co-occurrences of adjectives with product names to analyze customer sentiment.
- **Social Media Trend Analysis:** Extracting patterns from hashtags to identify trending topics.
- **Medical Data Analysis:** Detecting patterns in patient records to predict diagnoses based on symptoms.

9.5 Tables and Graphs for Visualization

Table 9.1 summarizes various pattern extraction techniques, their applications, and benefits.

tab: pattern_examples

Table 9.1: Summary of Pattern Extraction Techniques and Applications

Technique	Application Domain	Benefit
Regular Expression Matching	Text Mining	Extracts structured information from text
N-gram Modeling	NLP	Improves text classification by identifying frequent word sequences
Clustering	Data Mining	Groups similar data points for segmentation
Frequent Pattern Mining	Market Basket Analysis	Reveals frequent itemsets for understanding buying patterns

9.6 Conclusion

Pattern extraction enables efficient identification of meaningful structures from unstructured datasets. With a broad range of techniques, from regular expressions to machine learning, its applications are far-reaching, from NLP to market analysis and bioinformatics.

9.7 Pattern Recognition Techniques

Pattern recognition refers to the automated discovery of patterns and regularities in data. Several approaches can be employed, depending on the nature of the data and the complexity of the patterns being analyzed.

Rule-Based Approaches

Rule-based pattern recognition involves manually defining explicit rules and conditions to identify patterns within data. This method works well when patterns can be precisely characterized using clear logic. For example, rules can be defined using regular expressions to match specific text formats, such as phone numbers or email addresses. Rule-based systems are straightforward but may struggle with complex or noisy data.

:rule_based_techniques

Table 9.2: Overview of Rule-Based Pattern Recognition

Technique	**Example**	**Advantage**
Regular Expressions	Identifying dates, emails	Simple and efficient for structured patterns
Decision Rules	Classification based on specific conditions	Works well for well-defined and simple patterns
Conditional Statements	Text matching with if-else logic	Highly interpretable, easy to modify

Statistical Methods

Statistical methods rely on probabilistic models to recognize patterns by analyzing the distribution and correlation of data points. Common techniques include:

- **Clustering:** Groups similar data points together based on their statistical properties, such as K-means clustering.
- **Regression Analysis:** Identifies patterns by modeling the relationship between dependent and independent variables.
- **Bayesian Networks:** Uses probability distributions to predict the likelihood of various outcomes based on observed data.

Machine Learning Approaches

Machine learning methods are increasingly employed for pattern recognition, especially when dealing with large and complex datasets. These methods automatically learn patterns from

tab:statistical_methods

Table 9.3: Statistical Pattern Recognition Techniques

Technique	Application	Advantage
Clustering (e.g., K-means)	Grouping similar data points	Effective for unsupervised learning
Regression Analysis	Predicting relationships between variables	Suitable for continuous data
Bayesian Networks	Modeling probabilistic relationships	Deals well with uncertainty

annotated data (i.e., training data) and generalize these patterns to new, unseen data. Some popular machine learning algorithms include:

- **Decision Trees:** Use a tree-like structure to split data into distinct classes or categories based on learned decision rules.
- **Support Vector Machines (SVM):** Classifies data by finding the hyperplane that best separates different classes in a high-dimensional space.
- **Neural Networks:** Particularly deep neural networks (DNNs), mimic the structure of the human brain to recognize highly complex patterns in data, especially in tasks like image or speech recognition.

tab:ml_methods

Table 9.4: Machine Learning Pattern Recognition Techniques

Algorithm	Application	Advantage
Decision Trees	Classification tasks (e.g., email spam detection)	Easy to interpret, handles non-linear data
Support Vector Machines	High-dimensional classification (e.g., image classification)	Effective in high-dimensional spaces
Neural Networks	Complex pattern recognition (e.g., speech or image recognition)	Capable of capturing non-linear relationships

Text Mining Techniques

Text mining techniques are particularly useful for extracting patterns from unstructured text data, such as social media posts, customer reviews, or news articles. Key methods include:

- **Natural Language Processing (NLP):** A field of AI that focuses on the interaction between computers and human languages. NLP techniques are used for tasks such as tokenization, part-of-speech tagging, and named entity recognition (NER).
- **Sentiment Analysis:** A specialized text mining technique used to extract the sentiment (positive, negative, or neutral) from textual data, often employed in customer feedback analysis or social media monitoring.

- **Topic Modeling:** Discovers latent topics within a set of documents. Techniques like Latent Dirichlet Allocation (LDA) help in summarizing large corpora by identifying major themes.

Table 9.5: Text Mining Pattern Recognition Techniques

Technique	Application	Advantage
Natural Language Processing (NLP)	Named entity recognition (NER), machine translation	Handles unstructured text data efficiently
Sentiment Analysis	Opinion mining from social media posts	Extracts sentiment insights from large datasets
Topic Modeling (LDA)	Discovering topics in documents or reviews	Summarizes large corpora effectively

9.8 Conclusion

Pattern recognition techniques, ranging from rule-based systems to advanced machine learning models, are essential for extracting meaningful patterns from diverse datasets. The choice of technique depends on the nature of the data, the complexity of the patterns, and the desired outcomes.

9.9 Pattern Recognition Tables

Technique	Advantages	Disadvantages	Applications
Rule-Based	Precision, Explicit rules	Limited adaptability	Information retrieval
Statistical	Robust to noise, Data-driven	Sensitive to outliers	Market trend analysis
Machine Learning	Adaptability, Automation	Requires labeled data	Sentiment analysis
Text Mining	Language understanding, Contextual analysis	Complexity in linguistic nuances	Healthcare data analysis

Table 9.6: Comparison of Pattern Recognition Techniques

Application	Data Source	Patterns Extracted
Fraud Detection	Financial transactions	Unusual transaction patterns
Market Trend Analysis	Sales data, consumer behavior	Buying patterns, market trends
Sentiment Analysis	Social media, customer reviews	Positive/negative sentiment patterns
Healthcare Data Analysis	Patient records, medical literature	Disease prevalence, treatment patterns

Table 9.7: Patterns Extracted in Various Applications

Challenge	Mitigation
Lack of Labeled Data	Data augmentation, Transfer learning
Overfitting in ML Models	Regularization techniques, Cross-validation
Complexity in Text Mining	Advanced NLP models, Feature engineering
Ambiguity in Rule-Based Systems	Clear rule definition, Regular updates

tab:challenges_mitigations

Table 9.8: Challenges and Mitigations in Pattern Recognition

9.10 Use Cases

Information Retrieval

Pattern extraction plays a critical role in information retrieval systems. By recognizing patterns in user queries or documents, search engines can enhance the relevance and accuracy of results. For instance, recognizing patterns in keyword usage or linguistic structures helps match user intent with relevant documents more effectively.

Sentiment Analysis

In sentiment analysis, pattern extraction is used to detect linguistic patterns that indicate sentiment. This process is essential for analyzing public opinion, especially in social media or customer reviews. Patterns such as frequently co-occurring adjectives and product names help determine whether the sentiment is positive, negative, or neutral.

Fraud Detection

Fraud detection systems rely heavily on pattern extraction to identify irregular or anomalous patterns in financial transactions. By extracting these patterns, systems can flag potential fraudulent activities, such as unusual spending behaviors or abnormal transaction sequences, in real-time.

Healthcare Data Analysis

In healthcare, pattern extraction is used to analyze large datasets such as patient records or medical literature. This analysis can reveal trends, correlations, and insights that aid in the diagnosis and treatment of diseases. For example, identifying recurring symptom patterns can lead to early detection of diseases or personalized treatment plans.

Market Trend Analysis

Businesses use pattern extraction to analyze consumer behavior and market trends. By identifying patterns in sales data or customer purchasing habits, companies can forecast future trends, optimize product offerings, and make informed strategic decisions. This technique is particularly useful in market basket analysis and recommendation systems.

9.11 Conclusion

In conclusion, pattern extraction is a versatile and powerful technique in information extraction. By applying various pattern recognition methods, systems can uncover valuable insights from

diverse data sources, enhancing decision-making across a wide range of fields such as information retrieval, sentiment analysis, fraud detection, healthcare, and market analysis. [**Qamar2024**]

CHAPTER 10

Exercise

Exercise 1: Defining Text Mining

Question: Explain the concept of *text mining* in your own words. List the main steps involved in a text mining process. Additionally, provide an example of a practical application of text mining in the healthcare domain.

Solution: Text mining refers to the process of extracting useful and actionable information from large amounts of unstructured or semi-structured textual data. The goal of text mining is to transform raw text into structured data that can be analyzed to uncover patterns, relationships, or trends. It is widely used in fields like sentiment analysis, recommendation systems, and information retrieval.

Main steps in text mining:

- **Text Preprocessing:** Cleaning and preparing raw text, removing punctuation, stop words, and handling special characters. Tokenization, stemming, and lemmatization are also performed.
- **Feature Extraction:** Relevant features (words, phrases, etc.) are extracted from the text. Named entity recognition (NER), keyword extraction, and part-of-speech tagging are common techniques.
- **Text Analysis:** Techniques like classification, clustering, and topic modeling are used to analyze the text and extract meaningful insights.
- **Pattern Discovery:** Machine learning models find patterns and relationships in the text data.
- **Visualization and Interpretation:** Results are visualized and interpreted for decision-making or further analysis.

Example in Healthcare: Text mining can be used in healthcare to analyze electronic health records (EHRs) to extract patterns of drug interactions or predict patient symptoms based on historical data. This helps in improving patient care by identifying potential risk factors earlier.

Exercise 2: Document Features in Text Mining

Question: Given a collection of documents related to financial news, identify the most relevant document features that could be extracted for a text mining system. Explain why each feature is important for analyzing financial news.

Solution: In a text mining system for financial news, the following features would be essential:

- **Named Entities (e.g., companies, persons, locations):** Identifying names of companies (e.g., *Apple, Tesla*) and individuals (e.g., *CEO, investors*) is crucial because they often play central roles in financial events.
- **Dates and Time Expressions:** Extracting dates allows for tracking events over time, such as market crashes or quarterly earnings reports.
- **Numerical Values (e.g., stock prices, percentages):** Numerical data like stock prices, market indices, and percentage changes are key indicators of financial health and market movements.
- **Sentiment (positive, negative, neutral):** Analyzing sentiment helps gauge the market's reaction to specific news events like an earnings report or product launch.
- **Keywords and Topics:** Extracting keywords and categorizing documents based on topics (e.g., *mergers and acquisitions*, *market analysis*) helps classify and retrieve relevant information quickly.

Explanation: These features are critical because they help analysts and systems generate insights about the market, predict trends, and inform decision-making processes based on textual data from financial reports and news articles.

Exercise 3: Searching for Patterns and Trends

Question: Consider a text mining system that analyzes a large collection of legal documents. Describe how the system would search for patterns and trends in the documents. Provide two examples of patterns or trends that such a system might identify in the legal domain.

Solution: To search for patterns and trends in legal documents, the system would perform the following tasks:

- **Entity Recognition:** The system would identify key entities, such as judges, lawyers, and law firms, helping discover which professionals are frequently mentioned in certain case types.
- **Keyword Analysis:** Analyzing the frequency of specific legal terms (e.g., *contract breach*, *intellectual property*) to detect which legal issues are most common.
- **Text Classification:** Classify the documents into categories (e.g., *criminal law*, *corporate law*) to understand trends in different types of legal cases.
- **Timeline Analysis:** Using dates extracted from the documents, the system could track changes in the frequency of certain case types over time.

Examples of Patterns/Trends:

1. **Rising trend of intellectual property disputes:** The system might identify a growing number of documents mentioning *intellectual property disputes* over the last decade, indicating an increasing trend in this type of litigation.

2. **Judicial Bias:** By analyzing case outcomes and related entities (e.g., judge names), the system could detect that certain judges are more likely to rule in favor of defendants in specific case types.

These patterns can help legal professionals better understand the landscape and prepare for cases based on historical trends and outcomes.

Exercise 1: NER and Classification Accuracy Calculation

Question: A Named Entity Recognition (NER) system processed 100 sentences and identified 50 entities in total. Out of these 50 entities, 40 were correctly classified into their respective categories (e.g., Person, Organization, Location).

Calculate the following:

- Precision
- Recall
- F1-Score

Use the formulas below:

$$\text{Precision} = \frac{TP}{TP + FP}, \quad \text{Recall} = \frac{TP}{TP + FN}, \quad \text{F1-Score} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

Where:

- True Positives (TP) = 40
- False Positives (FP) = 10 (incorrectly classified entities)
- False Negatives (FN) = 20 (entities that were missed by the system)

Solution: Using the given values:

$$\text{Precision} = \frac{40}{40 + 10} = \frac{40}{50} = 0.8$$

$$\text{Recall} = \frac{40}{40 + 20} = \frac{40}{60} \approx 0.6667$$

$$\text{F1-Score} = \frac{2 \times 0.8 \times 0.6667}{0.8 + 0.6667} = \frac{1.0667}{1.4667} \approx 0.727$$

Results:

- Precision: 0.8 (80%)
- Recall: 0.6667 (66.67%)
- F1-Score: 0.727 (72.7%)

Exercise 2: Named Entity Recognition (NER) and Entity Classification

Question: Given the sentence *"Sundar Pichai became CEO of Google in 2015, based in Mountain View, California."*, apply Named Entity Recognition (NER) to identify and classify the following entities:

- *Sundar Pichai*
- *Google*
- *2015*
- *Mountain View*
- *California*

For each identified entity, also calculate its entity classification accuracy assuming the system classified 4 out of 5 correctly. Use the formula for accuracy:

$$\text{Accuracy} = \frac{\text{Correct Classifications}}{\text{Total Entities}}$$

Solution: Using Named Entity Recognition (NER), we classify the entities as follows:

- **Sundar Pichai**: Person
- **Google**: Organization
- **2015**: Date
- **Mountain View**: Location
- **California**: Location

Accuracy Calculation: The system correctly classified 4 out of the 5 entities:

$$\text{Accuracy} = \frac{4}{5} = 0.8 = 80\%$$

Exercise 3: Confusion Matrix and Error Analysis in NER

Question: A Named Entity Recognition system was tested on 100 sentences. It was tasked with identifying three types of entities: *Person*, *Organization*, and *Location*. The confusion matrix below summarizes the system's performance.

Actual / Predicted	**Person**	**Organization**	**Location**
Person	30 (TP)	5 (FP)	3 (FP)
Organization	2 (FP)	25 (TP)	6 (FP)
Location	1 (FP)	4 (FP)	24 (TP)

Table 10.1: Confusion Matrix for NER System

Based on the confusion matrix, calculate the following for the *Person* entity:

- Precision

- Recall
- F1-Score

Use the following formulas:

$$\text{Precision} = \frac{TP}{TP + FP}, \quad \text{Recall} = \frac{TP}{TP + FN}, \quad \text{F1-Score} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

Solution: For the *Person* entity, from the confusion matrix:

- True Positives (TP) = 30
- False Positives (FP) = 5 + 3 = 8 (Person misclassified as Organization or Location)
- False Negatives (FN) = 2 + 1 = 3 (Person entities misclassified as others)

$$\text{Precision} = \frac{30}{30 + 8} = \frac{30}{38} \approx 0.789$$
$$\text{Recall} = \frac{30}{30 + 3} = \frac{30}{33} \approx 0.909$$
$$\text{F1-Score} = \frac{2 \times 0.789 \times 0.909}{0.789 + 0.909} = \frac{1.434}{1.698} \approx 0.844$$

Results:

- Precision: 0.789 (78.9%)
- Recall: 0.909 (90.9%)
- F1-Score: 0.844 (84.4%)

10.1 Exercise: Pattern Extraction

Instructions: Answer the following questions based on your understanding of pattern extraction techniques and their applications. Some questions involve calculations or statistical analysis.

Questions

1. **(Understanding)** Define pattern extraction and explain its significance in natural language processing (NLP).
2. **(Regular Expressions)** Write a regular expression to extract email addresses from a text document.
3. **(N-grams Calculation)** Given the sentence: *"Pattern extraction is essential in data mining."*, compute all the bi-grams.
4. **(Clustering)** A data set has 6 data points: $\{(1,1),(2,1),(4,3),(5,4),(3,2),(5,5)\}$. Use the K-means clustering algorithm with $k = 2$. Calculate the initial centroids and assign each point to a cluster based on Euclidean distance.
5. **(Classification)** Explain how decision trees can be used in pattern extraction for text classification. Illustrate with an example.

6. **(Frequent Pattern Mining)** In a market basket analysis, the following transactions are recorded:

   - Transaction 1: $\{bread, milk, butter\}$
   - Transaction 2: $\{bread, butter\}$
   - Transaction 3: $\{milk, butter\}$
   - Transaction 4: $\{bread, milk\}$

   Using the Apriori algorithm, identify the frequent itemsets with a minimum support of 50%.

7. **(Statistical Analysis)** In a survey about customer reviews, 70% of respondents mention the product quality and 40% mention the delivery service. If 20% of the respondents mention both, calculate the probability that a respondent mentions either product quality or delivery service.

8. **(Pattern Recognition)** Describe a use case where neural networks are more suitable for pattern extraction compared to traditional methods like regular expressions. Provide a justification.

9. **(Evaluation Metric)** If a clustering algorithm assigns 100 data points to clusters, and 75 of them are correctly clustered, calculate the accuracy of the clustering.

10. **(Advanced NLP)** Perform named entity recognition (NER) on the following sentence: *"Microsoft was founded by Bill Gates in the United States."* Identify the entities and their types.

Solutions

1. **Pattern extraction** is the process of identifying repeated structures or trends in unstructured data. In NLP, it helps in recognizing patterns like frequently used terms or syntactic structures, which is crucial for tasks like named entity recognition or part-of-speech tagging.

2. The regular expression for extracting email addresses could be:

   ```
   [a-zA-Z0-9+_.-]+@[a-zA-Z0-9.-]+\.[a-zA-Z]{2,}
   ```

   The regular expression extracts valid email addresses by matching letters, numbers, and special characters like '+', '.' or '-'.

3. The bi-grams for the sentence *"Pattern extraction is essential in data mining."* are:

   ```
   {(Pattern, extraction), (extraction, is), (is, essential),
   (essential, in), (in, data), (data, mining)}
   ```

   Bi-grams are contiguous sequences of two words from the given sentence.

4. For the K-means clustering with $k = 2$ using the data points:

- Initial centroids: Let the centroids be $C_1 = (1, 1)$ and $C_2 = (5, 5)$.
- Distance calculations:

$$d((1, 1), C_1) = 0, \quad d((1, 1), C_2) = 5.66$$

$$d((5, 4), C_1) = 5, \quad d((5, 4), C_2) = 1$$

Based on the distances, points $(1, 1)$ and $(2, 1)$ are assigned to cluster 1, and points $(5, 4)$ and $(5, 5)$ are assigned to cluster 2.

5. A decision tree for text classification could split data based on the occurrence of specific keywords. For example, a decision node might ask, "Does the text contain the word 'positive'?" to classify text as expressing positive sentiment.

6. Using the Apriori algorithm:

   - Frequent itemsets with support $\geq 50\%$ are: $\{bread\}$, $\{milk\}$, $\{butter\}$, and $\{bread, butter\}$.

7. Using the formula for the union of two events:

$$P(A \cup B) = P(A) + P(B) - P(A \cap B)$$

$$P(A \cup B) = 0.70 + 0.40 - 0.20 = 0.90$$

   The probability is 90%.

8. Neural networks are suitable for tasks like image recognition, where complex, high-dimensional patterns must be identified. Traditional methods like regular expressions fail in such cases because the patterns are not explicitly defined but learned from data.

9. Clustering accuracy is calculated as:

$$\text{Accuracy} = \frac{\text{Correctly clustered points}}{\text{Total points}} = \frac{75}{100} = 0.75$$

   So, the accuracy is 75%.

10. The named entities in the sentence *"Microsoft was founded by Bill Gates in the United States."* are:

    - **Microsoft**: Organization
    - **Bill Gates**: Person
    - **United States**: Location

10.2 Exercise: Frequent Pattern Mining and Association Rules

Instructions: In this exercise, you are provided with a set of transactions from a market basket. Use the Apriori algorithm to extract frequent patterns and calculate the necessary metrics to complete the table.

Transactions

Consider the following transactions in a store:

Transaction ID	Items
1	$\{bread, milk, butter\}$
2	$\{bread, butter\}$
3	$\{milk, butter\}$
4	$\{bread, milk\}$
5	$\{milk\}$
6	$\{bread, butter\}$
7	$\{bread, milk, butter\}$
8	$\{bread, milk\}$
9	$\{milk, butter\}$
10	$\{bread, butter\}$

Questions

Using the Apriori algorithm, calculate the support, confidence, and lift for the following itemsets and rules, and fill in the missing values in the table below.

Itemsets and Rules

Itemset / Rule	Support (%)	Confidence (%)	Lift
$\{bread\}$		N/A	N/A
$\{milk\}$		N/A	N/A
$\{butter\}$		N/A	N/A
$\{bread, butter\}$		N/A	N/A
$\{milk, butter\}$		N/A	N/A
$\{bread\} \Rightarrow \{butter\}$			
$\{milk\} \Rightarrow \{butter\}$			
$\{bread, milk\} \Rightarrow \{butter\}$			

Formulas to use:

- **Support:** $\text{Support}(X) = \frac{\text{Transactions containing } X}{\text{Total Transactions}}$
- **Confidence:** $\text{Confidence}(X \Rightarrow Y) = \frac{\text{Transactions containing both } X \text{ and } Y}{\text{Transactions containing } X}$
- **Lift:** $\text{Lift}(X \Rightarrow Y) = \frac{\text{Confidence}(X \Rightarrow Y)}{\text{Support}(Y)}$

Solution

Step 1: Calculate the Support for Each Itemset

Support is the proportion of transactions that contain the itemset. First, count how many transactions contain each item or itemset, then divide by the total number of transactions, which is 10.

- {bread}: Appears in 7 transactions (1, 2, 4, 6, 7, 8, 10).
  Support for {bread}:
  $$\frac{7}{10} = 0.7 \text{ or } 70\%$$
- {milk}: Appears in 6 transactions (1, 3, 4, 5, 7, 8).
  Support for {milk}:
  $$\frac{6}{10} = 0.6 \text{ or } 60\%$$
- {butter}: Appears in 6 transactions (1, 2, 3, 6, 7, 9, 10).
  Support for {butter}:
  $$\frac{6}{10} = 0.6 \text{ or } 60\%$$
- {bread, butter}: Appears in 5 transactions (1, 2, 6, 7, 10).
  Support for {bread, butter}:
  $$\frac{5}{10} = 0.5 \text{ or } 50\%$$
- {milk, butter}: Appears in 4 transactions (1, 3, 7, 9).
  Support for {milk, butter}:
  $$\frac{4}{10} = 0.4 \text{ or } 40\%$$

Step 2: Calculate Confidence for Each Rule
Confidence is the conditional probability that an itemset Y is present, given that itemset X is present.

- {bread} $\Rightarrow$ {butter}: The rule asks, "Given that bread is present, how often is butter also present?"
  Confidence:
  $$\frac{\text{Transactions containing both bread and butter}}{\text{Transactions containing bread}} = \frac{5}{7} = 0.71 \text{ or } 71\%$$
- {milk} $\Rightarrow$ {butter}: The rule asks, "Given that milk is present, how often is butter also present?"
  Confidence:
  $$\frac{\text{Transactions containing both milk and butter}}{\text{Transactions containing milk}} = \frac{4}{6} = 0.67 \text{ or } 67\%$$
- {bread, milk} $\Rightarrow$ {butter}: The rule asks, "Given that both bread and milk are present, how often is butter also present?"
  Confidence:
  $$\frac{\text{Transactions containing bread, milk, and butter}}{\text{Transactions containing bread and milk}} = \frac{2}{4} = 0.5 \text{ or } 50\%$$

Step 3: Calculate Lift for Each Rule
Lift measures how much more likely Y is to appear given X, compared to how often Y appears generally.

- {bread} ⇒ {butter}:
  Lift:
  $$\frac{\text{Confidence of bread} \Rightarrow \text{butter}}{\text{Support of butter}} = \frac{0.71}{0.6} = 1.18$$
- {milk} ⇒ {butter}:
  Lift:
  $$\frac{\text{Confidence of milk} \Rightarrow \text{butter}}{\text{Support of butter}} = \frac{0.67}{0.6} = 1.12$$
- {bread, milk} ⇒ {butter}:
  Lift:
  $$\frac{\text{Confidence of bread, milk} \Rightarrow \text{butter}}{\text{Support of butter}} = \frac{0.5}{0.6} = 0.83$$

Final Table

Itemset / Rule	Support (%)	Confidence (%)	Lift
$\{bread\}$	70	N/A	N/A
$\{milk\}$	60	N/A	N/A
$\{butter\}$	60	N/A	N/A
$\{bread, butter\}$	50	N/A	N/A
$\{milk, butter\}$	40	N/A	N/A
$\{bread\} \Rightarrow \{butter\}$	50	71	1.18
$\{milk\} \Rightarrow \{butter\}$	40	67	1.12
$\{bread, milk\} \Rightarrow \{butter\}$	20	50	0.83

Exercise 3: Apriori Algorithm for Frequent Itemsets

Question: Using the following set of transactions, find the frequent itemsets using the **Apriori Algorithm** with a minimum support threshold of 50%:

Transaction 1: {A, B, D}
Transaction 2: {B, C, E}
Transaction 3: {A, B, D, E}
Transaction 4: {A, C, D}
Transaction 5: {B, D, E}
Transaction 6: {A, B, D, C}

Find all frequent itemsets (single, pair, and triple) that meet the minimum support threshold. Use the support formula:

$$\text{Support of an itemset} = \frac{\text{Number of transactions containing the itemset}}{\text{Total number of transactions}}$$

Solution: 1. **Step 1: Counting support for individual items:**

- A appears in Transactions 1, 3, 4, and 6 $\rightarrow$ 4 occurrences
- B appears in Transactions 1, 2, 3, 5, and 6 $\rightarrow$ 5 occurrences
- C appears in Transactions 2, 4, and 6 $\rightarrow$ 3 occurrences
- D appears in Transactions 1, 3, 4, 5, and 6 $\rightarrow$ 5 occurrences
- E appears in Transactions 2, 3, and 5 $\rightarrow$ 3 occurrences

The total number of transactions is 6, and we calculate the support for each item:

- Support of $A = \frac{4}{6} \approx 0.67$ (frequent)
- Support of $B = \frac{5}{6} \approx 0.83$ (frequent)
- Support of $C = \frac{3}{6} = 0.5$ (frequent)
- Support of $D = \frac{5}{6} \approx 0.83$ (frequent)
- Support of $E = \frac{3}{6} = 0.5$ (frequent)

2. **Step 2: Counting support for pairs (itemsets of size 2):**

- Support of *{A, B}* = Transactions 1, 3, 6 $\rightarrow \frac{3}{6} = 0.5$
- Support of *{A, D}* = Transactions 1, 3, 4, 6 $\rightarrow \frac{4}{6} \approx 0.67$
- Support of *{B, D}* = Transactions 1, 3, 5, 6 $\rightarrow \frac{4}{6} \approx 0.67$
- Support of *{B, E}* = Transactions 2, 3, 5 $\rightarrow \frac{3}{6} = 0.5$
- Support of *{D, E}* = Transactions 3, 5 $\rightarrow \frac{2}{6} \approx 0.33$ (not frequent)

3. **Step 3: Counting support for triples (itemsets of size 3):**

- Support of *{A, B, D}* = Transactions 1, 3, 6 $\rightarrow \frac{3}{6} = 0.5$
- Support of *{A, B, E}* = Transaction 3 $\rightarrow \frac{1}{6} \approx 0.17$ (not frequent)
- Support of *{B, D, E}* = Transaction 3, 5 $\rightarrow \frac{2}{6} = 0.33$ (not frequent)

4. **Frequent itemsets:**

- Single: *A, B, C, D, E*
- Pairs: *{A, B}, {A, D}, {B, D}, {B, E}*
- Triples: *{A, B, D}*

Conclusion: The frequent itemsets include individual items, pairs like *{A, B}, {B, E}*, and one triple *{A, B, D}* with minimum support of 50%.

Exercise 4: Calculating Lift for Association Rules

Question: Given the frequent itemset {Bread, Milk} from the following transactions:

Transaction 1: {Bread, Milk, Cheese}
Transaction 2: {Bread, Diaper, Beer, Milk}
Transaction 3: {Milk, Diaper, Beer, Cheese}
Transaction 4: {Bread, Milk, Diaper, Beer}
Transaction 5: {Bread, Milk, Cheese}

Calculate the **lift** of the rule *Bread* $\Rightarrow$ *Milk*, using the following formulas:

$$\text{Support of (X)} = \frac{\text{Transactions containing X}}{\text{Total transactions}}$$

$$\text{Lift (X} \Rightarrow \text{Y)} = \frac{\text{Support of (X} \cap \text{Y)}}{\text{Support of (X)} \times \text{Support of (Y)}}$$

Solution: 1. **Support of Bread**: *Bread* appears in Transactions 1, 2, 4, and 5. Therefore:

$$\text{Support of Bread} = \frac{4}{5} = 0.8$$

2. **Support of Milk**: *Milk* appears in Transactions 1, 2, 3, 4, and 5. Therefore:

$$\text{Support of Milk} = \frac{5}{5} = 1.0$$

3. **Support of (Bread $\cap$ Milk)**: *Bread* and *Milk* co-occur in Transactions 1, 2, 4, and 5. Therefore:

$$\text{Support of (Bread} \cap \text{Milk)} = \frac{4}{5} = 0.8$$

4. **Lift Calculation**:

$$\text{Lift (Bread} \Rightarrow \text{Milk)} = \frac{\text{Support of (Bread} \cap \text{Milk)}}{\text{Support of (Bread)} \times \text{Support of (Milk)}} = \frac{0.8}{0.8 \times 1.0} = \frac{0.8}{0.8} = 1.0$$

Conclusion: The lift of the rule *Bread* $\Rightarrow$ *Milk* is 1.0, indicating that *Bread* and *Milk* are independent of each other, as the lift is exactly 1.0.

Exercise 5: Confidence and Lift in Association Rules

Question: Consider the following set of transactions from a grocery store:

Transaction 1: {Milk, Bread, Butter}
Transaction 2: {Milk, Bread}
Transaction 3: {Bread, Butter}
Transaction 4: {Milk, Butter}
Transaction 5: {Milk, Bread, Butter}

Using the data above, calculate the following metrics for the association rule *Milk* $\Rightarrow$ *Butter*:

- Support of *Milk* and *Butter*
- Confidence of the rule

- Lift of the rule

Use the following formulas:

$$\text{Confidence } (\text{X} \Rightarrow \text{Y}) = \frac{\text{Support of } (\text{X} \cap \text{Y})}{\text{Support of } (\text{X})}$$

$$\text{Lift } (\text{X} \Rightarrow \text{Y}) = \frac{\text{Support of } (\text{X} \cap \text{Y})}{\text{Support of } (\text{X}) \times \text{Support of } (\text{Y})}$$

Solution:

1. **Support of (Milk ∩ Butter)**: *Milk* and *Butter* co-occur in Transactions 1, 4, and 5. Therefore:

$$\text{Support of } (\text{Milk} \cap \text{Butter}) = \frac{3}{5} = 0.6$$

2. **Support of Milk**: *Milk* appears in Transactions 1, 2, 4, and 5. Therefore:

$$\text{Support of Milk} = \frac{4}{5} = 0.8$$

3. **Support of Butter**: *Butter* appears in Transactions 1, 3, 4, and 5. Therefore:

$$\text{Support of Butter} = \frac{4}{5} = 0.8$$

4. **Confidence of the rule**:

$$\text{Confidence } (\text{Milk} \Rightarrow \text{Butter}) = \frac{\text{Support of } (\text{Milk} \cap \text{Butter})}{\text{Support of } (\text{Milk})} = \frac{0.6}{0.8} = 0.75$$

The confidence of the rule is 75%.

5. **Lift of the rule**:

$$\text{Lift } (\text{Milk} \Rightarrow \text{Butter}) = \frac{\text{Support of } (\text{Milk} \cap \text{Butter})}{\text{Support of } (\text{Milk}) \times \text{Support of } (\text{Butter})} = \frac{0.6}{0.8 \times 0.8} = \frac{0.6}{0.64} \approx 0.9375$$

The lift of the rule is approximately 0.9375, which means that the occurrence of *Milk* does not significantly increase the likelihood of *Butter* being bought together, as the lift is close to 1.

Conclusion: The rule *Milk* ⇒ *Butter* has a confidence of 75% and a lift of approximately 0.9375, indicating that the two items are weakly associated.

Exercise 6: Leverage and Conviction for Association Rules

Question: For the transactions given in the previous exercise, calculate the **Leverage** and **Conviction** of the rule *Milk* ⇒ *Butter*. Use the following formulas:

$$\text{Leverage } (\text{X} \Rightarrow \text{Y}) = \text{Support of } (\text{X} \cap \text{Y}) - (\text{Support of } (\text{X}) \times \text{Support of } (\text{Y}))$$

$$\text{Conviction } (\text{X} \Rightarrow \text{Y}) = \frac{1 - \text{Support of } (\text{Y})}{1 - \text{Confidence } (\text{X} \Rightarrow \text{Y})}$$

Solution:

1. **Leverage Calculation:** Using the values from the previous exercise:

$$\text{Leverage } (\text{Milk} \Rightarrow \text{Butter}) = 0.6 - (0.8 \times 0.8) = 0.6 - 0.64 = -0.04$$

The leverage of the rule is -0.04. A negative leverage indicates that *Milk* and *Butter* co-occur less often than would be expected if they were independent.

2. **Conviction Calculation:** Using the confidence from the previous exercise:

$$\text{Conviction (Milk} \Rightarrow \text{Butter)} = \frac{1 - 0.8}{1 - 0.75} = \frac{0.2}{0.25} = 0.8$$

The conviction of the rule is 0.8. Conviction measures the implication strength of the rule, and values closer to 1 indicate weaker implications.

Conclusion: The rule *Milk* $\Rightarrow$ *Butter* has a leverage of -0.04, suggesting a negative association, and a conviction of 0.8, indicating a weak association between the two items.

Exercise 7: Calculating Term Frequency (TF)

Question: Consider the following three documents in a small text corpus:

Document 1: "Data science is the future of technology."
Document 2: "Big data is driving modern technology."
Document 3: "Data science and big data are transforming technology."

Calculate the **Term Frequency (TF)** of the term "data" in each document using the formula:

$$\text{TF} = \frac{\text{Number of times term t appears in document}}{\text{Total number of terms in document}}$$

Solution:

1. **Document 1: "Data science is the future of technology."**

- Number of occurrences of "data": 1
- Total number of words: 7

$$\text{TF for "data"} = \frac{1}{7} \approx 0.143$$

2. **Document 2: "Big data is driving modern technology."**

- Number of occurrences of "data": 1
- Total number of words: 6

$$\text{TF for "data"} = \frac{1}{6} \approx 0.167$$

3. **Document 3: "Data science and big data are transforming technology."**

- Number of occurrences of "data": 2
- Total number of words: 9

$$\text{TF for "data"} = \frac{2}{9} \approx 0.222$$

Conclusion: The **Term Frequency (TF)** of "data" is:

- Document 1: 0.143
- Document 2: 0.167
- Document 3: 0.222

Exercise 8: Calculating TF-IDF for a Term

Question: Using the same three documents from the previous exercise, calculate the **TF-IDF (Term Frequency-Inverse Document Frequency)** for the term "data" in each document. The formula for **IDF** is:

$$\text{IDF} = \log\left(\frac{\text{Total number of documents}}{\text{Number of documents containing the term}}\right)$$

And the formula for **TF-IDF** is:

$$\text{TF-IDF} = \text{TF} \times \text{IDF}$$

Solution:
1. **Step 1: Calculate Inverse Document Frequency (IDF) for "data":**

- The term "data" appears in all three documents (Document 1, Document 2, and Document 3).
- Total number of documents: 3
- Number of documents containing "data": 3

$$\text{IDF for "data"} = \log\left(\frac{3}{3}\right) = \log(1) = 0$$

Since the term appears in all documents, the **IDF** for "data" is 0.
2. **Step 2: Calculate TF-IDF for "data" in each document:**

- Document 1:
$$\text{TF-IDF for "data"} = 0.143 \times 0 = 0$$
- Document 2:
$$\text{TF-IDF for "data"} = 0.167 \times 0 = 0$$
- Document 3:
$$\text{TF-IDF for "data"} = 0.222 \times 0 = 0$$

Conclusion: Since the term "data" appears in all documents, its **IDF** value is 0, making the **TF-IDF** for "data" 0 in all documents.

Exercise 9: Calculating TF-IDF with a Rare Term

Question: Suppose a new term "modern" appears only in Document 2 from the previous exercise. Calculate the **TF-IDF** for the term "modern" in Document 2 using the following values:

- Total number of documents: 3
- Number of documents containing "modern": 1
- Term Frequency (TF) of "modern" in Document 2: $\frac{1}{6} \approx 0.167$

Solution:
1. **Step 1: Calculate IDF for "modern":**

$$\text{IDF for "modern"} = \log\left(\frac{3}{1}\right) = \log(3) \approx 1.0986$$

2. **Step 2: Calculate TF-IDF for "modern" in Document 2:**

$$\text{TF-IDF for "modern"} = 0.167 \times 1.0986 \approx 0.1836$$

Conclusion: The **TF-IDF** for the term "modern" in Document 2 is approximately 0.1836, indicating that "modern" is a more significant term in the document since it appears less frequently across the corpus.

Exercise 10: Comparing TF-IDF of Multiple Terms

Question: Using the same documents, calculate the **TF-IDF** for both "science" and "technology" in Document 1. The relevant values are:

- Total number of documents: 3
- "science" appears in Documents 1 and 3.
- "technology" appears in all 3 documents.

Term frequencies:

- TF of "science" in Document 1: $\frac{1}{7} \approx 0.143$
- TF of "technology" in Document 1: $\frac{1}{7} \approx 0.143$

Solution:
1. **Step 1: Calculate IDF values:**

- **IDF for "science"**:

$$\text{IDF for "science"} = \log\left(\frac{3}{2}\right) = \log(1.5) \approx 0.176$$

- **IDF for "technology"**:

$$\text{IDF for "technology"} = \log\left(\frac{3}{3}\right) = \log(1) = 0$$

2. **Step 2: Calculate TF-IDF values:**

- **TF-IDF for "science" in Document 1:**

$$\text{TF-IDF for "science"} = 0.143 \times 0.176 \approx 0.0251$$

- **TF-IDF for "technology" in Document 1:**

$$\text{TF-IDF for "technology"} = 0.143 \times 0 = 0$$

Conclusion: The term "science" has a **TF-IDF** value of approximately 0.0251 in Document 1, whereas "technology" has a **TF-IDF** of 0 because it appears in all documents.

Bibliography

eldman2006 [1] Feldman, R., & Sanger, J. (2006). *The Text Mining Handbook: Advanced Approaches in Analyzing Unstructured Data*. Cambridge University Press. https://doi.org/10.1017/CBO9780511546914

Jo2019 [2] Jo, T. (2019). *Text Mining*. Studies in Big Data, 45.

George2022 [3] George, A. (2022). *Python Text Mining: Perform Text Processing, Word Embedding, Text Classification and Machine Translation (English Edition)*. BPB Publications.

Qamar2024 [4] Qamar, U., & Raza, M. S. (2024). *Applied Text Mining*.