Mahavir Education Trust's
SHAH & ANCHOR
KUTCHHI ENGINEERING COLLEGE
CHEMBUR, MUMBAI-400088

Chapter 4
Text Analytics

Ms. Tina Maru
Assistant Professor
Artificial Intelligence and Data Science Department

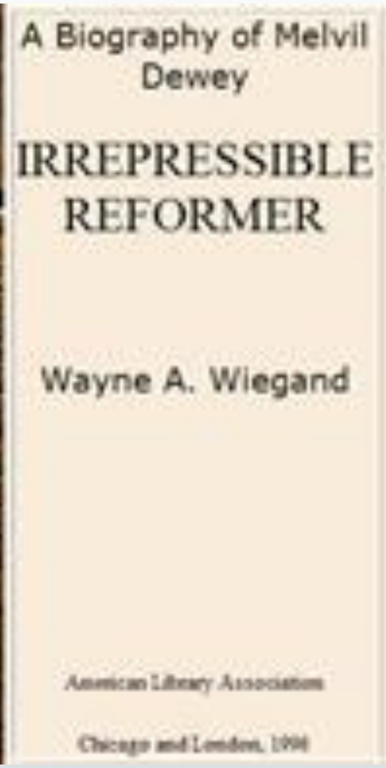
History of text mining

- The problem of accessing required information buried in the accumulation of vast quantities of text stored in various forms, was challenge.
- Two similar processes are used to access information from a series of documents.
  - The first is **INFORMATION RETRIEVAL**, which distinguishes a set of documents as an answer to a logical query using a set of key words, possibly augmented by a thesaurus.
  - The second process is **INFORMATION EXTRACTION**, which aims to extract from documents specific information that is then analyzed for trends or other data.
- This problem consisted of two basic tasks:
  - Access to information contained in various text documents required some sort of summarization processing to reduce the body of text to a tractable size.
  - Once the documents were summarized acceptably, the job of classifying the myriad of document summaries into some sort of logical arrangement was daunting.

History of text mining

- Approaches to access textual information developed in three venues:
  - Library Science for text summarization and classification
  - Information Science
  - Natural language processing

Library science for text summarization and classification



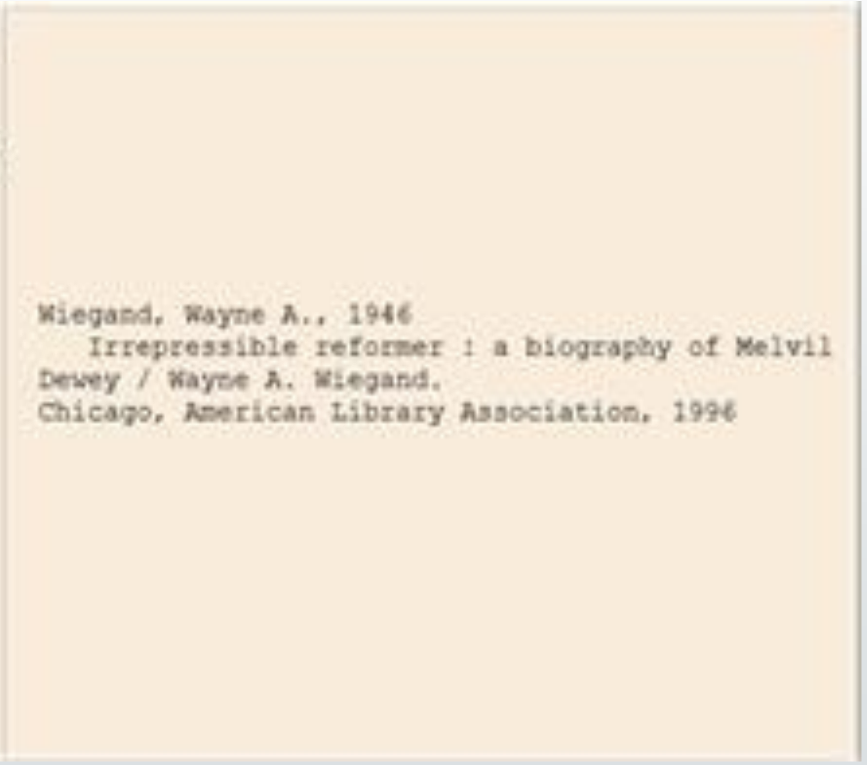
A Biography of Melvil Dewey

IRREPRESSIBLE REFORMER

Wayne A. Wiegand

American Library Association

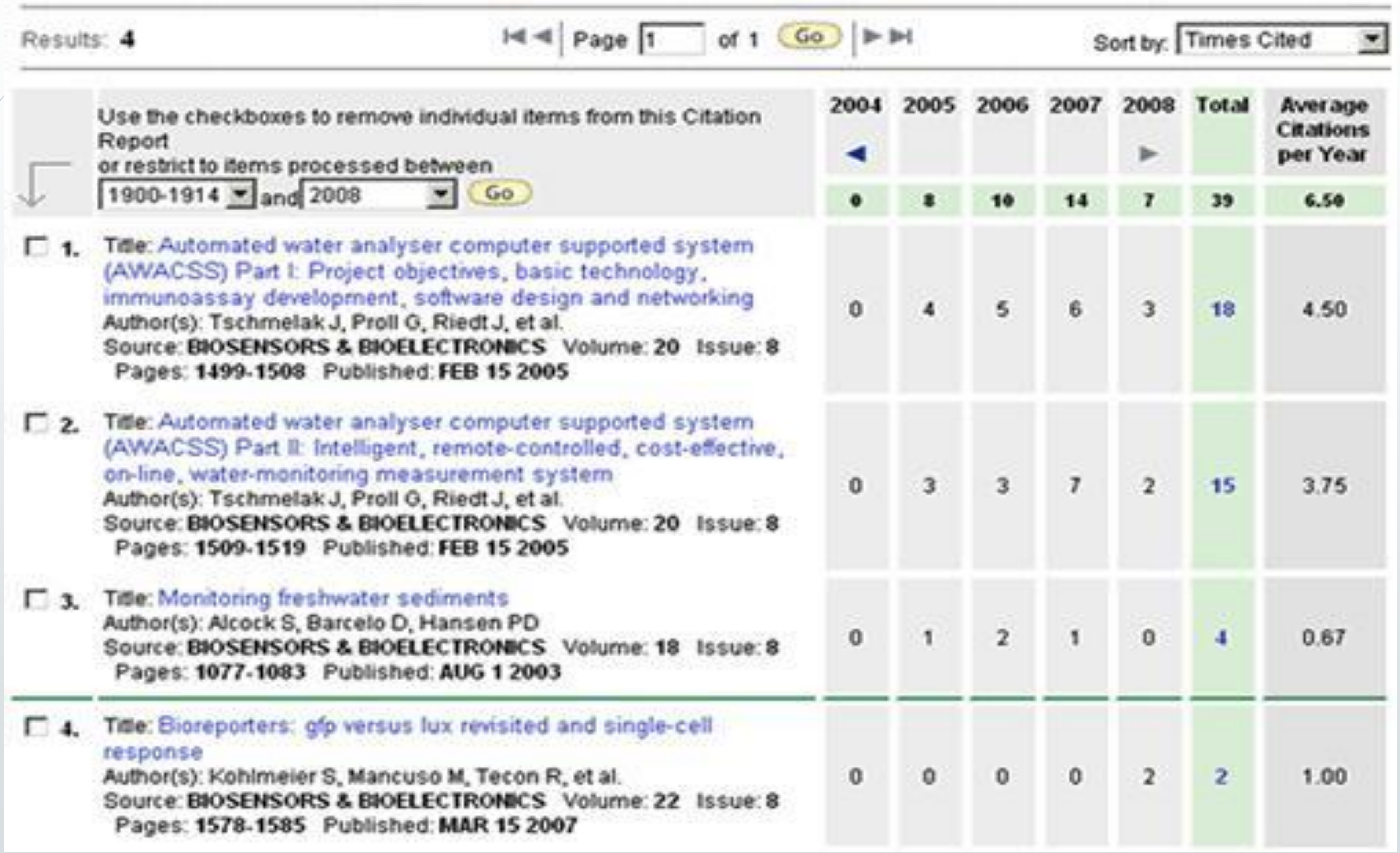
Chicago and London, 1996



```
Wiegand, Wayne A., 1946
   Irrepressible reformer : a biography of Melvil
Dewey / Wayne A. Wiegand.
Chicago, American Library Association, 1996
```

- The library card catalog at Yale University and an index card

Information Science



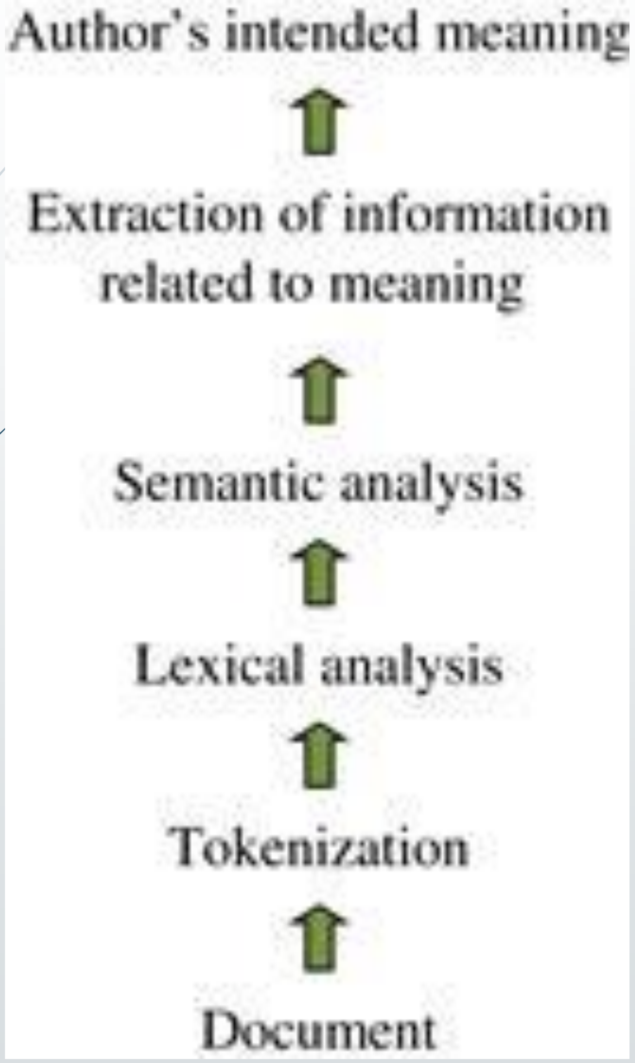
Results: 4 | Page 1 of 1 Go | Sort by: Times Cited

Use the checkboxes to remove individual items from this Citation Report or restrict to items processed between 1900-1914 and 2008 Go

	2004	2005	2006	2007	2008	Total	Average Citations per Year
	0	8	10	14	7	39	6.50
1. Title: Automated water analyser computer supported system (AWACSS) Part I: Project objectives, basic technology, immunoassay development, software design and networking Author(s): Tschmelak J, Proll G, Riedt J, et al. Source: BIOSENSORS & BIOELECTRONICS Volume: 20 Issue: 8 Pages: 1499-1508 Published: FEB 15 2005	0	4	5	6	3	18	4.50
2. Title: Automated water analyser computer supported system (AWACSS) Part II: Intelligent, remote-controlled, cost-effective, on-line, water-monitoring measurement system Author(s): Tschmelak J, Proll G, Riedt J, et al. Source: BIOSENSORS & BIOELECTRONICS Volume: 20 Issue: 8 Pages: 1509-1519 Published: FEB 15 2005	0	3	3	7	2	15	3.75
3. Title: Monitoring freshwater sediments Author(s): Alcock S, Barcelo D, Hansen PD Source: BIOSENSORS & BIOELECTRONICS Volume: 18 Issue: 8 Pages: 1077-1083 Published: AUG 1 2003	0	1	2	1	0	4	0.67
4. Title: Bioreporters: gfp versus lux revisited and single-cell response Author(s): Kohlmeier S, Mancuso M, Tecon R, et al. Source: BIOSENSORS & BIOELECTRONICS Volume: 22 Issue: 8 Pages: 1578-1585 Published: MAR 15 2007	0	0	0	0	2	2	1.00

- An example of Science Citation Index page

Natural Language Processing


Author's intended meaning
Extraction of information related to meaning
Semantic analysis
Lexical analysis
Tokenization
Document

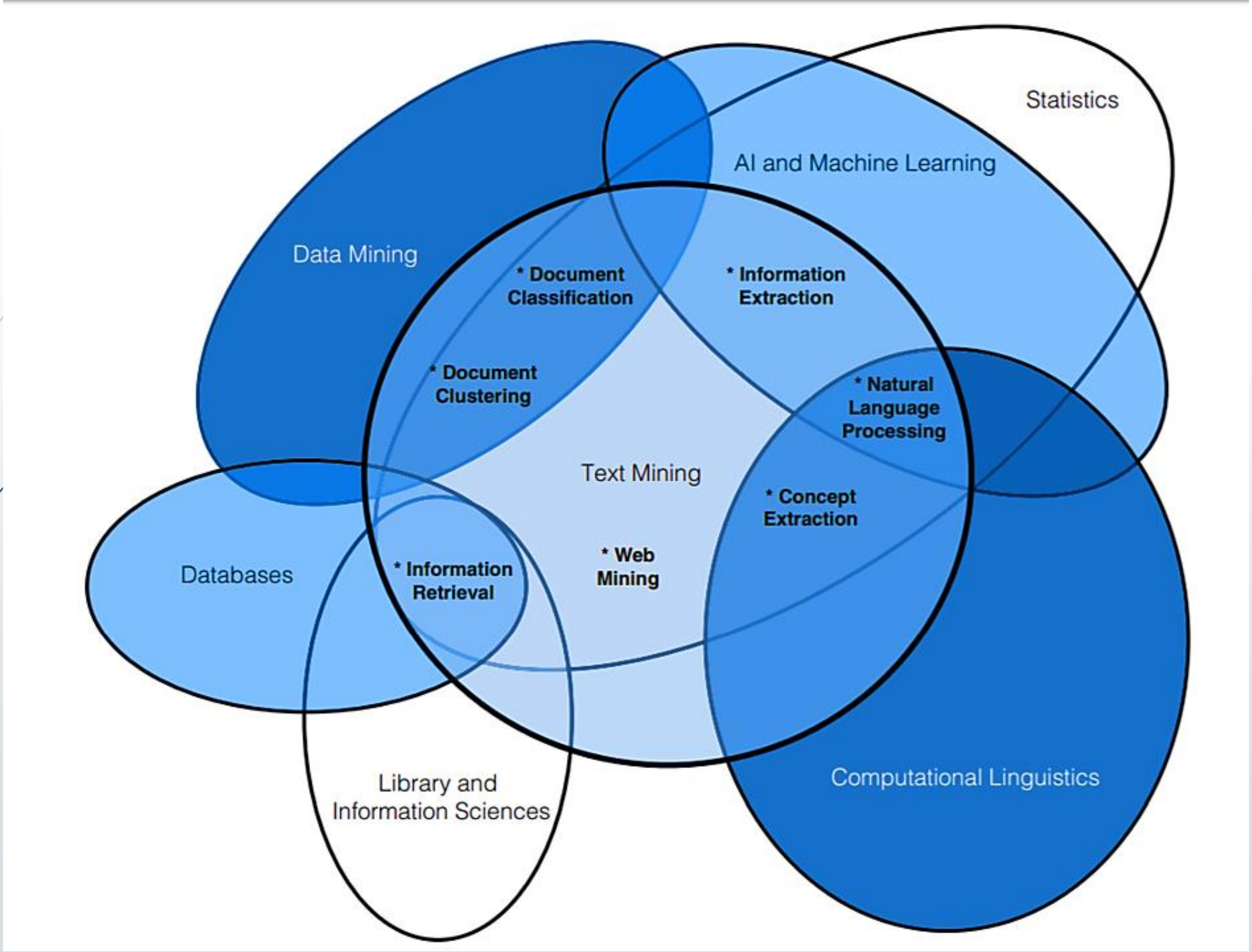
- Stages of Analysis in NLP.

WHAT IS TEXT MINING?

- Text mining and text analytics are broad umbrella terms describing a range of technologies for analyzing and processing semi structured and unstructured text data.
- The unifying theme behind each of these technologies is the need to “turn text into numbers” so powerful algorithms can be applied to large document databases.
- Converting text into a structured, numerical format and applying analytical algorithms require knowing how to, both use and combine techniques for handling text, ranging from individual words to documents to entire document databases.

THE SEVEN PRACTICE AREAS OF TEXT ANALYTICS

1. Search and information retrieval (IR):
2. Document clustering:
3. Document classification:
4. Web mining:
5. Information extraction (IE):
6. Natural language processing (NLP):
7. Concept extraction:


Statistics
AI and Machine Learning
Data Mining
* Document Classification
* Information Extraction
* Document Clustering
* Natural Language Processing
Text Mining
* Concept Extraction
Databases
* Information Retrieval
* Web Mining
Library and Information Sciences
Computational Linguistics

- Search and information retrieval (IR): Storage and retrieval of text documents, including search engines and keyword search.
- Document clustering: Grouping and categorizing terms, snippets, paragraphs, or documents, using data mining clustering methods.
- Document classification: Grouping and categorizing snippets, paragraphs, or documents, using data mining classification methods, based on models trained on labeled examples.
- Web mining: Data and text mining on the Internet, with a specific focus on the scale and interconnectedness of the web.

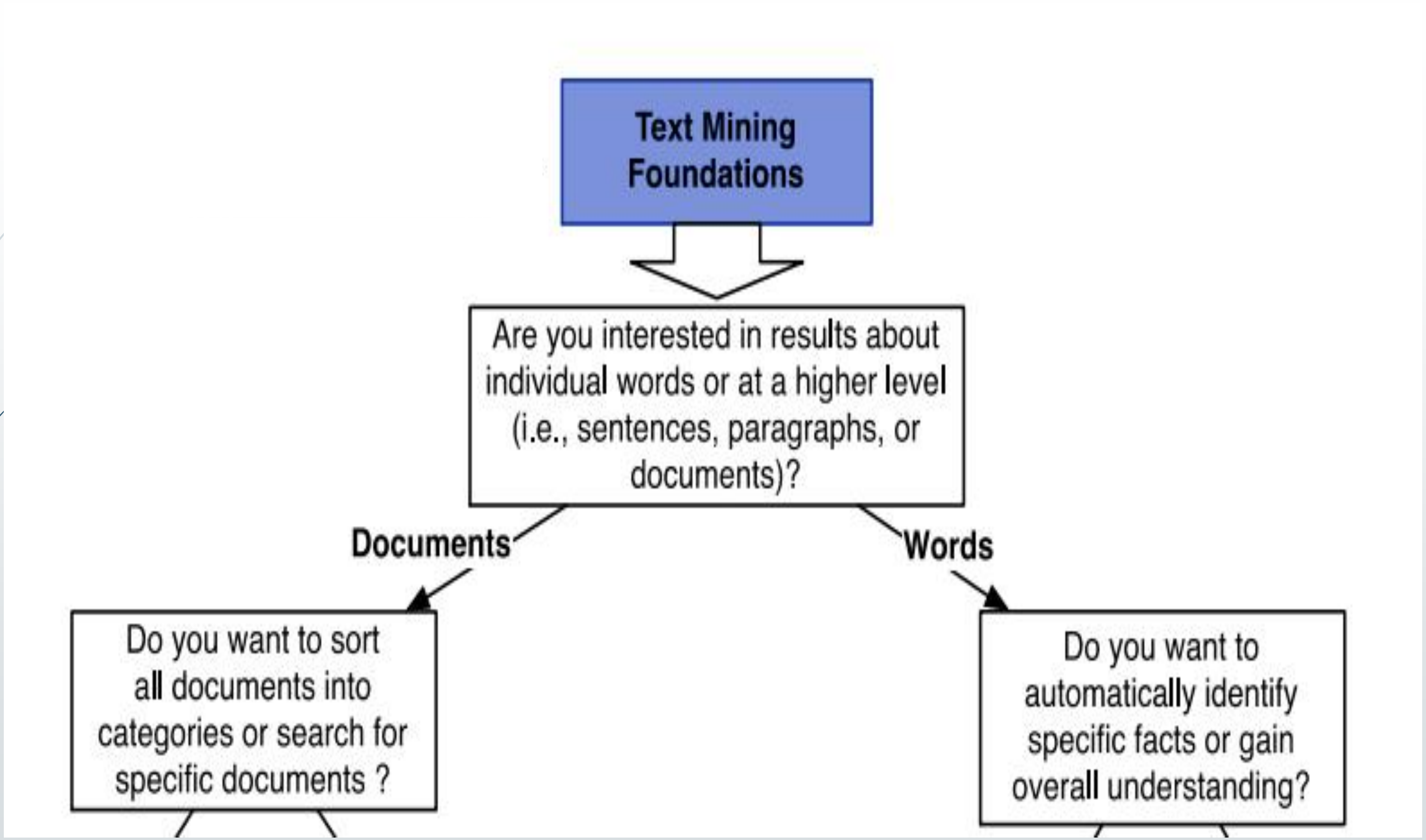
- Information extraction (IE): Identification and extraction of relevant facts and relationships from unstructured text; the process of making structured data from unstructured and semi structured text.
- Natural language processing (NLP): Low-level language processing and understanding tasks (e.g., tagging part of speech); often used synonymously with computational linguistics.
- Concept extraction: Grouping of words and phrases into semantically similar groups.

FIVE QUESTIONS FOR FINDING THE RIGHT PRACTICE AREA

- **Question 1: Granularity**
- This question finds the desired granularity (level of detail of focus) of the text mining task.
- While documents and words are both integral to successful text mining, an algorithm virtually always emphasizes one or the other.
- To determine the granularity of your text mining problem, ask yourself about the desired outcome: Is it about characterizing or grouping together words or documents?
- This is the biggest division between classes of text mining algorithms.

FIVE QUESTIONS FOR FINDING THE RIGHT PRACTICE AREA

- **Question 2: Focus**
- The focus of the algorithm: Are you interested in finding specific words and documents or characterizing the entire set?
- The two practice areas separated by this question - search and information extraction - both concentrate on identifying specific pieces of information within a document database, whereas the other solutions attempt to cluster or partition the space.


Text Mining Foundations
Are you interested in results about individual words or at a higher level (i.e., sentences, paragraphs, or documents)?
Documents
Words
Do you want to sort all documents into categories or search for specific documents ?
Do you want to automatically identify specific facts or gain overall understanding?

FIVE QUESTIONS FOR FINDING THE RIGHT PRACTICE AREA

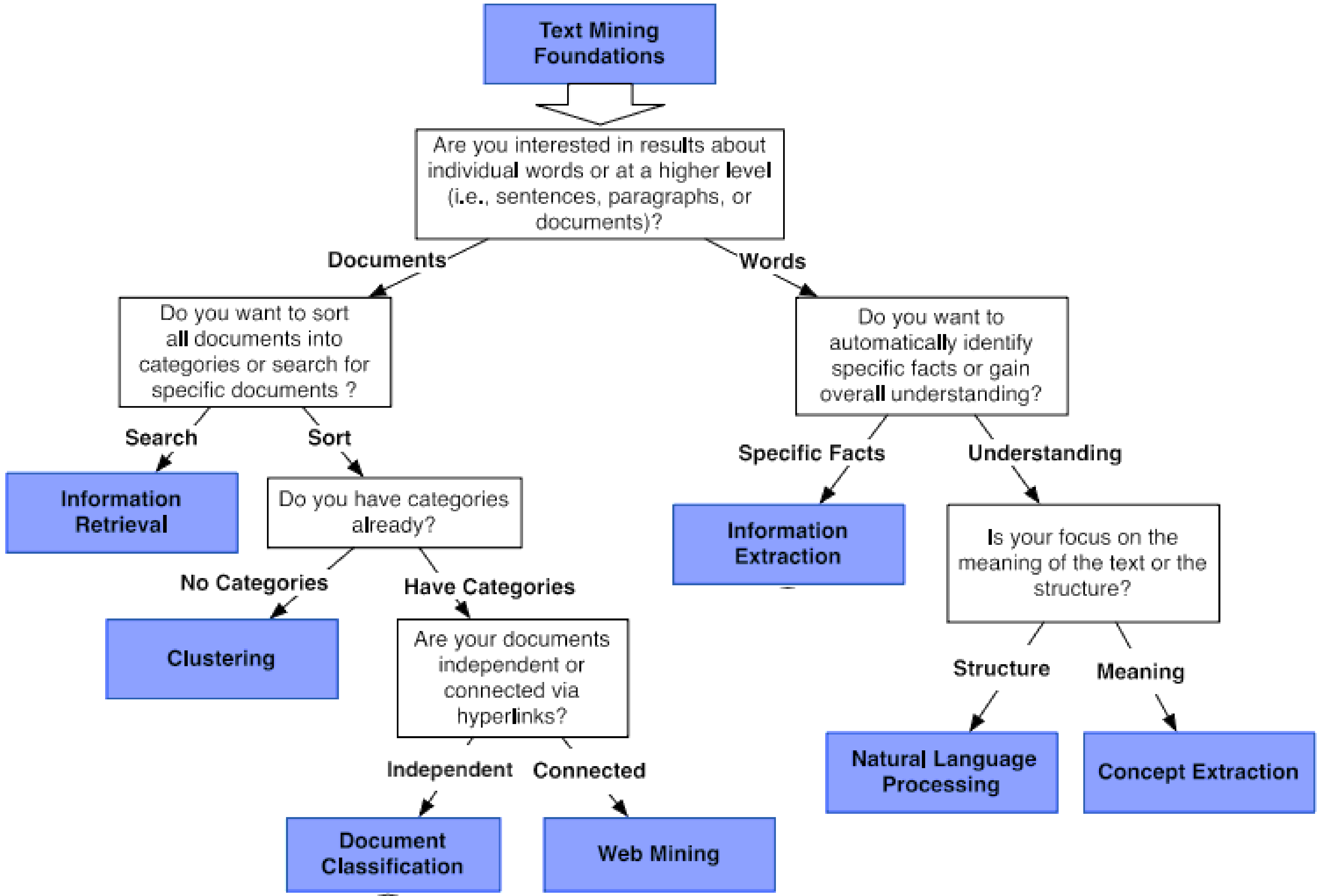
- **Question 3: Available Information**
- If you are interested in documents, the next question regards the available information at the time of analysis.
- This is equivalent to the supervised/unsupervised question from data mining.
- A supervised algorithm requires training data with an answer (outcome label) for positive and negative examples of the classes you're trying to model (such as distinguishing "interesting versus not interesting" articles for an analyst studying a specialized topic).
- An unsupervised algorithm does not require any labeled data, and it can be applied to any data set without any available information at analysis time. Supervised learning is much more powerful when possible to used that is, when enough example cases with target outcomes are known.

FIVE QUESTIONS FOR FINDING THE RIGHT PRACTICE AREA

- **Question 4: Syntax or Semantics**
- If you are interested in words, the major question is about syntax or semantics.
- Syntax is about what the words "say," while semantics is about what the words "mean."
- Because natural language is so fluid and complex, semantics is the harder problem. However, there are text mining algorithms to address both areas.

FIVE QUESTIONS FOR FINDING THE RIGHT PRACTICE AREA

- **Question 4: Syntax or Semantics**
- If you are interested in words, the major question is about syntax or semantics.
- Syntax is about what the words "say," while semantics is about what the words "mean."
- Because natural language is so fluid and complex, semantics is the harder problem. However, there are text mining algorithms to address both areas.


Text Mining Foundations
Are you interested in results about individual words or at a higher level (i.e., sentences, paragraphs, or documents)?
Documents
Words
Do you want to sort all documents into categories or search for specific documents ?
Search
Sort
Information Retrieval
Do you have categories already?
No Categories
Have Categories
Clustering
Are your documents independent or connected via hyperlinks?
Independent
Connected
Document Classification
Web Mining
Do you want to automatically identify specific facts or gain overall understanding?
Specific Facts
Understanding
Information Extraction
Is your focus on the meaning of the text or the structure?
Structure
Meaning
Natural Language Processing
Concept Extraction

FIVE QUESTIONS FOR FINDING THE RIGHT PRACTICE AREA

- **Question 5: Web or Traditional Text**
- The rise of the Internet (including blogs, Twitter, and Facebook) is largely responsible for the prominence that text mining holds today by making available a vast number of previously unreachable text documents. The structure and style of web documents provide both unique opportunities and challenges when compared to non web documents. Though many of the algorithms are theoretically the same for web and traditional text, the scale of the web and its unique structural characteristics justify defining two different categories.

Search and Information Retrieval

- Search and information retrieval covers indexing, searching, and retrieving documents from large text databases with keyword queries.
- With the rise of powerful Internet search engines, including Google, Yahoo!, and Bing, search and information retrieval has become familiar to most people.
- Nearly every computer application from email to word processing includes a search function.

Document Clustering

- Document clustering uses algorithms from data mining to group similar documents into clusters.
- Data mining has been a very active field for nearly two decades, and clustering algorithms preceded that, so clustering algorithms are widely available in many commercial data and text mining software packages.

Document Classification

- Document classification assigns a known set of labels to untagged documents, using a model of text learned from documents with known labels.
- Like document clustering, document classification draws from an enormous field of work in data mining, statistics, and machine learning.
- It is one of the most prominent techniques used in text mining.

Web Mining

- Web mining is its own practice area due to the unique structure and enormous volume of data appearing on the web.
- Web documents are typically presented in a structured text format with hyperlinks between pages.
- These differences from standard text present a few challenges and many opportunities.
- As the Internet becomes even more ingrained in our popular culture with the rise of Facebook, Twitter, and other social media channels, web mining will continue to increase in value.
- Though it is still an emerging area, web mining draws on mature technology in document classification and natural language understanding.

Information Extraction

- The goal of information extraction is to construct (or extract) structured data from unstructured text.
- Information extraction is one of the more mature fields within text mining, but it is difficult for beginners to work in without considerable effort, since it requires specialized algorithms and software.
- Furthermore, the training and tuning of an information extraction system require a large amount of effort.
- There are a number of commercial products available for information extraction, but all of them require some customization to achieve high performance for a given document database.

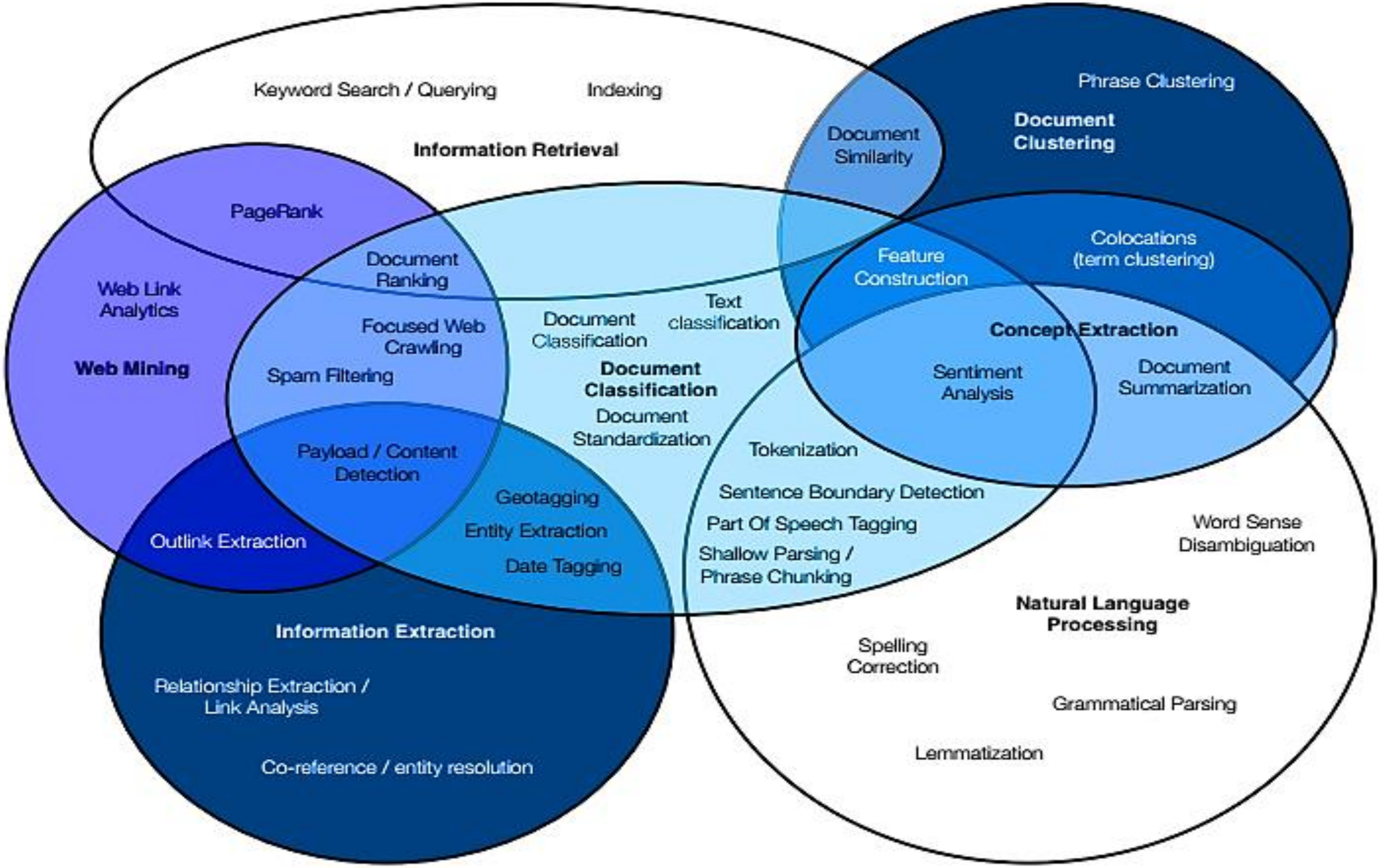
Natural Language Processing

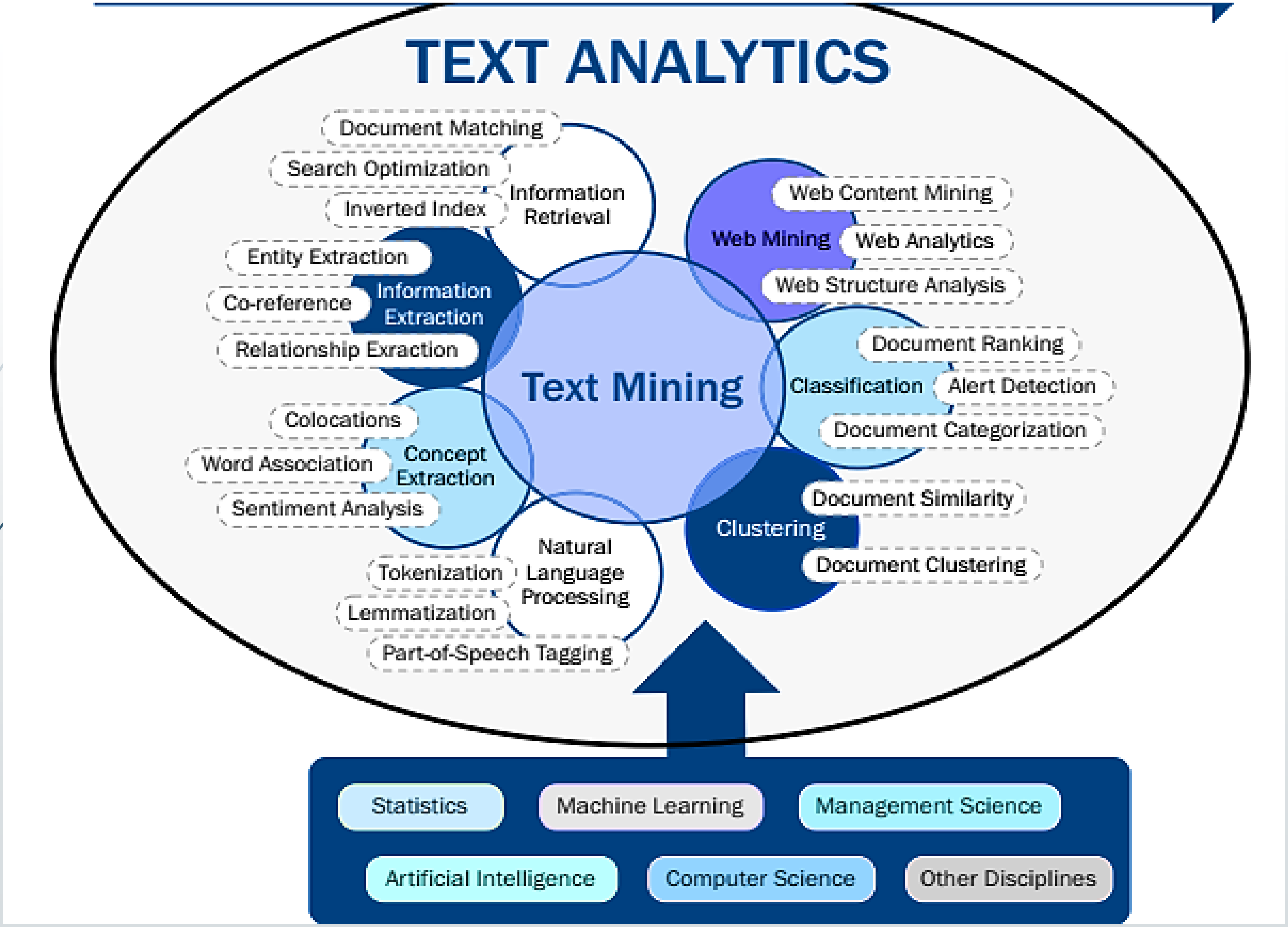
- Natural language processing (NLP) has a relatively long history in both linguistics and computer science.
- Recently, the focus of NLP has moved further into the text mining realm by considering statistical approaches.
- NLP is a powerful tool for providing useful input variables for text mining such as part of speech tags and phrase boundaries.

Concept Extraction

- Extracting concepts is, in some ways, both the easiest and the hardest of the practice areas to do. The meaning of text is notoriously hard for automated systems to “understand.”
- However, some initial automated work combined with human understanding can lead to significant improvements over the performance of either a machine or a human alone.

Lecture 2


Keyword Search / Querying
Indexing
Information Retrieval
Phrase Clustering
Document Clustering
Document Similarity
PageRank
Document Ranking
Web Link Analytics
Web Mining
Focused Web Crawling
Spam Filtering
Document Classification
Text classification
Document Classification
Document Standardization
Feature Construction
Colocations (term clustering)
Concept Extraction
Sentiment Analysis
Document Summarization
Payload / Content Detection
Outlink Extraction
Geotagging
Entity Extraction
Date Tagging
Tokenization
Sentence Boundary Detection
Part Of Speech Tagging
Shallow Parsing / Phrase Chunking
Word Sense Disambiguation
Natural Language Processing
Spelling Correction
Grammatical Parsing
Lemmatization
Information Extraction
Relationship Extraction / Link Analysis
Co-reference / entity resolution


TEXT ANALYTICS
Document Matching
Search Optimization
Inverted Index
Information Retrieval
Web Mining
Web Content Mining
Web Analytics
Web Structure Analysis
Entity Extraction
Co-reference
Information Extraction
Relationship Exraction
Text Mining
Classification
Document Ranking
Alert Detection
Document Categorization
Colocations
Word Association
Concept Extraction
Sentiment Analysis
Clustering
Document Similarity
Document Clustering
Tokenization
Lemmatization
Natural Language Processing
Part-of-Speech Tagging
Statistics
Machine Learning
Management Science
Artificial Intelligence
Computer Science
Other Disciplines

APPLICATIONS AND USE CASES

- Five basic types of analytical text mining applications and use cases can be identified that address most important text processing issues:
- 1. **Extracting "meaning" from unstructured text.** This application involves the understanding of core themes and relevant messages in a corpus of text, without actually reading the documents.
- 2. **Automatic text categorization**. Automatically classifying text is an efficient way to organize text for downstream processing.
- 3. **Improving predictive accuracy in predictive modeling or unsupervised learning**. Combining unstructured text with structured numeric information in predictive modeling or unsupervised learning (clustering) is a powerful method to achieve better accuracy.

APPLICATIONS AND USE CASES

- 4. **Identifying specific or similar/relevant documents**. Efficiently extracting from a large corpus of text those documents that are relevant to a particular topic of interest or are similar to a target document (or documents) is a vitally necessary operation in information retrieval.
- 5. **Extracting specific information from the text** ("entity extraction"). Automatically extracting specific information from the text (such as names, geographical locations, and dates) is an efficient method for presenting highly focused information for downstream analytical processing or for direct use by decision makers.

Extracting “Meaning” from Unstructured Text

- Extract specific contents or contextual meaning from a large corpus of small text documents
- or from a small corpus of large text documents
- that cannot be read and summarized in a practical manner.

Extracting "Meaning" from Unstructured Text

- Sentiment Analysis seeks to determine the general sentiments, opinions, and affective states of people reflected in a corpus of text.
  - What are my customers saying about me?
  - What are the emerging areas of concern or interest in a specific target group?
  - Analyzing open-ended responses to survey questions.
- Trending Themes In A Stream Of Text
- Warranty Claim Trends
- Insurance Claims,
- Fraud Detection, and So Forth

Summarizing Text

- Another use case of text mining is the extraction of meaning when the goal is to quickly summarize one or a few very large documents.
- There are two types of text summarizations.
- One type summarizes themes across the chapters or paragraphs of the text, in which case the individual paragraphs or chapters can be considered different documents of a larger corpus (the entire text).
- The goal of this type is to identify the different themes across the various documents (e.g., as just described) or to identify common dimensions or relationships among individuals, events, and so on.

Summarizing Text

- The second type summarizes the contents of a large text document into a meaningful narrative which cannot be accomplished effectively (yet) using automatic text mining methods and algorithms.
- In short, it is not realistic to expect that present computer algorithms are capable of summarizing the "essence" of a very large book into a single paragraph.
- At present, this can be done only in other highly subjective ways.

Text Analysis and Text Mining

- Text analysis to the representation, processing, and modeling of textual data to derive useful insights.
- An important component of text analysis is text mining, the process of discovering relationships and interesting patterns in large text collections.

Example

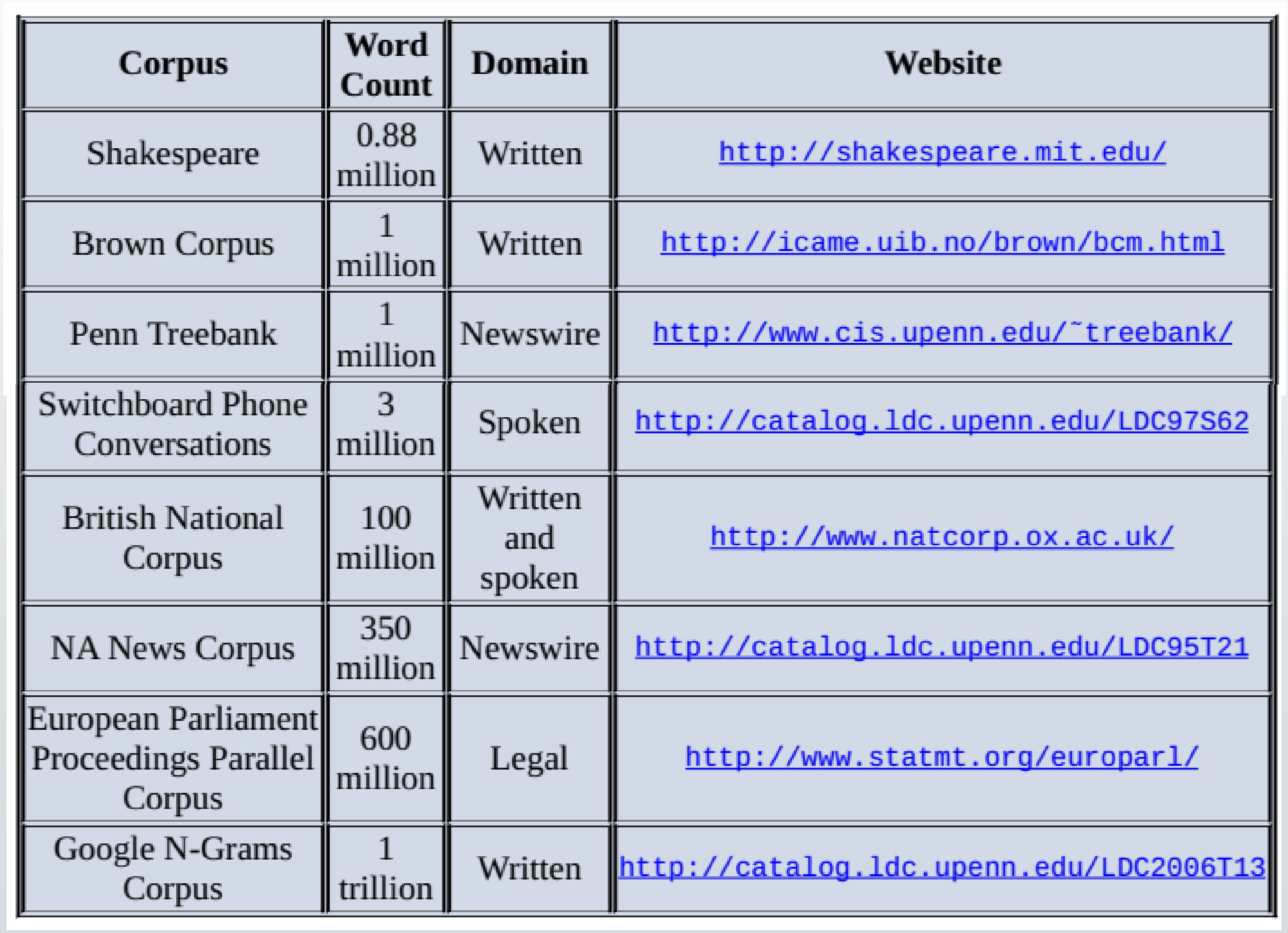
- Text analysis suffers from the curse of high dimensionality.
- Take the popular children’s book Green Eggs and Ham [1] as an example. Author Theodor Geisel (Dr. Seuss) was challenged to write an entire book with just 50 distinct words.
- He responded with the book Green Eggs and Ham, which contains 804 total words, only 50 of them distinct. These 50 words are:

a, am, and, anywhere, are, be, boat, box, car, could, dark, do, eat, eggs, fox, goat, good, green, ham, here, house, I, if, in, let, like, may, me, mouse, not, on, or, rain, Sam, say, see, so, thank, that, the, them, there, they, train, tree, try, will, with, would, you

- There’s a substantial amount of repetition in the book. Yet, as repetitive as the book is, modeling it as a vector of counts, or features, for each distinct word still results in a 50 - dimension problem.

CORPUS

- Text analysis often deals with textual data that is far more complex.
- A **CORPUS** (plural: corpora) is a large collection of texts used for various purposes in NLP.

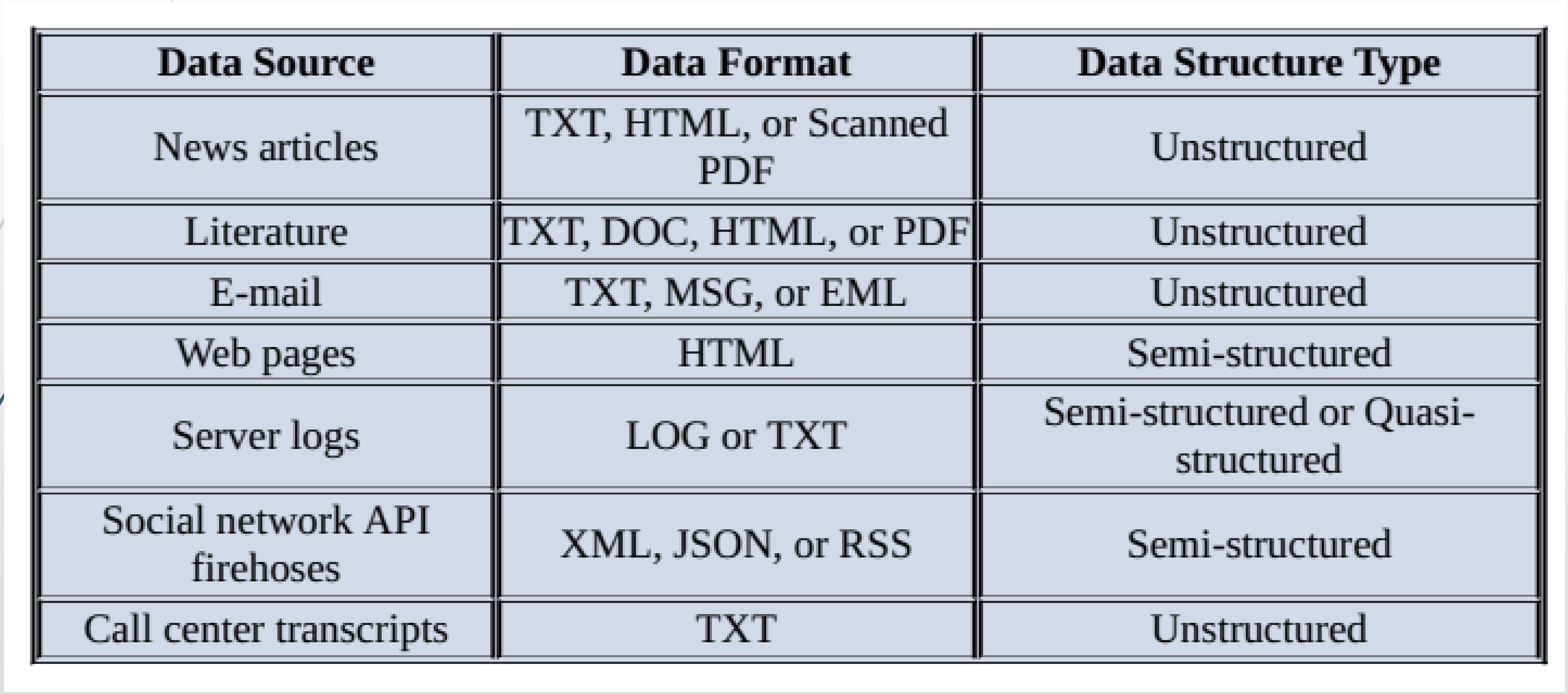


Corpus	Word Count	Domain	Website
Shakespeare	0.88 million	Written	http://shakespeare.mit.edu/
Brown Corpus	1 million	Written	http://icame.uib.no/brown/bcm.html
Penn Treebank	1 million	Newswire	http://www.cis.upenn.edu/˜treebank/
Switchboard Phone Conversations	3 million	Spoken	http://catalog.ldc.upenn.edu/LDC97S62
British National Corpus	100 million	Written and spoken	http://www.natcorp.ox.ac.uk/
NA News Corpus	350 million	Newswire	http://catalog.ldc.upenn.edu/LDC95T21
European Parliament Proceedings Parallel Corpus	600 million	Legal	http://www.statmt.org/europarl/
Google N-Grams Corpus	1 trillion	Written	http://catalog.ldc.upenn.edu/LDC2006T13

Challenges of corpus

- The smallest corpus in the list, the complete works of Shakespeare, contains about 0.88 million words.
- In contrast, the Google n-gram corpus contains one trillion words from publicly accessible web pages.
- Out of the one trillion words in the Google n-gram corpus, there might be one million distinct words, which would correspond to one million dimensions.
- The **high dimensionality** of text is an important issue, and it has a direct impact on the complexities of many text analysis tasks.
- Another major challenge with text analysis is that most of the time the text is **not structured**. It will be quasi-structured, semi-structured, or unstructured data.

Example data structure



Data Source	Data Format	Data Structure Type
News articles	TXT, HTML, or Scanned PDF	Unstructured
Literature	TXT, DOC, HTML, or PDF	Unstructured
E-mail	TXT, MSG, or EML	Unstructured
Web pages	HTML	Semi-structured
Server logs	LOG or TXT	Semi-structured or Quasi-structured
Social network API firehoses	XML, JSON, or RSS	Semi-structured
Call center transcripts	TXT	Unstructured

Text Analysis Steps

- A text analysis problem usually consists of three important steps: parsing, search and retrieval, and text mining.
- **PARSING** is the process that takes unstructured text and imposes a structure for further analysis.
  - The unstructured text could be a plain text file, a weblog, an Extensible Markup Language (XML) file, a Hyper Text Markup Language (HTML) file, or a Word document.
  - Parsing deconstructs the provided text and renders it in a more structured way for the subsequent steps.
- **SEARCH AND RETRIEVAL** is the identification of the documents in a corpus that contain search items such as specific words, phrases, topics, or entities like people or organizations.
  - These search items are generally called key terms.
  - Search and retrieval originated from the field of library science and is now used extensively by web search engines.

Text Analysis Steps

- **TEXT MINING** uses the terms and indexes produced by the prior two steps to discover meaningful insights pertaining to domains or problems of interest.
  - With the proper representation of the text, many of the techniques, such as clustering and classification, can be adapted to text mining.
  - K-means can be modified to cluster text documents into groups, where each group represents a collection of documents with a similar topic. The distance of a document to a centroid represents how closely the document talks about that topic.
  - Classification tasks such as sentiment analysis and spam filtering are prominent use cases for the naïve Bayes classifier.
- All three steps may not be included in all projects. Also they may have any sequence.

Part-of-Speech (POS) Tagging

- The goal of POS tagging is to build a model whose input is a sentence, such as:

1. he saw a fox

- and whose output is a tag sequence.
- Each tag marks the POS for the corresponding word, such as:

1. PRP VBD DT NN

- according to the Penn Treebank POS tags [3]. Therefore, the four words are mapped to pronoun (personal), verb (past tense), determiner, and noun (singular), respectively.

Lemmatization and Stemming

- These are techniques to reduce the number of dimensions and reduce inflections or variant forms to the base form to more accurately measure the number of times each word appears

Lemmatization

- With the use of a given dictionary, lemmatization finds the correct dictionary base form of a word.

For example, given the sentence:

1. obesity causes many problems

the output of lemmatization would be:

1. obesity cause many problem

Stemming

- Stemming refers to a crude process of stripping affixes based on a set of heuristics with the hope of correctly achieving the goal to reduce inflections or variant forms.
- After the process, words are stripped to become stems.
- A stem is not necessarily an actual word defined in the natural language, but it is sufficient to differentiate itself from the stems of other words.
- A well-known rule-based stemming algorithm is Porter’s stemming algorithm.
- It defines a set of production rules to iteratively transform words into their stems.

For the sentence shown previously:

1. obesity causes many problems

the output of Porter’s stemming algorithm is:

1. obes caus mani problem

A Text Analysis Example

- Consider the fictitious company ACME, maker of two products: bPhone and bEbook.
- ACME is in strong competition with other companies that manufacture and sell similar products.
- To succeed, ACME needs to produce excellent phones and eBook readers and increase sales.
- One of the ways the company does this is to monitor what is being said about ACME products in social media.
- In other words, what is the buzz on its products?

A Text Analysis Example

- ACME wants to search all that is said about ACME products in social media sites, such as Twitter and Facebook, and popular review sites, such as Amazon and ConsumerReports.
- It wants to answer questions such as these.
- Are people mentioning its products?
- What is being said? Are the products seen as good or bad?
- If people think an ACME product is bad, why? For example, are they complaining about the battery life of the bPhone, or the response time in their bEbook?

A Text Analysis Example

- ACME can monitor all these using three steps seen before: Parsing, search and retrieval and text mining.
- These steps will include following modules:
- Collect raw text (Phase 1 and Phase 2)
- Represent text (Phase 2 and Phase 3)
- TFIDF (Phase 3 to 5)
- Topic Modelling (Phase 3 to 5)
- Sentiment Analysis (Phase 3 to 5)
- Gain Insights (Phase 5 and 6)

A Text Analysis Example

- **Collect raw text** (Phase 1 and Phase 2)
  - The Data Science team monitors websites for references to specific products.
  - The websites may include social media and review sites.
  - Interact with social network APIs process data feeds, or scrape pages and use product names as keywords to get the raw data.
  - Regular expressions can be used to identify text that matches certain patterns.
  - Additional filters : regional studies.
  - Filter in data collection phase can reduce I/O workloads and minimize the storage requirements.

A Text Analysis Example

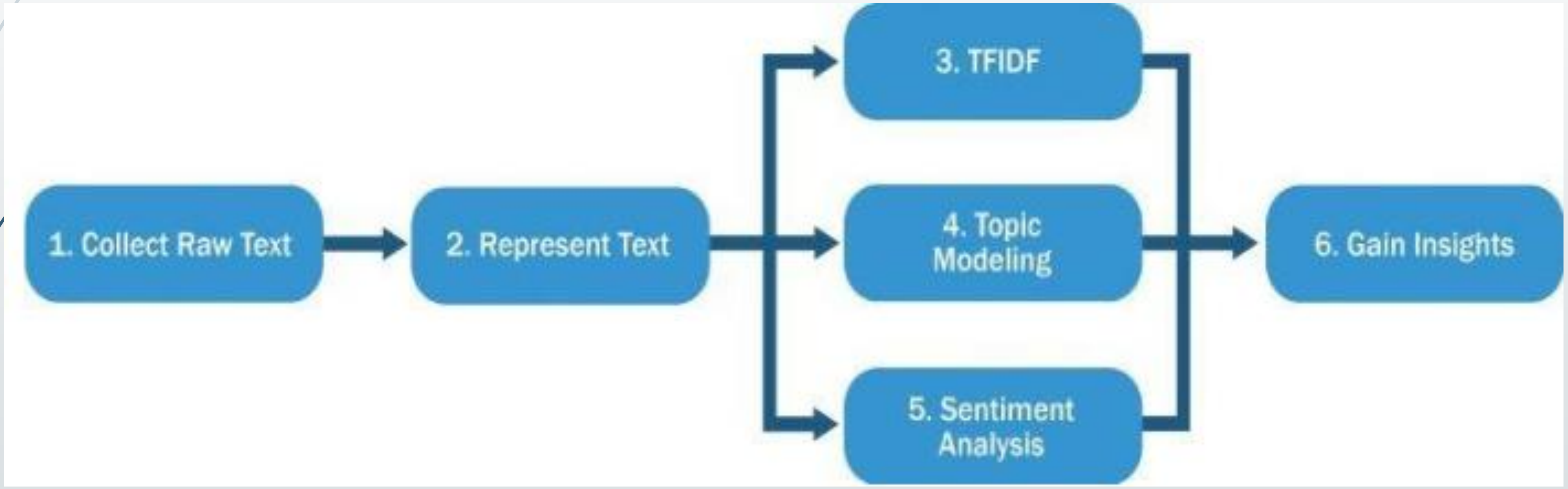
- **Represent text** (Phase 2 and Phase 3)
  - Convert each review into a suitable document representation with proper indices, and build a corpus based on these indexed reviews.
- Compute the usefulness of each word in the reviews using methods such as **TFIDF** (Phase 3 to 5).
- **Topic Modelling** (Phase 3 to 5): Categorize documents by topics.
- **Sentiment Analysis** (Phase 3 to 5): Determine sentiments of the reviews.
  - Identify whether the reviews are positive or negative.
  - Rating of a product.
  - Or sentiment analysis can be used on the textual data to infer the underlying sentiments.
  - Sentiments can be considered as positive, neutral, or negative.

A Text Analysis Example

- **Gain Insights** (Phase 5 and 6):
- Marketing gathers the results from the previous steps.
- Find out what exactly makes people love or hate a product.
- Use one or more visualization techniques to report the findings.
- Test the soundness of the conclusions and operationalize the findings if applicable.

A Text Analysis Example

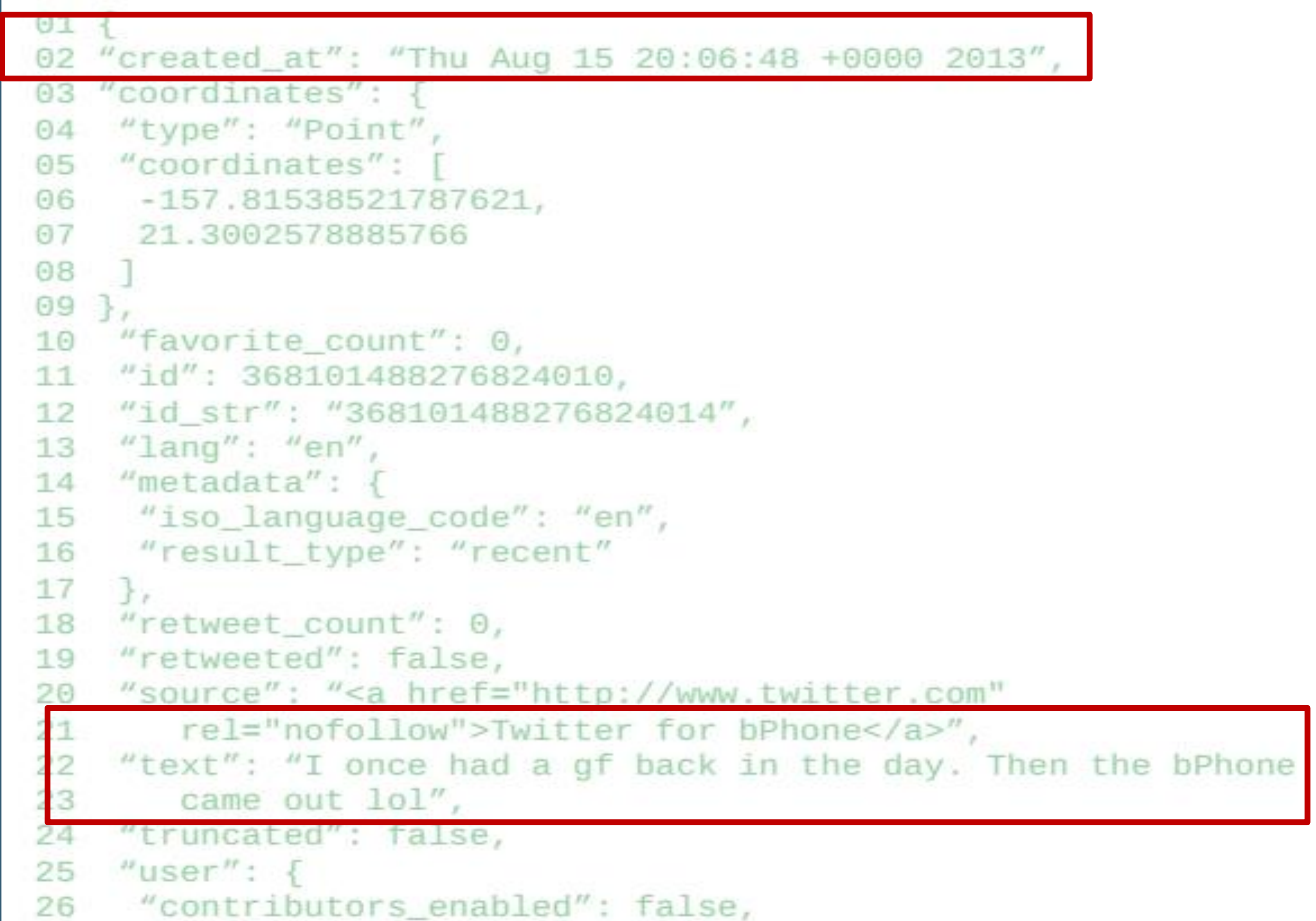
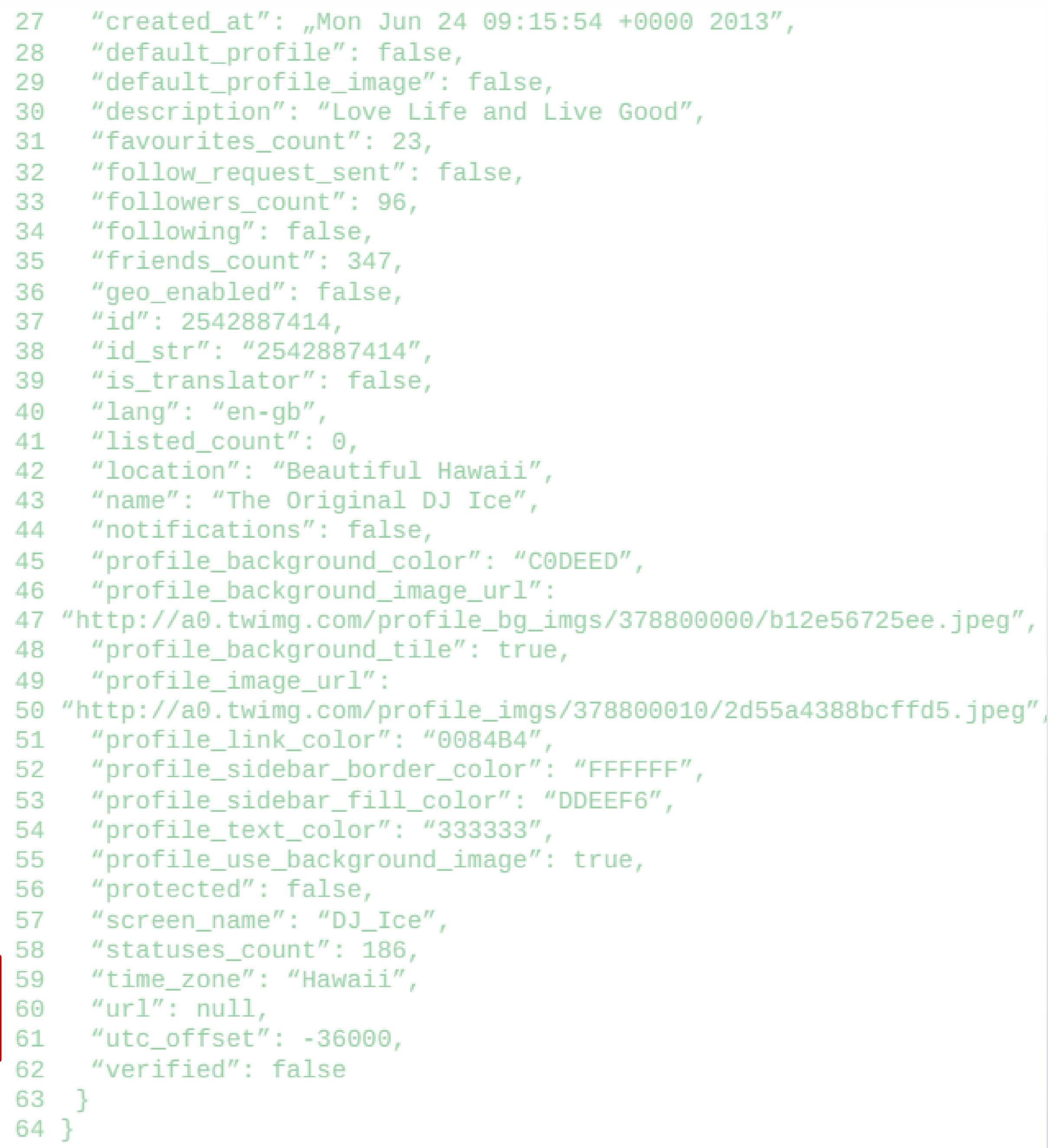
ACME's Text Analysis Process



Collecting Raw Data

- Monitoring various websites for user-generated contents.
- Data will be semi-structured data such as HTML web pages, Really Simple Syndication (RSS) feeds, XML, or JavaScript Object Notation (JSON) files.
- *bPhone* or *bEbook*
- *Twitter API* allows developers to choose from the Streaming API or the REST API to retrieve public Twitter posts.
- The fetched tweets are in the JSON format.

- A sample tweet that contains the keyword bPhone fetched using the Twitter Streaming API version 1.1

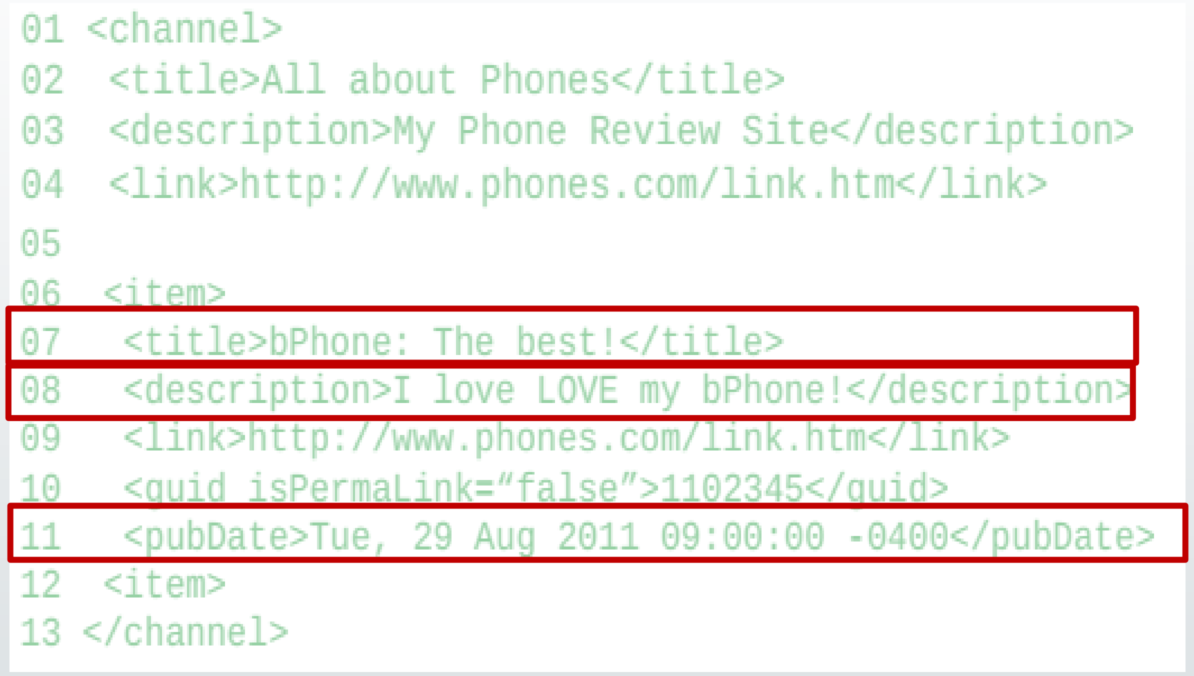
```
{
"created_at": "Thu Aug 15 20:06:48 +0000 2013",
"coordinates": {
 "type": "Point",
 "coordinates": [
  -157.81538521787621,
  21.3002578885766
 ]
},
 "favorite_count": 0,
 "id": 368101488276824010,
 "id_str": "368101488276824014",
 "lang": "en",
 "metadata": {
  "iso_language_code": "en",
  "result_type": "recent"
 },
 "retweet_count": 0,
 "retweeted": false,
 "source": "<a href="http://www.twitter.com"
    rel="nofollow">Twitter for bPhone</a>",
 "text": "I once had a gf back in the day. Then the bPhone
    came out lol",
 "truncated": false,
 "user": {
  "contributors_enabled": false,
  "created_at": „Mon Jun 24 09:15:54 +0000 2013",
  "default_profile": false,
  "default_profile_image": false,
  "description": "Love Life and Live Good",
  "favourites_count": 23,
  "follow_request_sent": false,
  "followers_count": 96,
  "following": false,
  "friends_count": 347,
  "geo_enabled": false,
  "id": 2542887414,
  "id_str": "2542887414",
  "is_translator": false,
  "lang": "en-gb",
  "listed_count": 0,
  "location": "Beautiful Hawaii",
  "name": "The Original DJ Ice",
  "notifications": false,
  "profile_background_color": "C0DEED",
  "profile_background_image_url":
 "http://a0.twimg.com/profile_bg_imgs/378800000/b12e56725ee.jpeg",
  "profile_background_tile": true,
  "profile_image_url":
 "http://a0.twimg.com/profile_imgs/378800010/2d55a4388bcffd5.jpeg",
  "profile_link_color": "0084B4",
  "profile_sidebar_border_color": "FFFFFF",
  "profile_sidebar_fill_color": "DDEEF6",
  "profile_text_color": "333333",
  "profile_use_background_image": true,
  "protected": false,
  "screen_name": "DJ_Ice",
  "statuses_count": 186,
  "time_zone": "Hawaii",
  "url": null,
  "utc_offset": -36000,
  "verified": false
 }
}
```

Collecting Raw Data – twitter sample

- The *created at* entry stores the timestamp that the tweet was published, and the *text field* stores the main content of the Twitter post.
- *Utilizing fields* such as *coordinates* (line 3 to 9), *user's local language* (lang, line 40), *user's location* (line 42), *time_zone* (line 59), and *utc_offset* (line 61) allows the analysis to focus on tweets from a specific region.

Collecting Raw Data – RSS file sample

- An RSS feed for a phone review blog



```
01 <channel>
02  <title>All about Phones</title>
03  <description>My Phone Review Site</description>
04  <link>http://www.phones.com/link.htm</link>

05
06  <item>
07   <title>bPhone: The best!</title>
08   <description>I love LOVE my bPhone!</description>
09   <link>http://www.phones.com/link.htm</link>
10   <guid isPermaLink="false">1102345</guid>
11   <pubDate>Tue, 29 Aug 2011 09:00:00 -0400</pubDate>
12  <item>
13 </channel>
```

- The content from the title (line 7), the description (line 8), and the published date (pubDate, line 11).

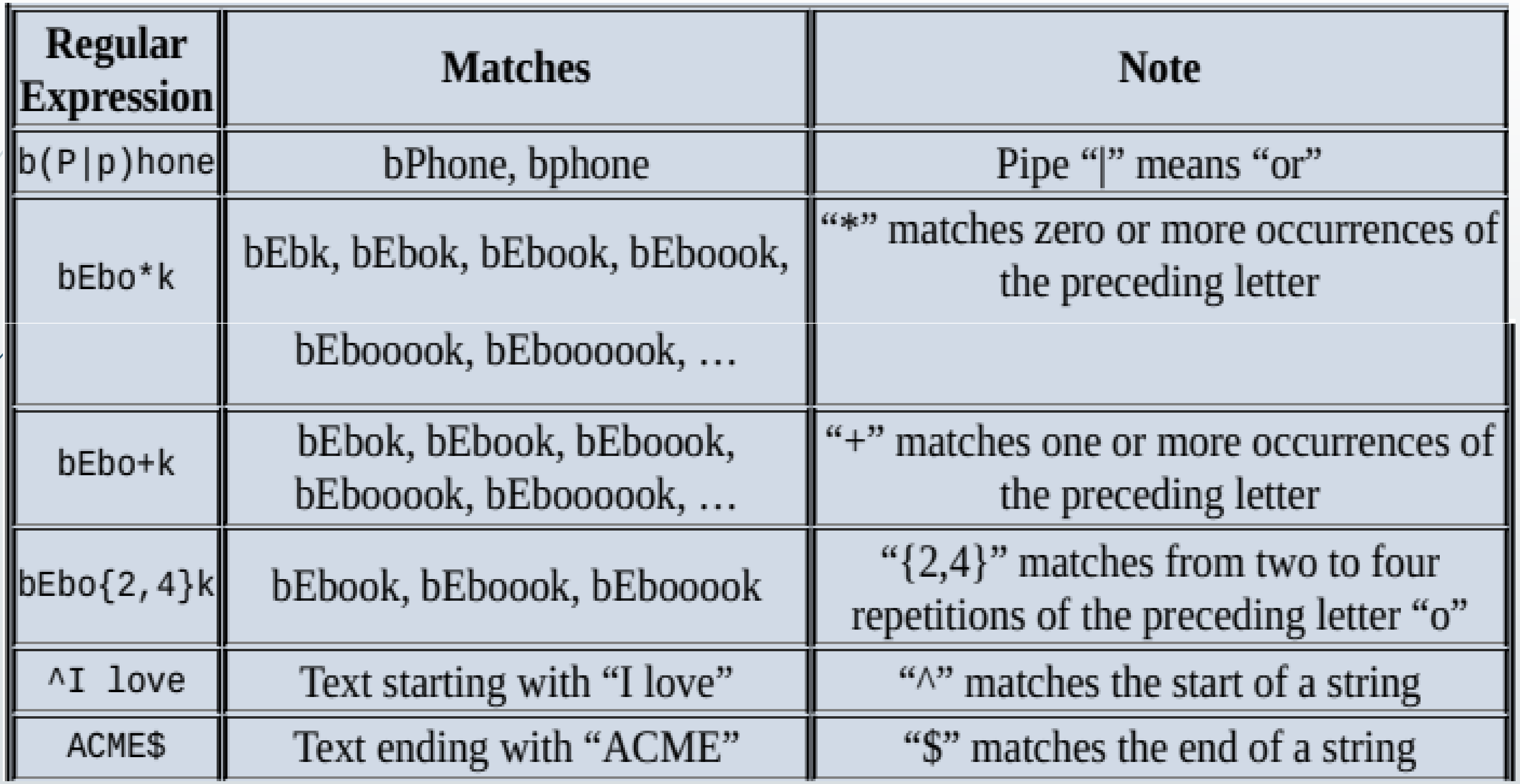
Collecting Raw Data – Web Scrapper

- If the plan is to collect user comments on ACME's products from online shops and review sites **where APIs or data feeds are not provided**, the team may have to **write web scrapers** to parse web pages and automatically extract the interesting data from those HTML files.
- A **WEB SCRAPER** is a software program (bot) that systematically browses the World Wide Web, downloads web pages, extracts useful information, and stores it somewhere for further study.
- Not possible to write a one-size-fits-all web scraper, since each website has its own pattern and can change from time to time.

Collecting Raw Data – Web Scrapper

- The team can then construct the web scraper based on the identified patterns.
- The scraper can use the curl tool to fetch HTML source code given specific URLs, use XPath and regular expressions to select and extract the data that match the patterns, and write them into a data store.
- Regular expressions can find words and strings that match particular patterns in the text effectively and efficiently.
- When matching the text, regular expressions can also take into account capitalizations, common misspellings, common abbreviations, and special formats for email addresses, dates, and telephone numbers.

Collecting Raw Data – Web Scrapper



Regular Expression	Matches	Note
`b(P|p)hone`	bPhone, bphone	Pipe “|” means “or”
`bEbo*k`	bEbk, bEbok, bEbook, bEboook, bEbooook, bEboooook, …	“*” matches zero or more occurrences of the preceding letter
`bEbo+k`	bEbok, bEbook, bEboook, bEbooook, bEboooook, …	“+” matches one or more occurrences of the preceding letter
`bEbo{2,4}k`	bEbook, bEboook, bEbooook	“{2,4}” matches from two to four repetitions of the preceding letter “o”
`^I love`	Text starting with “I love”	“^” matches the start of a string
`ACME$`	Text ending with “ACME”	“$” matches the end of a string

Collecting Raw Data – Web Scrapper

- If one chooses not to build a data collector from scratch, many companies such as GNIP and DataSift can provide data collection or data reselling services.
- Team should not violate the rights of the owner of the information and user agreements about use of websites during the data collection.
- Many websites place a file called robots.txt in the root directory which tells which pages are allowed and how to use website content correctly.

Lecture 3

Representing Text

- **Tokenization** is the task of separating (also called tokenizing) words from the body of text.
- Raw text is converted into collections of tokens after the tokenization, where each token is generally a word.
- A common approach is tokenizing on spaces. For example, with the tweet shown previously:

I once had a gf back in the day. Then the bPhone came out lol

- Tokenization based on spaces would output a list of tokens.

{I, once, had, a, gf, back, in, the, **day.**, Then, the, bPhone, came, out, lol}

Representing Text

- Use **lookup table to remove the period** if it appears at the end of a sentence.
- Or to tokenize the text based on **punctuation marks and spaces**.
- To tokenize words like we'll, can't based on punctuation will give we and 'll, can and 't. Here it makes **obscure words and distorts** the meaning. Such cases can affect sentiment analysis.
- Words like **state-of-the-art, Wi-Fi, and San Francisco** be considered one token or more?
- Should words like **Résumé, résumé, and resume** all map to the same token?
- Tokenization is even more difficult beyond English.
  - In German, for example, there are many **unsegmented compound nouns**.
  - In Chinese, there are **no spaces between** words.
  - Japanese has several **alphabets intermingled**. And so on.
  - Hindi, Marathi, Gujrati, Tamil…..

Representing Text

- What counts as a token **depends on the domain of the task** and select an appropriate tokenization technique that fits most situations well.
- In reality, it's common to **pair a standard tokenization technique with a lookup table** to address the contractions **and terms that should not** be tokenized.
- Sometimes it may not be a bad idea to develop one's **own tokenization from scratch.**

Representing Text

- Case folding reduces **all letters to lowercase** (or the opposite if applicable).
- Previous tweet, after case folding

i once had a gf back in the day. then the bphone came out lol

- If implemented incorrectly, case folding may reduce or change the meaning of the text and create additional noise.
- For example, when General Motors becomes general and motors, the downstream analysis may very likely consider them as separated words rather than the name of a company.
- When the abbreviation of the World Health Organization WHO or the rock band The Who become who, they may both be interpreted as the pronoun who.
- To handle this issue, Create a lookup table of words not to be case folded.
- Heuristics or rules-based strategies for the case folding.
- For example, the program can be taught to ignore words that have uppercase in the middle of a sentence.

Representing Text

- After tokenization and case folding, words need to be represented in a more structured way.
- One approach to represent text is called **bag-of-words**.
- Given a document, bag-of-words represents the document as a set of terms, ignoring information such as order, context, inferences, and discourse.
- Each word is considered a **term or token** (which is often the smallest unit for the analysis).
- In many cases, bag-of-words additionally assumes every term in the document is independent.
- The document then becomes a **vector with one dimension** for every **distinct term in the space**, and the **terms are unordered**.
- The permutation D* of a document D contains the same words exactly the same number of times but in a different order. Therefore, using the bag-of-words representation, document D and its permutation D* would share the same representation.

Representing Text

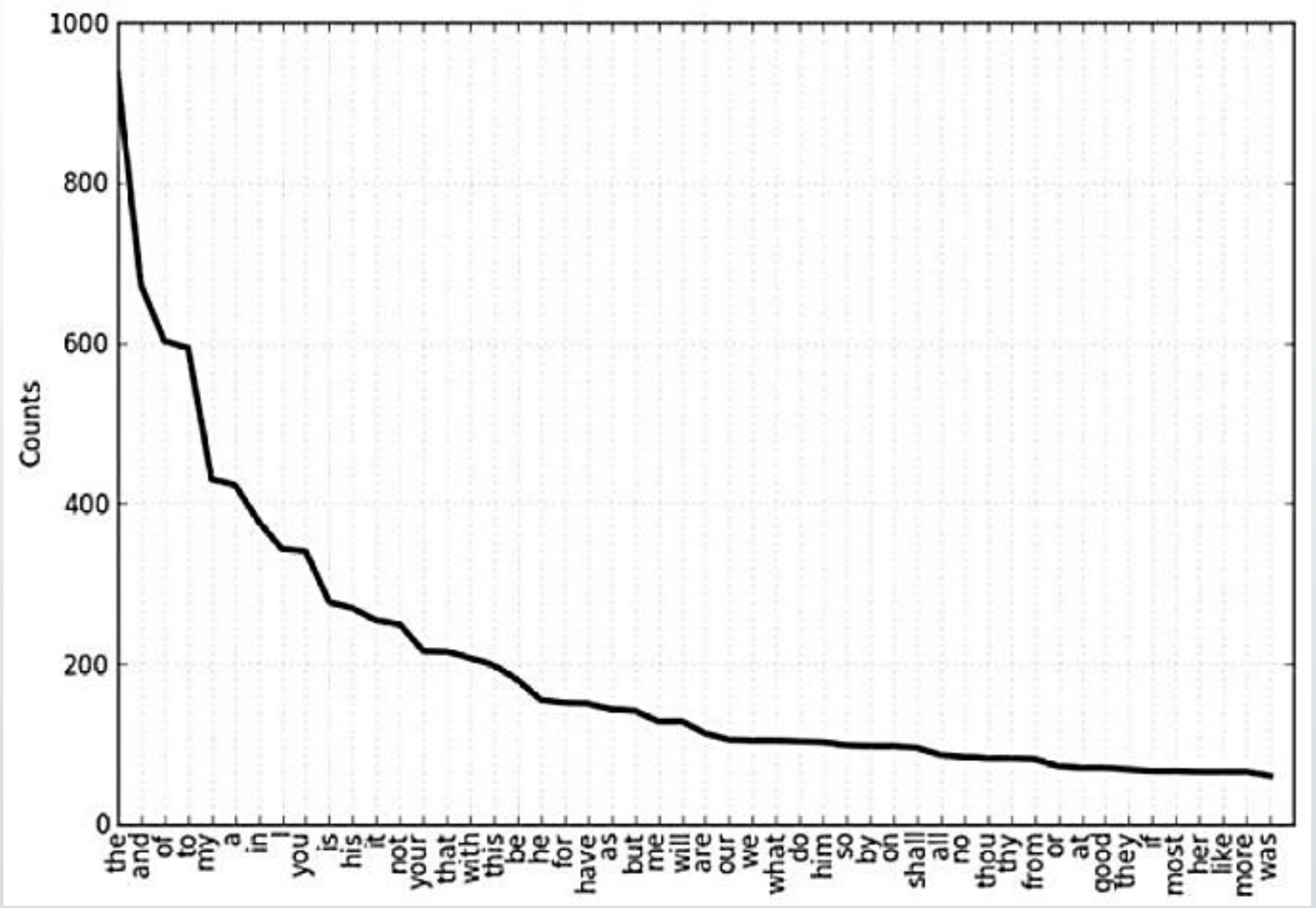
- After tokenization and case folding, words need to be represented in a more structured way.
- One approach to represent text is called **bag-of-words**.
- Given a document, bag-of-words represents the document as a set of terms, ignoring information such as order, context, inferences, and discourse.
- Each word is considered a **term or token** (which is often the smallest unit for the analysis).
- In many cases, bag-of-words additionally assumes every term in the document is independent.
- The document then becomes a **vector with one dimension** for every **distinct term in the space**, and the **terms are unordered**.
- The permutation D* of a document D contains the same words exactly the same number of times but in a different order. Therefore, using the bag-of-words representation, document D and its permutation D* would share the same representation.

Representing Text

- In paper by Salton and Buckley: the effectiveness of using single words as identifiers as opposed to multiple-term identifiers.
- **Bag-of-words uses single-term identifiers**, which are usually sufficient for the text analysis in place of multiple-term identifiers.
- **Term frequency** represents the weight of each term in a document, and it is proportional to the number of occurrences of the term in that document.
- Using single words as identifiers with the bag-of-words representation, the term frequency (TF) of each word can be calculated.

Representing Text

- The 50 most frequent words and the numbers of occurrences from Shakespeare's Hamlet



Representing Text

- The **word frequency distribution** roughly follows **Zipf's Law** that is, the i -th most common word occurs approximately 1/i as often as the most frequent term.
- In other words, **the frequency of a word is inversely proportional to its rank in the frequency table**.
- Other than bag-of-words, More advanced methods consider factors such as **word order, context, inferences, and discourse**.
- For example, one such method can keep track of the word order of every document and compare the normalized differences of the word orders.

Representing Text

- Besides extracting the terms, their **morphological features** may need to be included.
- The morphological features specify additional information about the terms, which may include **root words, affixes, part-of-speech tags, named entities, or intonation** (variations of spoken pitch).
- The features from this step contribute to the downstream analysis in classification or sentiment analysis.

Representing Text

- The set of features that need to be extracted and stored highly depends on the specific task to be performed.
- If the task is **to label and distinguish the part of speech**, for example, the features will include all the words in the text and their corresponding part-of-speech tags.
- If the task is to **annotate the named entities** like names and organizations, the features highlight such information appearing in the text.
- Constructing the features is no trivial task; quite often this is done entirely manually, and sometimes it requires domain expertise.

Representing Text

- Other method for creating features: **Topic modeling** provides a way to quickly analyze large volumes of raw text and identify the latent topics.
- Topic modeling may not require the documents to be labeled or annotated. It can discover topics directly from an analysis of the raw text.
- A topic consists of a cluster of words that frequently occur together and that share the same theme.

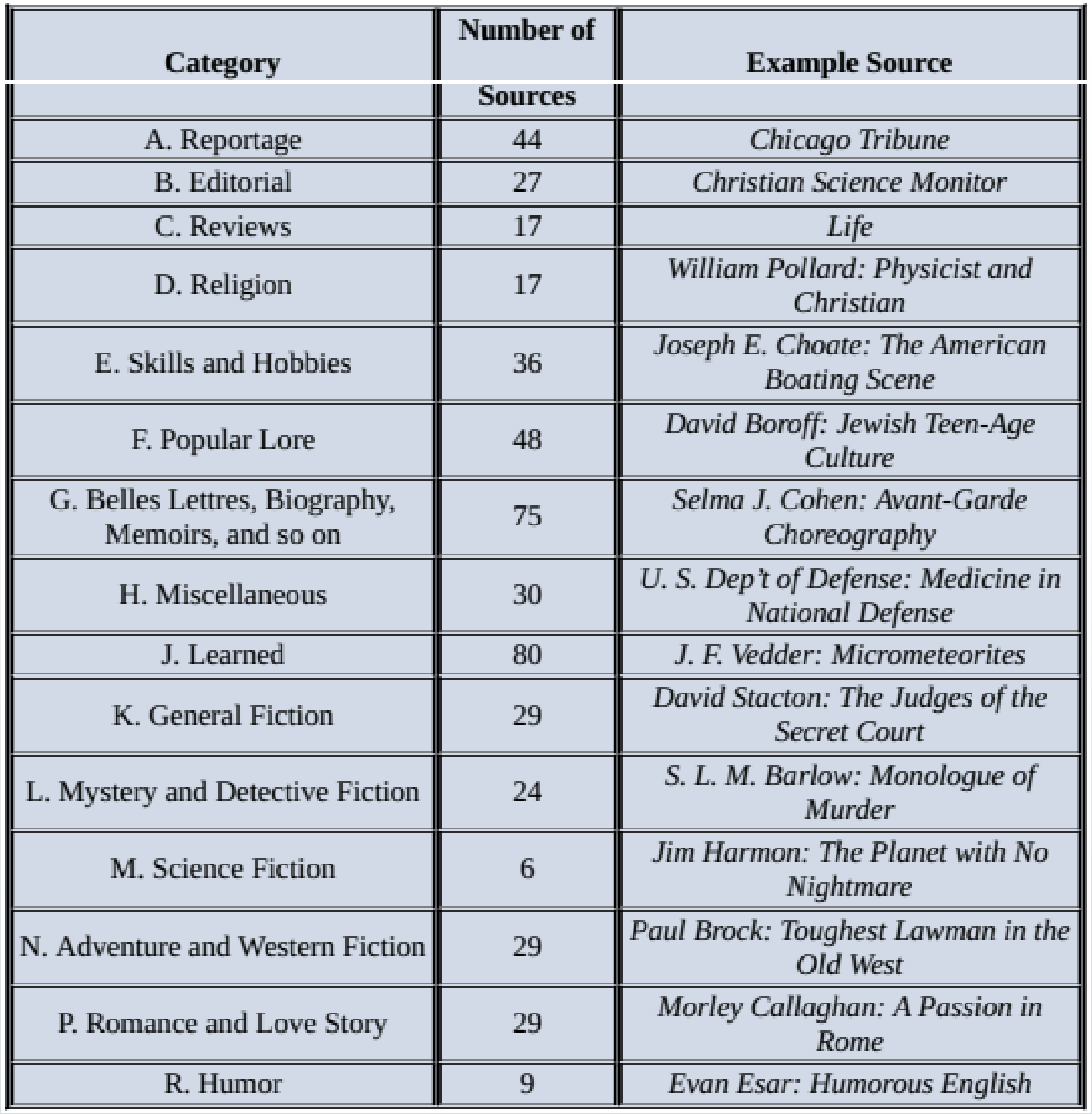
Representing Text

- It is important to create representation of corpus also.
- A **corpus** is a collection of documents.
- A corpus could be so large that it includes all the documents in one or more languages, or it could be smaller or limited to a specific domain, such as technology, medicine, or law.
- For a web search engine, the entire World Wide Web is the relevant corpus.
- Most corpora are much smaller.
- The Brown Corpus [15] was the first million-word electronic corpus of English, created in 1961 at Brown University.
- It includes text from around 500 sources, and the source has been categorized into 15 genres, such as news, editorial, fiction, and so on.

Representing Text

- Categories of Brown Corpus

- Many corpora focus on specific domains.
- For example, the **BioCreative corpora** are from biology,
- The **Switchboard corpus** contains telephone conversations, and the
- **European Parliament Proceedings Parallel Corpus** was extracted from the proceedings of the European Parliament in 21 European languages.



Category	Number of Sources	Example Source
A. Reportage	44	*Chicago Tribune*
B. Editorial	27	*Christian Science Monitor*
C. Reviews	17	*Life*
D. Religion	17	*William Pollard: Physicist and Christian*
E. Skills and Hobbies	36	*Joseph E. Choate: The American Boating Scene*
F. Popular Lore	48	*David Boroff: Jewish Teen-Age Culture*
G. Belles Lettres, Biography, Memoirs, and so on	75	*Selma J. Cohen: Avant-Garde Choreography*
H. Miscellaneous	30	*U. S. Dep't of Defense: Medicine in National Defense*
J. Learned	80	*J. F. Vedder: Micrometeorites*
K. General Fiction	29	*David Stacton: The Judges of the Secret Court*
L. Mystery and Detective Fiction	24	*S. L. M. Barlow: Monologue of Murder*
M. Science Fiction	6	*Jim Harmon: The Planet with No Nightmare*
N. Adventure and Western Fiction	29	*Paul Brock: Toughest Lawman in the Old West*
P. Romance and Love Story	29	*Morley Callaghan: A Passion in Rome*
R. Humor	9	*Evan Esar: Humorous English*

Representing Text

- Most corpora come with **metadata**, such as the size of the corpus and the domains from which the text is extracted.
- Some corpora (such as the Brown Corpus) include the **information content of every word appearing in the text**. **Information content** (IC) is a metric to denote the importance of a term in a corpus.
- The conventional way of measuring the IC of a term is to **combine the knowledge of its hierarchical structure from an ontology with statistics on its actual usage in text derived from a corpus**. Terms with **higher IC values** are considered **more important** than terms with lower IC values.
- For example, the word necklace generally has a higher IC value than the word jewelry in an English corpus because jewelry is more general and is likely to appear more often than necklace.
- Research shows that IC can help measure **the semantic similarity of terms**.
- In addition, such measures do not require an annotated corpus, and they generally achieve strong correlations with human judgment.

Representing Text

- Most corpora come with **metadata**, such as the size of the corpus and the domains from which the text is extracted.
- Some corpora (such as the Brown Corpus) include the **information content of every word appearing in the text**. **Information content** (IC) is a metric to denote the importance of a term in a corpus.
- The conventional way of measuring the IC of a term is to **combine the knowledge of its hierarchical structure from an ontology with statistics on its actual usage in text derived from a corpus**. Terms with **higher IC values** are considered **more important** than terms with lower IC values.
- For example, the word necklace generally has a higher IC value than the word jewelry in an English corpus because jewelry is more general and is likely to appear more often than necklace.
- Research shows that IC can help measure **the semantic similarity of terms**.
- In addition, such measures do not require an annotated corpus, and they generally achieve strong correlations with human judgment.

Representing Text

- In example, the team has collected the ACME product reviews and turned them into the proper representation with the techniques discussed.
- Next, the reviews and the representation need to be stored in a searchable archive for future reference and research.
- This archive could be a SQL database, XML or JSON files, or plain text files from one or more directories.

Representing Text

- Issues with corpus statistics or IC values.

1. Both traditional corpora and IC metadata do not change over time. Any term not existing in the corpus text and any newly invented words would automatically receive a zero IC value.
2. The corpus represents the entire knowledge base for the algorithm being used in the downstream analysis. The nature of the unstructured text determines that the data being analyzed can contain any topics, many of which may be absent in the given knowledge base.

- If the task is to research people's attitudes on musicians, a traditional corpus constructed 50 years ago would not know that the term U2 is a band; therefore, it would receive a zero on IC, which means it's not an important term

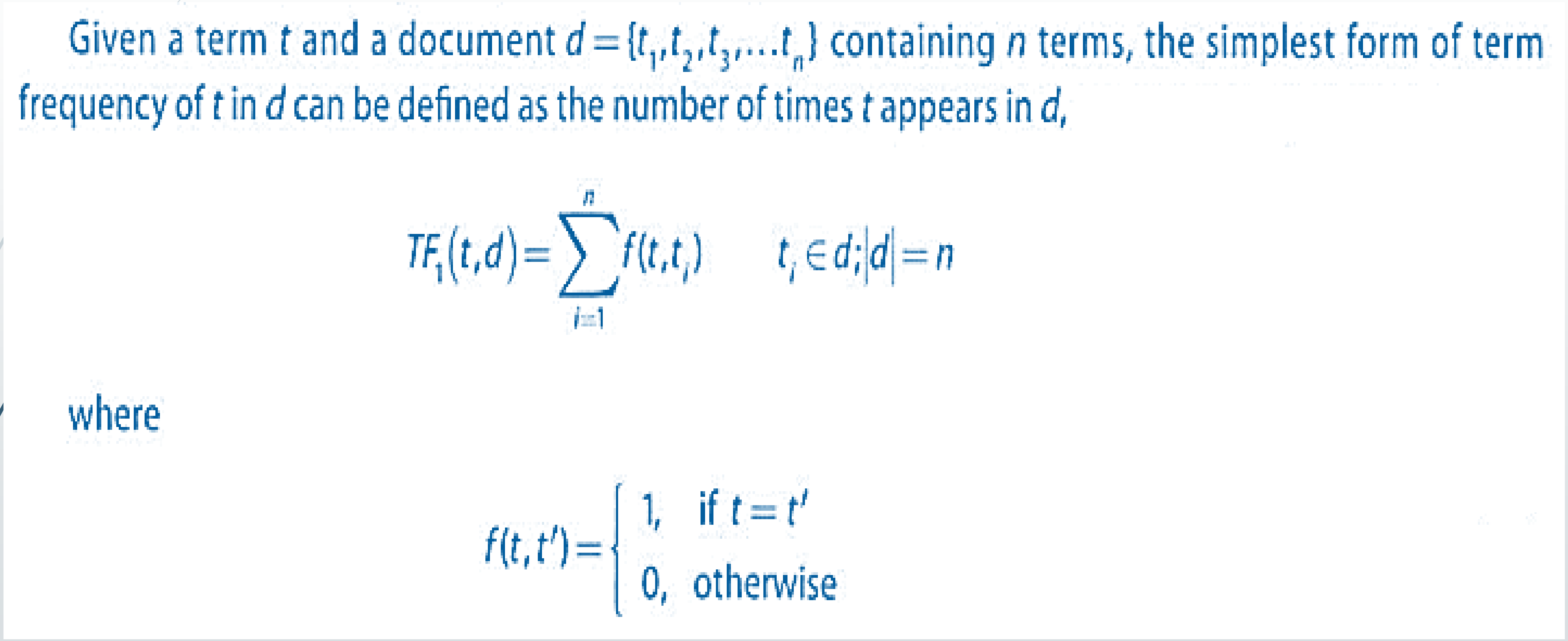
Representing Text

- It is necessary to come up with a metric that can easily **adapt to the context and nature of the text** instead of relying on a traditional corpus.
- Such a metric is **Term Frequency—Inverse Document Frequency (TFIDF)**, which is based entirely on all the fetched documents and which keeps track of the importance of terms occurring in each of the documents.

Term Frequency—Inverse Document Frequency (TFIDF)

- TFIDF, a measure widely used in information retrieval and text analysis.
- Instead of using a traditional corpus as a knowledge base, TFIDF directly works on top of the fetched documents and treats these documents as the "corpus."
- **TFIDF is robust and efficient on dynamic content, because document changes require only the update of frequency counts.**

TFIDF



Given a term t and a document $d = \{t_1, t_2, t_3, \ldots t_n\}$ containing n terms, the simplest form of term frequency of t in d can be defined as the number of times t appears in d,

$$TF_1(t,d) = \sum_{i=1}^{n} f(t,t_i) \qquad t_i \in d; |d| = n$$

where

$$f(t,t') = \begin{cases} 1, & \text{if } t = t' \\ 0, & \text{otherwise} \end{cases}$$

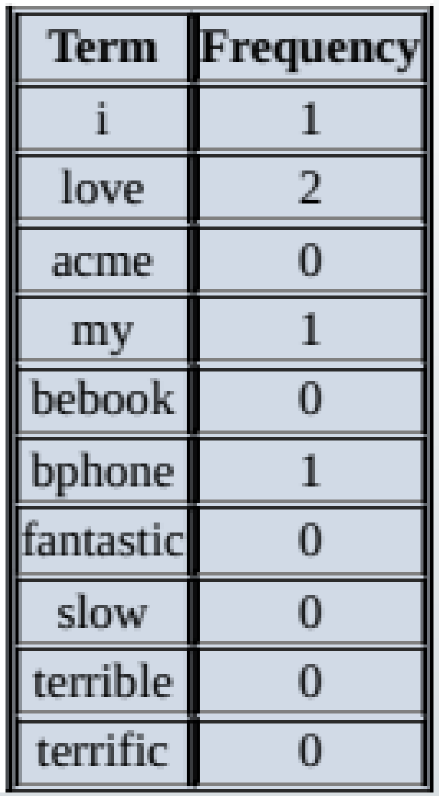
TFIDF

- To understand how the term frequency is computed, consider a bag-of-words vector space of 10 words:

i, love, acme, my, bebook, bphone, fantastic, slow, terrible, and terrific.

- Given the text I love LOVE my bPhone extracted from the RSS feed.
- Table shows term frequency vector after case folding and tokenization.
- The term frequency function can be logarithmically scaled, can be applied to word frequencies whose distribution also contains a long tail.

$$TF_2(t,d) = \log[TF_1(t,d) + 1]$$



Term	Frequency
i	1
love	2
acme	0
my	1
bebook	0
bphone	1
fantastic	0
slow	0
terrible	0
terrific	0

TFIDF

- Because longer documents contain more terms, they tend to have higher term frequency values.
- They also tend to contain more distinct terms.
- These factors can conspire to raise the term frequency values of longer documents and lead to undesirable bias favoring longer documents.
- To address this problem, the term frequency can be normalized.
- For example, the term frequency of term t in document d can be normalized based on the number of terms in d as shown in

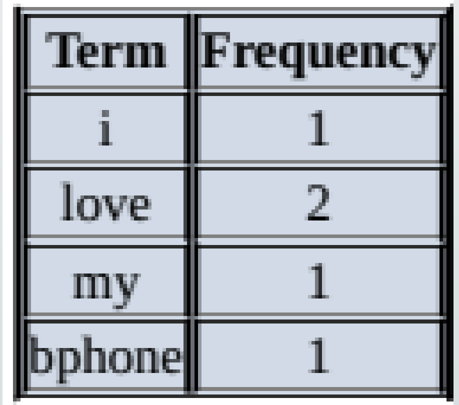
$$TF_3(t,d) = \frac{TF_1(t,d)}{n} \qquad |d| = n$$

TFIDF

- A **term frequency vector** (shown in Table previously) can become very high dimensional because the bag-of-words vector space can grow substantially to include all the words in English.
- The high dimensionality makes it **difficult to store and parse the text and contribute to performance issues** related to text analysis.
- For the purpose of **reducing dimensionality**, not all the words from a given language need to be included in the term frequency vector.
- In English, for example, it is common to remove words such **as the, a, of, and, to, and other articles** that are not likely to contribute **to semantic** understanding.
- These common words are called **STOP WORDS**.
- Lists of stop words are available in various languages for automating the identification of stop words. Among them is the **Snowball's stop words** list contains stop words in more than ten languages.

TFIDF

- Other way to reduce dimensionality is to **store a term and its frequency only if the term appears at least once in a document**.
- Any term not existing in the term frequency vector by default will have a frequency of 0.
- Therefore, the previous term frequency vector would be simplified to what is shown here.
- Lemmatization and Stemming can also reduce high dimensionality.



Term	Frequency
i	1
love	2
my	1
bphone	1

TFIDF

- Besides stop words, **words that are more general** in meaning tend to appear more often, thus having **higher term frequencies**.
- In an article about consumer telecommunications, the word ***phone*** would be likely to receive a high term frequency.
- As a result, the important keywords such as ***bPhone*** and bEbook and their related words could appear to be **less important**.
- In such case the search engine will not be able to assess how relevant each document is.
- Considering the importance of a term not only in a single document but in a collection of documents, or in a corpus.
- The additional variable **should reduce the effect of the term frequency** as the term appears in more documents.

TFIDF

- This additional variable is **inverted document frequency** (IDF), which is defined to be the number of documents in the corpus that contain a term.
- The IDF inversely corresponds to the document frequency (DF).
- Let a corpus D contain N documents. The document frequency of a term t in corpus D={d1,d2,d3…,dn} is defined as

$$DF(t) = \sum_{i=1}^{N} f'(t, d_i) \qquad d_i \in D; |D| = N$$

where

$$f'(t, d') = \begin{cases} 1, & \text{if } t \in d' \\ 0, & \text{otherwise} \end{cases}$$

TFIDF

- The Inverse document frequency of a term t is obtained by dividing N by the document frequency of the term and then taking the logarithm of that quotient,

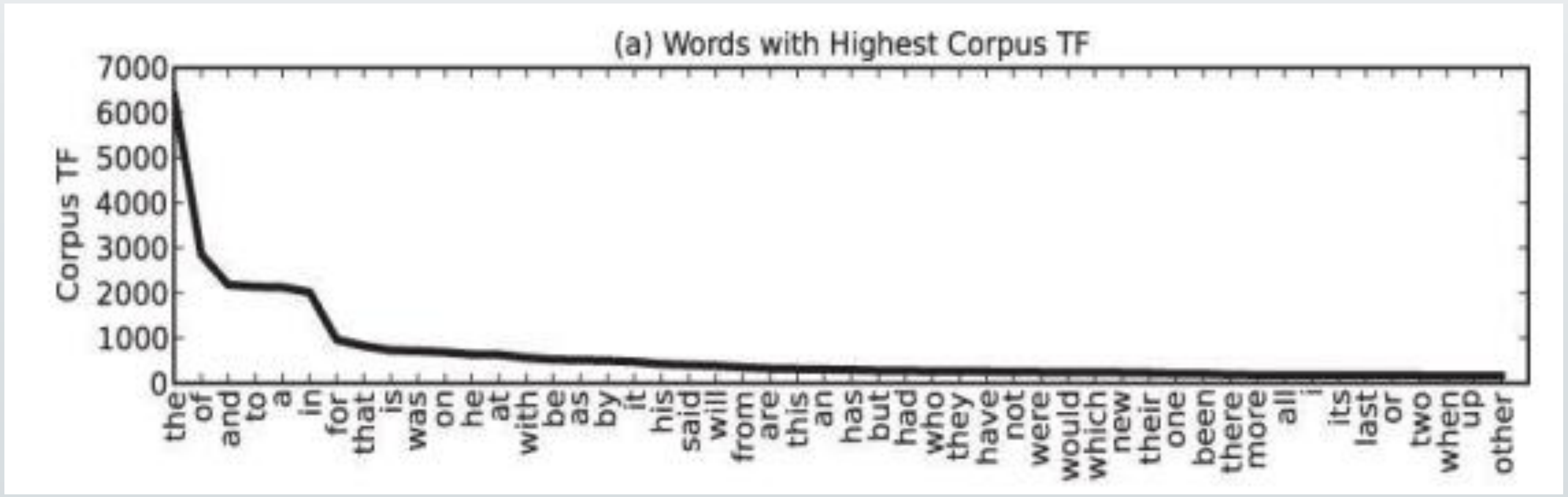
$$IDF_1(t) = \log \frac{N}{DF(t)}$$

- If the term is not in the corpus, it leads to a division-by-zero. A quick fix is to add 1 to the denominator,

$$IDF_2(t) = \log \frac{N}{DF(t)+1}$$

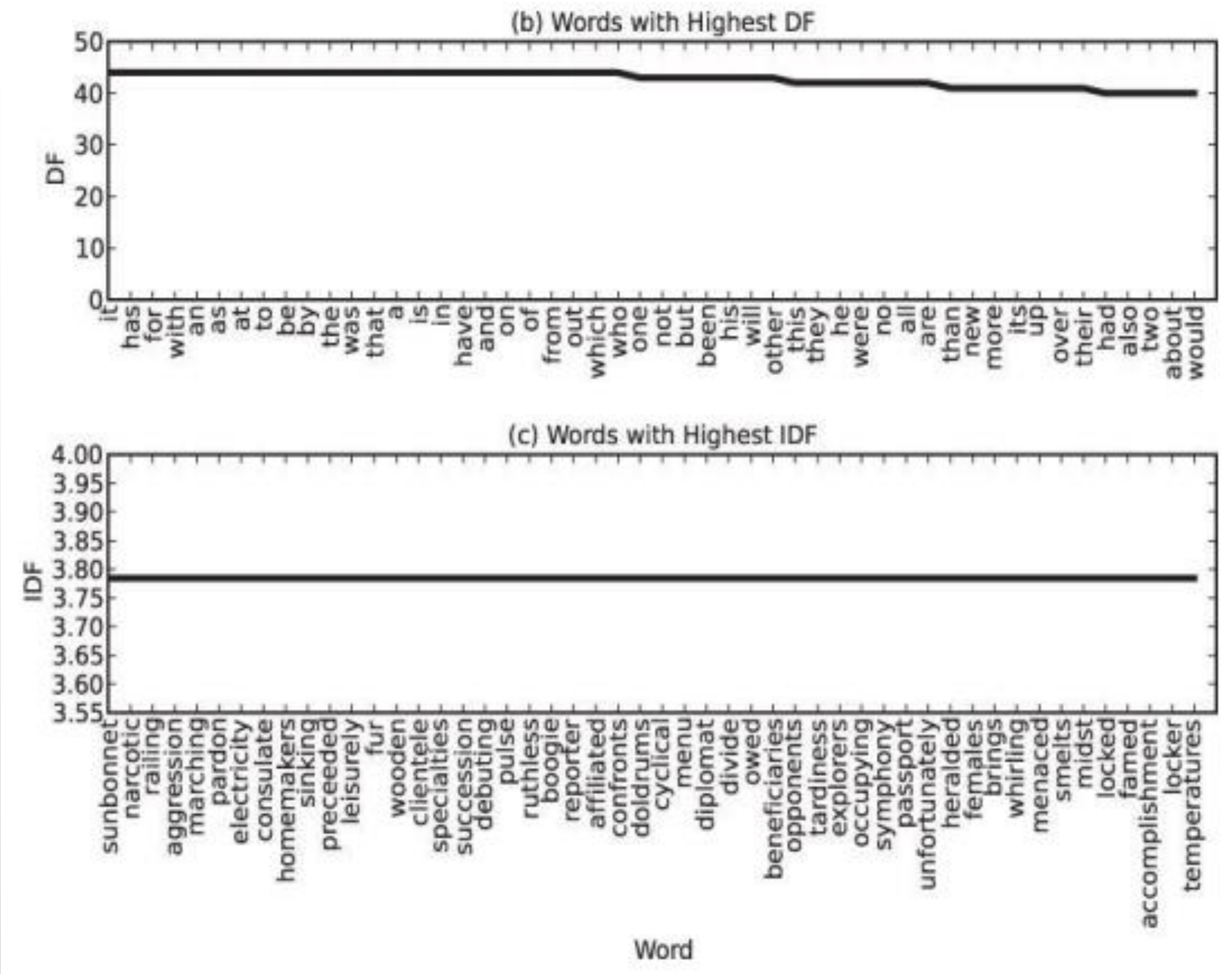
TFIDF

- Figure shows 50 words with (a) the highest corpus-wide term frequencies (TF), (b) the highest document frequencies (DF), and (c) the highest Inverse document frequencies (IDF) from the news category of the Brown Corpus.
- Stop words tend to have higher TF and DF because they are likely to appear more often in most documents.



TFIDF

- Figure shows 50 words with (a) the highest corpus-wide term frequencies (TF), (b) the highest document frequencies (DF), and (c) the highest Inverse document frequencies (IDF) from the news category of the Brown Corpus.



TFIDF

- Words with higher IDF tend to be more meaningful over the entire corpus.
- In other words, **the IDF of a rare term would be high**, and the IDF of a frequent term would be low.
- For example, if a corpus contains 1,000 documents, 1,000 of them might contain the word the, and 10 of them might contain the word bPhone.
- The IDF of the would be 0, and the IDF of bPhone would be log100, which is greater than the IDF of *the*.
- If a corpus consists of mostly phone reviews, the word phone would probably have high TF and DF but low IDF.

TFIDF

- Because the total document count of a corpus (N) remains a constant, IDF solely depends on the DF.
- All words having the **same DF value** therefore receive the **same IDF value**.
- IDF scores words higher that occur less frequently across the documents.
- Those words that score the lowest DF receive the same highest IDF.
- For example, *sunbonnet* and *narcotic* appeared in an equal number of documents in the Brown corpus; therefore, they received the same IDF values.
- To distinguish between two words that appear in an equal number of documents, methods to further weight words should be considered to refine the IDF score.

TFIDF

- The TFIDF (or TF-IDF) is a measure that considers both the prevalence of a term within a document (TF) and the scarcity of the term over the entire corpus (IDF).
- The TFIDF of a term *t* in a document *d* is defined as the term frequency of *t* in *d* multiplying the document frequency of *t* in the corpus as

$$TFIDF(t,d) = TF(t,d) \times IDF(t)$$

TFIDF

- TFIDF scores words higher that appear more often in a document but occur less often across all documents in the corpus.
- Note that **TFIDF applies to a term in a specific document, so the same term is likely to receive different TFIDF scores in different documents** (because the TF values may be different).
- TFIDF is efficient in that the calculations are simple and straightforward, and it does not require knowledge of the underlying meanings of the text.
- But this approach also reveals **little of the inter-document or intra-document statistical structure**.
- The next part we will see how **topic models** can address this shortcoming of TFIDF.

Lecture 4

Categorizing Documents by Topics

- ACME wants to categorize the reviews by topics.
- **A topic consists of a cluster of words that frequently occur together and share the same theme.**
- The topics of a document are not as straightforward as they might initially appear.
- *Consider these two reviews:*

1. The bPhone5x has coverage everywhere. It's much less flaky than my old bPhone4G.

2. While I love ACME's bPhone series, I've been quite disappointed by the bEbook. The text is illegible, and it makes even my old NBook look blazingly fast.

- Is the first review about bPhone5x or bPhone4G? Is the second review about bPhone, bEbook, or NBook? For machines, these questions can be difficult to answer.

Categorizing Documents by Topics

- If a review is talking about bPhone5x, the term bPhone5x and related terms (such as phone and ACME) are likely to appear frequently.
- A document typically consists of multiple themes running through the text in different proportions:
- for example, 30% on a topic related to phones, 15% on a topic related to appearance, 10% on a topic related to shipping, 5% on a topic related to service, and so on.

Categorizing Documents by Topics

- Document grouping can be achieved with clustering methods such as k-means clustering or classification methods such as support vector machines, k-nearest neighbors, or naïve Bayes.
- However, a more feasible and prevalent approach is to use topic modeling.
- **Topic modeling provides tools to automatically organize, search, understand, and summarize from vast amounts of information.**

Categorizing Documents by Topics

- Topic models are statistical models that examine words from a set of documents, determine the themes over the text, and discover how the themes are associated or change over time.
- The process of topic modeling can be simplified to the following.

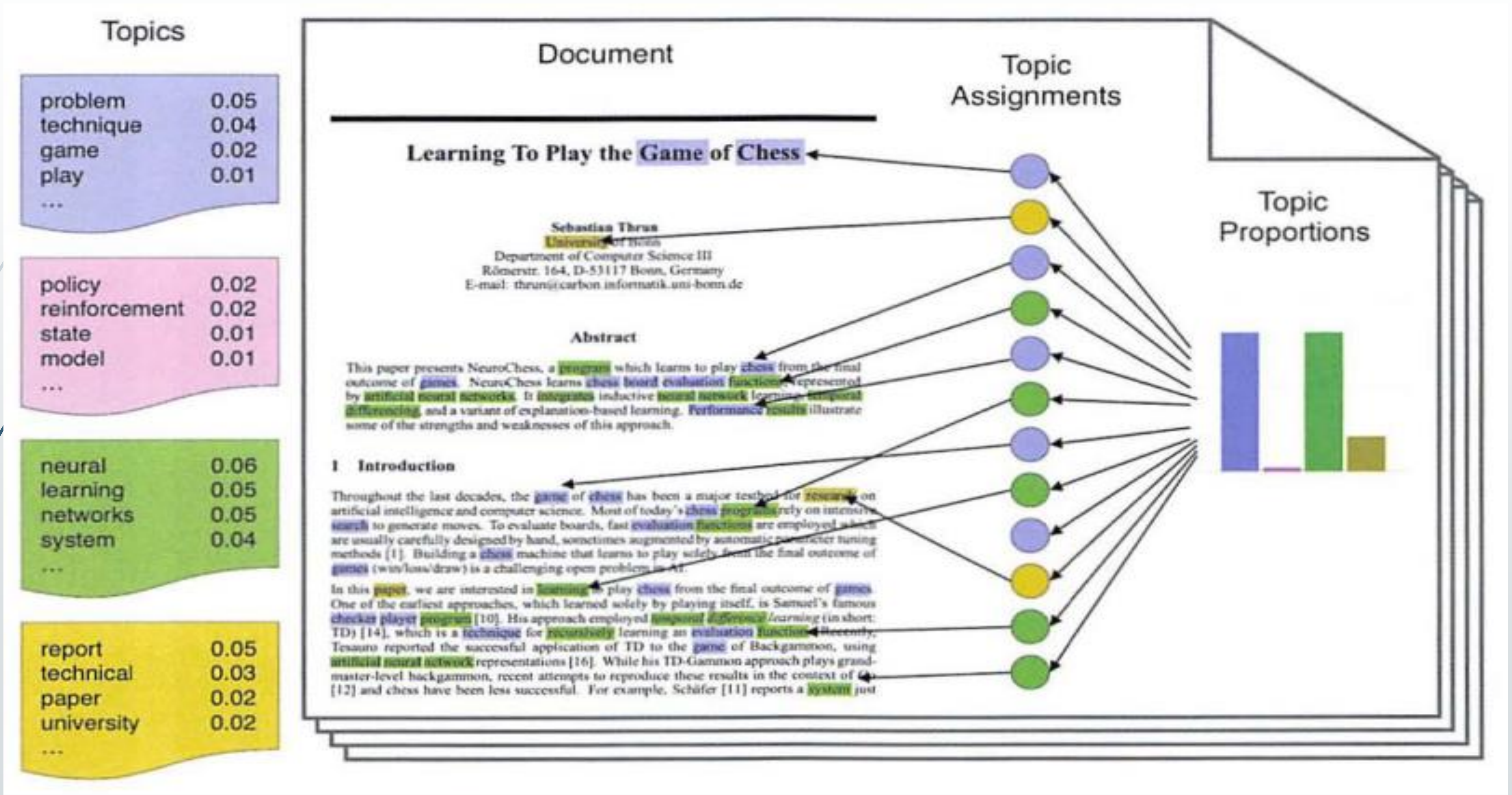
1. Uncover the hidden topical patterns within a corpus.
2. Annotate documents according to these topics.
3. Use annotations to organize, search, and summarize texts.

- A word from the vocabulary can reside in multiple topics with different weights. Topic models do not necessarily require prior knowledge of the texts. The topics can emerge solely based on analyzing the text.

Categorizing Documents by Topics

- The simplest topic model is **latent Dirichlet allocation** (LDA), a **generative probabilistic model** of a corpus proposed by David M. Blei and two other researchers.
- In generative probabilistic modeling, data is treated as the result of a generative process that includes hidden variables.
- LDA assumes that there is a fixed vocabulary of words, and the number of the latent topics is predefined and remains constant.
- LDA assumes that each latent topic follows a Dirichlet distribution over the vocabulary, and each document is represented as a random mixture of latent topics.

Categorizing Documents by Topics

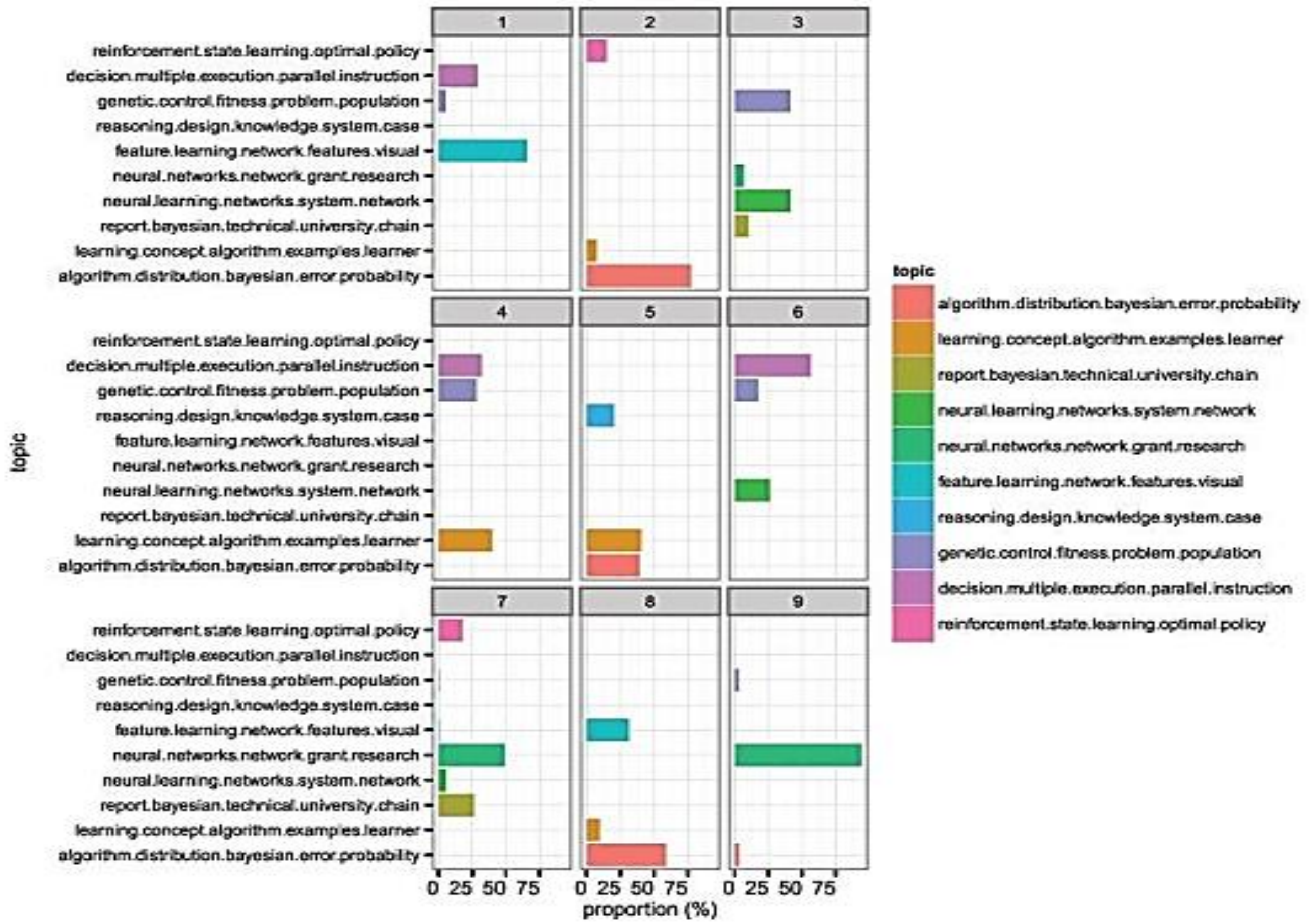


Categorizing Documents by Topics

- The left side of the figure shows four topics built from a corpus, where each topic contains a list of the most important words from the vocabulary.
- The four example topics are related to problem, policy, neural, and report.
- For each document, a distribution over the topics is chosen, as shown in the histogram on the right. Next, a topic assignment is picked for each word in the document, and the word from the corresponding topic (colored discs) is chosen.
- In reality, only the documents (as shown in the middle of the figure) are available. The goal of LDA is to infer the underlying topics, topic proportions, and topic assignments for every document.

Categorizing Documents by Topics

- Many programming tools provide software packages that can perform LDA over datasets.
- R comes with an ***lda*** package, that has built-in functions and sample datasets.
- The ***lda*** package was developed by David M. Blei's research group.
- Figure in next slide will show the distributions of ten topics on nine scientific documents randomly drawn from the ***cora*** dataset of the ***lda*** package.
- The ***cora*** dataset is a collection of 2,410 scientific documents extracted from the Cora search engine.


1
2
3
4
5
6
7
8
9
topic
reinforcement.state.learning.optimal.policy
decision.multiple.execution.parallel.instruction
genetic.control.fitness.problem.population
reasoning.design.knowledge.system.case
feature.learning.network.features.visual
neural.networks.network.grant.research
neural.learning.networks.system.network
report.bayesian.technical.university.chain
learning.concept.algorithm.examples.learner
algorithm.distribution.bayesian.error.probability
0 25 50 75
proportion (%)

Categorizing Documents by Topics

- The code that follows shows how to generate a graph using R and add-on packages such as lda and ggplot.

```
require("ggplot2")
require("reshape2")
require("lda")
# load documents and vocabulary
data(cora.documents)
data(cora.vocab)
theme_set(theme_bw())
# Number of topic clusters to display
K <- 10
# Number of documents to display
N <- 9
result <- lda.collapsed.gibbs.sampler(cora.documents,
K, ## Num clusters
cora.vocab,
25, ## Num iterations
0.1,0.1,
compute.log.likelihood=TRUE)
```

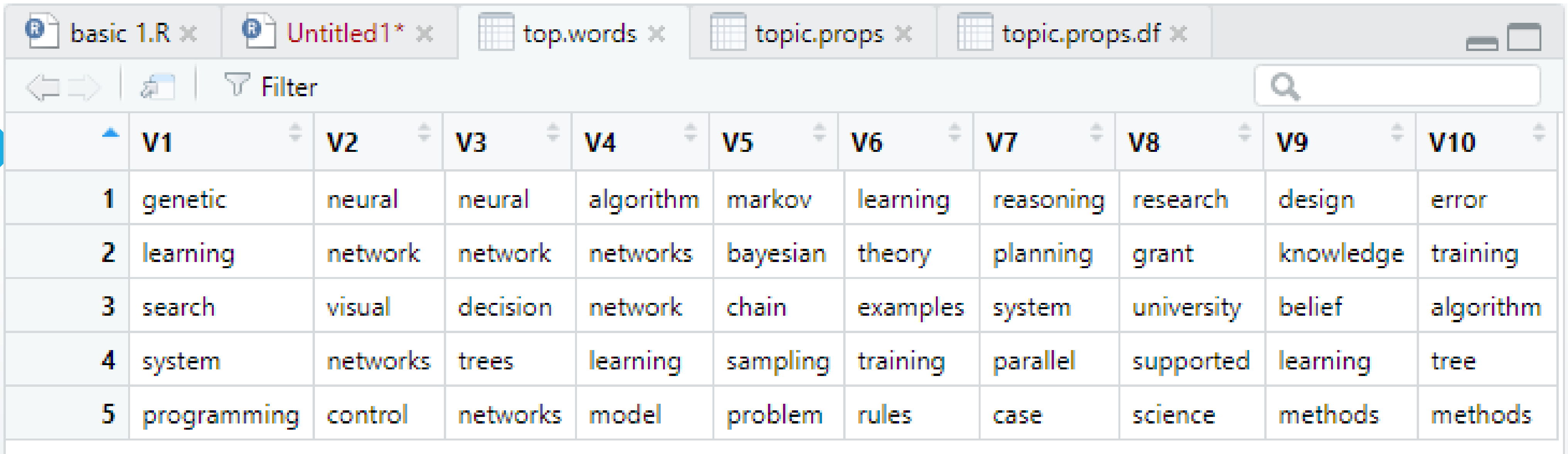
Categorizing Documents by Topics

# Get the top words in the cluster

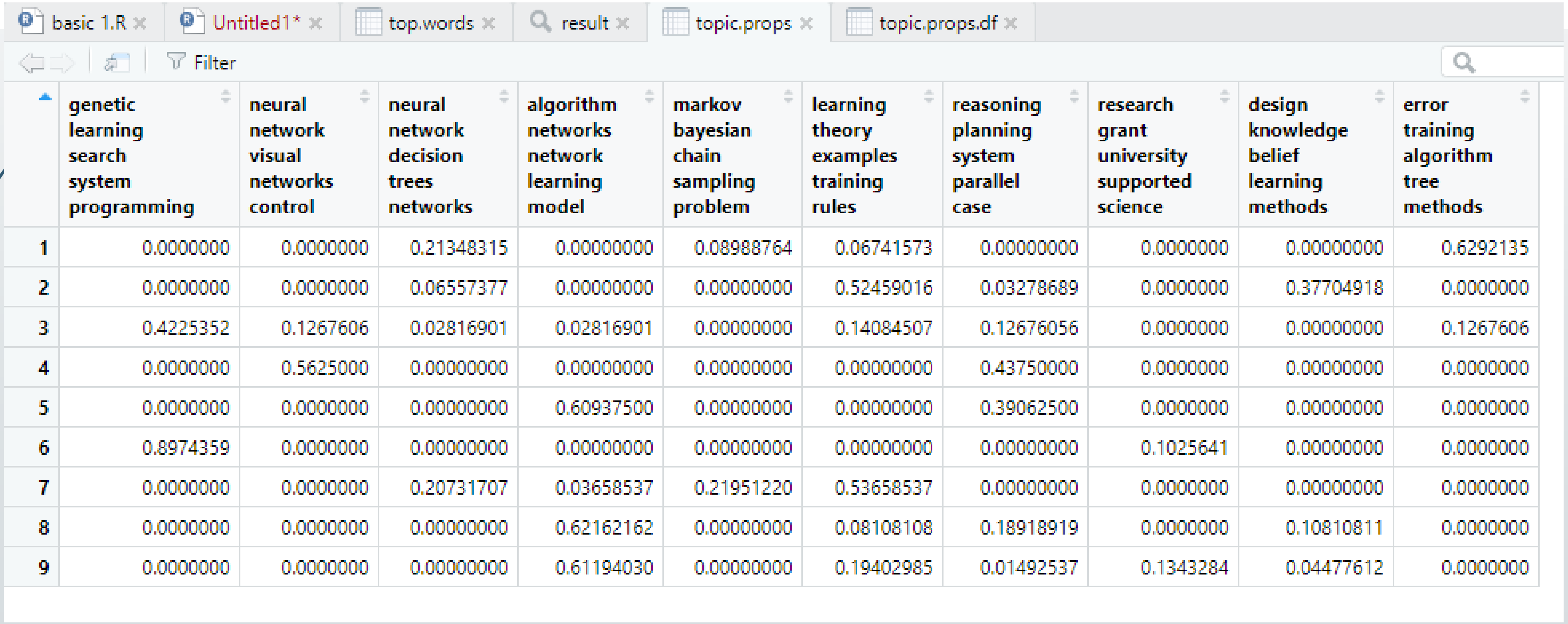
- top.words <- top.topic.words(result$topics, 5, by.score=TRUE)

# build topic proportions

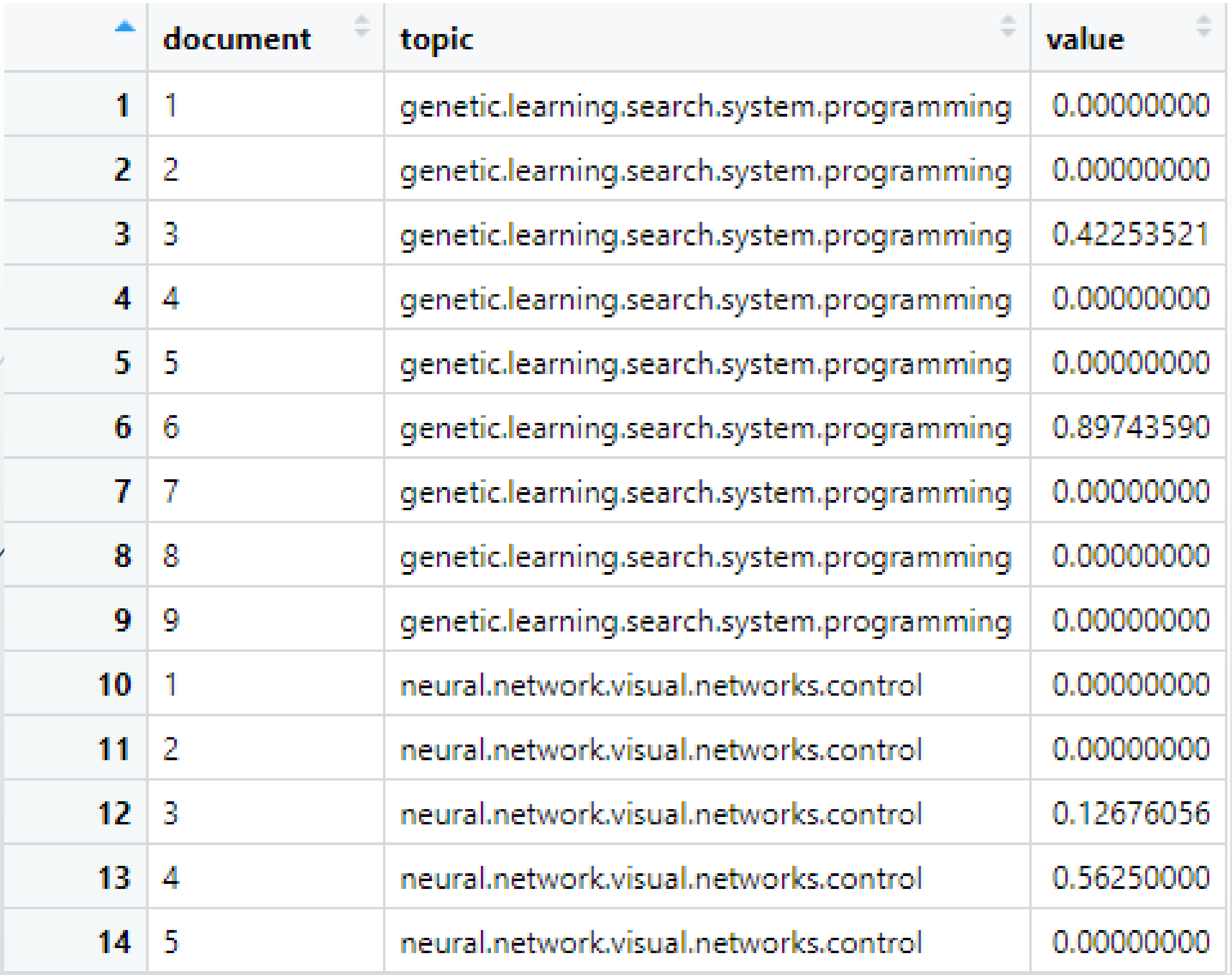
- topic.props <- t(result$document_sums) / colSums(result$document_sums)
- document.samples <- sample(1:dim(topic.props)[1], N)
- topic.props <- topic.props[document.samples,]
- topic.props[is.na(topic.props)] <- 1 / K
- colnames(topic.props) <- apply(top.words, 2, paste, collapse=" ")
- topic.props.df <- melt(cbind(data.frame(topic.props),
- document=factor(1:N)),
- variable.name="topic",
- id.vars = "document")
- qplot(topic, value*100, fill=topic, stat="identity",
- ylab="proportion (%)", data=topic.props.df,
- geom="histogram") +
- theme(axis.text.x = element_text(angle=0, hjust=1, size=12)) +
- coord_flip() +
- facet_wrap(~ document, ncol=3)



	V1	V2	V3	V4	V5	V6	V7	V8	V9	V10
1	genetic	neural	neural	algorithm	markov	learning	reasoning	research	design	error
2	learning	network	network	networks	bayesian	theory	planning	grant	knowledge	training
3	search	visual	decision	network	chain	examples	system	university	belief	algorithm
4	system	networks	trees	learning	sampling	training	parallel	supported	learning	tree
5	programming	control	networks	model	problem	rules	case	science	methods	methods



	genetic learning search system programming	neural network visual networks control	neural network decision trees networks	algorithm networks network learning model	markov bayesian chain sampling problem	learning theory examples training rules	reasoning planning system parallel case	research grant university supported science	design knowledge belief learning methods	error training algorithm tree methods
1	0.0000000	0.0000000	0.21348315	0.00000000	0.08988764	0.06741573	0.00000000	0.0000000	0.00000000	0.6292135
2	0.0000000	0.0000000	0.06557377	0.00000000	0.00000000	0.52459016	0.03278689	0.0000000	0.37704918	0.0000000
3	0.4225352	0.1267606	0.02816901	0.02816901	0.00000000	0.14084507	0.12676056	0.0000000	0.00000000	0.1267606
4	0.0000000	0.5625000	0.00000000	0.00000000	0.00000000	0.00000000	0.43750000	0.0000000	0.00000000	0.0000000
5	0.0000000	0.0000000	0.00000000	0.60937500	0.00000000	0.00000000	0.39062500	0.0000000	0.00000000	0.0000000
6	0.8974359	0.0000000	0.00000000	0.00000000	0.00000000	0.00000000	0.00000000	0.1025641	0.00000000	0.0000000
7	0.0000000	0.0000000	0.20731707	0.03658537	0.21951220	0.53658537	0.00000000	0.0000000	0.00000000	0.0000000
8	0.0000000	0.0000000	0.00000000	0.62162162	0.00000000	0.08108108	0.18918919	0.0000000	0.10810811	0.0000000
9	0.0000000	0.0000000	0.00000000	0.61194030	0.00000000	0.19402985	0.01492537	0.1343284	0.04477612	0.0000000



	document	topic	value
1	1	genetic.learning.search.system.programming	0.00000000
2	2	genetic.learning.search.system.programming	0.00000000
3	3	genetic.learning.search.system.programming	0.42253521
4	4	genetic.learning.search.system.programming	0.00000000
5	5	genetic.learning.search.system.programming	0.00000000
6	6	genetic.learning.search.system.programming	0.89743590
7	7	genetic.learning.search.system.programming	0.00000000
8	8	genetic.learning.search.system.programming	0.00000000
9	9	genetic.learning.search.system.programming	0.00000000
10	1	neural.network.visual.networks.control	0.00000000
11	2	neural.network.visual.networks.control	0.00000000
12	3	neural.network.visual.networks.control	0.12676056
13	4	neural.network.visual.networks.control	0.56250000
14	5	neural.network.visual.networks.control	0.00000000

Categorizing Documents by Topics

- Topic models can be used in document modeling, document classification, and collaborative filtering.
- Topic models not only can be applied to textual data, they can also help annotate images.
- Just as a document can be considered a collection of topics, images can be considered a collection of image features.

Determining Sentiments

- **Sentiment analysis** refers to a group of tasks that use statistics and natural language processing to mine opinions to identify and extract subjective information from texts.
- Early work on sentiment analysis focused on **detecting the polarity of product reviews** from Epinions and **movie reviews** from the Internet Movie Database (IMDb) at the document level.
- Later work handles sentiment analysis at the **sentence level**. More recently, the focus has shifted to **phrase-level and short-text forms** in response to the popularity of micro-blogging services such as Twitter.

Determining Sentiments

- To conduct sentiment analysis, one can manually construct lists of words with **positive sentiments** (such as brilliant, awesome, and spectacular) and **negative sentiments** (such as awful, stupid, and hideous).
- Such an approach can be expected to achieve accuracy around **60%**, and only **examination by corpus statistics** can outperform this.

Determining Sentiments

- Classification methods such as Naive Bayes, maximum entropy (MaxEnt), and support vector machines (SVM) are often used to extract corpus statistics for sentiment analysis.
- Related research has found out that these classifiers can score around 80% accuracy on sentiment analysis over unstructured data.

Determining Sentiments

- Depending on the classifier, the data may need to be **split into training and testing** sets. A useful rule of the thumb for splitting data is to produce a training set much bigger than the testing set.
- For example, **an 80/20 split** would produce 80% of the data as the training set and 20% as the testing set.
- **One or more classifiers are trained over the training set to learn the characteristics or patterns residing in the data.**
- The **sentiment tags in the testing data are hidden** away from the classifiers.
- After the training, **classifiers are tested** over the testing set to infer the **sentiment tags**.
- Finally, the **result is compared against the original sentiment tags** to evaluate the **overall performance** of the classifier.

Determining Sentiments

- To perform sentiment analysis using the naïve Bayes classifier over the movie review corpus.

```
import nltk.classify.util
from nltk.classify import
NaiveBayesClassifier
from nltk.corpus import movie_reviews
from collections import defaultdict
import numpy as np
# define an 80/20 split for train/test
SPLIT = 0.8
def word_feats(words):
feats = defaultdict(lambda: False)
for word in words:
      feats[word] = True
return feats
```

```
posids = movie_reviews.fileids(‘pos’)
negids = movie_reviews.fileids(‘neg’)
posfeats =
[(word_feats(movie_reviews.words(fileids=[f])),
‘pos’)
      for f in posids]
negfeats =
[(word_feats(movie_reviews.words(fileids=[f])),
‘neg’)
      for f in negids]
```

Determining Sentiments

- To perform sentiment analysis using the naïve Bayes classifier over the movie review corpus.

```
cutoff = int(len(posfeats) * SPLIT)
trainfeats = negfeats[:cutoff] + posfeats[:cutoff]
testfeats = negfeats[cutoff:] + posfeats[cutoff:]
print ‘Train on %d instances\nTest on %d instances’ % (len(trainfeats),
len(testfeats))
classifier = NaiveBayesClassifier.train(trainfeats)
print ‘Accuracy:’, nltk.classify.util.accuracy(classifier, testfeats)
classifier.show_most_informative_features()
# prepare confusion matrix
pos = [classifier.classify(fs) for (fs,l) in posfeats[cutoff:]]
pos = np.array(pos)
neg = [classifier.classify(fs) for (fs,l) in negfeats[cutoff:]]
neg = np.array(neg)
```

Determining Sentiments

- To perform sentiment analysis using the naïve Bayes classifier over the movie review corpus.

```
print ‘Confusion matrix:’
print ‘\t’*2, ‘Predicted class’
print ‘-‘*40
print ‘|\t %d (TP) \t|\t %d (FN) \t| Actual class’ % (
(pos == ‘pos’).sum(), (pos == ‘neg’).sum()
print ‘-‘*40
print ‘|\t %d (FP) \t|\t %d (TN) \t|’ % (
(neg == ‘pos’).sum(), (neg == ‘neg’).sum())
print ‘-‘*40
```

Determining Sentiments

- The output that follows shows that the naïve Bayes classifier is trained on 1,600 instances and tested on 400 instances from the movie corpus. The classifier achieves an accuracy of 73.5%.
- Most information features for positive reviews from the corpus include words such as outstanding, vulnerable, and astounding; and words such as insulting, ludicrous, and uninvolving are the most informative features for negative reviews.
- At the end, the output also shows the confusion matrix corresponding to the classifier to further evaluate the performance.

Determining Sentiments

Output

Train on 1600 instances

Test on 400 instances

Accuracy: 0.735

Most Informative Features

outstanding = True pos : neg = 13.9 : 1.0

insulting = True neg : pos = 13.7 :
1.0vulnerable = True pos : neg = 13.0 : 1.0

ludicrous = True neg : pos = 12.6 : 1.0

uninvolving = True neg : pos = 12.3 : 1.0

astounding = True pos : neg = 11.7 : 1.0

avoids = True pos : neg = 11.7 : 1.0

fascination = True pos : neg = 11.0 : 1.0

animators = True pos : neg = 10.3 : 1.0

symbol = True pos : neg = 10.3 : 1.0

Confusion matrix:

Predicted class

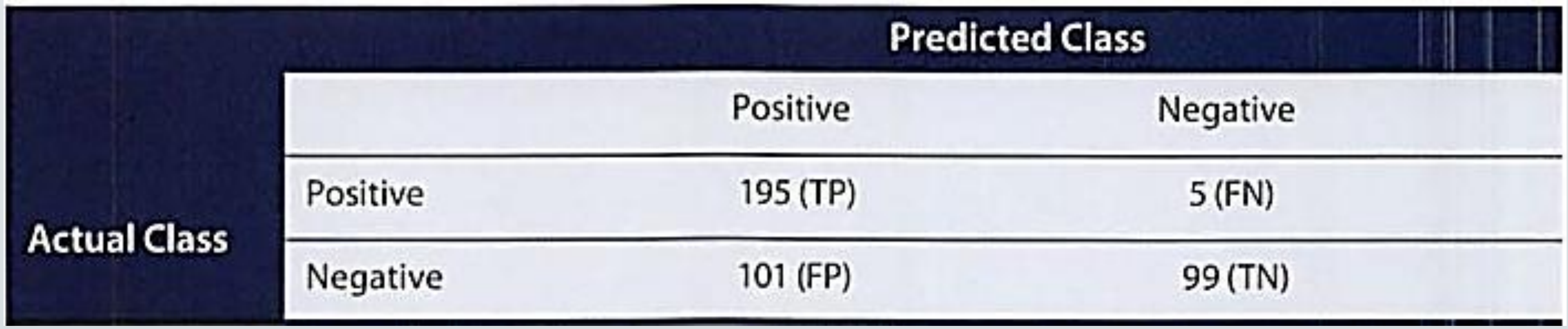
```
_________________

| 195 (TP) | 5 (FN) | Actual class
_________________

| 101 (FP) | 99 (TN) |
_________________
```

Determining Sentiments

- A confusion matrix is a specific table layout that allows visualization of the performance of a model over the testing set.
- Every row and column corresponds to a possible class in the dataset.
- Each cell in the matrix shows the number of test examples for which the actual class is the row and the predicted class is the column.
- Good results correspond to large numbers down the main diagonal (TP and TN) and small, ideally zero, off-diagonal elements (FP and FN)



		Predicted Class	
		Positive	Negative
Actual Class	Positive	195 (TP)	5 (FN)
	Negative	101 (FP)	99 (TN)

Determining Sentiments

$$Precision = \frac{TP}{TP + FP}$$

$$Recall = \frac{TP}{TP + FN}$$

- Precision and recall are two measures commonly used to evaluate tasks related to text analysis.
- Definitions of precision and recall are given as
- Precision is defined as the percentage of documents in the results that are relevant. If by entering keyword bPhone, the search engine returns 100 documents, and 70 of them are relevant, the precision of the search engine result is 0.7%.
- Recall is the percentage of returned documents among all relevant documents in the corpus. If by entering keyword bPhone, the search engine returns 100 documents, only 70 of which are relevant while failing to return 10 additional, relevant documents, the recall is $70/(70+10) = 0.875.$

Determining Sentiments

- Therefore, the naïve Bayes classifier receives a
- recall of $195/(195+5)=0.975$
- and a precision of .$195/(195+101)\approx 0.659.$
- A good classifier ideally should achieve both precision and recall close to 1.0.
- In information retrieval, a perfect precision score of 1.0 means that every result retrieved by a search was relevant (but says nothing about whether all relevant documents were retrieved), whereas a perfect recall score of 1.0 means that all relevant documents were retrieved by the search (but says nothing about how many irrelevant documents were also retrieved).
- Both precision and recall are therefore based on an understanding and measure of relevance.

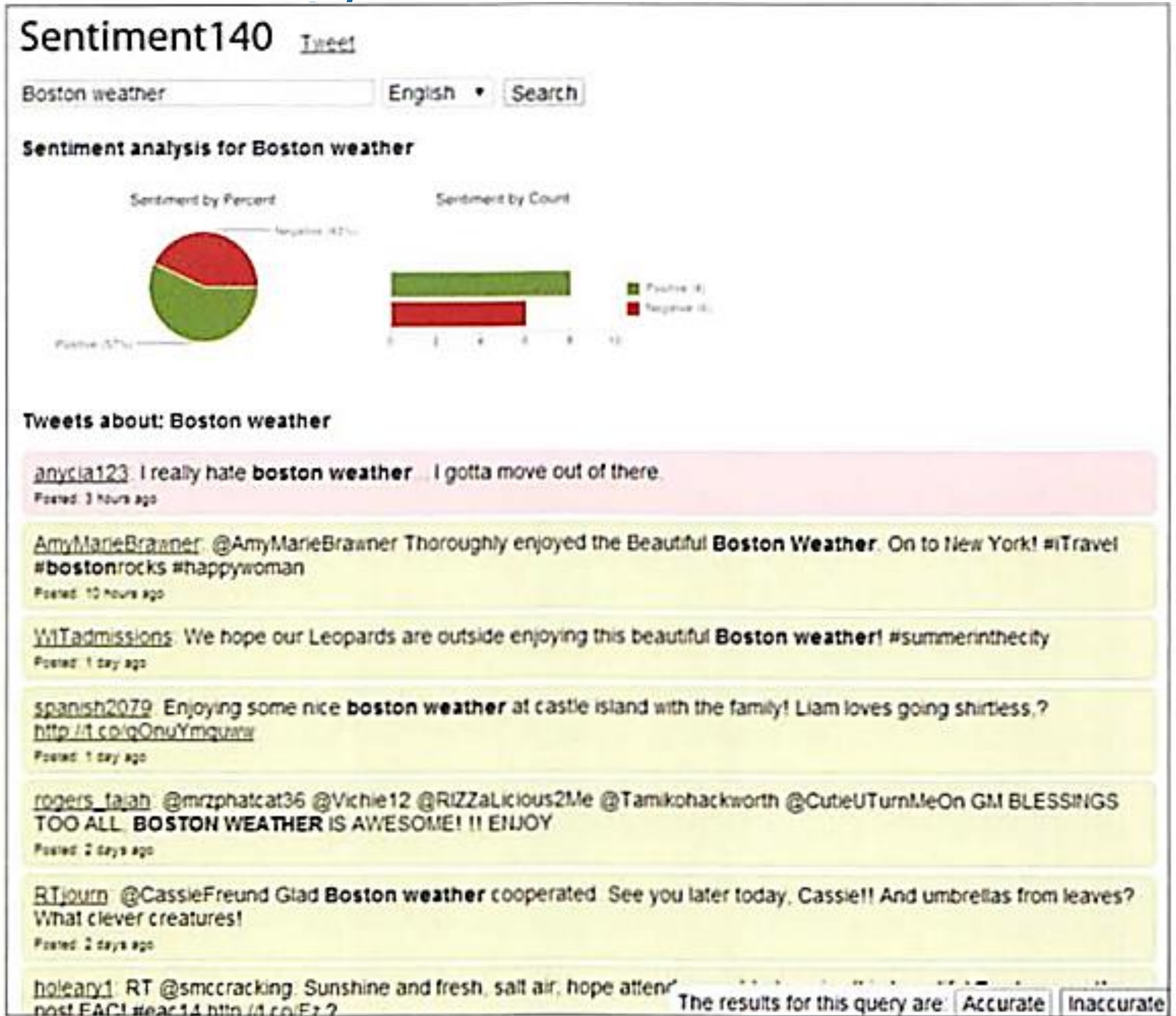
Determining Sentiments

- Classifiers determine sentiments solely based on the datasets on which they are trained.
- The domain of the datasets and the characteristics of the features determine what the knowledge classifiers can learn.
- For example, ***lightweight*** is a positive feature for reviews on laptops but not necessarily for reviews on wheelbarrows or textbooks.
- In addition, the training and the testing sets should share similar traits for classifiers to perform well.
- For example, classifiers trained on movie reviews generally should not be tested on tweets or blog comments.

Determining Sentiments

- For sentiment analysis based on larger amounts of streaming data such as millions or billions of tweets, it is less feasible to collect and construct datasets of tweets that are big enough or manually tag each of the tweets to train and test one or more classifiers. There are two popular ways to cope with this problem.
- The **first way to construct pretagged data,** as illustrated in recent work by Go et al. and Pak and Paroubek , **is to apply supervision and use emoticons such as :) and :(to indicate if a tweet contains positive or negative sentiments**.
- Words from these tweets can in turn be used as clues to classify the sentiments of future tweets. Go et al. use classification methods including **naïve Bayes, MaxEnt, and SVM** over the training and testing datasets to perform sentiment classifications.
- Their demo is available at **http://www.sentiment140.com**.

Determining Sentiments



Sentiment140 Tweet

Boston weather | English ▾ | Search

Sentiment analysis for Boston weather

Tweets about: Boston weather

anycia123: I really hate **boston weather** .. I gotta move out of there.
Posted: 3 hours ago

AmyMarieBrawner: @AmyMarieBrawner Thoroughly enjoyed the Beautiful **Boston Weather**. On to New York! #iTravel #**boston**rocks #happywoman
Posted: 10 hours ago

WITadmissions: We hope our Leopards are outside enjoying this beautiful **Boston weather!** #summerinthecity
Posted: 1 day ago

spanish2079: Enjoying some nice **boston weather** at castle island with the family! Liam loves going shirtless,? http://t.co/qOnuYmquww
Posted: 1 day ago

rogers_tajah: @mrzphatcat36 @Vichie12 @RIZZaLicious2Me @Tamikohackworth @CutieUTurnMeOn GM BLESSINGS TOO ALL. **BOSTON WEATHER** IS AWESOME! !! ENJOY
Posted: 2 days ago

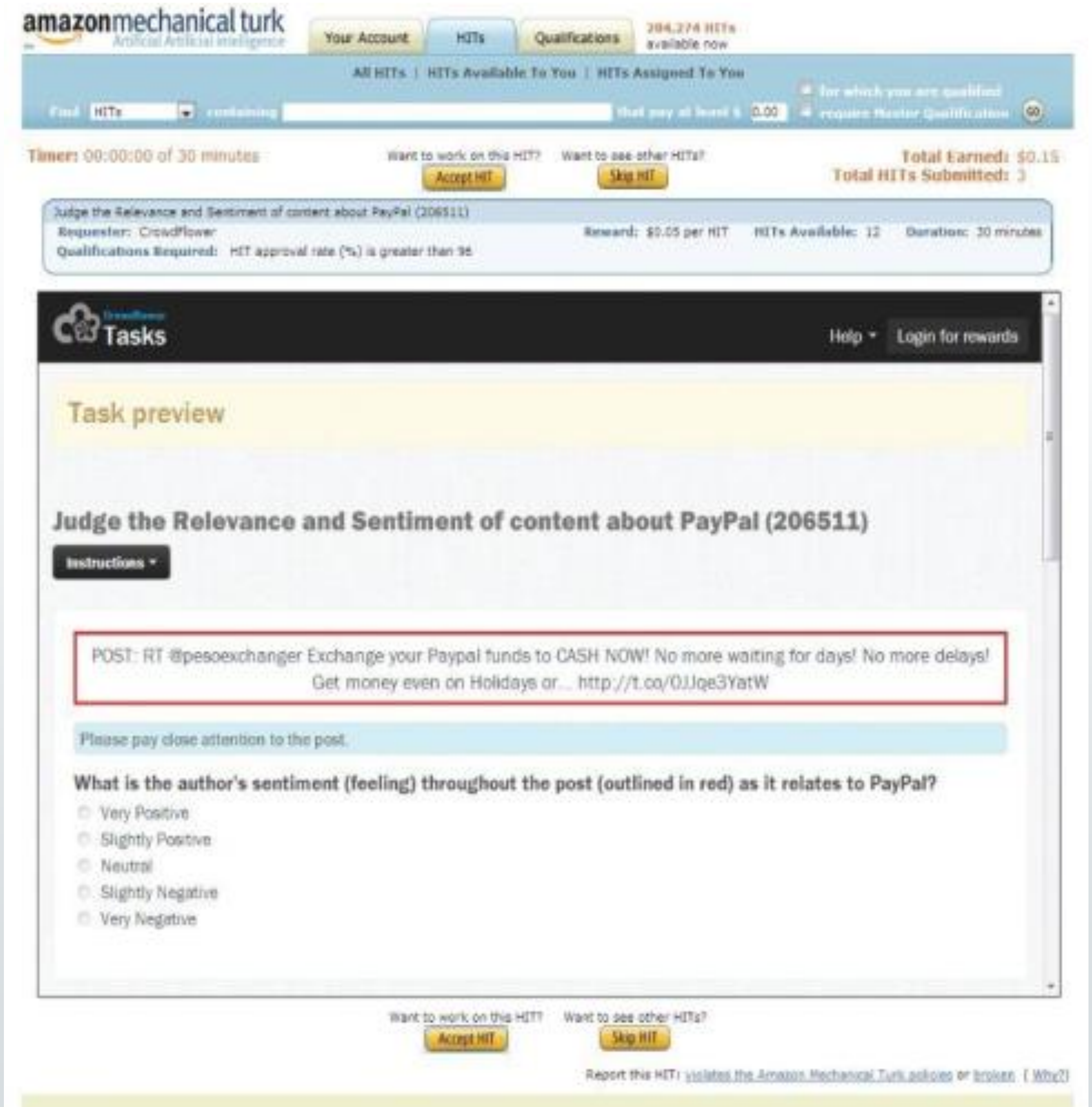
RTjourn: @CassieFreund Glad **Boston weather** cooperated. See you later today, Cassie!! And umbrellas from leaves? What clever creatures!
Posted: 2 days ago

holeary1: RT @smccracking: Sunshine and fresh, salt air, hope atten
post FAC! #eac14 http://t.co/Ez ?

The results for this query are: Accurate | Inaccurate

Determining Sentiments

- Second way , related research usually uses Amazon Mechanical Turk (MTurk) to collect human-tagged reviews.
- MTurk is a crowdsourcing Internet marketplace that enables individuals or businesses to coordinate the use of human intelligence to perform tasks that are difficult for computers to do.
- In many cases, MTurk has been shown to collect human input much faster compared to traditional channels such as door-to-door surveys.
- For the example sentiment analysis task, the Data Science team can publish the tweets collected to MTurk as Human Intelligence Tasks (HITs).


amazonmechanical turk
Artificial Artificial Intelligence

Your Account | HITs | Qualifications

204,274 HITs available now

All HITs | HITs Available To You | HITs Assigned To You

Find HITs containing that pay at least $ 0.00 for which you are qualified require Master Qualification GO

Timer: 00:00:00 of 30 minutes

Want to work on this HIT? Accept HIT

Want to see other HITs? Skip HIT

Total Earned: $0.15
Total HITs Submitted: 3

Judge the Relevance and Sentiment of content about PayPal (206511)
Requester: CrowdFlower
Reward: $0.05 per HIT
HITs Available: 12
Duration: 30 minutes
Qualifications Required: HIT approval rate (%) is greater than 98

CrowdFlower Tasks
Help
Login for rewards

Task preview

Judge the Relevance and Sentiment of content about PayPal (206511)

Instructions

POST: RT @pesoexchanger Exchange your Paypal funds to CASH NOW! No more waiting for days! No more delays! Get money even on Holidays or... http://t.co/OJJqe3YatW

Please pay close attention to the post.

What is the author's sentiment (feeling) throughout the post (outlined in red) as it relates to PayPal?

- Very Positive
- Slightly Positive
- Neutral
- Slightly Negative
- Very Negative

Want to work on this HIT? Accept HIT

Want to see other HITs? Skip HIT

Report this HIT: violates the Amazon Mechanical Turk policies or broken (Why?)

Gaining Insights

- The 300 reviews are visualized as a word cloud after removing stop words. A word cloud (or tag cloud) is a visual representation of textual data.
- Tags are generally single words, and the importance of each word is shown with font size or color.

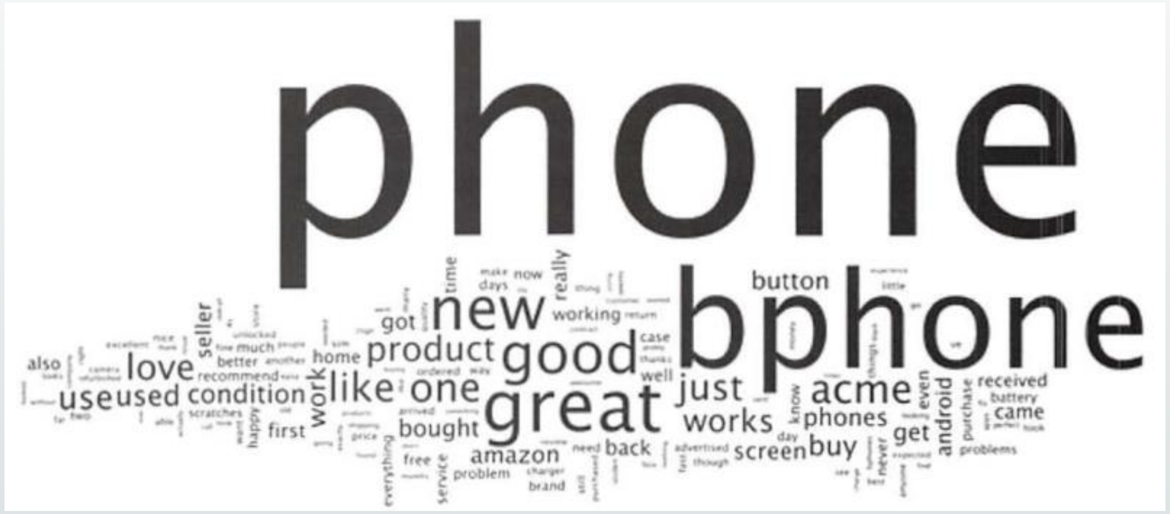
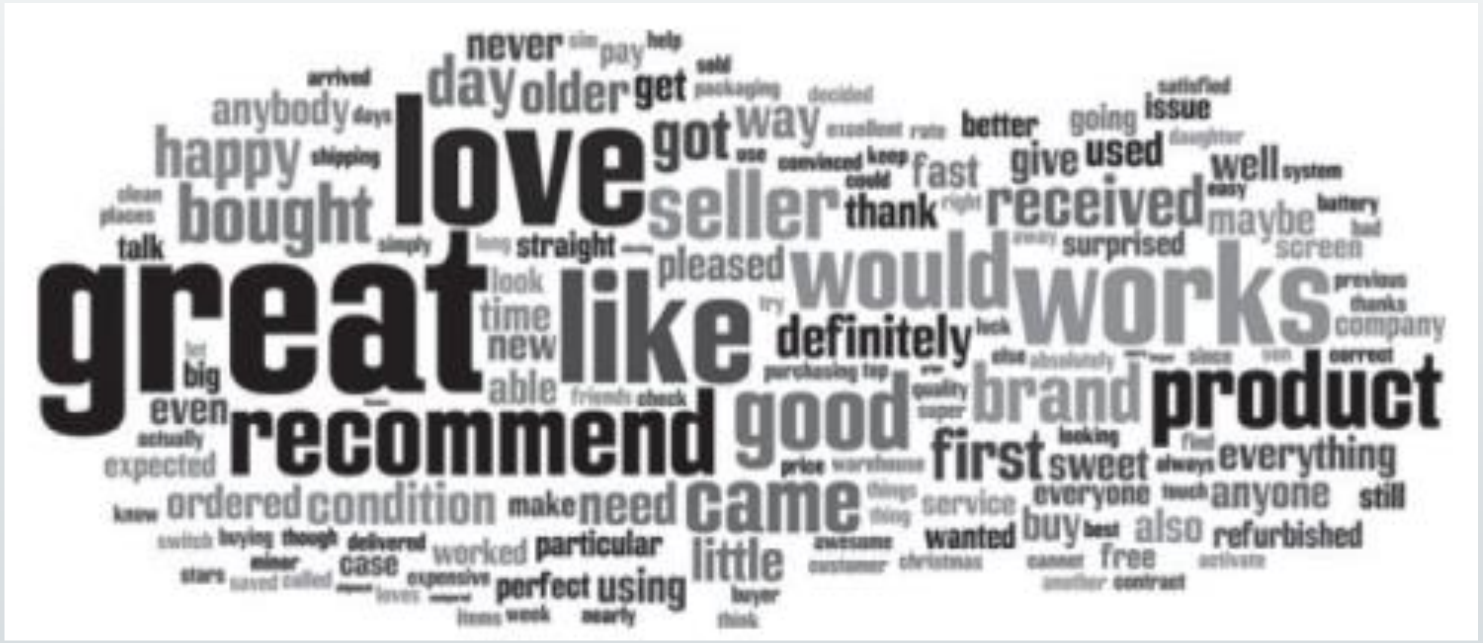


Figure shows the word cloud built from the 300 reviews. Most of the graph is taken up by the words phone and bphone, which occur frequently but are not very informative. Overall, the graph reveals little information.

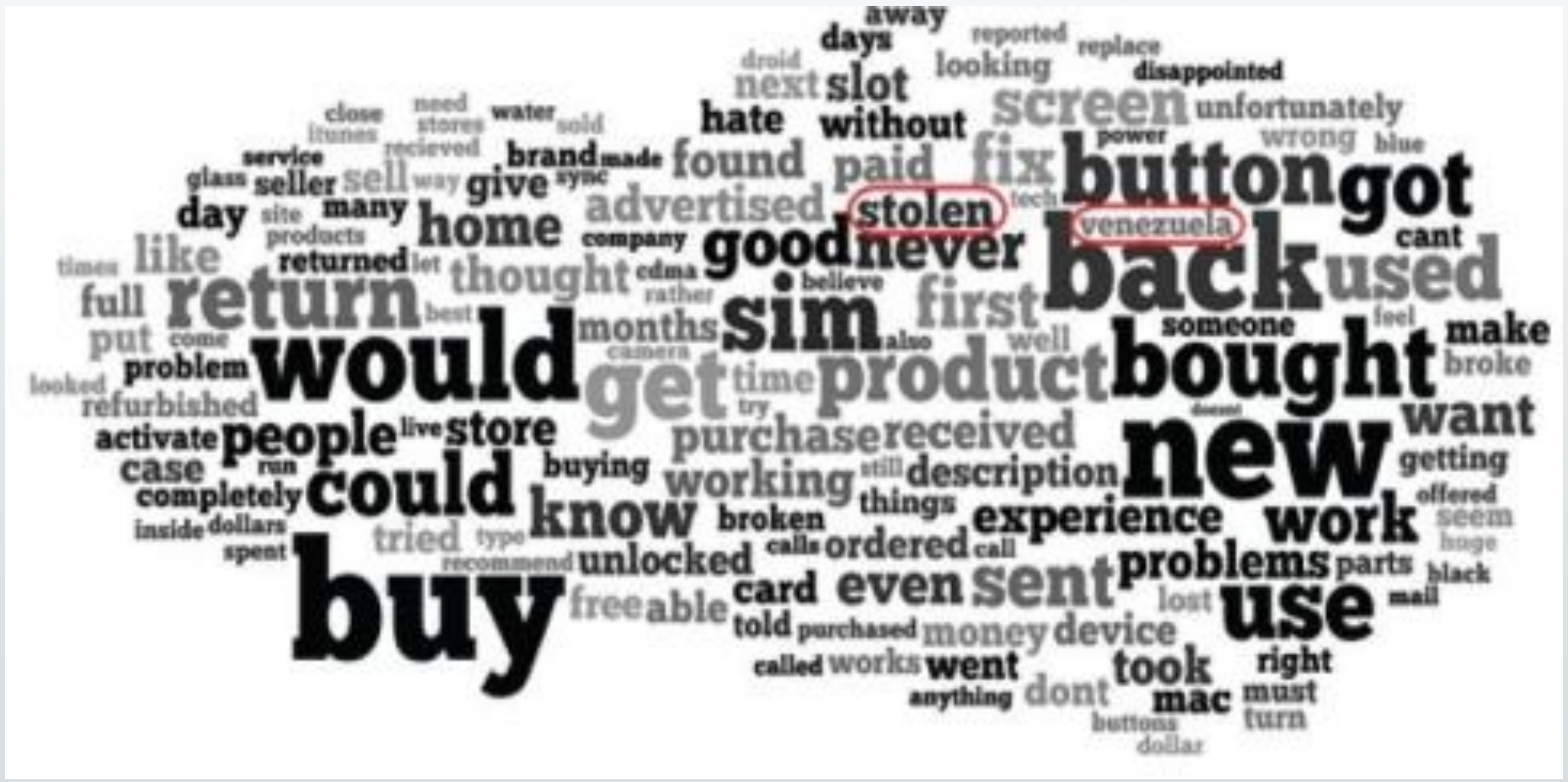
Gaining Insights

- To reveal more information, the team can remove words such as phone, bPhone, and ACME, which are not very useful for the study. Related research often refers to these words as domain-specific stop words. And take data of 5 star rating reviews.
- Figure shows the word cloud corresponding to 50 five star reviews extracted from the data.



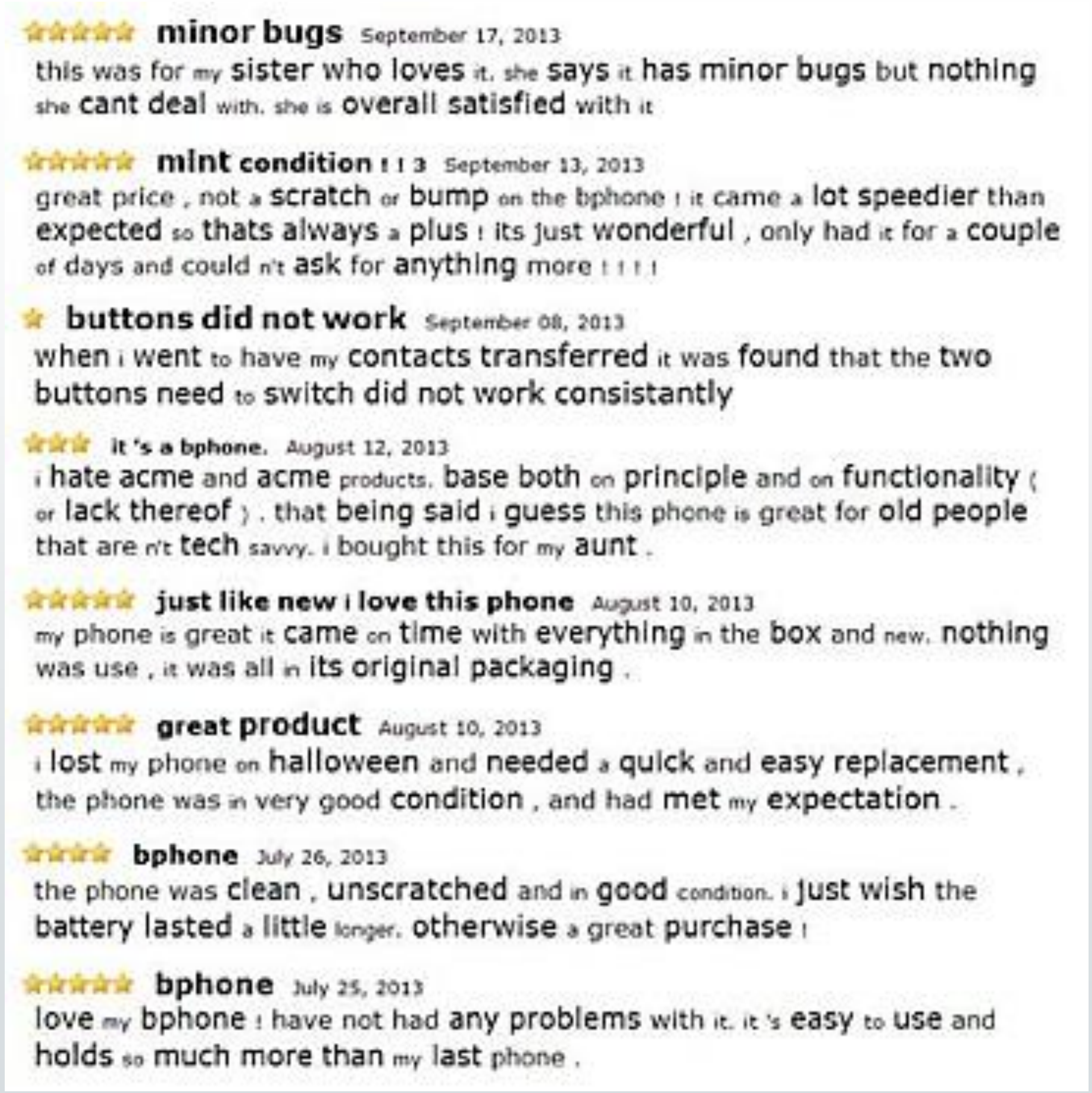
Gaining Insights

- Figure shows the word cloud of 70 one-star reviews. The words sim and button occur frequently enough that it would be advisable to sample the reviews that contain these terms and determine what is being said about buttons and SIM cards.



Gaining Insights

- TFIDF can be used to highlight the informative words in the reviews.
- Figure shows a subset of the reviews in which each word with a larger font size corresponds to a higher TFIDF value. Each review is considered a document. With TFIDF, data analysts can quickly go through the reviews and identify what aspects are perceived to make bPhone a good product or a bad product



★★★★★ **minor bugs** September 17, 2013
this was for my sister who loves it. she says it has minor bugs but nothing she cant deal with. she is overall satisfied with it

★★★★★ **mint condition ! ! 3** September 13, 2013
great price , not a scratch or bump on the bphone ! it came a lot speedier than expected so thats always a plus ! its just wonderful , only had it for a couple of days and could n't ask for anything more ! ! ! !

★ **buttons did not work** September 08, 2013
when i went to have my contacts transferred it was found that the two buttons need to switch did not work consistantly

★★★ **it 's a bphone.** August 12, 2013
i hate acme and acme products. base both on principle and on functionality (or lack thereof) . that being said i guess this phone is great for old people that are n't tech savvy. i bought this for my aunt .

★★★★★ **just like new i love this phone** August 10, 2013
my phone is great it came on time with everything in the box and new. nothing was use , it was all in its original packaging .

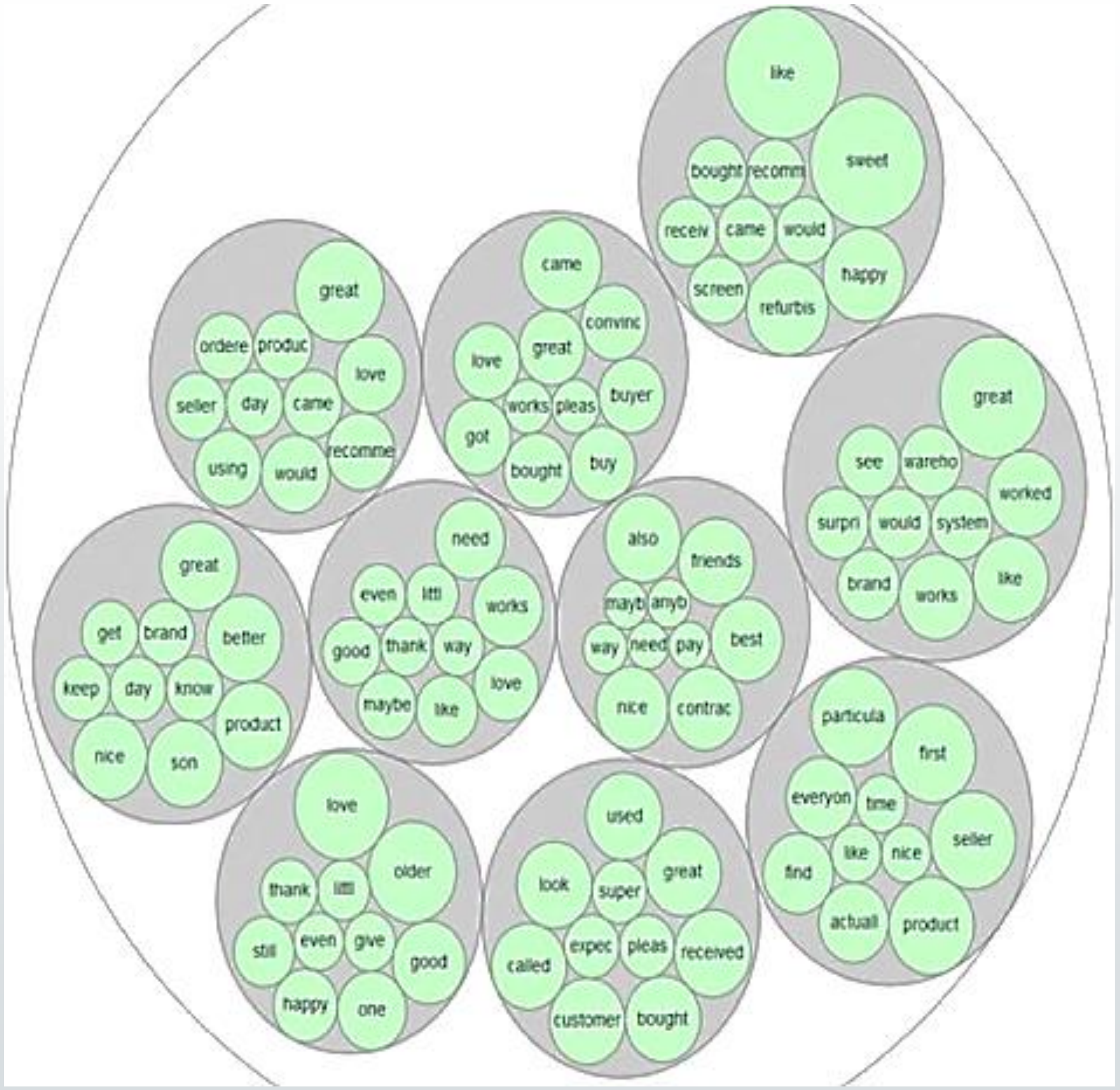
★★★★★ **great product** August 10, 2013
i lost my phone on halloween and needed a quick and easy replacement , the phone was in very good condition , and had met my expectation .

★★★★ **bphone** July 26, 2013
the phone was clean , unscratched and in good condition. i just wish the battery lasted a little longer. otherwise a great purchase !

★★★★★ **bphone** July 25, 2013
love my bphone ! have not had any problems with it. it 's easy to use and holds so much more than my last phone .

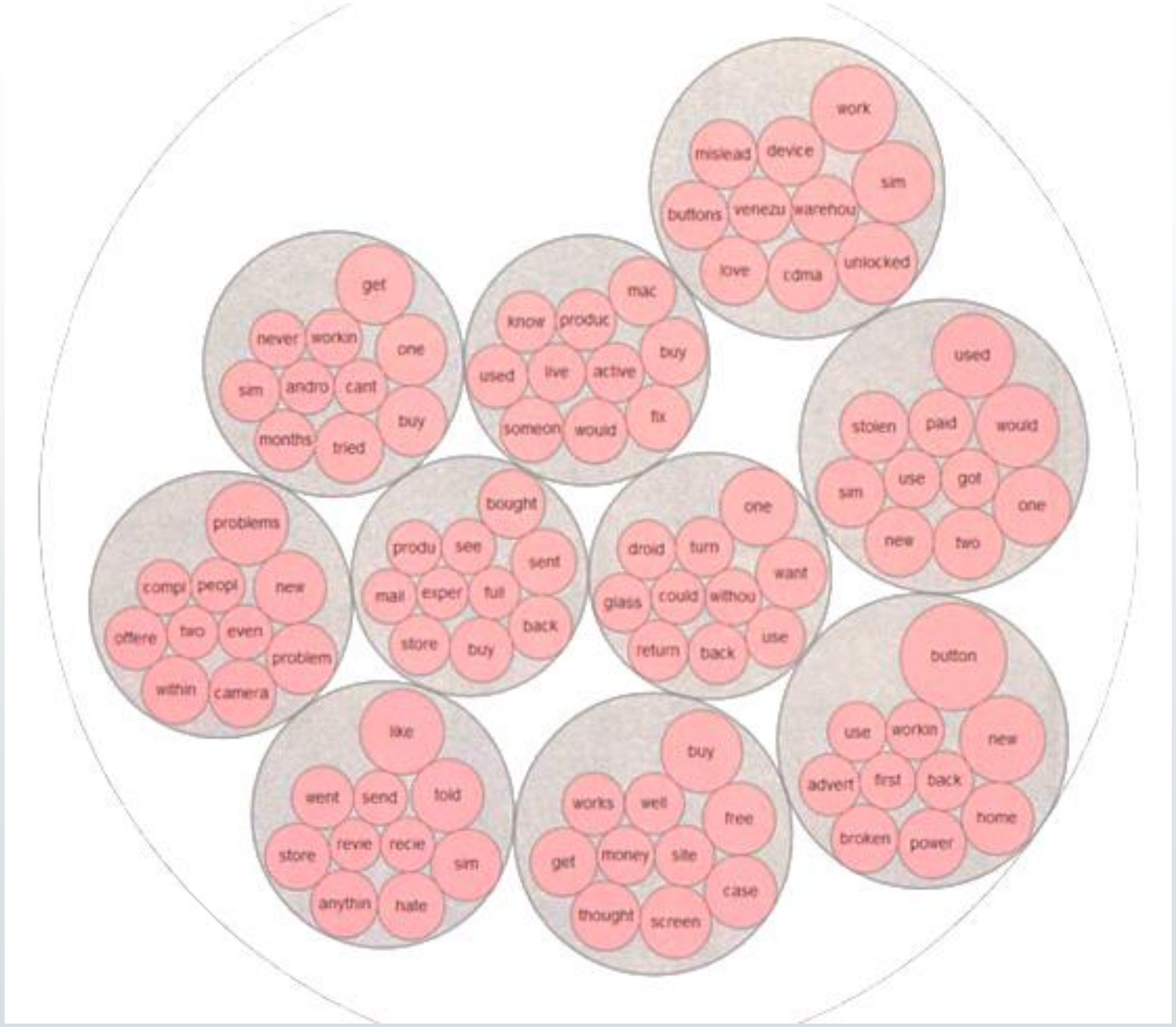
Gaining Insights

- Topic models such as LDA can categorize the reviews into topics. Figures 1 and 2 show circular graphs of topics as results of the LDA.
- These figures are produced with tools and technologies such as Python, NoSQL, and D3.js. Figure 1 visualizes ten topics built from the five-star reviews. Each topic focuses on a different aspect that can characterize the reviews.
- The disc size represents the weight of a word. In an interactive environment, hovering the mouse over a topic displays the full words and their corresponding weights.



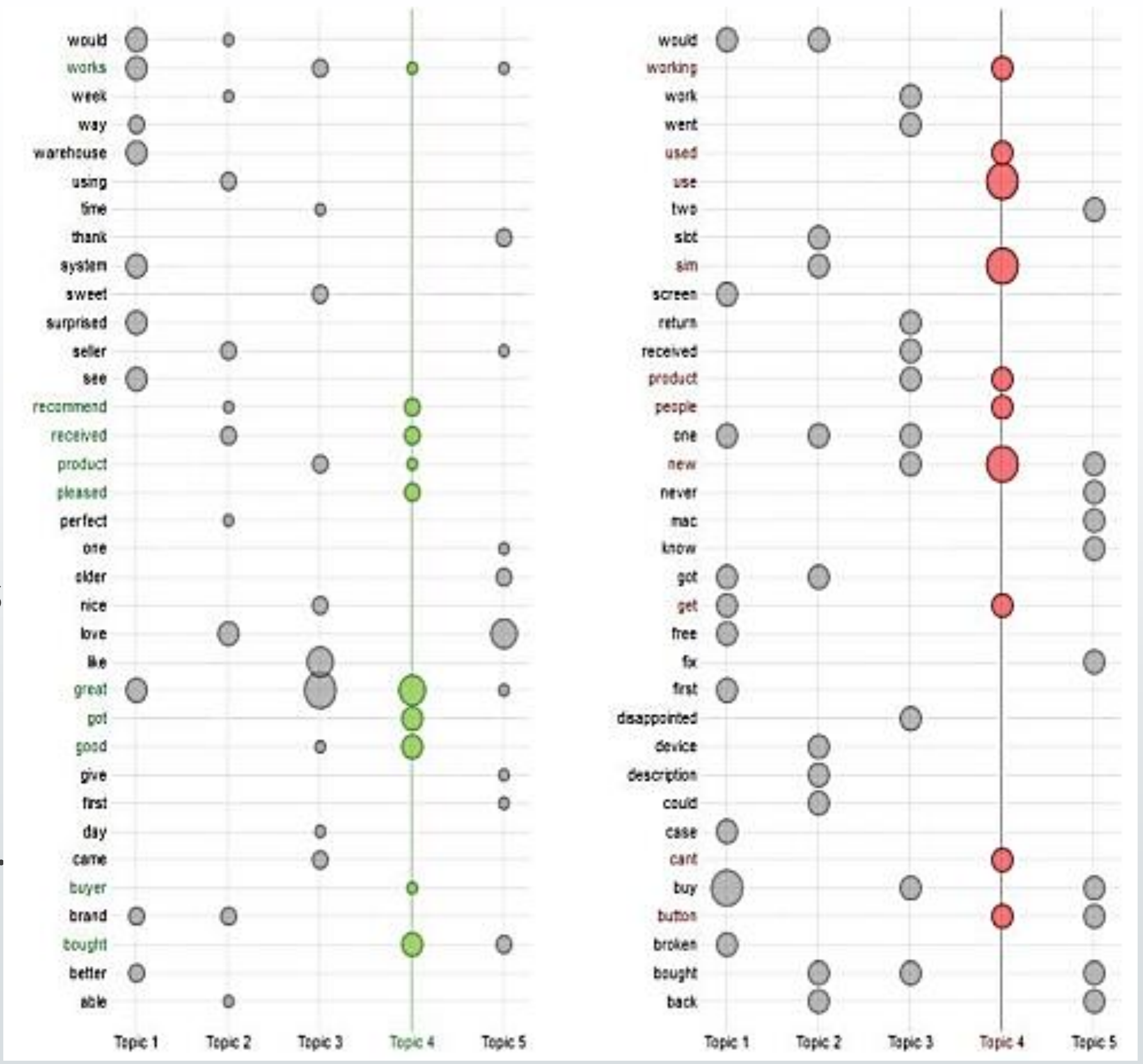
Gaining Insights

- Figure 2 visualizes ten topics from one-star reviews.
- For example, the bottom-right topic contains words such as button, power, and broken, which may indicate that bPhone has problems related to button and power supply.
- The Data Science team can track down these reviews and find out if that’s really the case.



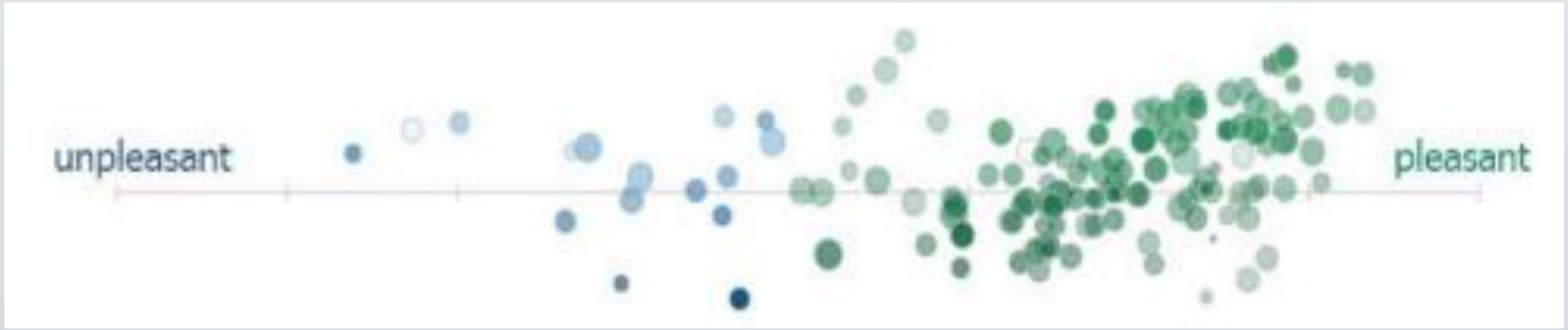
Gaining Insights

- Figure provides a different way to visualize the topics. Five topics are extracted from five-star reviews and one-star reviews, respectively.
- In an interactive environment, hovering the mouse on a topic highlights the corresponding words in this topic.
- The screenshots in Figure were taken when Topic 4 is highlighted for both groups.
- The weight of a word in a topic is indicated by the disc size.



Gaining Insights

- Sentiment analysis over 100 tweets from the popular microblogging site Twitter.
- The result is shown in Figure, the left side represents negative sentiments, and the right side represents positive sentiments.
- Vertically, the tweets have been randomly placed for aesthetic purposes. Each tweet is shown as a disc, where the size represents the number of followers of the user who made the original tweet. The color shade of a disc represents how frequently this tweet has been retweeted.
- The figure indicates that most customers are satisfied with ACME's bPhone.


unpleasant
pleasant