A Comprehensive Academic Guide to Text Mining

Sajedeh Talebi

University of Alzahra at Tehran, Iran

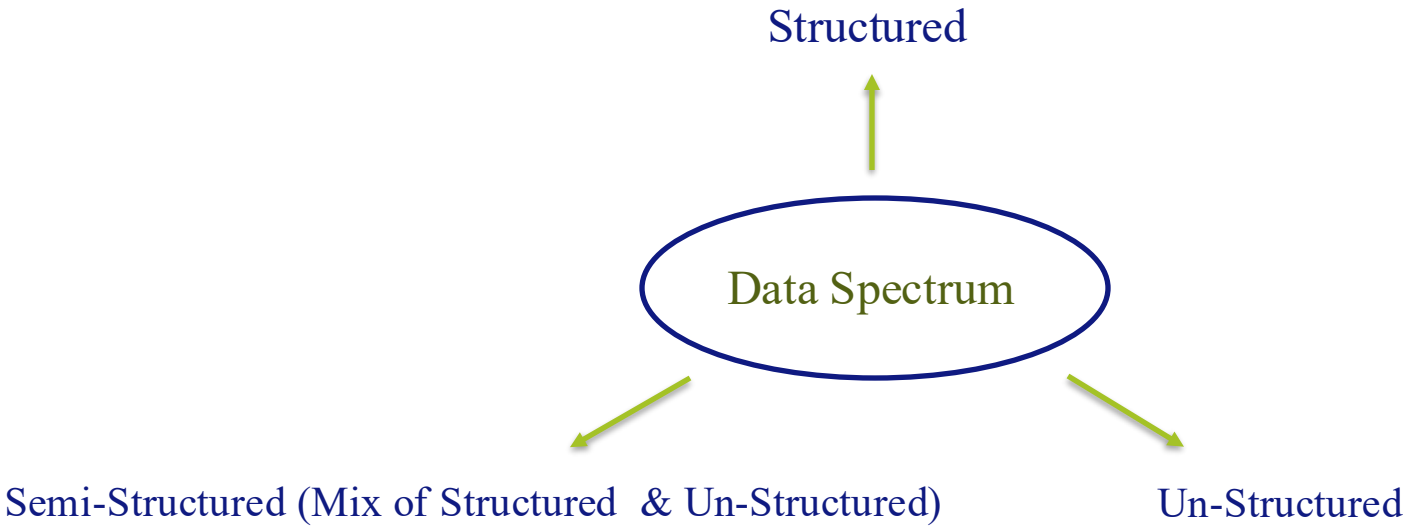
In the Spring 2024 course

Understanding Data Types

➢ What is Data?

Data refers to the collection of raw facts, figures, or observations that can be analyzed to generate valuable insights. It is the foundation for generating information and knowledge, which can be further transformed into actionable intelligence to support decision-making processes across various domains.

- Data can be categorized into three primary types based on structure:



Structured Data

➢ Structured Data

Structured data is organized information that adheres to a predefined schema, enabling easy processing and semantic consistency. It is often domain-specific and can be integrated with other data types. Real-time applications frequently utilize structured data.

➢ Examples of Structured Data

- Product Information

Attributes: Fixed fields including product name, manufacturer, release date, product category, description, and user manual.
Examples: Smartphones with attributes like brand, model, storage capacity, price, and release date.

- News Articles

Attributes: Fixed fields including article title, author, date published, news category, summary, and full article text.
Examples: Sports articles with attributes like title, journalist, publication date, sports category, and main story content.

Un-Structured Data

➢ Un-Structured Data

Unstructured data is information that does not follow a predefined model or schema, making it more complex to analyze and process. This data type comprises various formats, such as text documents, images, videos, and audio files.

➢ Examples of Un-Structured Data

- Social Media Posts

Content: Text, emojis, images, and videos in a non-fixed structure.
Platforms: Twitter, Facebook, Instagram, and other social media platforms.

- Customer Reviews

Content: Text-based opinions about products or services with varying lengths, formats, and grammar.
Platforms: E-commerce websites like Amazon, review sites like Yelp, or app store reviews.

- Surveillance Footage

Content: Video files from security cameras in various resolutions and formats.
Systems: Home security systems, public surveillance cameras, or traffic monitoring systems.

Semi-Structured Data

➢ Semi-Structured Data

Semi-structured data is a combination of structured and unstructured data, exhibiting some level of organization while lacking a rigid structure. This type of data often includes metadata tags or markers to help organize and categorize information. Semi-structured data can be stored in relational databases, NoSQL databases, or other data management systems designed to handle both structured and unstructured data.

➢ Examples of Semi-Structured Data

- Email Messages

Content: Text content, metadata tags for sender, recipient, subject, and date, as well as attachments in various formats.
Platforms: Email clients like Gmail, Outlook, or Apple Mail.

- Web Pages

Content: HTML tags to structure text, images, and other multimedia elements with varying layouts.
Platforms: Websites like news sites, blogs, or e-commerce platforms.

- XML and JSON Files

Content: Nested elements with varying attributes, often used in web services and APIs.
Applications: Data exchange between systems, web APIs, or configuration files.

A Quick Overview of Data Structure

- A Quick Overview of Data Structure: Unstructured, Semi-Structured, and Structured Data

- **Title: Employee Information**

- **Structured Section:**

Name: John Doe
Age: 35
Department: Engineering
Salary: $80,000

- **Semi-Structured Section:**

Project 1: Developing a new software application.
Project 2: Implementing a machine learning model for data analysis.
Project 3: Researching renewable energy technologies.

- **Unstructured Section:**

John Doe is an experienced engineer with a background in software development and data analysis. He has been with the company for 5 years and has contributed significantly to various projects. In his spare time, John enjoys hiking and playing guitar.

Text Mining: Transforming Unstructured Text into Actionable Insights

- Text Mining:

Text mining is a subfield of data mining that extracts insights from unstructured textual data using NLP, ML, and statistical analysis. It identifies patterns and trends, transforming unstructured text into structured data for better decision-making. Applications include user experience enhancement, healthcare support, legal automation, and cybersecurity threat detection, making it essential for organizations in today's digital landscape.

Text Mining: Uncovering Hidden Insights and Shaping Our World

- "Unveiling the Impact: Text Mining's Role in Our Lives”

- Personalized User Experiences
- Healthcare Decision Support
- Smart Assistants and Chatbots
- News and Media Curation
- Social Media Insights
- Educational Assistance and E-Learning:
- Legal and Compliance Automation
- Environmental Monitoring
- Cybersecurity Threat Detection

Text Mining: Uncovering Hidden Insights and Shaping Our World

➢ "Unveiling the Impact: Text Mining's Role in Our Lives"

- **Personalized User Experiences:** Text mining analyzes user preferences and behavior to create tailored content, recommendations, and interfaces, enhancing engagement and satisfaction.
- **Healthcare Decision Support:** Extracting insights from medical documents, research papers, and patient data to assist in diagnosis, treatment planning, and improving healthcare outcomes.
- **Smart Assistants and Chatbots:** Utilizing text mining to understand user intent and generate appropriate responses in conversational AI applications.
- **News and Media Curation:** Identifying relevant articles and topics from vast amounts of news data to provide personalized news feeds and content recommendations.
- **Social Media Insights:** Analyzing social media posts and conversations to uncover trends, sentiments, and user behaviors for better marketing strategies and customer understanding.

Text Mining: Uncovering Hidden Insights and Shaping Our World

➢ "Unveiling the Impact: Text Mining's Role in Our Lives"

- **Educational Assistance and E-Learning:** Text mining helps personalize learning materials, assess student performance, and provide feedback for improved educational outcomes.
- **Legal and Compliance Automation:** Identifying and extracting relevant information from legal documents to streamline compliance processes and improve decision-making.
- **Environmental Monitoring:** Analyzing text data from various sources to track environmental changes, monitor pollution levels, and assess the impact of climate change.
- **Cybersecurity Threat Detection:** Detecting and preventing cyber threats by analyzing text data from emails, chat logs, and other communication channels to identify phishing attempts, malware distribution, and other malicious activities.

Exploring Text Mining: Applications

➢ Text Mining's Applications (List as Subcategories!)

- Information Retrieval
- Market Intelligence
- Text Classification
- Sentiment Analysis
- Named Entity Recognition (NER)
- Topic Modeling
- Text Summarization

PyCharm

Google Colab

VSCode

Jupyter Notebook

Python IDEs (Integrated Development Environment) To Implementing Text Mining Steps.

Text Mining Application: Information Retrieval

➢ Information Retrieval

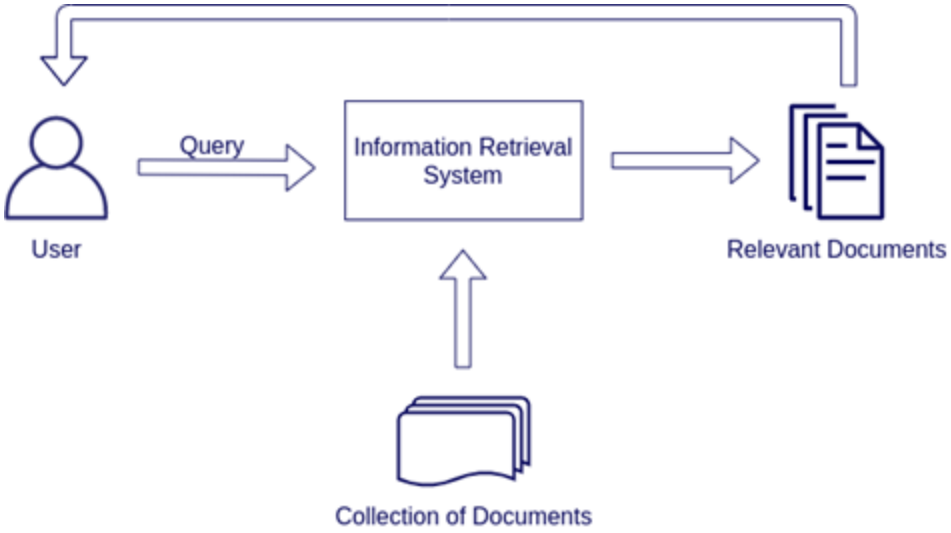
Information retrieval (IR) in text mining involves the process of finding and presenting relevant documents or passages from a large collection of text data in response to a user query. It's essentially about searching for information within textual sources.

➢ Information retrieval in text mining involves:

- Indexing: Creating a map of words to documents for efficient searching.
- Querying: Users input queries, which are matched against the index.
- Ranking: Results are prioritized based on relevance to the query.
- Presentation: Relevant documents or passages are presented to users, often with additional metadata.

Text Mining Application: Information Retrieval

- Information Retrieval



- e.g.

In a library catalog system, when a user searches for books on a specific topic, such as "artificial intelligence," the system retrieves relevant books from the library's collection based on the entered query.

- Application:

In web search engines like Google, when users enter search queries, the search engine retrieves relevant web pages from its index based on the keywords entered by the user.

Text Mining Application: Market Intelligence

➢ Market Intelligence

Text mining aids market intelligence by analyzing trends, competitors, and consumer data for informed decisions. It uncovers insights from unstructured text, helping understand strategies, trends, sentiment, preferences, segmentation, and predictions. For instance, analyzing reviews provides actionable insights into customer satisfaction. Utilizing text mining enhances market responsiveness and competitiveness.

➢ Text mining applied in market intelligence as:

- Competitor Analysis
- Trend Identification
- Brand Monitoring
- Consumer Insights
- Market Segmentation
- Predictive Analytics

Text Mining Application: Market Intelligence

➢ Text mining applied in market intelligence as:

- **Competitor Analysis:** Extracting insights from competitor data to understand strategies.

e.g. Analyzing reviews reveals strengths and weaknesses.

- **Trend Identification:** Scanning news and social media for emerging trends.

e.g. Monitoring social media identifies growing interests.

- **Brand Monitoring:** Tracking brand mentions to gauge sentiment.

e.g. Sentiment analysis on Twitter evaluates campaign impact.

- **Consumer Insights:** Analyzing feedback to uncover preferences.

e.g. Extracting insights from reviews guides product development.

- **Market Segmentation:** Segmenting customers based on demographics.

e.g. Clustering social media posts targets specific segments.

- **Predictive Analytics:** Building models to forecast behaviors.

e.g. Analyzing service data predicts churn rates.

Text Mining Application: Text Classification

➢ Text Classification:

Text classification, a fundamental task in text mining, involves categorizing text documents into predefined categories or classes based on their content. This process is also known as text categorization or document classification.

➢ Key Components of Text Classification:

- Text Preprocessing
- Feature Extraction
- Model Training
- Model Evaluation
- Prediction

Text Mining: Text Pre-Processing

➢ Text Pre-Processing:

When conducting data preprocessing for data mining tasks, the following sequence of techniques can be applied to enhance model performance:

English Language

- Spell-Checking and Correction
- Special Character Handling
- Normalization
- Part-of-Speech Tagging
- Stop-word Removal
- Lemmatization
- Tokenization
- Augmentation

Persian Language

- Special Character Handling
- Spell-Checking
- Normalization
- Part-of-Speech Tagging
- Stop-word Removal
- Lemmatization
- Tokenization

- highlighted the enhanced reliability of lemmatization over stemming techniques, making it a preferred choice for text analysis (Chopra et al., 2016).

Text Mining: Text Pre-Processing

- Text Pre-Processing

English Language Analysis

- **NLTK (Natural Language Toolkit)**: The most widely used library provides tools for tokenization/ stemming/ lemmatization/ part-of-speech tagging/ parsing, and more.
- **SpaCy**: offers pre-trained models/ tokenization/ part-of-speech tagging/ named entity recognition/ dependency parsing/ and sentence segmentation.
- **Gensim**: Used for topic modelling/ document similarity analysis Latent Semantic Analysis (LSA)/ Latent Dirichlet Allocation (LDA)/ and word embedding.
- **Transformers (Hugging Face)**: Used for text classification/ language generation/ and question-answering
- **TextBlob**: provides APIs for sentiment analysis/ part-of-speech tagging/ and noun phrase extraction.
- **Stanford NLP**: offers Pre-trained Models.

Text Mining: Text Pre-Processing

- Spell Checking and Correction (Text Quality Enhancement)

- **Original Text:** "I hav a problm with my computr. It's runnng very sloww and I can't acsess the internet. Plase help me fix it as soon as possibl".

- **Corrected Text:** "I have a problem with my computer. It's running very slow and I can't access the internet. Please help me fix it as soon as possible".

- **Original Text**: "I recived a brokn laptop from youre store. The scren is dmgd and its not trning onn. I ned an immdiate replacment."

- **Corrected Text**: "I received a broken laptop from your store. The screen is damaged, and it's not turning on. I need an immediate replacement."

PySpellChecker, Autocorrect, TextBlob and Enchant Python Libraries Utilize for Spell Checking

Text Mining: Text Pre-Processing

- Handling Special Characters (Preserving Textual Integrity)

- **Removing Special Characters and Emojis:**
- Before: "I received a text message with lots of emojis! 😊🎉📱"
- After: "I received a text message with lots of emojis"

- **URL Removal:**
- Before: "Check out our latest blog post: https://example.com/blog-post"
- After: " Check out our latest blog post"

- **Encoding/Decoding:**
- Before: "The café serves delicious coffee."
- After: "The cafe serves delicious coffee."

- **Filtering:**
- Before: "Please remove all symbols: @#$%^&*()_+"
- After: "Please remove all symbols"

re and Beautifulsoap Python Libraries Utilize for Handling Special Character

Text Mining: Text Pre-Processing

➢ Text Normalization

➢ **Case Normalization:**
- Before: "Education is the key to success"
- After: " education is the key to success"

➢ **Accent and Diacritic Removal:**
- Before: " The café is near the Eiffel Tower"
- After: " The cafe is near the Eiffel Tower"

➢ **Expanding Contractions:**
- Before: "I'm going to the party, but she isn't."
- After: " I am going to the party, but she is not."

➢ **Normalizing Numbers and Units:**
- Before: " The distance is approx. 5km "
- After: " The distance is approximately 5 kilometres "

re and Beautifulsoap Python Libraries Utilize for Text Normalizaing

Text Mining: Text Pre-Processing

- Part of Speech (POS Tags and Grammatical Categories)

- **Noun (NOUN):** "dog", "cat", "house", "book"
- **Verb (VERB):** "run", "eat", "sleep", "write"
- **Adjective (ADJ):** "happy", "big", "beautiful", "red"
- **Adverb (ADV):** "quickly", "very", "often", "well"
- **Pronoun (PRON):** "he", "she", "it", "they"
- **Preposition (ADP):** "in", "on", "at", "over"
- **Conjunction (CONJ):** "and", "but", "or", "so"
- **Determiner (DET):** "the", "a", "an", "this", "that"
- **Interjection (INTJ):** "Wow!", "Ouch!", "Oops!", "Hey!"

NLTK, spaCy and TextBlob Python Libraries Utilize for POS

Text Mining: Text Pre-Processing

- Custom Stop-Word Removal

- **Original Text:**
- "Machine learning algorithms, such as support vector machines and neural networks, have revolutionized various industries by enabling tasks like image recognition, natural language processing, and predictive analytics".

- **Text after Stop-word Removal using Custom Stop List:**
- "Machine learning algorithms, support vector machines neural networks, have enabling image recognition, natural language processing, predictive analytics".

- In this example, the words "revolutionized", "various", "industries", "tasks", and "like" have been removed based on the custom stop list, while the other words remain unaffected.

With custom stop word list: All words preserved for analysis

NLTK Python Library Utilize for Custom Word Removal

Text Mining: Text Pre-Processing

- Lemmatization

- Lemmatization reduces words to their base or root form, which helps in analyzing the meaning of text more effectively.

e.g.
Before: "The **universities** are conducting **researching** studies to improve educational **systems** and provide better opportunities for students "

After: " The **university** conduct **research** study to improve education **system** and provide good opportunity for **student** "

- Lemmatization reduces words to their base form for better text analysis:

- "universities" and "researching" -> "university" and "research"
- "studies" and "educational" -> "study" and "education"
- "systems" and "opportunities" -> "system" and "opportunity"
- "students" -> "student"

NLTK Python Library Utilize for Lemmatization

Text Mining: Text Pre-Processing

- Tokenization

- Tokenization (Word Level):

Text mining is the process of extracting knowledge from unstructured data.

['Text', 'mining', 'is', 'the', 'process', 'of', 'extracting', 'knowledge', 'from', 'unstructured', 'data']

- Tokenization (Sub-Word Level):

Unstructured data poses unique challenges in text mining.

['Un', 'struct', 'ured', 'data', 'poses', 'unique', 'challenges', 'in', 'text', 'mining']

- Tokenization (Character Level):

Data pre-processing is an essential step in text mining

['D', 'a', 't', 'a', ' ', 'p', 'r', 'e', 'p', 'r', 'o', 'c', 'e', 's', 's', 'i', 'n', 'g', ' ', 'i', 's', ' ', 'a', 'n', ' ', 'e', 's', 's', 'e', 'n', 't', 'i', 'a', 'l', ' ', 's', 't', 'e', 'p', ' ', 'i', 'n', ' ', 't', 'e', 'x', 't', ' ', 'm', 'i', 'n', 'i', 'n', 'g']

NLTK Python Library Utilize for Tokenization

Text Mining: Text Pre-Processing

- Text Augmentation Methods (Adding More!)

- **Back-Translation (Preserving Meaning):**
  - Original Sentence: "The cat sat on the mat".
  - Back-Translated Sentence: "Le chat s'est assis sur le tapis".
  - Translation to Target Language: "The cat sat on the carpet".
- **Random Word Replacement:**
  - Original Sentence: "The weather is nice today".
  - Sentence with Replacement: "The climate is pleasant today".
- **Masked Language Model (MLM) Pretraining:**
  - Original Sentence: "The quick brown fox jumps over the lazy dog".
  - Masked Sentence: "The [MASK] brown fox jumps over the lazy dog".
- **Style Transfer**
  - Original Sentence: "The movie was very entertaining".
  - Informal Style: "That movie was super fun!"
  - Formal Style: "The film provided a highly enjoyable experience".
- **Adversarial Training (maximize the model's loss):**
- Original Sentence: "The product is highly recommended".
- Adversarial Sentence: "The product is not recommended".

nlpaug Python Library Utilize for Text Augmentation

Text Mining: Text Pre-Processing

- Text Augmentation Methods (Adding More!)

- **Random Swap (Swapping Adjacent Words):**
- Original Sentence: "The weather is nice today".
- Swapped Sentence: "The ***nice is*** weather today".
- **Random Insertion (Enhancing Variation):**
- Original Sentence: "The weather is nice today ".
- Inserted Sentence: "The weather is ***very*** nice today".
- **Random Deletion (concise variations):**
- Original Sentence: "The weather ***is*** nice today ".
- Deleted Sentence: "The weather nice today ".

Amazon's Alexa Utilized (Back-Translation, Synonym Replacement & LLMs!)

Text Mining: Text Pre-Processing

- Text Pre-Processing

Persian Language Analysis

- **Hazm**: A comprehensive Python library for Persian language processing. It offers functionalities like tokenization, stemming, lemmatization, part-of-speech tagging, and more.
- **Persian NLP Toolkit (PNLPT)**: A Python toolkit designed for Persian tokenization, sentence segmentation, part-of-speech tagging, and dependency parsing.
- **Hazar**: A Python library for Persian text processing, such as tokenization, stemming, stop word removal, and spell checking.
- **FarsiYar:** An open-source Persian NLP toolkit developed by the University of Tehran, providing tokenization, stemming, lemmatization, POS tagging, and named entity recognition for Persian text.
- **WordCloudFa:** A Python Module that creates visual representations of Persian text data, with word size proportional to frequency, helping to identify keywords or themes quickly.
- **Parso** : Text Pre-Processing and NER

Text Mining: Text Pre-Processing

- Spell Checking and Correction (Persian Text)

- **Original Text:** "من یک کامپیوتر خریدم ولی کار نمی‌کنه. لطفا به من کمک کنید تا این مسئله رو برطرف کنم ".
- **Corrected Text:** "من یک کامپیوتر خریدم ولی کار نمی‌کند. لطفا به من کمک کنید تا این مسئله را برطرف کنم ".
- **Original Text:** " این کتاب خوبی است اما بسیار طولانیه. من نمی‌تونم تمام صفحات را بخوانم "
- **Corrected Text:** " این کتاب خوبی است اما بسیار طولانی است. من نمی‌اتونم تمام صفحات را بخوانم ".

Persian-Tools Python Library Utilize for Spell Checking

Text Mining: Text Pre-Processing

➢ Handling Special Characters (Persian Text)

➢ **Removing Special Characters and Emojis:**
- Before: " دوستم یک پیام خنده دار با ایموجی های خنده دار فرستاد! 😂😆🤣📱"
- After: " دوستم یک پیام خنده دار با ایموجی های خنده دار فرستاد "

➢ **URL Removal:**
- Before: " لینک خرید محصول را چک کنید: https://example.com/product "
- After: " لینک خرید محصول را چک کنید"

➢ **Encoding/Decoding:**
- Before: " من از **سایت‌های** وب خوشم نمی آید"
- After: " من از **سایت های** وب خوشم نمی آید"

➢ **Filtering:**
- Before: " تمام نشانه ها را باید حذف کنیم: +_()*&^%$#@"
- After: " تمام نشانه ها را باید حذف کنیم "

Hazm Python Library Utilize for Handling Special Character

Text Mining: Text Pre-Processing

➢ Text Normalization (Persian Text)

➢ **Persian Character Normalization:**

- Before: "من دارم **کتاب‌های** خوبی را می‌خوانم؛ آیا شما هم همین کار را می‌کنید "
- After: "ن دارم **کتابهای** خوبی را می‌خوانم؛ آیا شما هم همین کار را می‌کنید"

➢ **Accent and Diacritic Removal:**

- Before: "من ایران را دوست__دارم "
- After: "من ایران را دوست دارم "

➢ **Removing the Stretching of Persian Letters:**

- Before: " **ســـلام** چطوری؟ "
- After: " سلام چطوری؟ "

➢ **Removing Arabic Dialectic:**

- Before: " سَلام وَ عَلیکُم "
- After: "سلام و علیکم”

➢ **Unicode Normalization:**

- Before: " همین کار را در سال **هشتاد و پنج** انجام دادیم "
- After: " همین کار را در سال **1385** انجام دادیم "

Persian-Tools Python Library Utilize for Text Normalizing

Text Mining: Text Pre-Processing

- Part of Speech (Persian Text)

In Hazm, the part-of-speech tags use the following tags:

Noun (NOUN): "سگ " , "گربه" , "خانه"

Verb (VERB): "دویدن " , "نشستن" , "خوابیدن"

Adjective (ADJ): "زیبا " , "قرمز" , "شاد"

Adverb (ADV): "بسیار " , "سریعا" , "گاهی"

Pronoun (PRON): "او " , "ایشان" , "آنها"

Preposition (ADP): "در " , "بر" , "روی"

Conjunction (CONJ): "و " , "اما" , "یا"

Determiner (DET): "آن " , "این" , "یک"

Interjection (INTJ): "هی " , "آخ" , "وای"

Text Mining: Text Pre-Processing

- Custom Stop-Word Removal

Hazm allows you to use custom stop words when processing text ☺

- Original Text (before):
  - الگوریتمهای یادگیری ماشین به وسیله ماشینهای پشتیبان و شبکههای عصبی بسیار صنایع را با امکان سازی وظایفی چون تشخیص تصویر، پردازش زبان طبیعی و آنالیز پیشبینانه، به انقلاب رسیده است".
- Custom Stop-Words: "بسیار" و "است"

- Text after Stop-word Removal (after):
  - "الگوریتمهای یادگیری ماشین به وسیله ماشینهای پشتیبان شبکههای عصبی صنایع را با امکان سازی وظایفی تشخیص تصویر، پردازش زبان طبیعی، آنالیز پیشبینانه، به انقلاب رسیده".

- In the processed text, the words "بسیار" و "است" were removed based on the custom stop-word list provided. You can extend this list to include other stop words according to your needs.

Hazm Python Library Utilize for Custom Word Removal

Text Mining: Text Pre-Processing

- Context-aware Lemmatization (Persian Text)

- Original Text (before):
  - " مردان و زنان در بازار برای خرید دارو و غذا به فروشگاه می روند".

- Text after Lemmatization (after):
  - " مرد زن بازار خرید دارو غذا فروشگاه رفتن" .

- In this example:
- مردان is reduced to مرد
- و removed as it is a coordinating conjunction.
- زنان is reduced to زن
- و is removed as it is a preposition.
- برای is removed as it is a preposition.
- خرید remains the same as it is a base form.
- دارو remains the same as it is a base form.
- غذا remains the same as it is a base form.
- به is removed as it is a preposition.
- فروشگاه remains the same as it is a base form.
- می روند is reduced to رفتن

Hazm Python Library Utilize for Context-Aware Lemmatization

Text Mining: Text Pre-Processing

- Tokenization (Persian Text)

- Tokenization (Word Level):
- ['تکست ', 'ماینینگ', 'مطالبه', 'است', 'فرایند', 'استخراج', 'دانش', 'از', 'داده های', 'نامشخص', 'است']

- Tokenization (Sub-Word Level):
- ['ان', 'استراکچر', 'داده', 'پوزیشن', 'های', 'یونیک', 'چالش', 'ها', 'در', 'تکست', 'ماینینگ']

- Tokenization (Character Level):
- [' ', 'ر', 'م', ' ', 'ک', 'ی', ' ', 'ت', 'س', 'ا', ' ', 'گ', 'ن', 'ی', 'س', 'اس', 'ر', 'پ', 'ا', 'ر', 'پ', ' ', 'ا', 'ت', 'ا', 'د',
  'گ', 'ن', 'ی', 'ن', 'ا', 'ی', 'ا', 'م', ' ', 'ت', 'س', 'ک', 'ت', ' ', 'ر', 'د', ' ', 'ی', 'ر', 'و', 'ر', 'ض', ' ', 'ه', 'ل', 'ح']

Hazm Python Library Utilize for Tokenization

Text Mining: Feature Extraction

- Feature Extraction
- In text analysis, pre-processing comes before feature extraction. Pre-processing tasks like tokenization and normalization clean and standardize raw text. Feature extraction then derives key attributes from the pre-processed text, aiding subsequent analysis or modeling. Common feature extractor methods distill important insights from text data. Most frequent feature extraction methods include:

- Term Frequency-Inverse Document Frequency (TF-IDF)
- Word Embedding
- Topic Modelling
- NER

Text Mining : Feature Extraction

- Term Frequency-Inverse Document Frequency (TF-IDF)

$$W_{x,y} = tf_{x,y} \times \log\left(\frac{N}{df_x}\right)$$

Two-fold heuristics based on frequency

TF (Term frequency)

More frequent ***within*** a document → more relevant to semantics

e.g., "query" vs. "commercial

IDF (Inverse document frequency)

Less frequent ***among*** documents → more discriminative

e.g. "algebra" vs. "science"

TF-IDF

Term x within document y

$tf_{x,y}$ = frequent of x in y

df_x = number of documemts containing x

N= total number of documents

Text Mining : Feature Extraction

- Word Embedding

- **Word2Vec (Developed by Google)**
  - Skip-Gram
  - Continuous Bag Of Word (CBOW)

Glove (Developed by NLP group in 2014)

- Generates word embeddings using global word co-occurrence statistics, creating a matrix for factorization to obtain word vectors.

Fast-Text (Developed by Facebook in 2016)

- Enhances Word2Vec by representing words as a combination of character n-grams, improving the handling of sub-word information and rare or out-of-vocabulary words.

Bert (Developed by Google in 2018)

- BERT enhances word embeddings with transformer-based architecture, learning bidirectional contextual representations for improved context-based understanding in NLP tasks.

Text Mining : Feature Extraction

Word2Vec

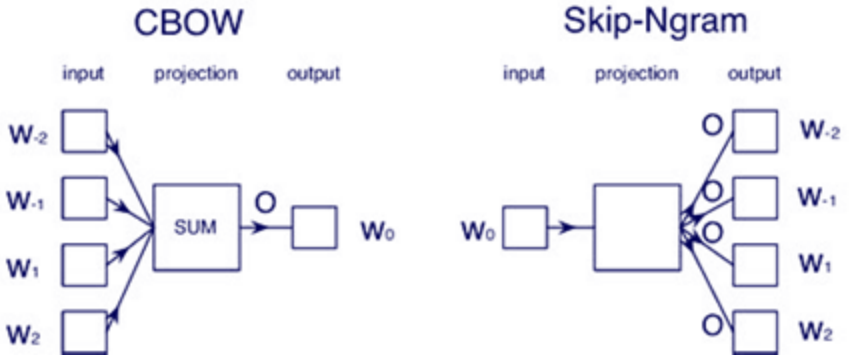
e.g. " The quick brown fox jumps over the lazy dog " *With a context window of size 2:*

- Skip-Gram

  (The, quick), (The, brown), (quick, brown), (quick, fox), (brown, fox), (brown, jumps), (fox, jumps), (fox, over), (jumps, fox), (jumps, over), (over, jumps), (over, the), (the, lazy), (lazy, dog)

- Continuous Bag Of Word (CBOW)

  (The, quick): The quick, (brown, The): The brown, (quick, The): quick The, (quick, brown): quick brown, (fox, quick): fox quick, (brown, fox): brown fox, (brown, jumps): brown jumps, (fox, jumps): fox jumps, (fox, over): fox over, (jumps, fox): jumps fox, (jumps, over): jumps over, (over, jumps): over jumps, (over, the): over the, (the, lazy): the lazy, (lazy, dog): lazy dog



Text Mining : Feature Extraction

GLoVe

- Consider the following simple corpus with two sentences:
- "The cat sat on the mat."
- "The dog played in the park."

Now, let's construct a co-occurrence matrix for this corpus *With a context window of size 2:*

each cell X_{ij} represents the co-occurrence count of the word in the row with the word in the column

	the	cat	sat	on	mat	dog	played	in	park
the	0	2	1	2	1	1	0	2	0
cat	2	0	1	0	1	0	0	0	0
sat	1	1	0	1	0	0	0	0	0
on	2	0	1	0	1	0	1	1	1
mat	1	1	0	1	0	0	0	0	0
dog	1	0	0	0	0	0	1	1	1
played	0	0	0	1	0	1	0	1	0
in	2	0	0	1	0	1	1	0	1
park	0	0	0	1	0	1	0	1	0

The cell $X_{cat,the}$ indicates that "cat" and "the" co-occur twice within a window size of 2.
The cell $X_{sat,on}$ indicates that "sat" and "on" co-occur once within a window size of 2.

Text Mining : Feature Extraction

Fast-Text

- Consider the word "apple" and assume we're using 3-grams (trigrams). Fast-Text would represent "apple" as the sum of vectors for the following n-grams:
  - "app" / "ppl" / "ple" / "apple"

The vector for "apple" would be the sum of the individual n-gram vectors:

- V("apple") = V("app") + V("ppl") + V("ple") + V("apple")

- Fast Text effectively handles rare and out-of-vocabulary words by representing them as combinations of their subword vectors, enabling meaningful representations even for unseen words in the training corpus.

- V("pineapple") = V("pin") + V("ine") + V("nea") + V("app") + V("ppe") + V("ple") + V("ineapple") + V("neapple") + V("eapple") + V("pineapple")

- This approach enables Fast-Text to learn better word representations and improves performance in various NLP tasks.

Text Mining : Feature Extraction

Bert (Transformer)

- **Masked Language Modelling (MLM):**

- Input Sentence: "The [MASK] is blue."
- Expected Output: "The sky is blue."

- In this example, BERT is presented with a sentence where one word is masked (in this case, "sky"). BERT's task is to predict the masked word based on the surrounding context. By successfully predicting the masked word ("sky"), BERT demonstrates its understanding of the sentence's context.

- **Next Sentence Prediction (NSP):**

- Input Sentences:
- Sentence 1: "I ordered a pizza for dinner".
- Sentence 2: "It was delicious".
- Expected Output: "Is sentence 2 a continuation of sentence 1?" (Answer: Yes)

- In this example, BERT is presented with two consecutive sentences. Its task is to determine whether the second sentence logically follows the first one. By correctly predicting that "It was delicious." is a continuation of "I ordered a pizza for dinner.", BERT shows its ability to understand relationships between sentences.

Text Mining : Feature Extraction

- Word-Embedings (Pros & Cons)

- **Word2Vec:**

Pros: Efficient training and inference, captures semantic relationships, simple architecture.
Cons: Ignores context dependency, may struggle with out-of-vocabulary words, limited to word-level semantics.

- **GloVe:**

Pros: Captures global word co-occurrence effectively, provides meaningful representations for rare words, efficient training.
Cons: Ignores word order, struggles with the context in ambiguous words, and pretrained models may not generalize to specific tasks.

- **FastText:**

Pros: Efficient with out-of-vocabulary words, handles morphologically rich languages well, trained on large corpora.
Cons: Lower quality embeddings, slower inference, requires hyperparameter tuning.

- **BERT:**

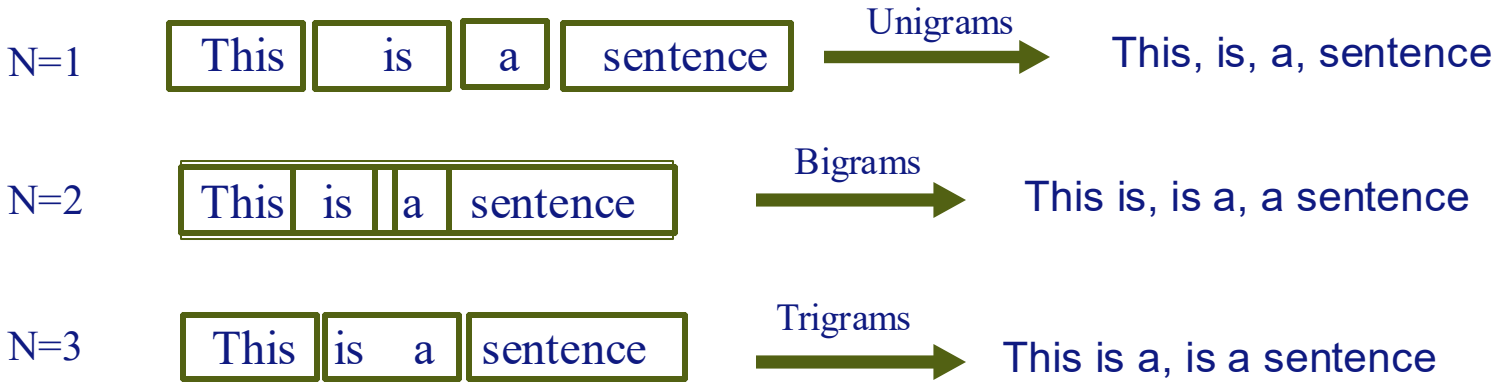
Pros: Captures bidirectional context, state-of-the-art performance, fine-tuning for specific tasks.
Cons: Requires substantial computational resources, needs labelled data for fine-tuning, and large pre-trained models.

Text Mining's Application: N-Gram

- N-Gram

- Common N-Gram types:

- Unigrams (1-grams): Individual words or characters, e.g., "This" or "is".
- Bigrams (2-grams): Sequences of two words or characters, e.g., "This is" or "ab".
- Trigrams (3-grams): Sequences of three words or characters, e.g., "This is a" or "abc".



Text Mining's Application: N-Gram

- N-Gram (Persian Text)

- Original Sentence:
  - "زبان فارسی یکی از پرمخاطب‌ترین زبان‌های دنیاست ".

N=1.

['زبان', 'فارسی', 'یکی', 'از', 'پرمخاطب', 'ترین', 'زبان', 'های', 'دنیاست'].

N=2.

[('زبان', 'فارسی'), ('فارسی', 'یکی'), ('یکی', 'از'), ('از', 'پرمخاطب'), ('پرمخاطب', 'ترین'), ('ترین', 'زبان'), ('زبان', 'های'), ('های', 'دنیاست')].

N=3.

[('زبان', 'فارسی', 'یکی'), ('فارسی', 'یکی', 'از'), ('یکی', 'از', 'پرمخاطب'), ('از', 'پرمخاطب', 'ترین'), ('پرمخاطب', 'ترین', 'زبان'), ('ترین', 'زبان', 'های'), ('زبان', 'های', 'دنیاست')].

Text Mining's Application: Topic Modeling

➢ Topic Modelling

- **Latent Dirichlet Allocation (LDA)**: assumes documents are mixtures of topics, with each word associated with a specific topic within the document.

e.g. " Given a collection of news articles" LDA identify topics such as "politics," "economy," and "sports".

- **Non-Negative Matrix Factorization (NMF)**: breaks down the document-term matrix into two matrices: one for topics and the other for topic distributions in documents.

e.g. In a collection of product reviews, NMF identifies topics such as "performance," "design," and "price ".

- **Latent Semantic Analysis (LSA):** applies singular value decomposition (SVD) to the document-term matrix to discover latent topics.

e.g. Given a set of academic papers, LSA identifies topics related to "machine learning," "natural language processing," and "data mining" based on co-occurrence patterns of words.

- **Hierarchical Dirichlet Process (HDP)**: an extension of LDA that automatically infers the number of topics from the data.

e.g. In a collection of social media posts, HDP discovers topics such as "technology," "entertainment," and "health," and represents each post accordingly.

Text Mining's Application: Topic Modeling

➢ Topic Modellings (Pros & Cons)

➢ **Latent Dirichlet Allocation (LDA):**
- **Pros:** Interpretable topics, Widely used, Scalable, Flexible
- **Cons:** Requires a predefined number of topics, Sensitivity to parameters, Assumes bag-of-words representation

➢ **Non-Negative Matrix Factorization (NMF):**
- **Pros:** Parts-based representation, No negative values, Simplicity, Less sensitive to parameters
- **Cons:** Less widespread usage, Initialization sensitivity, Lack of probabilistic interpretation

➢ **Latent Semantic Analysis (LSA):**
- Pros: Captures semantic meaning, Dimensionality reduction, No need for pre-processing
- Cons: Lack of topic interpretability, No probabilistic framework, Sensitivity to noise

➢ **Hierarchical Dirichlet Process (HDP):**
- **Pros:** An infinite number of topics, Flexibility, Handles data sparsity
- **Cons:** Complexity, Interpretability challenges, Sensitivity to hyperparameters

Text Mining's Application : Name Entity Recognition

- Name Entity Recognition (Important Entities extration)

➢ Person Named Entity Recognition:
- Example: "John Smith"
- Named Entity: [John Smith]

➢ Location Named Entity Recognition:
- Example: "Paris, France"
- Named Entity: [Paris, France]

➢ Organization Named Entity Recognition:
- Example: "Google Inc."
- Named Entity: [Google Inc.]

➢ Date Named Entity Recognition:
- Example: "January 1, 2023"
- Named Entity: [January 1, 2023]

➢ Numeric Named Entity Recognition:
- Example: "$100 million"
- Named Entity: [$100 million]

➢ Time Named Entity Recognition:
- Example: "9:00 AM"
- Named Entity: [9:00 AM]

NER categorizes important entities from unstructured text, enabling applications like information retrieval, question answering, and sentiment analysis.

Text Mining's Application : Sentiment Analysis

- Sentiment Analysis

- **Level of analysis:**

- **Aspect-based (ABSA):**

Review: "The sushi was excellent, but the service was slow".
Aspect 1: "sushi" → Sentiment: Positive
Aspect 2: "service" → Sentiment: Negative

- **Document-level:**

Review: "The laptop is fast, but its battery life is terrible". → Document-level sentiment: Negative

- **Sentence-level:**

Review: "The laptop is fast, but its battery life is terrible".
Sentence 1: "The laptop is fast". → Sentiment: Positive
Sentence 2: "Its battery life is terrible". → Sentiment: Negative

Text Mining's Application : Sentiment Analysis

- Sentiment Analysis

- **Sentiment Classes:**

- **Binary Classification:**

Binary Classification: A movie review is classified as either positive or negative.
Review: "The movie was boring and predictable". → Sentiment: Negative

- **Multi-class Classification:** A customer service feedback is classified as positive, negative, or neutral.

Feedback: "The support agent was friendly but couldn't solve my problem"
Sentiment: Neutral

- **Fine-grained Classification:** A book review is classified with more nuanced sentiment classes.

Review: "The book was captivating and beautifully written".
Sentiment: Very Positive

Text Mining's Application: Sentiment Analysis

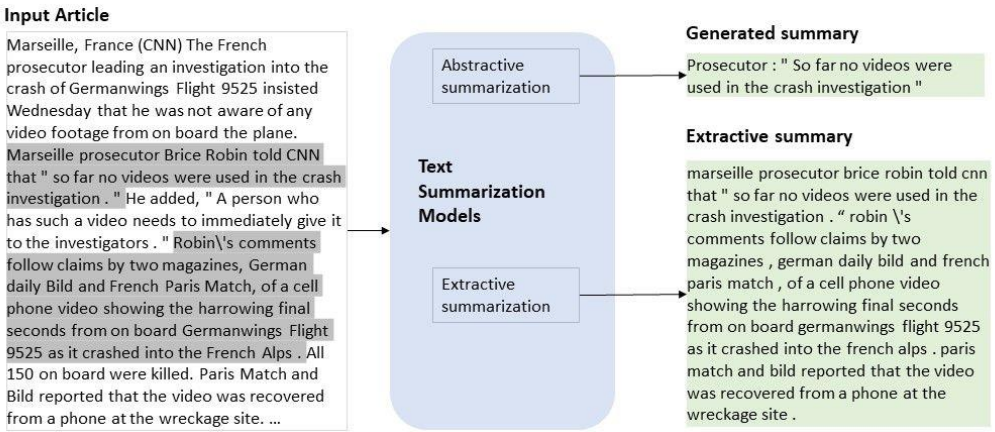
- Sentiment Analysis
- **Sentiment analysis applies to various real-world applications:**

- **Amazon:** Utilizes sentiment analysis on product reviews to understand customer satisfaction and improve product recommendations.
- **McDonald's:** Analyzes social media conversations to enhance customer satisfaction and improve menu offerings.
- **Nike:** Examines customer feedback for product perception insights and opportunities to enhance customer experience.
- **Uber:** Monitors user feedback and ratings to identify areas of improvement for services and driver performance.
- **Airbnb:** Uses sentiment analysis on guest reviews and host ratings to understand user preferences and enhance trust and satisfaction.
- **Coca-Cola:** Tracks social media conversations to evaluate marketing effectiveness and guide future campaigns.
- **Procter & Gamble (P&G):** Leverages sentiment analysis on customer feedback for improving products and marketing strategies.
- Star**bucks:** Monitors social media for brand sentiment, product feedback, and service improvement opportunities.

Text Mining's Application : Text Summarization

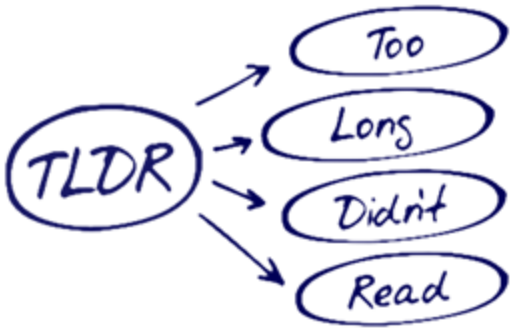
- Text Summarization
- **Extractive Summarization:**
  - Sentence scoring
  - Graph-based algorithms like TextRank or LexRank,
  - Machine learning methods such as SVM, KNN and RNN

- **Abstractive Summarization:**
- Sequence-to-sequence models (encoder-decoder architectures)
- Attention mechanisms
- Reinforcement learning
- Pretrained language models like BERT, GPT, or T5



Text Mining's Application : Text Summarization

- Text Summarization's Application



Google's TL;DR Feature (Summarization Based)

Text Mining: Modeling

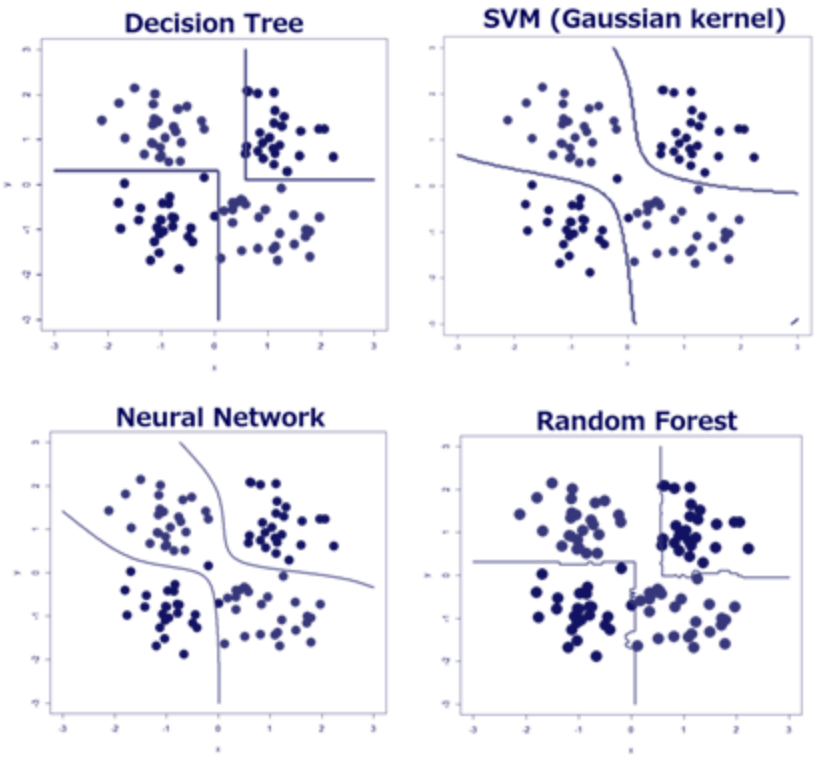
Modelling Feature Extraction from Text: A Structured Transformation Pipeline for Analysable Features

- Supervised Learning
- Unsupervised Learning
- Semisupervised Learning
- Transfer Learning

Text Mining: Modeling

Supervised Learning

- Classification
- Artificial Neural Network
- Support Vector Machine
- K-Nearest Neighbor
- Logistic Regression
- Gradian Boosting
- Random Forest
- Decision Tree
- Naive Bayes

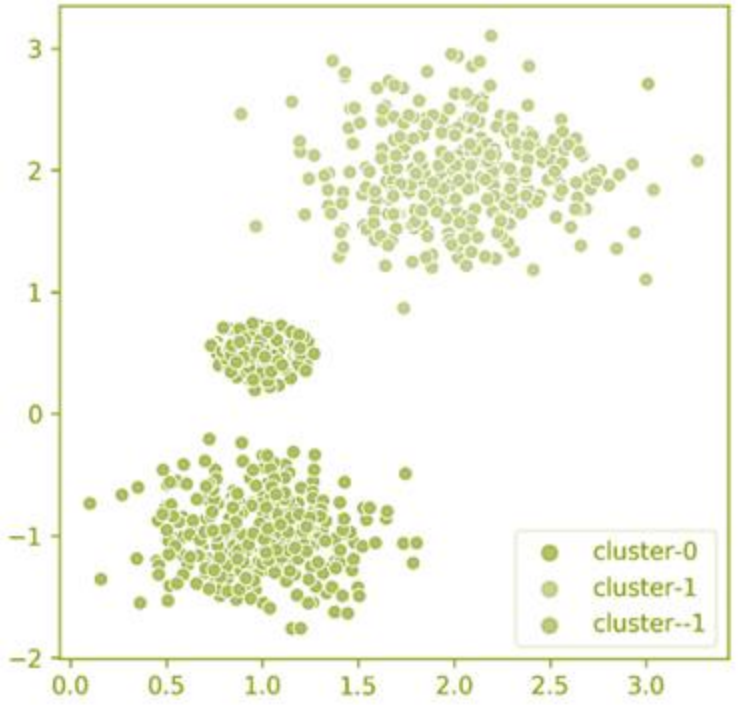


Performance of Supervised Learning Algorithms

Text Mining: Modeling

Unsupervised Learning

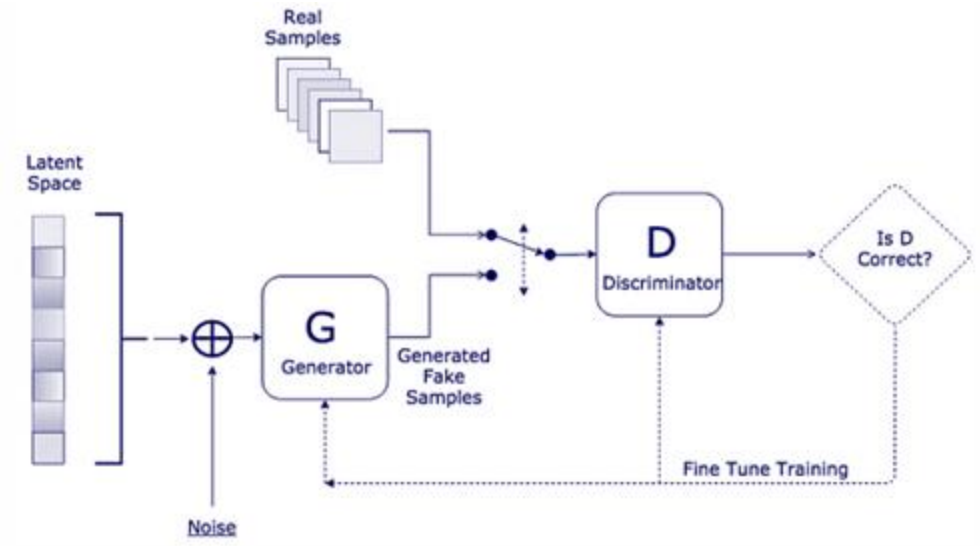
- DBSCAN (Density-Based Spatial Clustering of Applications with Noise)
- t-Distributed Stochastic Neighbour Embedding (t-SNE)
- Non-negative Matrix Factorization (NMF)
- Principal Component Analysis (PCA)
- Gaussian Mixture Models (GMM)
- Self-organizing Maps (SOM)
- Hierarchical clustering
- K-means clustering
- Autoencoders



Text Mining: Modeling

Semisupervised Learning

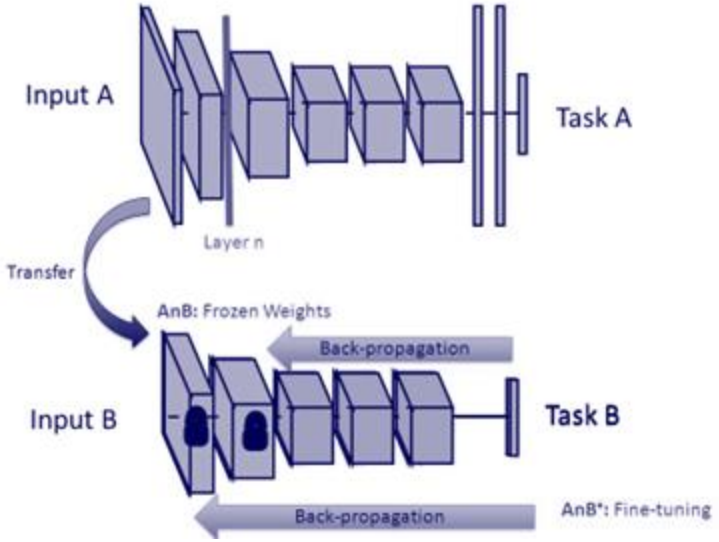
- Semi-Supervised Generative Adversarial Networks (GANs)
- Semi-Supervised Support Vector Machines (S3VM)
- Semi-supervised variational Autoencoders (VAEs)
- Transudative Support Vector Machines (TSVM)
- Graph-Based Semi-Supervised Learning
- Consistency Regularization Methods
- Label Propagation
- Self-training
- Co-training



Text Mining: Modeling

Transfer Learning

Transfer learning in NLP involves using knowledge from pre-trained models on large datasets to improve performance on specific tasks with smaller datasets. Fine-tuning pre-trained models on task-specific data, enables faster convergence and better results, especially when labelled data is scarce or expensive.



Text Mining: Document Similarity

➢ Document Similarity

Docs Similarity (Euclidean Distance): measures the similarity between two document vectors by computing their Euclidean distance in a high-dimensional space, aids tasks like clustering, retrieval, plagiarism detection, and summarization by quantifying the closeness of documents.

e.g. two corps represented by sets of words:

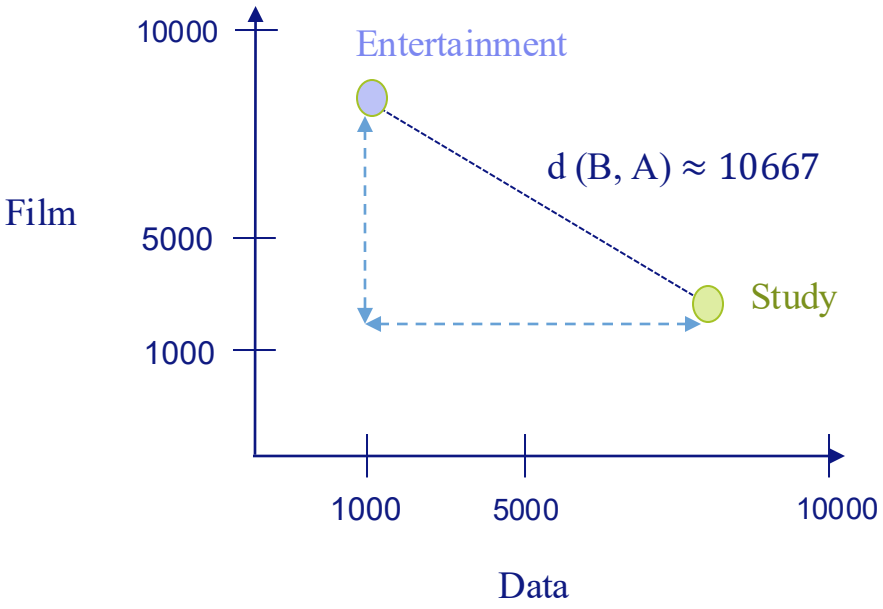
Corpse A: (500, 7000)

Corpse B: (9320, 1000)

$$\sqrt{(B_1 - A_1)^2 + (B_2 - A_2)^2}$$

$$c^2 = a^2 + b^2$$

$$d\,(B, A) = \sqrt{(8820)^2 + (-6000)^2}$$



Text Mining: Document Similarity

- Document Similarity's Applications

- Plagiarism Detection
- Text Summarization
- Duplicate Detection
- Content Recommendation
- Document Classification
- Question Answering

"Grammarly's Use of Document Similarity"

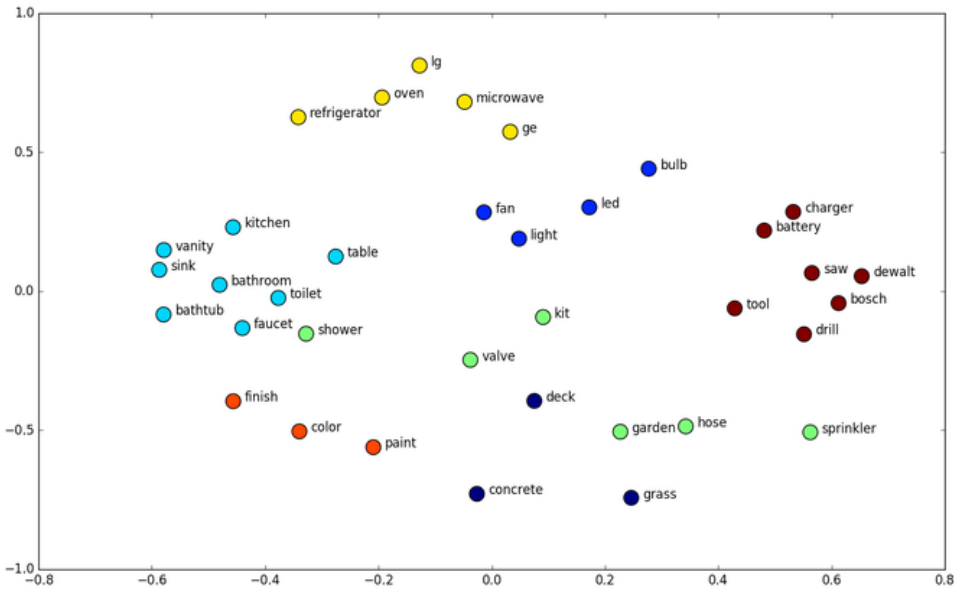
Text Mining: Visualization

➢ Visualization Methods

- **Embedding Visualizations:** revealing semantic relationships in 2D or 3D
- **Word Cloud**: Visualize the frequency or importance of words in a corps

Text Mining: Visualization

- Visualization Methods



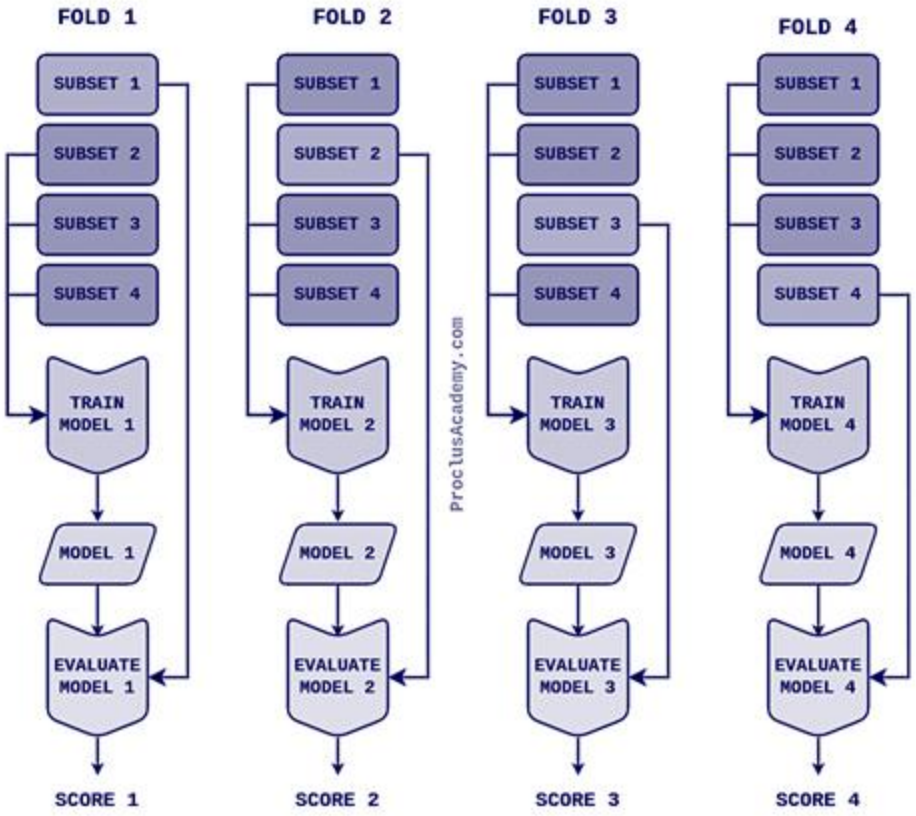
Embedding Visualization

Word-Cloud

Text Mining: Evaluation

- Final Evaluation

- Partitioning Data
- Training and Validation
- Performance Evaluation
- Reducing Variance
- Parameter Tuning
- Bias-Variance Trade-off
- Randomization
- Stratified K-Fold
- Choosing k
- Cross-Validation Variants



4 fold Cross-Validation

Text Mining: Evaluation

➢ Final Evaluation

- True Positive (TP): The number of instances that are correctly predicted as positive.
- True Negative (TN): The number of instances that are correctly predicted as negative.
- False Positive (FP): The number of instances that are incorrectly predicted as positive.
- False Negative (FN): The number of instances that are incorrectly predicted.

		Predicted Class: Positive	Predicted Class: Negative	
Actual Class	Positive	True Positive (TP)	False Negative (FN) Type II Error	**Sensitivity** $\frac{TP}{(TP+FN)}$
	Negative	False Positive (FP) Type I Error	True Negative (TN)	**Specificity** $\frac{TN}{(TN+FP)}$
		Precision $\frac{TP}{(TP+FP)}$	**Negative Predictive Value** $\frac{TN}{(TN+FN)}$	**Accuracy** $\frac{TP+TN}{(TP+TN+FP+FN)}$