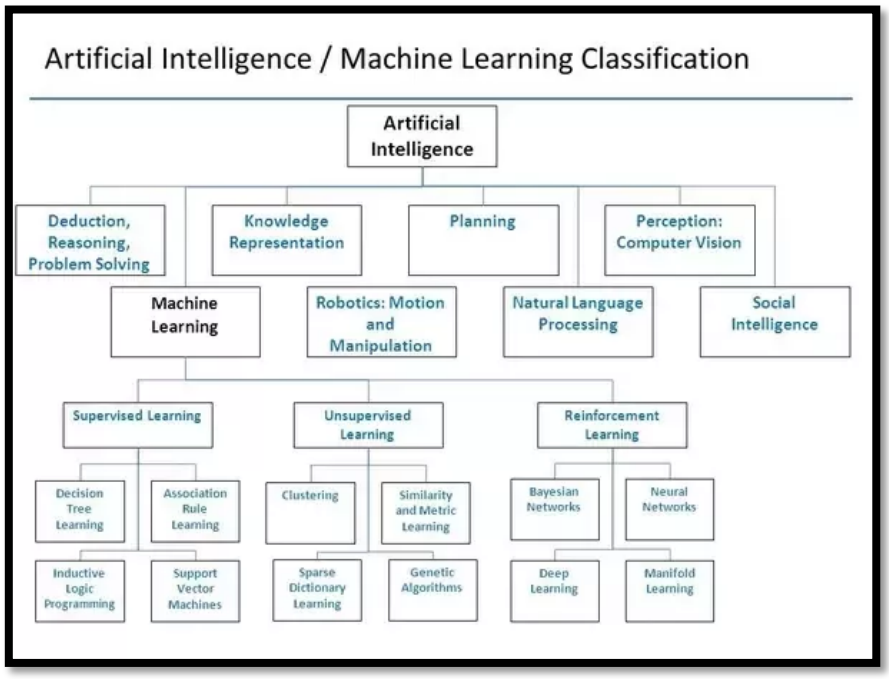
MSDS-7337-Natural Language Processing-Glossary

Rashmi Patel[1]
[1]Master of Science in Data Science
Southern Methodist University
Dallas, Texas, USA
rashmip@smu.edu

Abstract: This document contains the important terms and concepts that are used and implemented in Natural Language Processing (NLP).

Unit1: Introduction

What is Natural Language Processing (NLP)?

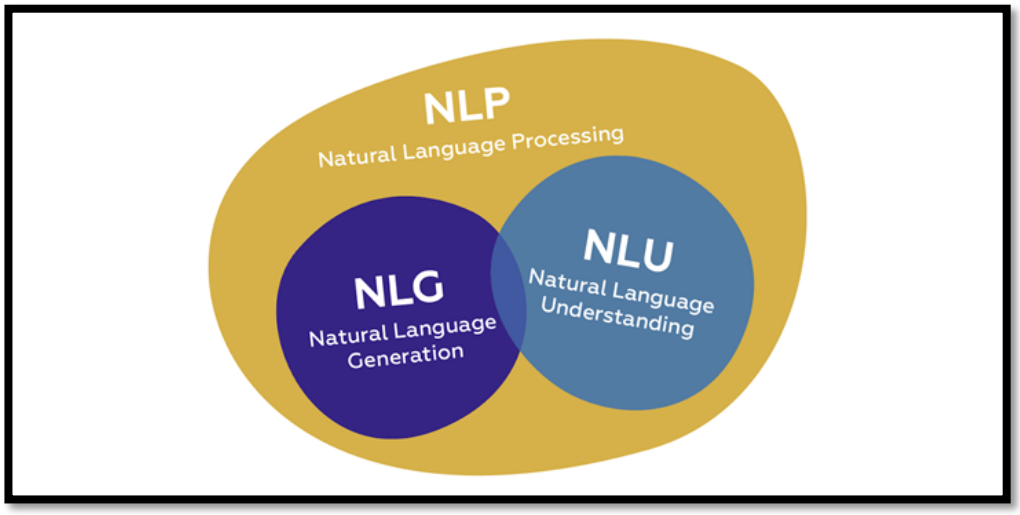


Natural Language Processing (NLP), a branch of artificial intelligence (AI), is nowadays most frequently used technology which concerns about how the computer/ machine can understand text and spoken words like humans. Natural Language Processing (NLP) is a combination of computational linguistics, machine learning and deep learning modelling techniques.

Natural Language vs. Artificial Language:

Natural languages are those that evolved or emerged gradually over time, largely unconsciously. Examples of Natural Languages are: English, Spanish, Japanese, sign language, etc.
Artificial Language are those that were designed, crafted, or invented with conscious purpose, largely all at once and not gradually. Examples of Artificial Languages are: LISP, C, C++, Python, etc

Two sides of NLP:



There are two sides of Natural Language Processing (NLP):

1. Natural Language Understanding (NLU)
2. Natural Language Generation (NLG)

Natural Language Understanding (NLU): The branch of NLP that uses computer software to understand input made in the form of natural language and produce a useful representation of some inputted natural language.
Natural Language Generation (NLG): The branch of NLP that uses computer software to generate usable, natural language output that is not just identical to its input.

Applications of NLP:

As we know that there are 2 sides of NLP, we will be looking at applications of both NLU and NLG.

Applications of NLU:

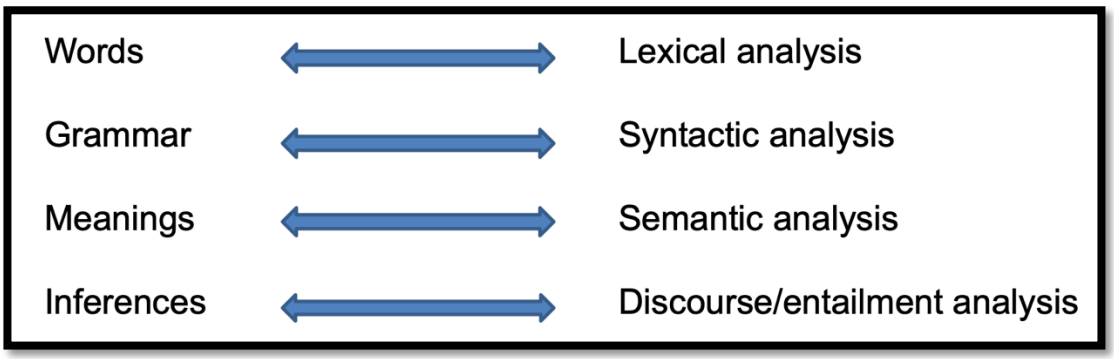
1. Text annotation
    a. Tagging
    b. Metadata extraction/generation
    c. Classification
    d. Document summarization
2. Corpus analytics
    a. Theme extraction
    b. Clustering
    c. Taxonomy mapping
    d. Sentiment analysis
3. Search applications
    a. Query repair
    b. Query refinement
    c. Results postprocessing
4. Advanced applications

   a. Machine translation
   b. Knowledge discovery
   c. Question handling
      i. Question typing/routing
      ii. Question-FAQ approximate matching

Applications of NLG:

1. Text annotation
   a. Document summarization
   b. Generation of callouts/headlines
2. Corpus analytics
   a. Labelling of clusters
   b. Synopsizing of corpus-wide topic and/or sentiment trends
3. Search applications
   a. Advanced capsule generation (summarization modified to fit the query)
   b. Advanced query refinement (next-gen version of "did you mean?" for disambiguation of query's meaning)
4. Advanced applications
   a. Machine translation (construction in target language)
   b. Knowledge discovery (human-friendly presentation)
   c. Question handling
      i. Question refinement
      ii. Question answering

Unit2: Levels of Analysis in NLP



Lexical analysis:

1. Fundamental foundation of all analyses that analyze words.
2. Analysis goes up and down through these levels to help correct errors and fill-in gaps for all other levels of analysis.

Terminology and Concepts in Lexical Analysis:

1. **Lexicon** - dictionary, vocabulary
2. **Morphology**-study of morphemes-units of which words are made
3. **Stemmer** - algorithm that provides root morpheme

4. **Metadata** - corpus derived meta-data supports developing : word frequency score, common collocation, and commonly co-occurring words and context words
5. **Collocations** - words commonly occurring together
6. **Head word** - specific listing in the lexicon
7. **Polysemous** - more than one meaning
8. **Text and Corpus Analytics** - includes spell correction, terminology extraction, lexical diversity measurement
9. **WordNet** - the Bible of word senses

Syntactic analysis:

1. Grammar like analysis. Putting full sentences through a program that can draw structure of grammar.
2. Analysis goes up and down through these levels to help correct errors and fill-in gaps for all other levels of analysis.
3. Second level of NLP that focuses on grammatical analysis of documents

Terminology and Concepts in Syntactic Analysis:

1. **Sentence boundary detection** - needed since syntax analysis is sentence by sentence analysis. grammar parsing relies on sentence boundary detection. non-trivial problem.
2. **Part of speech (POS)** - each word in a sentence can be assigned to a part of speech. most words have more than one part of speech; therefore, POS tagging is non-trivial. needs contextual clues - words preceding and succeeding,
3. **Penn Treebank Tag set** - commonly used tag sets for parts of speech, approximately 40 different POS with this set, support eventual grammar parsing
4. **Parsing** - break the sentence into grammar parts
5. **Lemmatization** - canonical (conventional) form that represents a set of related word forms
6. **Discrete text field analysis** - unitizing, normalizing, smart ETL.

Semantic analysis:

1. Entire meanings based on usage in a full like sentence way. 90% of data science projects stop here.
2. Analysis goes up and down through these levels to help correct errors.

Terminology and Concepts in Semantic Analysis:

1. **Named entity extraction (NEE a.k.a. NER)** - recognizes the entities (persons organizations, places, events) without typing.
2. **Relationship extraction (between NEs)** - What relation does one NE have to another? Sometimes we can answer this during the NEE process.
3. **Word sense disambiguation (WSD)** - still unsolved problem in AI - how to distinguish the meaning of a word in context, from all of the possible meanings of the word

4. **Classification** - we use a tree-structured graph to place documents into categories. Often, we employ machine learning methods, such as SVM (support vector machine).
5. Additional types of semantic analysis include:
    a. Tagging
    b. Topic segmentation
    c. Sentiment analysis

Discourse analysis:

1. Entire meanings that can be inferred based on another semantic meaning
2. Nuances of human conversation like metaphorical, ironic, comical, etc.

Terminology and Concepts in Discourse Analysis:

1. **Anaphora resolution** - the problem of resolving what a pronoun, or a noun phrase refers to.
2. **Discourse modelling** - A discourse model is a mental object that constitutes an. individual's knowledge of a discourse. It is constructed based on what has occurred in the discourse supplemented by general and specific knowledge.
3. **Question answering** – Most straightforward approach is matching pre-existing Q-A materials. The matching is often not straightforward. We can build models that adapt question formats.
4. **Textual entailment** -Inferences generated follows a strict logic. In textual entailment framework, the entailing and entailed texts are termed text (t) and hypothesis (h), respectively.
5. **Pragmatic analysis** – Inferences generated does not follow strict logic.

The below table summarizes typical levels of analysis employed in NLP. Generally, we use lower levels also when we move to "higher" levels of analysis.

***Minimum* level of analysis generally used for implementations**	**Lexical Analysis**	**Syntactic Analysis**	**Semantic Analysis**	**Discourse Analysis**
Stemming	X			
Keyword Tagging	X			
Spell Correction	X			
Terminology Extraction	X			
Reading Level Estimation	X			
Sentence-Boundary Detection	X	X		
POS-Tagging	X	X		
Parsing	X	X		
Lemmatization	X	X		
Unitizing/Normalizing	X	X		
NEE	X	X	X	
WSD	X	X	X	
Conceptual Tagging	X	X	X	
Classification	X	X	X	
Topic Segmentation	X	X	X	
Sentiment Analysis	X	X	X	
Anaphora Resolution	X	X	X	X
Discourse Modeling	X	X	X	X
Question Answering	X	X	X	X
Textual Entailment	X	X	X	X
Pragmatic Analysis	X	X	X	X

Unit3: Trade-Offs in NLP

Shallow vs. Deep:

1. When there is a need to answer a question like- Shall I build a very robust, exhaustive representation of text meaning, or shall I just pick out a few elements needed for my application? then the best approach is Shallow vs. Deep.
2. Shallow NLP works on breaking sentences into chunks whereas deep NLP consists of 3 or more hidden layers.

Statistical vs. Symbolic:

1. When there is a need to answer a question like-Shall I leverage powerful, complex statistical methods, or use rules that can be operated on by strict logic? then the best approach is Statistical vs. Symbolic.
2. In symbolic NLP we have a rule-based system whereby discrete tests are performed to apply each rule whereas a combination of signals or nodes of varying strengths is used in statistical NLP.

Feature engineering vs. Feature learning:

1. When there is a need to answer a question like- Shall I involve human experts to engineer features, or shall I use them only to establish a training set? the best approach is Feature Engineering vs. Feature learning.
2. In feature learning, deficiencies in precision or recall are addressed by expanding the training set whereas, deficiencies in precision or recall are addressed by examining counterexamples with the SMEs in feature engineering.

Top-down vs. bottom- up:

1. When there is a need to answer a question like- Shall I start with high-level classifications and text characteristics, then gradually break them down into more detail, or shall I do the reverse? then the best approach is Top-down vs. Bottom-Up.
2. Top-down approach involves building your NLP framework around a set of high-level concepts and categories into which everything fits whereas bottom-up approach involves building up a huge list of stemmed keywords from raw text, then try to roll them up into phrases, tag clouds, hypernym trees, and the like.
3. Non-technical people prefer top-down approach whereas technical people use bottom-up approach as this sets you up to have rich context information for every feature of each document.

Transparent vs. opaque (AI vs. XAI):

1. When there is a need to answer a question like- Shall I engineer a system that is explainable to an intelligent SME or is it okay to build something only my fellow data

scientists and engineers could understand? then the best approach is Transparent vs. Opaque.
2. Transparent means it is easy enough to see what the algorithm is doing whereas opaque is result of more complex machine learning and produces results not easily explainable

Unit4: Working in NLP

Job roles that utilize NLP:

1. Software engineer
2. Knowledge engineer
3. Data scientist
4. DBA
5. Applied linguistics researcher
6. Cognitive scientist
7. Marketing technologist

Sectors that utilize NLP:

Public and private sectors that use NLP include:
1. Information services
2. eCommerce
3. Customer service desks
4. Law enforcement or military
5. Legal
6. Business intelligence
7. Consumer devices
8. Embedded technologies
9. Publishing
10. Research institutes

Organizations that relate to NLP:

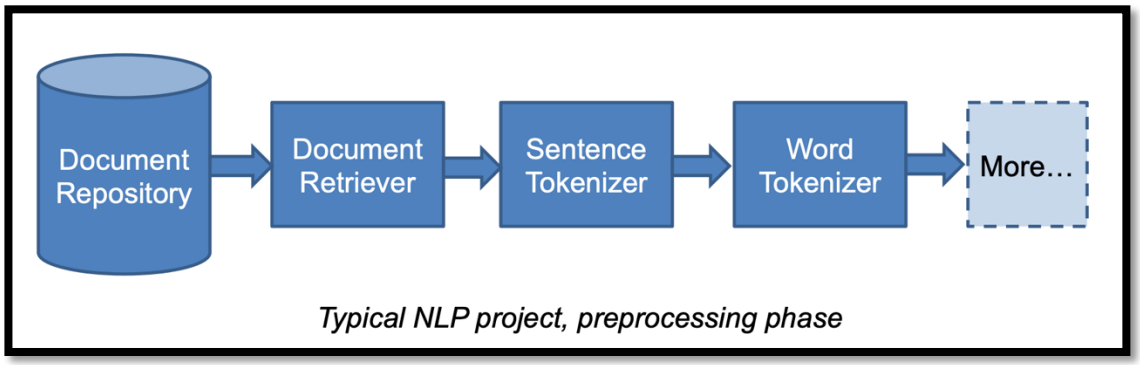
Associations supporting NLP include:
1. ACM- Association for Computing Machinery
2. IEEE-Institute of Electrical and Electronics Engineers (IEEE Computer Society)
3. AAAI- Association for the Advancement of Artificial Intelligence
4. IJCAI- International Joint Conference on Artificial Intelligence Organization
5. AAAL- American Association for Applied Linguistics
6. ICLA- International Cognitive Linguistics Association

Unit5: Low-Level Analysis

Text pre-processing:

To perform this step we need to do following:

1. Sentence segmentation (sentence tokenization)
2. Lexical analysis (word tokenization)



Text normalization:

To normalize a word-tokenized text source, we often address the following: •

1. Contractions, expansion
2. Stop words, removal
    a. **Content words**: Content words are an open class, meaning they are an unfinished list to which new words are readily added. Examples are dog, cat, angrily, run, clever, democracy, green, God, wisdom.
    b. **Function words**: are a closed class, meaning they are a fixed list to which no new words are added. Examples are is, am, are, was, were, he, she, you, we, they, if, then, therefore, possibly.

Stop words: Our baseline list of stop words is usually just the list of all the function words in our language. a stop word list is application specific, which means we usually add some content words to it that we want to discard. Examples are: : Mr., Mrs., Ms., Dr., Prof., Fr., etc.

3. Misspellings- Two fundamental approaches:
    a. **Edit-distance method**: Adding, replacing, or deleting a character. Our basic algorithm returns the word with the lowest edit distance, up to a maximum distance (typically three).
    b. **Fuzzy string compare**: Characters-in-common as a percentage of total characters. Characters must be in like-order.
4. Stemming- Stemming is a task wherein NLP utilizes morphology. Morphology studies how words can be made from smaller parts that have meaning (or have an impact on meaning). The smaller parts are called morphemes. Morphemes are of two main types:
    a. **Stems**-Supplies the main meaning. It can sometimes be a word (e.g., “jump” in “jumping”)
    b. **Affixes**-Adds or alters meaning. It cannot usually be a word by itself.

Low-level document feature extraction:

1. Primary features-Require us only to examine the document itself.
   Examples:
    a. Word frequencies.
    b. Collocations (n-grams)

2. Secondary features-Require us to compare features of the document to those of other documents. Examples:
    a. Differential frequency analysis (TF/IDF)
    b. Relative lexical diversity
    c. Reading level

Unit6: Lexical Knowledge Bases

Monosemy = having only one sense
Polysemy = having more than one sense

Lexical knowledge bases: A lexical knowledge base ("lexical KB") break the words into senses, and linking senses to senses via relations such as:

1. Synonym/antonym: Synonyms are words that have the same, or almost the same, meaning as another word whereas antonyms are words that have the opposite meaning of a given word.(https://courses.lumenlearning.com/buswriting/chapter/3-5-synonyms-and-antonyms/)
2. Hyponym/hypernym: A hypernym describes a broader term, for example cutlery, or dog. A hyponym is a more specialised and specific word, for example: spoon would be a hyponym of cutlery and Labrador would be a hyponym of dog. (https://www.mytutor.co.uk/answers/7824/A-Level/English-Language/what-is-the-difference-between-a-hyponym-and-a-hypernym/)
3. Holonym/meronym: Holonym is (a term that denotes a whole whose part is denoted by another term, such as 'face' in relation to 'eye' while meronym is (semantics) a term used to denote a thing that is a part of something else.(https://wikidiff.com/holonym/meronym)

WordNet: Lexical databases derived from the original Princeton WordNet

1. It groups words into synonym sets and interlink them using lexical and conceptual-semantic relations
2. A combination of dictionary and thesaurus - intuitively usable - supports automatic text analysis and artificial intelligence applications
3. WordNet was developed by the Cognitive Science Laboratory at Princeton University under the direction of Professor George A. Miller (Principal Investigator)
4. WordNet is considered to be the most important resource available to researchers in computational linguistics, text analysis (https://www.igi-global.com/dictionary/ontology-based–language-learning/32725)

Applications of lexical knowledge bases

1. Enhance usability of search engines
2. Writing evaluation and advice
3. Smarter tag clouds
4. Dating service
5. Job-seeker service
6. Taxonomy mapping

Unit7: Syntactic Analysis: POS Tagging

POS-tagging and Usage

1. Part-of-speech (POS) tagging is a quite simple concept. We identify the part of speech (noun, verb, adjective, etc.) associated to each word in a text.
2. Implementation : WordNet is an elementary resource to implement POS tagging, but WordNet has only four parts of speech, so it is a bit too fundamental for most useful purposes. There are several POS taggers of different complexity. The most widely used (currently) is Penn Treebank, which al- lows for forty-five tags. In addition to the basic four of WordNet, the Penn-Treebank set includes some additional distinctions to further refine noun terms (four noun distinctions), verb forms (six verb distinctions), and similarly for adjectives, adverbs, and also characterized functional words such as prepositions and articles.
3. How well does it work?-The Penn-Treebank provides approx. 93% accuracy, by some estimates. the best taggers in use achieve approx. 96% accuracy, and the best humans do not agree more definitively than that. so, automated tagging is at approximately the same level of accuracy as can be achieved even with expert human tagging.

Unit8: Syntactic Analysis: Parsing

Shallow parsing Using chunks- Depending upon the intended purpose of the parsing, an appropriate level of parsing can be chosen.

1. POS tagging is the shallowest parsing
2. Chunking is middle level parsing
3. Full grammar tree is full parse tree

Shallow parsing using chunks involves breaking the complete input sentence into small phrases which can be either noun, verb, etc.

Whatever syntactic elements we leave out of our chunks are called “chinks.”

Annotation for chunks: There is a standard annotation called “IOB”:

1. I a token is inside a chunk
2. O a token is outside any chunk (it’s a chink)
3. B a token begins a new chunk

Full grammar parsing- Full grammar parsing can be complex, as sentence structure (at least in English) provides possibility for highly complex structures. There are two types of full grammar parsing:

1. **Constituency parsers:** Constituency parsers break a sentence into subphrases and sub-sub-phrases, etc.
   Features of a typical constituency parse graph:
   a. Directed acyclic graph
   b. Root node is the sentence
   c. Leaf nodes are words
   d. Penultimate nodes are POS tags
   e. All other interior nodes are phrase tags
   f. Edges are unlabeled

2. **Dependency parsers**: Dependency parsers give us labelled relations between words—this can give us, for example, the subject and object of the main verb.
   Features of a typical dependency parse graph:
   a. Directed acyclic graph
   b. All nodes (beyond root) are words
   c. All edges are labelled
   d. Root node typically connects to the main verb

Creating full parse trees: one method is the CYK algorithm. the sentence structure is traversed searching for a match to pre-identified grammar rules. when a sequence
of POS-tagged words matches a grammar rule, then that provides tag for that portion of the sentence. the token sets are scanned at successively broader scope until (hopefully) the entire sentence has been contained within known grammar structures. python package pyStatParser includes a constituency parser implementing a variant of the CYK method. There are plenty of parsers out there, and here are a few suggestions:

1. Dependency parsing
   a. MST parser
   b. MALT parser
2. Constituency parsing
   a. Stanford parser
   b. Link grammar parser

Uses for full parse trees

Application of parse trees depends on the problem:

1. Text mining applications often use constituency parsers as a setup for information extraction.
2. Question-answering applications often use a dependency parser to validate answers.

If we combine a lexical KB* with syntax parsing, we can make improvements in:

1. Information extraction
2. Sentiment analysis

Unit9: Midterm

Unit10: Semantic Analysis: Semantic Relatedness

Semantic similarity: A concept which says that objects can be similar in many ways, across many different dimensions. Semantic similarity can consider similarity on the basis of: words; sense; text; taxonomy; frame; and context. There are various "dimensions" of semantic similarity:

- Word similarity
- Sense similarity
- Text similarity
- Taxonomy similarity
- Frame similarity
- Context similarity

Word similarity: There are two approaches used to measure word similarity:

- **Statistical word similarity**: It deals on identifying that how closely two words in a corpus are associated with each other. For determining the measurement of association, we can use following 3 ways:
    - **PPMI (positive pointwise mutual information):**
        - It is a measure of the relative distinction between dependent and independent word pairs. If a word is more likely to appear in combination more likely than it is to appear without the associated word, then there is a high dependency. A way to measure this is with PMI. positive PMI indicates that words in a pair are related. The way to express PMI is

$$PMI(w_i, w_j) = \log \frac{p(w_i, w_j)}{p(w_i)p(w_j)}$$

    - **Vector semantics and LSA (Latent semantic analysis):**
        - It is also known as distributional semantics - inspired by J.R. Firth" a word is characterized by the company it keeps". This approach makes use of context of the sentence which is used to judge word similarity as well as text similarity. A vector represents a distribution of other features found in the same context as each target word.
            - Term document matrix: The concept is two documents are considered similar if their vectors are similar.
            - Term content matrix: Rather than measure frequency of individual words, a term context matrix assembles a matrix of word counts and their near neighbors.
    - **Cosine similarity:**
        - Measures the cosine of angle between two vectors. The cosine similarity is the vector dot product and measures the Euclidean distance two vectors
    - **Jaccard similarity:**
        - Another measure of similarity / dissimilarity; Jac- card distance is the ration of the number of elements in two sets divided by the number of elements in either set, i.e., the intersection of the sets divided by the union of the sets. mathematically:

$$J(X, Y) = \frac{|X \cap Y|}{|X \cup Y|} * 100$$

- **Structural word similarity:** It deals with identifying that how close the two words in a corpus are within a semantic graph. For determining the closeness within semantic graph, we can use following 2 ways:
    - Ontological distance
    - Overlap of parse contexts

Unit11: Semantic Analysis: Document Clustering

It is an unsupervised machine learning technique because characteristics of groups are not available in advance and labelling is also not done. In document clustering, we organize a set of documents into groups having similar characteristics. In other words, we organize a corpus of documents into various groups based on some features/attributes/properties of the documents.

Methods of Clustering

There are many types of clustering methods, but two of the most common are:

- **Centroid-based methods**
  - The centroid -based methods typically involves k-means, k-medoid, "bisecting" k-means but typically the default is k-means.
  - Each cluster has a central representative member that represents each cluster and has features that distinguish that cluster from others.
  - For defining the distance metric, we use cosine, Jaccard, Manhattan etc.
- **Hierarchical methods**
  - This method creates a nested sequence of partitions, with a single, all-inclusive cluster at the top and singleton clusters of individual points at the bottom.
  - Each intermediate level can be viewed as combining two clusters from the next
  - Lower level (or splitting a cluster from the next higher level).
  - Resulting tree is called a **dendrogram**.
    - **Agglomerative Nesting-AGNES (bottom up)**: Start with the points as individual clusters and, at each step, merge the most similar or closest pair of clusters. This requires a definition of cluster similarity or distance.
    - **Divisive Analysis-DIANA (top down)**: Opposite to AGNES, this method starts with one, all-inclusive cluster and, at each step, split a cluster until only singleton clusters of individual points remain. In this case, we need to decide, at each step, which cluster to split and how to perform the split.
- Hierarchical clustering methods are usually "better" than centroid methods but slower. **(Quality vs efficiency)**
- Hierarchical clustering is often portrayed as the better-quality clustering approach but is limited because of its quadratic time complexity.
- K-means and its variants have a time complexity which is linear in the number of documents but are thought to produce inferior clusters.

Working with Clusters

After we are done with creating clusters. Now, we need to be able to visualize and label clusters and leverage them in applications.

The applications of clustering are:

- Search result clustering
  - More effective search engine results page
- Collection clustering
  - Alternative to search—browse/explore
- Cluster-based retrieval
  - Exemplar based: "more like this" or "more like these"

Unit12: Semantic Analysis: Text Classification

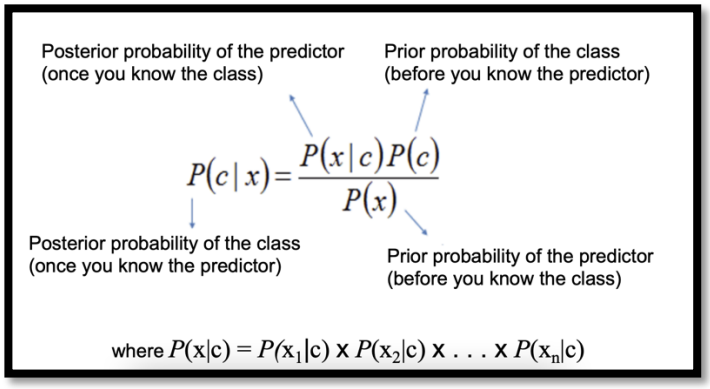
Document Classification

In contrast to clustering, document classification is : requires the classifications to be present in advance; it is a supervised learning method; requires pre-labelled groupings. considerations in document classification include

- **Content based vs. descriptor based classification**
  - In **content-based classification**, we start with two or more classes of existing content, where our task is to classify documents into the same categories. Most applications of content-based text classification involve the placement of documents into a binary classification, or into a preestablished subject-matter hierarchy.
    - An example of the binary classifier is a spam filter (or an offensive content filter).
    - An example of a multiclass application is the browsing taxonomy of broad content network (e.g., news website, blog network).
  - **In descriptor-based classification**, instead of example documents, there is a user-inputted description of the content desired. Applications of descriptor-based text classification can also be binary or multiclass.
    - Binary classification:
      - Legal discovery orders in court cases
      - FOIA* requests
    - Multiclassification
      - This would be, for example, the initial populating of taxonomies with documents.
      - This assumes a set of well-defined taxonomy nodes that lack pre-existing example documents.
- There are 2 approaches in descriptor-based classification:
  - Multinomial Naïve Bayes classifiers
  - SVM-based classifier

- **Naïve Bayes classification**

  Some important terminologies in **Naïve Bayes** models are as follows:
  - The ***class*** is what we're trying to predict from a set of attributes collectively termed the predictor.
  - The ***prior probability*** of the class is the global distribution of individuals into that class.
  - Likewise, the prior probability of the predictor is the global distribution of individuals into that predictor (items sharing that set of attributes).
  - The ***posterior probability*** of the predictor is the probability, just among items in the target class, of having the predictor attributes.
  - The ***posterior probability*** of the class is the probability, just among items sharing the predictor attributes, of falling into the target class.
  - What a Naïve Bayes classifier essentially does is calculate the posterior probability of the class.



- **SVMs for text classification**
    - Every SVM classifier is a binary classifier—but we can combine multiple SVMs to make a multiclass solution.
    - Ideally, there is a linear space within the margins of feature vector space, separating two classes. The possible demarcations between classes are called "hyperplanes."
    - **Kernel Trick:** A kernel is a method of placing a two-dimensional plane into a higher dimensional space, so that it is curved in the higher dimensional space. (In simple terms, a kernel is a function from the low dimensional space into a higher dimensional space.) A Kernel Trick is a simple method where a Non-Linear data is projected onto a higher dimension space so as to make it easier to classify the data where it could be linearly divided by a plane. This is mathematically achieved by Lagrangian formula using Lagrangian multipliers.

Unit13: Semantic Analysis: Topic Modelling

Topic Model Paradigms:

A topic is a:

- cluster of words that occur together and define a theme.
- Statistical language models for discovering hidden structure in a collection of documents.
- Algorithms that discover thematic structure in documents.
- A way to analyze large volumes of unlabeled text.

Topic modeling falls into a few broad categories:

- **Canonical**: It matches a preestablished list of domain-specific topics. E.g., Digital Music, Electronics, Home & Kitchen, Automotive
- **Organic:** Discover clusters from documents without establishing a list of topics beforehand
- **Entity- centric:** topics are related to a set of named entities in a domain. E.g., entities for a streaming movie service: actors, directors, producers, genre

Organic Topic Modeling

There are three ways to implement organic topic modelling:

- **LSA—latent semantic analysis**: LSA-based topic modeling tries to find groups of words associated with the largest variances between documents in the corpus. The steps to work with LSA are:
    - **Step 1:** Create a term-document matrix (using term count or TF-IDF): rows are unique words, columns represent documents
    - **Step 2:** Use Singular Value Decomposition (SVD) to reduce number of rows but preserve column similarity because the matrix can be very sparse and noisy
    - **Step 3:** Compare documents by cosine similarity between two vectors formed by two columns

- **LDA—latent Dirichlet allocation:** LDA construes topics as groups of words that have high co- occurrences among different documents in the corpus. Different topics can share keywords when those keywords co-occur frequently enough across the different topics. The algorithm for the LDA implementation is shown below:

1. Initialize the necessary parameters.
2. For each document, randomly initialize each word to one of the K topics.
3. Start an iterative process as follows and repeat it several times.
4. For each document D:
    a. For each word W in document:
        - For each topic T:
            - Compute $P(T|D)$, which is proportion of words in D assigned to topic T.
            - Compute $P(W|T)$, which is proportion of assignments to topic T over all documents having the word W.
        - Reassign word W with topic T with probability $P(T|D) \times P(W|T)$ considering all other words and their topic assignments.

LDA is akin to LSA except in the way that it utilizes probability distributions over words, rather than a topic-topic matrix, to guide the assignment of words to topics.

- **NMF—non-negative matrix factorization:** For all practical purposes, you can view NMF as a version of LDA in which the parameters have been tweaked to enforce a sparse number of topics. Another way to say it, is that NMF naturally produces sparse representations. The algorithm is as below:

- Standard random initialisation of NMF factors can lead to *instability* - i.e. significantly different results for different runs on the same data matrix.
- *NNDSVD*: Nonnegative Double Singular Value Decomposition (Boutsidis & Gallopoulos, 2008):
    - Provides a deterministic initialization with no random element.
    - Chooses initial factors based on positive components of the first *k* dimensions of SVD of data matrix **A**.
    - Often leads to significant decrease in number of NMF iterations required before convergence.

NMF is usually much cheaper in computation than LDA. NMF works better "out-of-the-box" on noisy texts (does not require as much tuning as LDA, in such cases). NMF often works better on very small corpora than either LSA or LDA.

Canonical Topic Modeling

The goal of canonical topic modelling is to determine a subset of canonical topics that are materially treated in a given corpus, showing which topics are contextually related in that corpus. The driving concern is usually to enable topic-driven exploration of the corpus by end users.
There are two basic options how to proceed:

- Constrain the organic topic model to the canonical list of topics.
- Use an IR approach, leveraging the canonical topic list to build queries, and analyze the hits.
    - When we see that two topic words used as queries share a lot of the same hits, it shows us that those topics are related—this is the **extensional** approach.
    - Topics that have close semantic distance are also related—this is the **intensional** approach.

Entity-Centric Topic Modeling

Your mission is to model whatever topics are strongly related to a set of named entities in a domain.

- Examples are websites that aggregate news on one of these: NFL, MLB, NBA, NHL, FIFA, etc.
- Also, news aggregators for movies, TV series, music groups, etc.

Applications of Topic Modelling

- Movie recommender
- News article recommender
- Book recommender
- Dating-website match recommender

Unit14: Semantic Analysis: Sentiment and Rhetoric

General Sentiment Analysis: Sentiment analysis is also known as opinion analysis, or opinion mining. Unstructured textual data has facts (objective) and opinions (subjective)

The basic scoring ranges between -1 to +1.

0 represents either

- neutral sentiment,
- no detection of sentiment,
- balanced net sentiment

There are 2 approaches for performing sentiment analysis

- **Supervised ML approaches:** The outline of applying the supervised ML approach to sentiment analysis is as follows:
    - Establish a training set.
    - Normalize texts.
    - Extract feature vectors.
    - Train a binary classifier, probably SVM.
    - After QA, decide if more training data is needed.

Advantages: Quick to implement when training data is ready-to-hand and other is- we don't need to develop a coded vocabulary

Disadvantages: It is not considered an explainable AI (XAI)

- **Unsupervised lexical KB approach:** The lexical approach uses existing sentiment vocabularies to support the analysis. The steps involved in sentiment analysis with lexical knowledge base are as:
    - Establish valence-weighted vocabularies.
    - Normalize texts
    - Extract feature vectors.
    - Execute a scoring algorithm.
    - After QA, tweak vocabulary and rerun until it passes QA.

    There is a need to maintain a tagged lexicon, and the difficulty to manage an up-to-date lexicon in the face of changing vocabulary and even changing nuances/sentiments associated to existing vocabulary. choices for established lexicons include:

    - **AFINN** - Affective Lexicon by Finn Nielsen, which includes approx. **2500** clues
    - **Liu's** lexicon - **6800** clues
    - **MPQA** (Multi-Perspective Question Answering) - subjectivity lexicon: **8,222** clues
    - **SentiWordNet**: labels all **100k+** WordNet synsets !! (autogenerated)
    - **VADER**(Valence Aware Dictionary for sEntiment Reasoning):**7,500** clues
    - Pattern library lexicon (**2,912** clues, mostly adjectives, with IDs from WordNet, valences hand coded or inferred from a nearest neighbour)
    - **Custom** - as you choose

Advantages: Lexical approach does not require training data to get started and it is eminently explainable (good XAI)

Disadvantages: It needs a coded vocabulary (lexical knowledge base) and it can be cumbersome to maintain in the face of new tropes

Advanced Sentiment Analysis: It includes working on complex emotions such as sadness, happiness, anger, and sarcasm. The statement can be either simple, negation or modifiers.

There are 3 ways to improve the score for general sentiment analysis done. They are as follows:

- Determining referents and/or topics to which sentiment attaches
    - A straightforward approach is just to run a chunker, and send NP-chunks and VP-chunks, instead of sentences, into the sentiment analyzer.

- Classifying sentiment into more categories than just negative and positive
- Picking up on non-sentiment vocabulary differences that align with sentiment around a topic, e.g., how rhetoric aligns with sentiment on an issue

Unit15: Final