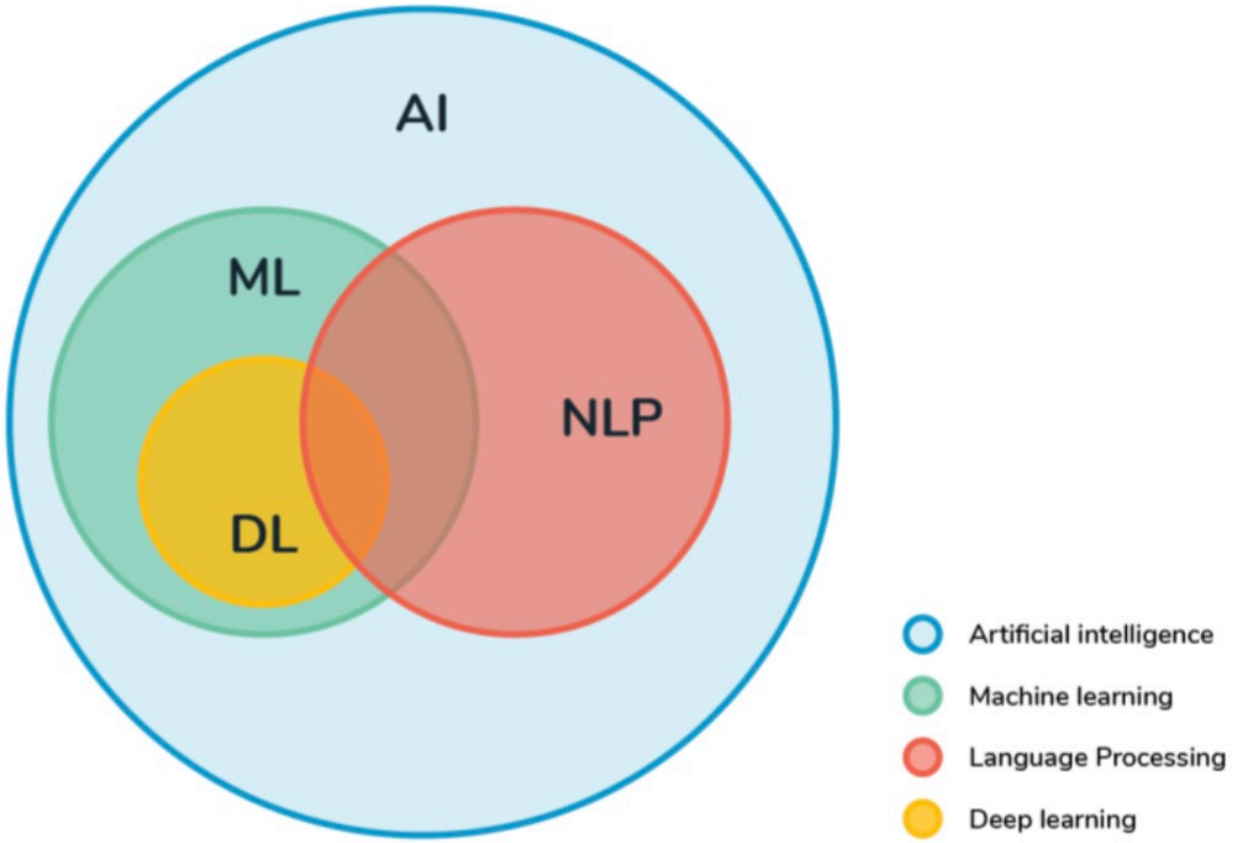
Natural Language Processing

Session-1

- Introduction to NLP
- NLP Theory

Introduction to NLP

What is Natural Language Processing (NLP) ?



NLP refers to the methods that enable the human language to be understood by the machine with some algorithms, models and techniques.

Introduction to NLP

Why Do We Need NLP?

Internet is full of user generated data

- Emails
- Chat logs
- Documents
- Employee Reviews
- etc...

Data is commonly preserved with text

We have to use this kind of DATA!!!

Introduction to NLP

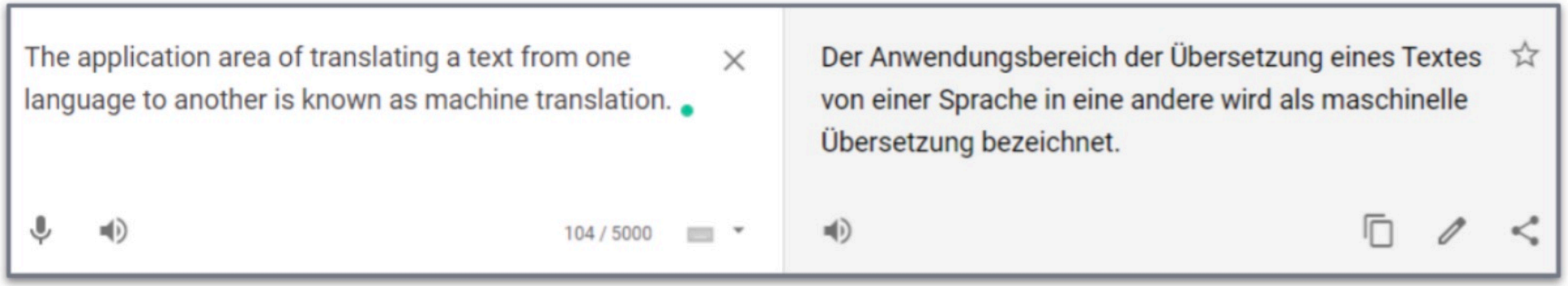
What are the main usage areas of the NLP ?

- Diagnosing (Healthcare)
- Sentiment Analysis
- ChatBot
- Machine translation
- Digital Assistants

Introduction to NLP

Some applications of NLP

Machine Translation



Source: bootcamp.DataJarLabs.com

Introduction to NLP

Some applications of NLP

Digital Assistants

First, these assistants need to convert our voices into the words. This task is known as **speech recognition.**

After these assistants turn our speeches into the text, the next stage is to apply NLP techniques to understand what we mean and rest their actions upon our intends.

AMAZON ECHO

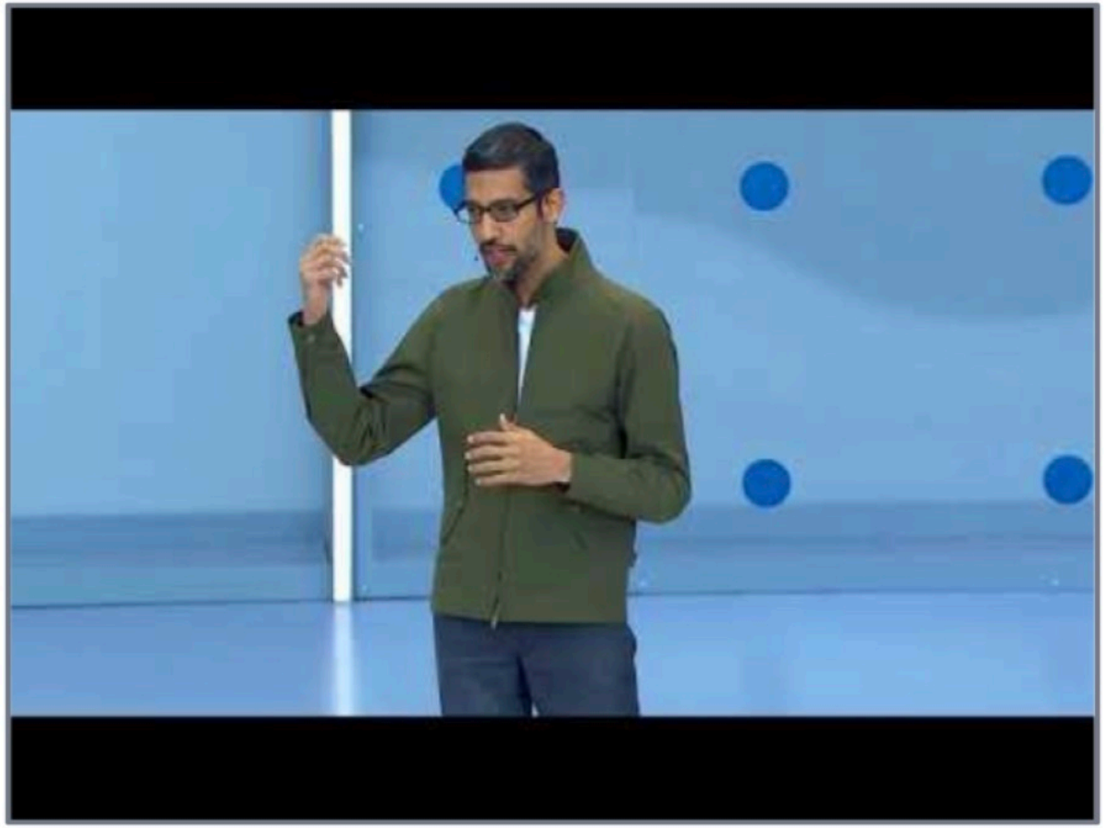
GOOGLE HOME

Source: bootcamp.DataJarLabs.com

Introduction to NLP

Some applications of NLP

To get an idea of where the **capabilities of digital assistants** reached, watch the conversation between Google Assistant and a real hair saloon where Google Assistant makes a hair-cut appointment on behalf of someone



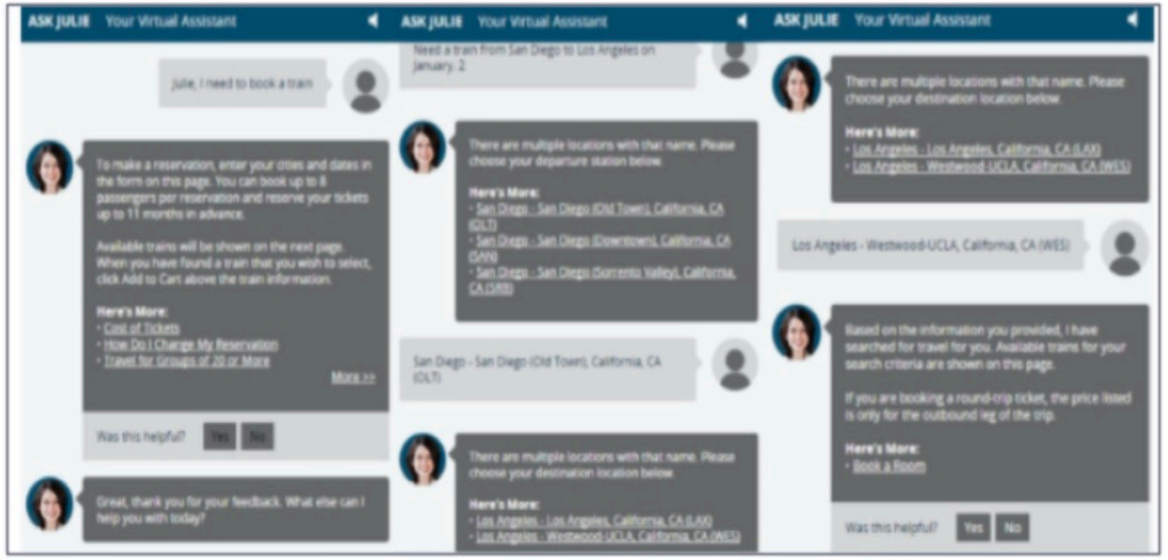
https://www.youtube.com/watch?v=D5VN56jQMWM

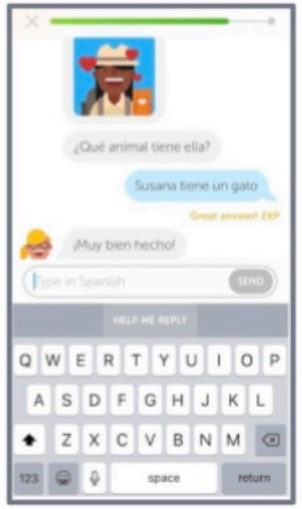
Source: bootcamp.DataJarLabs.com

Introduction to NLP

Some applications of NLP

Chat Bots

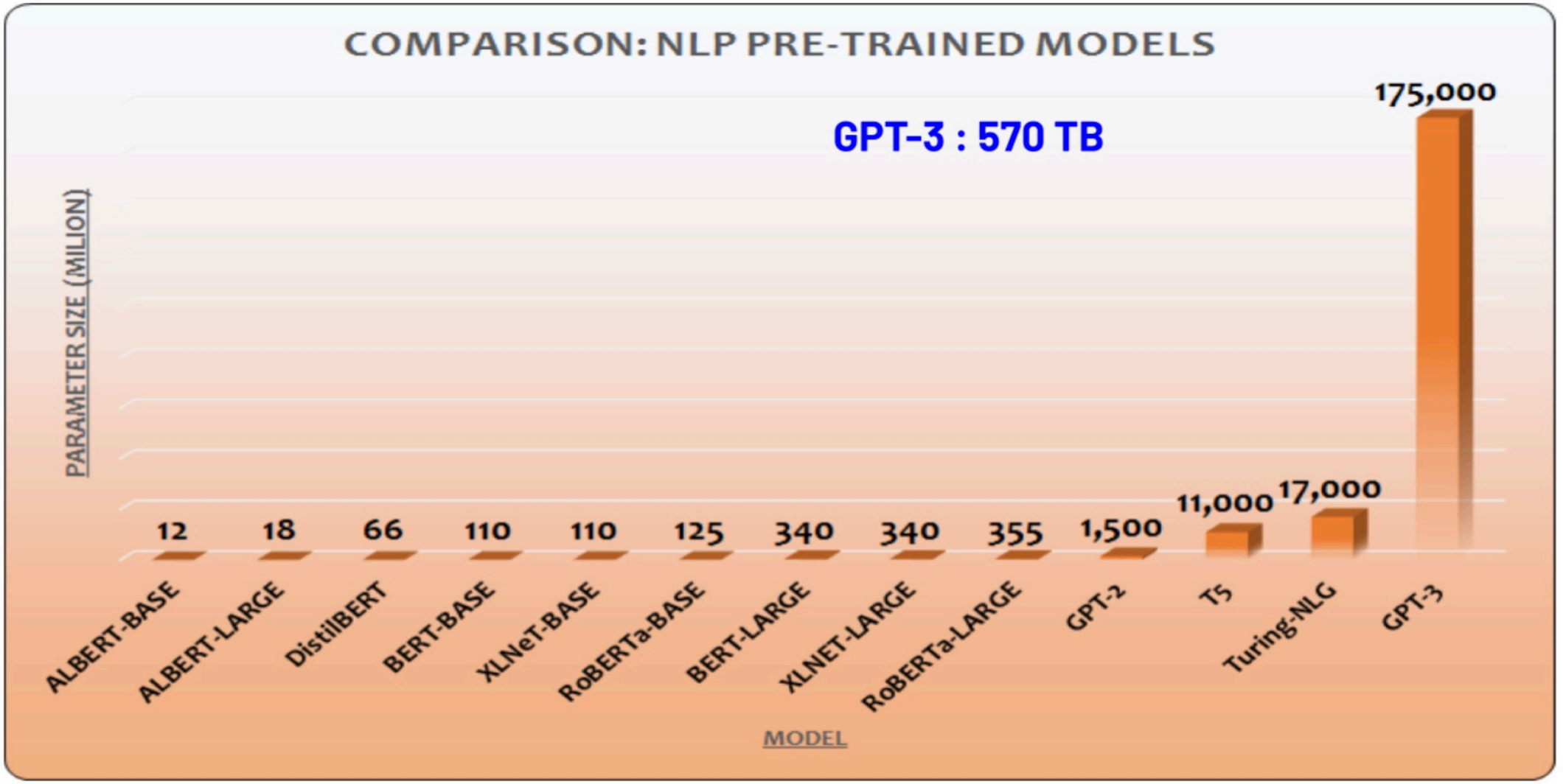




Source: bootcamp.DataJarLabs.com

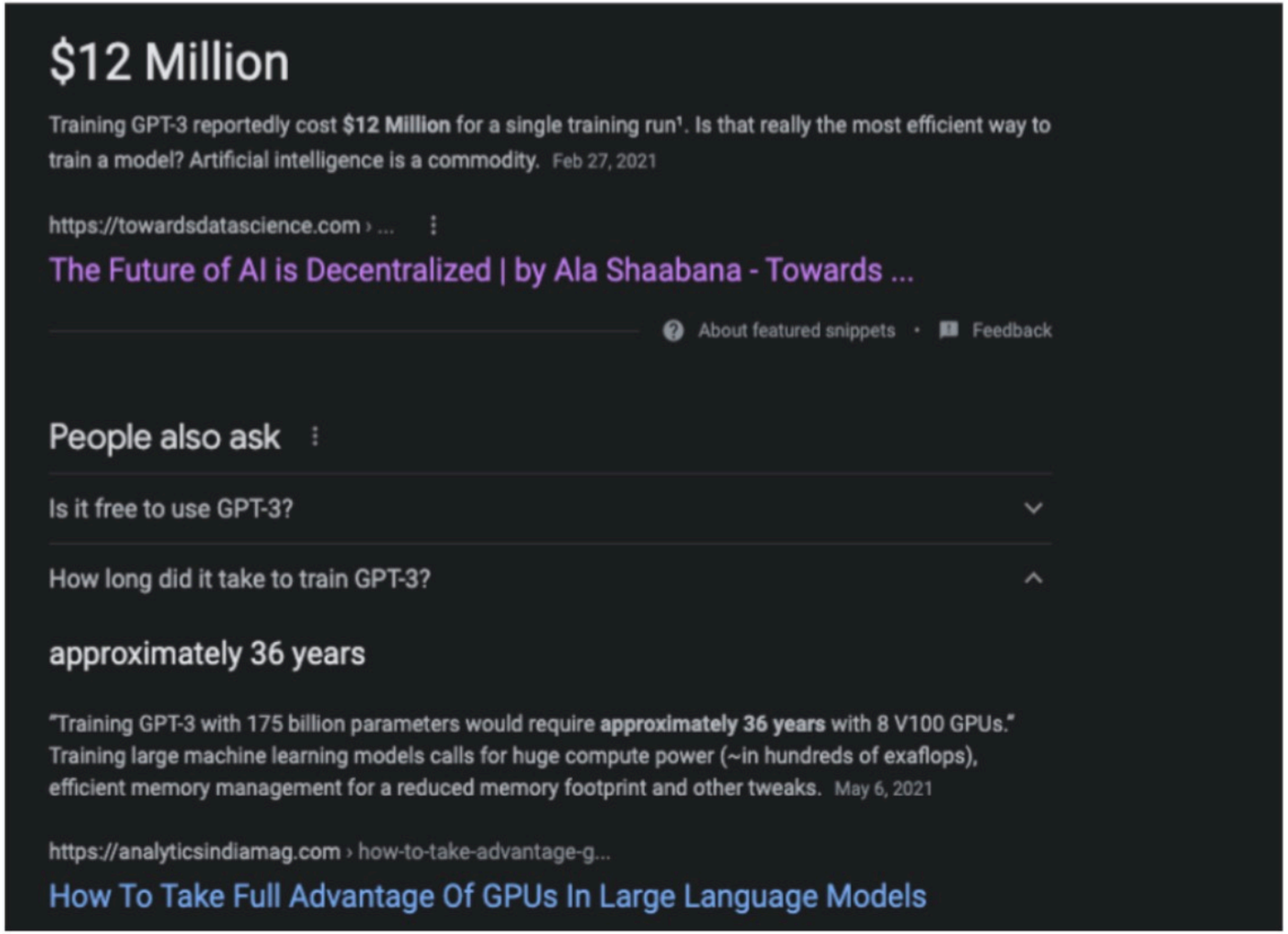
Introduction to NLP

Some applications of NLP (GPT-3)



Introduction to NLP

Some applications of NLP (GPT-3)

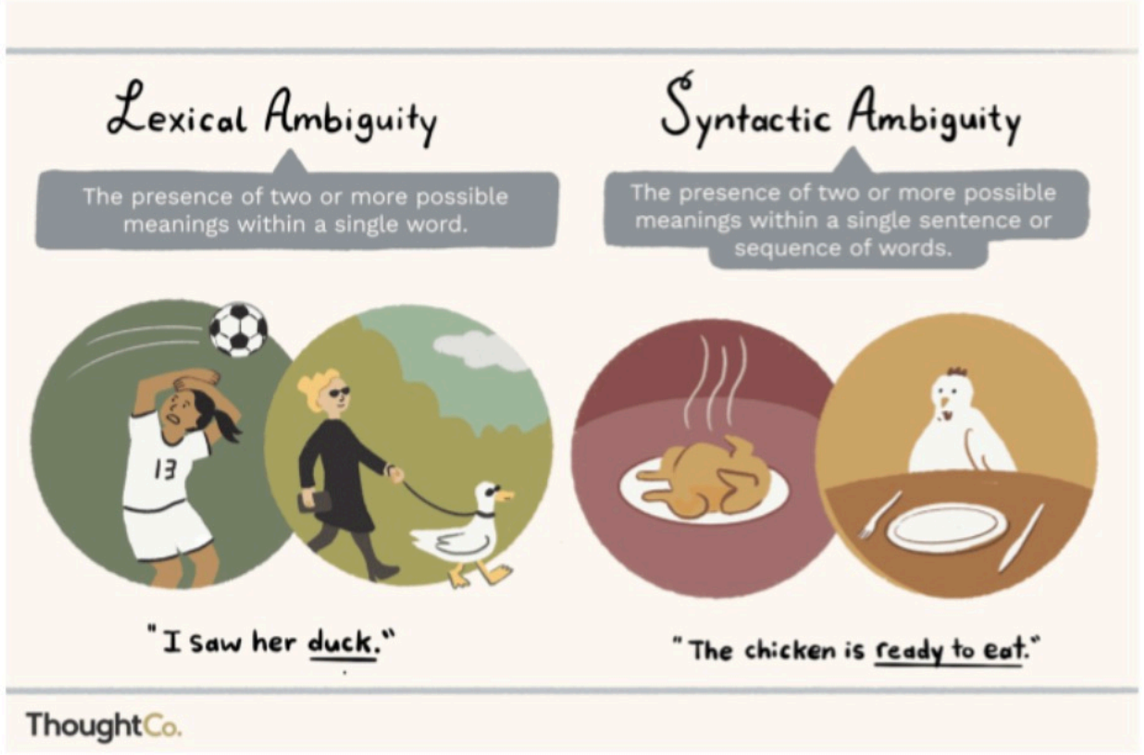


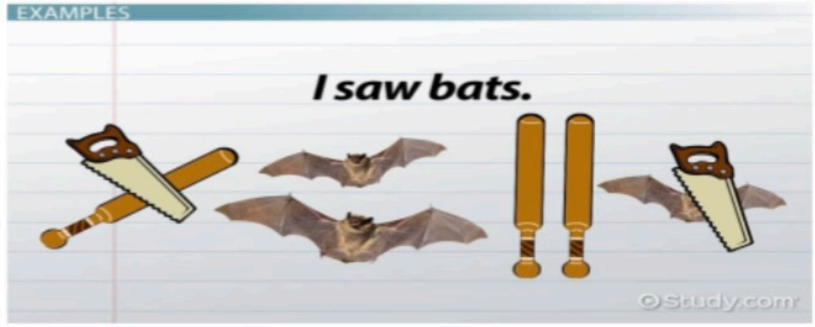
Introduction to NLP

Some Challenges of NLP

Ambiguity

The main challenge of NLP is the understanding and modeling of elements within a variable context. In a natural language, words are unique but can have different meanings depending on the context resulting in ambiguity on the lexical, syntactic, and semantic levels.





Source: bootcamp.DataJarLabs.com

Introduction to NLP

Some Challenges of NLP

Synonymy

We can express the same idea with different terms which are also dependent on the specific context: ***big*** and ***large*** can be synonyms when describing an object or a building but they are not interchangeable in all contexts, e.g. ***big*** can mean ***older***, *grown up* in phrases like *big sister*; while *large* does not have this meaning and could not be substituted here.

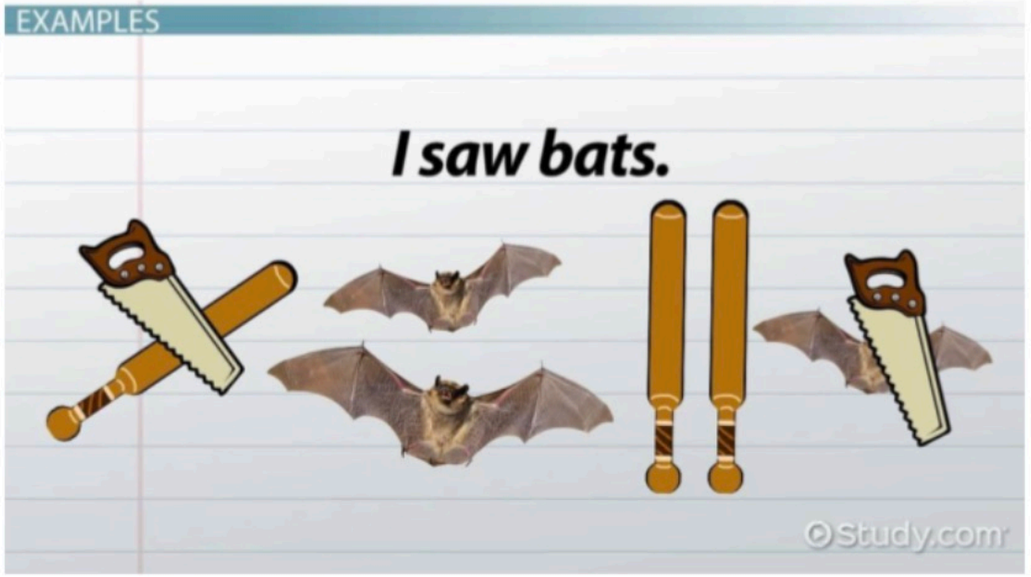
Source: bootcamp.DataJarLabs.com

Introduction to NLP

Some Challenges of NLP

Polysemy

The same word might have multiple different meanings.



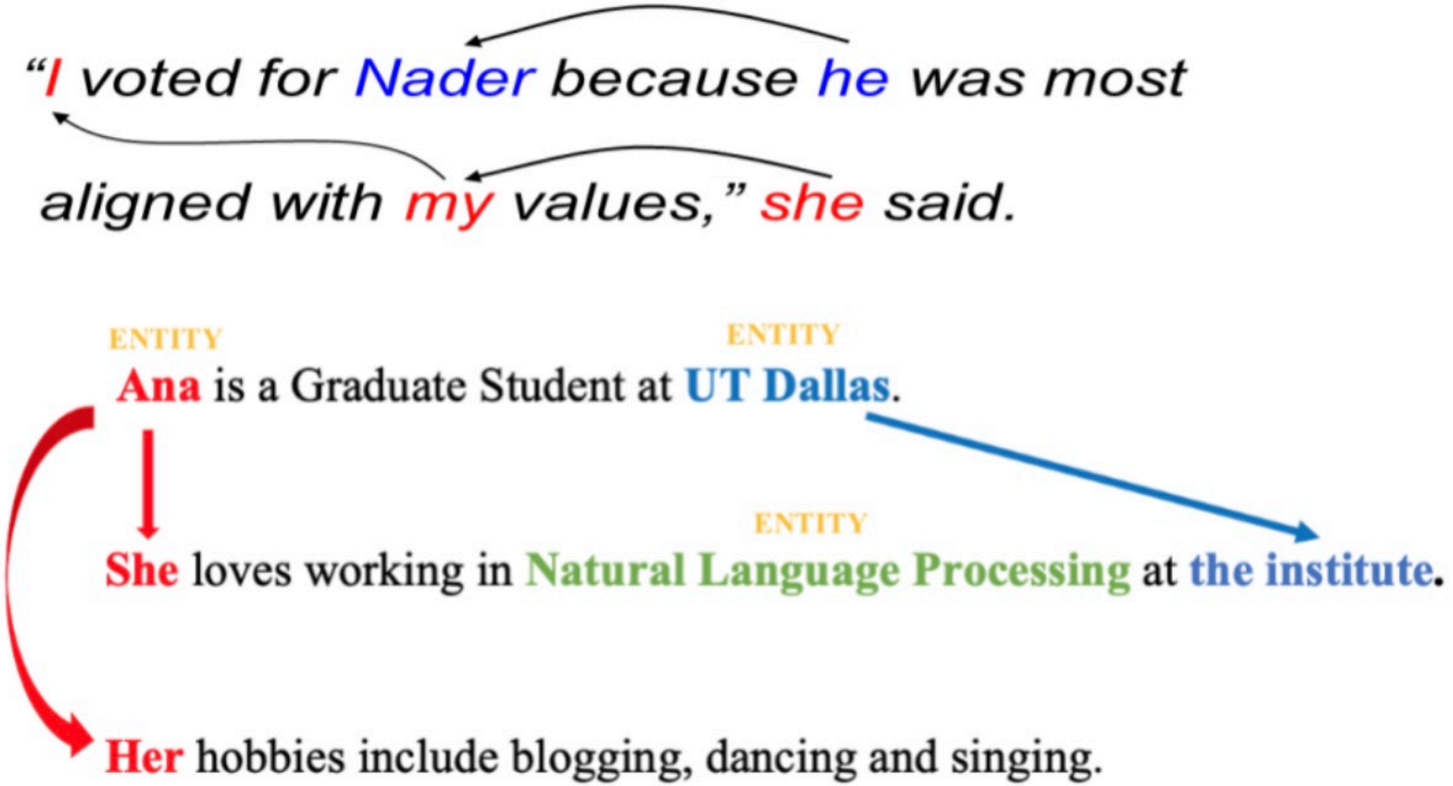
Source: bootcamp.DataJarLabs.com

Introduction to NLP

Some Challenges of NLP

Coreference

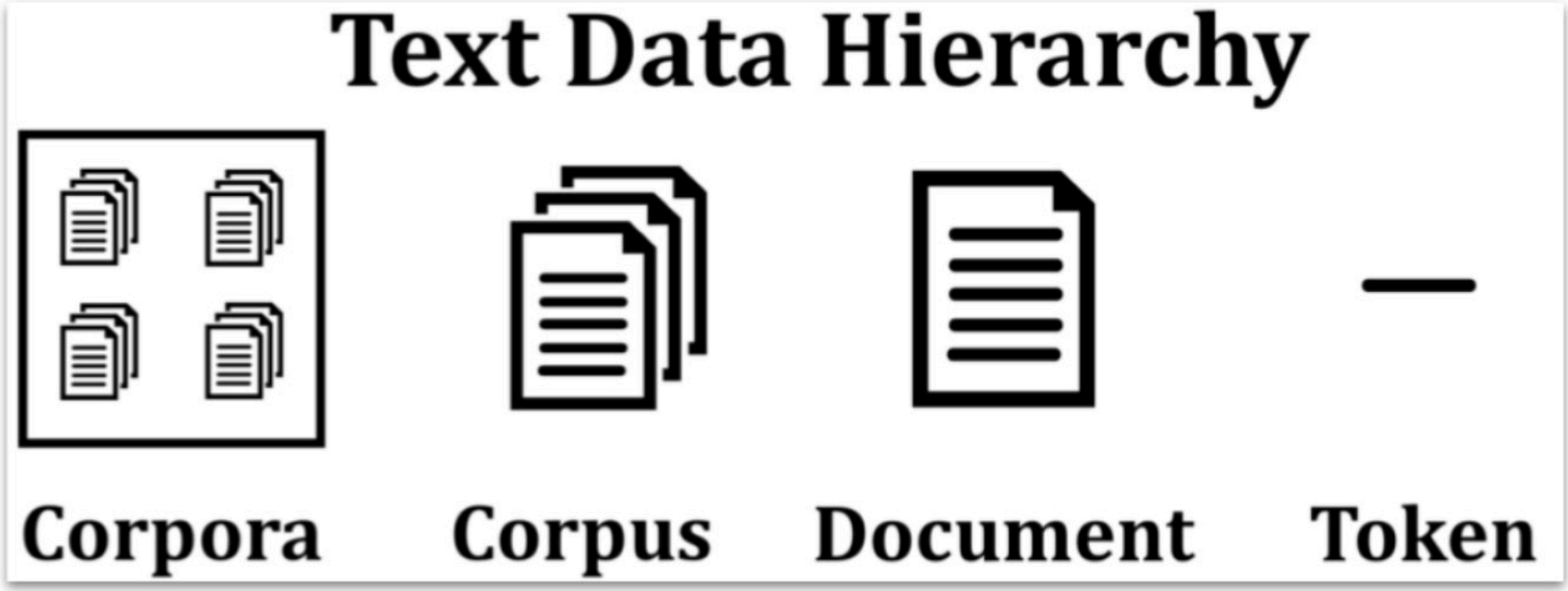
The process of finding all expressions that refer to the same entity in a text is called coreference resolution. It is an important step for a lot of higher-level NLP tasks that involve natural language understanding such as document summarization, question answering, and information extraction.



Source: bootcamp.DataJarLabs.com

NLP Theory-Basic Consepts

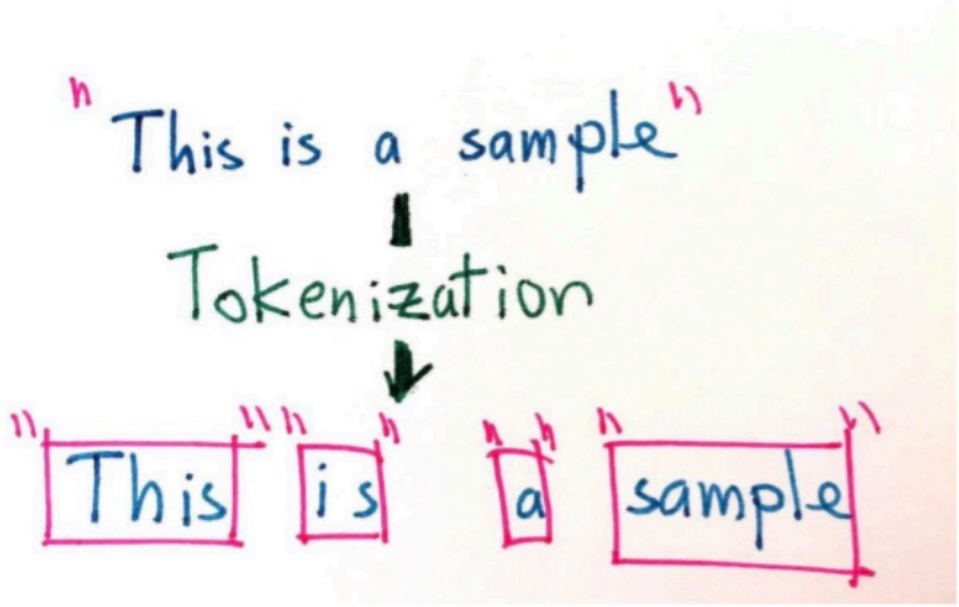
Document & Corpus



Example: You are implementing some NLP tasks using a book. Let's consider each paragraph of the book is a row in your dataset. The book itself is the corpus and each paragraph is a document.

NLP Theory-Cleaning

Tokenization



We can count the number of words in the text after tokenization.

Tokenization means splitting a sentence, paragraph, phrase, or an entire text document into smaller units, such as individual words or terms. You can consider each word as a token.

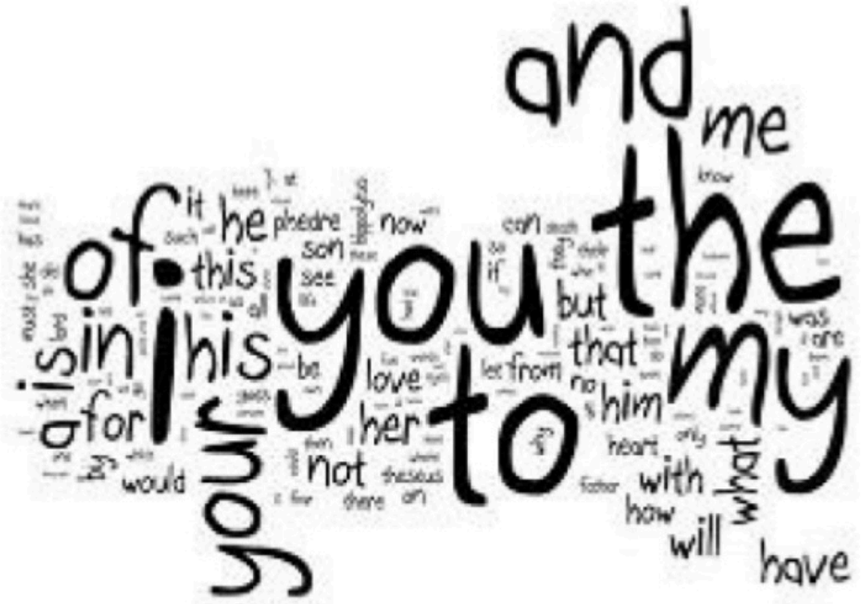
NLP Theory-Cleaning

Removing Punctuation

We cannot feed a machine learning model from raw text. We need to clean the text first. Removing the punctuation is usually one of the first steps of cleaning the text. Because the model does not need them.

NLP Theory-Cleaning

Removing Stopwords



Stopwords do not contribute to the meaning of the text deeply. These words introduce much noise because they appear more frequently than other words. We filter out these stopwords before doing any statistical analysis or creating a model.

NLP Theory-Cleaning

Removing Stopwords

['i', 'me', 'my', 'myself', 'we', 'our', 'ours', 'ourselves', 'you', "you're", "you've", "you'll", "you'd", 'your', 'yours', 'yourself', 'yourselves', 'he', 'him', 'his', 'himself', 'she', "she's", 'her', 'hers', 'herself', 'it', "it's", 'its', 'itself', 'they', 'them', 'their', 'theirs', 'themselves', 'what', 'which', 'who', 'whom', 'this', 'that', "that'll", 'these', 'those', 'am', 'is', 'are', 'was', 'were', 'be', 'been', 'being', 'have', 'has', 'had', 'having', 'do', 'does', 'did', 'doing', 'a', 'an', 'the', 'and', 'but', 'if', 'or', 'because', 'as', 'until', 'while', 'of', 'at', 'by', 'for', 'with', 'about', 'against', 'between', 'into', 'through', 'during', 'before', 'after', 'above', 'below', 'to', 'from', 'up', 'down', 'in', 'out', 'on', 'off', 'over', 'under', 'again', 'further', 'then', 'once', 'here', 'there', 'when', 'where', 'why', 'how', 'all', 'any', 'both', 'each', 'few', 'more', 'most', 'other', 'some', 'such', 'no', 'nor', 'not', 'only', 'own', 'same', 'so', 'than', 'too', 'very', 's', 't', 'can', 'will', 'just', 'don', "don't", 'should', "should've", 'now', 'd', 'll', 'm', 'o', 're', 've', 'y', 'ain', 'aren', "aren't", 'couldn', "couldn't", 'didn', "didn't", 'doesn', "doesn't", 'hadn', "hadn't", 'hasn', "hasn't", 'haven', "haven't", 'isn', "isn't", 'ma', 'mightn', "mightn't", 'mustn', "mustn't", 'needn', "needn't", 'shan', "shan't", 'shouldn', "shouldn't", 'wasn', "wasn't", 'weren', "weren't", 'won', "won't", 'wouldn', "wouldn't"]

List of NLTK library stopwords for English Language (179 words)

Source:
https://medium.com/@saitejaponugoti/stop-words-in-nlp-5b248dadad47

NLP Theory-Cleaning

Removing Stopwords

['whence', 'here', 'show', 'were', 'why', 'n't', 'the', 'whereupon', 'not', 'more', 'how', 'eight', 'indeed', 'i', 'only', 'via', 'nine', 're', 'themselves', 'almost', 'to', 'already', 'front', 'least', 'becomes', 'thereby', 'doing', 'her', 'together', 'be', 'often', 'then', 'quite', 'less', 'many', 'they', 'ourselves', 'take', 'its', 'yours', 'each', 'would', 'may', 'namely', 'do', 'whose', 'whether', 'side', 'both', 'what', 'between', 'toward', 'our', 'whereby', "'m", 'formerly', 'myself', 'had', 'really', 'call', 'keep', "'re", 'hereupon', 'can', 'their', 'eleven', "'m", 'even', 'around', 'twenty', 'mostly', 'did', 'at', 'an', 'seems', 'serious', 'against', "n't", 'except', 'has', 'five', 'he', 'last', "'ve", 'because', 'we', 'himself', 'yet', 'something', 'somehow', "'m", 'towards', 'his', 'six', 'anywhere', 'us', "'d", 'thru', 'thus', 'which', 'everything', 'become', 'herein', 'one', 'in', 'although', 'sometime', 'give', 'cannot', 'besides', 'across', 'noone', 'ever', 'that', 'over', 'among', 'during', 'however', 'when', 'sometimes', 'still', 'seemed', 'get', "'ve", 'him', 'with', 'part', 'beyond', 'everyone', 'same', 'this', 'latterly', 'no', 'regarding', 'elsewhere', 'others', 'moreover', 'else', 'back', 'alone', 'somewhere', 'are', 'will', 'beforehand', 'ten', 'very', 'most', 'three', 'former', "'re", 'otherwise', 'several', 'also', 'whatever', 'am', 'becoming', 'beside', "'s", 'nothing', 'some', 'since', 'thence', 'anyway', 'out', 'up', 'well', 'it', 'various', 'four', 'top', "'s", 'than', 'under', 'might', 'could', 'by', 'too', 'and', 'whom', "'ll", 'say', 'therefore', "'s", 'other', 'throughout', 'became', 'your', 'put', 'per', "'ll", 'fifteen', 'must', 'before', 'whenever', 'anyone', 'without', 'does', 'was', 'where', 'thereafter', "'d", 'another', 'yourselves', 'n't", 'see', 'go', 'wherever', 'just', 'seeming', 'hence', 'full', 'whereafter', 'bottom', 'whole', 'own', 'empty', 'due', 'behind', 'while', 'onto', 'wherein', 'off', 'again', 'a', 'two', 'above', 'therein', 'sixty', 'those', 'whereas', 'using', 'latter', 'used', 'my', 'herself', 'hers', 'or', 'neither', 'forty', 'thereupon', 'now', 'after', 'yourself', 'whither', 'rather', 'once', 'from', 'until', 'anything', 'few', 'into', 'such', 'being', 'make', 'mine', 'please', 'along', 'hundred', 'should', 'below', 'third', 'unless', 'upon', 'perhaps', 'ours', 'but', 'never', 'whoever', 'fifty', 'any', 'all', 'nobody', 'there', 'have', 'anyhow', 'of', 'seem', 'down', 'is', 'every', "'ll", 'much', 'none', 'further', 'me', 'who', 'nevertheless', 'about', 'everywhere', 'name', 'enough', "'d", 'next', 'meanwhile', 'though', 'through', 'on', 'first', 'been', 'hereby', 'if', 'move', 'so', 'either', 'amongst', 'for', 'twelve', 'nor', 'she', 'always', 'these', 'as', "'ve", 'amount', "'re", 'someone', 'afterwards', 'you', 'nowhere', 'itself', 'done', 'hereafter', 'within', 'made', 'ca', 'them']

List of spaCy library stopwords for English Language (326 words)

Source:
https://medium.com/@saitejaponugoti/stop-words-in-nlp-5b248dadad47

NLP Theory-Cleaning

Removing Stopwords

frozenset({'her', 'during', 'among', 'thereafter', 'only', 'hers', 'in', 'none', 'with', 'un', 'put', 'hence', 'each', 'would', 'have', 'to', 'itself', 'that', 'seeming', 'hereupon', 'someone', 'eight', 'she', 'forty', 'much', 'throughout', 'less', 'was', 'interest', 'elsewhere', 'already', 'whatever', 'or', 'seem', 'fire', 'however', 'keep', 'detail', 'both', 'yourselves', 'indeed', 'enough', 'too', 'us', 'wherein', 'himself', 'behind', 'everything', 'part', 'made', 'thereupon', 'for', 'nor', 'before', 'front', 'sincere', 'really', 'than', 'alone', 'doing', 'amongst', 'across', 'him', 'another', 'some', 'whoever', 'four', 'other', 'latterly', 'off', 'sometime', 'above', 'often', 'herein', 'am', 'whereby', 'although', 'who', 'should', 'amount', 'anyway', 'else', 'upon', 'this', 'when', 'we', 'few', 'anywhere', 'will', 'though', 'being', 'fill', 'used', 'full', 'thru', 'call', 'whereafter', 'various', 'has', 'same', 'former', 'whereas', 'what', 'had', 'mostly', 'onto', 'go', 'could', 'yourself', 'meanwhile', 'beyond', 'beside', 'ours', 'side', 'our', 'five', 'nobody', 'herself', 'is', 'ever', 'they', 'here', 'eleven', 'fifty', 'therefore', 'nothing', 'not', 'mill', 'without', 'whence', 'get', 'whither', 'then', 'no', 'own', 'many', 'anything', 'etc', 'make', 'from', 'against', 'ltd', 'next', 'afterwards', 'unless', 'while', 'thin', 'beforehand', 'by', 'amoungst', 'you', 'third', 'as', 'those', 'done', 'becoming', 'say', 'either', 'doesn', 'twenty', 'his', 'yet', 'latter', 'somehow', 'are', 'these', 'mine', 'under', 'take', 'whose', 'others', 'over', 'perhaps', 'thence', 'does', 'where', 'two', 'always', 'your', 'wherever', 'became', 'which', 'about', 'but', 'towards', 'still', 'rather', 'quite', 'whether', 'somewhere', 'might', 'do', 'bottom', 'until', 'km', 'yours', 'serious', 'find', 'please', 'hasnt', 'otherwise', 'six', 'toward', 'sometimes', 'of', 'fifteen', 'eg', 'just', 'a', 'me', 'describe', 'why', 'an', 'and', 'may', 'within', 'kg', 'con', 're', 'nevertheless', 'through', 'very', 'anyhow', 'down', 'nowhere', 'now', 'it', 'cant', 'de', 'move', 'hereby', 'how', 'found', 'whom', 'were', 'together', 'again', 'moreover', 'first', 'never', 'below', 'between', 'computer', 'ten', 'into', 'see', 'everywhere', 'there', 'neither', 'every', 'couldnt', 'up', 'several', 'the', 'i', 'becomes', 'don', 'ie', 'been', 'whereupon', 'seemed', 'most', 'noone', 'whole', 'must', 'cannot', 'per', 'my', 'thereby', 'so', 'he', 'name', 'co', 'its', 'everyone', 'if', 'become', 'thick', 'thus', 'regarding', 'didn', 'give', 'all', 'show', 'any', 'using', 'on', 'further', 'around', 'back', 'least', 'since', 'anyone', 'once', 'can', 'bill', 'hereafter', 'be', 'seems', 'their', 'myself', 'nine', 'also', 'system', 'at', 'more', 'out', 'twelve', 'therein', 'almost', 'except', 'last', 'did', 'something', 'besides', 'via', 'whenever', 'formerly', 'cry', 'one', 'hundred', 'sixty', 'after', 'well', 'them', 'namely', 'empty', 'three', 'even', 'along', 'because', 'ourselves', 'such', 'top', 'due', 'inc', 'themselves'})

List of gensim library stopwords for English Language (337 words)

Source:
https://medium.com/@saitejaponugoti/stop-words-in-nlp-5b248dadad47

NLP Theory-Cleaning

Stemming and Lemmatization

Original	Stemming	Lemmatization
New	New	New
York	York	York
is	is	be
the	the	the
most	most	most
densely	dens	densely
populated	popul	populated
city	citi	city
in	in	in
the	the	the
United	Unite	United
States	State	States

Stemming is a straightforward normalization technique, most often implemented as a series of rules that are progressively applied to a word to produce a normalized form.

We can think of lemmatization as a more sophisticated version of stemming. It reduces each word to its proper base form, that is, a word that we can find in a dictionary.

Source: https://www.baeldung.com/cs/stemming-vs-lemmatization

NLP Theory-Cleaning

Example

```
sample_text="Oh man, this is pretty cool. We will do more such things."
```

```
tokenization:
 ['oh', 'man', ',', 'this', 'is', 'pretty', 'cool', '.', 'we', 'will', 'do', 'more', 'such', 'things', '.']

removing punctation:
 ['oh', 'man', 'this', 'is', 'pretty', 'cool', 'we', 'will', 'do', 'more', 'such', 'things']

removing stopwords:
 ['oh', 'man', 'pretty', 'cool', 'things']

stemming
 ['oh', 'man', 'pretti', 'cool', 'thing']

lemmatization instead of stemming
 ['oh', 'man', 'pretty', 'cool', 'thing']
```

Source:
https://medium.com/@saitejaponugoti/stop-words-in-nlp-5b248dadad47

NLP Theory-Vectorization

Count Vectors (CountVectorizer)

TF-IDF Vectors (TfidfVectorizer)

Word Embedding (Word2Vec&GloVe)

Word vectorization is a methodology in NLP to map words from vocabulary to a corresponding numeric vector. In other words, it is the process of converting words into numbers.

NLP Theory-Vectorization

Count Vectors (CountVectorizer)

```
Document-1: John likes to watch movies. Mary likes movies too
```

```
Document-2: Mary also likes to watch football games
```

The CountVectorizer will convert a collection of text documents to a matrix of token counts. We can imagine this as a 2-Dimensional matrix. Where the 1-dimension is the entire vocabulary (1 column per word) and the other dimension is the actual documents, in this case, a row per document.

NLP Theory-Vectorization

Count Vectors (CountVectorizer)

Document-1 : John likes to watch movies. Mary likes movies too.

Document-2 : Mary also likes to watch football games.

unique tokens:	**John**	**likes**	**to**	**watch**	**movies**	**.**	**Mary**	**too**	**also**	**football**	**games**

cleaned tokens:	**john**	**likes**	**~~to~~**	**watch**	**movies**	**.**	**mary**	**~~too~~**	**~~also~~**	**football**	**games**

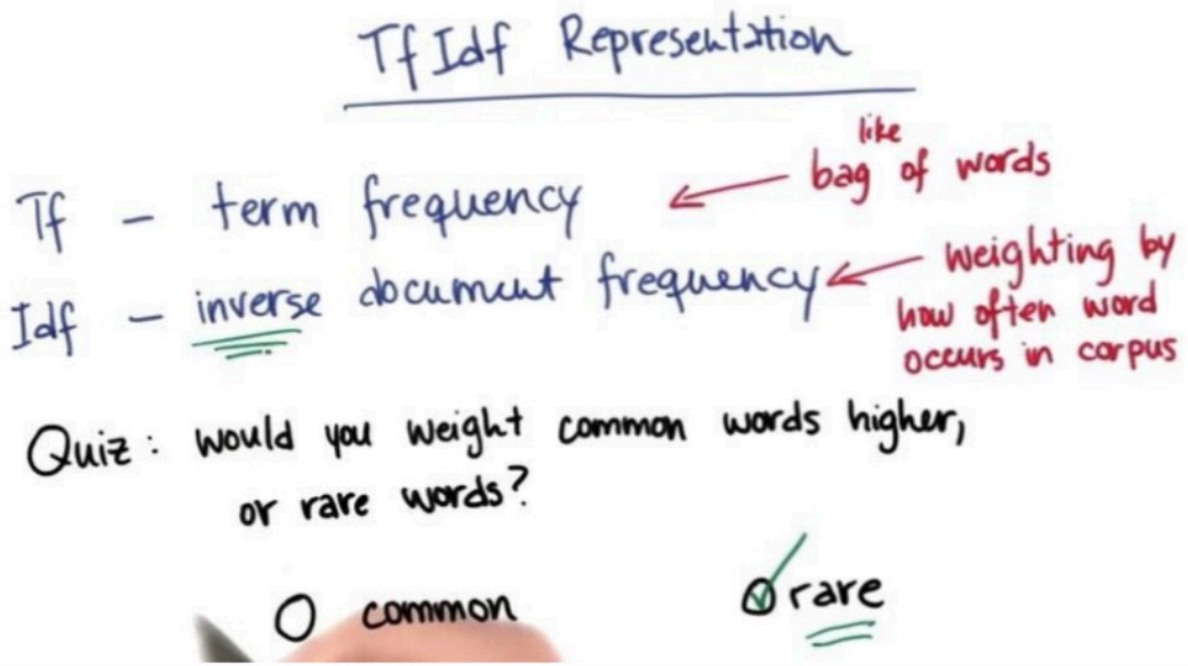
index	**football**	**game**	**john**	**like**	**mary**	**movie**	**watch**
1	0	0	1	2	1	2	1
2	1	1	0	1	1	0	1

Vectors of Each Document

vector-1	0	0	1	2	1	2	1
vector-2	1	1	0	1	1	0	1

NLP Theory-Vectorization

TF-IDF

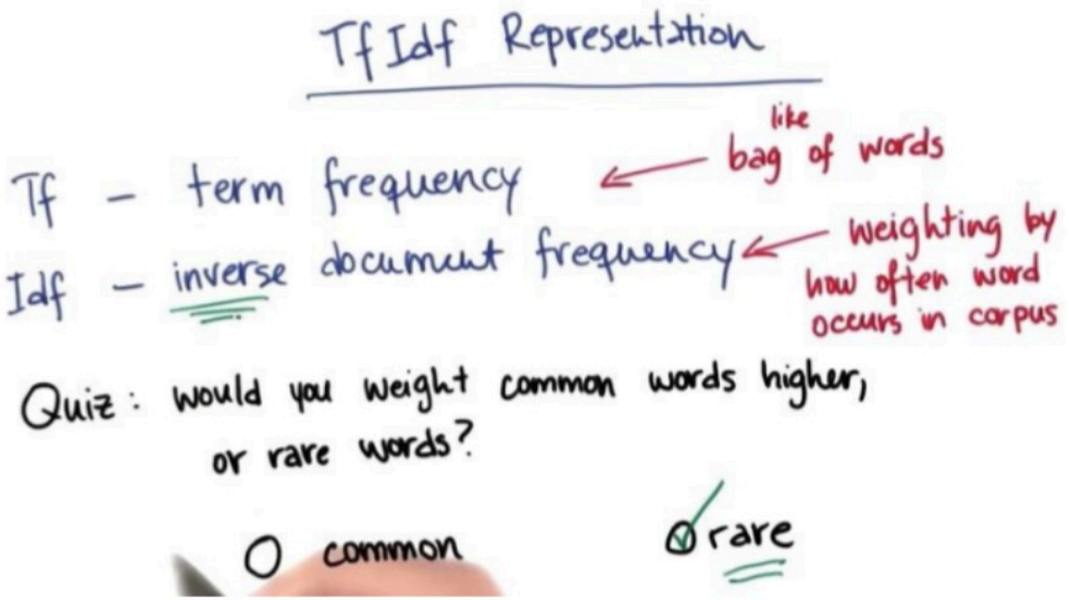


More advanced method than just counting the words.

The tf-idf weight is a weight used in NLP. This weight is a statistical measure used to evaluate how important a word is to a document in a corpus.

NLP Theory-Vectorization

TF-IDF



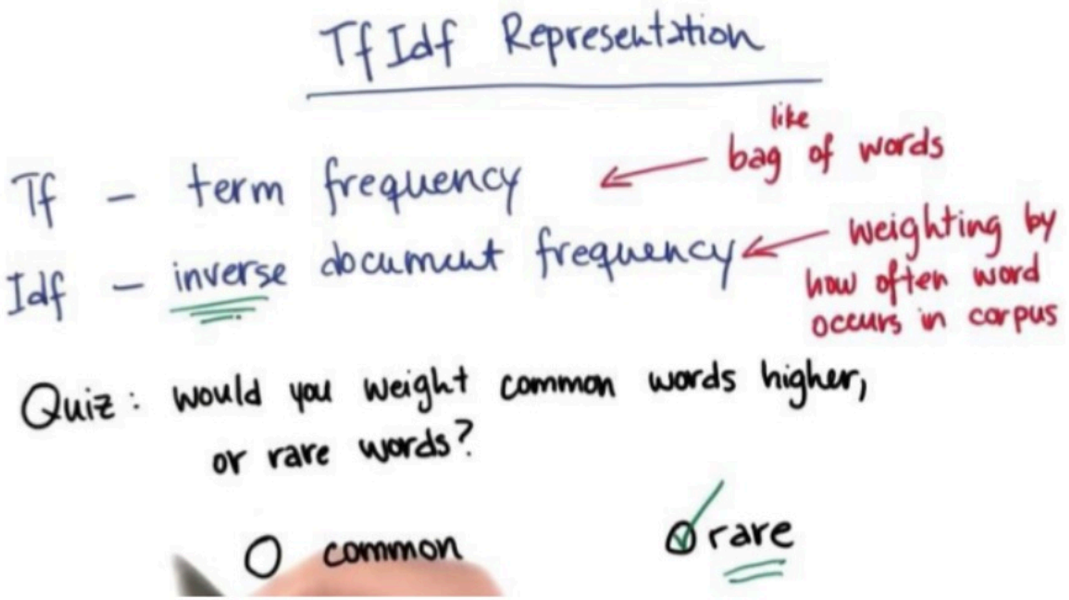
TF (Term Frequency)

TF measures how frequently a term occurs in a document.

TF(t) = (Number of times term t appears in a document) / (Total number of terms in the document).

NLP Theory-Vectorization

TF-IDF



IDF (Inverse Document Frequency)

IDF measures how important a term is.

IDF(t) = log10(Total number of documents in a corpus/ Number of documents with term t in it).

NLP Theory-Vectorization

TF-IDF

The importance increases proportionally to the number of times, a word appears in the document.

TF-IDF

But the importance decreases by the frequency of the word in the corpus.

NLP Theory-Vectorization

TF-IDF

Document-1 : John likes to watch movies. Mary likes movies too.

Document-2 : Mary also likes to watch football games.

Cleaned Document-1 : john likes ~~to~~ watch movies~~.~~ mary likes movies ~~too.~~

Cleaned Document-2 : mary ~~also~~ likes ~~to~~ watch football games~~.~~

Cleaned Document-1 : john like watch movie mary like movie

Cleaned Document-2 : mary like watch football game

john token for document-1

TF(john) = 1/7

DF(john) = 1/2

IDF(john) = log10(2/1+1) = 0.47

TF-IDF(john) = 1/7 x 0.47 = **0.07**

like token for document-2

TF(like) = 1/5

DF(like) = 2/2

IDF(like) = log10(2/2+1) = 0.3

TF-IDF(like) = 1/5 x 0.3 = **0.06**

Vectors of Each Document							
vector-1	0	0	0.07	0.04	0.04	0.13	0.04
vector-2	0.09	0.09	0	0.06	0.06	0	0.06

NLP Theory-Vectorization

TF-IDF Example

Consider a document containing 100 words wherein the word cow appears 3 times.

Assume the corpus has 10 million documents and the word cow appears in one thousand of these.

The **TF** for cow is then (3 / 100) = **0.03**

The DF is 1.000/10.000.000

Then, the **IDF** is calculated as log10(10.000.000 / 1.000+1) = **4**

Thus, the **TF-IDF weight** is the product of these quantities: 0.03 * 4 = **0.12**