# The many Application of Computational Science

INTRODUCING A SUBSET OF COMPUTATIONAL SCIENCE NATURAL LANGUAGE PROCESSING

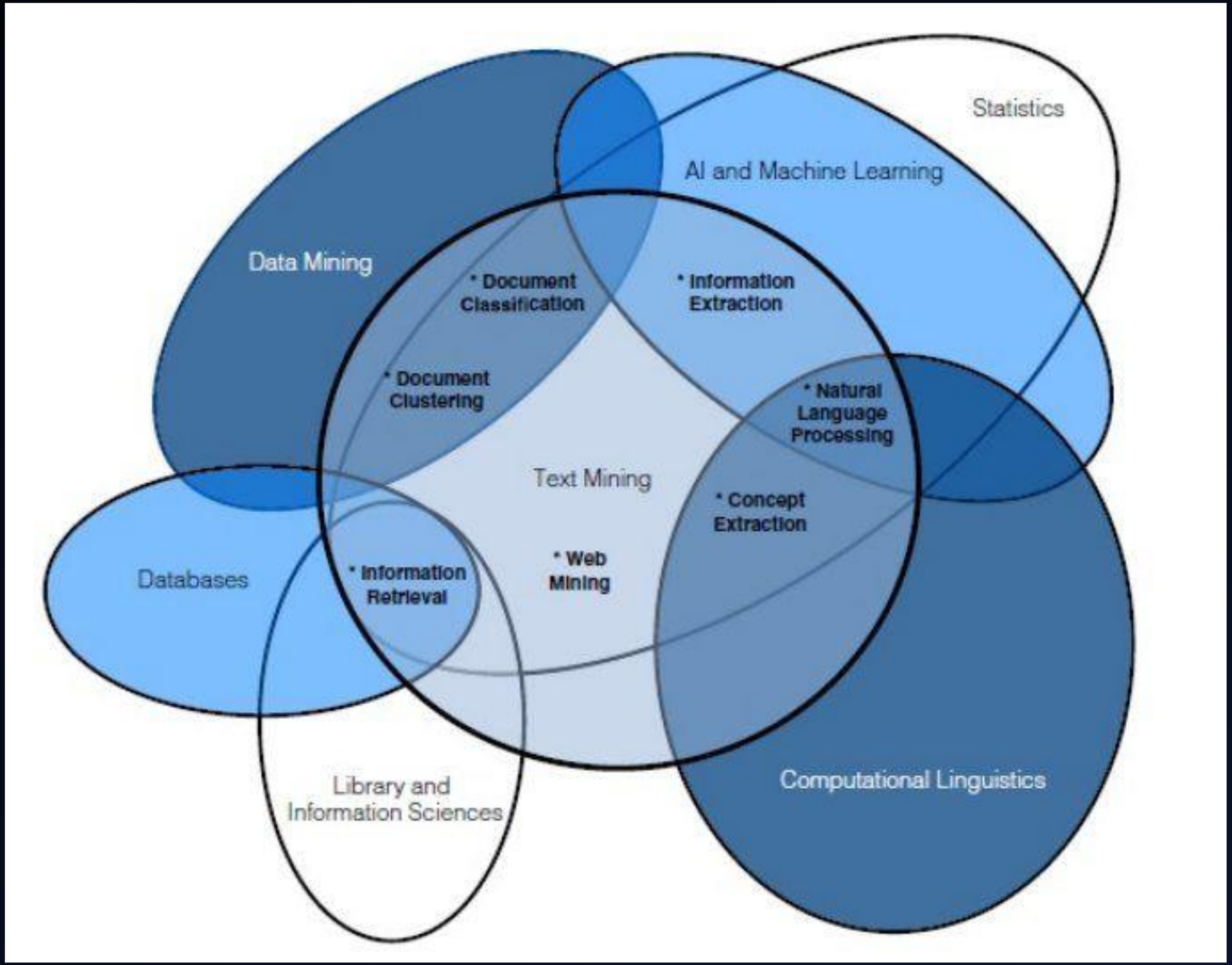
## Foreword

- There is no such thing as a stupid question -Albert Einstein

- *Stupid question only comes from a half-hearted mind*.

- If  I could not answer your question the way you anticipated, usually it's because of time-constraint or the occasional stack is overflowing.
  So don’t harbor bad feelings

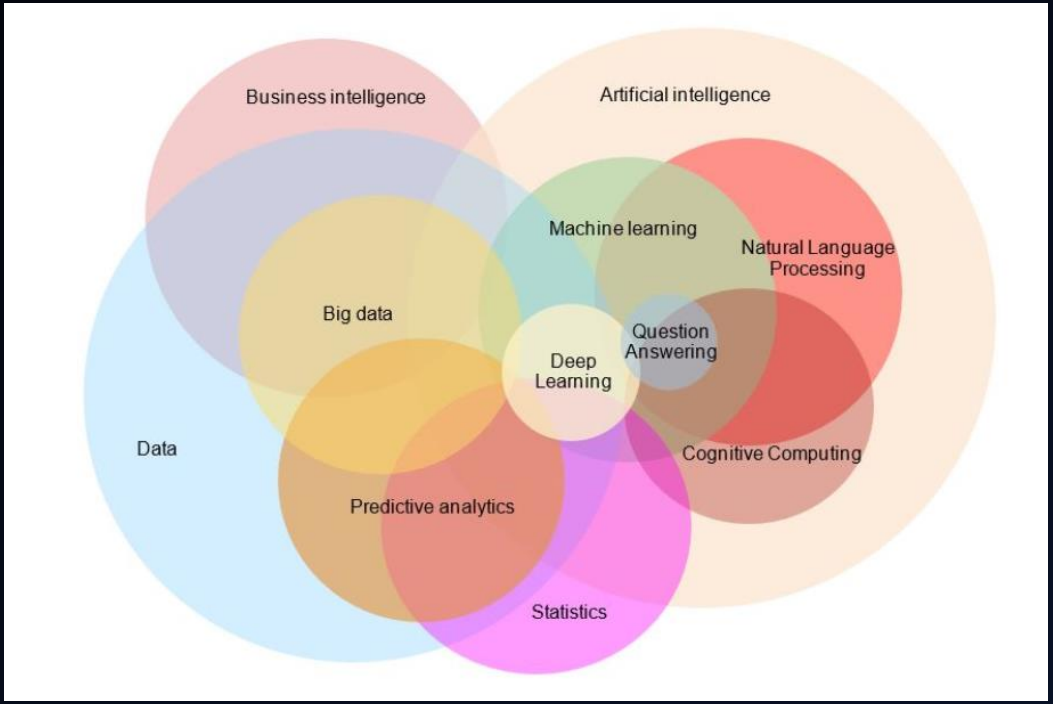
Computational Science is a interdisciplinary subject that is heavily based around mathematical concepts to solve complex **science** and engineering problems. *Definition Constantly evolving*



Very Broad

## Focusing on NLP a subset of Computational Linguistics

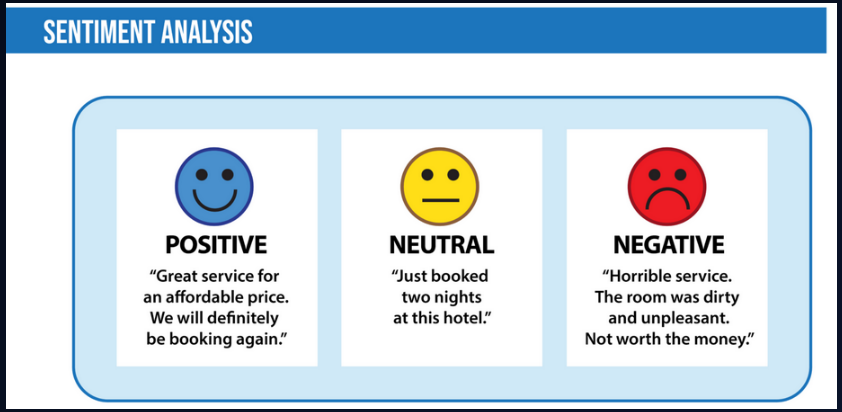
- Natural Language Processing (NLP)
  - AI and Machine Learning



Still very BROAD!

## What is NLP used for

- Sentiment Analysis
  - is the process of classifying the emotional intent of text. Generally, the input to a sentiment classification model is a piece of text, and the output is the probability that the sentiment expressed is positive, negative, or neutral. Typically, this probability is based on either hand-generated features, word n-grams, TF-IDF features, or using deep learning models to capture sequential long- and short-term dependencies. Sentiment analysis is used to classify customer reviews on various online platforms as well as for niche applications like identifying signs of mental illness in online comments.


SENTIMENT ANALYSIS
POSITIVE
"Great service for an affordable price. We will definitely be booking again."
NEUTRAL
"Just booked two nights at this hotel."
NEGATIVE
"Horrible service. The room was dirty and unpleasant. Not worth the money."

## What is NLP used for

- Grammatical error correction
  - models encode grammatical rules to correct the grammar within text. This is viewed mainly as a sequence-to-sequence task, where a model is trained on an ungrammatical sentence as input and a correct sentence as output. Online grammar checkers like Grammarly and word-processing systems like Microsoft Word use such systems to provide a better writing experience to their customers. Schools also use them to grade student essays.

## What is NLP used for

- Information retrieval
  - finds the documents that are most relevant to a query. This is a problem every search and recommendation system faces. The goal is not to answer a particular query but to retrieve, from a collection of documents that may be numbered in the millions, a set that is most relevant to the query.

## Focusing on NLP as a conversational agent

- Text generation
  - more formally known as natural language generation (NLG), produces text that's similar to human-written text. Such models can be fine-tuned to produce text in different genres and formats — including tweets, blogs, and even computer code. Text generation has been performed using Markov processes, LSTMs, BERT, GPT-2, LaMDA, and other approaches. It's particularly useful for autocomplete and **chatbots.**

## Focusing on Chatbots

- automate one side of a conversation while a human conversant generally supplies the other side. They can be divided into the following two categories: Database query: We have a database of questions and answers, and we would like a user to query it using natural language.
- Conversation generation: These chatbots can simulate dialogue with a human partner. Some are capable of engaging in wide-ranging conversations. A high-profile example is Google’s LaMDA, which provided such human-like answers to questions that one of its developers was convinced that it had feelings.

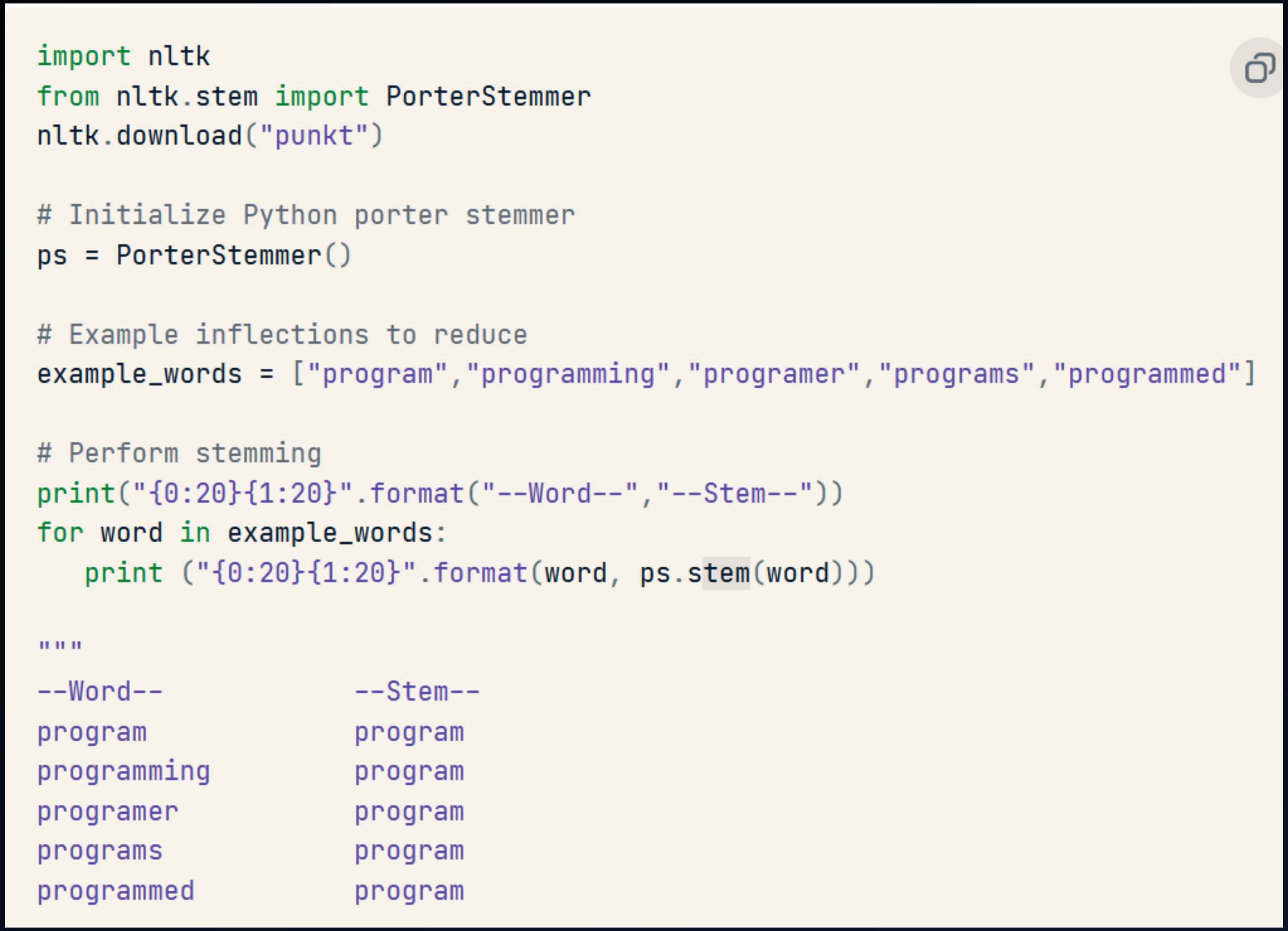
## Natural language processes

- **Data preprocessing:** Before a model processes text for a specific task, the text often needs to be preprocessed to improve model performance or to turn words and characters into a format the model can understand.
  - Stemming and lemmatization
  - Sentence segmentation
  - Stop word removal
  - Tokenization

## Data preprocessing

- Stemming and lemmatization
  - Stemming and lemmatization are **normalization** operations. in which you prepare words for analysis. For example, before calculating statistics on words in a body of text, you might convert all words to lowercase so that capitalized and lowercase words are not treated differently. Sometimes, you might want to use a word’s root to represent the word’s many forms. For example, in a given application, you might want to treat all of the following words as “program”: program, programs, programmer, programming and pro-grammed

# PorterStemmer abstraction



```python
import nltk
from nltk.stem import PorterStemmer
nltk.download("punkt")

# Initialize Python porter stemmer
ps = PorterStemmer()

# Example inflections to reduce
example_words = ["program","programming","programer","programs","programmed"]

# Perform stemming
print("{0:20}{1:20}".format("--Word--","--Stem--"))
for word in example_words:
   print ("{0:20}{1:20}".format(word, ps.stem(word)))

"""
--Word--            --Stem--
program             program
programming         program
programer           program
programs            program
programmed          program
```

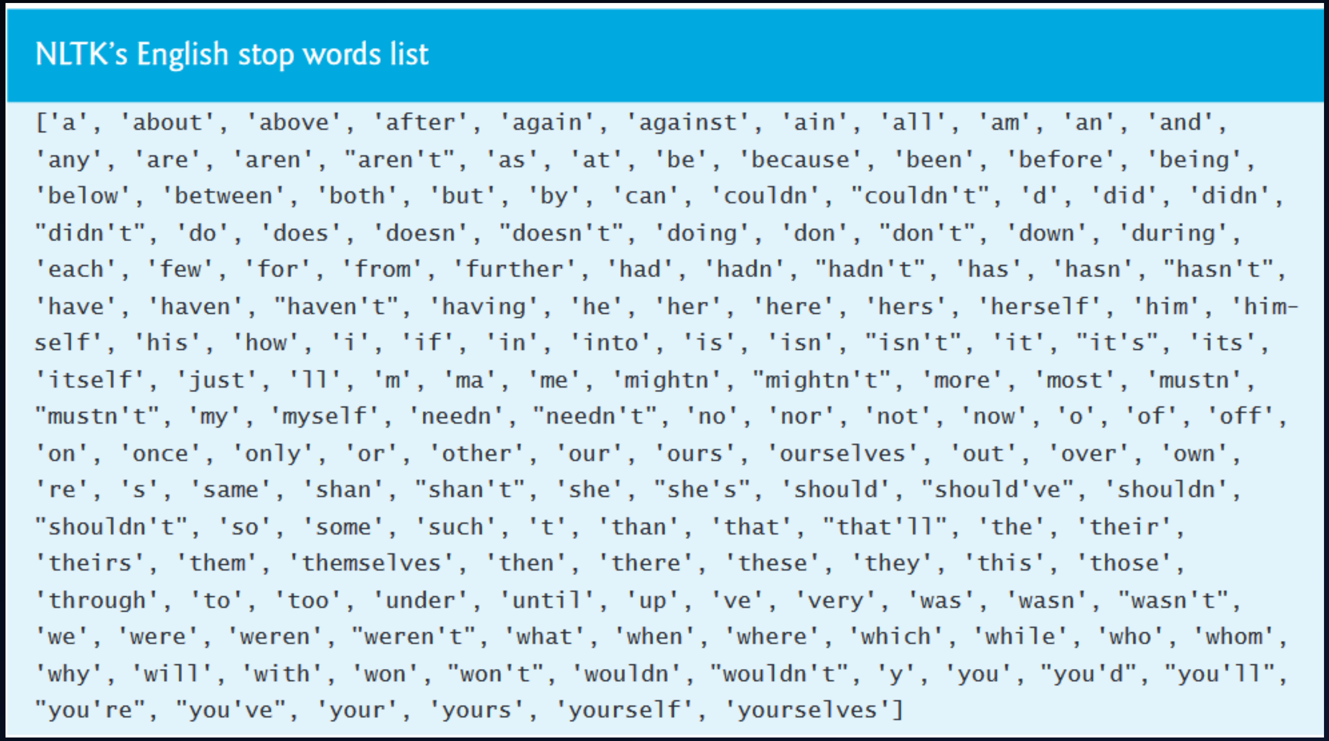
**CHOP-CHOP!**

## Data preprocessing

- **Text segmentation** breaks a large piece of text into linguistically meaningful sentence units. This is obvious in languages like English, where the end of a sentence is marked by a period, but it is still not trivial. A period can be used to mark an abbreviation as well as to terminate a sentence, and in this case, the period should be part of the abbreviation token itself. The process becomes even more complex in languages, such as ancient Chinese, that don’t have a delimiter that marks the end of a sentence.

## Stop word removal

- aims to remove the most commonly occurring words that don’t add much information to the text. For example, “the,” “a,” “an,” and so on.



NLTK’s English stop words list

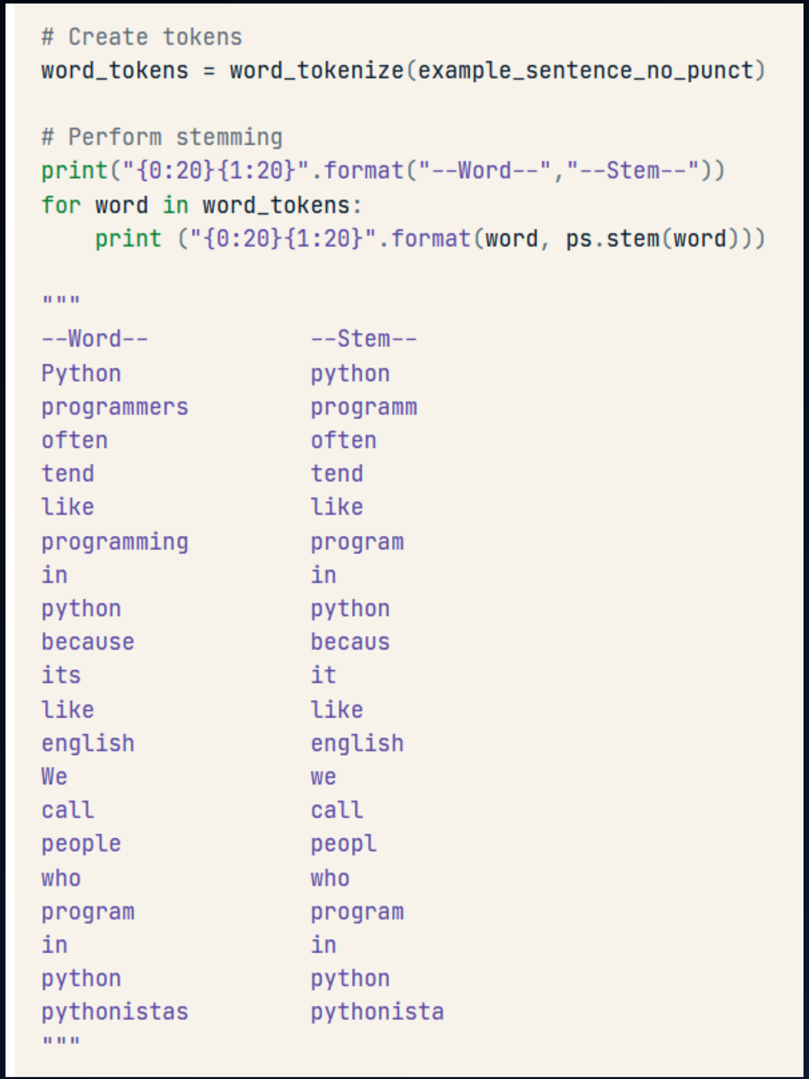
```
['a', 'about', 'above', 'after', 'again', 'against', 'ain', 'all', 'am', 'an', 'and',
'any', 'are', 'aren', "aren't", 'as', 'at', 'be', 'because', 'been', 'before', 'being',
'below', 'between', 'both', 'but', 'by', 'can', 'couldn', "couldn't", 'd', 'did', 'didn',
"didn't", 'do', 'does', 'doesn',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

## Tokenization

- splits text into individual words and word fragments. The result generally consists of a word index and tokenized text in which words may be represented as numerical tokens for use in various deep learning methods. A method that instructs language models to ignore unimportant tokens can improve efficiency.

## Tokenization in Action

**example_sentence = "Python programmers often tend like programming in python because it's like english. We call people who program in python pythonistas.**



```
# Create tokens
word_tokens = word_tokenize(example_sentence_no_punct)

# Perform stemming
print("{0:20}{1:20}".format("--Word--","--Stem--"))
for word in word_tokens:
    print ("{0:20}{1:20}".format(word, ps.stem(word)))

"""
--Word--            --Stem--
Python              python
programmers         programm
often               often
tend                tend
like                like
programming         program
in                  in
python              python
because             becaus
its                 it
like                like
english             english
We                  we
call                call
people              peopl
who                 who
program             program
in                  in
python              python
pythonistas         pythonista
"""
```

## Why do we need to normalize?

- **Variation Reduction:** Text data often contains variations of words due to different verb tenses, plural forms, and derivations. Normalization techniques like stemming and lemmatization reduce these variations to a common form, making it easier to analyze text and capture the core meaning of words.
- **Dimensionality Reduction:** Text data can have a high dimensionality due to the vast number of unique words. Stemming and lemmatization help reduce dimensionality by mapping similar words to a common root or lemma, which can improve the efficiency of text processing and modeling.

## Why do we need to normalize?

- **Improved Consistency:** Normalization enhances the consistency of text data. For example, the words "run," "running," and "ran" might all be stemmed or lemmatized to the common root "run." This consistency can lead to better results in text classification, sentiment analysis, and other NLP tasks.
- **Feature Extraction:** Stemming and lemmatization are often used as part of the feature extraction process in NLP. By reducing words to their base forms, you can create more meaningful features for machine learning models