

# INTRODUCTION TO NLP

By Ujjwal Nanda

# ROAD MAP

- Types of Data
- Text Dataset
- NLP Definition
- Real world application of NLP
- Business Use Case
- Regular Expression (RegEx)
- RegEx Functions
- Application of RegEx
- Python Lib for NLP
- Text Pre-processing
- What is Corpus, Token , N-grams, Stop Words
- Text Normalization : Stemming and Lemmatization
- Word Frequency Analysis
- Named Entity Recognition with Spacy
- Topic Modelling
- Unsupervised NLP
- Demo

# TYPES OF DATA

## Structured Data:

- Quantitative in the form of numbers and values
- Fixed dimensions
- Well organized
- Tabular
- Key value pair

Eg: Excel ,CSV, Json file

```
{
  "id": 123,
  "name": "Sandeep",
  "subject": "Computer",
  "marks": 23
},
{
  "id": 193,
  "name": "Raja",
  "subject": "Mathematics",
  "marks": 25
},
{
  "id": 223,
  "name": "Smith",
  "subject": "Geography",
  "marks": 20
},
}
```

Fig. Json format

Emp Id	Name	Department	Hire Date	Training
1234	J. Jones	Sales	18-Jun-92	24
2345	S. Smith	Production	12-Feb-98	40
3456	A. Adams	Sales	18-Nov-98	60
4567	B. Bates	Advertising	10-Mar-85	16
5678	D. Davis	Production	26-Jul-99	56
6789	C. Cole	Shipping	18-May-91	32
7890	E. Ellis	Sales	15-Dec-98	80
8901	F. Files	Advertising	17-Oct-90	24
9012	G. Gates	Advertising	15-Mar-99	48

Fig. Excel format

# TYPES OF DATA

## Unstructured Data:

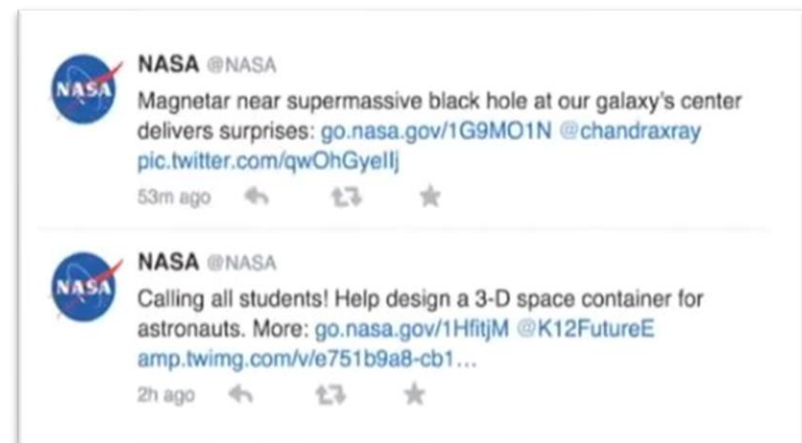
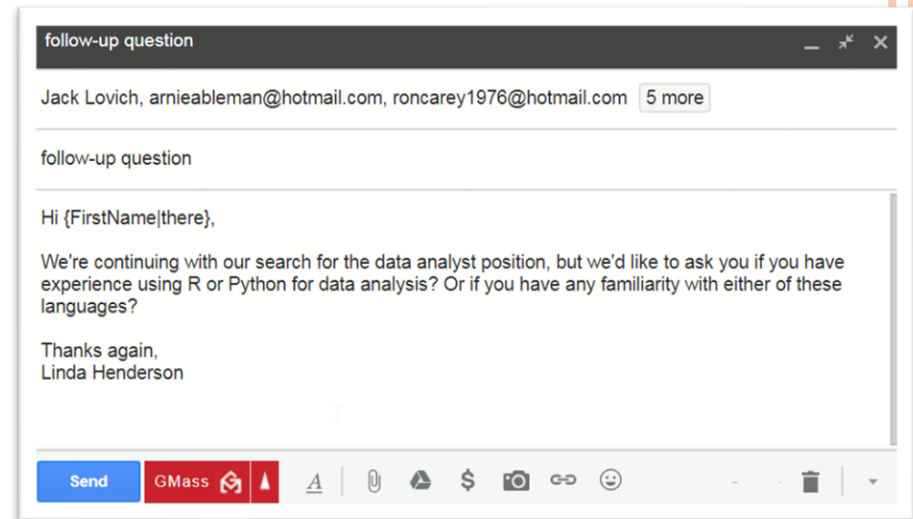
- Qualitative data in the form of text files , audio files, video files
- No fixed dimension
- No structure
- Not in the form of tabular form
- A huge array formats.

Eg: Images of Cat,  
Tweet

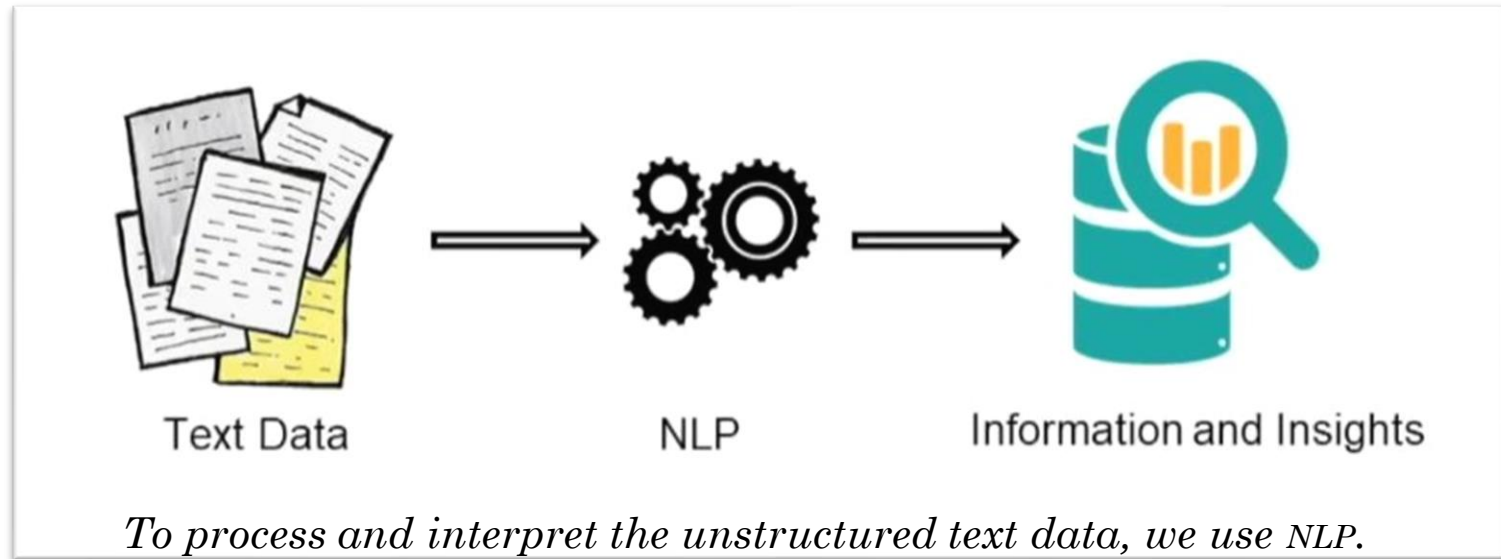


# TEXT DATASET

- **Text Data** : Words arranged in a meaningful manner.
- Written form of language
- Grammar and defined structures.
- Examples:
  - Social media tweets, posts comments
  - Conversations: messages, emails and chats
  - Articles: news, blogs transcript



# NATURAL LANGUAGE PROCESSING (NLP)



- Natural language processing (NLP) is the technique by which computers understand the human language.
- NLP allows you to perform a wide range of tasks such as classification, summarization, text-generation, translation and more.

# REAL WORLD APPLICATION OF NLP.

- Sentiment analysis
- Speech recognition
- Text Classification
- Machine Translation
- Semantic Search
- News Article Summarization
- Answering Questions
- Topic modelling

# BUSINESS USE CASE EXAMPLE 1:

## Automatic Categorization of Customer Queries

### NLP Solution :

- Analyze : top keywords and phrases
- Map : keywords and department descriptions
- Model : rules for automatic categorization





# BUSINESS USE CASE EXAMPLE 2:

## Identify Patients at Risk of Cancer

### NLP Solution :

- Analyze : history of patients and drugs prescribed
- Identify : entities from their history
- Model : business rules to classify the risk



# REGULAR EXPRESSION (REGEX)

- A RegEx, or Regular Expression, is a sequence of characters that forms a search pattern.
- RegEx can be used to check if a string contains the specified search pattern
- Useful in finding information and pattern embedded in the text.
- Some common usages of RegEx are
  - Search a string
  - Finding a string
  - Replace a part of string

## REGEX FUNCTIONS:

- `match` : finds the first occurrence of pattern in the string
- `search` : locates the pattern in the string
- `findall` : find all occurrences of patterns in the string
- `sub` : search and replace
- `split` : split the text by the given regular expression pattern

# EXAMPLES :

## *Example 1:*

```
import re

string = "Tiger is the national animal of India"
pattern = "Tiger"
result = re.match(pattern, string).group(0)
>> Tiger

string = "The national animal of India is Tiger"
pattern = "Tiger"
result = re.search(pattern, string).group(0)
>> Tiger
```

## *Example 2:*

```
string = "John 34-3456 12-05-2007, XYZ 56-4532 11-11-2011, ABC 67-8945 12-01-2009"
pattern = r'\d{2}-\d{2}-\d{4}'
re.findall(date_pattern, text)
>> ['12-05-2007', '11-11-2011', '12-01-2009']

string = 'this is;a sample;text,string'
pattern = r'[.,\s]'
re.split(pattern, text)
>> ['this', 'is', 'a', 'sample', 'text', 'string']

string = "cricket is a popular sport of India"
pattern = "India"
replacement = "the world"
re.sub(pattern, replacement, string)
>> "cricket is a popular sport of the world"
```

Link for Cheat Sheat : <https://www.geeksforgeeks.org/python-regex-cheat-sheet/>

# APPLICATION OF REGEX.

- Text Cleaning
- Text mining
- Segmentation of words from sentences
- Segmentation of sentences from paragraphs
- Noise removal/Flagging
- Information retrieval from texts

**> Will show you demo of RegEx with real world use case.**

# PYTHON LIBRARIES FOR NLP

- **NLTK** : (Natural Language Toolkit) is a suite that contains libraries and programs for statistical language processing.
- **SpaCy**: It is the most trending and advanced library for implementing NLP today. It is many features that provide clear advantage for processing text data and modeling.
- **Genism**: It was developed for topic modelling. It supports the NLP tasks like Word Embedding, text summarization and many others.
- **Transformers**: It was developed by HuggingFace and provides state of the art models. It is an advanced library known for the transformer modules, it is currently under active development.

# TEXT PRE-PROCESSING

- The raw text data often referred to as **text corpus** has a lot of noise.
- There are punctuation, suffices and stop words that do not give us any information.
- Text Processing involves preparing the text corpus to make it more usable for NLP tasks.

# WHAT IS CORPUS ?


- A **text corpus** is a large and structured set of **texts** (nowadays usually electronically stored and processed).
- **Text corpora** are used to do statistical analysis and hypothesis testing, checking occurrences or validating linguistic rules within a specific language territory.
- Corpus : Collection of text documents
- Corpus>Documents>Paragraphs>Sentences>Tokens



# WHAT IS A TOKEN?

- The words of a text document/file separated by spaces and punctuation are called as **tokens**.
- Tokens are smaller units of text (words , phrases, n-grams)

**Tokenization:** The process of extracting tokens from a text file/document is referred as tokenization.

**Most frequently used  
Methods :** 

White space tokenizer / Unigram tokenizer

Sentence : "I went to New-York to play football"

Tokens : "I", "went", "to", "New-York", "to", "play", "football"

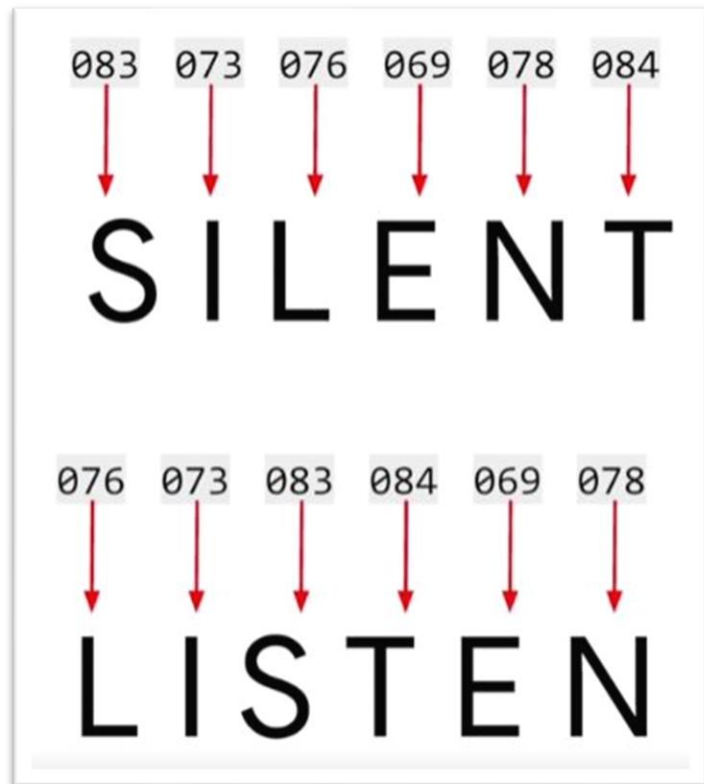
Regular expression tokenizer

Sentence : "Football;Cricket;Golf Tennis"

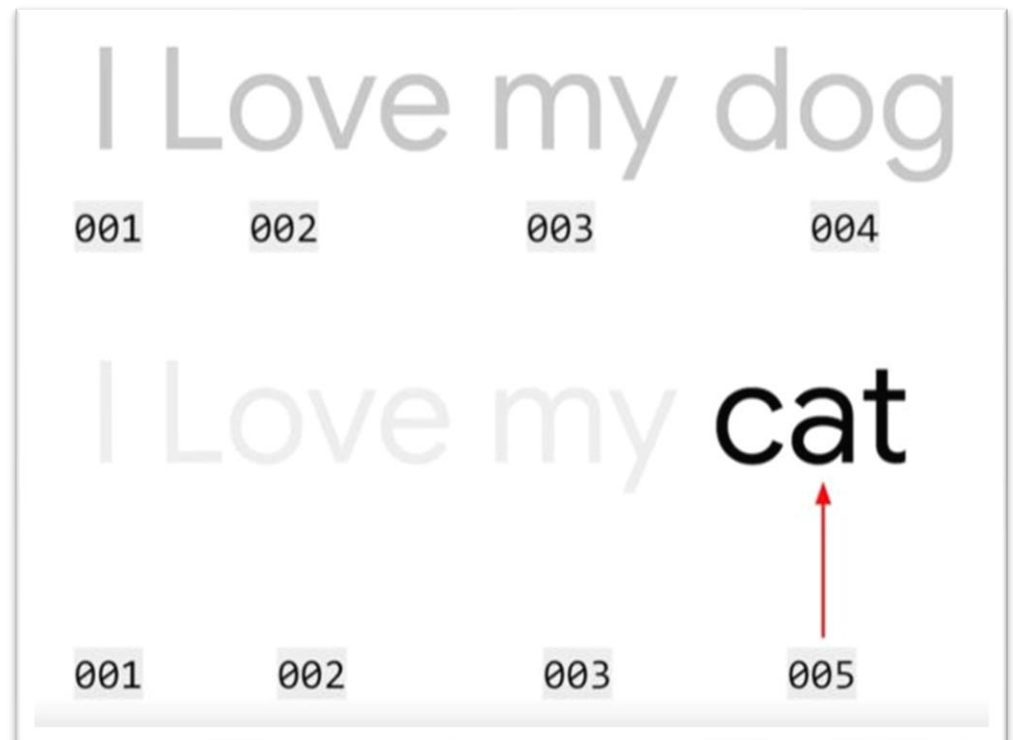
`re.split(r'[;,\s]', line)`

Tokens : "Football", "Cricket", "Golf", "Tennis"

# HOW WE TOKENIZE ..



Not Using ASCII value



Tokenizing the words

# WHAT ARE N-GRAMS ?

- N grams are combinations of n-words / characters together
- Eg:

Sentence : I love my phone

Unigrams (n =1) : I, Love, my, phone

Bigrams (n=2) : I Love, Love my, my phone

Trigrams (n=3) : I love my, love my phone

## This is Big Data AI Book

**Uni-Gram**

This	Is	Big	Data	AI	Book
------	----	-----	------	----	------

**Bi-Gram**

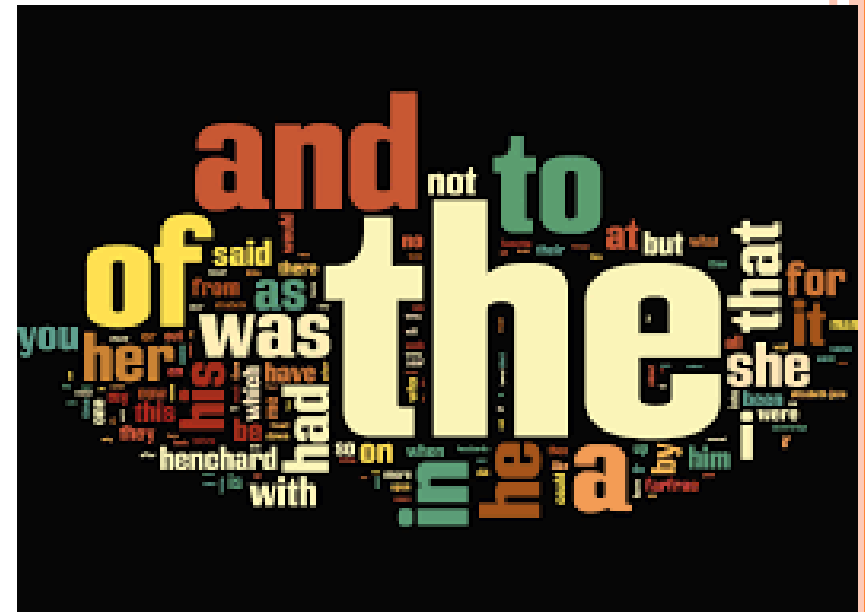
This is	Is Big	Big Data	Data AI	AI Book
---------	--------	----------	---------	---------

**Tri-Gram**

This is Big	Is Big Data	Big Data AI	Data AI Book
-------------	-------------	-------------	--------------

# WHAT ARE STOP WORDS ?

- **Stop words** are the words in any language which does not add much meaning to a sentence.
- They can safely be ignored without sacrificing the meaning of the sentence.
- These are some of the most common, short function words, such as the, is, at, which, and on.



**Fig. Word cloud of stop words**

# TEXT NORMALIZATION

- Morpheme : base form of the word
- Structure of Token:

<prefix> <**morpheme**> <suffix>

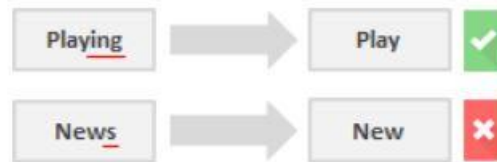
Example:

**Antinationalist** : Anti + **national** + ist

**Normalization** is a process of converting a token into its base form (morpheme). It helps in reducing the unique token counts, removing variation of the words.

# WHAT IS STEMMING ?

- Stemming means reducing a word to its 'root form'.
- Some words say 'Playing', 'Played', 'Plays'. we know that these words are not very distinct from each other.



- The word 'New' is not semantically correct for 'News'.
- Stemming may give us root words that are not present in the dictionary

# WHAT IS LEMMATIZATION ?

- It is similar to stemming, except that the root word is correct and always meaningful.
- For example, verbs in past tense are changed into present (e.g. “went” is changed to “go”) and synonyms are unified (e.g. “best” is changed to “good”), hence standardizing words with similar meaning to their root.



- Lemmatization also takes into consideration the context of the word in order to **solve other problems like disambiguation**. Eg. Similar words like Bat

# WORD FREQUENCY ANALYSIS

- Once the stop words are removed and lemmatization is done ,the tokens we have can be analysed further for information about the text data.
- The words which occur more frequently in the text often have the key to the core of the text.
- So, we shall try to store all tokens with their frequencies for the same purpose.



## Example

```
#import Counter
import collections
from collections import Counter

# creating nlp obj using spacy
data='It is my birthday today. I could not have a birthday party. I felt sad'
data_doc=nlp(data)

#creating a list of tokens after preprocessing
list_of_tokens=[token.text for token in data_doc if not token.is_stop and not token.is_punct]

#passing the list of tokens to Counter
token_frequency=Counter(list_of_tokens)
print(token_frequency)
```

```
Counter({'birthday': 2, 'today': 1, 'party': 1, 'felt': 1, 'sad': 1})
```

From above output, the word ‘ birthday ‘ is most common.

So,it gives us an idea that the text is about a birthday.

# NAMED ENTITY RECOGNITION

- Suppose you have a collection of news articles text data. What if you want to know what companies/organizations have been in the news ? How will you do that ?
- The answer to all these questions is NER.
- NER is the technique of identifying named entities in the text corpus and assigning them pre-defined categories such as person names, locations, organizations etc..

# NER USING SPACY

- Spacy is highly flexible and advanced library. Named entity recognition through spacy is easy.
- Every token of a spacy model, has an attribute `token.label_` which stores the category/ label of each entity.

The few common labels are:

- 'ORG' : companies, organizations, etc..
- 'PERSON' : names of people
- 'GPE' : countries, states, etc..
- 'PRODUCT' : vehicles, food products and so on
- 'LANGUAGE' : Names of different languages
- There are many other labels too.

# NER WITH SPACy EXAMPLE

```
# Creating a spacy doc of a sentence
```

```
sentence=' The building is located at London. It is the headquarters of Yahoo. John works there.  
doc=nlp(sentence)
```

```
# Print named entities
```

```
for entity in doc.ents:  
    print(entity.text, '--', entity.label_)
```

```
London -- GPE
```

```
Yahoo -- ORG
```

```
John -- PERSON
```

```
English -- LANGUAGE
```

# TOPIC MODELLING

- Is as a method for uncovering hidden structures in sets of texts or documents.
- In essence it clusters texts to discover latent topics based on their contents, processing individual words and assigning them values based on their distribution.
- This technique is based on the assumptions that each document consists of a mixture of topics and that each topic consists of a **set of words**, which means that if we can spot these hidden topics we can unlock the meaning of our texts.
- **Latent Dirichlet Allocation (LDA)** is probably the most commonly used. This relatively new algorithm (invented less than 20 years ago) works as an unsupervised learning method that discovers different topics underlying a collection of documents.

# UNSUPERVISED NLP

- In **unsupervised learning** methods like LDA, there is no output variable to guide the learning process and data is explored by algorithms to find patterns.
- To be more specific, LDA finds groups of related words by:
  - Assigning each word to a random topic, where the user defines the number of topics it wishes to uncover. You don't define the topics themselves (you define just the number of topics) and the algorithm will map all documents to the topics in a way that words in each document are mostly captured by those imaginary topics.
  - The algorithm goes through each word iteratively and reassigns the word to a topic taking into considerations the probability that the word belongs to a topic, and the probability that the document will be generated by a topic. These probabilities are calculated multiple times, until the convergence of the algorithm.

Unlike other clustering algorithms like *K-means* that perform hard clustering (where topics are disjointed)

LDA assigns each document to a mixture of topics, which means that each document can be described by one or more topics (e.g. Document 1 is described by 70% of topic A, 20% of topic B and 10% of topic C) and reflect more realistic results.

Thank You..