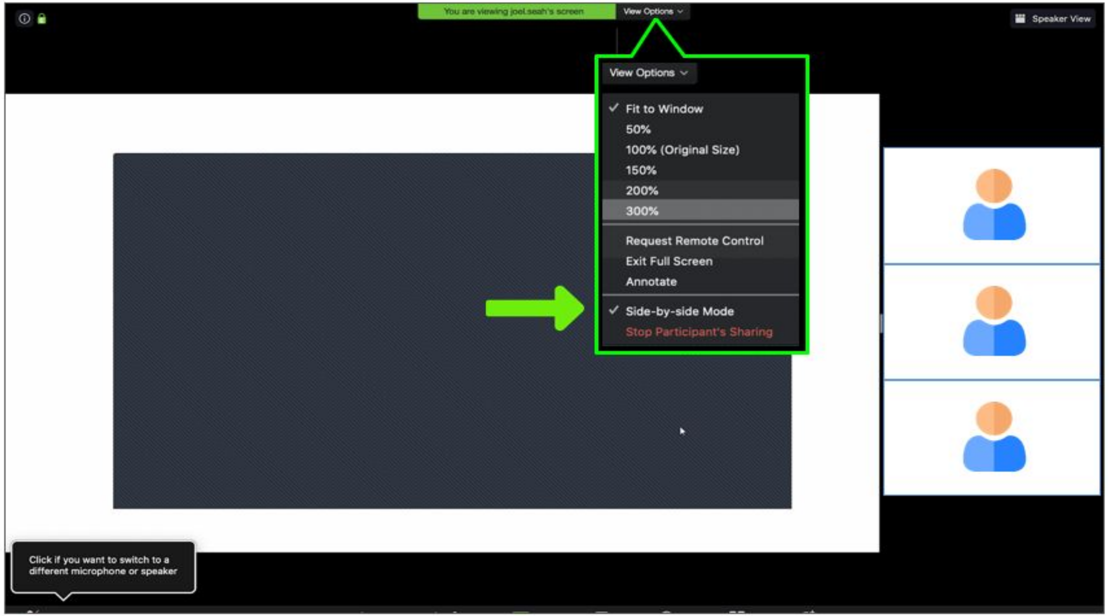
# We’ll Be Starting Shortly!

To help us run the workshop smoothly, kindly:

- Submit all questions using the Q&A function
- If you have an urgent request, please use the “Raise Hand” function

# Using Zoom: Viewing Mode




## Side-By-Side Mode

- When sharing screen (slide share)
- With small thumbnails of people on the sidebar

**STEPS:**

1. View Options
2. Side-By-Side Mode

# NLP - Sentiment Analysis

Smartcademy

ENGLISH

CHINESE

# Natural Language Processing

PYTHON

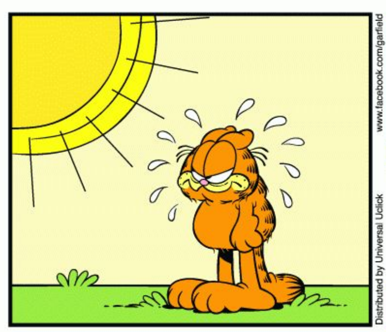
JAVA

# Natural Language Processing

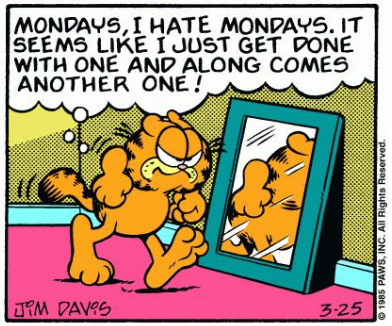
# Natural Language Processing

Garfield was trying to stay cool

- **GARFIELD WAS TRYING TO STAY COOL**



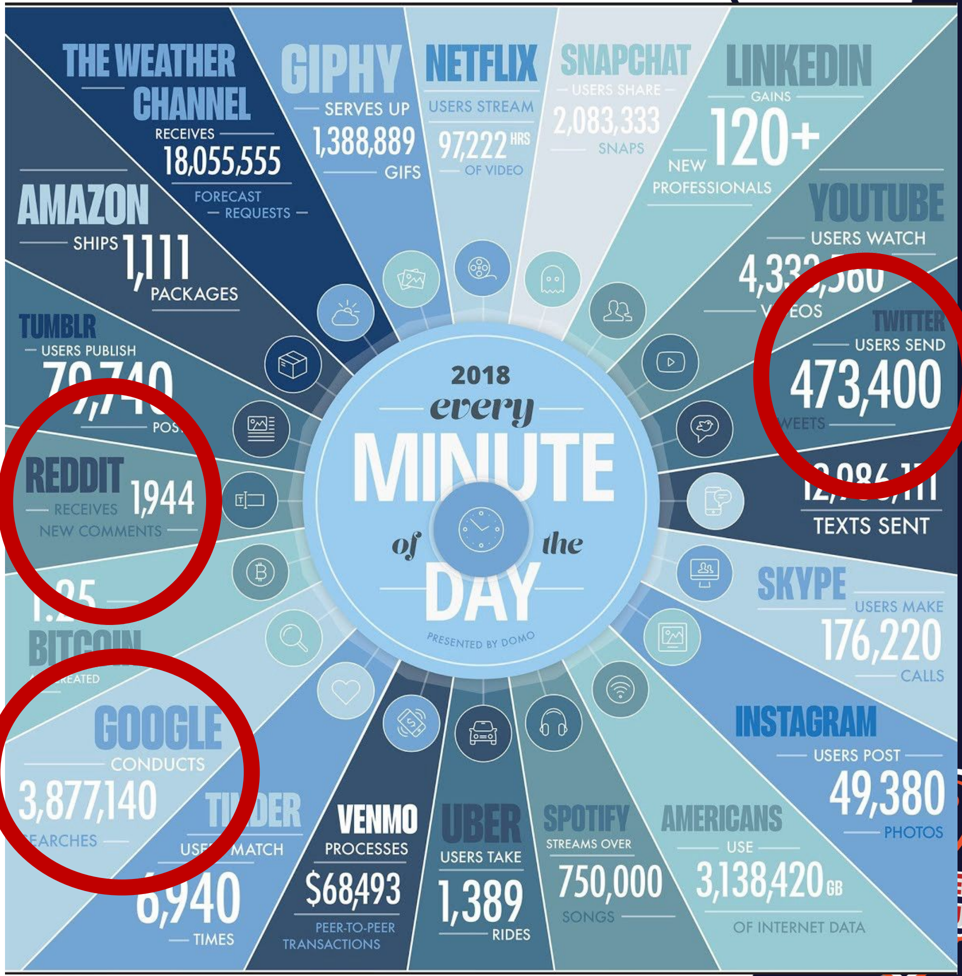

- **GARFIELD WAS TRYING TO STAY COOL**




- **GARFIELD WAS TRYING TO STAY COOL**

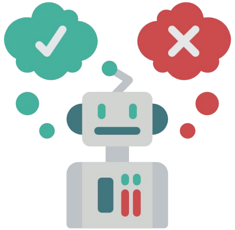
# WHY

- Natural Language
- Convey information between 2 people
- Structured Vs Unstructured Data
- NLP is the interdisciplinary field combining computer science and linguistics



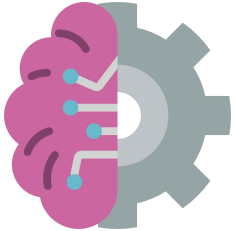

Source:
https://www.domo.com/solution/data-never-sleeps-6

# Natural Language Processing - NLP



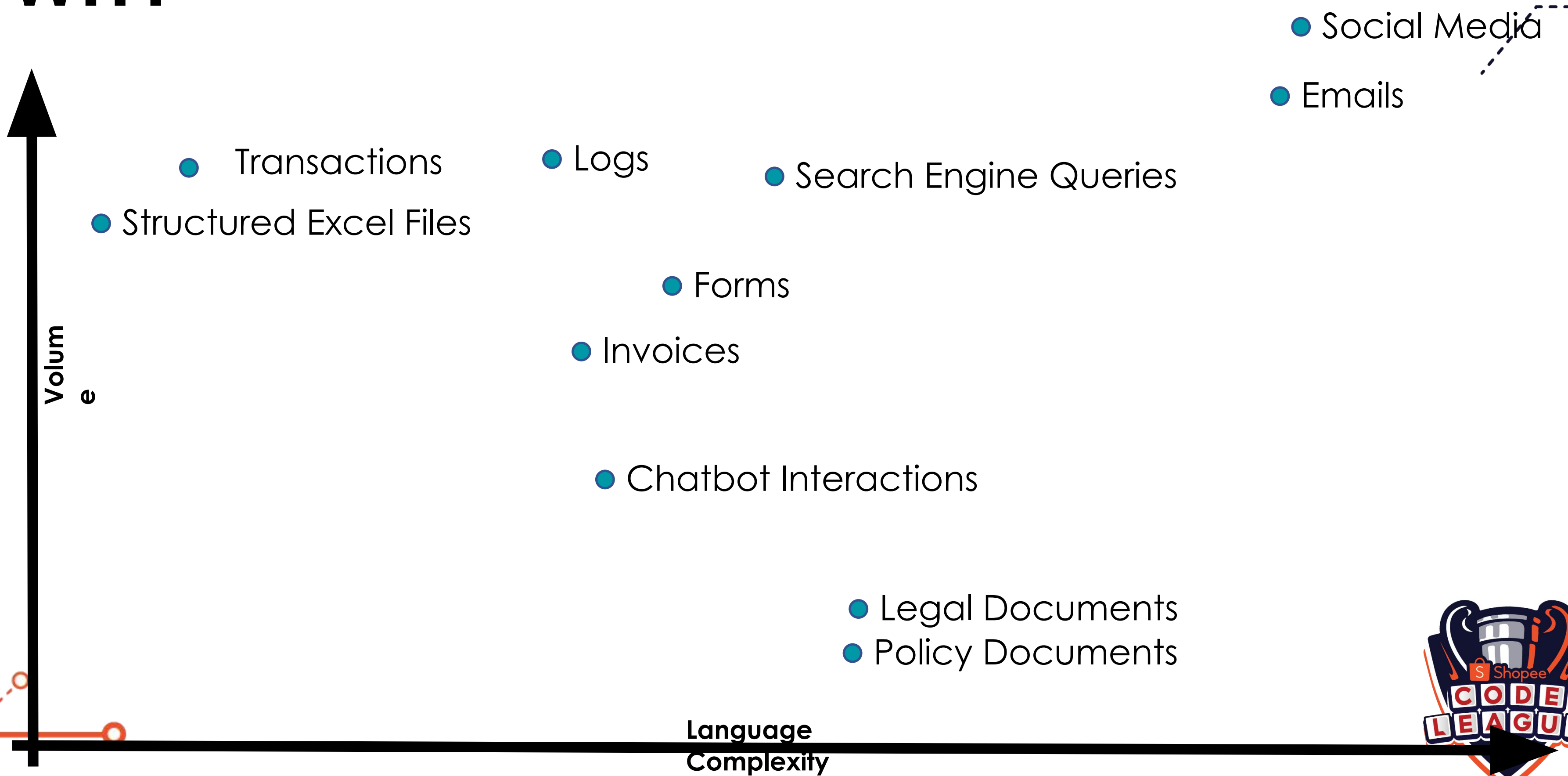
- MACHINE CAN INTERPRET HUMAN LANGUAGE
  - Facilitates the Human Machine Interaction
  - Enables the Machine to Machine Interaction

- DATA DRIVEN AND KNOWLEDGE DRIVEN
  - Machine Learning for data classification and generation
  - Semantic reasoning for data discovery and disambiguation



- SIMULATING HUMAN BRAIN
  - Current models performs well at individual task, still needs improvements for multiple tasks

# WHY



Social Media
Emails
Transactions
Logs
Search Engine Queries
Structured Excel Files
Forms
Invoices
Chatbot Interactions
Legal Documents
Policy Documents
Volum
e
Language
Complexity

# Applications of NLP

1. Sentiment analysis
2. Chatbot
3. Speech recognition
4. Language Translation
5. Information retrieval/extraction
6. Advertisement matching

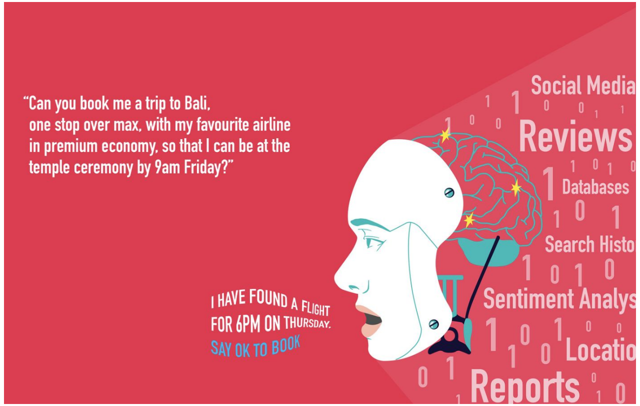
# Applications of NLP

1. Sentiment analysis
2. Chatbot
3. Speech recognition
4. Language Translation
5. Information retrieval/extraction
6. Advertisement matching

**Nordstrom digs into 5-star customer reviews and finds a shipping problem.**

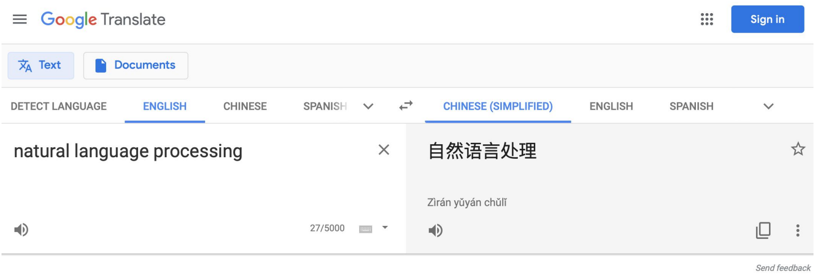
## Applications of NLP

1. Sentiment analysis
2. Chatbot
3. Speech recognition
4. Language Translation
5. Information retrieval/extraction
6. Advertisement matching



# Applications of NLP

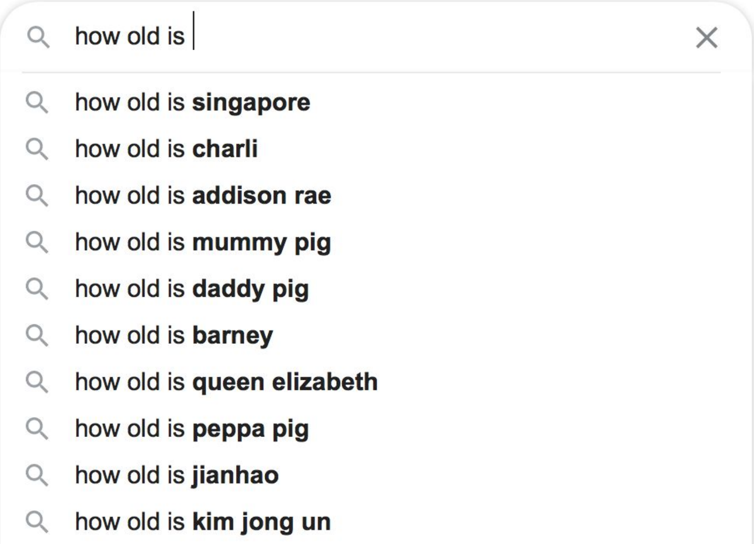
1. Sentiment analysis
2. Chatbot
3. Speech recognition
4. Language Translation
5. Information retrieval/extraction
6. Advertisement matching

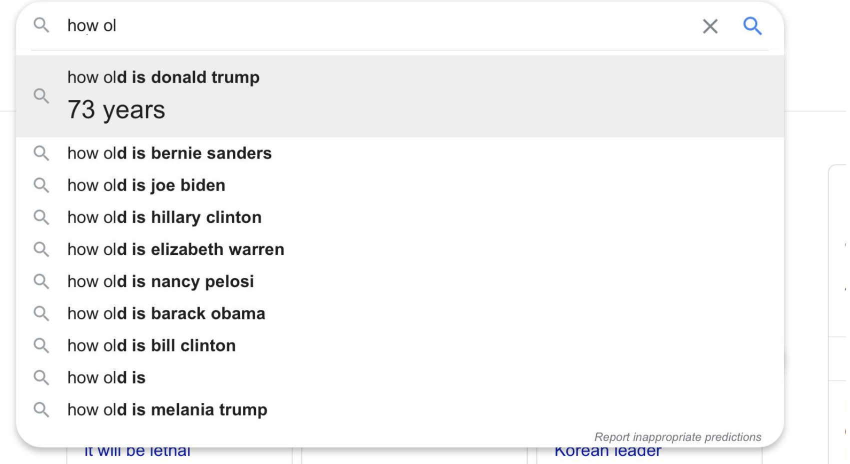


# Applications of NLP

1. Sentiment analysis
2. Chatbot
3. Speech recognition
4. Language Translation
5. Information retrieval/extraction
6. Advertisement matching

# E.g. Semantic search engine



# Walmart’s semantic search engine increased conversion rates by 10-15%

# Applications of NLP

1. Sentiment analysis
2. Chatbot
3. Speech recognition
4. Language Translation
5. Information retrieval/extraction
6. Advertisement matching

# Natural Language Understanding

# TEXT NORMALISATION

**She bought 10 apples and 10 oranges from the nearby grocer .**

- CONVERTING ALL LETTERS TO LOWER OR UPPER CASE
  **she bought 10 apples and 10 oranges   from the nearby grocer .**
- CONVERTING NUMBERS INTO WORDS OR REMOVING NUMBERS
  **she bought apples and oranges   from the nearby grocer .**
- REMOVING PUNCTUATIONS, ACCENT MARKS AND OTHER DIACRITICS
  **she bought apples and oranges  from the nearby grocer**
- REMOVING WHITE SPACES
  **she bought apples and oranges from the nearby grocer**
- REMOVING STOP WORDS, AND PARTICULAR WORDS
  **bought apples oranges nearby grocer**

You can add your own Stop word. Go to your NLTK download **directory path** -> **corpora** -> **stopwords** -> update the stop word **file** depends on your language which one you are using. Here we are using english (stopwords.words('english')).

# PRE-PROCESSING

**She bought 10 red apples and 10 cans of coca cola from the nearby grocer.**

| | |
|---|---|
| TOKENISATION | **“bought” “red” “apples” “cans” “coca” “cola” “nearby” “grocer”** |
| N-GRAMS | **“red apples” “coca cola” “nearby grocer”** |
| STEMMING | **“bought” “appl” “can” “coca” “cola” “nearbi” “grocer”** |
| PART OF SPEECH (POS) TAGGING | **[('She', 'PRP'), ('bought', 'VBD'), ('10', 'CD'), ('apples', 'NNS'), ('and', 'CC'), ('10', 'CD'), ('cans', 'NNS'), ('of', 'IN'), ('coca', 'NN'), ('cola', 'NN'), ('from', 'IN'), ('the', 'DT'), ('nearby', 'JJ'), ('grocer', 'NN')]** |
| NAMED ENTITY RECOGNITION | **(S She/PRP bought/VBD 10/CD apples/NNS and/CC 10/CD cans/NNS of/IN (NP coca/NN) (NP cola/NN) from/IN (NP the/DT nearby/JJ grocer/NN))** |

# Tokenisation

Taking a text or set of text and breaking it up into its individual tokens (sentences, words, characters)

**She bought 10 red apples and 10 cans of coca cola from the nearby grocer.**

TOKENISATION **“bought” “red” “apples” “cans” “coca” “cola” “nearby” “grocer”**

- New York, Los Angeles, Singapore Management University

- **Language specific:**

Chinese: 地铁站
French: L’ensemble

- **Context is often missing: “can”**

# N-GRAMS

Sequence of N words, good for putting keywords into local context

**bought red apples cans coca cola nearby grocer**

**NGRAMS** **"bought red" "red apples" "apples can" "coca cola" "nearby grocer"**

| | |
|---|---|
| **BIGRAMS** | **"Coca cola"** |
| **TRIGRAMS** | **The Three Musketeers** |
| **4-GRAMS** | **National University of Singapore** |
| **5-GRAMS** | **etc** |

- Compression algorithms (the PPM variety especially) where the length of the grams depends on how much data is available for providing specific contexts.
- Approximate string matching (e.g. BLAST for genetic sequence matching)
- Predictive models (e.g. name generators)
- Speech recognition (phonemes grams are used to help evaluate the likelihood of possibilities for the current phoneme undergoing recognition)

# STEMMING & LEMMATISATION

Reduce inflectional forms and sometimes derivationally related forms of a word to a **common base form, to bring variant forms of a word together**

**She bought 10 red apples and 10 oranges from the nearby grocer.**

**STEM** **“bought” “appl” “orang” “nearbi” “grocer”**

**LEMMATIZE** **“buy” “apple” “orange” “nearby” “grocer”**

**SUFFIX**
**-ing**
**-ed**
**-es**
**-s**
**…**

```
application
 Stemming: applic Lemmatizing: application
applying
 Stemming: appli Lemmatizing: apply
applies
 Stemming: appli Lemmatizing: apply
applied
 Stemming: appli Lemmatizing: apply
apply
 Stemming: appli Lemmatizing: apply
apples
 Stemming: appl Lemmatizing: apples
apple
 Stemming: appl Lemmatizing: apple
```

**Porter**: Most commonly used stemmer, and provides Java support.

**Snowball**: Improvement over the Porter algorithm, even Porter admits it is better than his original algorithm. Slightly faster computation time than porter, with a fairly large community around it.

To view the entire algorithm: http://people.scs.carleton.ca/~armyunis/projects/KAPI/porter.pdf

# PART OF SPEECH TAGGING

Marking up a word in a corpus to a corresponding part of a speech tag, based on its context and definition

I **left** my keys in my **left** pocket.

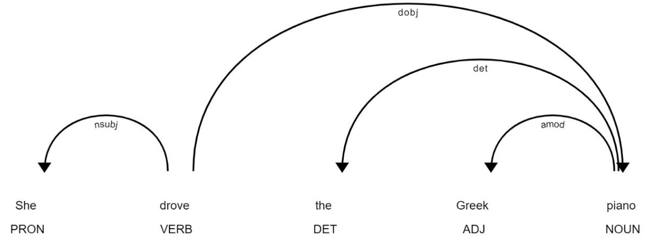
**PART OF SPEECH (POS) TAGGING**

[('I', 'PRP'), **('left', 'VBD'),** ('my', 'PRP$'), ('keys', 'NNS'), ('in', 'IN'), ('my', 'PRP$'), **('left', 'JJ'),** ('pocket', 'NN')]

Left - VBD verb, past tense
took

Left - JJ adjective

Building parse trees, which are used in building Named Entity Recognisers and extracting relations between words, helps in Syntactic and semantic analysis




Types:
1. Lexical Based Methods
2. Rule-Based Methods
3. Probabilistic Methods
4. Deep Learning Methods

# NAMED ENTITY Recognition

Identify all textual mentions of the named entities and classify them into pre-defined categories

**She bought 10 red apples and 10 cans of coca cola from the nearby grocer.**

**NAMED ENTITY RECOGNITION**

**(S She/PRP bought/VBD 10/CD apples/NNS and/CC 10/CD cans/NNS of/IN (NP coca/NN) (NP cola/NN) from/IN (NP the/DT nearby/JJ grocer/NN))**

**Stanford's Named Entity Recognizer** is based on an implementation of linear chain Conditional Random Field (CRF) sequence models. Model is only trained on instances of ***PERSON***, ***ORGANIZATION*** and ***LOCATION*** types.
Based on training data, the model will support different types of entities: https://spacy.io/api/annotation#section-named-entities

| Samples of Pre-defined categories | Examples |
|---|---|
| Names of people | Joan, Jeremy, Adam |
| Organisations | Accenture, Apple, GoJek |
| Locations | City Hall, Mount Fuji, |
| Expressions of times | June, 1980, 2008-03-10 |
| Percent | 100%, Twenty pct, |
| Monetary value | 18 Euros, $19, 600 Yen |

Each POS tag is attached to a single word, while NER tags can be attached to multiple words.

https://towardsdatascience.com/a-comparison-between-spacy-ner-stanford-ner-using-all-us-city-names-c4b6a54

# PRE-PROCESSING

**She bought 10 red apples and 10 cans of coca cola from the nearby grocer.**

| | |
|---|---|
| TOKENISATION | **“bought” “red” “apples”  “cans” “coca” “cola” “nearby” “grocer”** |
| N-GRAMS | **“red apples”  “coca cola” “nearby grocer”** |
| STEMMING | **“bought” “appl” “can” “coca” “cola” “nearbi” “grocer”** |
| PART OF SPEECH (POS) TAGGING | **[('She', 'PRP'), ('bought', 'VBD'), ('10', 'CD'), ('apples', 'NNS'), ('and', 'CC'), ('10', 'CD'), ('cans', 'NNS'), ('of', 'IN'), ('coca', 'NN'), ('cola', 'NN'), ('from', 'IN'), ('the', 'DT'), ('nearby', 'JJ'), ('grocer', 'NN')]** |
| NAMED ENTITY RECOGNITION | **(S She/PRP bought/VBD 10/CD apples/NNS and/CC 10/CD cans/NNS of/IN (NP coca/NN) (NP cola/NN) from/IN (NP the/DT nearby/JJ grocer/NN))** |

# DOCUMENT TERM MATRIX

1

**ORIGINAL STATEMENT**

D1: Natural language processing is fun!
D2: Natural language processing is not fun!
D3: Drinking beer is fun!

2

**PROCESSED STATEMENT**

D1: natur languag process fun
D2: natur languag process fun
D3: drink beer fun

3

**VECTOR OUTPUT**

| | natur | languag | process | fun | drink | beer |
|---|---|---|---|---|---|---|
| D1 | 1 | 1 | 1 | 1 | | |
| D2 | 1 | 1 | 1 | 1 | | |
| D3 | | | | 1 | 1 | 1 |

Final vectors:
D1: (1,1,1,1,0,0)
D2: (1,1,1,1,0,0)
D3: (0,0,0,1,1,1)

# TERM FREQUENCY VS. TERM FREQUENCY – INVERSE DOCUMENT FREQUENCY

- TERM FREQUENCY (TF)
  - Frequency of the term in the document
  - i.e. if the word appears twice, the frequency in the vector will be 2

- TERM FREQUENCY - INVERSE DOCUMENT FREQUENCY (TF-IDF)
  - Words that appear across multiple documents are less important (less discriminative)
  - Give higher weightage to words that appear less
  - $IDF(W) = log\frac{N}{df(W)}$
  - N = Number of documents
  - df(W) = Number of documents the word appears in
  - $TF - IDF\ (W) = TF(W) \times IDF(W)$

$IDF(W) = log\frac{100}{20}$

$TF - IDF\ (W) = 25 \times log\frac{100}{20}$

100 movie reviews
20 on movie reviews
'Avengers' □ 25 times

## Hands-on

- 01 NLU.ipynb

# Your Feedback Matters!

bit.ly/3hmJ3Nr