Intro to Natural Language Processing

Pragya Paudyal

Data Science Lecturer

3rd May 2023

Interactivity

- Please utilise the slido for both this and the case studies with external speakers.

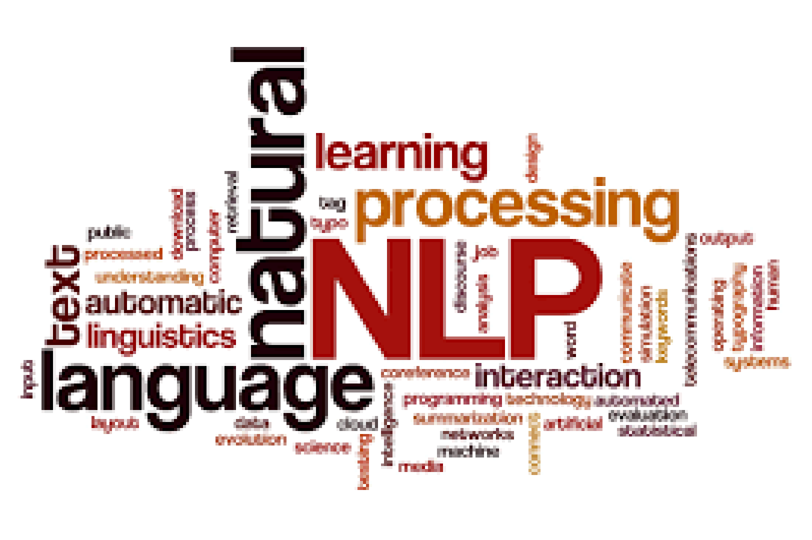
- You can also email Data.Science.Campus.Faculty@ons.gov.uk

slido

#DSGP

What is NLP?

Natural Language Processing (NLP) is a branch of AI that gives computers the ability to interpret, manipulate and comprehend human language.



Why do we need NLP?

- Organizations have an unparalleled volume of unstructured voice and text data to potentially benefit from.

- It is NLP analysis and modelling that allows us to harness the potential hidden within and generate value for the public good.

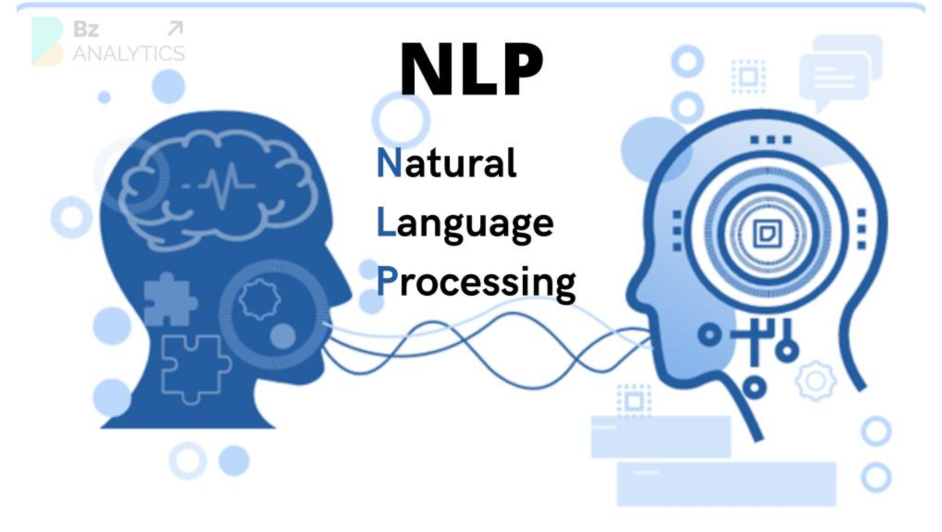
Challenges in the NLP Field

- Ambiguity – multiple possible interpretations of language.
- Synonyms and Homonyms – similarities and exactness in languages leading to different resolution.
- Irony and Sarcasm
- Slang

Where is NLP used?

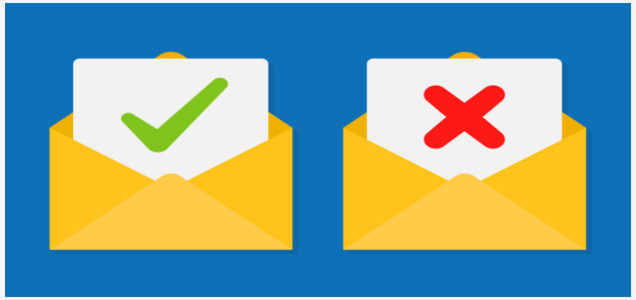
- Can you think of some examples that we take for granted in 2024?

- Think language, voice, text etc.

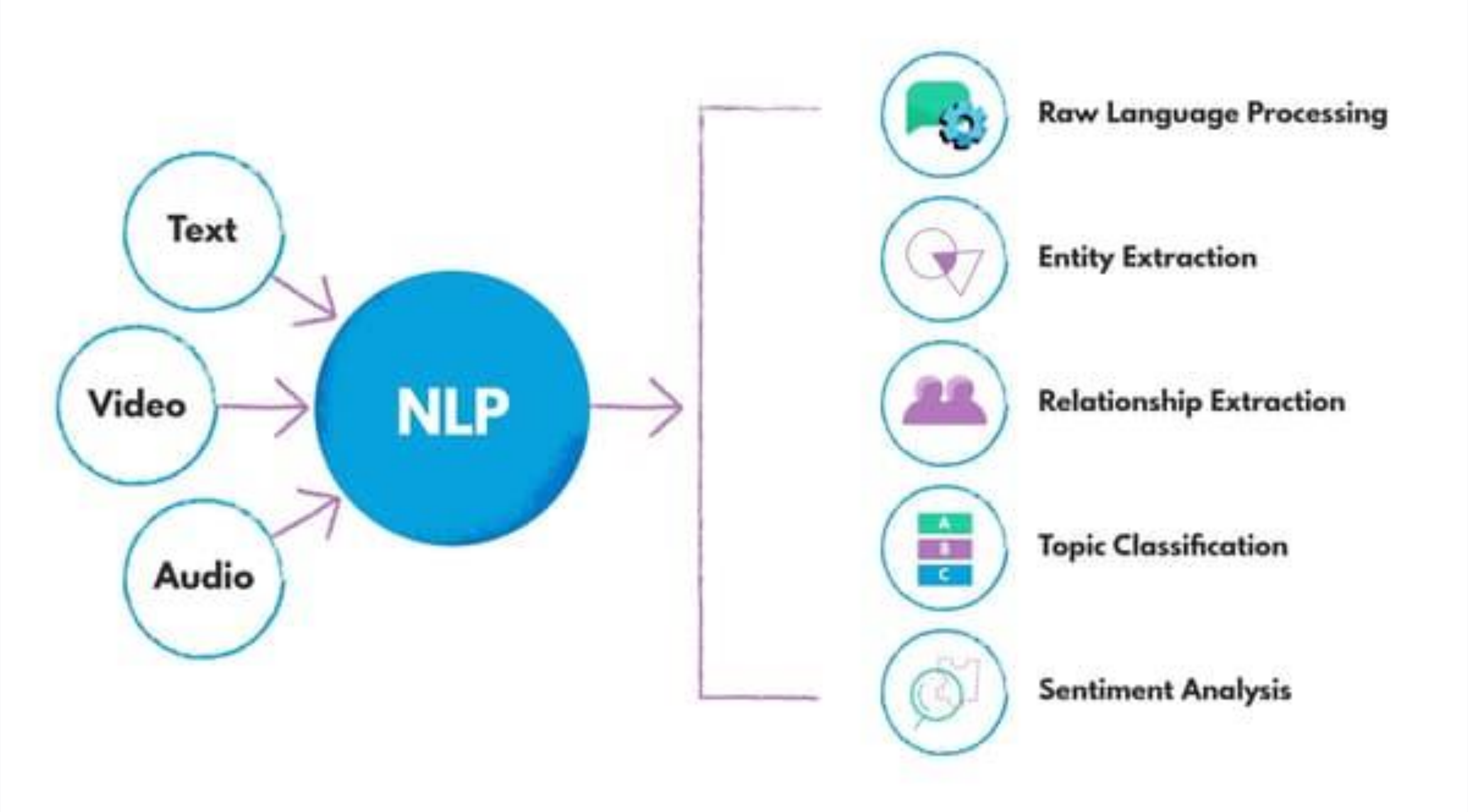


NLP applications in everyday life

- Email filters
- Smart Assistants
- Search Results
- Predictive Text
- Language Translation
- Text Analytics
- Spell check
- Voice to text
- Chatbots

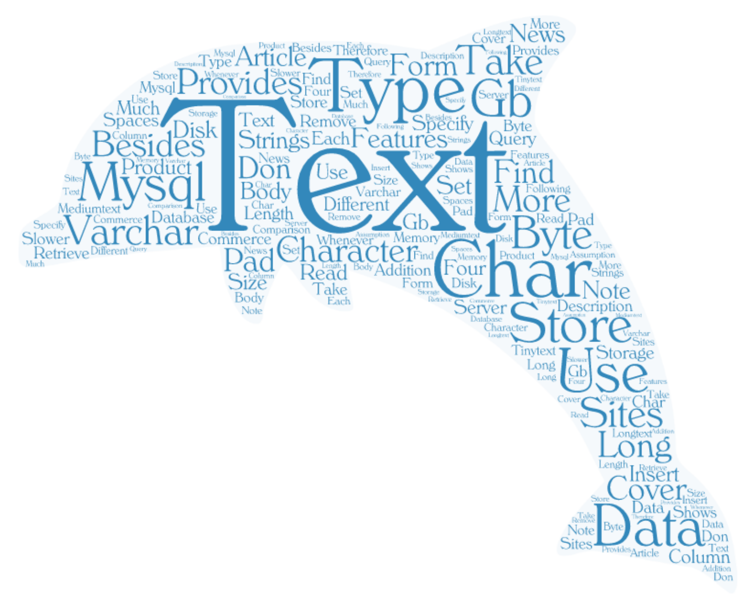


NLP – Higher Level



NLP – module focus

- NLP applications comprise many types of unstructured data.
- Audio and video data are more complex and often require distributed computing and deep learning approaches.
- We will focus on text data in this module, the best place to start with NLP.



Preprocessing

- Before linguistic analysis/modelling, we prepare the unstructured text data.

- Structured process, where **order matters** in the formulaic steps applied.

- Mixture of common string manipulations and unique NLP methods.

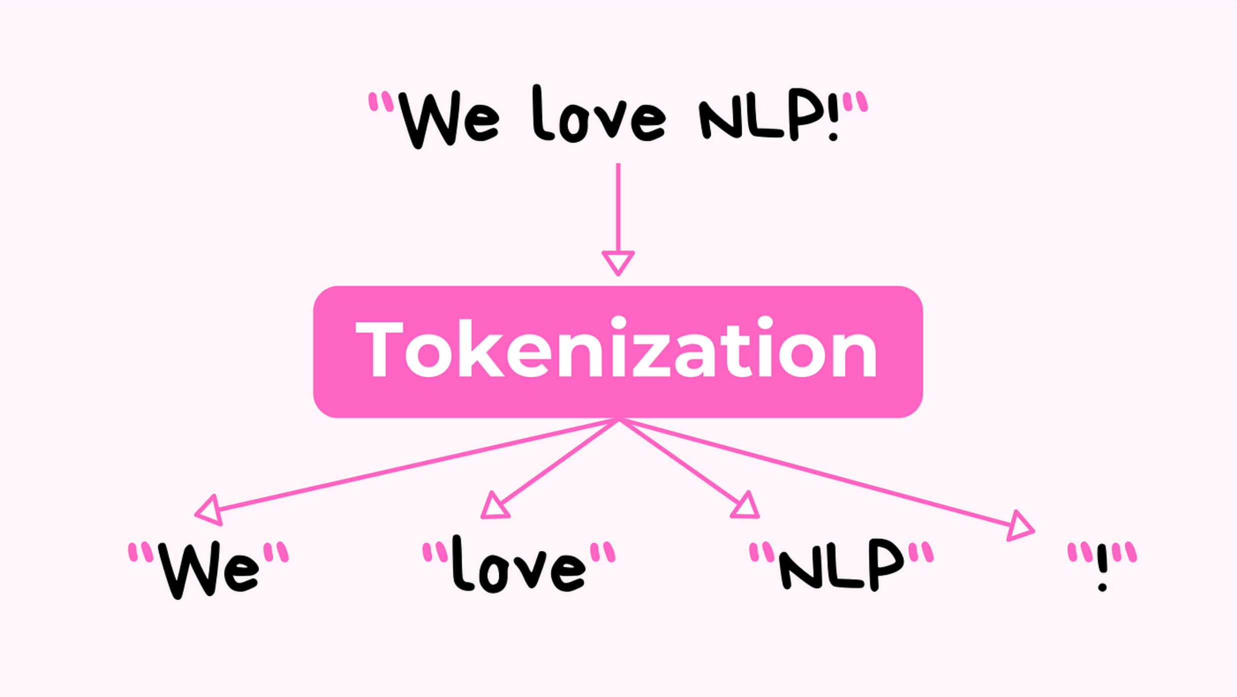
Preprocessing - Tokenization

- Often the first step of the cleaning pipeline.
- Process of splitting text into individual **tokens.**
- A token is a meaningful unit of text, often **words or sentences**.

Tokenization Visually

- Essential, as it separates linguistic elements like **punctuation**.

- We can then apply further cleaning steps to address **noise**.



Noise in text data

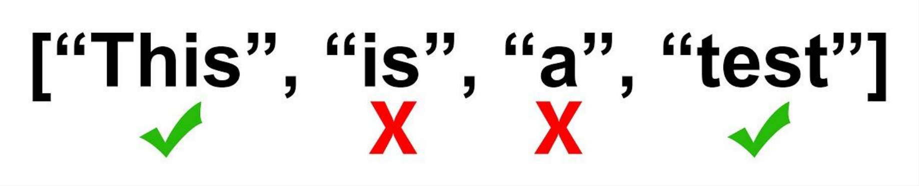
- Noise is multi-faceted in text data.

- Our language is so complex, with many connectives, clauses, context clues, punctuation and so on.

- How we deal with these depends on the problem to solve.

Pre-processing – Stop words

- A very important step when cleaning text is to remove **stop words** from the dataset.
- These are common words any language such as “the”, “and” “it”, “is”.
- There are many dictionaries of these that require careful considerations on which to use.

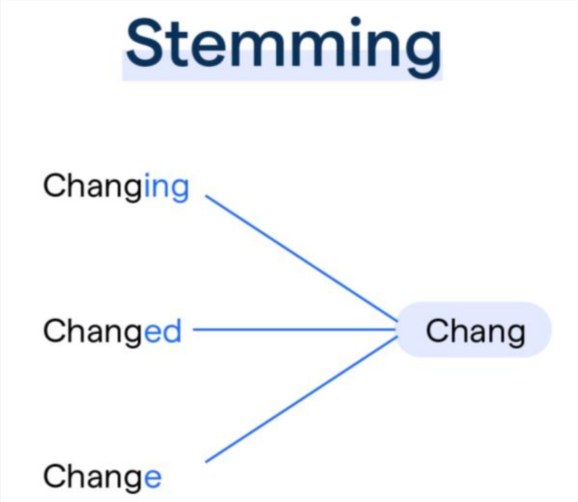


[“This”, “is”, “a”, “test”]

Pre-processing – Stemming & Lemmatization

- Stemming reduces words to their root form (known as a stem) even if this stem **has no meaning**.

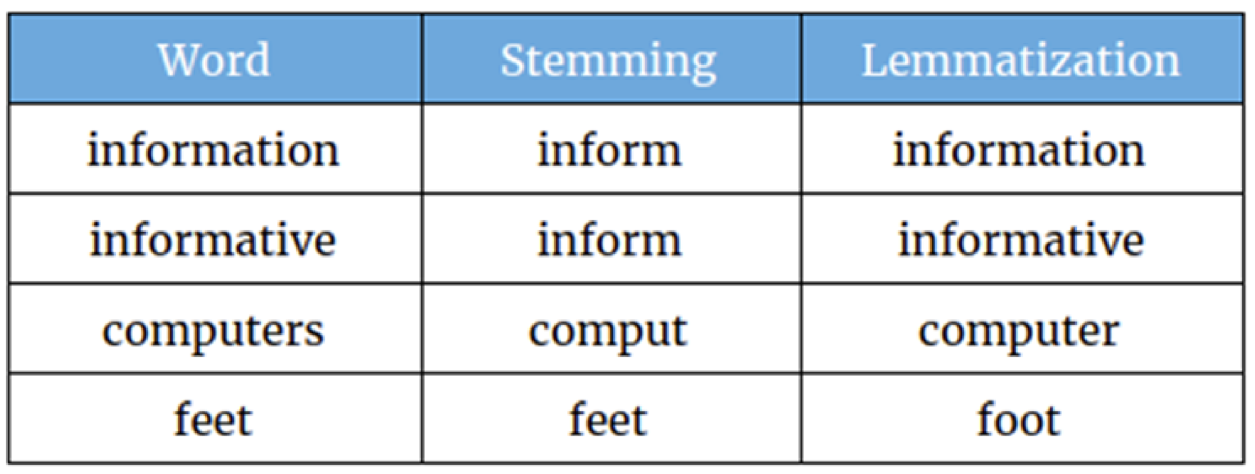
- Lemmatization reduces words to their root **dictionary form** (known as a lemma) a.k.a, the root word has meaning and we removing suffixes only.



Example – Stems vs. Lemmas

- Stemming can be quite crude and has less overall application.

- However, they are very useful when the meaning behind words is unimportant.



Word	Stemming	Lemmatization
information	inform	information
informative	inform	informative
computers	comput	computer
feet	feet	foot

Pre-processing – Common Steps

Numerous important techniques are employed during this process, such as:

- Lower-casing the text
- Removing punctuation
- Removing numbers
- Removing whitespace
- Fixing mis-spelled words (much harder!)

Pre-processing – why is order important?

- Remember that programming languages are **case-sensitive**.
- Alphanumeric characters are ordered A-Za-z (or A to z) with A-Z (1-26) and a-z (27-52).
- This means **The** and **the** are unique values, and only one appears in the stop words dictionary.

Regular Expressions

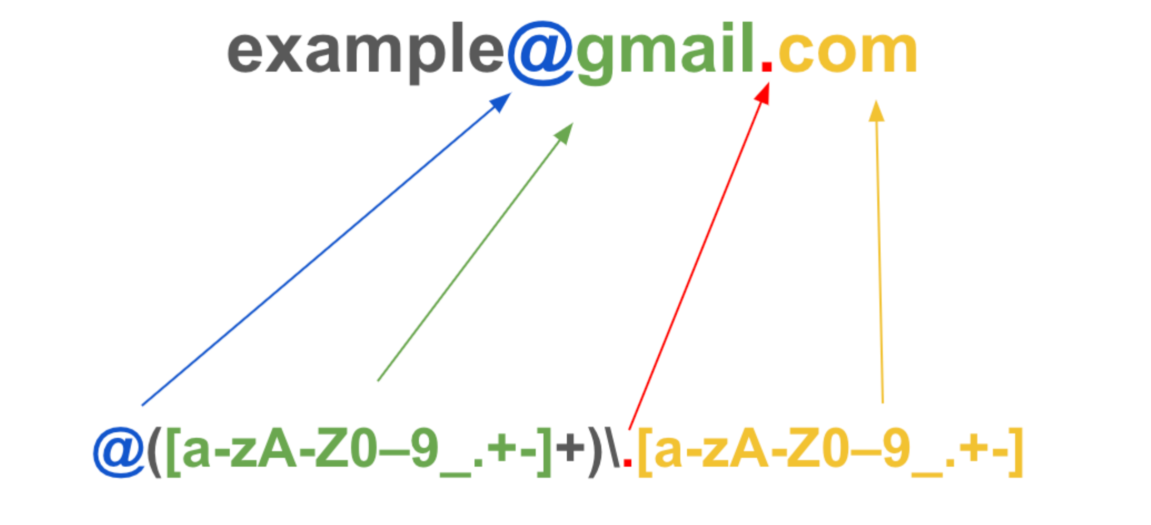
- Whilst cleaning, we will perform more complex editing using **pattern matching** functions on strings (think find and replace).
  - For example, we may want to convert “Natural Language Processing” to “NLP”.

- However, with more general searches (emails that end in a specific domain perhaps) we need to generalise our pattern matching.
  - To accomplish this, we need **Regular Expressions (Regex)**.

Why are regex useful?

- We can match any known sequence of characters by adding another component to our **pattern** that we ask the string functions to find matches for.

- This takes some getting used to syntax wise but can be applied to **any** text you are working with.



example@gmail.com

`@([a-zA-Z0–9_.+-]+)\.[a-zA-Z0–9_.+-]`

An Amazing Regex Resource!

- Chapter 2 delves deeper into the construction of Regex and is an essential resource to refer to.

- We recommend the fantastic regex101 website. This lets you:
    - Build regex and test the matches.
    - Obtain explanations of matches and mis-matches
    - Refer to dictionaries of regex elements at any time

Text Mining

Following a cleaning process, we start text mining.

This is the study of the **structural relationships between words**, approaches include:

- Exploratory analysis
- Summarization
- Categorization

Why is Text Mining Useful?

- Forms the essential process of converting **unstructured** text data to **semi-structured** data.
- Aims to identify patterns, themes and topics of interest from information sources.
- Used to measure customer opinions, product reviews and feedback which supports fact-based decision making.

Text Mining – Breaking it Down

- We attempt to separate valuable keywords from a mass of other words and use them to identify meaningful patterns or make predictions.

- Following this we can apply analytical techniques such as Cluster Analysis and Classification.

- One such method is the **Bag of Words** approach.

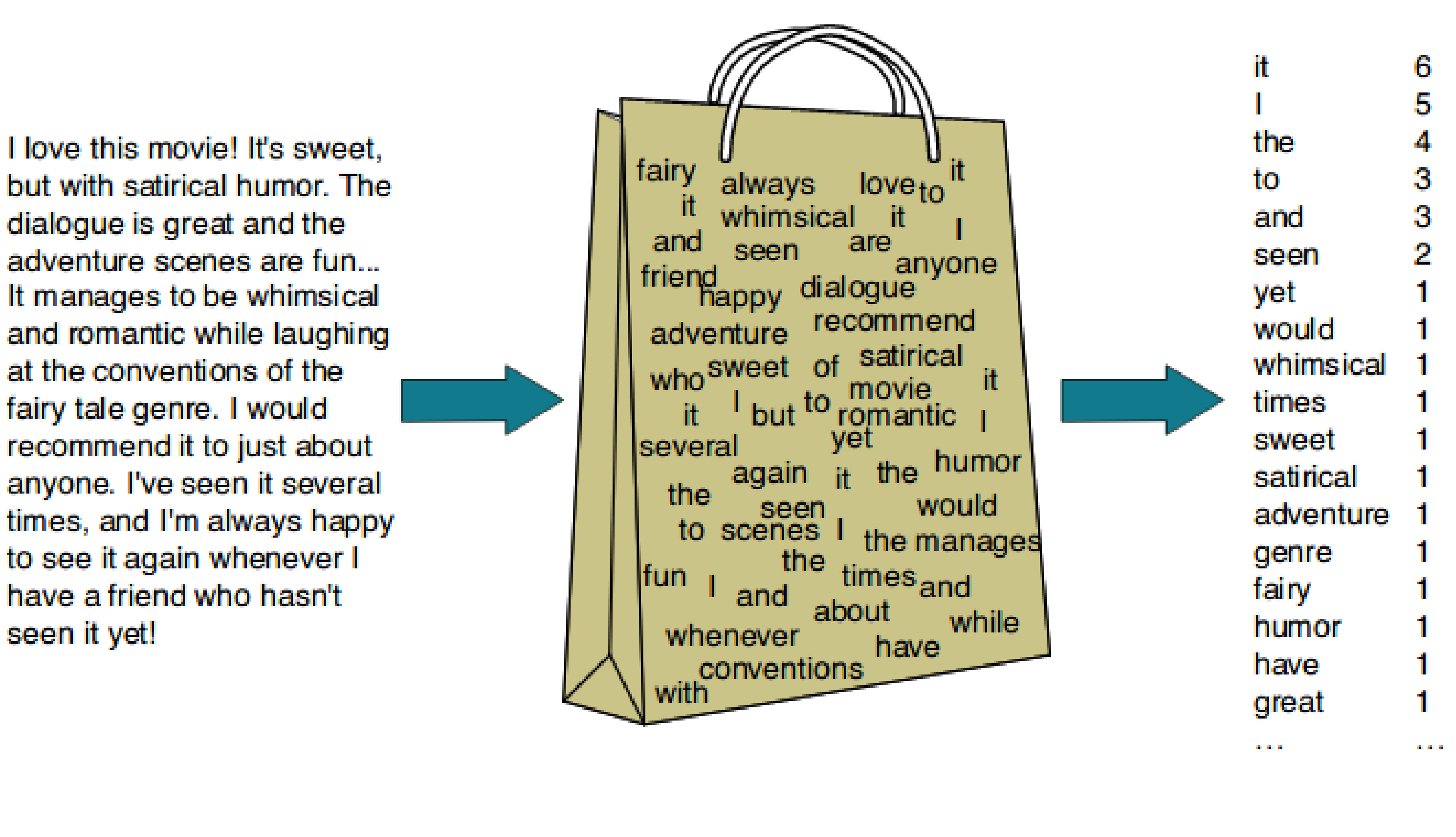
A bag of words approach

- A bag-of-words is a **text representation** method that describes the **occurrence** of words within a document.

- It is called a “bag” of words, because order and structure of words are ignored.

- It aims to answer whether words occur (and how often) rather than where they occur.

Bag of Words - Visually



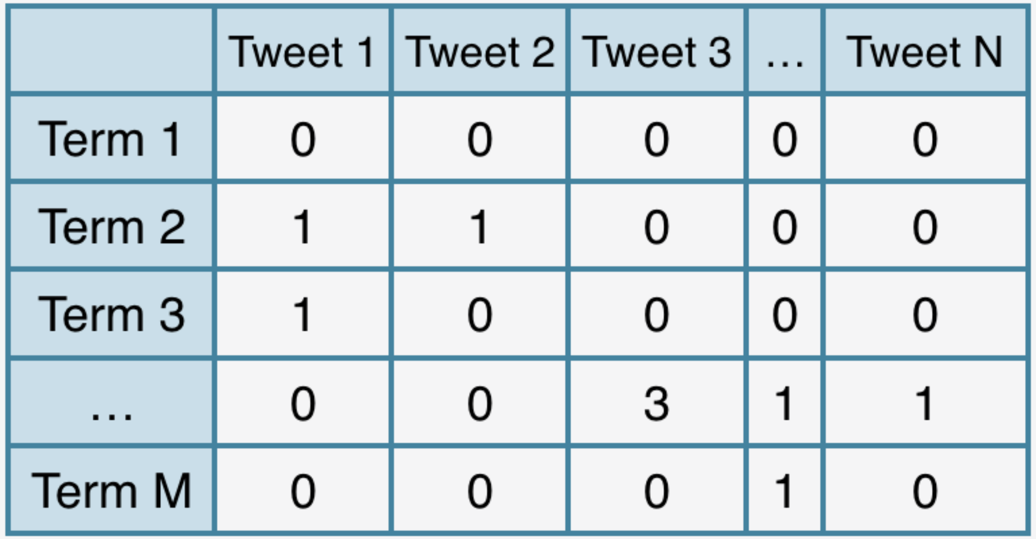
I love this movie! It's sweet, but with satirical humor. The dialogue is great and the adventure scenes are fun... It manages to be whimsical and romantic while laughing at the conventions of the fairy tale genre. I would recommend it to just about anyone. I've seen it several times, and I'm always happy to see it again whenever I have a friend who hasn't seen it yet!

it	6
I	5
the	4
to	3
and	3
seen	2
yet	1
would	1
whimsical	1
times	1
sweet	1
satirical	1
adventure	1
genre	1
fairy	1
humor	1
have	1
great	1
...	...

Bag of Words – New Terminology

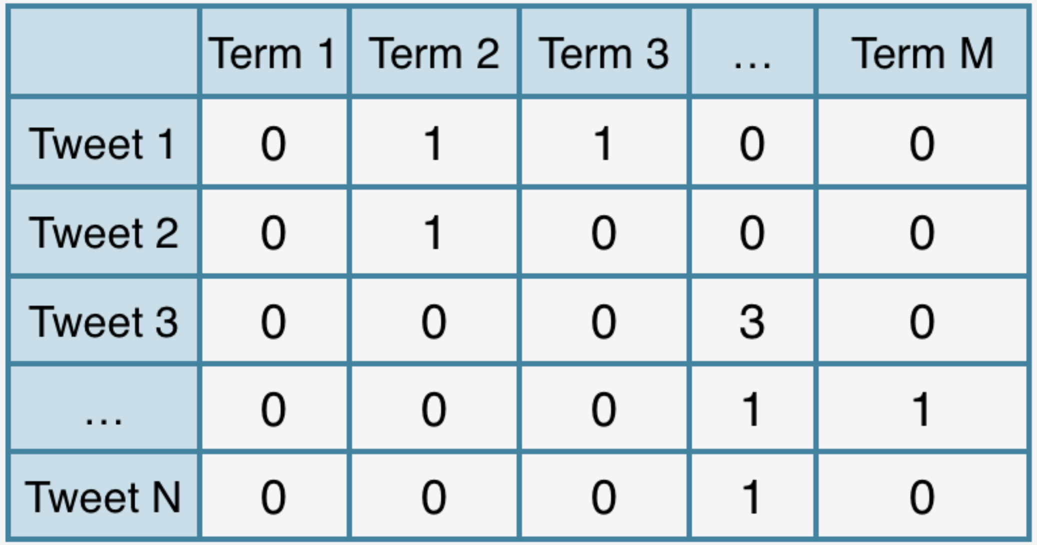
- This approach is usually used when we have a collection of **documents**, each of which are bodies of text.
- For bag of words, we create a special object known as a **corpus,** or a collection of documents.
- From a corpus, we create special data structures which are transposes of each other:
  - Document-Term Matrix
  - Term-Document Matrix

Bag of Words – Matrices



	Tweet 1	Tweet 2	Tweet 3	…	Tweet N
Term 1	0	0	0	0	0
Term 2	1	1	0	0	0
Term 3	1	0	0	0	0
…	0	0	3	1	1
Term M	0	0	0	1	0

Term Document Matrix (TDM)



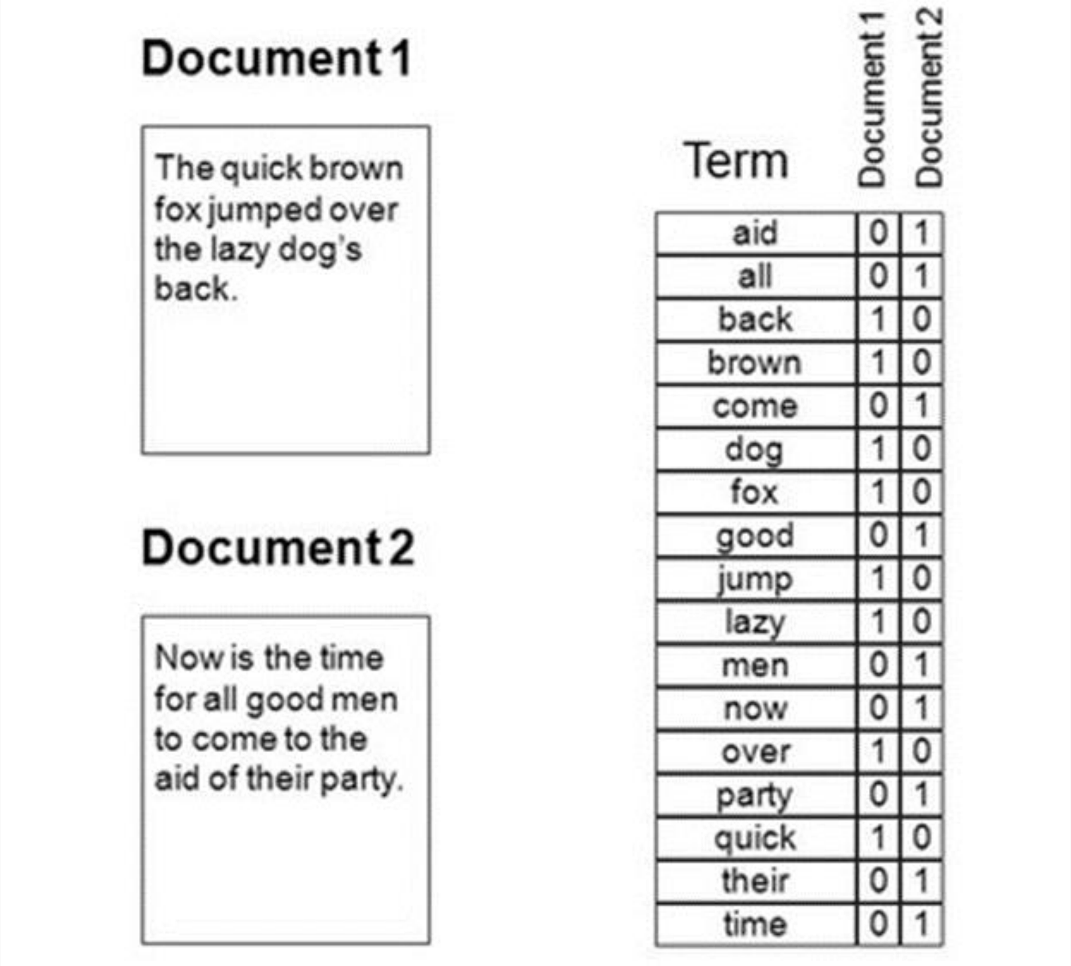
	Term 1	Term 2	Term 3	…	Term M
Tweet 1	0	1	1	0	0
Tweet 2	0	1	0	0	0
Tweet 3	0	0	0	3	0
…	0	0	0	1	1
Tweet N	0	0	0	1	0

Document Term Matrix (DTM)

Example - TDM

- Notice that in smaller corpora such as this (2 documents), we get numerous binary vectors.

- Sometimes this can lead to the issue of **sparsity**.



Document 1

The quick brown fox jumped over the lazy dog's back.

Document 2

Now is the time for all good men to come to the aid of their party.

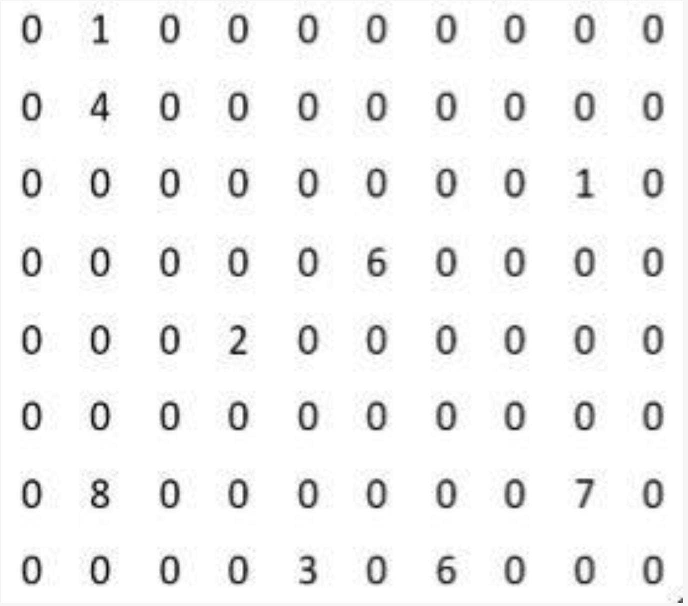
Term	Document 1	Document 2
aid	0	1
all	0	1
back	1	0
brown	1	0
come	0	1
dog	1	0
fox	1	0
good	0	1
jump	1	0
lazy	1	0
men	0	1
now	0	1
over	1	0
party	0	1
quick	1	0
their	0	1
time	0	1

Bag of Words - Sparsity

- Since each document can have hundreds of thousands of terms, we expect the matrices to be incredibly large.

- Inherently, we get **sparse** vectors, those with a high percentage of 0s.

- These are difficult for traditional algorithms to model efficiently.



0	1	0	0	0	0	0	0	0	0
0	4	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	1	0
0	0	0	0	0	6	0	0	0	0
0	0	0	2	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	8	0	0	0	0	0	0	7	0
0	0	0	0	3	0	6	0	0	0

Bag of Words - Application

- Consider a search engine which extracts all documents (a.k.a webpages/HTMLs) that match the keyword(s) typed.

- How does the search engine know which webpages to return?

More on Search Engines

- Engines use a page ranking algorithm alongside text mining to identify the **most relevant** web pages.
- They follow a logic that gives a high weighting to **rare** keywords with high frequency of appearance.
- These lexicons of rare words are unique to each search engine.

Bag of Words - Advantages

- Simple to understand and implement using popular packages.

- Highly flexible customisation of the corpora created.

- Bag of Words is often the **baseline model** or simpler model to start a linguistic modelling problem.

Bag of Words - Disadvantages

- **Vocabulary**: Needs to be managed so the size doesn’t get out of hand.
- **Sparsity**: Sparse representations are harder to model and draw information from.
- **Meaning**: We lost context and sentiment behind the words, which can provide useful information.

Complexity of the human language

- Teaching a machine to analyze the various grammatical nuances of language is incredibly complex.

- Teaching a machine to understand how **context can affect tone** is even more difficult.

- **Sentiment Analysis** is the technique that attempts to do this.

Sentiment Analysis – What is it?

- Most brands use social media and feedback mechanisms like surveys to analyse opinions about their products.

- Sentiment Analysis techniques are usually employed to identify three things:
  - Polarity of the expression
  - Subject of the expression
  - Opinion Holder of the expression

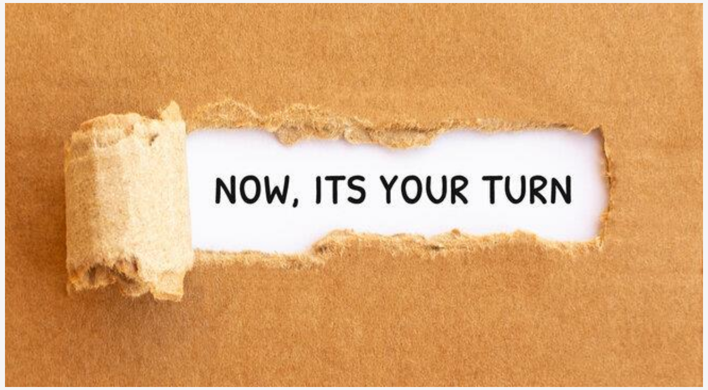
Sentiment Analysis – Definitions

- **Polarity**: Is the opinion positive or negative (some include neutrality).
- **Subject**: The thing that is being talked about.
- **Opinion Holder**: The person, or entity that expresses the opinion.

Sentiment Analysis – Exercise

Take the following three sentences and state the polarity, subject and opinion contained.

1. I love flying British Airways because they have the best food.
2. The Fiat Punto is the ugliest car I've ever seen.
3. I love this phone but wouldn't recommend it to my friends.



Sentiment Analysis – Answers

1. Polarity – Positive, Subject – British Airways, Opinion Holder – Customer

2. Polarity – Negative, Subject – Fiat Punto, Opinion Holder – Reviewer

3. Polarity – Neutral?, Subject – Phone in question, Opinion Holder – Phone owner

Sentiment Analysis - Efficiency

- Doing Sentiment Analysis by hand is clearly possible but is time consuming.

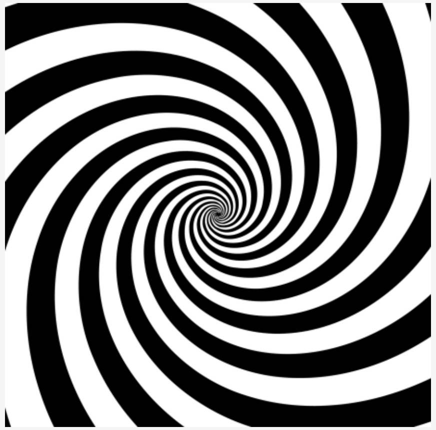
- Sentiment analysis models can determine the polarity in the text much more efficiently.

Sentiment Analysis - Considerations

- Efficient does not mean perfect in this case.

- We need to construct specific elements for **linguistic devices** for more accurate sentiment scanning, such as:
  - Context based tagging
  - N-grams (pairs, trios etc of words)

- We end up with very flexible models, but when do we stop? No easy answer.



Sentiment Analysis – Harder Example

Penny stopped at Costa Coffee on her way home. She thought a coffee was good every few hours, but it turned out to be too expensive there.

- Multiple sentiment and contextual threads here. Some aggregation would be useful!

Sentiment Analysis - Approaches

- Knowledge-Based: Categorize text based on unambiguous “affect words” like **love, like, hate** and so on.

- Statistical Methods: Model detects the sentiment holder as well as the subject in the sentence.

- Hybrid Approaches: A combination of the two with additional linguistic techniques to pick up semantics.

Sentiment Analysis – Lexicon Methods

This knowledge-based technique is an incredibly popular baseline approach.

It uses a lexicon of pre-defined positive and negative words and matches these to the text.

A **sentiment score** is calculated, averaging out the numbers of positive and negative matches.

Sentiment Analysis – Sentiment Scores

- This is the score that determines the sentiment classification of the text. It is calculated as:

sum(positives) – sum(negatives)

- Once calculated, threshold checks are made and a classification is provided (pos, neg, neut etc).

Sentiment Analysis – Lexicon Issues

- Lexicons are largely crowd-sourced and validated by volunteers, leading to a possible lack of credible peer review.

- However, there are so many neutral words in language that some are likely missing.

- Lexicons don’t consider qualifiers **before** the word, which can **invert** the sentiment, for example “no good”.

Sentiment Analysis – Considerations

- We may hesitate to apply these dictionaries to historical documents, since the language is so different.
  - Analysing a Jane Austen novel would be quite different!

- Applying this on large chunks of text usually averages out the sentiment, so tokenisation becomes even more paramount here!

Topic Modelling

- **Unsupervised** NLP technique, gives us an idea of what text is about quickly.
- A **topic** is a label or collection of words that often occur together.
- For example, consider the topic **weather**, what words would be associated with this topic?

Topic Modelling - Latent Dirichlet Allocation

- Very popular algorithm that takes a Document Term Matrix (DTM) as it’s input.

- Outputs matrices, one with the **prevalence of topics in documents**, and the other with the **probabilities of words belonging to the topics**.

- As it is unsupervised, we provide a value of K topics to look for beforehand, usually informed with knowledge of the data.

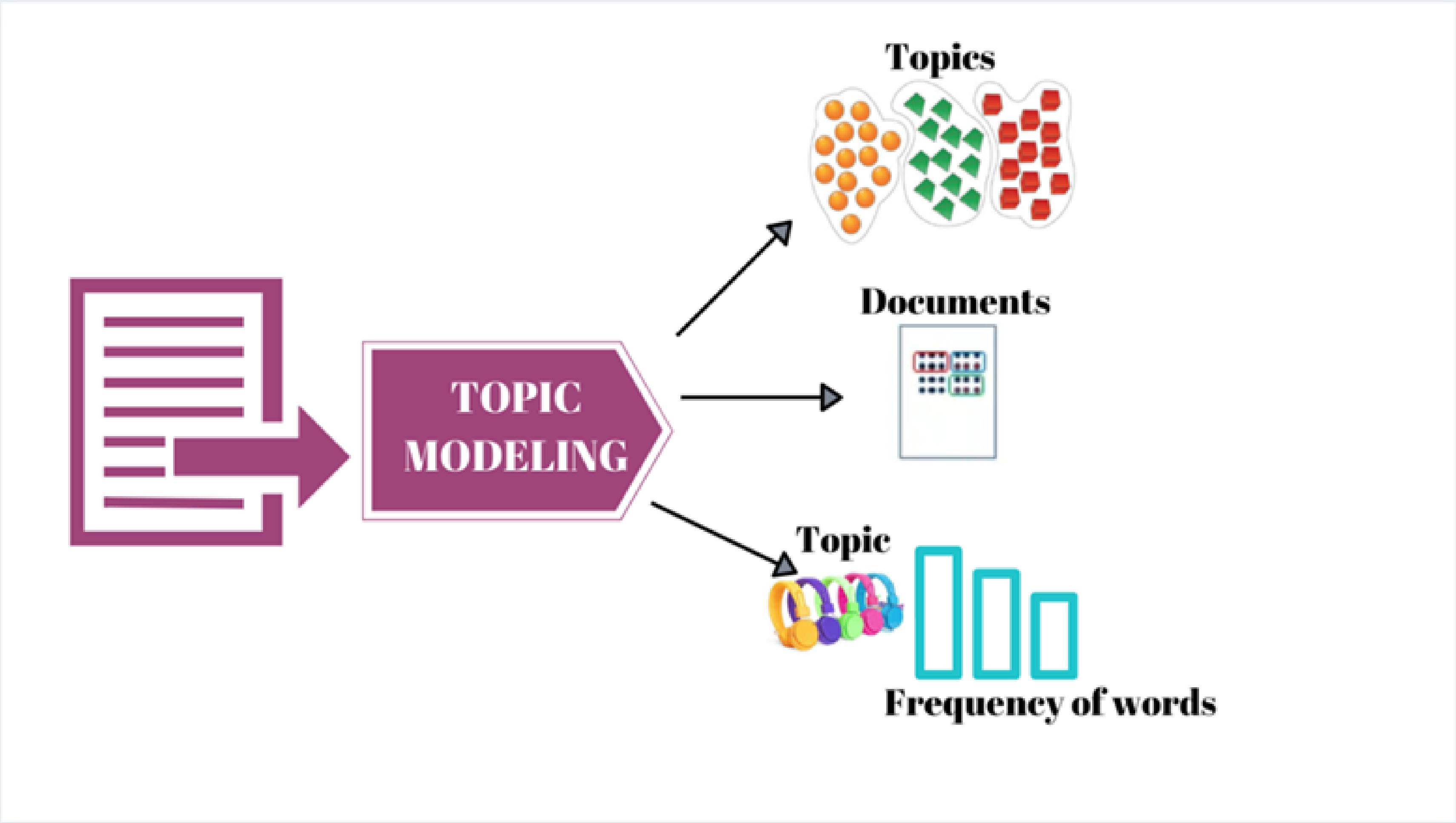
LDA – Thought Process

- **Every document is a mixture of topics**. For example, in 2 topic models we could say “Document 1 is 90% topic A and 10% topic B” etc.

- **Every topic is a mixture of words**. For example, we could imagine a two-topic model for news, one for “politics” and another “entertainment” and so on.

- This is essentially **probabilistic clustering**, but a little fuzzy as documents can be associated with more than one topic.

LDA - Visually



Topic Modelling - Considerations

- We must ensure stop words are dealt with beforehand, as they will likely make up most of the corpus.

- LDA estimates two probabilities simultaneously, which leads to long runtimes.

- When documents are often about the same subject matter, certain words become redundant as the corpus increases in size.

Resources

- Text Mining with R: A Tidy Approach by Julia Silge & David Robinson
- Analyzing Text with the Natural Language Toolkit by Steven Bird, Ewan Klein and Edward Loper
- A Beginner's Guide to Latent Dirichlet Allocation (LDA) by Ria Kulshrestha on Towards Data Science.

Questions?