TOPIC MODELLING

**There are two types of topic analysis techniques**

* Topic modelling is an ‘unsupervised’ machine learning technique, in other words, one that doesn’t require training.
* Topic classification is a ‘supervised’ machine learning technique, one that needs training before being able to automatically analyze texts.

**Introduction to Topic Modeling**

This is known as ‘unsupervised’ machine learning because it doesn’t require a predefined list of tags or training data that’s been previously classified by humans. You could say that unsupervised techniques are a short-term or quick-fix solution, while supervised techniques are more of a long-term solution that will help your business grow.

Topic modelling, in the context of Natural Language Processing, is described as a method of uncovering hidden structure in a collection of texts.Topic modelling is a form of unsupervised learning that identifies hidden relationships in data. Topic modelling works in an exploratory manner, looking for the themes (or topics) that lie within a set of text data. There is no prior knowledge about the themes required in order for topic modelling to work.

Analytics Industry is all about obtaining the “Information” from the data. With the growing amount of data in recent years, that too mostly unstructured, it’s difficult to obtain the relevant and desired information. But, technology has developed some powerful methods which can be used to mine through the data and fetch the information that we are looking for.One such technique in the field of text mining is Topic Modelling. As the name suggests, it is a process to automatically identify topics present in a text object and to derive hidden patterns exhibited by a text corpus. Thus, assisting better decision making.

Natural language processing is the processing of languages used in the system that exists in the library of “nltk” where this is processed to cut, extract and transform to new data so that we get good insights into it. It uses only the languages that exist in the library because NLP-related things exist there itself so it cannot understand the things beyond what is present in it.Large amounts of data are collected every day. As more information becomes available, it becomes difficult to get holdup on what we are looking for. So, we need tools and techniques to organize, search and understand vast quantities of information.Even though, topic Modeling has been used to group large amounts of documents, few applications of Topic Modeling have been used on research papers, and a researcher is required to have programming skills and statistical knowledge to successfully conduct an exploratory literature review using Topic Modeling.Topic modeling is recognizing the words from the topics present in the document or the corpus of data. This is useful because extracting the words from a document takes more time and is much more complex than extracting them from topics present in the document.Topic modelling is used when you have a set of text documents (such as emails, survey responses, support tickets, product reviews, etc), and you want to find out the different topics that they cover and group them by those topics.

For example, let’s say you’re a software company and you want to know what customers are saying about particular features of your product. Instead of spending hours going through heaps of feedback, in an attempt to deduce which texts are talking about your topics of interest, you could analyze them with a topic modeling algorithm.

Topic Modelling is different from rule-based text mining approaches that use regular expressions or dictionary based keyword searching techniques. It is an unsupervised approach used for finding and observing the bunch of words (called “topics”) in large clusters of texts.Topic modelling is a machine learning technique that automatically analyses text data to determine cluster words for a set of documents. This is known as ‘unsupervised’ machine learning because it doesn’t require a predefined list of tags or training data that’s been previously classified by humans. Topic modelling provides us with methods to organize, understand and summarize large collections of textual information.

Topic Modelling refers to the process of dividing a corpus of documents in two:

* A list of the topics covered by the documents in the corpus
* Several sets of documents from the corpus grouped by the topics they cover.

In unsupervised machine learning algorithms such as topic modelling require less manual input than supervised algorithms. That’s because they don't need to be trained by humans with manually tagged data. However, they do need high-quality data, and not only that – they need it in bucket loads, which may not always be easy to come by.At the end of your topic modelling analysis, you’ll receive collections of documents that the algorithm has grouped together, as well as clusters of words and expressions that it used to infer these relations.If you don’t have a lot of time to analyze texts, or you’re not looking for a fine-grained analysis and just want to figure out what topics a bunch of texts are talking about, you’ll probably be happy with a Topic Modelling algorithm.

Example:Intuitively, given that a document is about a particular topic, one would expect particular words to appear in the document more or less frequently: "dog" and "bone" will appear more often in documents about dogs, "cat" and "meow" will appear in documents about cats, and "the" and "is" will appear approximately equally in both. A document typically concerns multiple topics in different proportions; thus, in a document that is 10% about cats and 90% about dogs, there would probably be about 9 times more dog words than cat words.

A good topic model should result in – “health”, “doctor”, “patient”, “hospital” for a topic – Healthcare, and “farm”, “crops”, “wheat” for a topic – “Farming”.

### Topic modeling APIs: Application programming interfaces (APIs) are a great way to seamlessly connect applications and extend the functionality of your apps. Luckily, there are plenty of topic modelling tools with their own API, and various languages in the data science community that are ideal for these machine learning models. Let’s take a closer look:

#### Open source- If you know how to code, there are many open source libraries for implementing a topic modelling solution from scratch. These are great because they offer flexibility and customization, and give you complete control of the whole process – from the pre-processing of data (tokenization, stopwords removal, stemming, lemmatization, etc), to feature extraction and training of the model (choosing the algorithm and its parameters).

**Applications of topic modelling**

* It helps in discovering hidden topical patterns that are present across the collection.
* It helps in Annotating documents according to these topics.
* It helps in using these annotations to organize, search and summarize texts. They are very useful for the purpose for document clustering, organizing large blocks of textual data, information retrieval from unstructured text and feature selection.
* Applications of Topic Models also reviews topic models’ ability to unlock large text collections for qualitative analysis. It reviews their successful use by researchers to help understand fiction, non-fiction, scientific publications, and political texts.
* Applications of Topic Models is aimed at the reader with some knowledge of document processing, basic understanding of some probability, and interested in many application domains.
* Topic modeling can be used in graph based models to obtain semantic relationship between words. It can be used in text summarization to quickly find out what the document or book is explaining about.
* It can be used in exam evaluation to avoid biasing towards candidates. It also saves a lot of time and helps students get their results quickly.
* It can provide improved customer service by identifying the keyword the customer is asking about and acting accordingly. This increases the trust of customers as they received the help needed at the right time without any inconvenience. This drastically improves the customer loyalty and in turn increases the value of the company. It can identify the keywords of search and recommend products to the customers accordingly.

# Limitations of topic modeling

* Fixed K (the number of topics is fixed and must be known ahead of time)
* Uncorrelated topics (Dirichlet topic distribution cannot capture correlations)
* Non-hierarchical (in data-limited regimes hierarchical models allow sharing of data)
* Static (no evolution of topics over time)
* Bag of words (assumes words are exchangeable, sentence structure is not modelled)
* Unsupervised (sometimes weak supervision is desirable, e.g. in sentiment analysis)

**Methods in Topic Modelling:**

This highly important process can be performed by various algorithms or methods. Some of them are:

* Non Negative Matrix Factorization (NMF)
* Latent Semantic Analysis (LSA)
* Parallel Latent Dirichlet Allocation (PLDA)
* Pachinko Allocation Model (PAM)
* Latent Dirichlet Allocation (LDA)

Still there are many research going on to improve the algorithms to understand the complete context of the documents.

* 1. Non negative matrix factorization: NMF is a matrix factorization method where we make sure that the elements of the factorized matrices are non-negative. Consider the document-term matrix obtained from a corpus after removing the stop words. The matrix can be factorized into two matrices term-topic matrix and topic-document matrix. There are many optimization models to perform the matrix factorization. Hierarchical Alternating Least Square is a faster and better way to perform NMF. Here the factorization occurs by updating one column at a time while keeping the other columns as constant.
  2. Parallel latent dirirchlet allocation: It is also known as Partially Labelled Dirichlet Allocation. Here, the model assumes that there exists a set of n labels and each of these labels are associates with each topics of the given corpus. Then the individual topics are represented as the probabilistic distribution of the whole of corpus similar to the LDA. Optionally, there could also be a global topic assigned to every document such that there are l global topics where l is the number of individual documents in the corpus. The method also assumes that there exists only one label for every topic in the corpus. With the labels given before developing the model, this process is very quick and precise compared to the above methods.

## Pachinko allocation model: Pachinko Allocation Model (PAM) is an improved method of Latent Dirichlet Allocation model. LDA model brings out the correlation between words by identifying topics based on the thematic relationships between words present in the corpus. But PAM improvises by modeling correlation between the generated topics. This model has greater power in determining the semantic relationship precisely as they also take into account of the relation between topics. The model is named after Pachinko, a popular game in Japan. The model makes use of Directed Acrylic Graphs to understand the correlation between topics. DAG is a finite directed graph to show how the topics are related.

## 

* 1. Latent semantic analysis: Latent Semantic Analysis is also an unsupervised learning method used to extract relationship between different words in a pile of documents. This aids us in choosing the correct documents required. It simply acts as a dimensionality method used to reduce the dimension of the huge corpus of text data. These unnecessary data acts as a noise in determining the correct insights from the data.

Latent Semantic Analysis (LSA) is one of the most frequent topic modelling methods analysts make use of. It is based on what is known as the [distributional hypothesis](https://en.wikipedia.org/wiki/Distributional_semantics) which states that the semantics of words can be grasped by looking at the contexts the words appear in. In other words, under this hypothesis, the semantics of two words will be similar if they tend to occur in similar contexts.

That said, LSA computes how frequently words occur in the documents – and the whole corpus – and assumes that similar documents will contain approximately the same distribution of word frequencies for certain words. In this case, syntactic information (e.g. word order) and semantic information (e.g. the multiplicity of meanings of a given word) are ignored and each document is treated as a bag of words.

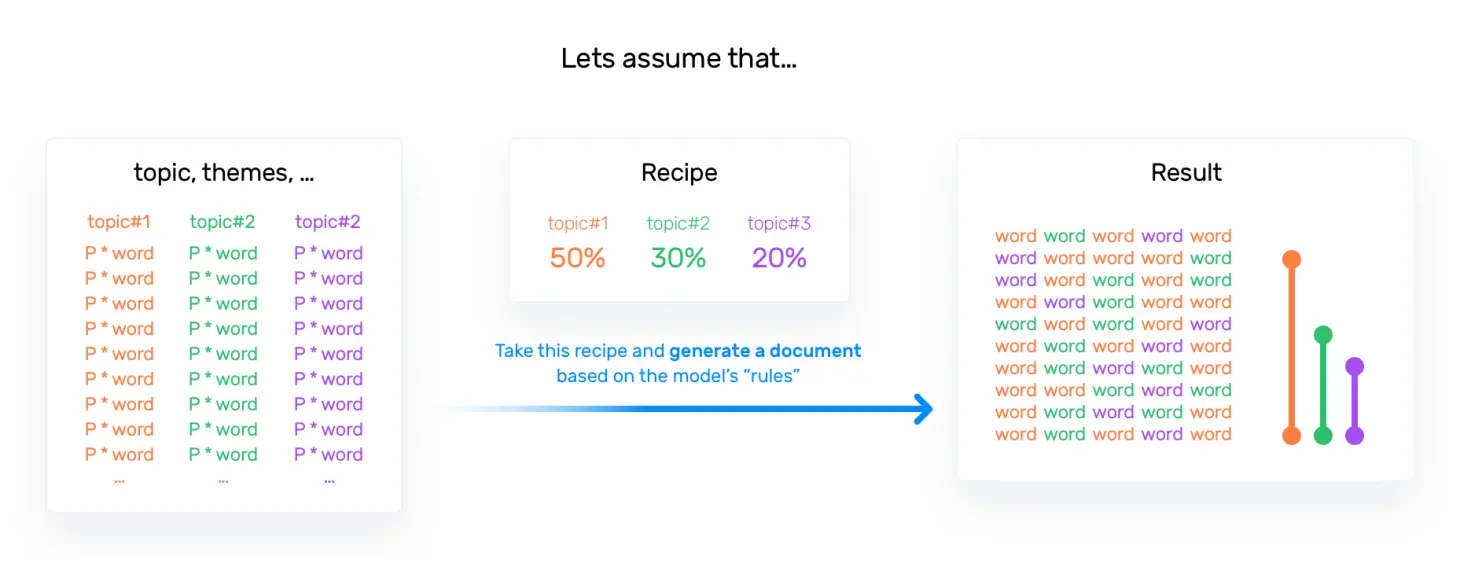
The standard method for computing word frequencies is what is known as [tf-idf](https://monkeylearn.com/blog/what-is-tf-idf/). This method computes frequencies by taking into consideration not only how frequent words are in a given document, but also how frequent words are in all the corpus of documents. Words with a higher frequency in the full corpus will be better candidates for document representations than less frequent words, regardless of how many times they appear in individual documents. As a result, tf-idf representations are much better than those that only take into consideration word frequencies at document level.

Once tf-idf frequencies have been computed, we can create a Document-term matrix which shows the tf-idf value for each term in a given document. This matrix will have rows for every document in the corpus and columns for every term considered.

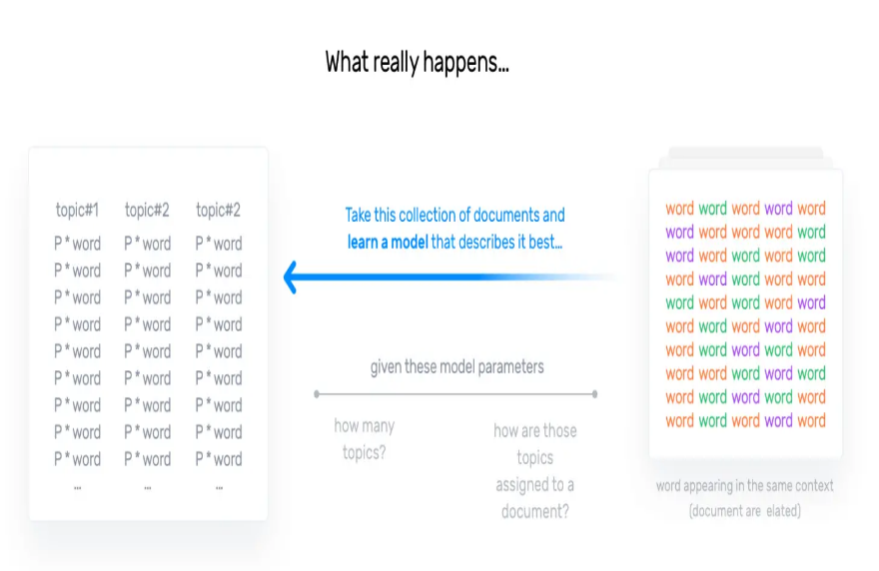
## Latent dirichlet allocation (lda): Latent Dirichlet Allocation (LDA) and LSA are based on the same underlying assumptions: the distributional hypothesis, (i.e. similar topics make use of similar words) and the statistical mixture hypothesis (i.e. documents talk about several topics) for which a statistical distribution can be determined. The purpose of LDA is mapping each document in our corpus to a set of topics which covers a good deal of the words in the document.

What LDA does in order to map the documents to a list of topics is assign topics to arrangements of words, e.g. n-grams such as best player for a topic related to sports. This stems from the assumption that documents are written with arrangements of words and that those arrangements determine topics. Yet again, just like LSA, LDA also ignores syntactic information and treats documents as bags of words. It also assumes that all words in the document can be assigned a probability of belonging to a topic. That said, the goal of LDA is to determine the mixture of topics that a document contains.

In other words, LDA assumes that topics and documents look like this:



And, when LDA models a new document, it works this way:



[What's the difference between Latent Semantic Indexing (LSI) and Latent Dirichlet Allocation (LDA)?](https://www.quora.com/Whats-the-difference-between-Latent-Semantic-Indexing-LSI-and-Latent-Dirichlet-Allocation-LDA)

LSI (also known as Latent Semantic Analysis, LSA) learns latent topics by performing a matrix decomposition (SVD) on the term-document matrix.LDA is a generative probabilistic model, that assumes a Dirichlet prior over the latent topics.In practice, LSI is much faster to train than LDA, but has lower accuracy.

The main difference between LSA and LDA is that LDA assumes that the distribution of topics in a document and the distribution of words in topics are Dirichlet distributions. LSA does not assume any distribution and therefore, leads to more opaque vector representations of topics and documents.

### Parameters of LDA: Alpha and Beta Hyperparameters – alpha represents document-topic density and Beta represents topic-word density. Higher the value of alpha, documents are composed of more topics and lower the value of alpha, documents contain fewer topics. On the other hand, higher the beta, topics are composed of a large number of words in the corpus, and with the lower value of beta, they are composed of few words.

A third hyperparameter has to be set when implementing LDA, namely, the number of topics the algorithm will detect since LDA cannot decide on the number of topics by itself.

The output of the algorithm is a vector that contains the coverage of every topic for the document being modelled. It will look something like this [0.2, 0.5, etc.] where the first value shows the coverage of the first topic, and so on. If compared appropriately, these vectors can give you insights into the topical characteristics of your corpus.

For more information on how those probabilities are computed, the statistical distributions assumed by the algorithm, or how to implement LDA, you can refer to [the original LDA paper](http://www.jmlr.org/papers/volume3/blei03a/blei03a.pdf).

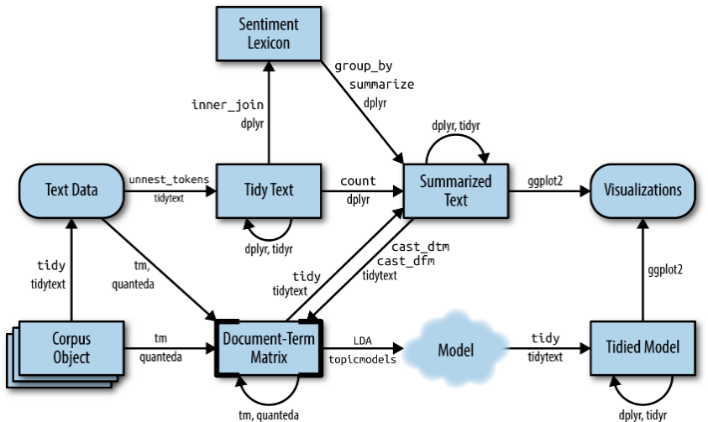
Also, for more information on how to compare vector representations to get insights into document similarity or the distribution of topics over a document corpus, you might want to read about [cosine similarity](https://en.wikipedia.org/wiki/Cosine_similarity) or other [similarity measures](https://en.wikipedia.org/wiki/Similarity_measure). All of these comparisons can be used to compute similarities between the output vectors of both LSA and LDA.

Number of Topics – Number of topics to be extracted from the corpus. Researchers have developed approaches to obtain an optimal number of topics by using Kullback Leibler Divergence Score.

Number of Topic Terms – Number of terms composed in a single topic. It is generally decided according to the requirement. If the problem statement talks about extracting themes or concepts, it is recommended to choose a higher number, if problem statement talks about extracting features or terms, a low number is recommended.

Number of Iterations / passes – Maximum number of iterations allowed to LDA algorithm for convergence.

Latent Dirichlet allocation is one of the most common algorithms for topic modeling.



A-**Every document is a mixture of topics.**

**B-Every topic is a mixture of words.**

[library](https://rdrr.io/r/base/library.html)(topicmodels)

We can use the [LDA()](https://rdrr.io/pkg/topicmodels/man/lda.html) function from the topicmodels package

### Word-topic probabilities

The [tidy()](https://generics.r-lib.org/reference/tidy.html) method, originally from the broom package (Robinson [2017](https://www.tidytextmining.com/references.html#ref-R-broom)), for tidying model objects. The tidytext package provides this method for extracting the per-topic-per-word probabilities, called ββ (“beta”), from the model.

[library](https://rdrr.io/r/base/library.html)([tidytext](https://github.com/juliasilge/tidytext))

this has turned the model into a one-topic-per-term-per-row format.

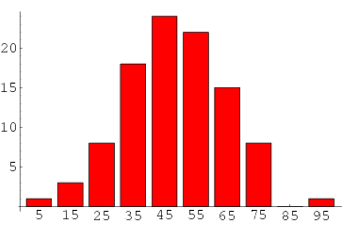
### Document-topic probabilities

Besides estimating each topic as a mixture of words, LDA also models each document as a mixture of topics. We can examine the per-document-per-topic probabilities, called γγ (“gamma”), with the matrix = "gamma" argument to [tidy()](https://generics.r-lib.org/reference/tidy.html)

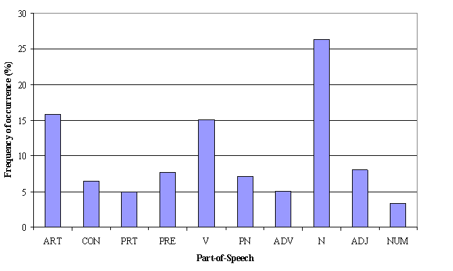
**Tips to improve results of topic modeling:**

The results of topic models are completely dependent on the features (terms) present in the corpus. The corpus is represented as document term matrix, which in general is very sparse in nature. Reducing the dimensionality of the matrix can improve the results of topic modelling. Based on my practical experience, there are few approaches which do the trick.

1. Frequency Filter – Arrange every term according to its frequency. Terms with higher frequencies are more likely to appear in the results as compared ones with low frequency. The low frequency terms are essentially weak features of the corpus, hence it is a good practice to get rid of all those weak features. An exploratory analysis of terms and their frequency can help to decide what frequency value should be considered as the threshold.



1. Part of Speech Tag Filter – POS tag filter is more about the context of the features than frequencies of features. Topic Modelling tries to map out the recurring patterns of terms into topics. However, every term might not be equally important contextually. For example, POS tag IN contain terms such as – “within”, “upon”, “except”. “CD” contains – “one”,”two”, “hundred” etc. “MD” contains “may”, “must” etc. These terms are the supporting words of a language and can be removed by studying their post tags.



1. Batch Wise LDA –In order to retrieve most important topic terms, a corpus can be divided into batches of fixed sizes. Running LDA multiple times on these batches will provide different results, however, the best topic terms will be the intersection of all batches.

**Assumptions of LDA for Topic Modelling:**

* Documents with similar topics use similar groups of words
* Latent topics can then be found by searching for groups of words that frequently occur together in documents across the corpus
* Documents are probability distributions over latent topics which signifies certain document will contain more words of a specific topic.
* Topics themselves are probability distribution over words

# 

**Libraries**

Python is one of the most popular programming languages for machine learning and data analysis. Its focus on code readability makes it super easy-to-use, and it has a large community of contributors who have developed a wide range of options to implement NLP models.

One of the top choices for topic modelling in Python is [Gensim](https://radimrehurek.com/gensim/), a robust library that provides a suite of tools for implementing LSA, LDA, and other topic modelling algorithms.

[NLTK](https://www.nltk.org/) is a framework that is widely used for topic modelling and text classification. It provides plenty of corpora and lexical resources to use for training models, plus different tools for processing text, including tokenization, stemming, tagging, parsing, and semantic reasoning. Although NLTK can be quite slow and difficult to use, it’s the most well-known and complete NLP library out there.

[SpaCy](https://spacy.io/) is the fastest framework for training NLP models. Although it is less flexible and supports fewer languages than NLTK, it’s much easier to use. SpaCy also provides built-in word vector and uses deep learning for training some models.

[Scikit-learn](https://scikit-learn.org/) provides a wide variety of algorithms for building machine learning models. It has excellent documentation, and intuitive methods that make it easy to train a model for topic modelling. If you are a beginner, you’ll find lots of useful tutorials that will help you get started with topic modelling using scikit-learn.

# import numpy as np

# import pandas as pd

# import gensim

# import gensim.corpora as corpora

# from gensim.utils import simple\_preprocess

# from gensim.models import CoherenceModel

# import spacy

# from spacy.lemmatizer import Lemmatizer

# from spacy.lang.en.stop\_words import STOP\_WORDS

# import en\_core\_web\_lg

# from tqdm import tqdm\_notebook as tqdm

# from pprint import pprint

Web extraction

**Why**, web scraping?

Web scraping, or web content extraction, can serve an unlimited number of purposes.Whether you're a new business or a growing one, web scraping helps you 10x your business growth with web data.

Here are some reasons why you should use Web Scraping for your next project:

* Technology makes it easy to extract data
* Access to technology is probably the most important factor of all, because it enables pretty much anyone to do web scraping at scale very easily.
* There's a lot of content on the web to help you master web scraping and probably even more service providers such as Captain Data to help you collect data.
* As websites are getting more complicated to scrape (like scraping a single page application), new tools such as Puppeteer make it possible to scrape virtually anything.
* Furthermore, deploying bots at scale has becoming increasingly accessible. It enables companies to extract data at any scale.
* Innovation at the speed of light

**Introduction** to web scraping?

Web scraping, web harvesting, or web data extraction is [data scraping](https://en.wikipedia.org/wiki/Data_scraping) used for [extracting data](https://en.wikipedia.org/wiki/Data_extraction) from [websites](https://en.wikipedia.org/wiki/Website). The web scraping software may directly access the [World Wide Web](https://en.wikipedia.org/wiki/World_Wide_Web) using the [Hypertext Transfer Protocol](https://en.wikipedia.org/wiki/Hypertext_Transfer_Protocol) or a web browser. While web scraping can be done manually by a software user, the term typically refers to automated processes implemented using a [bot](https://en.wikipedia.org/wiki/Internet_bot) or [web crawler](https://en.wikipedia.org/wiki/Web_crawler). It is a form of copying in which specific data is gathered and copied from the web, typically into a central local [database](https://en.wikipedia.org/wiki/Database) or spreadsheet, for later [retrieval](https://en.wikipedia.org/wiki/Data_retrieval) or [analysis](https://en.wikipedia.org/wiki/Data_analysis).

Web scraping a web page involves fetching it and extracting from it. Fetching is the downloading of a page (which a browser does when a user views a page). Therefore, web crawling is a main component of web scraping, to fetch pages for later processing. Once fetched, then extraction can take place. The content of a page may be [parsed](https://en.wikipedia.org/wiki/Parsing), searched, reformatted, its data copied into a spreadsheet or loaded into a database. Web scrapers typically take something out of a page, to make use of it for another purpose somewhere else. An example would be to find and copy names and telephone numbers, or companies and their URLs, or e-mail addresses to a list

# It not only includes social networking sites, such as Facebook, Twitter, Instagram, LinkedIn…etc., but also includes blogs, wikis, and news sites.

Web scraping is the process of collecting structured web data in an automated fashion. It’s also called web data extraction. Some of the [main use cases of web scraping](https://www.zyte.com/learn/what-is-web-scraping-used-for-learn/what-is-web-scraping-used-for/) include price monitoring, [price intelligence](https://www.zyte.com/learn/price-intelligence/), news monitoring, [lead generation](https://www.zyte.com/learn/lead-generation/), and [market research](https://www.zyte.com/learn/market-research/) among many others.

In general, web data extraction is used by people and businesses who want to make use of the vast amount of publicly available web data to make smarter decisions.

If you’ve ever copy and pasted information from a website, you’ve performed the same function as any web scraper, only on a microscopic, manual scale. Unlike the mundane, mind-numbing process of manually extracting data, web scraping uses intelligent automation to retrieve hundreds, millions, or even billions of data points from the internet’s seemingly endless frontier.

## **The basics** of web scraping

It’s extremely simple, in truth, and works by way of two parts: a web crawler and a web scraper. The web crawler is the horse, and the scraper is the chariot. The crawler leads the scraper, as if by hand, through the internet, where it extracts the data requested. [Learn the difference between web crawling & web scraping](https://www.zyte.com/learn/difference-between-web-scraping-and-web-crawling/)and how they work.

### The crawler

A web crawler, which we generally call a “spider,” is an artificial intelligence that browses the internet to index and searches for content by following links and exploring, like a person with too much time on their hands. In many projects, you first “crawl” the web or one specific website to discover URLs which then you pass on to your scraper.

### The scraper, A web scraper is a specialized tool designed to accurately and quickly extract data from a web page. Web scrapers vary widely in design and complexity, depending on the project. An important part of every scraper is the data locators (or selectors) that are used to find the data that you want to extract from the HTML file - usually, XPath, CSS selectors, regex, or a combination of them is applied.

One of the things we really like is how scraping and crawling are enabling businesses to create new products and innovate faster.

Take for example a price comparison website like Kayak, a technical SEO product like Botify or even a job board that is built from multiple sources. Without being able to extract web data, these companies would not be able to exist.The use cases are unlimited. And it really puts the bar higher in terms of innovation; by enabling easy access to web data to everyone, web scraping forces you to enhance your value proposition.It helps you innovate faster because you can test and execute new ideas faster. Let’s say you want to build a product referencing independent artists and their music … but you need a database! Well, you better start scraping.Better access to company data

Over the past decade, governments in many countries like France decided to open their data to the world. But ... (there's always a but!) it's not quite useful, or at least it needs to be enriched with other sources.

In France we have the Sirene database. They have an API (a bit sluggish) but it's a great start. Let's say you have a SIRET (a unique company identifier), here's what you could do:

* Enrich the SIRET with the Sirene API
* Find the company website's domain thanks to its name by searching and cross-referencing multiple search engines
* Look up the company on various websites depending on the company's typology: LinkedIn, AngelList, Yellow Pages and so on
* Aggregate the results, by attributing scores (this could be a bit tricky)

And voilà, you have a fully enriched company profile with everything you need: number of employees, date of creation, business category, etc. This is typically what any sales team dream to automate for their CRM.

Lead generation to build a sales machine

Well, I think you see it coming: if you have better access to company data, it also means you can build an automated sales machine.

Brand monitoring for everyone

The brand monitoring market is growing very fast. And for once, I think that we can all agree that checking other customer’s reviews has becoming a basic step when buying online.Consumers are more and more educated, they like to be recommended products and to be reassured that they're making the "right choice".Strangely enough, businesses do not always check reviews and ratings.Why? Well, it's not that easy. There are so many platforms gathering reviews and ratings that you need to extract reviews from each website and then aggregate them.

You could also monitor social networks and combine it with sentiment analysis to quickly respond to haters or reward users who love you.The outcome of improving your brand image in terms of ROI is just clear!

Here's how we collected reviews for a brand monitoring SaaS.

Market analysis at scale: Everyone talks about Big Data and Business Intelligence. But in the end, what really matters is quality over quantity.You don't need big data, but rather smart data.Let's say you sell machines and spare parts. There's obviously a "used" market. But how do you know a specific spare part is sold for? I mean, if you could just optimize the price by 10% ... imagine the additional revenues at scale!Web scraping to the rescue, you "just" need to collect data on specific websites that distributors use. And voilà, you can build an argus that is fed from the data you extract.Although in this case, data processing might be a bit tricky since product references are not always the same!

Here's how we automated market analysis at scale for a corporate client.

Data(base) enrichment on demand: I’ve already covered this topic a bit in the previous examples. But you have to understand that the possibilities are endless:

* You can’t post a small ad in a platform like Craigslist? There’s a bot for that
* You need to build a database for your new product
* You can add search or product metrics from other platforms
* Data provided by your users aren’t enough? Well, you get it

Again, at the risk of repeating myself, web data is not only a mean to boost your business from a sales or marketing point of view. It also enables you to enhance your product and foster innovation.

You've been tasked with building a model that will classify houses. Your product owner wants you to use deep learning, because they think it's a great option for such a use case.You need a large volume to build your training set. And you're definitely not going to do this by hand.Want to predict the stock market? Web. Scraping.You need to predict your competitor's pricing? Scrape that data!Web scraping is actually the data scientist’s best friend. But you're a data scientist, not a freaking bot! You want to analyse and build predictive models, not clean and extract web data.So don't reinvent the wheel, use a platform or ask us to do it for you.

If you’re serious about SEO, you probably use tools such as SEMrush or keywords finder like Uber suggest. It’s simple: these simply won’t exist without data extraction.Using such tools, you can quickly find out your SEO competitors for a particular search term.You can determine the title tags and the keywords they are targeting to get an idea of what is driving traffic to their website.If you have a website with lots of content (1K+ URLs), you could also perform a technical SEO analysis to check out broken links and verify how is your content performing across your entire website.

Finally, you have to know that one of the best uses of web scraping is testing. If you’re a developer, I’m sure you heard of Selenium.If you want to build user testing scenarios or monitor a website’s performance, you need a bot.Companies like Ip Label have built products that automate this kind of testing.

## **The web scraping process**

1. Identify the target website
2. Collect URLs of the pages where you want to extract data from
3. Make a request to these URLs to get the HTML of the page
4. Use locators to find the data in the HTML
5. Save the data in a JSON or CSV file or some other structured format

## **For what web scraping used for?**

### Price intelligence: In our experience, price intelligence is the biggest use case for web scraping. Extracting product and pricing information from e-commerce websites, then turning it into intelligence is an important part of modern e-commerce companies that want to make better pricing/marketing decisions based on data.

How web pricing data and price intelligence can be useful:

* Dynamic pricing
* Revenue optimization
* Competitor monitoring
* Product trend monitoring
* Brand and MAP compliance

### **Applications** of Web Scraping:

* Some of the practical applications of web scraping could be: Gathering resume of candidates with a specific skill
* Extracting tweets from twitter with specific hashtags, Lead generation in marketing, Scraping product details and reviews from e-commerce websites.
* Apart from the above use-cases, web scraping is widely used in natural language processing for extracting text from the websites for training a deep learning model.

### Market research: Market research is critical – and should be driven by the most accurate information available. High quality, high volume, and highly insightful web scraped data of every shape and size is fueling market analysis and business intelligence across the globe.

* Market trend analysis
* Market pricing
* Optimizing point of entry
* Research & development
* Competitor monitoring

### Alternative data for finance: Unearth alpha and radically create value with web data tailored specifically for investors. The decision-making process has never been as informed, nor data as insightful – and the world’s leading firms are increasingly consuming web scraped data, given its incredible strategic value.

* Extracting Insights from SEC Filings
* Estimating Company Fundamentals
* Public Sentiment Integrations
* News Monitoring

### Real estate: The[digital transformation of real estate](https://www.zyte.com/blog/web-scraping-real-estate-data-use-cases/) in the past twenty years threatens to disrupt traditional firms and create powerful new players in the industry. By incorporating web scraped product data into everyday business, agents and brokerages can protect against top-down online competition and make informed decisions within the market.

* Appraising Property Value
* Monitoring Vacancy Rates
* Estimating Rental Yields
* Understanding Market Direction

### News & content monitoring: Modern media can create outstanding value or an existential threat to your business - in a single news cycle. If you’re a company that depends on timely news analyses, or a company that frequently appears in the news, [web scraping news data](https://www.zyte.com/data-types/news-scraping-api/) is the ultimate solution for monitoring, aggregating, and parsing the most critical stories from your industry.

* Investment Decision Making
* Online Public Sentiment Analysis
* Competitor Monitoring
* Political Campaigns
* Sentiment Analysis

### Lead generation: Lead generation is a crucial marketing/sales activity for all businesses. In the 2020[Hubspot report,](https://www.hubspot.com/marketing-statistics?__hstc=234333761.f597576fe5cf26f374c4dcd373e79bb7.1578558839179.1607600764366.1607603907810.490&__hssc=234333761.1.1607603907810&__hsfp=480828235) 61% of inbound marketers said generating traffic and leads was their number 1 challenge. Fortunately, web data extraction can be used to get access to structured lead lists from the web.

### Brand monitoring: In today’s highly competitive market, it's a top priority to protect your online reputation. Whether you sell your products online and have a strict pricing policy that you need to enforce or just want to know how people perceive your products online,[brand monitoring with web scraping](https://www.zyte.com/brand-monitoring/) can give you this kind of information.

### Business automation: In some situations, it can be cumbersome to get access to your data. Maybe you have some data on your own website or on your partner’s website that you need in a structured way. But there’s no easy internal way to do it and it makes sense to create a scraper and simply grab that data. As opposed to trying to work your way through complicated internal systems.

### MAP monitoring: Minimum advertised price (MAP) monitoring is the standard practice to make sure a brand’s online prices are aligned with their pricing policy. With tons of resellers and distributors, it’s impossible to monitor the prices manually. That’s why web scraping comes in handy because you can keep an eye on your products’ prices without lifting a finger.

**Top 5** Social Media Scraping Tools

# 1. [Octoparse](https://www.octoparse.com/)**- Octoparse**was developed for non-coders to accommodate complicated web scraping jobs.

**2. Dexi.io-** Dexi.io supports creating three kinds of robots: **extractor, crawler, and Pipes**.Dexi.io does require some programming skills to master, but you can integrate third-party services for captcha solving, cloud storage, text analysis (MonkeyLearn service integration), and even with AWS, Google Drive, Google Sheets…

### **3. OutWit Hub-**Outwit Hub offers a **simplistic graphic user interface**, as well as sophisticated scraping functions and data structure recognition. With no prior programming background required, OutWit Hub can extract and export links, email addresses, RSS news and data tables to Excel, CSV, HTML or SQL databases.

### **4. Scrapinghub-**Scrapinghub is a cloud-based web crawling platform. The app consists of 4 great tools:  **Scrapy Cloud** for deploying and running web crawlers based on Python; **Portia**is an open-source software to extract data without coding; **Splash** is also an open-source JavaScript rendering tool to extract data from web pages that use JavaScript; **Crawlera**is a tool to avoid being blocked by websites, by crawler from multiple locations and IPs.

**5. Parsehub-** Parsehub is another **coding-free desktop scraper** . It offers a graphical interface to select and extract the data from JavaScript and AJAX pages. Data can be scraped from nested comments, maps, images, calendars, and even pop-ups. Moreover, Parsehub also has a browser-based extension to launch your scraping task instantly. Data can be exported as  Excel, JSON, or via API.

## Library for Web Scraping

## Requests (HTTP for Humans)

## lxml

* Beautiful Soup: Beautiful Soup is a Python package for parsing HTML and XML documents. It creates parse trees that is helpful to extract the data easily. Beautiful Soup is a pure Python library for extracting structured data from a website. It allows you to parse data from HTML and XML files.It acts as a helper module and interacts with HTML in a similar and better way as to how you would interact with a web page using other available developer tools.

## Selenium: Selenium is a web testing library. It is used to automate browser activities.

## Scrapy

* MechanicalSoup.
* Urllib.
* Import.io
* Phantom.js
* Octoparse
* Diffbot
* 80legs

Other Library

* import pandas as pd: Pandas is a library used for data manipulation and analysis. It is used to extract the data and store it in the desired format.
* import numpy as np
* import matplotlib.pyplot as plt
* import seaborn as sns%matplotlib inlineimport re
* import timefrom datetime\
* import datetime
* import matplotlib.dates as mdates
* import matplotlib.ticker as tickerfrom urllib.request
* import urlopenfrom bs4
* import requests

**Websites** for data scraping

* Amazon
* Twitter
* Google
* Facebook
* Youtube

Exploratory Data Analysis – EDA

Exploratory data analysis is an approach of analyzing data sets to summarize their main characteristics, often using statistical graphics and other data visualization methods.Exploratory data analysis (EDA) is used by data scientists to analyze and investigate data sets and summarize their main characteristics, often employing data visualization methods.It helps determine how best to manipulate data sources to get the answers you need, making it easier for data scientists to discover patterns, spot anomalies, test a hypothesis, or check assumptions. The main purpose of EDA is to help look at data before making any assumptions. It can help identify obvious errors, as well as better understand patterns within the data, detect outliers or anomalous events, find interesting relations among the variables.

Four Types of EDA

* Univariate non-graphical
* Multivariate non-graphical
* Univariate graphical
* Multivariate graphical

Common graphs

* Bar plot
* Histogram
* Box chart
* Scatter plot

Steps involved in EDA

* Data Cleaning
* Imputation Technique
* Data Analysis and Visualization
* Transformations
* EDA for Topic Modelling

We have conducted the following EDA for our project:

1. Data Cleaning

* Converting to lowercase
* Removing punctuations
* Removing stopwords
* Removing mentions,urls,emojis, numbers
* Removing spaces and newlines

1. Created Tfidf matrix and plotted

* Word cloud
* Word frequency bar plot

We will perform the following steps:

* Identification of variables and data types.
* Tokenization: Split the text into sentences and the sentences into words. Lowercase the words and remove punctuation.
* Words that have fewer than 3 characters are removed.
* Pre-processing of data (stop words, Outlier treatment).
* Words are lemmatized — words in third person are changed to first person and verbs in past and future tenses are changed into present.
* Words are stemmed — words are reduced to their root form.
* Analyzing the basic metric.
* Bivariate Analysis.
* Variable transformations.
* Missing value treatment.

Basic statistics on text:

Descriptive statistics:First of all, we extract the basics statistics: count of rows, unique rows, frequencies.This attributes in the text data will tell us if we need to remove repeated rows or how many rows contains null values in any column.

Text length:The length of the samples in the dataset is very important, as it can affect how you represent your text as features for the ML models. For example, TF-IDF is usually too sparse for short texts and average Word2Vec is usually too noisy for long texts. The length can also affect the algorithm you use.

We calculate the number of words in each tweet and look at the length distribution.

Exploring relevant features in the data: In this section we will create some additional features that provide relevant information about the composition of our texts.

Statistical Count Features from headlines and text that we are going to explore:

* Sentence Count — Total number of sentences in the text
* Word Count — Total number of words in the text
* Character Count — Total number of characters in the text excluding spaces
* Sentence density — Number of sentences relative to the number of words
* Word Density — Average length of the words used in the headline
* Punctuation Count — Total number of punctuations used in the headline
* Stopwords Count — Total number of common stopwords in the text

Then, we calculate these features on our dataset .

## After analyzing the feature distributions on the text variable Now we can calculated the new features, we can analyze the descriptive statistics to identify the main insights on the data distribution and outliers.

# Categorizing and POS tagging words:Another group of features we can inspect in text data are the Part-Of-Speech tagging. The process of classifying words into their parts of speech and labeling them accordingly is known as part-of-speech tagging, POS-tagging, or simply tagging. Parts of speech are also known as word classes or lexical categories.Our target in the next section is to identify the POS tags and analyze its distribution on the dataset. Every word in a text is tagged as a noun, determiner, adjective, adverb Maybe we can observe any interesting behavior but it is not frequent, usually, the distribution of the tags is coherent and depends on the domain or context.

# Check for Unknown words:It may be very common that unknown words are included in our texts and summaries, consequently, we should analyze them and you probably have to define how to deal with them. Most of the unknown words are names, surnames, locations or even misspelled words, which we have to decide to correct or not them.

In order to search for these words, we need a vocabulary to compare to. In this case, we use the Glove embeddings, checking if our words are included in those embeddings.We can show the distribution of the unknown words in our texts to get a fast insight of their relevance.

Use of stopwords and punctuations: Now that we have a more accurate vision of the composition of our texts, we need to analyze the use of stopwords and punctuation, this analysis will indicate us if these “special type” of characters will be removed or transform when we train our models.As we did previously, the NLTK library provide us with a list of stopwords for English texts, so we can look for them in our dataset. Now, lets explore the histograms of the count of stop words and punctuations, to get a better intuition about the texts we are going to work with.

# Most frequent terms and Wordclouds:The domain or context of our texts will determine the most frequent words, therefore, it is important to verify what those words are and thus identify the domains and confirm that they are the expected ones.“A Wordcloud (or Tag cloud) is a visual representation of text data. It displays a list of words and the importance of each being shown with font size or color (the bigger the more frequent). This format is useful for quickly perceiving the most relevant terms on a document or set of documents.”We will draw the word cloud for the source texts and the summaries to compare if they are very similar, it will allow us to check that the relevant concepts have been correctly extracted in the summaries.

Model Building

LDA Implementation

1. Loading data
2. Data cleaning
3. Exploratory analysis
4. Preparing data for LDA analysis
5. LDA model training
6. Analyzing LDA model results