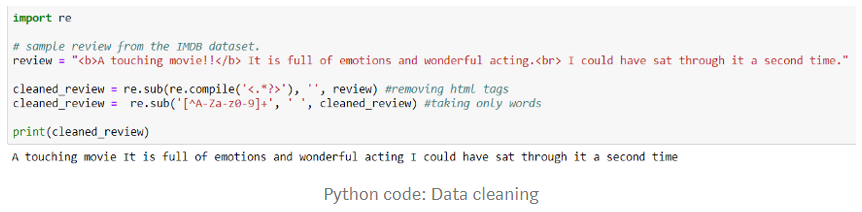
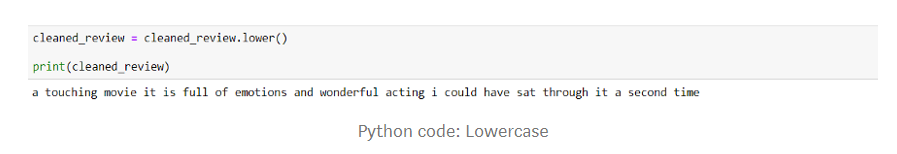
**NLP project**

1. Pre-processing text using NLP
2. Extracting features from text
3. Supervised learning on text
4. Unsupervised learning on text

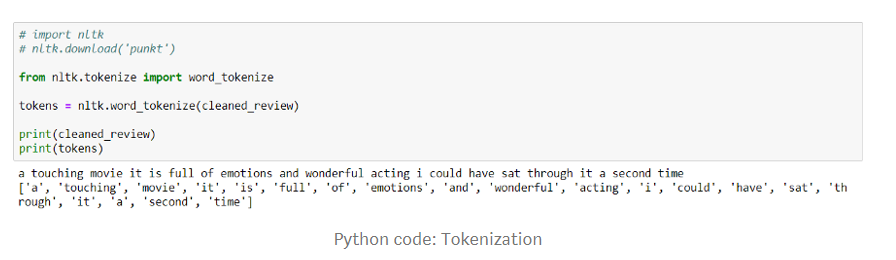
**Pre-processing text:** Textual datamining technique that involves transforming the raw data into an understandable format.

1. Sentence segmentation (data cleaning is a *necessary step*)

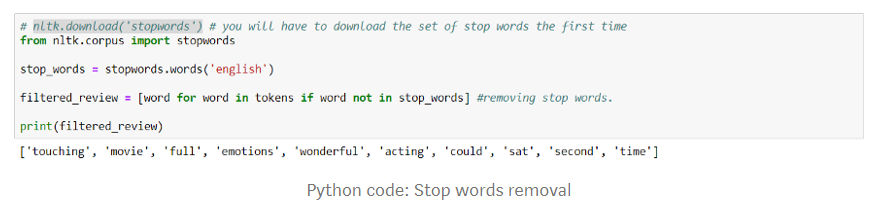




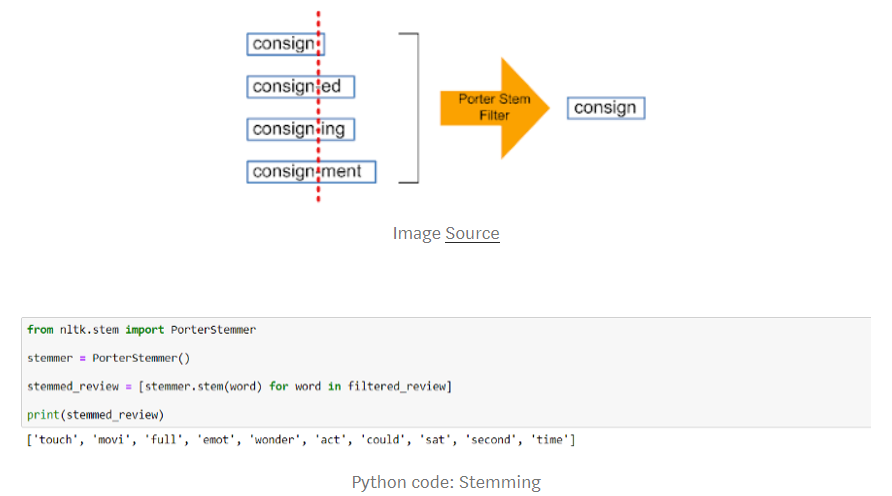
1. Tokenization (process of breaking up text document into individual words/tokens) & Normalization (removing digits & punctuations, expanding abbreviations, stemming & lemmatization, *necessary step*)



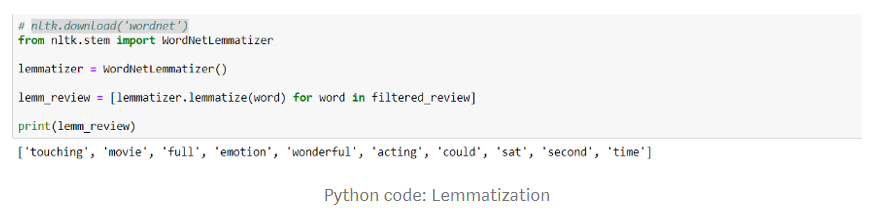
*! pip install nltk*



Stemming is the process of reducing a word to its stem/root word. It removes the morphological affixes from words, leaving only the word stem.



The stem word may or may not be a valid word in the language. The root word in lemmatization belongs to a valid word in the language.



1. Part-of-speech (POS) tagging (rule-based, statistical, and deep learning based)
2. Named entity recognition

**Feature extraction from text data (vectorization):** Converting text into numbers(encoding data in numbers), also called **word embedding**

We can use supervised ML when our historical training data contains labels. On the other hand, unsupervised ML is applied when there no labels in the data. Unsupervised ML methods aim to summarize or compress the data. An example the problem of detecting spam email versus anomaly detection. In the former case, we would have the training data with the spam/not-spam labels, in the latter case, we would have to detect anomalous emails based on the training set of emails.

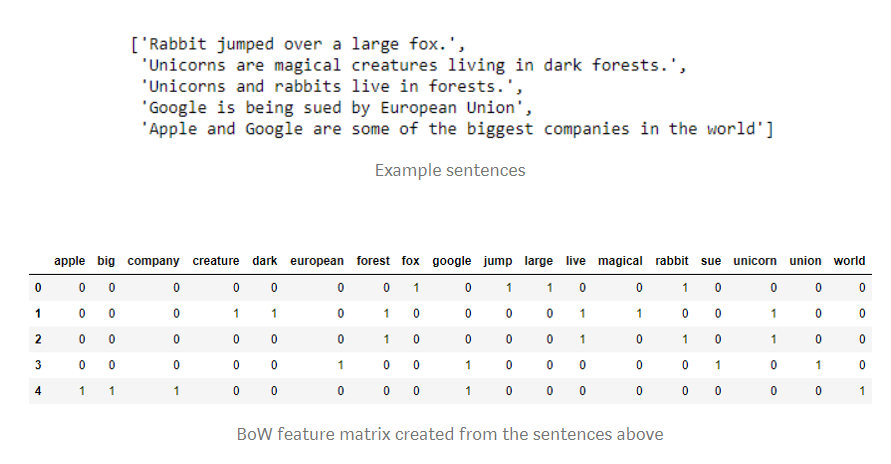
N-grams can be used when we want to preserve sequence information in the document, like what word is likely to follow the given one. Unigrams (N=1) don’t contain any sequence information because a word is taken individually.

*In NLP tasks, each text sentence is called a document and collection of such documents is referred to as text corpus.*

The easiest approach of converting texts into numeric vectors is to use the Bag-of-Words (BoW) method. The principle of BoW is to take all the unique words/tokens from the text and create a text corpus called vocabulary. Using the vocabulary, each sentence/document can be represented as a vector consisting of ones and zeros, depending on whether a word from the vocabulary is present in the sentence or not.

The intuition behind BoW is that two sentences are said to be similar if they contain similar set of words.

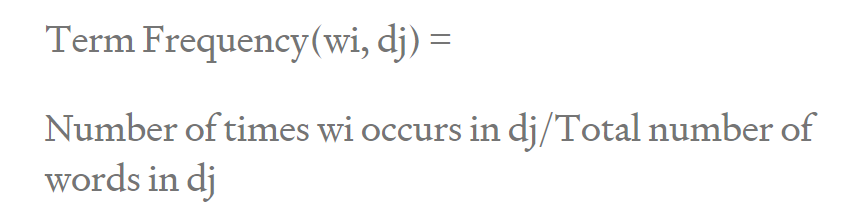
Example:

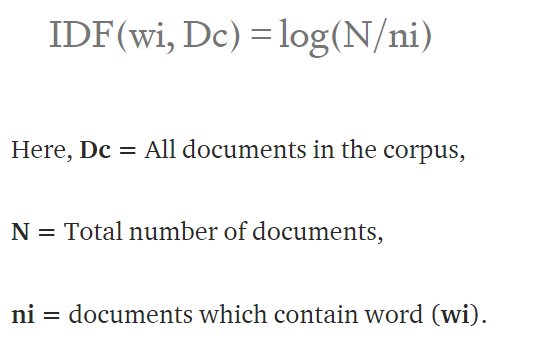


In order to add more context to the vocabulary, tokens may be grouped together. This method is called N-gram approach. An N-gram is a sequence of N tokens. For example, a 2-gram (bigram) is a sequence of two words, while a trigram is a sequence of three.

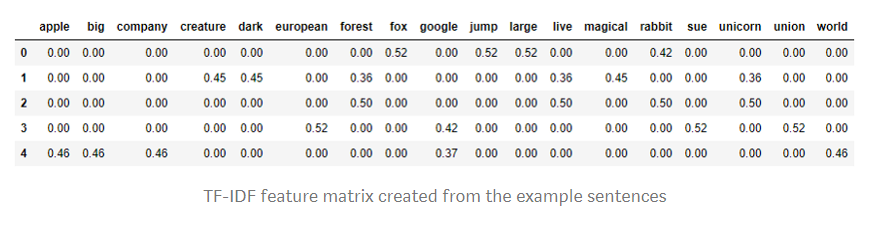
Once the vocabulary is chosen, be it 1-, 2-, or 3-gram, occurrences of the grams must be counted. We could use the BoW approach but the downside of this is that popular (frequent?) words become too important. Thus, the most popular method for this is called term frequency-inverse document frequency (TF-IDF).

TF-IDF consists of TF that captures the importance of the word wrt the length of the sentence and inverse document frequency (IDF) which captures in how many sentences the gram occurs with respect to the total number of sentences, thus highlighting the rarity of the word. Intuitively, a word has a higher TF-IDF score if it occurs frequently in a document but infrequently in the set of all the documents (determines how unique is the word in the corpus).





TF\*IDF gives more weightage to words which occurs less in the corpus (more in the document).



Score(D, T) = TF(D, T) \* log(N/docFrequency(T))

**Supervised Learning:**

When dealing with texts, we mostly encounter classification problems (categorical target/label).

Once the text data has been converted into numeric form, ML algorithms can be applied to it. This process is called training the model — the model learns patterns from the features to predict the labels. The model can be optimized for better performance using model parameters through a process called hyperparameter tuning. The resulting model is then evaluated on unseen/test data. The performance of the model is measured using various metrics, such as accuracy, precision, recall, F1 score, to name a few. These scores are built for comparing the true labels of the data to predicted labels.

Typical algorithms that are used for text classification:

* Multinomial Naive Bayes — built on Bayes’ theorem, using the assumption that each feature is independent of the other. Multinomial Naive Bayes is an extension used for classification tasks with more than two different labels (multi-class/ multi-label classification).
* Logistic Regression — an algorithm that uses a Sigmoid function to predict categorical values.
* [Support Vector Machines](https://medium.com/machine-learning-101/chapter-2-svm-support-vector-machine-theory-f0812effc72?source=post_page---------------------------) (SVM) — an algorithm that uses a line or a hyper-plane (in case there are more than two features, thus creating a multidimensional space) to separate classes.
* Random Forest — an ensemble method that trains decision trees on various subsets of data in parallel (*unsure of performance on text data*)
* Gradient Boosting Machine (GBM) — an ensemble method that train a sequence of weak learners, such as decision trees, to achieve accurate results. XGBoost is one of the most popular implementations of this family of algorithms (*unsure of performance on textual data*).

**Unsupervised Learning:**

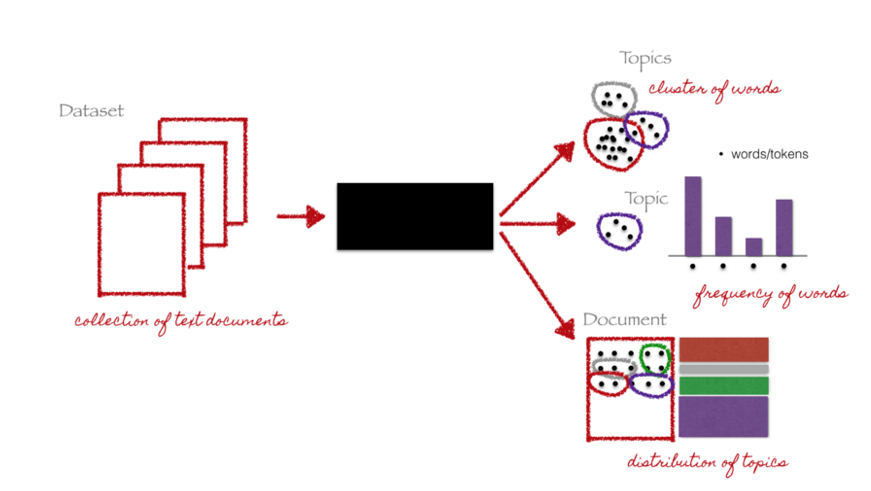
Used when data do not have labels.

Several available clustering algorithms:

* Connectivity-based clustering — Also known as hierarchical clustering, connects data points based on the distance between them. There are two types of strategies used to connect these points: agglomerative, a “bottom-up” approach, where each data point becomes its own cluster, with pairs of clusters merged iteratively, or “top-down” divisive approach whereby the whole data space is one cluster being split recursively. For agglomerative hierarchical clustering, two additional metrics are necessary: a distance metric that shows how similar two data points are, Euclidean, Hamming, or cosine distances are typical examples and a linkage criterion that shows how similar groups of data points are.
* Centroid-based clustering — data is divided into clusters based on the points’ closeness to the centroids of the clusters. K-means is the most popular implementation of the algorithm. The basic algorithm is as follows: (1.) select k as several clusters, (2.) assign data points into clusters, (3.) compute cluster centroids, (4.) reassign data points to the closest centroid, (5.) repeat the previous two steps until the centroids do not change.
* Density-based clustering — data space is divided and clustered into regions of density. DBSCAN and OPTICS are two popular algorithms that extract the dense regions of the data space, leaving behind the “noisy” data in the sparse regions. OPTICS tries to overcome DBSCAN’s weakness of performing badly in the borders and data sets of varying density.

**Text summarization:**

* 1. **Automated:** Automated text summarization is a process of using ML algorithms to create summaries of documents or a set of documents. These algorithms perform best with many documents and/or long documents.
  2. **Topic modelling:** Focuses on extracting themes from a collection of documents (when texts are diverse). Topic models are often called probabilistic statistical models because they use statistical techniques, such as singular value decomposition (SVD), to uncover latent semantic structures from texts. SVD relies on matrix factorization that is a technique from linear algebra which divides the feature matrix, into smaller components. Methods, such as Latent Semantic Indexing (LSI), Latent Dirichlet Allocation (LDA), and Non-Negative Matrix Factorization (NNMF) take advantage of techniques from linear algebra to divide a document into topics, which are essentially clusters of words.

****

The resulting vector contains all topics with a weight.

Information retrieval with keywords: Similar content can be grouped by their topics (group of keywords).