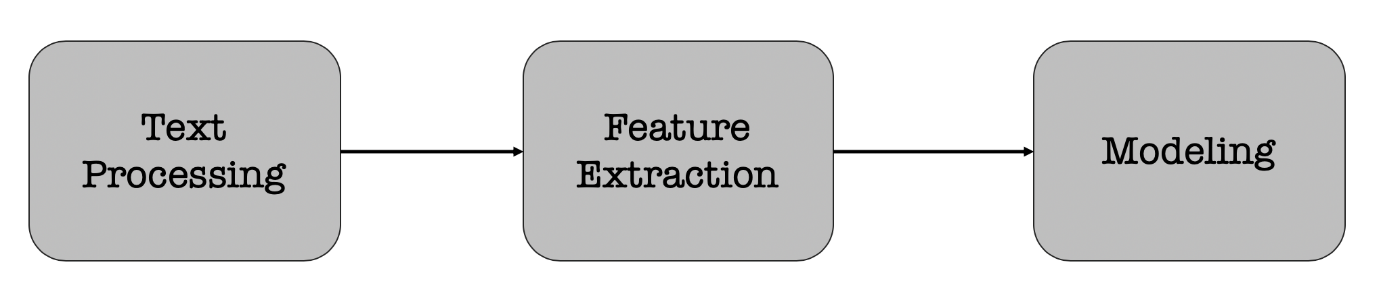
**The NLP Pipeline**

The NLP Pipeline involves the following stages.

1. Text Processing
   * Cleaning
   * Normalization
   * Tokenization
   * Stop Word Removal
   * Part of Speech Tagging
   * Named Entity Recognition
   * Stemming and Lemmatization
2. Feature Extraction
   * Bag of Words
   * TF-IDF
   * One-hot Encoding
   * Word Embeddings
3. Modeling

[](https://github.com/chaitanyakasaraneni/nlp_pipeline/blob/master/images/nlp_pipeline.png)

Each stage transforms text in some way and produces an intermediate result that the next stage needs. For example,

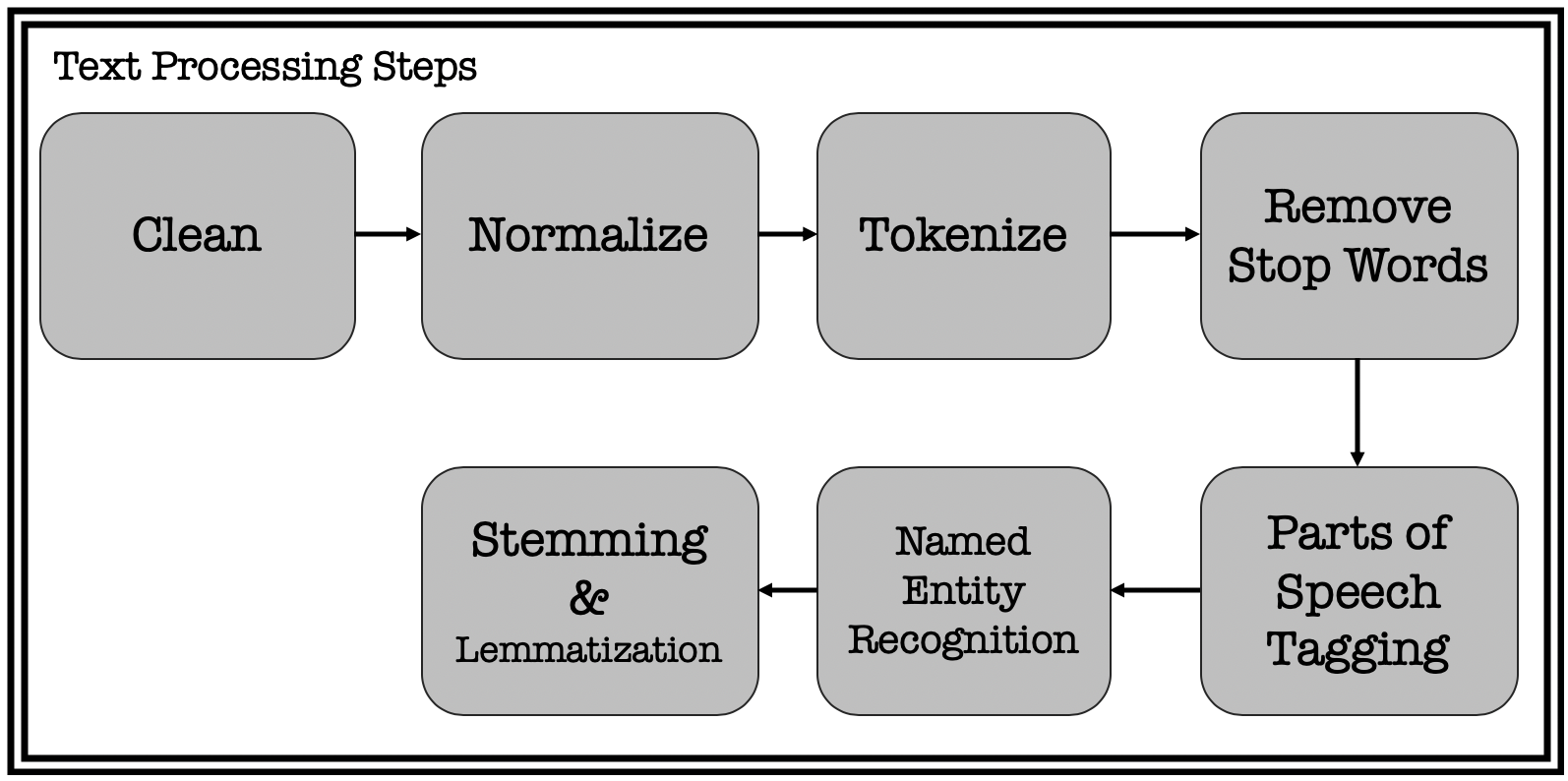
* **Text Processing** — take raw input text, clean it, normalize it, and convert it into a form that is suitable for feature extraction.
* **Feature Extraction:** Extract and produce feature representations that are appropriate for the type of NLP task you are trying to accomplish and the type of model you are planning to use.
* **Modeling:** Design a model, fit its parameters to training data, use an optimization procedure, and then use it to make predictions about unseen data.

Text Processing

Text processing is first stage of NLP pipeline that discusses how text data extracted from different sources is prepared for the next stage — **feature extraction**.

* **Cleaning** — The first step in text processingis to clean the data. i.e., removing irrelevant items, such as HTML tags. This can be done in many ways. Example includes using regular expressions, [beautiful soup library](https://www.crummy.com/software/BeautifulSoup/bs4/doc/), CSS selector, etc.
* **Normalization**— The cleaned data is then normalized by converting all words to lowercase and removing punctuation and extra spaces
* **Tokenization** — The normalized data is split into words, also known as tokens
* **Stop Words removal**— After splitting the data into words, the most common words (a, an, the, etc.), also known as stop words are removed
* **Parts of Speech Tagging**— The parts of speech are identified for the remaining words
* **Named Entity Recognition** — The next step is to recognize the named entities in the data
* **Stemming and Lemmatization** — Converting words into their canonical / dictionary forms, using **stemming and lemmatization.**

Image for post



Steps in Text Processing

\* **Stemming** is a process in which a word is reduced to its stem/root form. i.e., the word running, runs, etc.. can all be reduced to “run”.  
\* **Lemmatization** is another technique used to reduce words to a normalized form. In this case, the transformation actually uses a dictionary to map different variants of a word to its root. With this approach, the non-trivial inflections such as is, are, was, were, are mapped back to root ‘be’.

After performing these steps, the text will look very different from the original data, but it captures the essence of what was being conveyed in a form that is easier to work with.

# Feature Extraction

Text data is represented on modern computers using an encoding such as ASCII or Unicode that maps every character to a number. Computer stores and transmits these values as binary, zeros and ones, which have an implicit ordering. Individual characters don’t carry much meaning at all and can mislead the NLP algorithms.

## **Bag of words (BOW) model**

A bag of words model treats each document as an un-ordered list or bag of words. The word document refers to a unit of text that is being analyzed. For example, while performing a sentiment analysis on tweets, each tweet is considered as a document.

## **Term Frequency — Inverse Document Frequency (TF-IDF)**

One limitation of bag of words approach is that it treats every word as being equally important. Whereas, some words occur very frequently in a corpus. Consider a financial document for example. “Cost” or “price” is a very common term.

This limitation can be compensated for by counting number of documents in which each word occurs, known as document frequency, and then dividing the term frequency by document frequency of that term.

This gives us a metric that is proportional to frequency of a term in document, but inversely proportional to number of documents it appears in. This highlights the words that are more unique to a document, thus better for characterizing it.

This approach is called Term Frequency — Inverse Document Frequency (TF-IDF).

## **One-hot encoding**

Another way to represent words is to use one-hot encoding. It’s just like bag of words but only that each word is kept in each bag and a vector is built for it.

## **Word Embeddings**

One-hot encoding doesn’t work in every situation. It breaks down when there is a large vocabulary to deal with, because the size of word representation grows with number of words. It is required that word representation is limited to a fixed-size vector.

In other words, an embedding for each word is to be found in vector space that is exhibiting some desired properties. i.e. if two words are similar in meaning, they should be closer to each others compared to the words that are not. And if two pairs of words have similar difference in meanings, they should be approximately equally separated in the embedded space.

This representation can be used for various purposes like finding analogies, synonyms and antonyms, classifying words as positive, negative, neutral, etc.

# Modeling

The final stage of the NLP pipeline is **modeling**, which includes designing a statistical or machine learning model, fitting its parameters to training data, using an optimization procedure, and then using it to make predictions about unseen data.