**NLP(Natural Language Processing)**

### Natural language processing (NLP) is a branch of [artificial intelligence](https://www.sas.com/en_in/insights/analytics/what-is-artificial-intelligence.html) that helps computers understand, interpret and manipulate human language.

When we have all features in text format then we use NLP in that case to train our model.

## Why is NLP important?

### **Large volumes of textual data**

Natural language processing helps computers communicate with humans in their own language and scales other language-related tasks.

### **Structuring a highly unstructured data source**

NLP is important because it helps resolve ambiguity in language and adds useful numeric structure to the data for many downstream applications, such as speech recognition or text analytics.

**Note:**

* Natural language processing includes many different techniques for interpreting human language, ranging from statistical and machine learning methods to rules-based and algorithmic approaches.
* We need a broad array of approaches because the text- and voice-based data varies widely, as do the practical applications.
* Basic NLP tasks include tokenization and parsing, lemmatization/stemming, part-of-speech tagging, language detection and identification of semantic relationships.

If you ever diagramed sentences in grade school, you’ve done these tasks manually before.

**Example: Amazon Alexa, Spam detection, google assistant**

Jupyter notebook: **Natural Language Processing.ipynb**

1. **Tokenization:**

* Convert paragraphs in sentences and sentences into words.

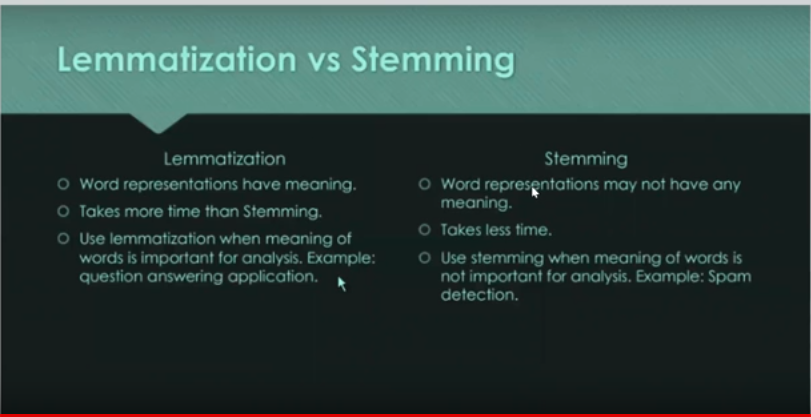
1. **Stemming: *Stemming*** usually refers to a crude heuristic process that chops off the ends of words in the hope of achieving this goal correctly most of the time, and often includes the removal of derivational affixes.

[**https://nlp.stanford.edu/IR-book/html/htmledition/stemming-and-lemmatization-1.html**](https://nlp.stanford.edu/IR-book/html/htmledition/stemming-and-lemmatization-1.html)

1. **Lemmatization:** Due to limitations of stemming we use Lemmatization.

*Lemmatization* usually refers to doing things properly with the use of vocabulary and morphological analysis of words, normally aiming to remove inflectional endings only and to return the base or dictionary form of a word, which is known as the *lemma*

Impo.: [**https://nlp.stanford.edu/IR-book/html/htmledition/stemming-and-lemmatization-1.html**](https://nlp.stanford.edu/IR-book/html/htmledition/stemming-and-lemmatization-1.html)

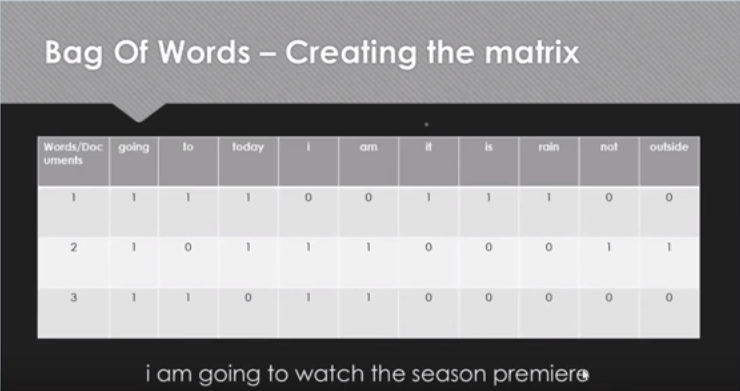


# **Natural Language Processing|BagofWords:** Also known as **Document metrics**

In order to convert those words, we have prepared above in the integer number format we use this **BagOfWords**

**Steps:**

1. Lower the sentences.
2. Tokenization.
3. Histogram: to count the frequency of the words and sort them.
4. Filter words: take more frequent words.
5. Creating document metrics like below: and this metrics ML model will understand now.



1. Now train your data with this input.

# 5) **Natural Language Processing|TF-IDF for Machine Learning| Text Processing.:** Due to the disadvantages of BOW, we use this.: Give more importance to some specific words.

Another problem with the bag of words approach is that it doesn’t account for noise. In other words, certain words are used to formulate sentences but do not add any semantic meaning to the text.

For example, the most commonly used word in the English language is the which represents 7% of all words written or spoken. You couldn’t make deduce anything about a text given the fact that it contains the word the.

On the other hand, words like good and awesome could be used to determine whether a rating was positive or not.

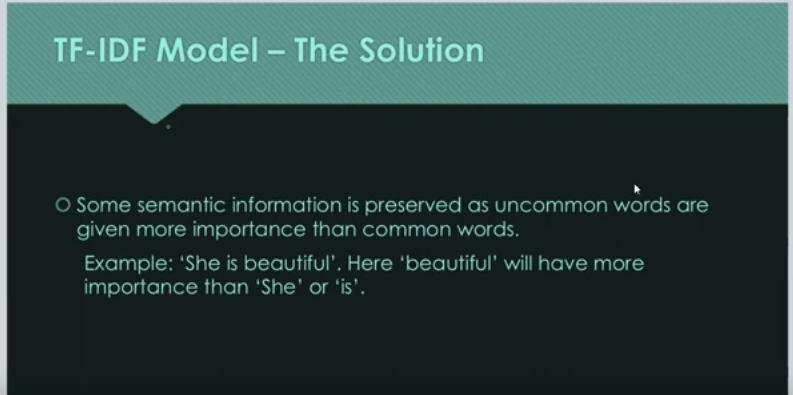
## Term Frequency (TF)

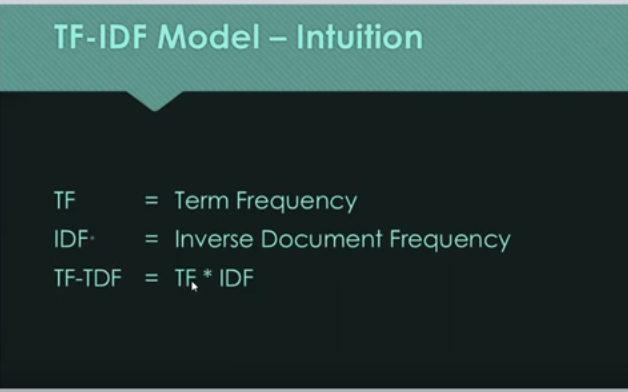
The number of times a word appears in a document divided by the total number of words in the document. Every document has its own term frequency.

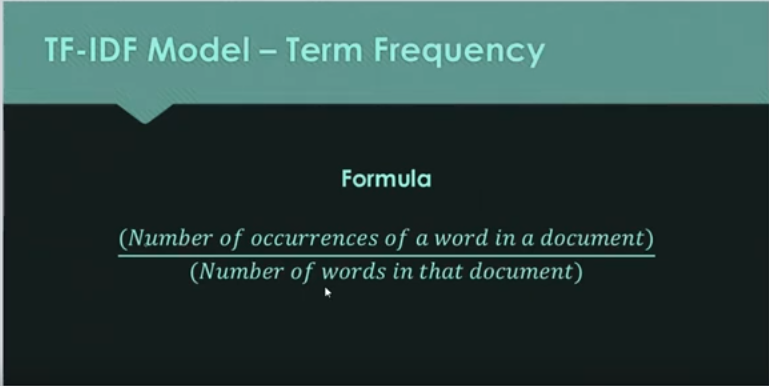
## Inverse Data Frequency (IDF)

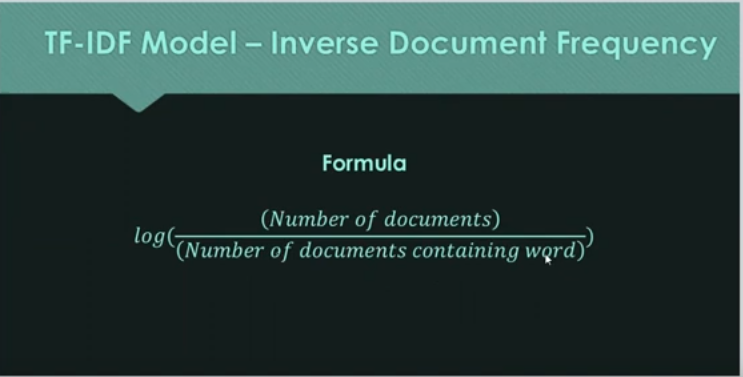
The log of the number of documents divided by the number of documents that contain the word w. Inverse data frequency determines the weight of rare words across all documents in the corpus.

Lastly, the TF-IDF is simply the TF multiplied by IDF.









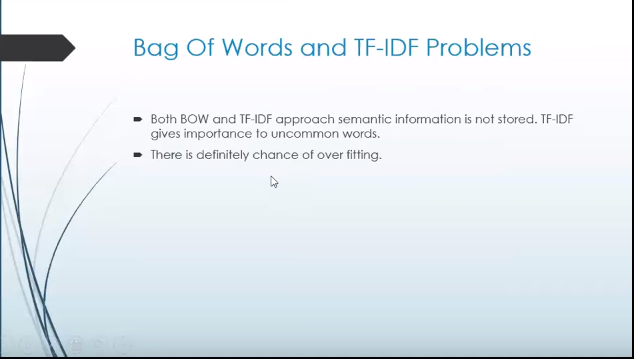
# **6) Implementing a Spam classifier in python| Natural Language Processing**

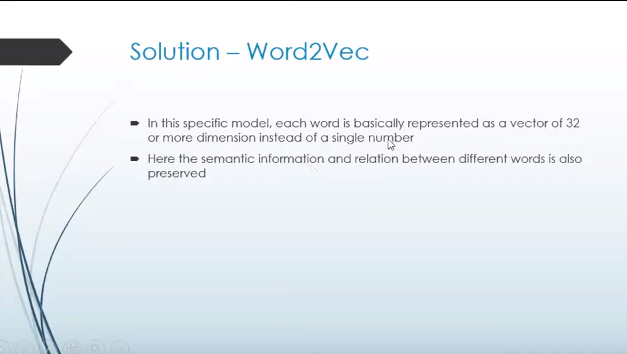
Jupyter notebook: **Natural Language Processing.ipynb**

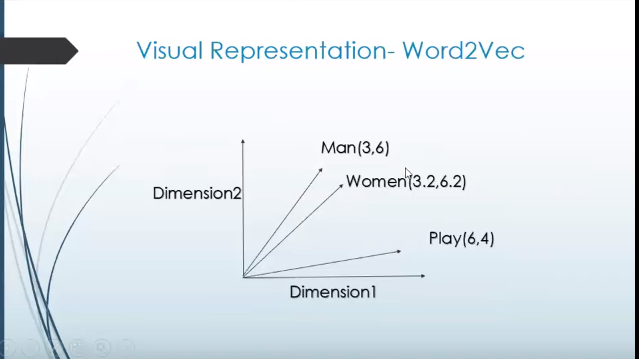
# **Word2Vec Easily Explained:** Due to the disadvantages of Tf-IDF, we use this(Overfitting) - Semantic information not stored.

Word2Vec is an efficient and effective way of representing words as vectors. The whole body of the text is encapsulated in some space of much lower dimension. In this space, all vectors have certain orientation and it is possible to explicitly define their relationship with each other.

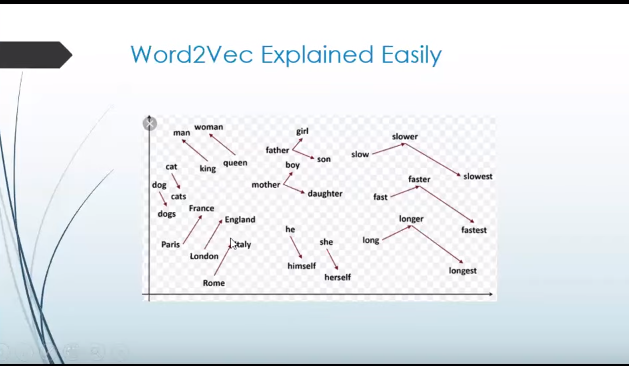
Impo.: <http://www.insightsbot.com/word2vec-explained-easily/>



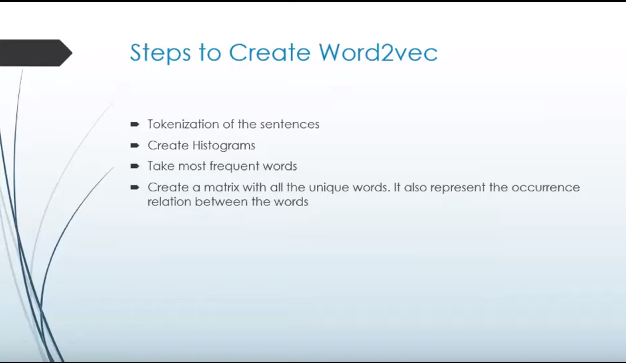




More interrelated words are closest to each other.



**Steps:**



The purpose and usefulness of Word2vec are to group the vectors of similar words together in vector-space. That is, it detects similarities mathematically. Word2vec creates vectors that are distributed numerical representations of word features, features such as the context of individual words. It does so without human intervention.

Given enough data, usage, and contexts, Word2vec can make highly accurate guesses about a word’s meaning based on past appearances. Those guesses can be used to establish a word’s association with other words (e.g. “man” is to “boy” what “woman” is to “girl”), or cluster documents and classify them by topic. Those clusters can form the basis of search, sentiment analysis and recommendations in such diverse fields as scientific research, legal discovery, e-commerce and customer relationship management.

The output of the Word2vec neural net is a vocabulary in which each item has a vector attached to it, which can be fed into a deep-learning net or simply queried to detect relationships between words.

## **What is Word Embeddings?**

## **Different types of Word Embeddings**

The different types of word embeddings can be broadly classified into two categories-

1. Frequency-based Embedding
2. Prediction based Embedding

(Check this: [**https://www.analyticsvidhya.com/blog/2017/06/word-embeddings-count-word2veec/**](https://www.analyticsvidhya.com/blog/2017/06/word-embeddings-count-word2veec/) for the above question’s answers)