What is fake news?

* The news which is fabricated to spread wrong propaganda and distract the people from the real news
* There is so much fake news in circulation, it is difficult to find sources of accurate and unfabricated news.

Dataset Information:-

* train.csv: A full training dataset with the following attributes:
* id: unique id for a news article
* title: the title of a news article
* author: author of the news article
* text: the text of the article; could be incomplete
* label: a label that marks the article as potentially unreliable
* 1: unreliable
* 0: reliable

test.csv: A testing training dataset with all the same attributes at train.csv without the label.

Data Pre-processing

* NLTK(Natural Language Toolkit) in python has a list of stopwords stored in 16 different languages.s the result of a search query.
* We are using porterstemmer which stems the words for example
* program : program
* programs : program
* programer : program
* programing : program
* programers : program
* A stop word is a commonly used word (such as “the”, “a”, “an”, “in”) that a search engine has been programmed to ignore, both when indexing entries for searching and when retrieving them as the result of a search query.

CounterVectorization

* The **CountVectorizer** provides a simple way to both tokenize a collection of text documents and build a vocabulary of known words, but also to encode new documents using that vocabulary. You can use it as follows: Create an instance of the **CountVectorizer**
* by using the counter vectorization we create a bag of words.
* For example : v = CountVectorizer(ngram\_range=(1, 2))
* "an apple a day keeps the doctor away“
* The vectorizing will be
* u'an': 0,
* u'an apple': 1,
* u'apple': 2,
* u'apple day': 3,

u'away': 4,

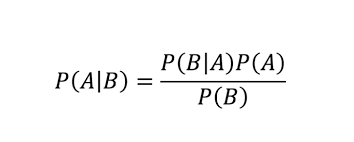
Data Normalisation

* **Normalization** is a technique often applied as part of **data** preparation for **machine learning**. The goal of **normalization** is to change the values of numeric columns in the dataset to use a common scale, without distorting differences in the ranges of values or losing information
* It helps to increase the weight of more important features.
* This is done by TFIDF (**term frequency inverse document frequency**)
* TF-IDF for a word in a document is calculated by multiplying two different metrics:
* The **term frequency** of a word in a document. There are several ways of calculating this frequency, with the simplest being a raw count of instances a word appears in a document. Then, there are ways to adjust the frequency, by length of a document, or by the raw frequency of the most frequent word in a document.
* The **inverse document frequency** of the word across a set of documents. This means, how common or rare a word is in the entire document set. The closer it is to 0, the more common a word is. This metric can be calculated by taking the total number of documents, dividing it by the number of documents that contain a word, and calculating the logarithm.
* So, if the word is very common and appears in many documents, this number will approach 0. Otherwise, it will approach 1.

What is the Naive Bayes Classifier:

* The Naive Bayes Classifier is a deterministic algorithm that uses the Bayes theorem to classify data. Let’s look at an example:

Suppose that you wanted to predict the probability that it would rain today: In the last few days, you have collected data by looking at the clouds in the sky.



Why do we optimize the hyperparameter

* For classifying query point in NB P(Y=1|W) or P(Y=0|W) (considering binary classification) here W is vector of words W= [w1, w2, w3.... wd] d = number of features
* So, to find probability of all these at training time  
  P(w1|Y=1) \* P(w2|Y=1) \*.....P(wd|Y=1)) \* P(Y=1)
* Same above should be done for Y=0.
* Now at testing time, consider you encounter word which is not present in train set then its probability of existence in a class is zero, which will make whole probability 0, which is not good.
* Consider W\* word not present in training set
* P(W\*|Y=1) = P(W\*,Y=1)/P(Y=1)
* So to get rid of this problem we do Laplace smoothing. we add alpha to numerator and denominator field.
* = 0 + alpha / Number of training point where Y=1 + (Number of class labels in classifier \* alpha)
* After applying the naïve bayes approach, the efficiency of the model we made had low accuracy and not efficient.
* So to increase the efficiency we used the Naïve bayes with classifier, and with the best value of the alpha as calculated when we use the model on the data we got high accuracy and highly efficient.

The **multinomial Naive Bayes classifier** is suitable for classification with discrete features (e.g., word counts for text classification). The **multinomial** distribution normally requires integer feature counts