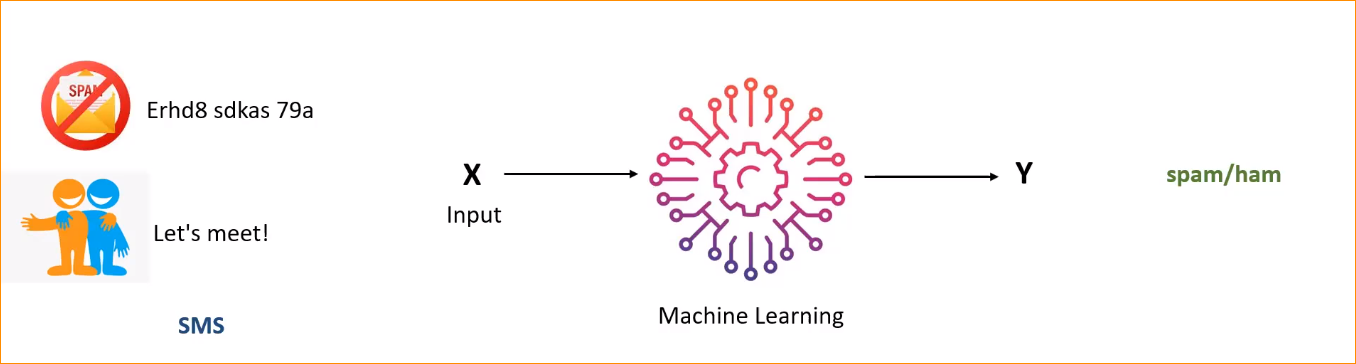
Here are several reasons why NLP is employed for spam classification:

* **Volume of Messages**: The sheer volume of messages sent daily makes manual filtering impractical. NLP offers an automated and scalable solution.
* **Dynamic Nature**: Spammers continually evolve their tactics, making it challenging to rely solely on predefined rules or keywords. NLP models can adapt and learn from new patterns.
* **Sophistication of Spams**: Modern spams often employ sophisticated tactics, including text obfuscation, linguistic nuances, and context-aware content, which require advanced NLP techniques for detection.
* **User Experience**: Unwanted messages can degrade the user experience, leading to frustration and potential mistrust in the platform. Effective spam classification enhances user satisfaction.
* **Security Concerns**: Spams are not merely annoying but can also pose security risks, including phishing attacks, malware distribution, and identity theft. Accurate spam detection helps mitigate these threats.
* **Regulatory Compliance**: Many jurisdictions have regulations regarding unsolicited electronic communications. Implementing robust spam classification mechanisms ensures compliance and avoids legal repercussions.
* **Resource Optimization**: By filtering out irrelevant or harmful messages, NLP-based spam classification helps in optimizing resources, such as storage, bandwidth, and computational power.
* **Multi-modal Content**: With the increasing use of multimedia content, including images, videos, and voice messages, NLP techniques can be extended to analyze and classify spams across various modalities.
* **Feedback Loop:** NLP-based spam classification systems can incorporate feedback loops where misclassified messages reported by users are used to retrain and improve the model's accuracy over time.
* **Integration with Other Systems:** NLP-based spam classification can be integrated with other security and communication systems, enhancing the overall infrastructure's robustness against various threats.



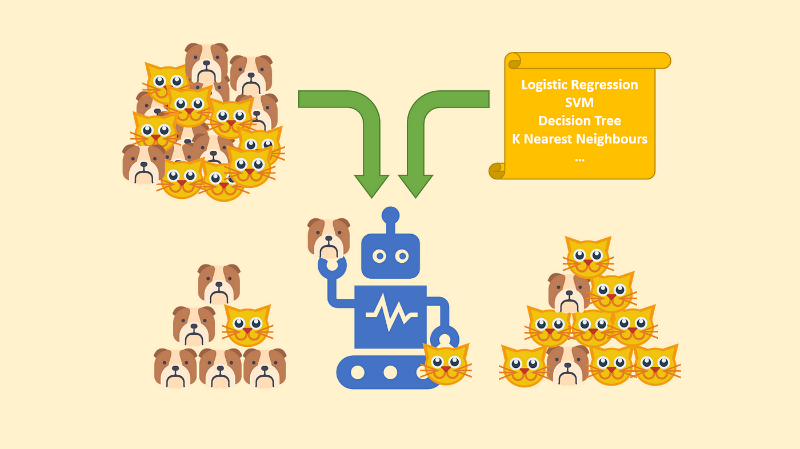
In this example – SMS or Email it can classified any text spam/ham

There are two classes (x and y) for our model, In here label details – data set

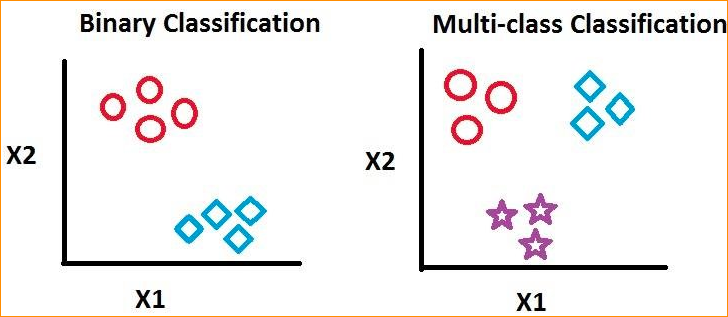
So we have to pre process the data set, then we will do the feature extraction and Train the ML model, test the model. So we call this problem as classification.

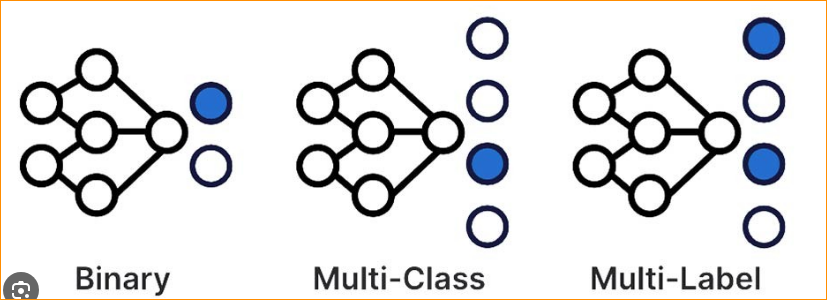
Model take desitions among pre define set of classes.

So the number of classes are to -🡪 call it as a binary classification



If there more than 2 classes we said multiclass classification.





Steps -

1. Split the data set in to two potions. For that use ratio of 80% to 20% or If want to use larger set data so can use 60% to 40%

In here I am divide 80 to training and 20 to testing.

Write a Code –

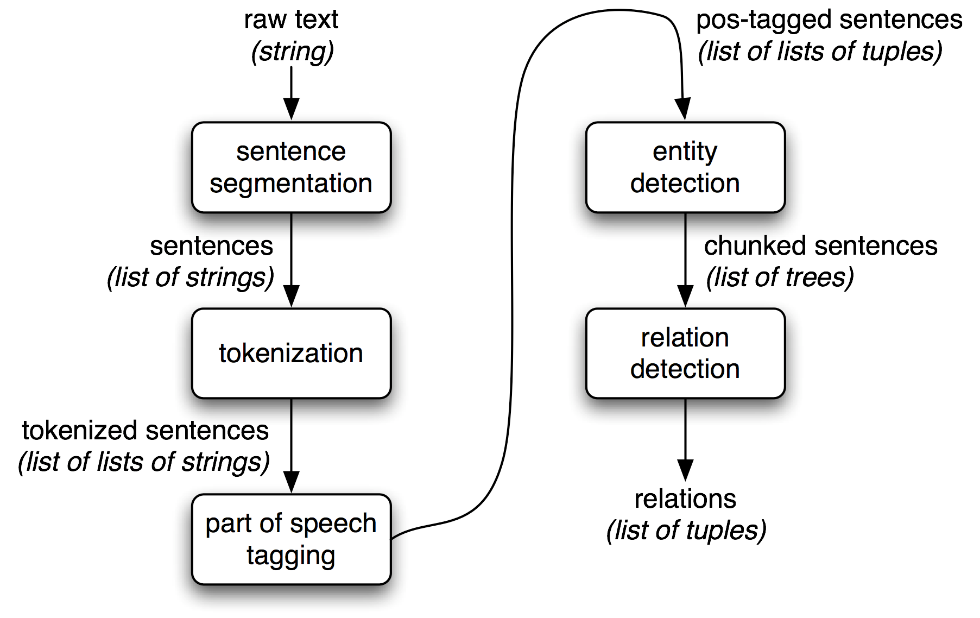
Using the pandas library to read the data.

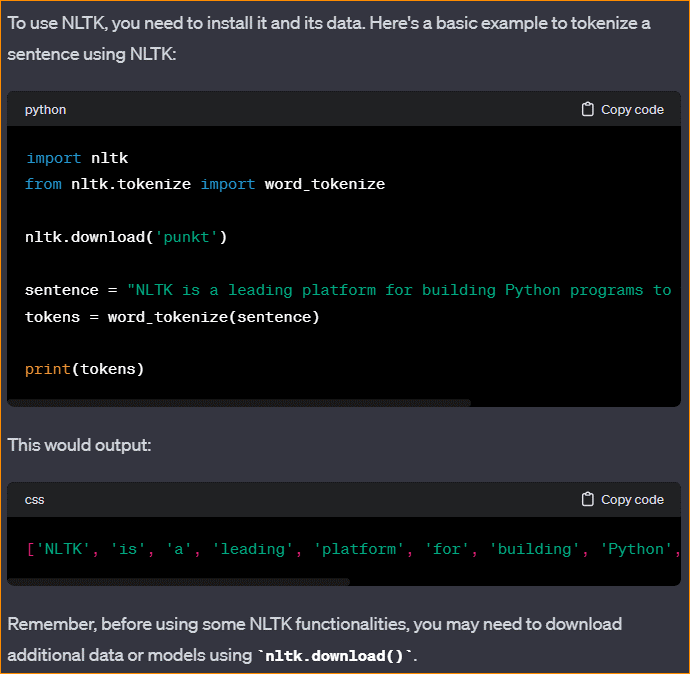
**Nltk (Natural Language Toolkit) –**

NLTK (Natural Language Toolkit) is a leading platform for building Python programs to work with human language data. It provides easy-to-use interfaces to over 50 corpora and lexical resources such as WordNet, along with a suite of text processing libraries for classification, tokenization, stemming, tagging, parsing, and more.

Here's a brief overview of some of the functionalities provided by NLTK:

* **Tokenization**: Breaking text into words, sentences, or other meaningful units.
* **Part-of-speech (POS) Tagging**: Assigning grammatical parts of speech to words in a sentence (e.g., noun, verb, adjective).
* **Named Entity Recognition (NER):** Identifying entities in text such as persons, organizations, locations, etc.
* **Stemming and Lemmatization:** Reducing words to their base or root form.
* **Parsing:** Analyzing the grammatical structure of sentences.
* **Sentiment Analysis:** Determining the sentiment or emotion conveyed by a piece of text.
* **WordNet Integration:** Access to WordNet's lexical database, which includes semantic relationships between words.





Note End

Load the data –

Panda read the data and create the data frame.



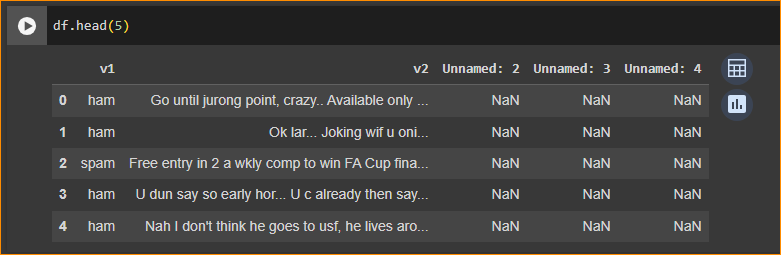
You Should change your location(Upload CSV file to google drive)

In here Dataset taking from – [www.kaggle.com](http://www.kaggle.com)

In this code when you use csv file, should use encoding methods.

1. Let visualize the data set

In here, give first 5 data column.



Column – V1, V2, Unnamed : 2, Unnamed : 3, Unnamed : 4

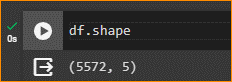
Row – Single messages that corresponding label.

V1 – Spam or ham

V2 – message . (It is not structured data)

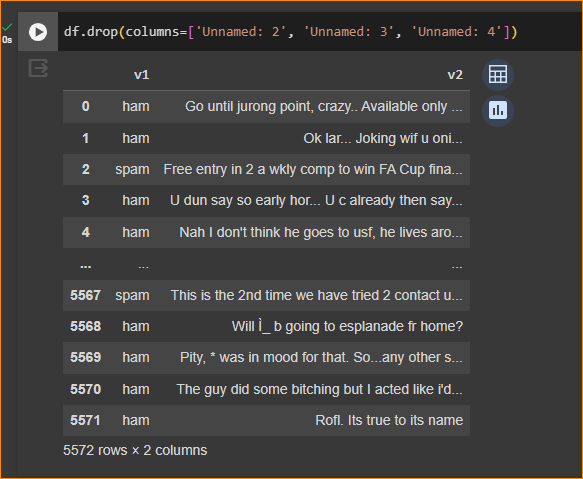
Unnamed columns we don’t want so we drop that.

1. Data set Size



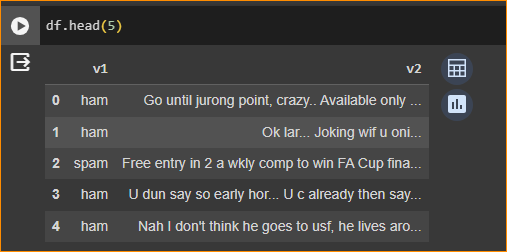
Rows 5572 and Column = 5

1. Drop the unwanted columns

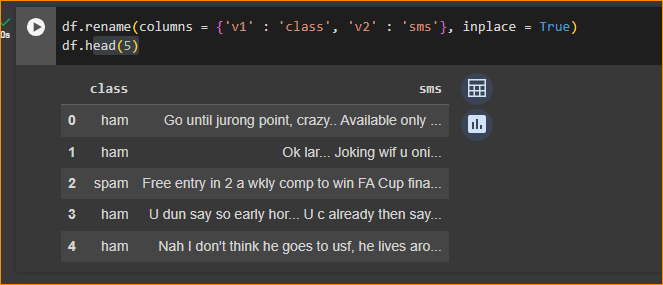


And put inplace = true that mean replace the original one (df)

Check –

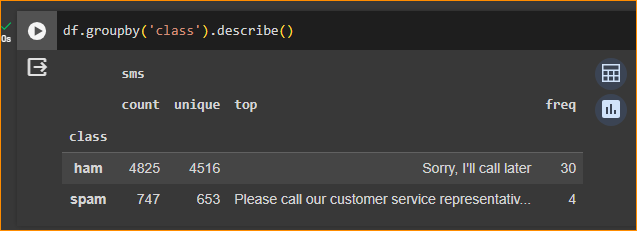


1. Rename v1 and v2 to meaning full name-



If you want to get random data can use = df.sample()

1. Group by the calss



In here 747 “spam” sms and 4825 ham sms

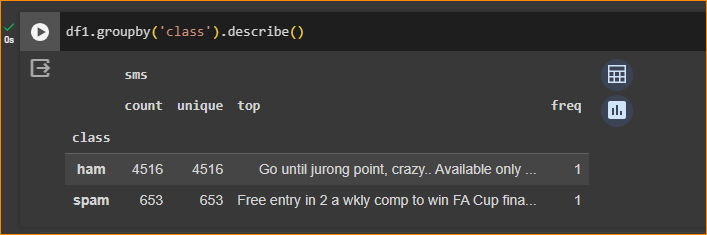
Unique column show how many unique columns are there.

Most occurring duplicate one is “Sorry, I will call later”

1. Remove the duplicate-

And here I have kept the 1st one.

Now can see we successfully remove Di



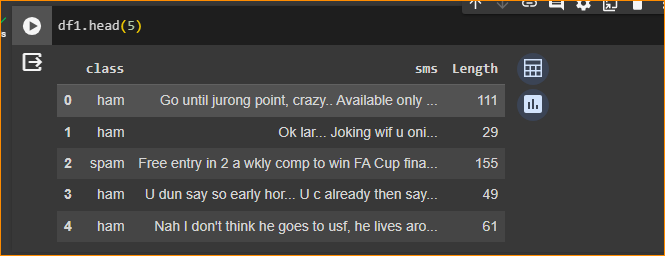
1. Data Set Visualization



Create a new column call “Length” -

How the dataset look like?

* Create new column called – length
* Which will count the length of SMS
* In here what happen, 1st SMS count the length and it will apply and the result will store under the length column. 2nd SMS count length and store under the length column.



1. Visualize the data

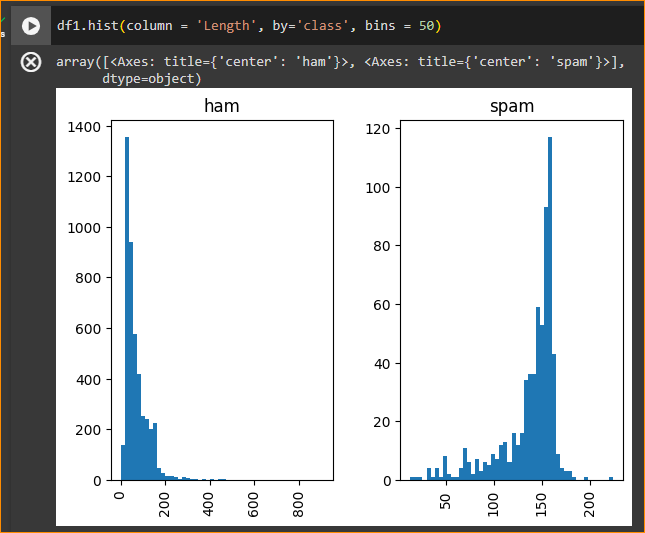


Here using histogram to visualize the data.

There are plot details about column length.

How does the length vary in the histogram?

So create separate histogram for ham and Spam



30

In here most of ham messages are short – most messages around 30 length

But spam messages around 150-180 length.

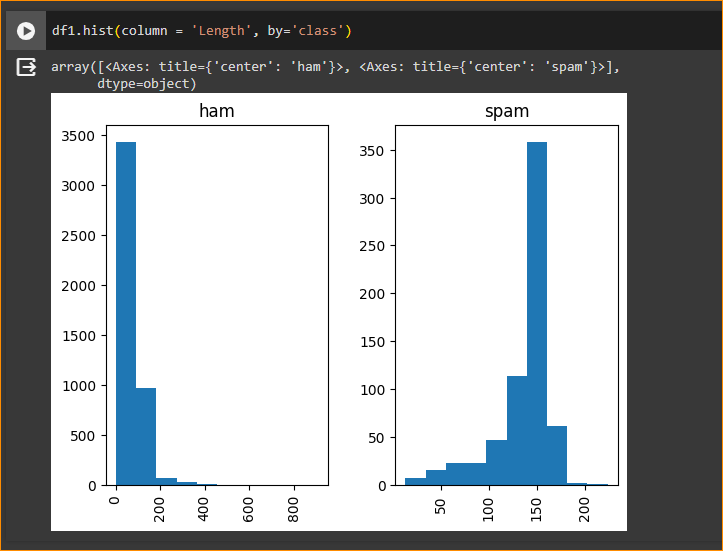
So in here SMS have more characters, model may be decided that is spam message.

bins=50: Specifies the number of bins to use in the histograms.

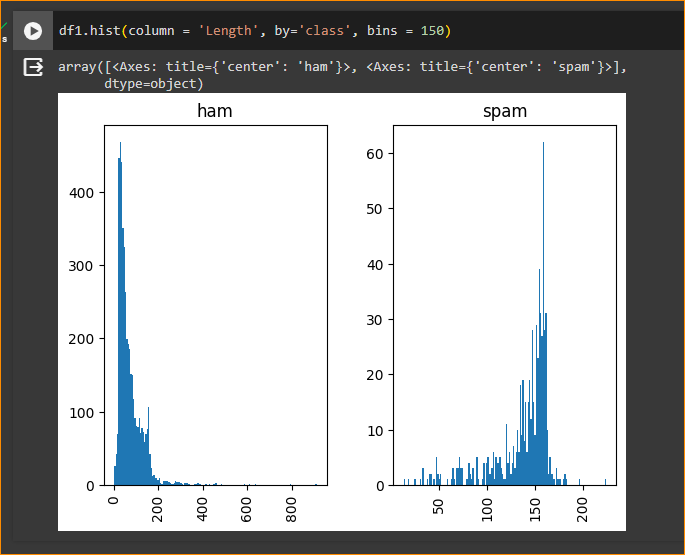
That mean between 0-50 How many increases are there?

So it will plot corresponding height.

Without bins – Cannot get Idea.



Bins = 150



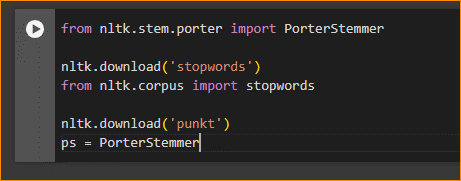
1. Preprocessing

For that I use nltk.stem.porter

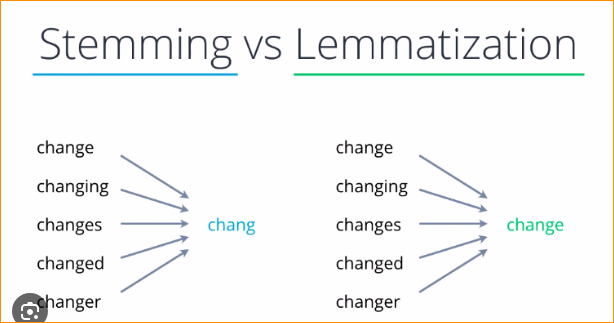
Note Start -

The **nltk.stem.Porter** module in the **Natural Language Toolkit (NLTK)** provides the Porter stemming algorithm. Stemming is the process of reducing words to their root or base form. The Porter stemming algorithm is one of the most commonly used stemming algorithms.

Note End –

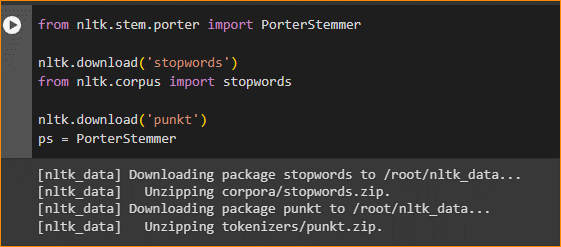


In here we will use Stemming but also use Lemmatization.

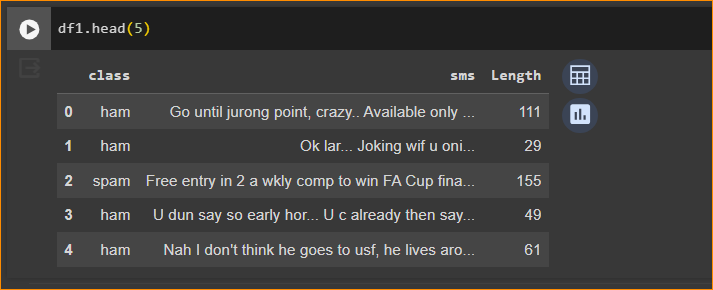


Nltk has their own stop words, can customize particular domain. We will donload that and use that stop word of nltk.

‘punkt’ – To do stammer there should be download the rules.

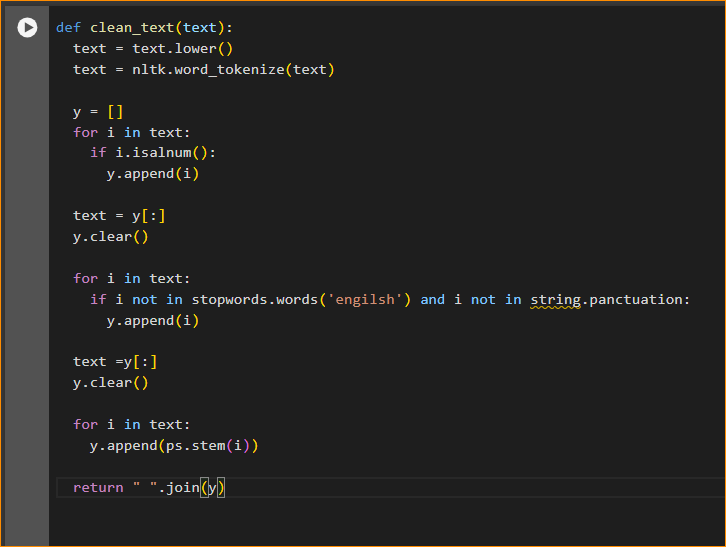


Data before pre-processing-

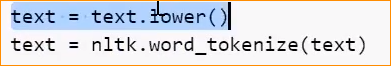


So what are the Preprocessing tasks –

1. Lower Case
2. Tokenization
3. Removing special characters
4. Removing stop words and punctuation
5. Stemming



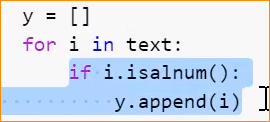
* Here I have defined the function call – Clean\_text
* Here we pass the text and function do cleaning process.



Take the lowercase of the SMS.

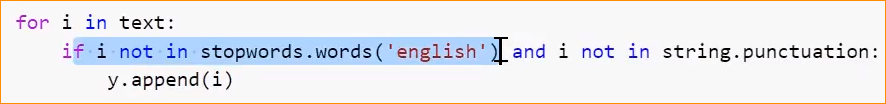
Then we apply tokenization.

Can use nltk tokenizer. This will be working on the white space.



Here checking the text contain only Alfa numeric values. (alperbert or numbers), if there any special characters it will drop. characters are . After that we will append that in to the collection.

We are gathering all the valid tokens here.



In here checking whether it stop word or not.

If it is not stop word we will consider that token.

Then we check it is punctuation or not if it has we will removing that. If will not both append to that in to list.



Finally, we adding a stemming function.

Finally, we recreate the text, using all the tokens using white spaces.

1. Now we apply those clean function each SMS on the data set.



In here go to SMS column and take each SMS and put those in to function.

And Create a new column call SMS clean.

Original SMS vs Clean SMS –

Removed stop word, punctuation all and we are keeping the numbers.

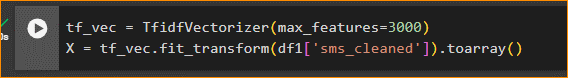


Feature Extraction –



In here we use Tf-IDF vectorization. For the feature extraction.

For that we Sklearn library. It has so many machine learning algorithms as well.



Define vectorizer and define how many features that we want (max\_features = 3000)

Based on the size and frequency of the word we will be taking that the top 3000.

In here 1st 3000 tokens are considered as a vectorization.

We add to that transformation to column and add to array

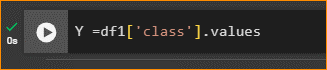


In here 5169 SMS es available and each SMS we converted a 3000-dimensional vector. That means vocabulary size is 3000.

This is basically in 2D array.

Rows Represent as a SMSes columns represent the tokens in the vocabulary. That Size is 3000.

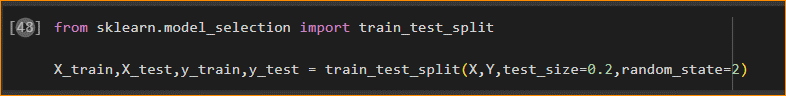
In here X is input your model.



In our Output is the class column (ham or spam)

1. Learning

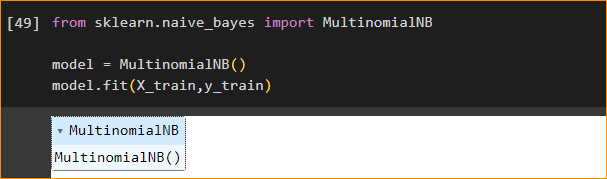
This Split data in to two potions.



We have split data set in to two potions.

Now sklearn has its own split method.

Can utilize and can give test size 0.2 (20%) for testing. And 80 % for training

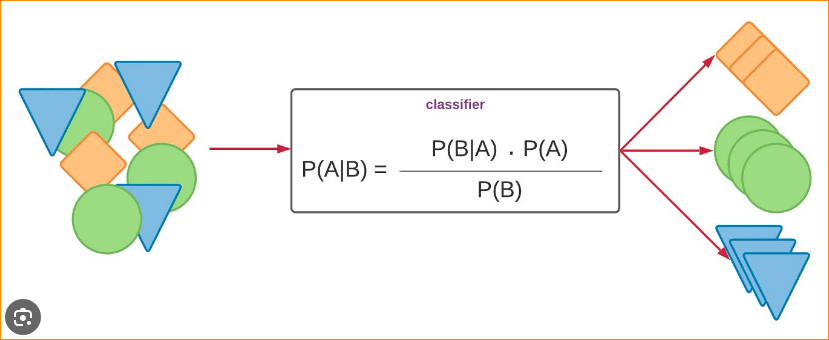


Here I used algorithm call “naïve\_bayes”

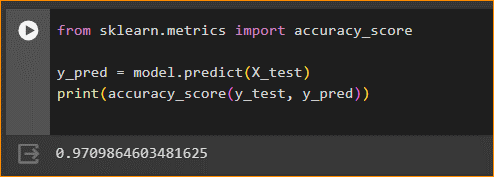
In here Will give training data for the training portion. Model will learn using the X and Y.

Note Start about naïve\_bayes

The Naïve Bayes algorithm is a family of probabilistic algorithms based on the Bayes' theorem, with an assumption of independence between features. Despite its simplicity, Naïve Bayes classifiers often work well in many complex real-world situations, such as text classification and spam filtering.



Note End -----



In here

Will give only X to the model and and check what is the Y

And check the accuracy

In here we will ask, what is the Y for available in test data.

The model Give Y

But we have original test data.

We have original one from the data set(y\_test- y value in the training data )

We will compare those two (y\_test and y\_pred) how many are correct.

That is simple our accuracy.

Answer – 0.97 that mean 97% predictions are correct.