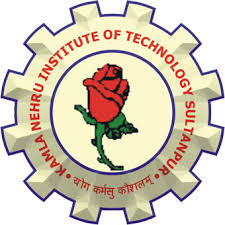
# REPORT ON:

**SPAM CLASSIFIER**

A MACHINE LEARNING PROJECT

Assignment 2



Submitted by :

Aviral Singh (18216)

Chakresh Varshney (18217)

Divyanshu Shukla (18219)

Garima Ahuja (18219)

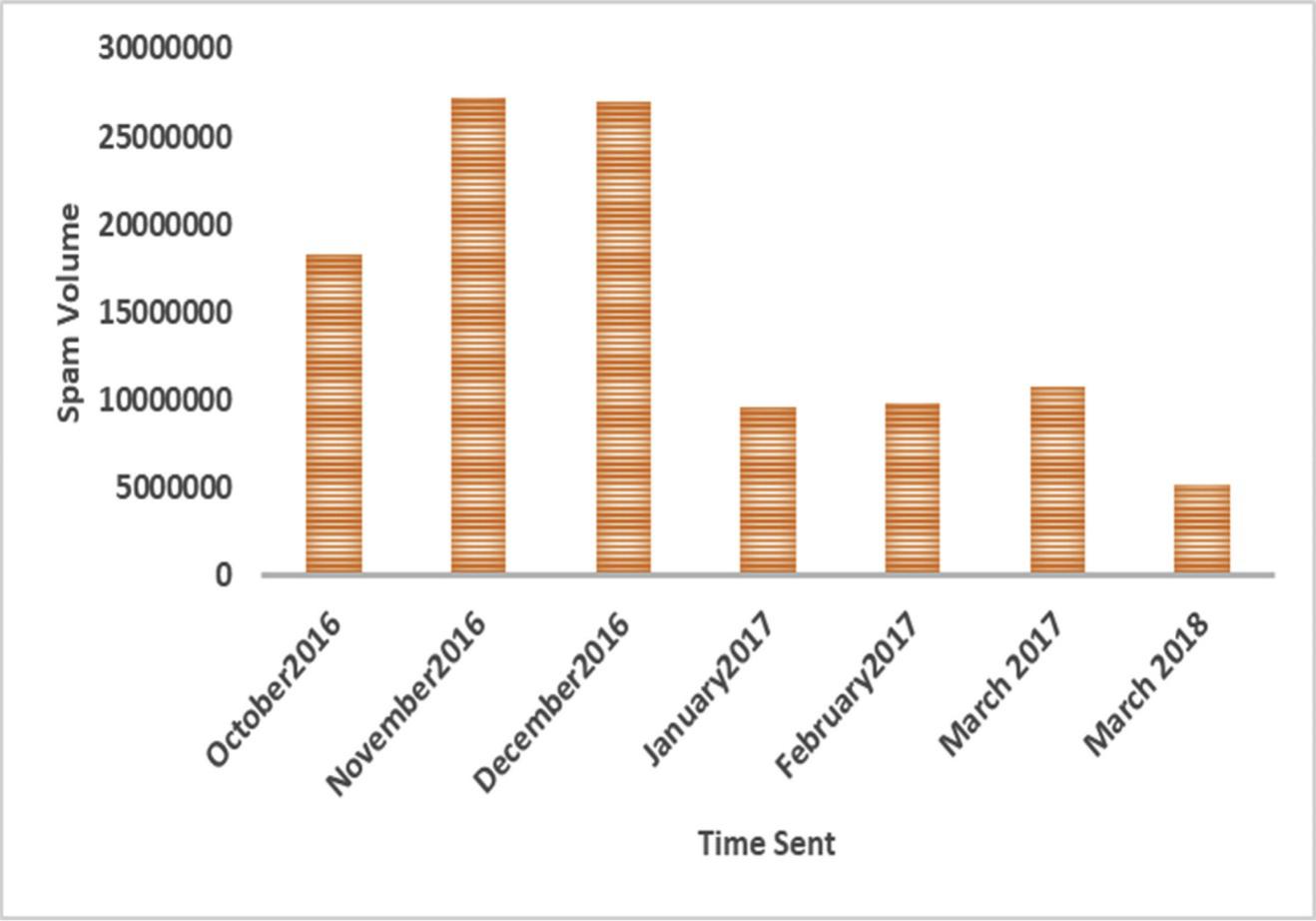
Submitted to :

Mr AWADESH SIR

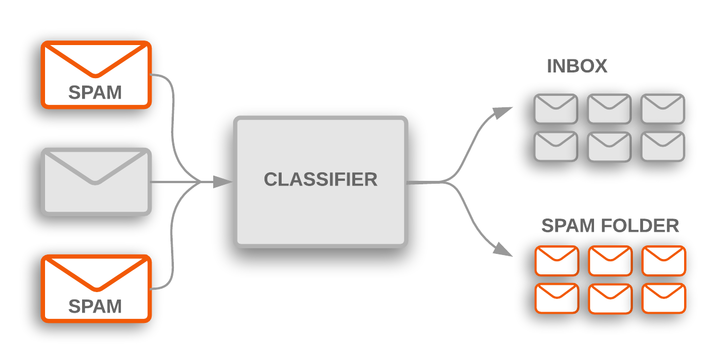
# **Introduction**

* In recent times, unwanted commercial bulk emails called spam has become a huge problem on the internet. The person sending the spam messages is referred to as the spammer. Such a person gathers email addresses from different websites, chatrooms, and viruses.
* Spam prevents the user from making full and good use of time, storage capacity and network bandwidth. The huge volume of spam mails flowing through the computer networks have destructive effects on the memory space of email servers, communication bandwidth, CPU power and user time.
* The menace of spam email is on the increase on yearly basis and is responsible for over 77% of the whole global email traffic.
* Users who receive spam emails that they did not request find it very irritating. It is also resulted to untold financial loss to many users who have fallen victim of internet scams and other fraudulent practices of spammers who send emails pretending to be from reputable companies with the intention to persuade individuals to disclose sensitive personal information like passwords, Bank Verification Number (BVN) and credit card numbers.

The volume of spam emails containing malware and other malicious codes between the fourth quarter of 2016 and first quarter of 2018 is depicted in figure below.

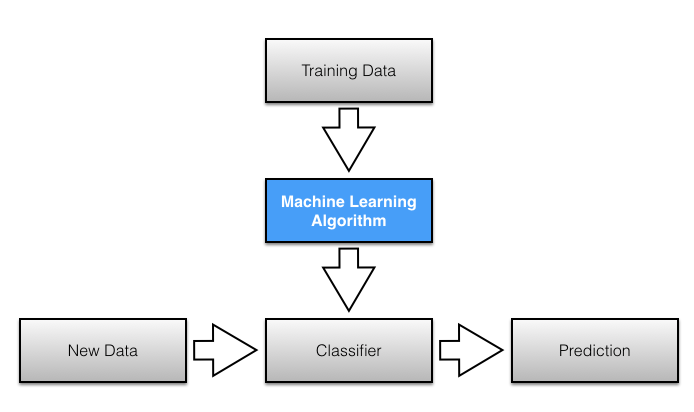


## **What our model Does:**



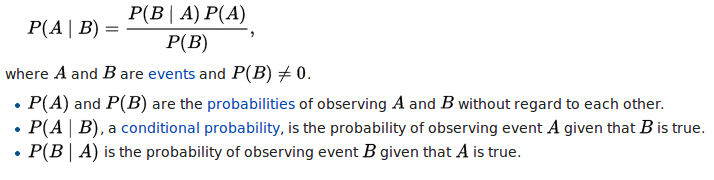
## **How It works:**

* Extract the text and the target class from the dataset. Extract the features of the test using naive Bayes vectorizer for the Input features. Split the skewed data into shuffled sets using stratified shuffle split in sklearn library. Use standard classifiers to classify the data into spam or ham.

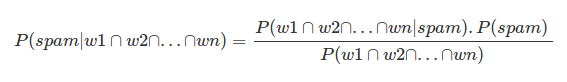


## **Basic Theory:**

**Bayes Theorem:** We can do this by using a simple, yet powerful theorem from probability theory called [Baye’s Theorem](https://en.wikipedia.org/wiki/Bayes%27_theorem). It is mathematically expressed as



We have a message m = (w*1*, w*2*, . . . . , w*n*), where (w*1*, w*2*, . . . . , w*n*) is a set of unique words contained in the message. We need to find



If we assume that occurrence of a word are independent of all other words, we can simplify the above expression to

Image for post

In order to classify we have to determine which is greater

Image for post

|  |
| --- |
| **Naïve Bayes Classification Algorithm for Email Spam Classification** |
| 1: Input Email Message dataset 2: Parse each email into its component tokens 3: Compute probability for each token S [W] = Cspam(W)/(Cham(W) + Cspam(W)) 4: Store spamminess values to a database 5:for each message M do 6: while (M not end) do 7: scan message for the next token Ti 8: query the database for spamminess S(Ti) 9: compute probabilities of message collected S [M] and H [M] 10: compute the total message filtering signal by: I [M] = f (S [M], H [M]) 11: *I[M]=(I+S[M]−H[M])/2*  12*:* if I [M] > threshold then 13: msg is labeled as spam 14: else 15: msg is labeled as non-spam 16: end if 17: end while 18: end for 19: return Final Email Message Classification (Spam/Valid email) 20: end |

**STOP WORDS:** There are some English words which appear very frequently in all documents and so have no worth in representing the documents. These are called STOP WORDS and there is no harm in deleting them. Example: the, a, for etc. There is also some domain specific (in this case email) stop words such as mon, true, email, sender, from etc. So, we delete these words.

**STEMMING:** The next step to be performed is stemming. Stemming is used to find a root of a word and thus replacing all words to their stem which reduces the number of words to be considered for representing a document. Example: sings, singing, sing have sing as their stem.

## **Prerequisites:**

* Python
* scikit-learn / sklearn
* Pandas
* NumPy
* Matplotlib
* NLP library
* An IDE to work in - something like Jupyter or Spyder.

## **Dataset:**

In this project we have taken [dataset](https://www.kaggle.com/uciml/sms-spam-collection-dataset) from Kaggle website.

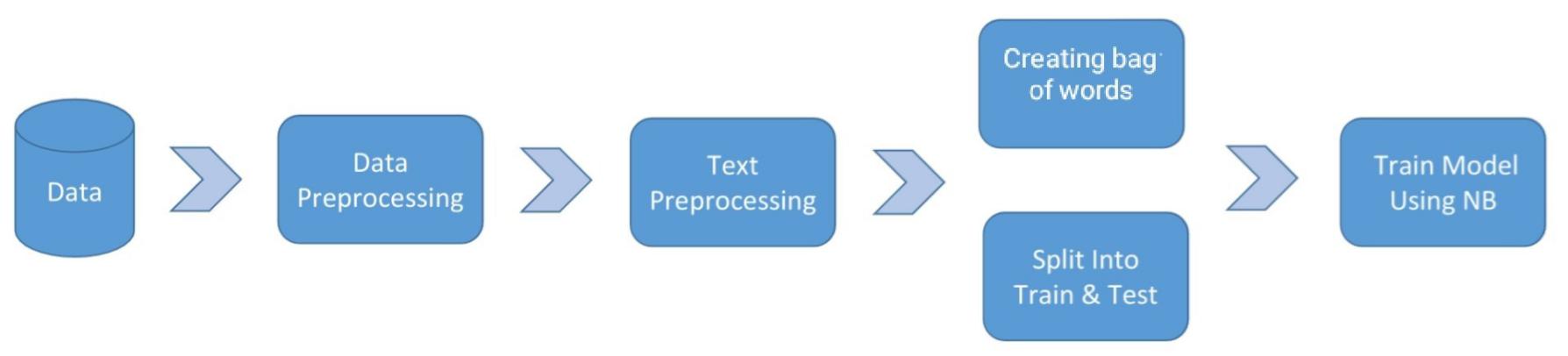
The files contain one message per line. Each line is composed by four columns:

* Class- contains the label (ham or spam) (named as v1)
* Message - contains the raw text. (named as v2)
* Two columns with null values. (named as v3 and v4)

The SMS Spam Collection is a set of SMSs tagged messages that have been collected for SMS Spam research. It contains one set of SMS messages in English of 5,574 messages, tagged according being ham (legitimate) or spam.

Class type: SPAM- 87% and HAM-13%

## **Model Pipeline:**

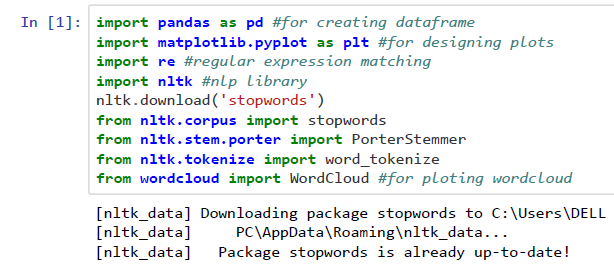


## **Components:**

* Using naive Bayes for feature extraction of the text data for the messages.
* Use splits for skewed data (Since the number of hams are far more than the number of spam messages, the data is skewed)
* Use stratified shuffled split for the split of skewed data.
* Use different standard classifiers for classification of the SMS.
* Compare the accuracy of various classifiers using standard classification metrics.

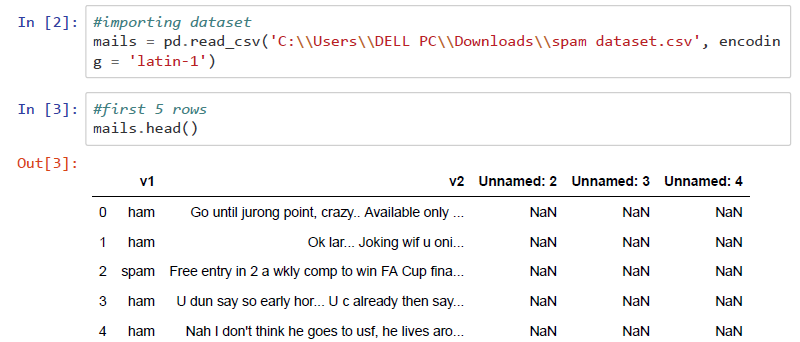
**Procedure:**

**1. Loading dependencies**

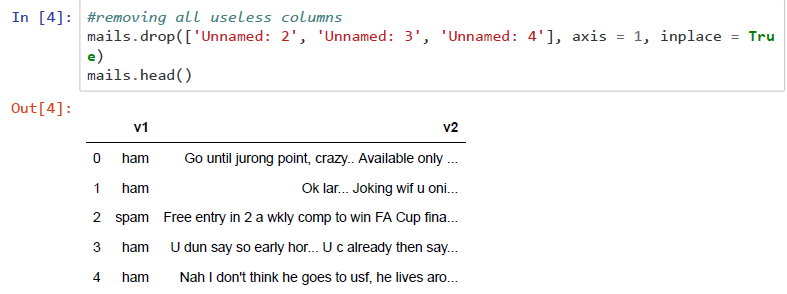


We are going to make use of NLTK for processing the messages, Word Cloud and matplotlib for visualization and pandas for loading data, NumPy for generating random probabilities for train-test split.

**2. Loading Data-set**



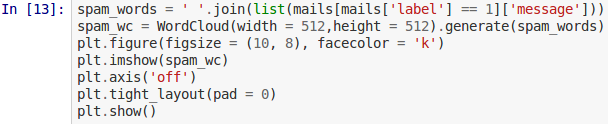
We do not require the columns ‘Unnamed: 2’, ‘Unnamed: 3’ and ‘Unnamed: 4’, so we remove them. We rename the column ‘v1’ as ‘label’ and ‘v2’ as ‘message’. ‘ham’ is replaced by 0 and ‘spam’ is replaced by 1 in the ‘label’ column. Finally, we obtain the following data frame.



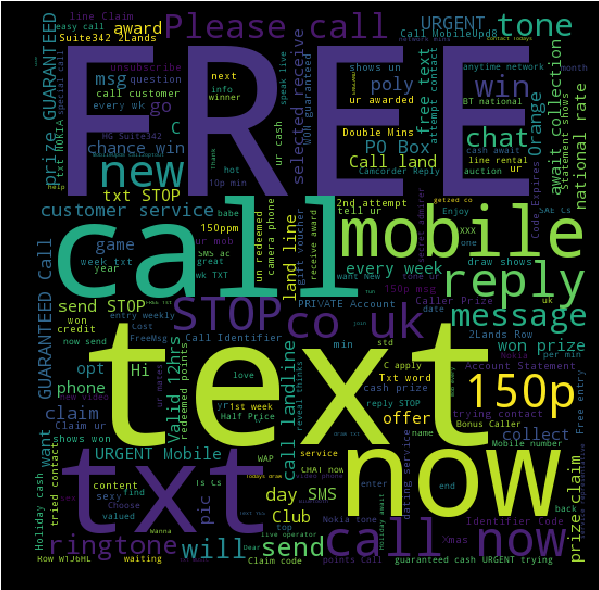


**3. Visualizing data**

Let us see which are the most repeated words in the spam messages! We are going to use [WordCloud](https://github.com/amueller/word_cloud) library for this purpose.



This results in the following



As expected, these messages mostly contain the words like ‘FREE’, ‘call’, ‘text’, ‘ringtone’, ‘prize claim’ etc.

Similarly, the word cloud of ham messages is as follows:



## **4.Creating the model**

We are going to implement Bag of words technique.

**Preprocessing**:Before starting with training we must preprocess the messages. First of all, we shall make all the character lowercase. This is because ‘free’ and ‘FREE’ mean the same and we do not want to treat them as two different words.

Then we tokenize each message in the dataset. Tokenization is the task of splitting up a message into pieces and throwing away the punctuation characters. For eg.:

Image for post

The words like ‘go’, ‘goes’, ‘going’ indicate the same activity. We can replace all these words by a single word ‘go’. This is called stemming. We are going to use [Porter Stemmer](https://tartarus.org/martin/PorterStemmer/), which is a famous stemming algorithm.

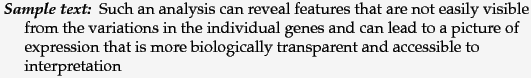
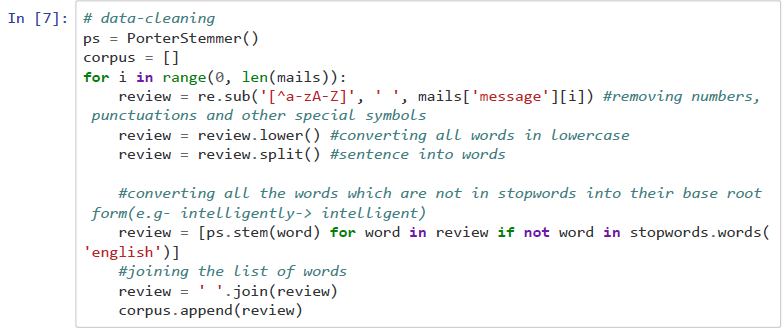


Image for post

We then move on to remove the stop words. Stop words are those words which occur extremely frequently in any text. For example words like ‘the’, ‘a’, ‘an’, ‘is’, ‘to’ etc. These words do not give us any information about the content of the text. Thus, it should not matter if we remove these words for the text.

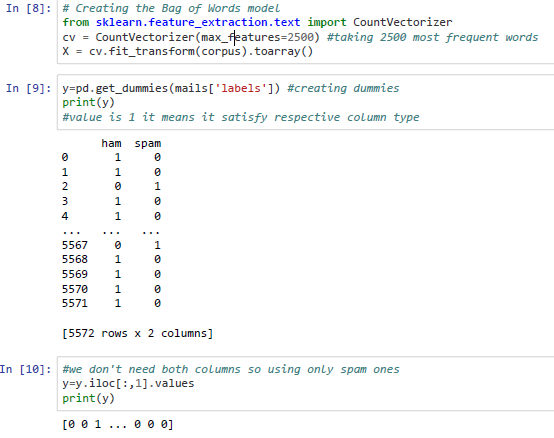


**Bag of Words**: In Bag of words model we find the ‘term frequency’, i.e. number of occurrences of each word in the dataset. Thus, for word w,

Image for post

and

Image for post



## **5.Train-Test Split**

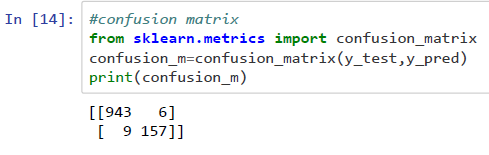
Here we are splitting imported into 80-20. 80% for training and 20% for testing.

## 

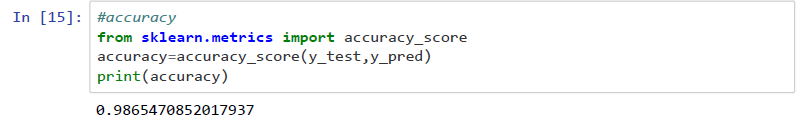
6.Training the model

## 

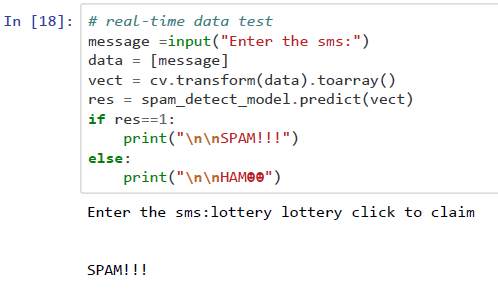
7.Confusion matrix



8.Accuracy Result



9.Input testing



10.Conclusion

In this report, we discussed how the classification process works. From the conducted experiment we can say that the use of Naïve Bayes classifier is considered to be the best option for email classification as it has high accuracy and precision score. Due to the ease in implementation of naïve Bayes algorithm, it is suitable to use in creating the web application for classification of emails.

11.References

<https://www.kaggle.com/muzzzdy/sms-spam-detection-with-various-classifiers>