Faculty of Engineering & Technology
Electrical & Computer Engineering Department

ARTIFICIAL INTELLIGENCE
ENCS3340

Automatic Tweet Spam Detection

Prepared by:
ShahEd Jamhour 1180654, Yasmeena Assi 1180899 & Rand Farhoud 1181567

Instructor: Dr. Aziz Qaroush
Date: January 2022.

Abstract

The aim of this project is to learn about machine learning techniques and apply them to real-life applications. The main task is to develop an automated tweet spam detector by pre-processing the data, extracting features, and building different classifiers (Decision tree, Naive Bays, and Random Forest). Testing of data will be carried out using a Python program with the help of the Scikit-Learn and using Google Collaboratory.

Table of Contents

Introduction

1. Machine learning

The field of machine learning is an extension of artificial intelligence and computer science which uses data and algorithms to imitate human learning and gradually improve its accuracy.

There are three primary categories of machine learning classifiers:

1. Supervised machine learning

   Supervised Machine learning is defined by its use of labeled datasets to train algorithms to classify data or predict outcomes accurately. As input data is fed into the model, it adjusts its weights until the model has been fitted appropriately. This type of algorithm is used for classification problems e.g. Decision Tree, Artificial neural network

2. Unsupervised machine learning

   It uses machine learning algorithms to analyze and cluster unlabeled datasets. These algorithms discover hidden patterns or data groupings without the need for human intervention.

3. Semi-supervised learning

   Semi-supervised learning offers a happy medium between supervised and unsupervised learning. During training, it uses a smaller labeled data set to guide classification and feature extraction from a larger, unlabeled data set

2. Automatic Spam Tweet Detection

Recent years have seen an increase in the popularity of social networking sites. Twitter is considered to be the fastest-growing site so spammers take advantage of this popularity by using Twitter for their malicious purposes.

The term spam refers to tweets posted by known fake Twitter accounts that are politically motivated, automatically generated, or clickbait in nature.

With both manual and automated tools, Twitter provides an environment free of spam and According to Twitter, spam detection features fall into three categories: (1) account-based features, (2) tweet-based features, and (3) relationship between the tweet's sender and receiver.

These features include number of hashtags,mentions and URLS the tweet has, the ratio of the number of following and the number of followers, average tweets per day,, etc.

3. Scikit-Learn Library

The Sklearn library is a collection of machine learning algorithms for data mining tasks. With Sklearn, you can preprocess data, classify, predict, cluster, and create association rules. Sklearn supports data preprocessing, classification, and feature selection, which are all standard data mining tasks.

Related work:

Methodology:

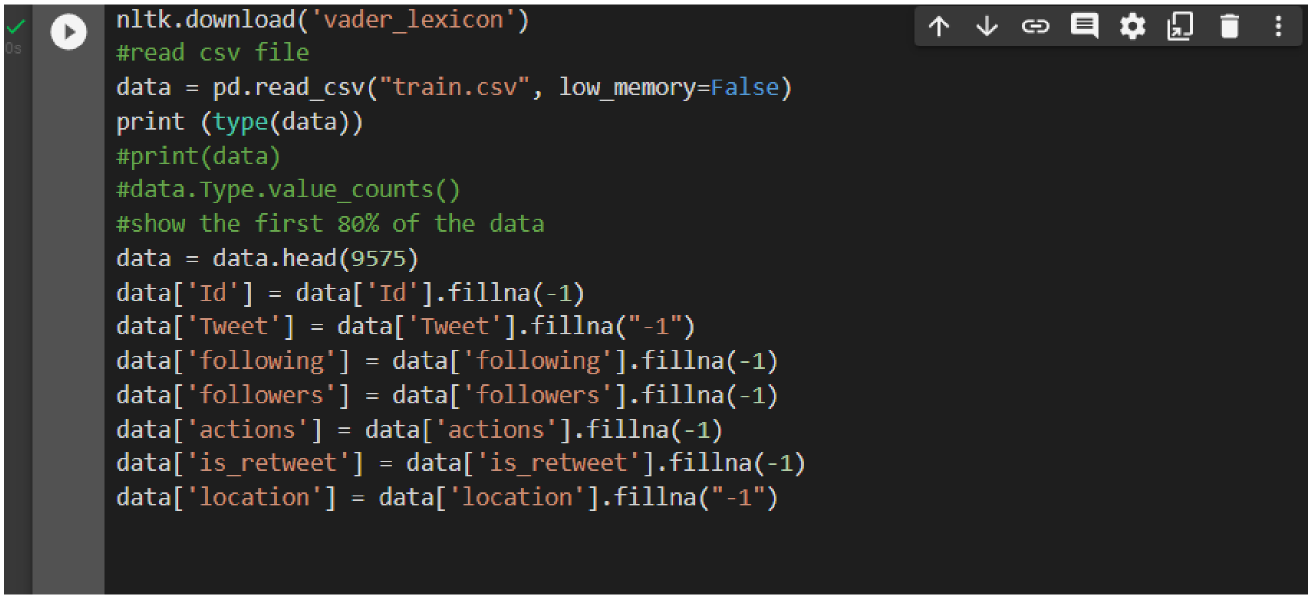
In this part , we are going to summarize briefly , what we did to the tweets , in order to detect the spam in the tweet.
to do that,these steps were done:
1. raw data was read , then we implemented the pre-processing .
2. feature extraction techniques were implemented .
3.we have chosen the model to train our data , and made sure to balance our data , then trained it .

To do the above steps , we used many python libraries , and we used Google colaboratory , because here we have a large number of data, so it provides us a suitable size memory to do the feature extraction . Running python script takes a lot af computing power and a lot of time , but here we allowed to use the googl e's server computing power, so we don't need to be worry about time .

Here a description for each step in our project:

A. Reading data

The CSV file which holds the data was read using Pandas Library and stored in the variable "data" as shown in fig below.



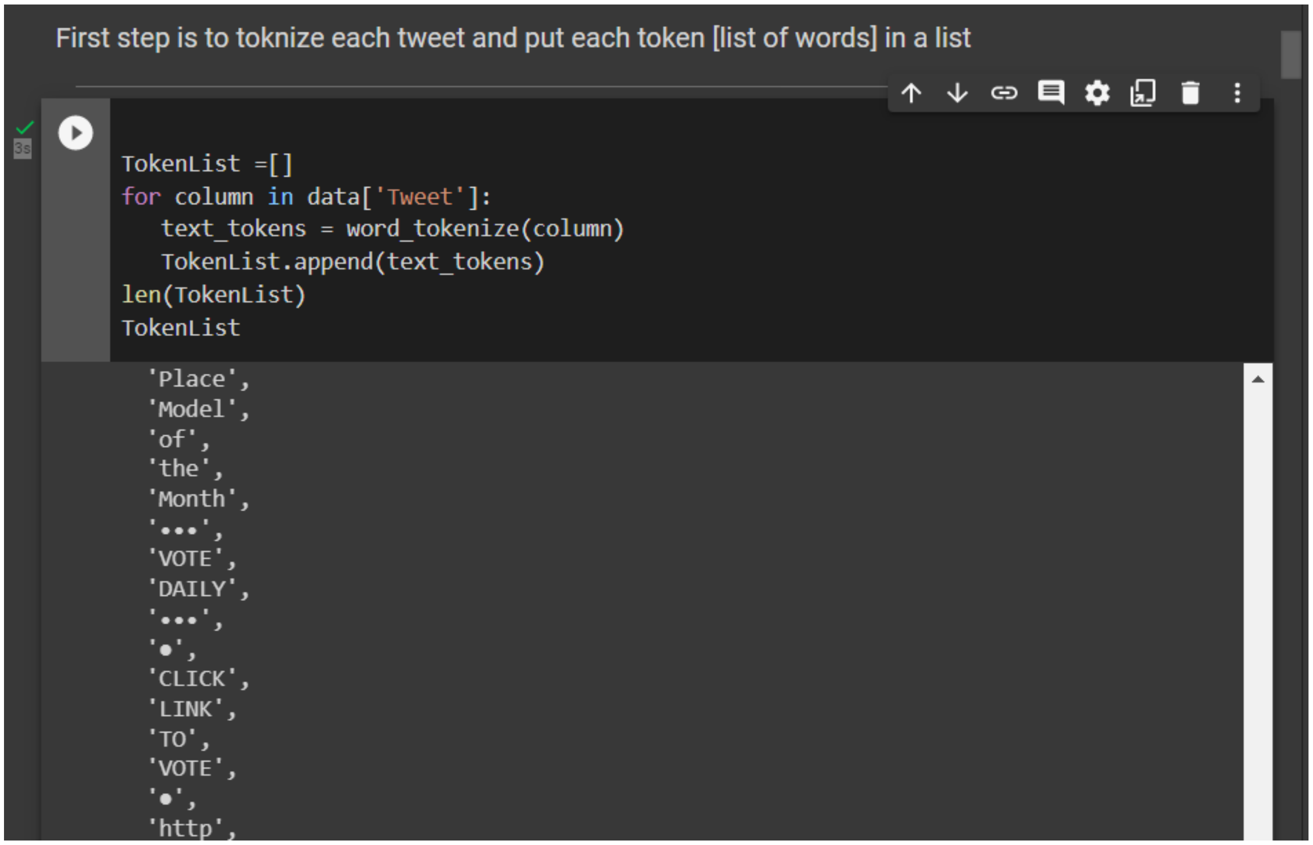
```
nltk.download('vader_lexicon')
#read csv file
data = pd.read_csv("train.csv", low_memory=False)
print (type(data))
#print(data)
#data.Type.value_counts()
#show the first 80% of the data
data = data.head(9575)
data['Id'] = data['Id'].fillna(-1)
data['Tweet'] = data['Tweet'].fillna("-1")
data['following'] = data['following'].fillna(-1)
data['followers'] = data['followers'].fillna(-1)
data['actions'] = data['actions'].fillna(-1)
data['is_retweet'] = data['is_retweet'].fillna(-1)
data['location'] = data['location'].fillna("-1")
```

Only 70% of the data(which equals 8378Rows) was trained using data.head function. Then, Nan values were handled by Substituting the Nan value with (-1) using fillna() function as shown above. This step was done because The sklearn implemented algorithms can't perform on datasets that have such values.

3.1 step 1: Pre-Processing

Here in this step , raw data was organized before building the trained model , this step makes us more comfortable with data , also it enhances the quality of the data .So , here are the steps of the pre-processing :

1.Tekonization: in this step, Sensitive data was exchanged for nonsensitive data,Here we are going to divide the text into chunks, called tokens.In order to use it to remove the stop words .

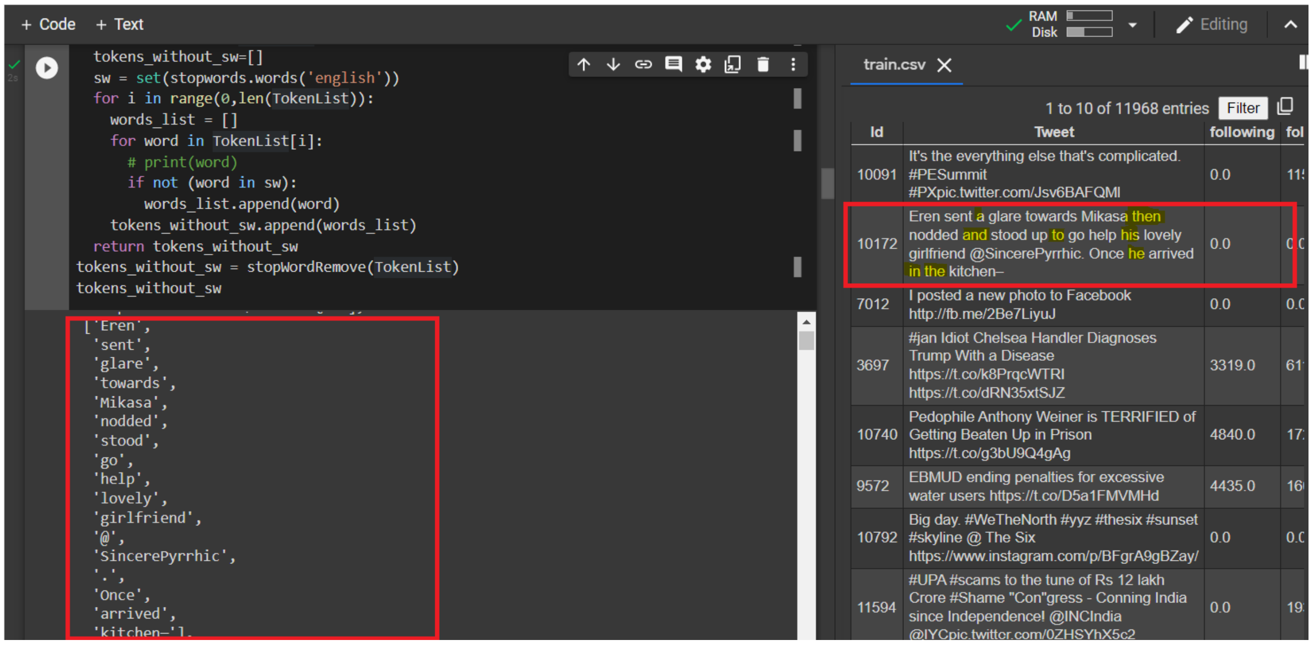

First step is to toknize each tweet and put each token [list of words] in a list

```
TokenList =[]
for column in data['Tweet']:
   text_tokens = word_tokenize(column)
   TokenList.append(text_tokens)
len(TokenList)
TokenList
```

```
  'Place',
  'Model',
  'of',
  'the',
  'Month',
  '•••',
  'VOTE',
  'DAILY',
  '•••',
  '●',
  'CLICK',
  'LINK',
  'TO',
  'VOTE',
  '●',
  'http',
```

The tweet was tokenized using **word_tokenized** function. Each word was tokenized as a list element as can be seen in the fig above.

2. Removing stop words from the tweet content: Here in this step we are going to remove the stop words like (are,the,is,am, then , a… etc), from the tweet content , this step is important ,we are going to use it in lemmatization, and when we get the similarity afterwords.



```
tokens_without_sw=[]
sw = set(stopwords.words('english'))
for i in range(0,len(TokenList)):
  words_list = []
  for word in TokenList[i]:
    # print(word)
    if not (word in sw):
      words_list.append(word)
  tokens_without_sw.append(words_list)
return tokens_without_sw
tokens_without_sw = stopWordRemove(TokenList)
tokens_without_sw
```

```
['Eren',
 'sent',
 'glare',
 'towards',
 'Mikasa',
 'nodded',
 'stood',
 'go',
 'help',
 'lovely',
 'girlfriend',
 '@',
 'SincerePyrrhic',
 '.',
 'Once',
 'arrived',
 'kitchen–'],
```

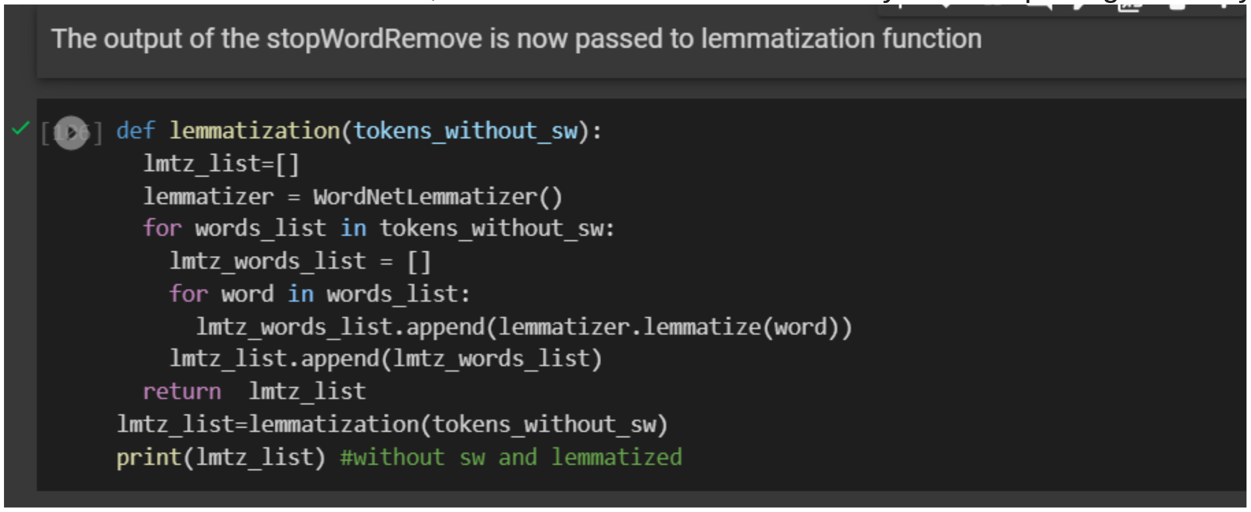
train.csv

1 to 10 of 11968 entries

Id	Tweet	following	fol
10091	It's the everything else that's complicated. #PESummit #PXpic.twitter.com/Jsv6BAFQMI	0.0	11
10172	Eren sent a glare towards Mikasa then nodded and stood up to go help his lovely girlfriend @SincerePyrrhic. Once he arrived in the kitchen–	0.0	0.0
7012	I posted a new photo to Facebook http://fb.me/2Be7LiyuJ	0.0	0.0
3697	#jan Idiot Chelsea Handler Diagnoses Trump With a Disease https://t.co/k8PrqcWTRI https://t.co/dRN35xtSJZ	3319.0	61
10740	Pedophile Anthony Weiner is TERRIFIED of Getting Beaten Up in Prison https://t.co/g3bU9Q4gAg	4840.0	17
9572	EBMUD ending penalties for excessive water users https://t.co/D5a1FMVMHd	4435.0	16
10792	Big day. #WeTheNorth #yyz #thesix #sunset #skyline @ The Six https://www.instagram.com/p/BFgrA9gBZay/	0.0	0.0
11594	#UPA #scams to the tune of Rs 12 lakh Crore #Shame "Con"gress - Conning India since Independence! @INCIndia @IYCpic.twitter.com/0ZHSYhX5c2	0.0	19

Stop words which are (a, then, and, to, his ,he,in) were removed from the tokenized list using (stopW ordRemove) function as can be seen in the above fig.
Then it was stored in (tokens_without_sw), then passed it to the Lemmatization function.

3.Lemmatization:In this step , we are going to group together the different inflected forms ,it normally aims to remove inflectional endings only and to return the base or dictionary form of a word, which is known as the lemma , Here we use the the vocabulary and morphological analysis of words.



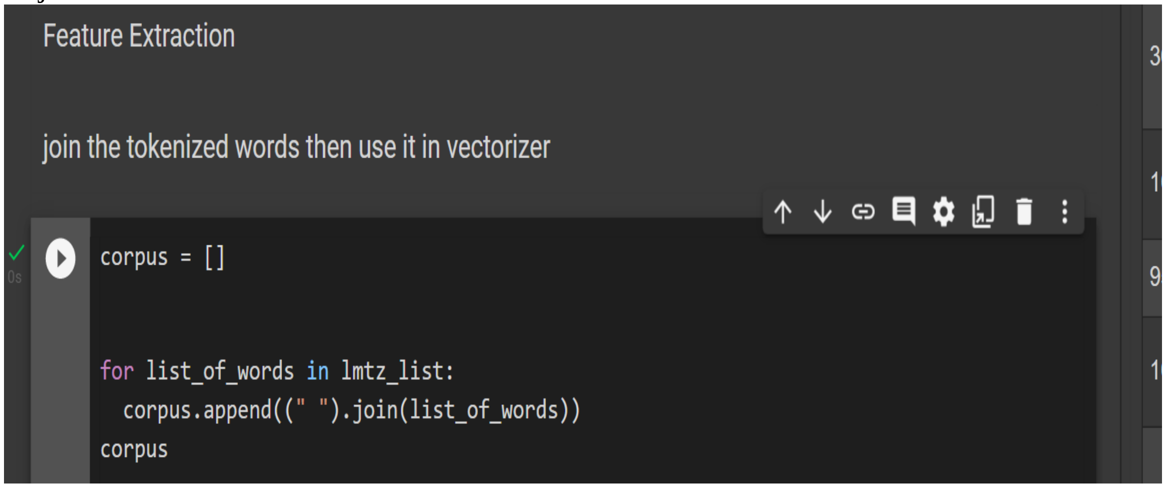
The output of the stopWordRemove is now passed to lemmatization function

```
def lemmatization(tokens_without_sw):
  lmtz_list=[]
  lemmatizer = WordNetLemmatizer()
  for words_list in tokens_without_sw:
    lmtz_words_list = []
    for word in words_list:
      lmtz_words_list.append(lemmatizer.lemmatize(word))
    lmtz_list.append(lmtz_words_list)
  return  lmtz_list
lmtz_list=lemmatization(tokens_without_sw)
print(lmtz_list) #without sw and lemmatized
```

2. Feature extraction :

Raw Data includes redundant data, and as our goal is to build a good classifier, we need to get a complete and meaningful set of the needed features. Feature Extraction aims to create and deduce new features from existing ones. Our feature extraction process consists of the following:

1. Rejoin the tokenize tweets

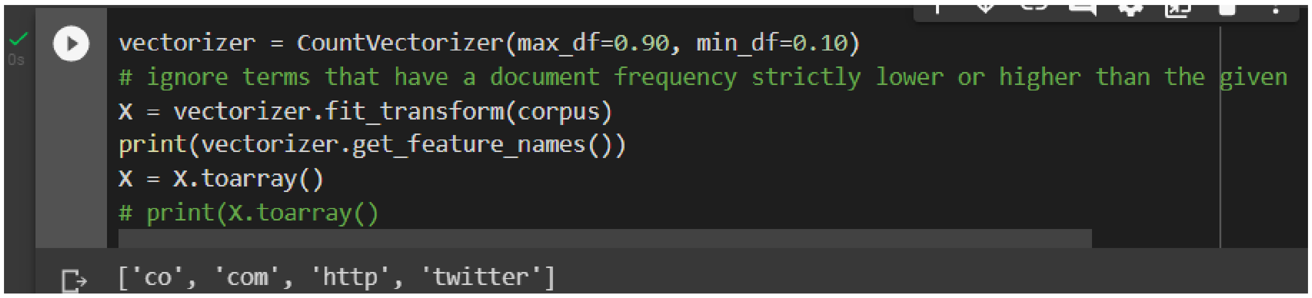


Feature Extraction

join the tokenized words then use it in vectorizer

```
corpus = []


for list_of_words in lmtz_list:
  corpus.append((" ").join(list_of_words))
corpus
```

2. *CountVectorizer* to create the bag of words array

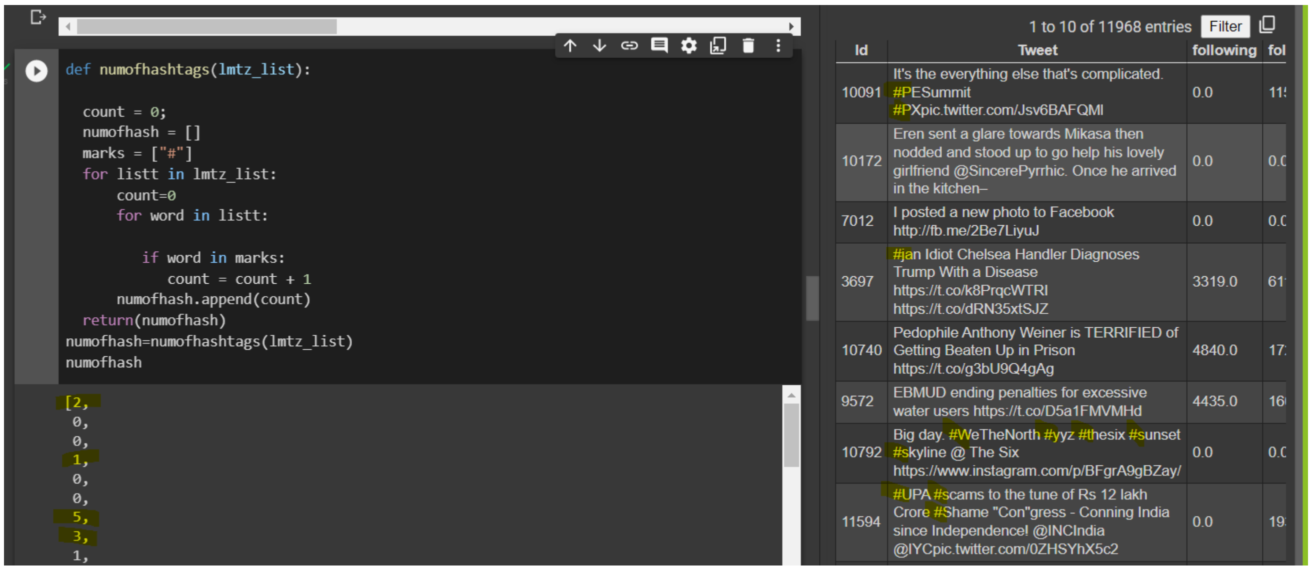


```
vectorizer = CountVectorizer(max_df=0.90, min_df=0.10)
# ignore terms that have a document frequency strictly lower or higher than the given
X = vectorizer.fit_transform(corpus)
print(vectorizer.get_feature_names())
X = X.toarray()
# print(X.toarray())
```

```
['co', 'com', 'http', 'twitter']
```

When building the vocabulary ignore terms that have a document frequency strictly higher than the given max threshold (corpus-specific stop words) and ignore terms that have a document frequency strictly lower than the given min threshold.

3. Number of Hashtags: This feature is very important , as it indicates whether or not the tweet is spam, because a lot of hashtags in the tweet , Spammers tweets mostly contain hashtags to attract the attention of legitimate users.



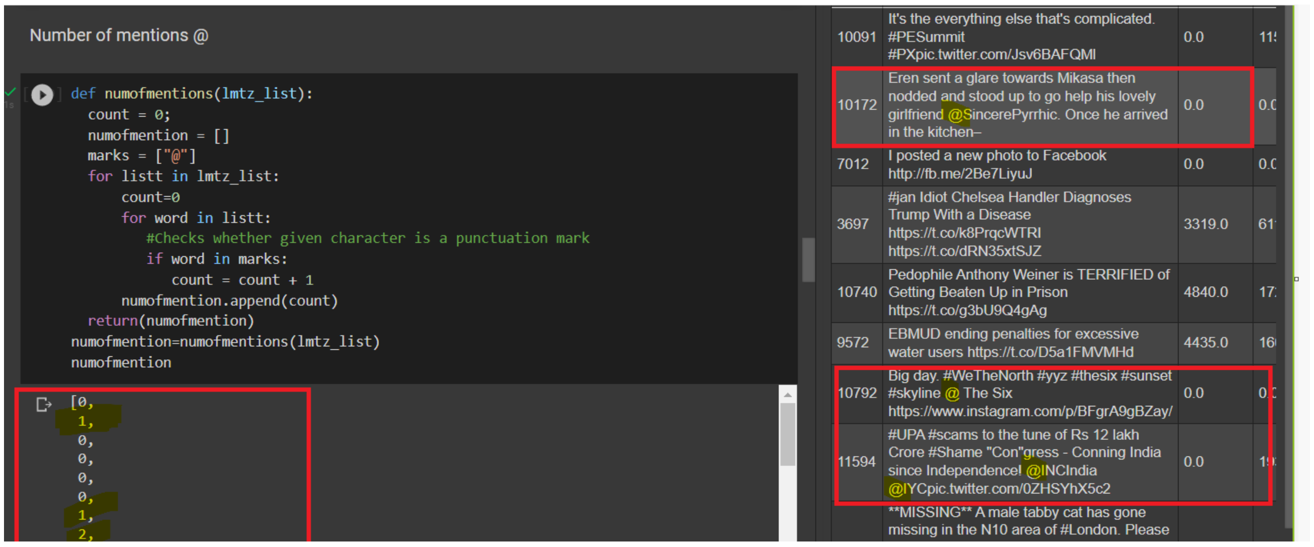
```
def numofhashtags(lmtz_list):

  count = 0;
  numofhash = []
  marks = ["#"]
  for listt in lmtz_list:
      count=0
      for word in listt:

         if word in marks:
            count = count + 1
      numofhash.append(count)
  return(numofhash)
numofhash=numofhashtags(lmtz_list)
numofhash
```

```
[2,
 0,
 0,
 1,
 0,
 0,
 5,
 3,
 1,
```

1 to 10 of 11968 entries Filter

Id	Tweet	following	fol
10091	It's the everything else that's complicated. #PESummit #PXpic.twitter.com/Jsv6BAFQMl	0.0	11!
10172	Eren sent a glare towards Mikasa then nodded and stood up to go help his lovely girlfriend @SincerePyrrhic. Once he arrived in the kitchen–	0.0	0.0
7012	I posted a new photo to Facebook http://fb.me/2Be7LiyuJ	0.0	0.0
3697	#jan Idiot Chelsea Handler Diagnoses Trump With a Disease https://t.co/k8PrqcWTRI https://t.co/dRN35xtSJZ	3319.0	61
10740	Pedophile Anthony Weiner is TERRIFIED of Getting Beaten Up in Prison https://t.co/g3bU9Q4gAg	4840.0	17.
9572	EBMUD ending penalties for excessive water users https://t.co/D5a1FMVMHd	4435.0	16
10792	Big day. #WeTheNorth #yyz #thesix #sunset #skyline @ The Six https://www.instagram.com/p/BFgrA9gBZay/	0.0	0.0
11594	#UPA #scams to the tune of Rs 12 lakh Crore #Shame "Con"gress - Conning India since Independence! @INCIndia @IYCpic.twitter.com/0ZHSYhX5c2	0.0	19

4. number of mentions : This is another important feature , it also gives a hint if this tweet is a spam or not. A lot of mentions in a tweet indicates that the tweet is a spam .



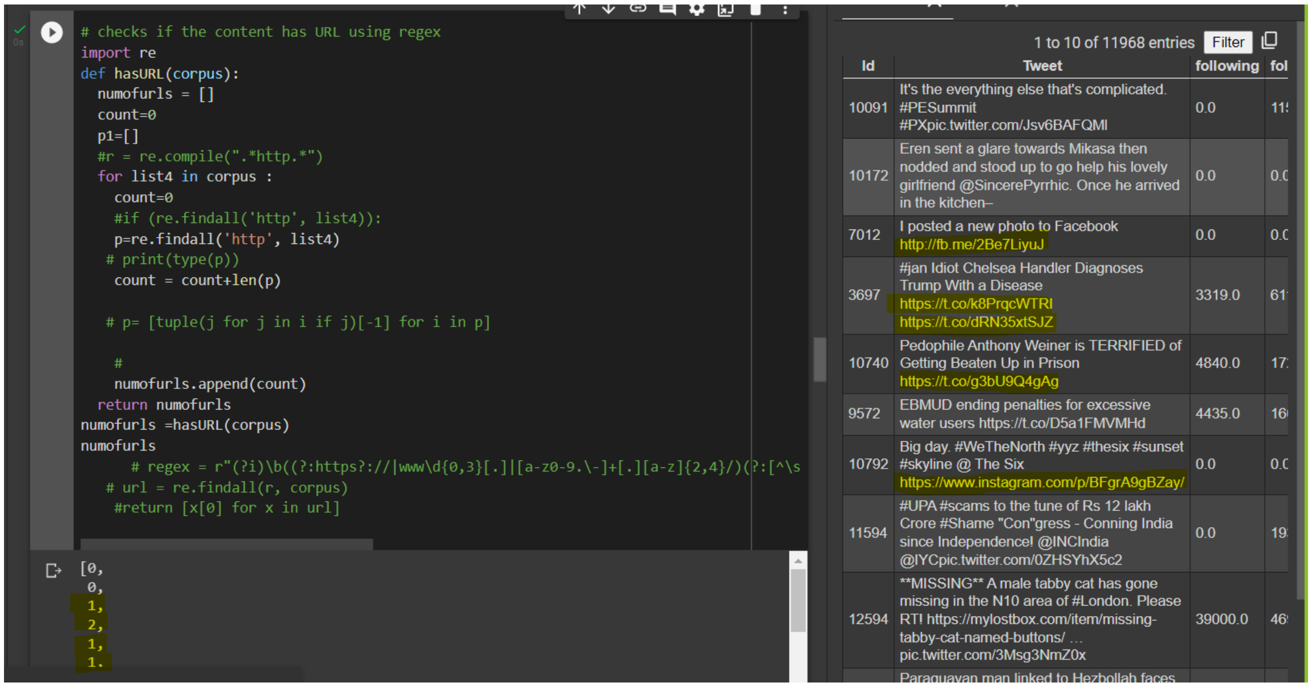
Number of mentions @

```
def numofmentions(lmtz_list):
  count = 0;
  numofmention = []
  marks = ["@"]
  for listt in lmtz_list:
      count=0
      for word in listt:
        #Checks whether given character is a punctuation mark
        if word in marks:
            count = count + 1
      numofmention.append(count)
  return(numofmention)
numofmention=numofmentions(lmtz_list)
numofmention
```

```
[0,
 1,
 0,
 0,
 0,
 0,
 1,
 2,
```

10091	It's the everything else that's complicated. #PESummit #PXpic.twitter.com/Jsv6BAFQMl	0.0	11!
10172	Eren sent a glare towards Mikasa then nodded and stood up to go help his lovely girlfriend @SincerePyrrhic. Once he arrived in the kitchen–	0.0	0.0
7012	I posted a new photo to Facebook http://fb.me/2Be7LiyuJ	0.0	0.0
3697	#jan Idiot Chelsea Handler Diagnoses Trump With a Disease https://t.co/k8PrqcWTRI https://t.co/dRN35xtSJZ	3319.0	61
10740	Pedophile Anthony Weiner is TERRIFIED of Getting Beaten Up in Prison https://t.co/g3bU9Q4gAg	4840.0	17.
9572	EBMUD ending penalties for excessive water users https://t.co/D5a1FMVMHd	4435.0	16
10792	Big day. #WeTheNorth #yyz #thesix #sunset #skyline @ The Six https://www.instagram.com/p/BFgrA9gBZay/	0.0	0.0
11594	#UPA #scams to the tune of Rs 12 lakh Crore #Shame "Con"gress - Conning India since Independence! @INCIndia @IYCpic.twitter.com/0ZHSYhX5c2	0.0	19
	MISSING A male tabby cat has gone missing in the N10 area of #London. Please		

5. if the tweet has a URL or not : Spammers' tweets mostly contain links to attract the attention of legitimate users and lead them to their malicious purposes



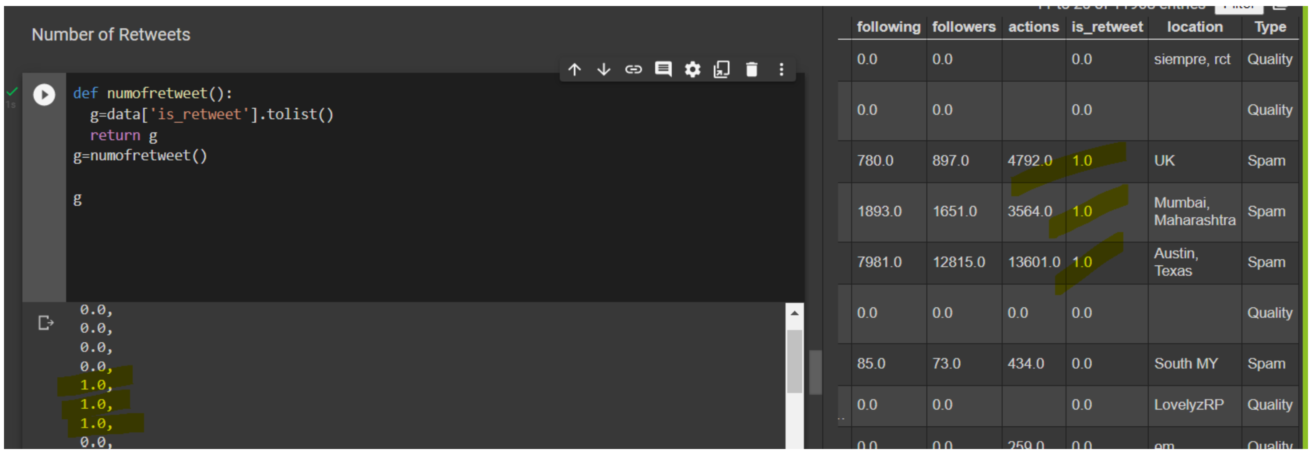
```
# checks if the content has URL using regex
import re
def hasURL(corpus):
  numofurls = []
  count=0
  p1=[]
  #r = re.compile(".*http.*")
  for list4 in corpus :
    count=0
    #if (re.findall('http', list4)):
    p=re.findall('http', list4)
   # print(type(p))
    count = count+len(p)

   # p= [tuple(j for j in i if j)[-1] for i in p]

    #
    numofurls.append(count)
  return numofurls
numofurls =hasURL(corpus)
numofurls
      # regex = r"(?i)\b((?:https?://|www\d{0,3}[.]|[a-z0-9.\-]+[.][a-z]{2,4}/)(?:[^\s
   # url = re.findall(r, corpus)
    #return [x[0] for x in url]
```

```
[0,
 0,
 1,
 2,
 1,
 1.
```

1 to 10 of 11968 entries

Id	Tweet	following	fol
10091	It's the everything else that's complicated. #PESummit #PXpic.twitter.com/Jsv6BAFQMI	0.0	11
10172	Eren sent a glare towards Mikasa then nodded and stood up to go help his lovely girlfriend @SincerePyrrhic. Once he arrived in the kitchen–	0.0	0.0
7012	I posted a new photo to Facebook http://fb.me/2Be7LiyuJ	0.0	0.0
3697	#jan Idiot Chelsea Handler Diagnoses Trump With a Disease https://t.co/k8PrqcWTRI https://t.co/dRN35xtSJZ	3319.0	61
10740	Pedophile Anthony Weiner is TERRIFIED of Getting Beaten Up in Prison https://t.co/g3bU9Q4gAg	4840.0	17
9572	EBMUD ending penalties for excessive water users https://t.co/D5a1FMVMHd	4435.0	16
10792	Big day. #WeTheNorth #yyz #thesix #sunset #skyline @ The Six https://www.instagram.com/p/BFgrA9gBZay/	0.0	0.0
11594	#UPA #scams to the tune of Rs 12 lakh Crore #Shame "Con"gress - Conning India since Independence! @INCIndia @IYCpic.twitter.com/0ZHSYhX5c2	0.0	19
12594	**MISSING** A male tabby cat has gone missing in the N10 area of #London. Please RT! https://mylostbox.com/item/missing-tabby-cat-named-buttons/ … pic.twitter.com/3Msg3NmZ0x	39000.0	46
	Paraguayan man linked to Hezbollah faces		

6. number of retweets :
   Retweets- Retweets are the replies to any tweet using @RT symbol and spammers use maximum @RT in their tweets.



Number of Retweets

```
def numofretweet():
  g=data['is_retweet'].tolist()
  return g
g=numofretweet()

g
```

```
 0.0,
 0.0,
 0.0,
 0.0,
 1.0,
 1.0,
 1.0,
 0.0,
```

following	followers	actions	is_retweet	location	Type
0.0	0.0		0.0	siempre, rct	Quality
0.0	0.0		0.0		Quality
780.0	897.0	4792.0	1.0	UK	Spam
1893.0	1651.0	3564.0	1.0	Mumbai, Maharashtra	Spam
7981.0	12815.0	13601.0	1.0	Austin, Texas	Spam
0.0	0.0	0.0	0.0		Quality
85.0	73.0	434.0	0.0	South MY	Spam
0.0	0.0		0.0	LovelyzRP	Quality
0.0	0.0	259.0	0.0	em	Quality

7. the ratio of the number of following and number of followers:

This feature is very important , as it indicates whether or not the tweet is spam, because Compared to legitimate users, spammers are more likely to follow too many legitimate accounts to attract attention.

Since spammers are not followed by legitimate users, spammers have fewer followers than legitimate users. Also, their tweets are less likely to receive likes and retweets than those from legitimate users.

The ratio (Followers/Following+Followers) was considered. since:

1. Numbers of followers-spammers have less number of

followers.

2. Numbers of followings-Spammers tend to follow a

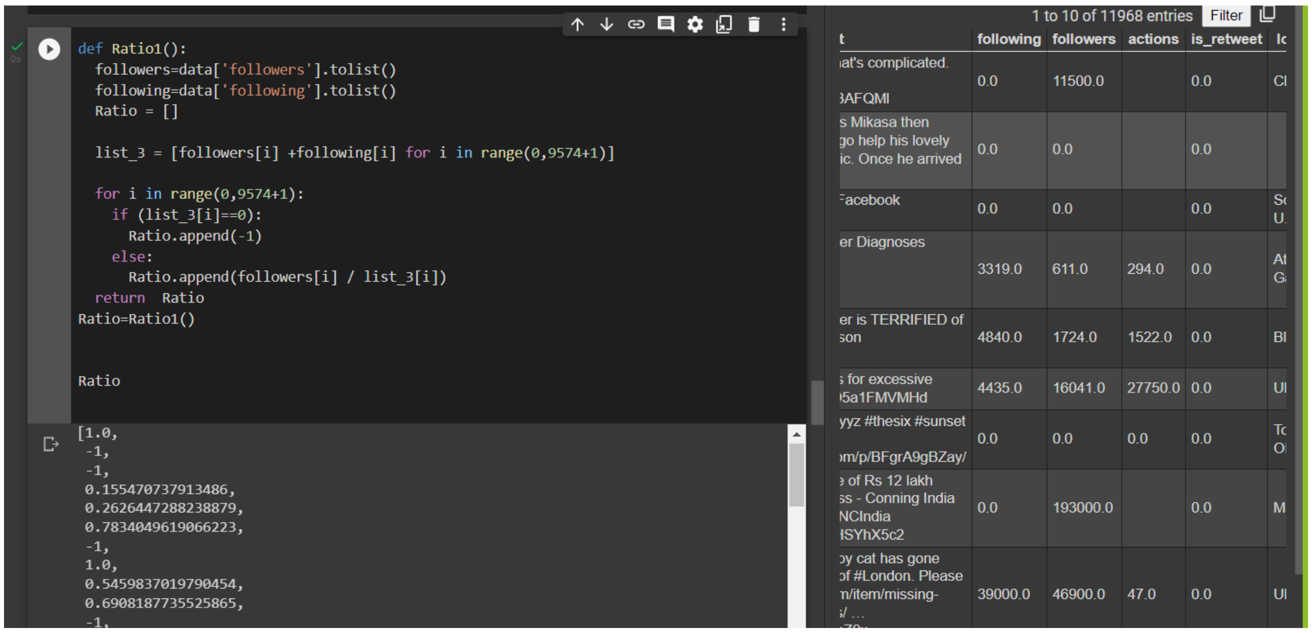
large number of users.

3. Followers/Following Ratio- this ratio is less than 1

for spammers.

4. Reputation is defined as the ratio of followers to the

sum of followers



```
def Ratio1():
  followers=data['followers'].tolist()
  following=data['following'].tolist()
  Ratio = []

  list_3 = [followers[i] +following[i] for i in range(0,9574+1)]

  for i in range(0,9574+1):
    if (list_3[i]==0):
      Ratio.append(-1)
    else:
      Ratio.append(followers[i] / list_3[i])
  return  Ratio
Ratio=Ratio1()


Ratio
```

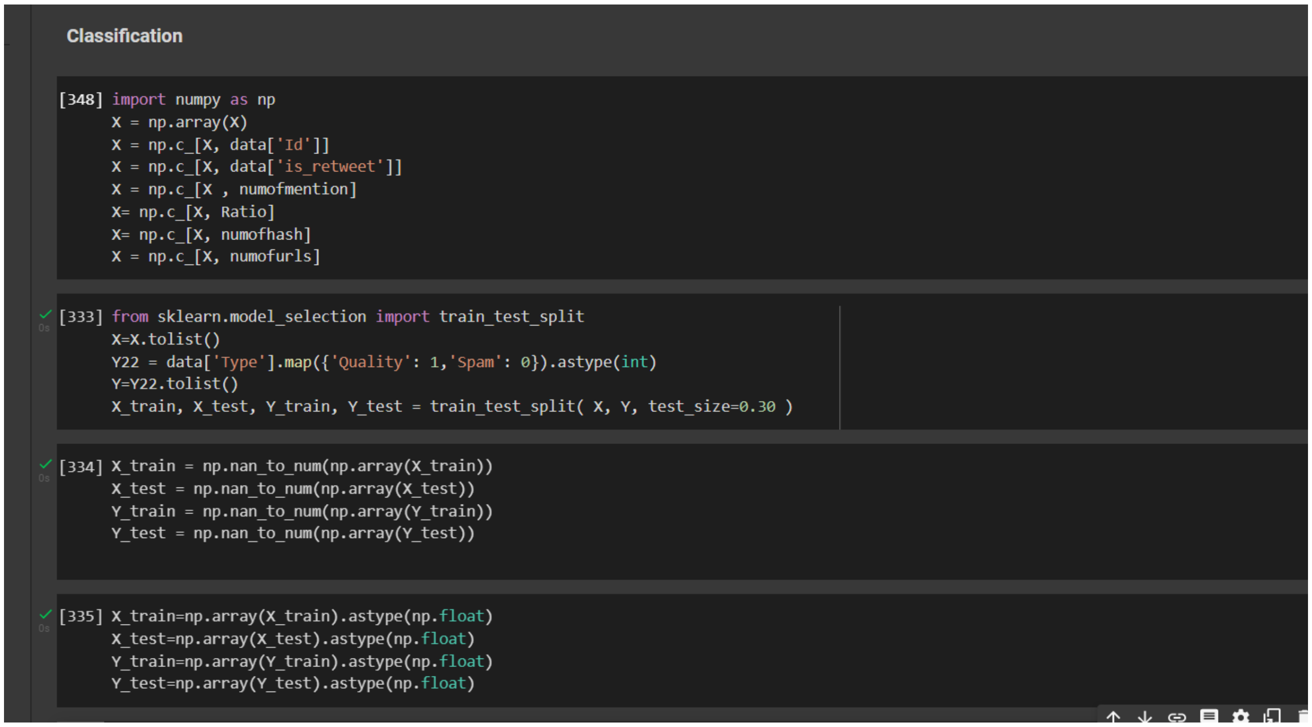
```
[1.0,
 -1,
 -1,
 0.155470737913486,
 0.2626447288238879,
 0.7834049619066223,
 -1,
 1.0,
 0.5459837019790454,
 0.6908187735525865,
 -1,
```

1 to 10 of 11968 entries

t	following	followers	actions	is_retweet	lc
ıat's complicated. 3AFQMI	0.0	11500.0		0.0	Cl
s Mikasa then go help his lovely ic. Once he arrived	0.0	0.0		0.0	
Facebook	0.0	0.0		0.0	S U.
er Diagnoses	3319.0	611.0	294.0	0.0	A G
er is TERRIFIED of son	4840.0	1724.0	1522.0	0.0	Bl
s for excessive)5a1FMVMHd	4435.0	16041.0	27750.0	0.0	Ul
yyz #thesix #sunset om/p/BFgrA9gBZay/	0.0	0.0	0.0	0.0	T O
e of Rs 12 lakh ss - Conning India NCIndia ISYhX5c2	0.0	193000.0		0.0	M
oy cat has gone of #London. Please n/item/missing- s/ …	39000.0	46900.0	47.0	0.0	Ul

The first account for example has 11500 followers while they don't follow anyone.
so their following Ratio= 11500/(11500+0) =1.

4. Classification :
These steps were applied before applying classifiers:



Classification

```
[348] import numpy as np
      X = np.array(X)
      X = np.c_[X, data['Id']]
      X = np.c_[X, data['is_retweet']]
      X = np.c_[X , numofmention]
      X= np.c_[X, Ratio]
      X= np.c_[X, numofhash]
      X = np.c_[X, numofurls]
```

```
[333] from sklearn.model_selection import train_test_split
      X=X.tolist()
      Y22 = data['Type'].map({'Quality': 1,'Spam': 0}).astype(int)
      Y=Y22.tolist()
      X_train, X_test, Y_train, Y_test = train_test_split( X, Y, test_size=0.30 )
```

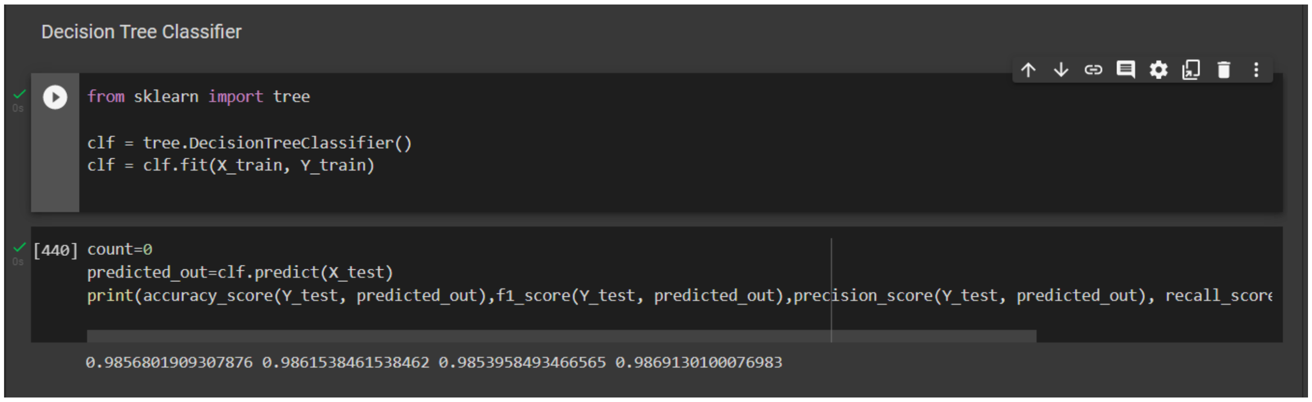
```
[334] X_train = np.nan_to_num(np.array(X_train))
      X_test = np.nan_to_num(np.array(X_test))
      Y_train = np.nan_to_num(np.array(Y_train))
      Y_test = np.nan_to_num(np.array(Y_test))
```

```
[335] X_train=np.array(X_train).astype(np.float)
      X_test=np.array(X_test).astype(np.float)
      Y_train=np.array(Y_train).astype(np.float)
      Y_test=np.array(Y_test).astype(np.float)
```

np.c Translates slice objects to concatenation along the second axis.
Since the Classifier's output classes are either 1 or 0 so
Quality was given a value of 1 and Spam was given a value of 0. Then, The dataset was divided dataset into 70% training 30% testing finally, NaN data was handled.now the data is ready to get passed to classifiers.

Result and Testing Classifiers:

1. Decision tree Before Balancing

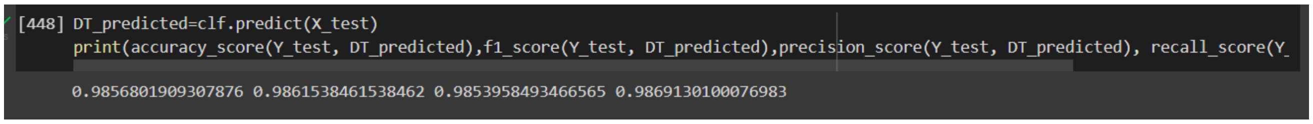


Decision Tree Classifier

```
from sklearn import tree

clf = tree.DecisionTreeClassifier()
clf = clf.fit(X_train, Y_train)
```

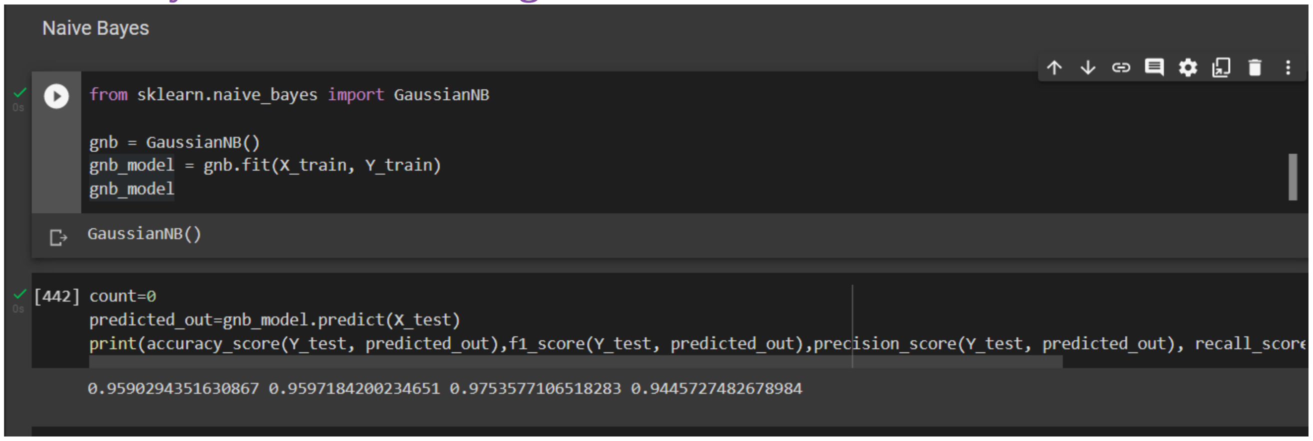
```
[440] count=0
      predicted_out=clf.predict(X_test)
      print(accuracy_score(Y_test, predicted_out),f1_score(Y_test, predicted_out),precision_score(Y_test, predicted_out), recall_score

      0.9856801909307876 0.9861538461538462 0.9853958493466565 0.9869130100076983
```

After Balancing



```
[448] DT_predicted=clf.predict(X_test)
      print(accuracy_score(Y_test, DT_predicted),f1_score(Y_test, DT_predicted),precision_score(Y_test, DT_predicted), recall_score(Y_

      0.9856801909307876 0.9861538461538462 0.9853958493466565 0.9869130100076983
```

Naive Bayes before balancing :

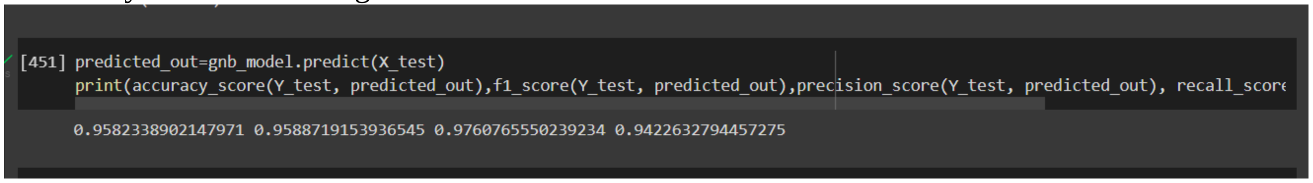


Naive Bayes

```
from sklearn.naive_bayes import GaussianNB

gnb = GaussianNB()
gnb_model = gnb.fit(X_train, Y_train)
gnb_model

GaussianNB()
```

```
[442] count=0
      predicted_out=gnb_model.predict(X_test)
      print(accuracy_score(Y_test, predicted_out),f1_score(Y_test, predicted_out),precision_score(Y_test, predicted_out), recall_score

      0.9590294351630867 0.9597184200234651 0.9753577106518283 0.9445727482678984
```

Naive Bayes after Balancing:



```
[451] predicted_out=gnb_model.predict(X_test)
      print(accuracy_score(Y_test, predicted_out),f1_score(Y_test, predicted_out),precision_score(Y_test, predicted_out), recall_score

      0.9582338902147971 0.9588719153936545 0.9760765550239234 0.9422632794457275
```

Conclusion

Reverences

[1] Additive White Gaussian Noise (AWGN) - Wireless Pi accessed on 19-12-2021
[2] What is AWGN - Noisecom accessed on 19-12-2021
[3] What is signal-to-noise ratio and how is it measured? (techtarget.com) accessed on 19-12-2021

Appendix