PalArch's Journal of Archaeology of Egypt / Egyptology

Spam Classification Using Random Forest Algorithm

[1] *Dr Rashmi Agrawal*

[1] Professor, Manav Rachna International Institute of Research and Studies

Email: [1] drrashmiagrawal78@gmail.com

Dr Rashmi Agrawal: Spam Classification Using Random Forest Algorithm -- Palarch's Journal Of Archaeology Of Egypt/Egyptology 17(6). ISSN 1567-214x

Keywords: Text Mining, Stemming, Text Pre-processing, Classification Random Forest.

ABSTRACT

Natural language Processing (NLP) is an important component of machine learning which have many applications like sentiment analysis, social network analysis, chatbots, text classification, rating of reviews and many more. Researchers are constantly handling the largely available text data now a days and utilising it to apply text mining so that the useful and relevant information can be withdrawn.

In this paper, we aim to apply the Random forest technique for text classification. In the first part we will build the word cloud and in the second section we will apply machine learning technique on the spam classification dataset.

1. Introduction

Text Mining/ Text Processing or Text analytics are the common terms used for handling text data. We can define text mining as the process of retrieving the information from the text data. For a data scientist, handling text data is the need of the hour as reviewing data from the website, finding sentiment analysis from twitter, understanding behaviour of the people , these all are the prime concern areas for data analytics. As analysing text data in comparison of numerical data is too complex but using the machine learning techniques this task is an interesting area for the data scientist in the current scenario. Really text data is messy if viewed as in the raw form which is stored in various formats but after applying the text mining techniques it gives out the results which can give a boost to various businesses. For example – the companies may be interested to know the views of their customers which the customers

are posting in the compnay's blog/ website/ Facebook page. It is nearly impossible to evaluate it manually but by applying the machine learning technique the business firms can get the enormous text data/views of their customers.

Text mining permits to considerate the text better than anything else. Text mining supports to discover patterns and relationships that exist in a hefty text. Text mining makes use of machine learning algorithms to analyse the information in a well presented and human readable manner. At present we have R and Python in which the text mining can be applied using the machine learning. To use text mining, various libraries are required to install.

2. Literature Review

Text mining applications have practiced marvellous developments because of web 2.0 and social networking applications (Aggarwal, C. C., & Zhai, C 2012). A detail Survey of text mining was performed by Berry in 2004. Survey on text mining facilities in R was also presented by Meyer et al (2008) and explained that how typical application tasks can be carried out using their framework. They presented techniques for count-based analysis methods, text classification, text clustering, and string kernels. Since this period researchers are actively involved with various applications using text mining. Text classification is one of the important applications of text mining where many researchers have placed their energies.

Joulin et al (2016) explored a modest and effectual baseline for text classification. Their researches proved that fast text classifier fastText is often on par with deep learning classifiers in terms of accuracy, and many orders of magnitude faster for training and evaluation.

Inductive transfer learning has significantly wedged computer vision. An efficient method was proposed by Howard and Ruder (2018) for Universal Language Model Fine-tuning which is an operative transfer learning technique that can be useful to any task in NLP.

Kowsari et al (2019) provided a brief overview of text classification algorithms. This overview covers different text feature extractions, dimensionality reduction methods, existing algorithms and techniques, and evaluations methods.

3. Creating a Word Cloud

The current section describes how to create wonderful word clouds with R. NLP, tm and RColorBrewer libraries must be installed in R before applying the text mining.

We downloaded the news dataset from github and stored in csv format.

A snapshot of the dataset-

1. '--Longevity Increase Seen Around the World: WHO\',health'
2. '--Chikungunya spreading, mosquito-borne virus is being viewed as a national threat\',health'
3. '--Family rejects plan to solve medical custody dispute\',health'
4. '--FDA proposes program for faster approval of medical devices\',health'

5. '--Red wine, chocolate, grapes may not improve your health\',health'
6. '--USM, hospitals form network to study West Nile virus\',health'

Using the paste function we collapsed the data and converted into a chunk of data as shown below-

--Longevity Increase Seen Around the World: WHO\',health--Chikungunya spreading, mosquito-borne virus is being viewed as a national threat\',health--Family rejects plan to solve medical custody dispute\',health--FDA proposes program for faster approval of medical devices\',health--Red wine, chocolate, grapes may not improve your health\',health--USM, hospitals form network to study West Nile virus\',health--Afrezza gets FDA approval to improve glycemic control in adults with diabetes ...\',health--Study: E-Cigarettes Can Help Kick Smoking Habit\',health--Many Women Don\'t Need Yearly Pelvic Exams,

3.1 Cleaning the text

To perform the cleaning of data, we performed the following steps-

1) Convert whole data into the lowercase using the tolower() function.
2) Remove the punctuations and digits which can be easily done using the gsub function.
3) Remove the stopwords and other unnecessary words from our text we have used the function removewords()
4) Remove single letters, by gsub() function.
5) The last step of data cleaning is to remove the remove white spaces using stripWhitespace() function.

3.2 Counting Frequency of Words

So far we have cleaned our data and next step is to count the frequency of words. As we already made the chunk of data by using paste function so we need to split each word in order to count the frequency of words. The snapshot of result obtained is shown in figure 1-

" 'longevity' 'increase' 'seen' 'around' 'world' 'health' 'chikungunya' 'spreading' 'mosquito' 'borne' 'virus' 'viewed' 'national' 'threat' 'health' 'family' 'rejects' 'plan' 'solve' 'medical' 'custody' 'dispute' 'health' 'fda' 'proposes' 'program' 'faster' 'approval' 'medical' 'devices' 'health' 'red' 'wine' 'chocolate' 'grapes' 'may' 'improve' 'health' 'health' 'usm' 'hospitals' 'form' 'network' 'study' 'west' 'nile' 'virus' 'health' 'afrezza' 'gets' 'fda' 'approval' 'improve' 'glycemic' 'control' 'adults' 'diabetes' 'health' 'study' 'cigarettes' 'can' 'help' 'kick' 'smoking' 'habit' 'health' 'many' 'women' 'don' 'need' 'yearly' 'pelvic' 'exams' 'doctors' 'group' 'says' 'health' 'university' 'wisconsin' 'researchers' 'seek' 'another' 'way' 'combat' 'deer' 'ticks' 'health' 'cialis' 'makers' 'want' 'drug' 'available' 'counter' 'health' 'dramatic' 'increase' 'north' 'east' 'skin' 'cancer' 'rates' 'health' 'soda' 'industry' 'study' 'says' 'drink' 'diet' 'soda'

Fig 1- Data after cleaning

We created the word frequency table to have the count of each word. Based on this word frequency table, the word cloud of the most frequent words can be plotted using the wordcloud() function shown in fig 2..

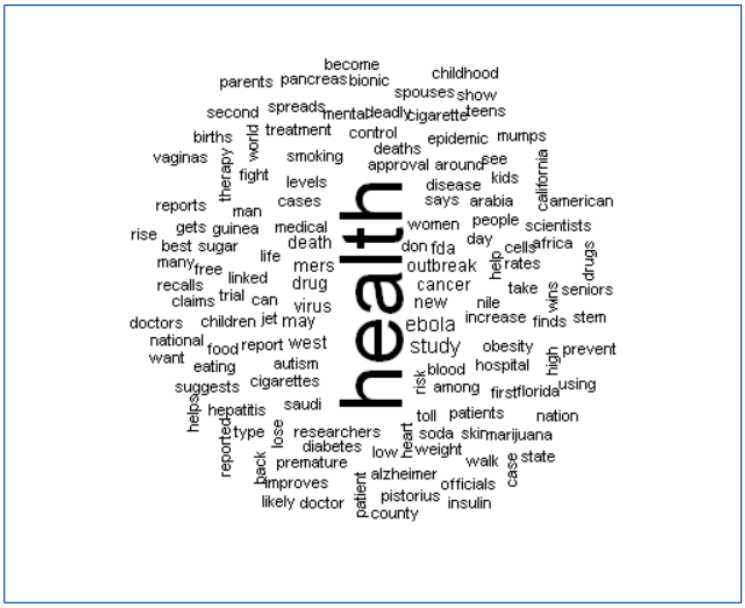

Fig 2- Word Cloud

4. Machine Learning Techniques on Text data

To apply the machine learning technique on the text data we used the spam text classification data downloaded from UCI machine learning repository. Snapshot of this data is shown in figure 3 below-

```
head(t1)
```

category	text
ham	Go until jurong point, crazy.. Available only in bugis n great world la e buffet... Cine there got amore wat...
ham	Ok lar... Joking wif u oni...
spam	Free entry in 2 a wkly comp to win FA Cup final tkts 21st May 2005. Text FA to 87121 to receive entry question(std txt rate)T&C's apply 08452810075over18's
ham	U dun say so early hor... U c already then say...
ham	Nah I don't think he goes to usf, he lives around here though
spam	FreeMsg Hey there darling it's been 3 week's now and no word back! I'd like some fun you up for it still? Tb ok! XxX std chgs to send, £1.50 to rcv

Fig 3- Spam Classification Data

It classifies the data as spam or ham based on its text contents. This dataset have only two columns category and text and contains 5572 instances. Before applying the random forest algorithm to perform the text classification we pre-processed the dataset with the following steps-

i) Creating the Text Corpus

For pre-processing the variable containing text need to be converted into a corpus. Collection of documents is known as a corpus. To create a corpus out of given variable corpus function is used as shown below-

corpus = Corpus(VectorSource(t1$text))

head(corpus)

<<SimpleCorpus>> Metadata: corpus specific: 1, document level (indexed): 0 Content: documents: 6

ii) Converting corpus into lower case

To train our model using the machine learning, it is necessary that the model should take “World”, “world” or “WORLD” all these words as same. ...'

iii) Removing punctuation and stopwords

iv) Stemming

Stemming is the process of generating the base or root word morphologically. For example the words friendship and friendly can be reduced to its base word friend. Although stemming is an important concept before training the model but sometimes it also produces some errors. There are mainly two types of errors in stemming-

-Under-stemming

-Over-stemming

When two words are stemmed to same root but belongs to different stems, case of over-stemming occurs. It is also considered as the false positives. For example- the words like university and universal are stemmed they generate the root "universe" which gives rise to false positive.

The opposite of the above is under-stemming. In this two words are stemmed to same root that are not of different stem. It is also considered as the false negative. For example the words datum and data are stemmed to datu and dat respectively which gives rise to under-stemming.

There are various famous algorithms to handle the stemming. Some of the algorithms used are-

a) **Porter's Stemmer algorithm (**Kraaij et al 1995)
b) **Lovins Stemmer (Porter 2005)**
c) **Dawson Stemmer (Paice 1990)**
d) **Krovetz Stemmer (Krovetz 2000)**
e) **Xerox Stemmer (Utzey et al 1997)**
f) **N-Gram Stemmer (Mayfoeld et al 2003)**

v) Creating Document Term Matrix

The tm package provides a fuction called DocumentTermMatrix() which generates a matrix. In this matrix, rows and columns represents the words of the documents and the value shows the frequency of that word. When we create the document term matrix many times the problem of sparsity (containing many zeros in the cells) occurs in the matrix. Therefore, in many machine learning models this sparsity of the matrix is removed. In R it can be easily done using removeSparseTerms().

In order to perform the predictive modelling of the data, it is desirable to convert this matrix into a data frame which is a widely acceptable and easy handling data format.

got	great	point	wat	world	lar	appli	final	free	may	...	princess	mani	bore	shall	probabl	dad	mate	await	wan	category
1	1	1	1	1	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	ham
0	0	0	0	0	1	0	0	0	0	...	0	0	0	0	0	0	0	0	0	ham
0	0	0	0	0	0	1	1	1	1	...	0	0	0	0	0	0	0	0	0	spam
0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	ham
0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	ham
0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	spam

Fig 4-Converting matrix into data frame

We find that nearly 87% of the observation belongs to category ham and only 13% of the observation belongs to category spam as shown in figure 4.

ham spam

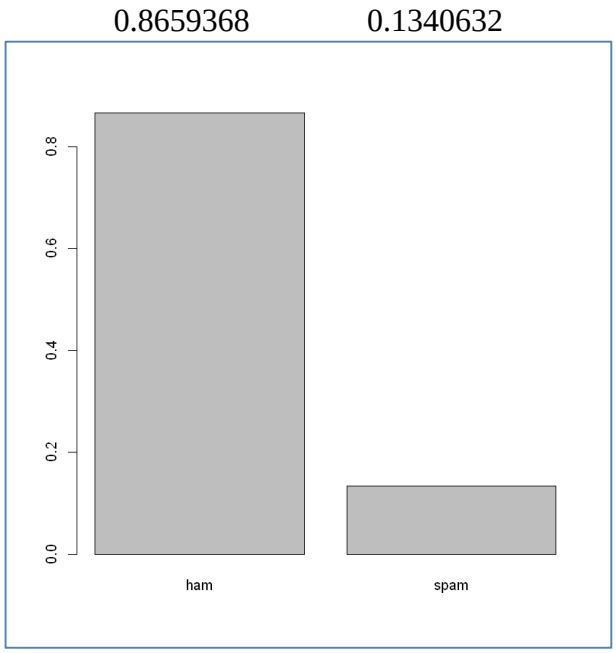

Fig 4- categorization

So now it becomes the baseline accuracy of the predictive model.

vi) Splitting Data into Training and Testing-

To train the model we split the data into two parts -training data and testing data. We kept the split percentage as 70% for training data and 30% for testing data and we applied the machine learning algorithm Random Forest.

vii) Applying Machine Learning Algorithm-Random Forest

This algorithm many times outperform over other classification algorithm such as decision tree as this algorithm works on the basis of ensembling. To implement this model in R, we need to use the randomforest library.

Confusion Matrix Generated is shown in table 1 below-

Table 1- Confusion Matrix

	Ham	Spam
Ham	1440	7
Spam	36	188

It shows the model provides 97.4 % accuracy

5. Evaluation of the Predictive Model

Here, we set the baseline accuracy as 86 percent. It is clearly visible that the Random Forest model is expediently thumping this baseline model by accomplishing the accuracy of 97 percent. The four fold plot of the confusion matrix is shown below-

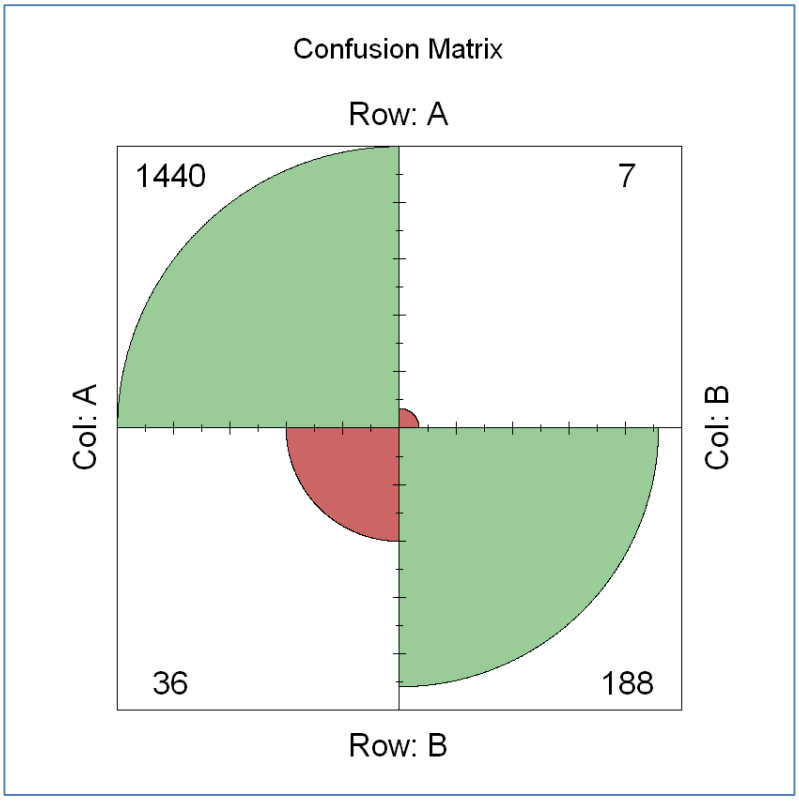

Figure 5 : Four Fold Plot of Confusion Matrix using Random Forest Algorithm

6. Conclusion and Future Scope-

Text classification is the most fundamental task in the applications of natural language processing. In this paper we performed text classification using Random forest algorithm on Spam classification data set. Our results show that the greater accuracy can be achieved using the random forest algorithm. Text classification and text analytics is an active area of research in which it is required to develop new algorithms to get better accuracy and also there are various challenges in text pre-processing hence efforts are also required to develop simple algorithms to handle text pre-processing as well.

References-

Aggarwal, C. C., & Zhai, C. (Eds.). (2012). Mining text data. Springer Science & Business Media.

Berry, M. W. (2004). Survey of text mining. Computing Reviews, 45(9), 548.

Meyer, D., Hornik, K., & Feinerer, I. (2008). Text mining infrastructure in R. Journal of statistical software, 25(5), 1-54.

Joulin, A., Grave, E., Bojanowski, P., & Mikolov, T. (2016). Bag of tricks for efficient text classification. arXiv preprint arXiv:1607.01759.

Howard, J., & Ruder, S. (2018). Universal language model fine-tuning for text classification. arXiv preprint arXiv:1801.06146.

Kowsari, K., Jafari Meimandi, K., Heidarysafa, M., Mendu, S., Barnes, L., & Brown, D. (2019). Text classification algorithms: A survey. Information, 10(4), 150.

Kraaij, Wessel, and Renée Pohlmann. "Evaluation of a Dutch stemming algorithm." The New Review of Document and Text Management 1 (1995): 25-43.

Porter, Martin F. "Lovins revisited." Charting a New Course: Natural Language Processing and Information Retrieval. Springer, Dordrecht, 2005. 39-68.

Paice, Chris D. "Another stemmer." ACM Sigir Forum. Vol. 24. No. 3. New York, NY, USA: ACM, 1990.

Krovetz, Robert. "Viewing morphology as an inference process." Artificial intelligence 118.1-2 (2000): 277-294.

Utzey Jan, Eric Gaussier Hinrich Sch, and O. Pedersenz. "Xerox TREC-5 site report: routing, filtering, NLP, and Spanish tracks." (1997).

Mayfield, James, and Paul McNamee. "Single n-gram stemming." Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval. 2003.