Classification of Covid-19 Vaccine Opinion and Detection of Symptom-Reporting on Twitter Using Neural Networks

Vishal Nair[1]

[1]St. Stephen's College,Delhi University,New Delhi,India

Abstract

This paper describes my work for the Information Retrieval from Microblogs during Disasters.This track is divided into two sub-tasks. Task 1 is to build an effective classifier for 3-class classification on tweets with respect to the stance reflected towards COVID-19 vaccines.Task 2 is to devise an effective classifier for 4-class classification on tweets that can detect tweets that report someone experiencing COVID-19 symptoms.This paper proposes a classification method based on MLP classifier model.The evaluation shows the performance of our approach, which achieved 0.304 on F-Score in Task 1 and 0.239 on F-Score in Task 2.

Keywords

Information Retrieval, Microblogs during Disasters, tweets, F-score

1. Introduction

In the unfortunate time of Covid-19 pandemic a society-scale vaccination is the only long term remedy.Covid-19 vaccination is being held everywhere around the globe with full force.But a number of people are skeptical about the usage of vaccines owing to various reasons.In such cases it is really important to understand public sentiments towards vaccines, and social media can be used to gain a lot of data quickly about people's stance on vaccines. Apart from this it is very crucial for us to understand and identify people who are suffering with Covid-19 symptoms.Microblogging sites such as Twitter have become an important sources of situational information during disaster events.We will use symptom-reporting tweets for this purpose.Thus Sentiment analysis of tweets regarding people's opinion on Covid-19 vaccine using ML models will give us a better picture about people's view on vaccination and also we can understand which tweets actually inform if someone is experiencing Covid-19 symptoms. FIRE 2022 microblog track provided the training and test data (tweets) for both the tasks.
The two tasks are explained below-

- **Data-set-I**
  To perform 3-class classification with respect to the stance reflected towards COVID-19 vaccines. The 3 classes are described below:

Forum for Information Retrieval Evaluation, December 9-13, 2022, India
v1292002@gmail.com (V. Nair)

  - AntiVax - the tweet indicates hesitancy (of the user who posted the tweet) towards the use of vaccines.
  - ProVax - the tweet supports / promotes the use of vaccines.
  - Neutral - the tweet does not have any discernible sentiment expressed towards vaccines or is not related to vaccines

- **Data-set-II**
  To perform 4-class classification on tweets that can detect tweets that report someone experiencing COVID-19 symptoms. The 4 classes are described below:
  - Primary Reporting - The user (who posted the tweet) is reporting symptoms of himself/herself.
  - Secondary Reporting - The user is reporting symptoms of some friend / relative / neighbour / someone they met.
  - Third-party Reporting - The user is reporting symptoms of some celebrity / third-party person.
  - Non-Reporting - The user is not reporting anyone experiencing COVID-19 symptoms, but talking about symptom-words in some other context. This class includes tweets that only give general information about COVID-19 symptoms, without specifically reporting about a person experiencing such symptoms.

2. Methodology

The following methodology is used for the classification of the two data-sets-

2.1. Pre-processing of tweets

The initial step is to do pre-processing of the tweets in the training data-set to make them suitable for classification.For this a function is defined in python that removes or filters certain unnecessary terms from the tweets.This function is then further used to remove twitter handles (@user),special characters,stop words, numbers, punctuations and short words from the tweets.

2.2. Tokenization

Tokenization is the process of tokenizing or splitting a string, text into a list of tokens. One can think of token as parts like a word is a token in a sentence, and a sentence is a token in a paragraph.Tweets are tokenized simply using the split function in python

2.3. Stemming

It is the process of reducing the word to its word stem that affixes to suffixes and prefixes or to roots of words known as a lemma. In simple words stemming is reducing a word to its base word or stem in such a way that the words of similar kind lie under a common stem. Stemming is important in natural language processing(NLP). For example, the word “program” can also take the form of “programmed” or “programming.” When tokenized, all three of those words result in different tokens.Stemming is an option to handle that at indexing time.

Lancaster stemming is applied to the dataset using nltk library in python.The Lancaster stemmers are more aggressive and dynamic compared to the other stemmers like snowball and Porter stemmer. The Lancaster stemmer is really faster, but the algorithm is really confusing when dealing with small words. But they are not as efficient as Snowball Stemmers. The Lancaster stemmers save the rules externally and basically uses an iterative algorithm.

2.4. Feature Extraction

After Stemming tf-idf vectorization of the tweets is done to make them suitable to be applied on MLP classifier model.
Tf-idf Vectorization
Bag of words (BoW) converts the text into a feature vector by counting the occurrence of words in a document. It is not considering the importance of words. Term frequency -Inverse document frequency (TFIDF) is based on the Bag of Words (BoW) model, which contains insights about the less relevant and more relevant words in a document. The importance of a word in the text is of great significance in information retrieval. Example- If you search something on the search engine, with the help of TFIDF values, search engines can give us the most relevant documents related to our search.It is a measure of the frequency of a word (w) in a document (d).

Term Frequency is defined as the ratio of a word's occurrence in a document to the total number of words in a document. The denominator term in the formula is to normalize since all the corpus documents are of different lengths.

$$TF(w,d) = \frac{occurrences\ of\ w\ in\ d}{total\ no.\ of\ words\ in\ d} \tag{1}$$

Inverse Document Frequency (IDF) It is the measure of the importance of a word. Term frequency (TF) does not consider the importance of words. Some words such as' of', 'and', etc. can be most frequently present but are of little significance. IDF provides weightage to each word based on its frequency in the corpus D. IDF of a word (w) is defined as

$$IDF(w,D) = ln(\frac{Total\ no.\ of\ documents\ in\ corpus\ D}{No.\ of\ documents\ containing\ w}) \tag{2}$$

Term Frequency - Inverse Document Frequency (TF-IDF)-
It is the product of TF and IDF.TF-IDF gives more weightage to the word that is rare in the corpus (all the documents).TF-IDF provides more importance to the word that is more frequent in the document.
Since TF values lie between 0 and 1, not using ln can result in high IDF for some words, thereby dominating the TF-IDF. We don't want that, and therefore, we use ln so that IDF should not completely dominate the TF-IDF.
Xtrain is obtained after tf-idf vectorization and Ytrain is the labels column of the training dataset.

2.5. Applying Neural Network Model

Neural Network model was selected for classification because Neural networks are complex models, which try to mimic the way the human brain develops classification rules. A neural net

consists of many different layers of neurons, with each layer receiving inputs from previous layers, and passing outputs to further layers. The way each layer output becomes the input for the next layer depends on the weight given to that specific link, which depends on the cost function, and the optimizer. The neural net iterates for a predetermined number of iterations, called epochs. After each epoch, the cost function is analyzed to see where the model could be improved.Also we can change the size of the hidden layers and see which combination gives the best results. MLP Classifier implements a multi-layer perceptron (MLP) algorithm that trains using Backpropagation.MLP trains on two arrays: array X of size (nsamples, nfeatures), which holds the training samples represented as floating point feature vectors; and array y of size (nsamples,), which holds the target values (class labels) for the training samples.

2.6. Working with the test Dataset

After training our model using the training dataset we can now apply the same pre-processing techniques along with tokenization and Stemming on the test dataset.Feature Extraction is applied on test dataset in the same way.Now the test dataset is in the suitable required format and can be used to to predict or classify the tweets using our model.

3. Classification of Tweets

3.1. Classification Analysis

One of the most important part of learning classification algorithms is to analyse our results in order to understand how well the algorithms work and how efficient they are. There are a number of classification analysis methods,the most well-known of them is the f-score or f-measure.
Precision-
Precision (P) is the fraction of retrieved documents that are relevant
Precision = P(relevant|retrieved)
Recall-
Recall (R) is the fraction of relevant documents that are retrieved
Recall = P(retrieved|relevant)
These notions can be made clear by examining the following contingency table or the confusion matrix:

Table 1
Confusion Matrix

	Relevant	Non-relevant
Retrieved	true positives(tp)	false positives(fp)
Not Retrieved	false negatives(fn)	true negatives(tn)

Then Precision and Recall is given by

$$P = tp/(tp + fp)$$
$$R = tp/(tp + fn)$$

The F score can be interpreted as a harmonic mean of the precision and recall, where an F score reaches its best value at 1 and worst score at 0. The relative contribution of precision and recall to the F score are equal. The formula for the F score is:

$$F = \frac{1}{\alpha\frac{1}{P} + (1-\alpha)\frac{1}{R}} = \frac{(\beta^2+1)PR}{\beta^2 P + R} \tag{3}$$

where $\beta^2 = \frac{1-\alpha}{\alpha}$,$\alpha \in [0,1]$ and thus $\beta^2 \in [0,\infty]$. The default balanced F measure equally weights precision and recall, which means making α = 1/2 or β = 1. It is commonly written as F1,even though the formulation in terms of α more transparently exhibits the F measure as a weighted harmonic mean. When using β = 1, the formula on the right simplifies to:

$$F_{\beta=1} = \frac{2PR}{P+R} \tag{4}$$

However, using an even weighting is not the only choice. Values of $\beta < 1$ emphasize precision, while values of $\beta > 1$ emphasize recall. For example, a value of $\beta = 3$ or $\beta = 5$ might be used if recall is to be emphasized. Recall, precision, and the F measure are inherently measures between 0 and 1, but they are also very commonly written as percentages, on a scale between 0 and 100.

The following table shows the macro F1 Score and accuracy of our classification model as these were the metrics provided by the FIRE 2022 microblog track.

Table 2
Result

Data-set	Macro F1-Score	accuracy
I	0.304	0.353
II	0.239	0.435

3.2. Result Analysis

On working with the training data sets it was seen that stemming of textual data improved the results.This is attributed to the fact that the English language has several variants of a single term. These variances in a text corpus result in data redundancy when developing NLP or machine learning models. Such models may be ineffective.
It is essential to normalize text by removing repetition and transforming words to their base form through stemming from building a robust model.But still stemming doesn't always guarantee

perfect results due to errors in the stemming process.There are mainly two errors in stemming, such as: Over-stemming: It occurs when two words stem from the same root of different stems. Over-stemming can also be regarded as a false positive. Under-stemming: Under-stemming occurs when two words are stemmed from the same root that is not of different stems. Under-stemming can be interpreted as false negatives.
In general, stemming is straightforward to implement and fast to run. The trade-off here is that the output might contain inaccuracies, although they may be irrelevant for some tasks, like text indexing. Instead,lemmatization can be used which would provide better results by performing an analysis that depends on the word's part-of-speech and producing real, dictionary words. As a result, lemmatization is harder to implement and slower compared to stemming.

3.3. Conclusion and Future Work

It was observed that preprocessing,Tokenization and Stemming of the datasets improves the F-score.Hence they are very important while performing classification of textual data.
The results in the classification can be further improved to a larger extent by applying sentence embedding techniques like S-bert,Doc2Vec and Universal sentence encoder because on using tf-idf vectorisation we ignore the semantics behind the tweets which results in the poor classification of data.But still tf-idf is a better approach than using a simple bag of words model because here we take into account the importance or weight of the the words.My next step will be to try to to understand and dive deeper into more complex topics like sentence embedding and usage of neural networks for classification to improve on the results and come with new and original models which will be highly reliable and accurate.

Acknowledgments

It is matter of great pleasure for me to acknowledge my feelings of extreme gratitude and sincere regards to Dr. Kripabandhu Ghosh, Assistant Professor,CDS,IISER Kolkata, for his regular and dedicated guidance provided.

References

[1] For Covid-19 datasets IRMiDis FIRE 2022- https://sites.google.com/view/irmidis-fire2022/irmidis?authuser=0
[2] Christopher D. Manning, Prabhakar Raghavan and Hinrich Schütze, Introduction to Information Retrieval, Cambridge University Press. 2008.
[3] Hands-on Machine Learning with Scikit-Learn, Keras, and TensorFlow by Aurélien Géron
[4] Overview of Classification Methods in Python with Scikit-Learn, Dan Nelson,StackAbuse
[5] scikit-learn.org