IJARCCE

ISSN (Online) 2278-1021
ISSN (Print) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Vol. 9, Issue 5, May 2020

Political Bias Classification for Indian Context

S Lokesh[1], Shrirama C S[2], Shamantha Krishna K G[3], Vijay Kumar[4], Venkatesh H S[5]

Associate Professor, Computer Science and Engineering, The National Institute of Engineering, Mysuru, India[1]

Student, Computer Science and Engineering, The National Institute of Engineering, Mysuru, India[2,3,4,5]

Abstract: In the contemporary world no society in any country has a single ideology, especially in a democratic country like India. As any other country, we also have right-wing which enforces nationalism, conservatism and left-wing which is based on liberalism. Also, due to the growth of the Internet, social media has emerged as a great platform for people to express their opinions. Twitter has become a de facto standard platform for political based conversations. There are large numbers of tweets that are posted on twitter daily, most of them are biased towards right-wing or left-wing and the news reports from most of the media are also biased. In this paper we will present a machine learning model which can predict the bias in any given document for Indian context.

Keywords: Political-Bias, Right-Wing, Left-Wing, Centre, Twitter.

I. INTRODUCTION

In today's world, in any society, there exists different ideologies. In India, we have people who follow right-wing ideology who are nationalists, conservatives and they enforce Indian culture, likewise we have people who follow left-wing ideology and are liberals, who accept foreign culture and enforce secularism. However today, these ideologies are gaining much political interest, since there are two big national parties in India, one of which is right leaned, while the other is noticeably left leaned. The other noticeable thing is the growth of social media. Almost everyone today uses different social media to express what they feel. Twitter has become a mainstream for political leaders to express their opinions. There are lots of tweets getting posted online every day. There are also different media which are reporting news from different parts of the country and on various issues. It is found that the tweets from twitter and the reporting from the media are commonly biased towards any of the left wing or right-wing ideology. It is also quite agreeable that people following one ideology will never tweet or report the other way, so a tweet or a news report from a person or media who has right-wing ideology will always be right or centre but never left and same is true for a person or media who has left-wing ideology, that is they never tweet or report in right. In this paper, we will present a machine learning model which will detect this bias in any document for Indian context. Our machine learning model can detect political bias of a particular user or for a particular hashtag or for any media report.

II. RELATED WORKS

Previously also, there have been efforts from people to analyse data from twitter for different purposes. In [1] they have collected twitter data and have applied social-network analysis for French context. In [2] they have collected twitter data and have applied sentiment analysis techniques to study the political bias of the people towards the political parties in Spanish context. In [3] they have made efforts to identify the media bias in the USA.

III. IMPLEMENTATION

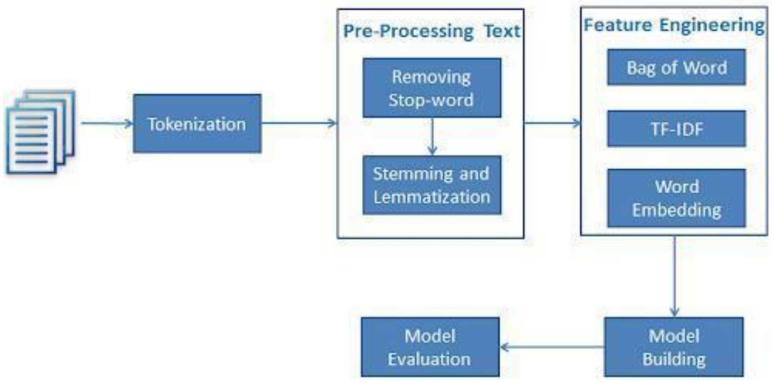


Fig. 1. Steps involved in building sentiment analysis-based machine learning model.

IJARCCE

ISSN (Online) 2278-1021
ISSN (Print) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Vol. 9, Issue 5, May 2020

In this section, we will explain how we build our machine learning model, which can detect political bias in any of the documents, which may contain tweets of a person, tweets related to a particular hashtag or for any media report. The below figure gives the overall picture of steps involved in building our model.

A. *Collecting the dataset Dataset is the first*: Foremost and important part of any machine learning project. We collect data from twitter through standard twitter API. Twitter consists of tweets from different people who may be from different background politics, cinema, technology, art and they may tweet different things apart from politics. We also have to collect tweets occasionally and retrain the model otherwise the model becomes outdated. We collected 1,500 tweets from 50 users. 25 users from each right-wing and left-wing. So, our total dataset included 75,000 tweets. These data are from spokespersons related to different political parties rather than any ministers who are in higher position. Because such personalities usually refrain from tweeting in a biased way.

B. Tokenization: Tokenization means breaking the larger sentence into smaller tokens and removing certain characters like punctuation marks.

C. Pre-processing text and stop-word removal: In this stage unwanted words or noise in the data are removed. Twitter allows users to tweet in different languages and in a multilinguistic country like India, people use different languages to express their opinions. We narrow our work only to English. People also make use of emojis in their tweets to express their opinions better. So apart from English, all the other words are removed from the tweets. Stop words are those words in any language, which are not useful while analyzing the sentiment, some of the more frequently used English stop words are "a", "of", "the", "and" which are generally regarded as 'functional words'.
We, in our work combined around 50 tweets as one entity, this was done because,

- We cannot get much information about the bias from one tweet.
- A tweet may also be very short and sarcastic.

D. Vectorization: The tokens after all the above steps are converted to machine readable form using vectorization methods. We used the following vectorization models.

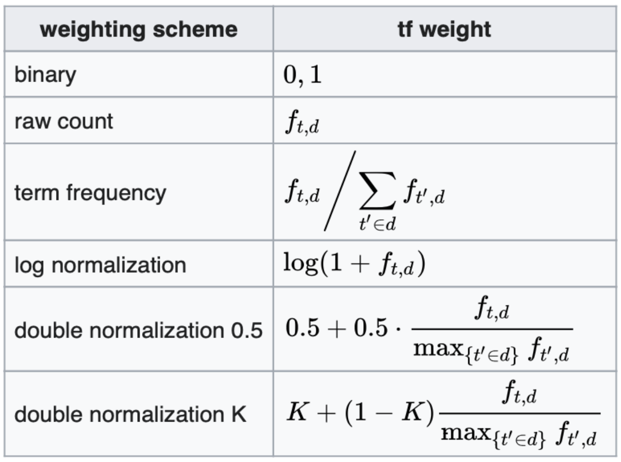
● Bag of words: It is a dictionary where each word in the document is taken as the key and number of occurrences of that word in the document is taken as its corresponding value. If BoW1 and BoW2 represents the bag of words representations for two text documents, and if we combine those two documents to make another text document, then the bag of word representation of that document, BoW3 is the disjoint union summing the multiplicity of each word in the combined document

$$BoW3 = BoW1 \uplus BoW2$$

● Term frequency-Inverse document frequency (tfidf): The tf-idf vectorization model determines the most important words in a collection of words.
Term frequency: It is the number of times a word appears in the document. If the term t occurs in a document d, if we denote the frequency of t in d as ft,d, then term frequency

$$tf(t,d) = f_{t,d}$$

There are different variants of term frequency (tf) weight, which is given in Fig. 2.



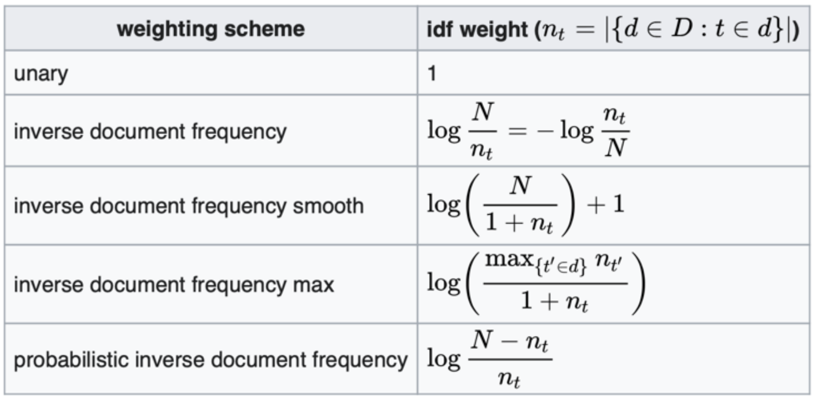
weighting scheme	tf weight
binary	$0, 1$
raw count	$f_{t,d}$
term frequency	$f_{t,d} \Big/ \sum_{t' \in d} f_{t',d}$
log normalization	$\log(1 + f_{t,d})$
double normalization 0.5	$0.5 + 0.5 \cdot \frac{f_{t,d}}{\max_{\{t' \in d\}} f_{t',d}}$
double normalization K	$K + (1 - K)\frac{f_{t,d}}{\max_{\{t' \in d\}} f_{t',d}}$

Fig. 2. Variants of term frequency(tf)

Inverse document frequency: It is the measure of how important the word is, in the collection of words, in the given documents or how much information a word provides i.e. whether the word is common or uncommon across all documents. It is calculated as logarithmic inverse fraction of the documents that contain that term t. If N is the total number of documents, if | {d ∈ D: t ∈ d} | denotes the number of documents where the term t appears, then inverse document frequency of term t in documents D is defined as

$$\mathrm{idf}(t, D) = \log \frac{N}{|\{d \in D : t \in d\}|}$$

If the term t does not appear in any of the documents, then | {d ∈ D: t ∈ d} | will become zero. So, to avoid 'divide by zero' error, it is adjusted as 1 + | {d ∈ D: t ∈ d}



weighting scheme	idf weight ($n_t = \|\{d \in D : t \in d\}\|$)
unary	1
inverse document frequency	$\log \frac{N}{n_t} = -\log \frac{n_t}{N}$
inverse document frequency smooth	$\log\left(\frac{N}{1 + n_t}\right) + 1$
inverse document frequency max	$\log\left(\frac{\max_{\{t' \in d\}} n_{t'}}{1 + n_t}\right)$
probabilistic inverse document frequency	$\log \frac{N - n_t}{n_t}$

Fig. 3. Variants of inverse document frequency(idf).

Fig. 3. shows different variants of inverse document frequency.
Then term frequency inverse document frequency is calculated as

$$tfidf(t, d, D) = tf(t, d).idf(t, D)$$

E.g. Assume that the word "dog" appears 4 times in a document which contains 100 words, the term frequency for the word "dog" is (4 / 100) = 0.04. Let the number of documents we have is 10 million and "dog" appears in 1000 of these documents, then the inverse document frequency is calculated as log (10,00,000/1,000) = 4, therefore the tf-idf weight for the word "dog" is 0.04*4 = 0.16

Tf-idf gives larger values for less frequent words in the t documents. Tf-idf value is high when both idf and tf values for a term are high i.e the word is rare in the whole document but frequent in that document. We used both bag of words model and tf-idf model for vectorizing, as expected the model trained with tf-idf
model vectorized data gave more accuracy than the bag of words model.

E. Classifiers: The vectorized training data is passed to classifiers to which construct a classification model which can be further used to predict to which class the new data belongs to. The different classifiers used in our work are:

1) Guassian Naïve Bayes
2) Linear regression
3) Multinomial Naïve Bayes
4) Lagrangian Support Vector machine.

We also used a voted classifier that is taking the mode from the predictions of above classifiers.

IV. USE CASES AND RESULTS

In this section we consider different use cases and analyze the results for each use case.

1) Model trained with Bag of words model vectorized data.
When we fed the data vectorized with bag of words model and fed this to Lagrangian Support Vector Machine based classifier, the accuracy was very low and was around 65%. This is mainly because the bag of words model just consider the presence or absence of a word. No other factor like the frequency of the words, sequence of their occurrence is not captured. It might be impactful in sentiment analysis applications like classifying reviews into good or bad, where you

have the luxury of a narrow set of keywords, but for a more complicated application like this, it doesn't fit in well. But still it can be used keep as a reference metric as you go on deriving the suitable model for the application.

2) Model trained with Tf-Idf vectorized data.
This model was more accurate than the earlier model and the accuracy was around 80%. This is expected as we involve more considerations in tf-idf. As mentioned before tfidf values more for the rarely occurring words, which means that the words which have impact for the sentence or the document is very well captured. Since it relieves on frequency of words and not the just occurrence, it tends to give more good results. Also word sequence can be considered using n-grams. This accounted for the raise in model accuracy. But still there was significant amount of wrong predictions observed. We used 4 different classifiers and built a custom voted classifier among them. As we observed all 4 classifiers tends to give the same result, with some exceptions on edge cases like the sarcastic content. Even after making the changes to classifiers iteration and customizing the different parameters like learning rate, there wasn't any significant increase in accuracy.

As we looked into the reason for this, we found that the data we are considering is actually not the ground truth. What it means to be the ground truth data in this context is, having the leftist content in the left dataset, and rightist content in the right dataset. But the method we involved breaks this. As we manually picked the twitter accounts, we are labeling all the tweets from that account to be either of the classes. But that might not be the case. For example, a noted person who has a leftist mindset need not always tweet in a leftist way, he can also tweet considerable amount of neutral content, just like saying "congrats ISRO on your successful launch…….". We are losing considerable amount of ground truth by labelling this tweet as left. The next time classifier sees ISRO, it might end up suggesting that is a left biased tweet. Clearly this shouldn't happen. But we cannot also do the tedious job of labelling thousands of tweets manually. To overcome this, we changed our assumption of "a biased person always tweets content to his bias" to "a biased person need not always tweet biased content, but when he does, it is the class of bias which he is identified with". This fairer assumption proved to be more valid as we see the results in the upcoming paragraphs.

To incorporate this assumption to our data set, we involved another technique, instead of taking a single tweet and wrongly labelling it, we took document of tweets, i.e., the tweets in bundles. In other words we combine series of tweets to extend the length of a single data point. This would cause merging of some of the neutral content along with the biased content. But still the document ends up having more biased content. Thus for a major extent we can avoid the mistake of labelling a neutral tweet to be biased. As we can see, a quick question arises, "what should be the size of the bundle of tweets?" If we have the size really big, then we end up having lesser number of data points for training. If we have the size really narrow, then the condition is no better than labelling a single tweet. We decided to have 50 tweets merged to a single entity to form a labelled data point. This would cut the data points for training, but statistical classifiers saturate with more data points. So, the data points we have suffice, for our classifiers.

An appreciable increase in accuracy was observed when we used this technique.Here are some of the models and their results with tfidf vectorized data. We used n-gram range of (1-3) to in the model. It means that largest sequence of three words are considered by tf-idf vectorizer, which in-turn adds to more accuracy, we avoided considering beyond 4, since it might lead to over-fitting. We also restricted the size of feature vector to be 10,000. We used different metric to analyse the results, each model gave a good result. Support values for centre were less, since the dataset contained less amount of neutral data, that should be acceptable.

1) Guassian Naïve Bayes

	Precision	Recall	F1-score	Support
Centre	0.95	1.00	0.97	9707
Left	0.99	0.97	0.98	11612
Right	0.99	0.97	0.98	10637
Accuracy			0.98	31956
Macro average	0.98	0.98	0.98	31956
Weighted average	0.98	0.98	0.98	31956

2) Linear regression

	Precision	Recall	F1-score	Support
Centre	1.00	0.97	0.99	9707
Left	0.96	1.00	0.98	11612
Right	1.00	0.97	0.98	10637
Accuracy			0.98	31956
Macro average	0.98	0.98	0.98	31956
Weighted average	0.98	0.98	0.98	31956

3) Multinomial Naïve Bayes

	Precision	Recall	F1-score	Support
Centre	0.99	0.97	0.98	9707
Left	0.95	0.98	0.96	11612
Right	0.98	0.96	0.97	10637
Accuracy			0.97	31956
Macro average	0.97	0.97	0.97	31956
Weighted average	0.97	0.97	0.97	31956

4) Lagrangian Support Vector machine.

	Precision	Recall	F1-score	Support
Centre	1.00	0.98	0.99	9707
Left	0.96	1.00	0.98	11612
Right	1.00	0.97	0.99	10637
Accuracy			0.98	31956
Macro average	0.99	0.98	0.99	31956
Weighted average	0.99	0.98	0.98	31956

V. CONCLUSION

In this paper, we have provided evidence that the political bias can be tracked through social media platforms, such as Twitter. The natural language processing can be effectively implemented for sentiment mining tasks, to predict the sentiment of a statement or a document. Our work is analogous to sentiment analysis, but the sentiment involved here is in the form of bias. So, it is little difficult to get good results in our case unlike other sentiment analysis projects or recommender systems. Our research helps in identifying the ideology of a person by analyzing his tweets. It detects whether he is left leaned, right leaned or neutral when his twitter handle is given and it can also detect the political bias of hashtag.Data is the most important thing in any of the work related to machine learning.
As interesting as it might sound, it is difficult to bring this into practice. The model might become source dependent, that means it might depend on the profiles on which we train the classifier. This can be easily removed by considering more profiles and less tweets from every profile. Thus avoiding the classifier to get trained on a narrow set of sources. The models depend heavily on the data. In our case the context of ideologies keeps changing over time. In other words, the model might become time dependent, that is the time frame during which the data is trained. So, it is difficult to capture the real ground truth. We need to have the context as reference when we judge the content as biased. This might need occasional retraining of model with the new data.

REFERENCES

[1]. Karina Sokolova, Charles Perez PSB, Elections and the Twitter community: the case of right-wing and left-wing primaries for the 2017 French Presidential election.
[2]. Digital Object Identifier 10.1109/ACCESS.2019.2917398 Indicator Proposal for Measuring Regional Political Support for the Electoral Process on Twitter: The Case of Spain’s 2015 and 2016 General Elections.
[3]. Automated identification of media bias in news articles: an interdisciplinary literature review.