HATE SPEECH DETECTION

MSc. DATA SCIENCE AND ECONOMICS

Tommaso Pessina*
Department of Economics, Management and Quantitative Methods
Department of Computer Science
University of Milan
Milan, Italy
tommaso.pessina@studenti.unimi.it

August 9, 2021

ABSTRACT

This project is meant to develop a hate speech detection process in order to classify a sentence in offensive, sexist, both or neither.
We will use some NLP techniques which, in combination with some regression methods, will result in a text classification pipeline.
We used three main Text Mining methods:

- Term Frequency–Inverse Document Frequency;
- Bgrams;
- Bag Of Words;

***K*eywords** NLP · Logistic Regression · Python · Text Classification · More

1 Introduction

Firstly we should spent few word about the datasets used in the development of this project.
The first dataset used is a collection of tweet classified as:

- 0: hate speech;
- 1: offensive language;
- 2: neither.

The second one is a collection of sentences classified as:

- "1" represents sexism;
- "0" represents no sexism or neutrality.

The basic idea is to proceed by step, i.e. first we classify the sentence to hate/offensive/neither and then, we check for sexism.

2 Data cleaning

Now we should analyse how we cleaned both datasets for the regression methods. First we should delete word like "@user" and "@url" because, since the first dataset is a collection of tweet, this two word are really frequent but they do not carry useful information.

*MSc. Data Science and Economics student | Bsc. Computer Science

Then we should, for simplicity, convert everything to lowercase and delete any unwanted character (e.g. symbol or number) by means of Regular Expression.
It's also important to remove all the stopword (we use "english" as language) and re-write all the contraction present in the dataset in extended form.
Then, if it is necessary, tokenize the sentence and obtain the bigrams thank to the Natural Language ToolKit library.

3 Model Selection

First we divide the dataset into train and test using the 70-30 rule, which imply that the 70% of our dataset will be the train set.

We choose to fit a Logistic Regression because usually with text problem, a simpler model will perform better. We should precise that, normally, Logistic Regression work with binary classification problem, but in the hate speech detection case the label are not dichotomous, as matter of fact we have a multi-class problem.
Since we use the sklearn.linear_model.LogisticRegression method, this will not be a problem because in the multiclass case, the training algorithm uses the one-vs-rest (OvR) scheme if the "multi_class" option is set to "ovr", and uses the cross-entropy loss if the "multi_class" option is set to "multinomial" (we use the multiclass default value that is "multinomial"). Although we address to this algorithm as Logistic Regression, sklearn will select and train a Multinomial Logistic regression method.
Then we choose to fit a model with Bag Of Word and one with TF-IDF method. Moreover, in order to select the best parameters, we run a GridSearch 5-cross validation.
In particular:

- The TF-IDF (Term Frequency–Inverse Document Frequency) is a method intended to reflect how important a word is to a document in a collection or corpus.

- The Bag Of Word (or BoW) method consist in, like the name may suggest, put all the word of a corpus in a "bag" (e.g. an array) and we may also associate to each word it's frequency in the document.

As said, for each method we will run a GridSearch 5-cross validation which will help us to select the best parameters for our algorithm which will lead us to select the best method (i.e. the one with highest score) for our problem.

Note that, for the Sexism classification problem, we run also a BiGrams method. This imply that we clean further the dataset by creating the bigrams (i.e. couples of two adjacent words) and to define a new vectorizer. We will check also this method with the GridSearch 5-cross validation method.
Finally, we propose a "pipeline", that here will act like a main function, that can be used as hate speech detection method (with sexism classification).

4 Model result

In this chapter we will discuss about the result of both methods and we will start by the hate/offensive speech detection. First we define a Vectorizer which will help us to fit the training true label data, then we create a logistic regression method, validate it with a 5-cv which will return an Accuracy range as represented in Figure 1.

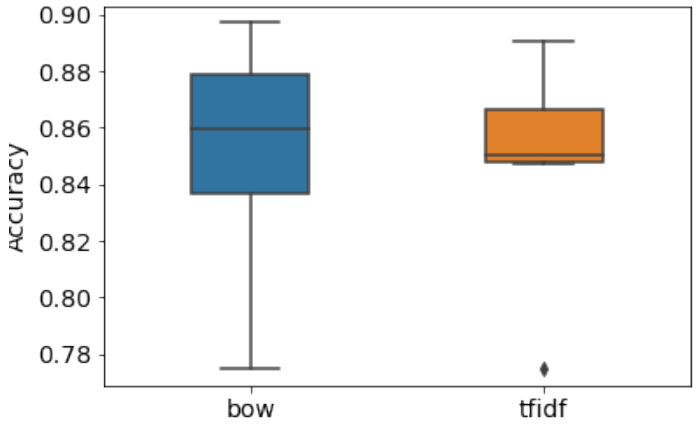

Figure 1: Grid-CV results

Moreover, the test score are:

- Test Score with bow features 0.88;
- Test Score with tf-idf features 0.84.

Afterwards we can train again our logistic regression method with the best parameters selected by our previous Cross-Validation, which will output a test score of:

- Test Score with bow features 0.89;
- Test Score with tf-idf features 0.88.

Since the result are pretty close, we run a test with four sentences in order to analyse the prediction probability of each sentence for selecting the overall best method, which [in this case] is the Bag Of Word.

To conclude the first part, we execute also a feature analysis which will give us the more relevant word in terms of text classification. In Figure 2 we can see that, for example, the three most relevant word are "fag***", "fuc*", "nig**".

	coefficient	word
9338	1.509241	f a g
9337	0.789540	f u c
9336	0.675509	n i g
9335	0.628503	h i t
9334	0.563003	i g g
...	...	...
4	-0.324189	r a t
3	-0.333453	t o
2	-0.411769	i r d
1	-0.429031	a m
0	-0.509568	b i r

9339 rows × 2 columns

Figure 2: Hate speech feature

Now we should carrying on our analysis to the sexism part which, in principle, will give us a performance of:

- Test Score with bow features 0.77;
- Test Score with tf-idf features 0.74;
- Test Score with bigrams features 0.66.

Again we run a 5-grid cross-validation and we can see in Figure 3 the result of each method for each cross-validation step.

	bow	tfidf	bgrams
0	0.540881	0.540881	0.540881
1	0.618868	0.778616	0.540881
2	0.801258	0.779874	0.563522
3	0.777358	0.761006	0.602516
4	0.766038	0.742138	0.612579
5	0.754717	0.730818	0.588679

Figure 3: Sexism grid-CV results

Finally we can train the three methods with the best parameters and we obtain:

- Test Score with bow features 0.81;
- Test Score with tf-idf features 0.77;
- Test Score with bgrams features 0.62.

We can see that the Bag Of Word model perform better, so for now we will chose this one as winner for this job. We run a test with three sentences in order to check further which method lead us to a more consistent result and turns out that TF-IDF and Bgrams has similar results. Since from the test score the TF-IDF method has an higher score than the bgrams method (moreover it is pretty close to the BoW which the highest one) we will select the TF-IDF method as winner for this job.

Finally we can analyse the most relevant word in terms of sexism text classification. In Figure 4 we can see that, for example, the four most relevant word are "(sexist, right)", "(sexist, realist)", "(blond, want)" and "(women, cannot)".

	coefficient	word
3820	2.226084	(sexist, right)
3819	2.226084	(sexist, realist)
3818	2.226084	(blond, want)
3817	1.999973	(women, cannot)
3816	1.969758	(believe, women)
...	...	...
4	-1.845930	(good, man)
3	-1.850283	(opinion, valid)
2	-2.031441	(one, said)
1	-2.268173	(women, men)
0	-2.268713	(impo, ant)

Figure 4: Sexism feature

5 Conclusion

Finally, from our analysis turns out that:

- for the hate speech detection the most relevant word are actually bad word meanwhile, for the sexism detection, the most relevant word are negative sentence or words in support of sexism;
- for the hate speech detection method we select a Bag Of Word Logistic Regression method, meanwhile, for the sexism detection we select a TF-IDF Logistic Regression Method;

In order to check and verify our proposed method we create a pipeline for this two method. We propose 3 sentences:

- sentece1 = "I love my wife"
- sentece2 = "if you want to work you must dress like a b***h"
- sentece3 = "you are so retarded!"

We proceed by cleaning our sentences and to pass them to our Text Classificiation algorithm that will return a dictionary with the sentence as key and the output as value.
We may now observe the result of the two dictionary, one for each method, and we obtain that:

- 'I love my wife' is NOT offensive or sexist.
- 'if you want to work you must dress like a b***h' is OFFENSIVE and SEXIST.
- 'you are so retarded!' is OFFENSIVE.