


Hate Speech Detection Using Text Mining and Machine Learning

Safae Sossi Alaoui, Faculty of Sciences and Techniques, Moulay Ismail University of Meknes, Morocco*

Yousef Farhaoui, Faculty of Sciences and Techniques, Moulay Ismail University of Meknes, Morocco

 <https://orcid.org/0000-0003-0870-6262>

Brahim Aksasse, Faculty of Sciences and Techniques, Moulay Ismail University of Meknes, Morocco

ABSTRACT

Automatic hate speech detection on social media is becoming an outstanding concern in modern countries. Indeed, hate speech towards people brings about violent acts and social chaos; hence, law prohibits it, and it engenders moral and legal implications. It is crucial that we can precisely categorize hate speech and not hate speech automatically. This allows us to identify easily real people who represent a threat for our society. In this paper, the authors applied a complete text mining process and naïve bayes machine learning classification algorithm to two different data sets (tweets_Num1 and tweets_Num2) taken from Twitter to better classify tweets. The results obtained demonstrate that the model performed well regarding different metrics based on the confusion matrix including the accuracy metric, which achieved 87.23% on the first dataset and 93.06% on the second.

KEYWORDS

Hate Speech, Machine Learning, Naïve Bayes, Sentiment Analysis, Text Mining

1 INTRODUCTION

Recently, people communicate and discuss their opinions in digital form more and more by taking advantage of online social networks like Twitter, Facebook, and Instagram and so on. These social media have many benefits to humanity in enhancing culture diversity; otherwise, the dark side of social media causes hazardous consequences when it comes to attack others by harassing, bullying and threatening them using hateful expressions known as hate speech. Hate speech (Chetty & Alathur, 2018) can be defined as a threatening and abusive language which expresses hatred against a particular group especially on the basis of race, color, religion, ethnicity and even gender.

Generally speaking, media have a significant impact on individuals' beliefs and perceptions (Mastorocco & Minale, 2018). Indeed, when social media have been exploited as a tool to convey hate, racist and terroristic contents, it can engender crimes and violent acts (Jendryke & McClure, 2019). For example, Chetty and Alathur (2018) emphasized the strong correlation between hate speech and terrorist activities. As a result, collaborative efforts between government, Internet service providers and online social networks will effectively define policies to combat both hate speech and terrorism.

In order to fight Cyber hate, many organizations have enforced their policies towards law, technology and education so as to prevent and reduce its negative influences (Blaya, 2019).

To handle hate speech automatically, it can be seen as a part from sentiment analysis or opinion mining (Hussein, 2018) which utilizes the natural language processing (NLP), text mining and

computational algorithms to automate the identification and extraction of subjective information from text. The hate speech is a behavior built from education, TV and many other factors. It is hard to design a hate speech detector since it depends on the language of the hater. There exist three sentiment analysis techniques (Medhat et al., 2014) lexicon based method, machine learning approach, and hybrid approach. Effectively, the mechanisms of hate speech detection are part of the mentioned approaches; the **lexicon-based methods** tend to calculate semantic orientation of words or phrases in a text by means of a dictionary which provides words with a positive or negative sentiment value assigned to each of the words. The **machine learning approaches** are used to get a discriminative function that can separate hate speech from normal speech. Machine learning algorithms are programs; that considered as an evolution of the regular algorithms, which can automatically learn from data and improve from experience, without human intervention. In our case, the hate speech detection can be seen as a supervised learning problem where both the inputs and outputs are already known, which means that the data used to train the algorithm is already labeled with correct answers in order to generate reasonable predictions for the response to new data. Since the outputs are discrete, the classification algorithms (Sossi Alaoui et al., 2018, 2017) are used to categorize the data into specific groups or classes. Finally, **the hybrid approach** that combines machine learning methods with lexical-based approaches.

In this paper, we focus on machine learning approach because lexical based method tend to confuse between terms used in hate speech and offensive language and therefore it gives low precision (Davidson et al., 2017).

The main objective of this paper is to propose both a framework and a model to detect automatically hate speech in social media for both binary and multiclass problems by using two public datasets taken from Twitter. The framework will describe a full implementation of a text mining process and the model will be based on Naïve Bayes a supervised machine learning algorithm. The reason behind choosing Naïve Bayes and not another machine learning classification algorithm (Sossi Alaoui et al., 2018, 2017) is according to numerous research works that will be discussed in the next section, which were conducted a comparative study of several machine learning algorithms; Naïve Bayes was the best method in terms of different performance measures and which proved its efficiency and simplicity in dealing with almost all problems related to sentiment analysis (Alam & Yao, 2019). Precisely, the accuracy of Naïve Bayes algorithm as S. Alam and N. Yao (2019) has been considerably improved after the application of preprocessing steps compared to maximum entropy (MaxE), and support vector machines (SVM) for sentiment analysis.

The motivation behind this work is to overcome the difficulties of generalizing the resulting models to detect hateful text-based content, which are present in the literature, and to propose a framework for the construction of a precise model capable of detecting hate speech automatically by performing a complete text mining process based on Naïve Bayes; a probabilistic classification machine learning algorithm and applied to two different datasets in terms of data size, type of classification task (binary or multiclass) and data sources (data.world and Kaggle).

The remainder of the paper is organized as follows. Previous related works are discussed in section II. The methodology of this work is explained in detail by designing and describing the process of text mining analytics in section III. Section IV presents the results obtained. Finally, section V concludes the paper.

2 RELATED WORKS

In order to offer an overview of the main works related to this area of research, this section has focused on numerous papers on sentiment analysis and hate speech detection. First, Gonçalves et al. (2013) made a comparison between eight popular sentiment methods expressly SentiWordNet, SASA, PANAS-t, Emoticons, SentiStrength, LIWC, SenticNet, and the Happiness Index using a web service named iFeel. They additionally developed a competitive approach in terms of coverage and agreement.

Equally important, prior studies have attempted to analyze sentiment in different languages other than English (Al-Twairsh et al., 2017), for instance; Boudad et al. (2018) gave a global review of Arabic sentiment analysis, and discussed the levels of sentiment analysis namely document level, sentence level, and aspect level. They also explained its approaches, including supervised, unsupervised and hybrid methods and finally they presented the challenges of sentiment analysis about the specific characteristics of Arabic language.

While papers dealing with hate speech have raised more and more today, for example; Almatarneh et al. (2019) evaluated the efficiency of supervised learning classifiers to recognize Hate Speech in Twitter for two languages: English and Spanish. The dataset in English contains 9000 tweets for training, 1000 for developing and 2805 for the test and concerning the Spanish dataset, it consists of 4469 tweets for training, 500 for developing, and 415 for the test.

Based on the most influence linguistic features namely, N-grams Features (Term Frequency-Inverse Document Frequency (TF-IDF) and CountVectorizer) and Doc2Vec, they compared several classifiers in terms of F1 measure including Support Vector Machine (SVM), Naïve Bayes (Gaussian NB and Complement NB), Decision Tree (DT), Nearest Neighbors (KN), Random Forest (RF) and Neural Network (NN). The results obtained showed that Complement Naive Bayes, Support Vector Machine, and Random Forest visibly outperformed the other classifiers and achieved the highest F1 scores with successively 0.76 and 0.77 for English tweets and Spanish tweets when using N-grams features apart from whether the representations are CountVectorizer or TF-IDF.

Ruwandika and Weerasinghe (2018) made a comparison between five supervised and unsupervised classification algorithms namely; Support Vector Machine, Naïve Bayes Classifier, Logistic Regression Classifier, Decision tree Classifier and K-Means Clustering algorithm. The experiment was done by a local English text dataset dealing with hate speech, which consists of 1500 comments; two-thirds of those comments were manually annotated as hate or no hate.

From all the supervised and unsupervised methods tested, Naïve Bayes classifier with Tf-idf features achieved good results with an accuracy of 0.739, a precision of 0.75, a recall of 0.739 and F-score of 0.719.

Kiilu et al. (2018) aimed at developing a reliable tool for detection of hate tweets. The collected training data consists of 45645 tweets and test data of 22820 tweets. This training data was gathered from Tweepy API, which contains sample sentences and words from a text file and were classified manually. The proposed approach encompasses pre-processing to eliminate undesirable parts of speech using n-grams, and tweet classification and evaluation. After generating the models based on the following algorithms namely NU Support Vector Classification (NUSVC), MultinomialNB, BernoulliNB, Logistic Regression, Linear SVC and SGD classification. Results obtained showed Naive Bayes classifier reached meaningfully better performance than existing methods in hate speech detection algorithms with precision, recall, and accuracy values of 58%, 62% and 67.47%, respectively.

Davidson et al. (2019) measured the racial bias in hate speech and abusive language in five datasets of Twitter which contained respectively 130k tweets annotated as Racism, Sexism and Neither, 2876 tweets annotated as Racism, Sexism, Racism and Sexism and Neither, 24,783 tweets annotated as hate speech, offensive language, or neither, 20,360 tweets related to anti-black racism, Islamophobia, homophobia, antisemitism, and sexism. And annotated as Harassment and Non, 91,951 tweets annotated as Hate, Abusive, Spam and Neither. They found that tweets written in African American English were abusive at significantly higher rates.

They trained for each dataset the classifier regularized logistic regression with bag-of-words features to predict the class of unseen tweets. The generated models are measured using three criteria namely Precision Recall and F1 for each class, for instance, the values of precision varied between 0,32 to 0,96.

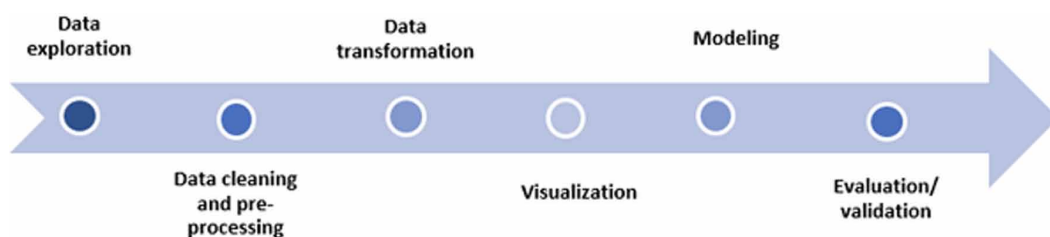
Each work in the literature proposed a model, but none suggested a general framework to be pursued. Our contributions are to propose a framework to be followed so as to obtain the highest performance when detecting hate speech in social media, and to recommend the best machine learning

algorithm based on existing articles in terms of robust performance criteria. Our work will also be novel in terms of the tool used which offers well developed packages which handle text effectively and provide facilities to visualize data properly and finally in terms of the selected datasets which seem to be huge and public compared to other papers.

3 METHODOLOGY

In this work, we conducted a complete text mining process, as shown in Figure 1, based on six major steps; starting with data description, then we do data exploration, data cleaning and pre-processing, data transformation, visualization, modeling and finally evaluation/validation.

Figure 1. The methodology of work



3.1 Dataset and Tool Description

The present study adopts two datasets taken from Twitter; a well-known social networking service on which users post and interact with tweets. The two datasets that deal with hate speech are named respectively: tweets_Num1 and tweets_Num2.

tweets_Num1 (*Hate Speech and Offensive Language - Dataset by Thomasrdavidson | Data. World, n.d.*) is created by Thomas Davidson et al. in 2017, they used Twitter API to search for tweets with terms belonging to hate speech lexicon. They extracted 85.4 million tweets from 33,458 Twitter users. Then, they took a random sample of this corpus and had coded manually by CrowdFlower (CF) workers (Davidson et al., 2017). **tweets_Num1** contains 24783 observations of six (6) attributes; count, hate_speech, offensive_language, neither, class and tweet described in Table 1. While, tweets_Num2 (Twitter Sentiment Analysis, n.d.) provided by Analytics Vidhya (Analytics Vidhya, n.d.) in 2019, which is a community from India interested in building the next generation data science ecosystem. It contains 31962 observations of three (3) attributes; id, label, and tweet, as shown in Table 2.

In this work, we utilized R, which is an advanced statistical programming language, using an integrated development environment (IDE) for R named RStudio. R includes various collections of functions, named a package, which form a fundamental unit of reproducible R code.

The packages used in this study are NLP which provides classes and methods for natural language processing, tm offers text mining tasks, dplyr enables manipulation of data frames, e1071 allows the use of functions for statistic and probabilistic algorithms, ggplot2 offers data visualization, caret includes functions for training and plotting classification and regression models, RColorBrewer manages color palettes, wordcloud provides visual presentations of words.

3.2 Data Exploration

Data exploration consists of synthesizing the main characteristics of a dataset.

To display compactly the internal structure of an R object, we can use the function “str {utils}”. As shown in Figure 2 and Figure 3, the two datasets; tweets_Num1 and tweets_Num2, are represented

Table 1. Description of the tweets_Num1

Attribute's name	type	Attribute description
count	Integer	The number of users who coded each tweet
hate_speech	Integer	The number of users who judged the tweet to be hate speech.
offensive_language	Integer	The number who judged the tweet to be offensive
neither	Integer	The number of users who judged the tweet to be neither offensive nor hate speech.
class	Integer	Class label for majority of users. 0 - hate speech, 1 - offensive language, 2 - neither
tweet	Character	The message posted in Twitter

Table 2. Description of the tweets_Num2

Attribute's name	type	Attribute description
id	Integer	Identifier for each tweet
label	Integer	Class label of tweet 0 – not hate speech, 1 – hate speech
tweet	Character	The message posted in Twitter

Figure 2. The structure of tweets_Num1

```

Console ~/
> # view the structure of dataset
> str(tweets_Num1)
'data.frame':  24783 obs. of  7 variables:
 $ X          : int  0 1 2 3 4 5 6 7 8 9 ...
 $ count      : int  3 3 3 3 6 3 3 3 3 3 ...
 $ hate_speech : int  0 0 0 0 0 1 0 0 0 1 ...
 $ offensive_language: int  0 3 3 2 6 2 3 3 3 2 ...
 $ neither    : int  3 0 0 1 0 0 0 0 0 0 ...
 $ class      : int  2 1 1 1 1 1 1 1 1 1 ...
 $ tweet      : chr  "!!! RT @mayasolovely: As a woman you should
    
```

Figure 3. The structure of tweets_Num2

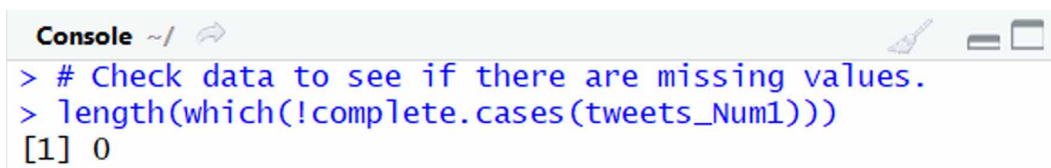
```

Console ~/
> str(tweets_Num2)
'data.frame':  31962 obs. of  3 variables:
 $ id   : int  1 2 3 4 5 6 7 8 9 10 ...
 $ label: int  0 0 0 0 0 0 0 0 0 0 ...
 $ tweet: chr  " @user when a father is dysfunctional and is so selfish
    
```

as data frames; which can be defined as a two-dimensional array or a table containing columns and rows representing respectively; variables and observations and created by loading the datasets from CSV files.

Outliers and Missing data are two real problems facing data scientists in almost all research and may cause a significant effect on drawing conclusions. For that reason, we ought to check our data, specifically if it contains missing values or not. According to Figure 4 and Figure 5, we have not, fortunately, any missing values for the two datasets.

Figure 4. Checking missing values in tweets_Num1






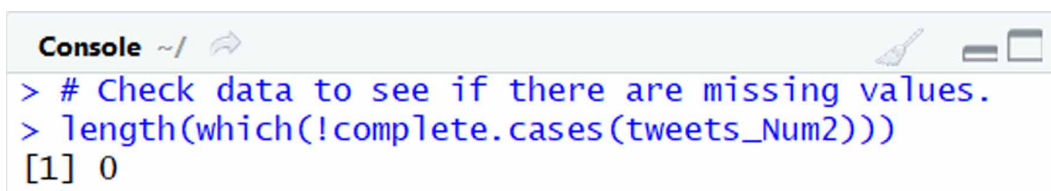



```
Console ~/     
> # Check data to see if there are missing values.  
> length(which(!complete.cases(tweets_Num1)))  
[1] 0
```

Figure 5. Checking missing values in tweets_Num2



```
Console ~/     
> # Check data to see if there are missing values.  
> length(which(!complete.cases(tweets_Num2)))  
[1] 0
```

We plot the distribution of text length of tweets by each class in the two datasets as shown in Figures 6 and 7, by adding a new feature for the length of each tweet.

3.3 Data Cleaning and Pre-Processing

After extracting text from the two data sets, we build Corpora, which is a collection of documents containing (natural language) text, by using the function “Corpus” provided by the package “tm”. Then, we move to text cleaning and pre-processing; described in Figure 8, based on common preprocessing functions which include converting all characters lowercase “tolower()”, removing all punctuation marks “removePunctuation()”, getting rid of numbers “removeNumbers()”, eliminating excess whitespace “stripWhitespace”, removing URL “removeURL()”. Finally, we conducted Natural Language Processing (NLP) for filtering from text English Stop words like: “I”, “me”, “my”, “myself”, etc... In addition, Stemming for reducing a word with prefixes and suffixes to a root or base word known as stem/lemma, simply put, the words: “written”, “writer”, “writing”, “wrote” might all be reduced to a common representation “write”.

3.4 Data Transformation

Transforming unstructured text into structured data is necessary for any kind of analysis. Yet, it remains a complicated process in which data need to be encoded in a way to be used by machine learning algorithms. As shown in Figure 9, we transform the tweets to represent numerically text named Bag of Words.

Bag of words known for short as BoW describes the occurrence of words within a document. It is considered as a way of extracting features from the text for use in machine learning algorithms.

Figure 6. The distribution of text lengths of tweet in tweets_Num1 by class

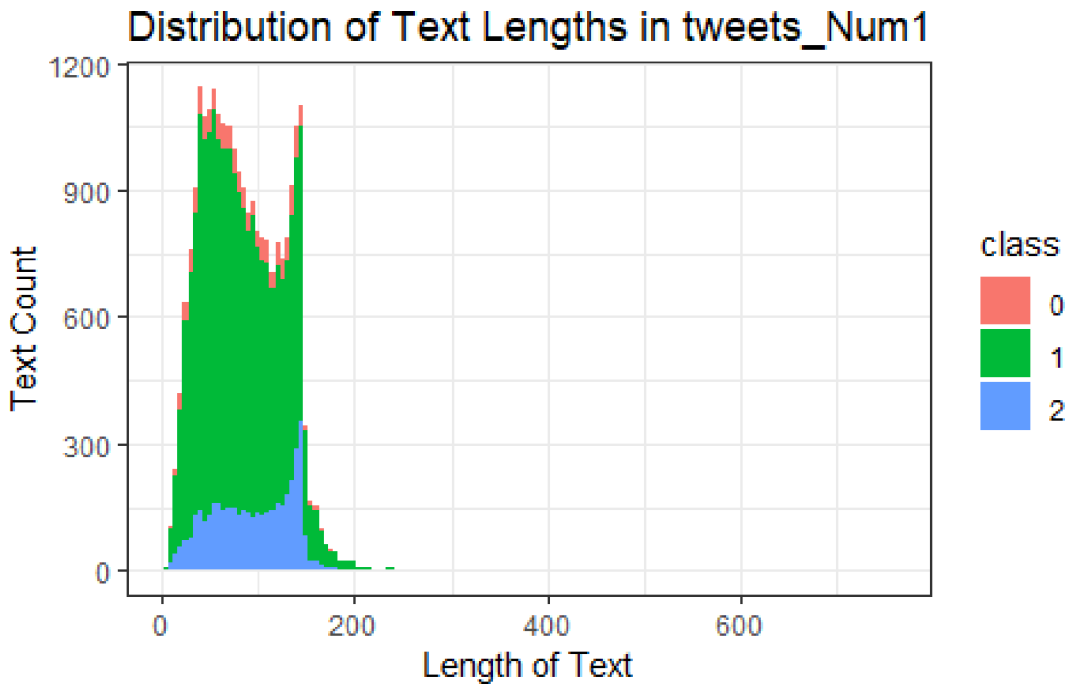


Figure 7. The distribution of text lengths of tweet in tweets_Num2 by label

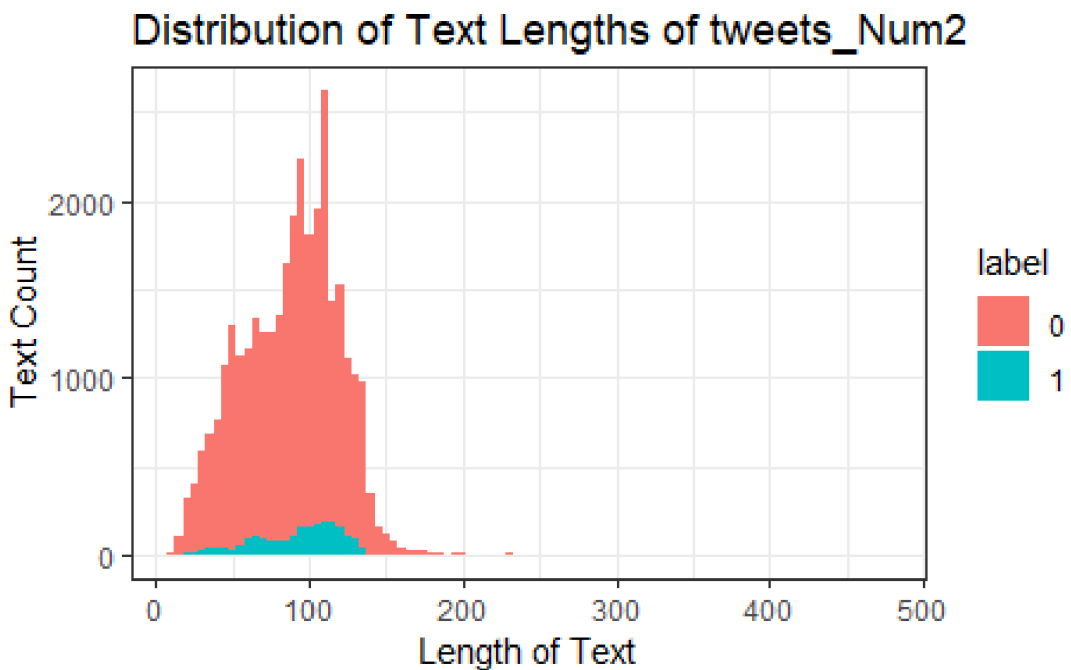
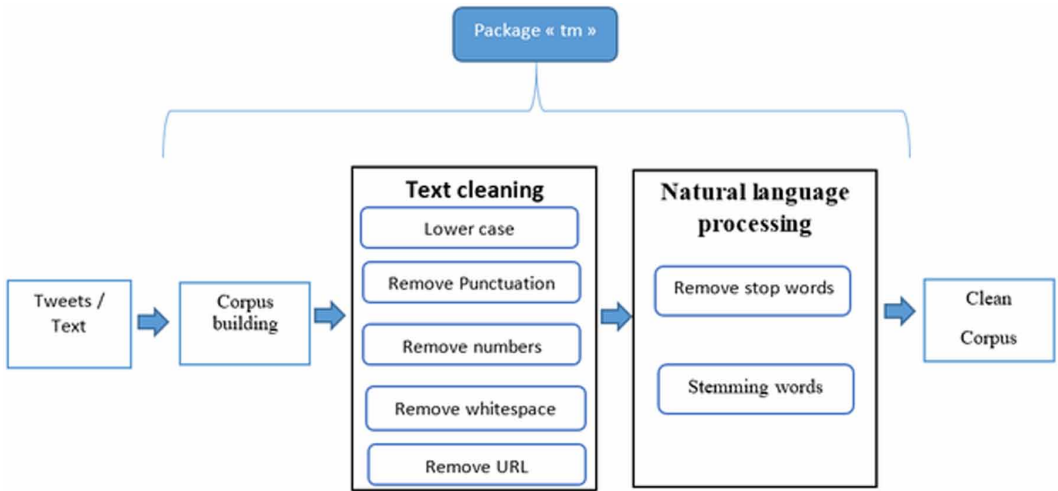


Figure 8. Cleaning and preprocessing tweets



Document-term matrix (DTM) or Term-document matrix implements the concept of BoW, it calculates the frequency of terms that occur in a set of documents. It is a mathematical matrix with two dimensions; in which the rows represent the terms (words) t_i and the columns are the documents d_j and an entry $m_{i,j}$ which can be calculated depending on the chosen term weighting schemes (TF, IDF, TF-IDF.. etc).

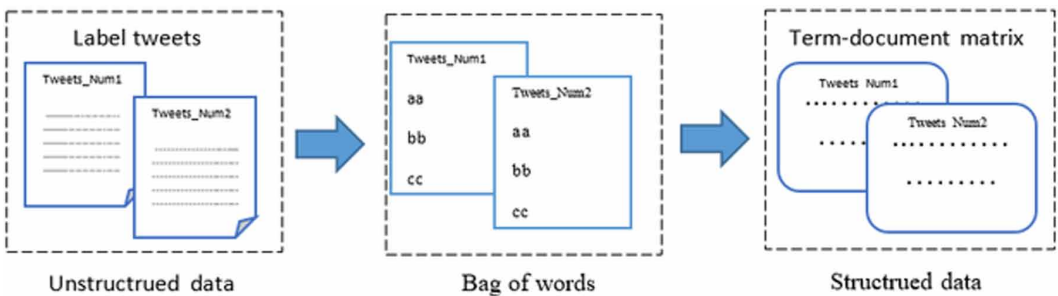
In the case of:

- Term Frequency (TF): $m_{i,j} = tf_{i,j}$ which is the number of occurrences of term t_i in document d_j
- Inverse Document Frequency (IDF): $m_{i,j} = idf_i$

$$idf_i = \log_2 \left(\frac{|D|}{|\{d \mid t_i \in d\}|} \right) \quad (1)$$

Where the cardinal of D ($|D|$) is the total number of documents and the cardinal of d ($|\{d \mid t_i \in d\}|$) the number of documents where term t_i appears.

Figure 9. Transforming unstructured data into structured data or DTM



- Term Frequency - Inverse Document Frequency (TF-IDF): $m_{i,j} = tf_i df_{i,j}$

$$Tf_i df_{i,j} = tf_{i,j} \cdot idf_i \quad (2)$$

In R language, it defaults to `weightTf` for term frequency weighting which we had chosen in this research.

3.5 Data Visualization

Word cloud named also text cloud or tag cloud is a famous visualization of words typically used to focus on trending terms based on the most data keywords. Therefore, the more a precise word presents in a source of textual data, the bigger and bolder it appears in the word cloud. Figures 10, 11 and 12, present word clouds of the three classes of tweets_Num1: “hate_speech”, “offensive_language” and “neither”. Figures 13 and 14 show word cloud of the two labels of tweets_Num2; “hate_speech” and “not_hate_speech”.

3.6 Modeling

The modeling process adopted in this paper, covers a full implementation of Naïve Bayes machine learning (Kang & Jameson, 2018) model applied to the detection of Hate speech in Twitter as shown in Figure 15. After cleaning and pre-processing task, we move to annotate our text into training set with 70% and test set with 30%. Training set is the initial set of data that can learn from and make predictions on new data. To perform our model, we use the test set which provides an unbiased evaluation of a model fit and produce sophisticated results. In addition, we randomly split our text because the proportion of class ought to be similar in both training and test data, for that we check it in the two datasets as shown in Figure 16 and 17.

Naïve Bayes is a probabilistic machine learning classification algorithm that consists of Bayes’ theorem; it assumes that the presence of a specific feature in a class is independent to the presence of any other feature.

3.7 Evaluation

To evaluate our proposed model, we use confusion matrix and its associated metrics. To describe the associated statistics of confusion matrix, we suppose a binary classification problem, the confusion matrix with notation as shown below in Box 1.

The formulas (*ConfusionMatrix Function | R Documentation*, n.d.) used are:

$$\text{Sensitivity} = a / (a + c) \quad (3)$$

$$\text{Specificity} = d / (b + d) \quad (4)$$

$$\text{Prevalence} = (a + c) / (a + b + c + d) \quad (5)$$

$$\text{Detection rate} = a / (a + b + c + d) \quad (6)$$

$$\text{Detection prevalence} = (a + b) / (a + b + c + d) \quad (7)$$

$$\text{Balanced accuracy} = (\text{sensitivity} + \text{specificity}) / 2 \quad (8)$$

$$\text{Precision} = a / (a + b). \quad (9)$$

$$\text{Recall} = a / (a + c) \quad (10)$$

$$F1 = (1 + \beta^2) * \text{precision} * \text{recall} / ((\beta^2 * \text{precision}) + \text{recall}) \quad (11)$$

(Where $\beta = 1$ for this function)

$$\text{PPV} = (\text{sensitivity} * \text{prevalence}) / ((\text{sensitivity} * \text{prevalence}) + ((1 - \text{specificity}) * (1 - \text{prevalence}))) \quad (12)$$

$$\text{NPV} = (\text{specificity} * (1 - \text{prevalence})) / (((1 - \text{sensitivity}) * \text{prevalence}) + ((\text{specificity}) * (1 - \text{prevalence}))) \quad (13)$$

Otherwise, the accuracy is the ratio of number of correct predictions to the total number of input samples. It indicates how close a measured value is to the actual (true) value.

$$\text{Accuracy} = (a + d) / (a + b + c + d) \quad (14)$$

Kappa or Cohen's kappa indicates how the model exceeded random predictions in terms of accuracy.

$$\text{Kappa} = ((\text{Observed Accuracy} - \text{Expected Accuracy}) / (1 - \text{Expected Accuracy})) \quad (15)$$

4. RESULTS

First, the experiments were performed on an Intel® Core™ i5-6300U CPU @ 2.40 GHz 2.50 GHz machine with 8 GB Ram using the language R. The results obtained after applying the previous modeling process are based on confusion matrix which describes the performance of a classification model by calculating a cross-tabulation of observed and predicted classes and it allows the visualization

Figure 12. Word cloud for "Neither" class



Figure 13. Word cloud of "hate_speech" label

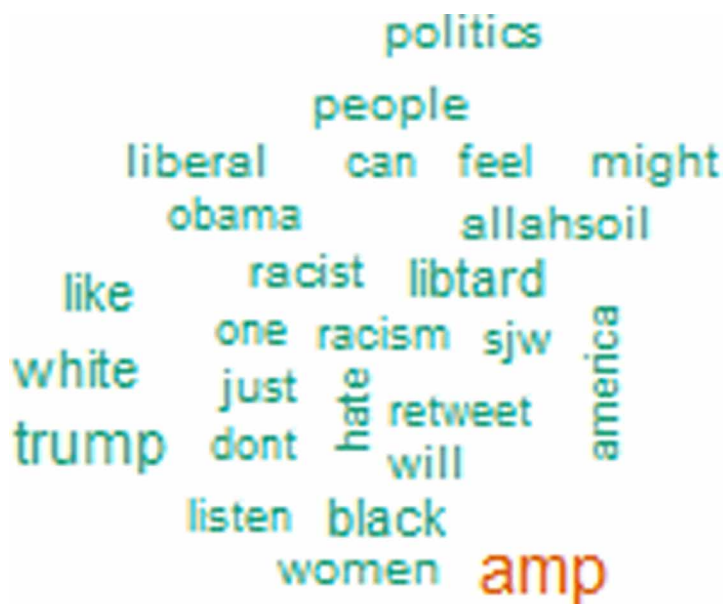


Figure 14. Word cloud of “not_hate_speech” label

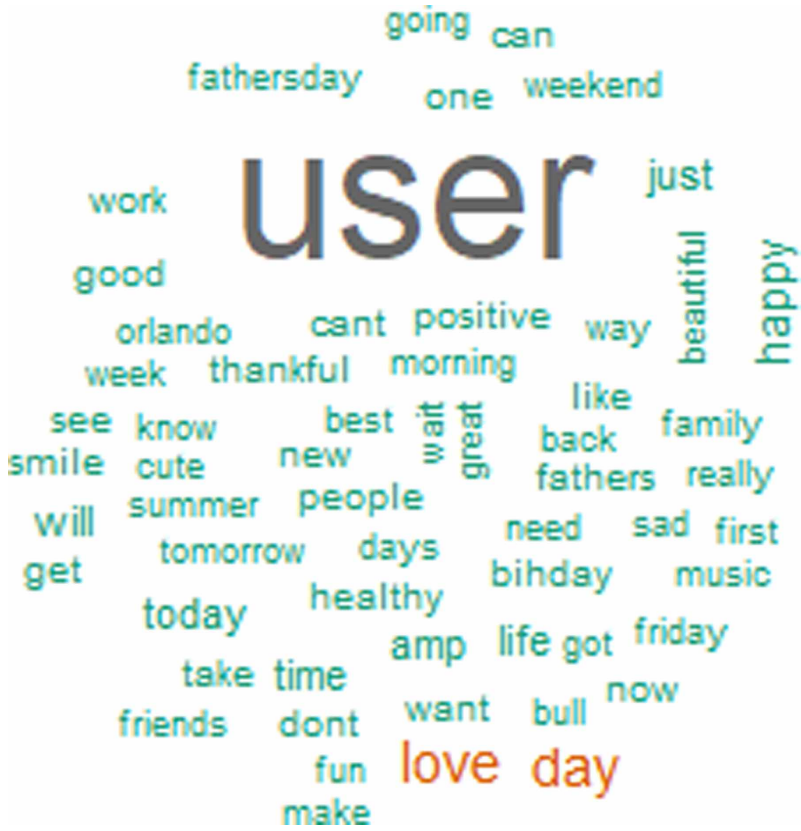


Figure 15. The complete modeling process

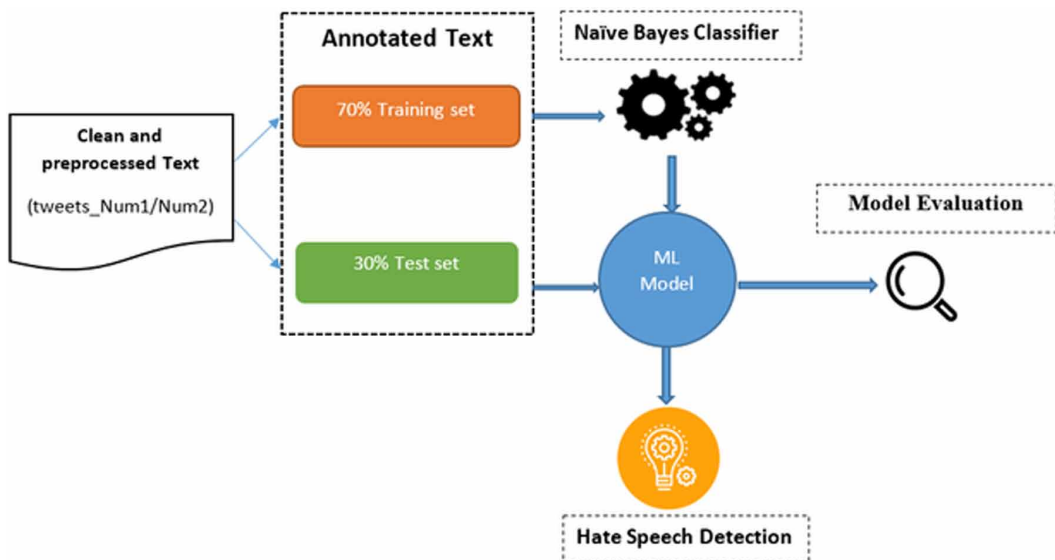


Figure 16. Checking the proportion of class in tweets_Num1

```

Console ~/
> # check that the proportion of class is similar
> prop.table(table(train_class))
train_class
      0      1      2
0.05905882 0.77505882 0.16588235
> prop.table(table(test_class))
test_class
      0      1      2
0.05473468 0.77270975 0.17255557
    
```

Figure 17. Checking the proportion of class in tweets_Num2

```

Console ~/
> # check that the proportion of class is similar
> prop.table(table(train_class))
train_class
      0      1
0.93076233 0.06923767
> prop.table(table(test_class))
test_class
      0      1
0.92775823 0.07224177
    
```

Box 1.

	Actual	
Predicted	Event	No Event
Event	a	b
No Event	c	d

of correct and incorrect predictions which are summarized with percent values and broken down by each class. We used the package “caret” to call the confusion matrix function “confusionMatrix()”.

As shown in Figure 18, the confusion matrix for tweets_Num1 clarifies that the actual class is the same as the predicted respectively 50% for the class 0 and 91.4% for the class 1 and 80.2% for the class 2.

In Figure 19, the confusion matrix for tweets_Num2 shows that the actual class is the same as the predicted respectively 94.6% for the class 0 and 72.4% for the class 1.

Table 3 and Table 4 provide the values of the associated confusion matrix measures for both datasets. For the first dataset, sensitivity assesses the ability of the generated model to correctly

Figure 18. Confusion matrix for tweets_Num1

		Actual		
		0	1	2
Predicted	0	50.0%	6.1%	4.9%
	1	39.9%	91.4%	14.9%
	2	10.1%	2.5%	80.2%

Figure 19. Confusion matrix for tweets_Num2

		Actual	
		0	1
Predicted	0	94.6%	27.6 %
	1	5.4%	72.4%

identify tweet ‘class. For example, concerning the class 0, sensitivity is the proportion of hateful tweets that got predicted correctly as hateful. It represents the percentages of the diagonal of the confusion matrix in Figure 18 that are successively 0.50, 0.91, and 0.80 for classes 0, 1 and 2. While specificity defines the metric that evaluates the ability of our model to determine the proportion of actual negative tweets, which got predicted correctly, in this case the specificity ‘values are 0.94, 0.79 and 0.97.

Similarly for the second dataset, the sensitivity value is 0.94 and the specificity value is 0.72. The recall values are the same as the sensitivity values for the two datasets. Precision is equal to the Pos Pred value, which represents the proportion of tweets which truly have class x among all those that were classified as class x. F1 combines the precision and recall measures. Indeed, the calculated values of these metrics are high especially for binary classification.

Table 5 provides overall statistics of the two datasets used in this paper and highlight the values of Accuracy and kappa indicator. For the first dataset named tweets_Num1, the value of accuracy is 87.23%, while Kappa has the value of 66.8%. The second dataset tweets_Num2, the value of accuracy is 93.06%, and Kappa has the value of 55.2%.

As shown in Figures 20 and 21 the accuracy of the two datasets are high therefore the models created will detect in a good way hate speech on social media.

Table 3. Results of conf.mat\$byClass tweets_Num1

<i>Class</i>	<i>Class: 0</i>	<i>Class:1</i>	<i>Class:2</i>
<i>Sensitivity</i>	0.5000000	0.9143665	0.8019360
<i>Specificity</i>	0.9411445	0.7908423	0.9703416
<i>Precision</i>	0.3297214	0.9369569	0.8493691
<i>Recall</i>	0.5000000	0.9143665	0.8019360
<i>F1</i>	0.3973881	0.9255239	0.8249713
<i>Prevalence</i>	0.05473468	0.77270975	0.17255557
<i>Detection Rate</i>	0.02736734	0.70653989	0.13837852
<i>Detection Prevalence</i>	0.08300141	0.75407940	0.16291918
<i>Balanced Accuracy</i>	0.7205722	0.8526044	0.8861388
<i>Pos Pred Value</i>	0.3297214	0.9369569	0.8493691
<i>Neg Pred Value</i>	0.9701555	0.7309300	0.9591711

5. CONCLUSION

To sum up, the present study aims at boosting decision making in automatic hate speech detection, which poses serious dangers for the cohesion of a democratic society. The various models and features outlined in the literature are hard to compare efficiently since the results are assessed on individual datasets that are frequently not publicly available. The lack of general mechanisms for its automatic detection makes the task difficult to handle specifically when textual content is combined with offensive language.

This work seeks to suggest both a framework and a model for detecting hate speech on social media in order to mitigate state of the art problems and to enhance the generalization of a model able to distinguish hate speech from both normal speech and offensive language. We adopted a full implementation of text mining process based on many steps including data description, data

Table 4. Results of conf.mat\$byClass in tweets_Num2

<i>Class</i>	0
<i>Sensitivity</i>	0.9457931
<i>Specificity</i>	0.7242026
<i>Precision</i>	0.9790120
<i>Recall</i>	0.9457931
<i>F1</i>	0.9621159
<i>Prevalence</i>	0.9315174
<i>Detection Rate</i>	0.8810227
<i>Detection Prevalence</i>	0.8999101
<i>Balanced Accuracy</i>	0.8349979
<i>Pos Pred Value</i>	0.9790120
<i>Neg Pred Value</i>	0.4955071

Table 5. Overall Statistics of the two datasets

<i>Dataset 'name</i>	<i>tweets_Num1</i>	<i>tweets_Num2</i>
<i>Accuracy</i>	0.8723	0.9306
<i>Kappa</i>	0.668	0.552

Figure 20. The accuracy of tweets_Num1

```

Console ~/ ↵
> # Prediction Accuracy
> conf.mat$overall['Accuracy']
Accuracy
0.8722858
    
```

Figure 21. The accuracy of tweets_Num2

```

Console ~/ ↵
> # Prediction Accuracy
> conf.mat$overall['Accuracy']
Accuracy
0.930618
    
```

exploration, data cleaning and pre-processing, data transformation, visualization, modeling and lastly evaluation/validation. We succeeded in building an accurate model based on Naïve Bayes; a robust machine learning classification algorithm, which proved its efficiency based on several scientific articles dealing with text and sentiment analysis. The proposed model was generated from two different datasets in terms of data size, type of classification task (binary or multiclass) and data sources. The performance achieved are promising, the first dataset reached the accuracy value of 87.23%, while the second attended the value of 93.06%. For future work, and in order to examine the possibility of improving the performance of our model, we plan to use deep learning.

REFERENCES

- Alam, S., & Yao, N. (2019). The impact of preprocessing steps on the accuracy of machine learning algorithms in sentiment analysis. *Computational & Mathematical Organization Theory*, 25(3), 319–335. doi:10.1007/s10588-018-9266-8
- Almatarneh, S., Gamallo, P., Pena, F. J. R., & Alexeev, A. (2019). *Supervised Classifiers to Identify Hate Speech on English and Spanish Tweets*. Springer. doi:10.1007/978-3-030-34058-2_3
- Al-Twairesh, N., Al-Khalifa, H., Al-Salman, A., & Al-Ohali, Y. (2017). AraSenTi-Tweet: A Corpus for Arabic Sentiment Analysis of Saudi Tweets. *Procedia Computer Science*, 117, 63–72. doi:10.1016/j.procs.2017.10.094
- Vidhya, A. (n.d.). Retrieved January 29, 2022, from <https://www.analyticsvidhya.com/myfeed/>
- Blaya, C. (2019). Cyberhate: A review and content analysis of intervention strategies. *Aggression and Violent Behavior*, 45, 163–172. doi:10.1016/j.avb.2018.05.006
- Boudad, N., Faizi, R., Oulad Haj Thami, R., & Chiheb, R. (2018). Sentiment analysis in Arabic: A review of the literature. *Ain Shams Engineering Journal*, 9(4), 2479–2490. doi:10.1016/j.asej.2017.04.007
- Chetty, N., & Alathur, S. (2018). Hate speech review in the context of online social networks. *Aggression and Violent Behavior*, 40, 108–118. doi:10.1016/j.avb.2018.05.003
- ConfusionMatrix function | R Documentation. (n.d.). Retrieved October 25, 2019, from <https://www.rdocumentation.org/packages/caret/versions/6.0-84/topics/confusionMatrix>
- Davidson, T., Bhattacharya, D., & Weber, I. (2019). Racial Bias in Hate Speech and Abusive Language Detection Datasets. *Proceedings of the Third Workshop on Abusive Language Online*, 25–35. doi:10.18653/v1/W19-3504
- Davidson, T., Warmesley, D., Macy, M., & Weber, I. (2017). Automated Hate Speech Detection and the Problem of Offensive Language. *Proceedings of the Eleventh International AAAI Conference on Web and Social Media (ICWSM 2017)*, 4.
- Gonçalves, P., Araújo, M., Benevenuto, F., & Cha, M. (2013). Comparing and combining sentiment analysis methods. *Proceedings of the First ACM Conference on Online Social Networks - COSN '13*, 27–38. doi:10.1145/2512938.2512951
- Hate Speech and Offensive Language—Dataset by thomasr davidson | data.world. (n.d.). Retrieved October 20, 2019, from <https://data.world/thomasr davidson/hate-speech-and-offensive-language>
- Hussein, D. M. E.-D. M. (2018). A survey on sentiment analysis challenges. *Journal of King Saud University - Engineering and Science*, 30(4), 330–338. doi:10.1016/j.jksues.2016.04.002
- Jendryke, M., & McClure, S. C. (2019). Mapping crime – Hate crimes and hate groups in the USA: A spatial analysis with gridded data. *Applied Geography (Sevenoaks, England)*, 111, 102072. doi:10.1016/j.apgeog.2019.102072
- Kang, M., & Jameson, N. J. (2018). Machine Learning: Fundamentals. In M. G. Pecht & M. Kang (Eds.), *Prognostics and Health Management of Electronics* (pp. 85–109). John Wiley and Sons Ltd., doi:10.1002/9781119515326.ch4
- Kiilu, K. K., Okeyo, G., Rimiru, R., & Ogada, K. (2018). Using Naïve Bayes Algorithm in detection of Hate Tweets. *International Journal of Scientific and Research Publications*, 8(3). Advance online publication. doi:10.29322/IJSRP.8.3.2018.p7517
- Mastrorocco, N., & Minale, L. (2018). News media and crime perceptions: Evidence from a natural experiment. *Journal of Public Economics*, 165, 230–255. doi:10.1016/j.jpubeco.2018.07.002
- Medhat, W., Hassan, A., & Korashy, H. (2014). Sentiment analysis algorithms and applications: A survey. *Ain Shams Engineering Journal*, 5(4), 1093–1113. doi:10.1016/j.asej.2014.04.011
- Ruwandika, N. D. T., & Weerasinghe, A. R. (2018). Identification of Hate Speech in Social Media. *2018 18th International Conference on Advances in ICT for Emerging Regions (ICTer)*, 273–278. doi:10.1109/ICTER.2018.8615517

Sossi Alaoui, S., Farhaoui, Y., & Aksasse, B. (2018). Classification algorithms in data mining. *International Journal of Tomography & SimulationTM*, 31(4).

Sossi Alaoui, S., Farhaoui, Y., & Aksasse, B. (2017). A Comparative Study of the Four Well-Known Classification Algorithms in Data Mining. *Advanced Information Technology, Services and Systems*, 362–373. 10.1007/978-3-319-69137-4_32

Twitter Sentiment Analysis. (n.d.). Retrieved October 20, 2019, from <https://kaggle.com/arkhoshghalb/twitter-sentiment-analysis-hatred-speech>

Safae Sossi Alaoui is a computer engineer in a Moroccan public agency. She received her PhD in 2020 from the Department of Computer Science at the Faculty of Science and Techniques in Errachidia, Moulay Ismail University, Morocco. Her research focuses on machine learning, Big data and decision making. In addition, she is a state engineer in telecommunications and information technology. She graduated from the National Institute of Posts and Telecommunications (INPT) in 2015.

Yousef Farhaoui is a Professor at the Department of Computer Science in Faculty of Sciences and Techniques, Moulay Ismail University, Morocco. He received his PhD degree in Computer Security from the University IBN Zohr. His research interest includes computer security, big data, data mining, data warehousing, data fusion, etc.

Brahim Aksasse is a Full Professor in the Department of Computer Science at the Faculty of Science and Technology Errachidia, Morocco. He graduated from Fez University in 2000. He spent three years as a Postdoctoral Fellow at the University of Bordeaux1 France (2001–2004). He is the Head of the Systems Analysis and Applied Informatics research team. His research works concern signal modelling, image filtering, image indexing and multidimensional spectral analysis.