Sentiment Analysis Using Machine Learning Algorithms

Fatma Jemai
Higher Institute of Computer Sciences of KEF
University of Jendouba
Kef, Tunsia
jemai.fatma@isikef.u-jendouba.tn

Mohamed Hayouni
Innov'com Research Laboratory
Higher School of Communications
(Sup'Com)
University of Carthage
Ariana, Tunisia
mohamed.hayouni@supcom.rnu.tn

Sahbi Baccar
CESI Graduate School of Engineering
Rouen, France
dr.sahbi.baccar@ieee.org

2021 International Wireless Communications and Mobile Computing (IWCMC) | 978-1-7281-8616-0/21/$31.00 ©2021 IEEE | DOI: 10.1109/IWCMC51323.2021.9498965

***Abstract*—This work aims at building a classifier able of predicting the polarity of a comment while using Machine Learning (ML) algorithms. Our work is essentially divided into three tasks: data extraction, processing and modelling. In order to build our model, we use the NLTK dataset. Then, we use text mining techniques to generate and process the variables. Based on a supervised probabilistic machine learning algorithm, we tended to create a classifier to classify our tweets into positive and negative sentiments then we opt for two experiments to evaluate the performance of our model. Compered to previous reported works, we achieve greater precision.**

***Index Terms*—Machine Learning (ML), NLTK, Sentiment analysis(SA)**

I. Introduction

Due to the evolution of web 1.0 to web 2.0, it is easiest for users to generate and share their thoughts, views and approaches on the Internet. This leads to an explosive growth of subjective information (opinion data) on the Web. Therefore, goal in collecting, studying and exploiting this information is increasing from which several concepts have appeared such as Sentiment Analysis (SA) which focuses on opinion mining (identification and classification) from textual data.

These informations can help people to make their decisions such as: business applications knowing the preferences of customers and improving the quality of products sold or just for marketing analysis; policy comments can help politicians clarify their political strategy; event reviews can help stakeholders reflect on their activities; public relations, etc.

More people have started to express their sentiments and opinions on the internet and on various social media. This has led to an increase in the number of user generated sentences containing information about sentiments making it impossible for humans to read and analyze all of them. Thus, automatic analysis of opinions expressed on various web platforms is increasingly becoming important for making effective decision and it is also difficult because extracting emotions is a complex activity which must understand the content and capture the hidden sentiments in the written text from which it is inevitable to study new methods.

The paper is organized as follows: section II provides the literature review. The proposed system is presented in section III. The experiment and the results are highlighted in section IV. Finally, the conclusion and future work are presented in section V.

II. Related Work

In recent years, due to the massive spread of online reviews, sentiment analysis has attracted a lot of research attention. Therefore, a voluminous studies have been devoted to this area of research.

There are authors who have discussed data preprocessing to remove data noise. For example, the work carried out by Jinyan Li and al [1] presents an experiment that compares the possibilities of using different filters in the training dataset during preprocessing to find the factors that influence the algorithms of SA. The results showed that in sentiment analysis the sentiment trending words have some influence on the outcome of the prediction. After removing the high frequency words, the accuracy of the prediction results decreases, especially for unique high frequency words of each class.

Huma Parveen and Shikha Pandey in [2] compare the result with and without considering emoticons. They observed that when preprocessing with emoticons, the results were much more accurate than preprocessing without emoticons because the tweets contained sentiments in the form of emoticons. Also, they deduced that there are a lot of neutral sentiments when removing emoticons but when preprocessing is used with emoticons, neutral tweets decrease significantly due to emoticon conversion.

Other work was done by Soumya S. and Pramod K.V. [3] working on classifying Malayalam tweets into positive and negative using different machine learning algorithms such as NB, SVM and RF. In this work four different functionalities such as BOW, TF-IDF, Unigram with Sentiwordnet and Unigram with Sentiwordnet including negation words are taken into account for the formation of the feature vectors of the input dataset. All the classifiers with the last two features showed better precision compared to other features and especially the Random Forest classifier which shows higher 95.6 % precision while considering Unigram with Sentiwordnet including negation words as a feature.

Rashedul Amin Tuhin and al. in [4], discussed sentiment analysis for the Bangla text document using two proposed methods (Naïve Bayes and the topical approach) which were applied at the article and sentence level in addition to comparative analysis of the performances between these two methods. From three experiments, the authors see that the Bayesian classifier gives less precision at the article level, while the topical model always performs better. The reason for this lower precision for Naive Bayes is quite simple. The main reason is the operation of multiplication between the elements of a set of elements. However, this is not the case for the Topical Approach as it takes the decision operation by a summation and comparison operation.

While most of the research in this area focuses on comparing the accuracy and classification performance of algorithms (ML). Poornima. A and K. Sathiya Priya in [5] compared the performance of three machine learning approaches (SVM, Multinomial Naive Bayes and logistic regression) in the classification of Twitter phrases or data. This comparison shows that the precision of the logistic regression is about 86.23 %, SVM is 85.69 % and Multinomial Naive Bayes is 83.54 %, so when the bigram model was used, logistic regression works better compared to other supervised machine learning algorithms for sentiment analysis on Twitter [5].

In [6] the main objective is the implementation of machine learning algorithms that process the textual statistical data provided by the Yelp data set (data in English) , Yelp is a review forum that provides reviews of local businesses and can categorize it as positive or negative sentiment. The authors were using Naive Bayes, Multinomial Naive Bayes, Bernoulli Naive Bayes, Logistic Regression and Linear SVC for the classification phase. Analysis of the results shows that they have succeeded in obtaining a satisfactory level of precision for a training project but are not acceptable in real world machines so they need to be improved by providing more data for the training.

Shamantha Rai B and Sweekriti M Shetty in [7] also researched Twitter opinion mining for a specific keyword, then Naive Bayes behavior is studied in comparison to SVM and Random Forest (RF). The first observation here is that increasing the number of tweets will obviously increase the execution time of the algorithms. However, the second observation is that the execution time of RF is much higher than that of SVM and NB.

In [8], the authors Meylan Wongkar and Apriandy Angdresey applied the analysis of the public's feelings of Twitter towards the presidential candidates of the Republic of Indonesia for the period 2019-2024, obtained using a crawler. A comparison of the accuracy of the Naïve Bayes method was made with other classification methods such as SVM and KNN using RapidMiner which shows that the accuracy of the Naïve Bayes method is better (80.90 %), compared to K-Nearest Neighbor (KNN) which has an accuracy rate of 75.58 % and with the lowest accuracy value of 63.99 % using SVM.

In [9], Md. Golam Sarowar et al developed an efficient and accurate sentiment analysis classifier for Bengali e-commerce sites. They created manually their own StopWords database with around 900 words in Bengali and they have been using web scraping in python to collect comments and reviews on Bangla on various sites. Then its data is tokenized using the NLTK python library and filtered using the created StopWords. Then, they converted the data to digital from the TF-IDF (Term Frequency - Inverse Document Frequency) information retrieval mechanism and finally they treated them using the K-Nearest Neighbor (KNN) with Support Vector Machine (SVM). For the comparative study, the authors were using the logistic regression, the convolutional neural network based on the PCA and the random forest due to their exceptionally remarkable characteristics during classification. The results showed that the level of efficiency of this proposed hybrid approach is three times faster than all the other considered approaches.

III. PROPOSED SYSTEM

Sentiment analysis is performed to rank sentences using machine learning algorithms. First of all, we have the Tweet collection phase, then the preprocessing, data preparation and classification phases to reach the phase of Evaluation.

Our choice fell on the python programming language because it offers high level tools and a simple syntax to use and Anaconda as a development environment because it is the right way to install machine learning tools. We will be using the Natural Language Toolkit (NLTK) guide package for all NLP tasks in this document and the open source Scikit-learn library for Machine learning.

A. Data Collection Phase

The data used in this work consists of a dataset of sample English tweets from the NLTK package. NLTK's Twitter corpus currently contains a sample of 20k (20,000 non-sentimental tweets) Tweets (named 'twitter _samples') retrieved from the Twitter Streaming API, plus another 10k which are split by sentiment into negative and positive (5,000 tweets with negative feelings, 5,000 tweets with positive feelings).

B. Preprocessing Phase

Tweets contain a lot of slang words and punctuation marks. The language also in its original form can not be processed accurately by a machine, so we have to clean up our tweets to make it easier to understand and use by a supervised machine learning algorithm.

- **Data tokenization :** this is a more popular technique which is used to break down a body of text into several sentences and each sentence into a list of constitutive words (i.e. a process of creating tokens).
- **Delete stop words :** these are the most common words in a language eg “is”, “the”, and “a” in English. They are generally not relevant when processing the language.
- **Remove URL :**remove any URL from the tweet. It includes removing URLs starting HTTP, https and also pic :\\ then replace with an empty string.

- **Remove @ mentions :** these Twitter usernames are preceded by an @ symbol, which does not provide any relevant information about the sentiment of the text.
- **Change to lowercase :** to make things easier, we convert the data to lowercase, where a body of text is converted entirely to lowercase.
- **Lemmatization :** it is an algorithm that analyzes the structure of the word and its context to convert it into a normalized form, it normalizes a word with the context of the vocabulary and the morphological analysis of a word in the text.But before running a lemmatizer, we need to determine the context of each word in our text. This is achieved by a tagging algorithm, which evaluates the relative position of a word in a sentence (this is the identification of words as nouns, verbs, adjectives, adverbs, etc.).

The Preprocessing process is as shown in the Fig.1.

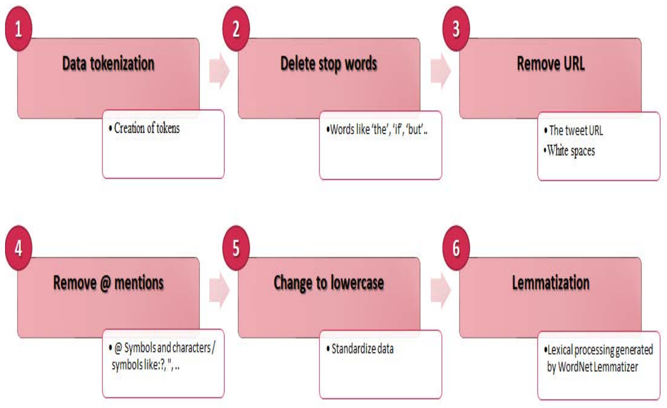


Fig. 1. Pretreatment process.

C. Data preparation

During the data preparation step, we will prepare the data for sentiment analysis by converting the tokens to Python dictionary format with words as keywords and True as values, mixed randomly. Then, we divide the mixed data into two parts, the purpose of the first part is to build the model, while the next part tests the performance of the model in a 70: 30 ratio for training and testing. Since the number of tweets is 10000, we can use the first 7000 tweets from the mixed dataset to train the model and the last 3000 to test the model.

D. Classification Phase

After the data is divided into training and test set, the machine learning algorithm can be used to learn from the training data.

The following algorithms were used:

*a) **Naïve Bayes(NB) classification (supervised classification – probabilistic)** :* Naïve Bayes classifiers are powerful and simple supervised machine learning algorithms. This classification is named Naive Bayes after Thomas Bayes, who proposed Bayes theorem to determine probability. It allows probabilistic classifications, which assign the probability of belonging to a class. It assumes that the value of a particular characteristic is independent of the value of any other characteristic given the class variable.

In the case of NB, there are three different versions of the technique NB which are often used for classification, were used Multinomial Naïve Bayes and Bernoulli Naïve Bayes.

- **Multinomial NB :** this version of NB is often applied for classification text. It is called respectively binary independence model and unigram language model which requires discrete characteristics as input data. Multinomial generally works best with larger vocabulary sizes. Here, each feature has a multinomial distribution forming a feature vector that represents the frequency of occurrence of that feature in a particular instance.
- **Bernoulli NB :** Here the features are independent binary variables representing the presence or absence rather than probabilities as in Multinomial Naive Bayes [6]. So it is also called binary independence model, ie a Bayesian network without dependencies between words and binary word characteristics and unigram language model in the literature. The Bernoulli NB works well with small vocabulary sizes. This version of NB is used where they can be multiple entities and each is assumed to be a binary valued variable. In text classification, the word occurrence vector is used for training and then for classification.

The table I below gives a comparison between Multinomial NB and Bernoulli NB:

TABLE I
A COMPARISON BETWEEN MULTINOMIAL NB AND BERNOULLI NB

Multinomial Naïve Bayes	**Bernoulli Naïve Bayes**
Applied for text classification	Applied for text classification
Called respectively binary independence model and unigram language model.	Called respectively the binary independence model and unigram language model
Usually works best with larger vocabulary sizes	Works well with small vocabulary sizes
Documents are designated by an integer term count vector in the multinomial model. Due to the multinomial distribution of each class.	The terms being present and absent have an important role in the representation of the binary vector. In other words, regardless of whether the term appears in the document, it is placed as 1 in the binary vector and otherwise zero. Frequencies of terms in documents are not captured by this model.
This model, regardless of the position of terms, can use term frequencies.	This model does not use term frequency information.
The multinomial variant would simply ignore a non-existent feature.	The Bernoulli NB classifier explicitly penalizes the non-occurrence of a feature i which is an indicator of the class y.

*b) **Support Vector Machine (SVM) (classification linéaire ou non linéaire supervisée) :*** SVM is a popular supervised machine learning algorithm proposed by Vapnik in 1992 that can be a linear or non-linear classifier. SVM is also called a large margin classifier because the algorithm finds the hyperplane with the largest margin, i.e. the greatest distance to the nearest sample points.

SVC (Support Vector Classifier) and NuSVC (Nu-Support Vector Classification) are similar methods, but except slightly different sets of parameters and have different mathematical formulations. On the other hand, LinearSVC (Linear Support Vector Classifier) another implementation of the classification of support vectors for the case of a linear kernel.

IV. EXPERIMENTATION AND RESULTS

All the experiments are carried out on an Intel (R) Core (TM) i3-6006U processor, CPU @ 2.00 GHZ 2.00 GHZ, RAM 4.00 GB. We were able to achieve a satisfactory level of accuracy in the classification of tweeter reviews through supervised learning.

Compared to [6], we achieve greater precision. The table II below gives a comparison of the results.

TABLE II
A COMPARISON BETWEEN MULTINOMIAL NB AND BERNOULLI NB

Classifiers	**Our work (accuracy %)**	**Article [6] (accuracy %)**
Naive Bayes	99.73	79.12
Multinomial NB	99.70	78.92
Bernoulli NB	99.67	73.22
Logistic_Regression	99.53	78.88
LinearSVC_classifier	99.67	75.32

The main objective of this work is to be able to teach the computer to read and understand an English sentence typed by a human and to be able to classify it as a positive or negative feeling.

This work gives a relatively interesting accuracy for a first try on the test dataset but it could be improved because our model succeeds in testing positive and negative feelings successfully but unfortunately it fails to test a more complex feeling such as sarcasm because of a sufficient amount of training data.

In this article, we will prepare a dataset of sample tweets from the NLTK package for NLP with different data cleansing methods. Once the dataset is ready for processing, we'll train a model on pre-classified tweets and use the model to classify the sample tweets into negative and positive sentiments.

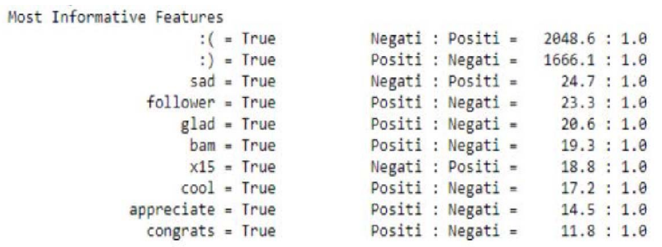
We will be using the NLTK package in Python for all of the NLP tasks in this tutorial. In the first step, we will install NLTK, download and locally store the sample tweets that we will use to train and test our model using the "nltk.download ('twitter_samples')" command run.

Once the samples have been downloaded, we are ready to begin processing the data. The first part of understanding data is to use a process called tokenization, or splitting strings into smaller parts called tokens. The basic way to divide language into tokens is to divide text based on spaces and punctuation. First of all we need to download the punkt module which helps us tokenize words and phrases. NLTK provides various useful interfaces for the tokenization of words for example word_tokenize, TreebankWordTokenizer and RegexpTokenizer. For this task, we'll be tweaking the default NLTK package method in Python for tweets.Tokenized() method.

In the second part of the processing, we remove the noise from the dataset. Noise is any part of the text that does not add meaning or information to the data and it is specific to each project. In this article, we'll use regular expressions in Python to find and remove these items. We are first searches for a substring that matches a URL starting with http:// or https://, followed by letters, numbers, or special characters. Once a pattern is matched, the. sub() method replaces it with an empty string, or ". Similarly, to remove @ mentions, we are substitutes the relevant part of text using regular expressions. We are uses the re library to search @ symbols, followed by numbers, letters, or _, and replaces them with an empty string. Then, we can remove punctuation using the library string. In addition to this, we will also remove stop words using a built-in set of stopwords in NLTK. Then, we converted to lowercase all tweets. We use the string lower () method, which converts all uppercase characters in a string to lowercase characters and returns it. This helps us to reduce the size of the data.

A sentiment analysis model that we will build would associate tweets with a positive or a negative sentiment. We will need to split our dataset into two parts. The purpose of the first part is to build the model, whereas the next part tests the performance of the model. By default, the data contains all positive tweets followed by all negative tweets in sequence. When training the model, we should provide a sample of our data that does not contain any bias. To avoid bias, we've added code to randomly arrange the data using the .shuffle() method of random.

Finally, we use supervised machine learning algorithms to build the model. We use the .train () method to train the model and the .accuracy () method to test the model on the test data. Fig.2 presents the most informative features, each row of the output shows the occurrence ratio of a token in positive and negative tagged tweets in the training dataset.



```
Most Informative Features
                     :( = True           Negati : Positi =   2048.6 : 1.0
                     :) = True           Positi : Negati =   1666.1 : 1.0
                    sad = True           Negati : Positi =     24.7 : 1.0
               follower = True           Positi : Negati =     23.3 : 1.0
                   glad = True           Positi : Negati =     20.6 : 1.0
                    bam = True           Positi : Negati =     19.3 : 1.0
                    x15 = True           Negati : Positi =     18.8 : 1.0
                   cool = True           Positi : Negati =     17.2 : 1.0
             appreciate = True           Positi : Negati =     14.5 : 1.0
               congrats = True           Positi : Negati =     11.8 : 1.0
```

Fig. 2. List of words and their probabilities.

Negati: Positi or Positi: Negati is a probability ratio of that particular word occurring is a positive or a negative sentiment respectively. The first row of data means that in all tweets containing the token :(, the ratio of negative tweets to positive tweets was 2048.6 of 1. We can see that the two most discriminating elements in the text are emoticons. words like ”sad” lead to negative feelings, while ”cool” and ”glad” are associated with positive feelings.

V. Conclusion and Future Work

Sentiment detection becomes an emerging mission that contains several challenges. The aim of this work is the study of methods and approaches which ensure the automatic classification of sentiments in the form of positive or negative polarity. Different techniques are used in this article. The latest ones used are generated from data from NLTK's Twitter corpus which currently contains a 30k sample of tweets. We preprocess the data using Tokenization, Lemmatization, removal of stop words, URLs, @ mention, punctuation and special characters then we change all tweets to lowercase. Then we prepare the data where we have converted the cleaned tokens to Python dictionary with words as keywords and True as values and split into training and test data in a ratio of 70:30. We then applied supervised machine learning algorithms to classify our tweets into positive and negative and compared them in terms of accuracies.

Future work also includes improving our model to predict sarcasm, so we need to provide a sufficient amount of training data to train it accordingly. Additionally, we only considered positive, negative, and neutral polarity for labeling tweets, but other different labels that clearly explain sentiment can also be incorporated. With the fact that a large number of tweets are generated every minute and many of them are in Arabic language, future plans include the application of the hybrid classification technique to examine its effectiveness with Arabic tweets.

References

[1] J. Li, S. Fong, Y. Zhuang, and R. Khoury, “ Hierarchical Classification in Text Mining for Sentiment Analysis,” in 2014 International Conference on Soft Computing and Machine Intelligence, september. 2014, p. 46-51, doi: 10.1109/ISCMI.2014.37.

[2] H. Parveen and S. Pandey, “Sentiment analysis on Twitter Data-set using Naive Bayes algorithm ,” in 2016 2nd International Conference on Applied and Theoretical Computing and Communication Technology (iCATccT), july. 2016, p. 416-419, doi: 10.1109/ICATCCT.2016.7912034.

[3] S. S. and P. K.v., “ Sentiment analysis of malayalam tweets using machine learning techniques ,” ICT Express, april. 2020, doi: 10.1016/j.icte.2020.04.003.

[4] R. A. Tuhin, B. K. Paul, F. Nawrine, M. Akter, and A. K. Das, “ An Automated System of Sentiment Analysis from Bangla Text using Supervised Learning Techniques ,” in 2019 IEEE 4th International Conference on Computer and Communication Systems (ICCCS), february. 2019, p. 360-364, doi: 10.1109/CCOMS.2019.8821658.

[5] A. Poornima and K. S. Priya, “ A Comparative Sentiment Analysis Of Sentence Embedding Using Machine Learning Techniques ,” in 2020 6th International Conference on Advanced Computing and Communication Systems (ICACCS), march 2020, p. 493-496, doi: 10.1109/ICACCS48705.2020.9074312.

[6] H. S and R. Ramathmika, “ Sentiment Analysis of Yelp Reviews by Machine Learning ,” in 2019 International Conference on Intelligent Computing and Control Systems (ICCS), may 2019, p. 700-704, doi: 10.1109/ICCS45141.2019.9065812.

[7] R. B. Shamantha, S. M. Shetty, and P. Rai, “Sentiment Analysis Using Machine Learning Classifiers: Evaluation of Performance ,” in 2019 IEEE 4th International Conference on Computer and Communication Systems (ICCCS), févr. 2019, p. 21-25, doi: 10.1109/CCOMS.2019.8821650.

[8] M. Wongkar and A. Angdresey, “ Sentiment Analysis Using Naive Bayes Algorithm Of The Data Crawler: Twitter ,” in 2019 Fourth International Conference on Informatics and Computing (ICIC), oct. 2019, p. 1-5, doi: 10.1109/ICIC47613.2019.8985884.

[9] Md. G. Sarowar, M. Rahman, Md. N. Yousuf Ali, and O. F. Rakib,“ An Automated Machine Learning Approach for Sentiment Classification of Bengali E-Commerce Sites ,” in 2019 IEEE 5th International Conference for Convergence in Technology (I2CT), mars 2019, p. 1-5, doi: 10.1109/I2CT45611.2019.9033741.