

Intelligent System Assignment: Sentiment Detection over labeled tweets

Delivery Date: 30 January of 2022

Student: Pablo Cabargas

Github link to code: <https://github.com/pablocabargas/IntelligentSystemAssignment.git>

Data used: <https://www.kaggle.com/c/tweet-sentiment-extraction/data>

1. Introduction

The following document exposes 2 different approaches to vectorize text snippets that are tweets which have been labeled with a positive, negative or neutral sentiment label. The dataset used consists of 31014 tweets with 9685 positive, 8782 negative and 12548 neutral tweets. For each vectorization used, a set of classification algorithms has been used and their performances have been compared, identifying the vectorization that provides the best performance in terms of accuracy.

2. Methodology

The procedure has been the following

- a) Vectorization of Bag of Words: The tweets data set has been vectorized using the method `CountVectorizer` from the library `sklearn.feature_extraction.text`. This method uses the bag of words algorithm to vectorize.
- b) Vectorization of Term frequency-inverse document frequency (TF-IDF): This vectorization method relies on the term frequency which is the frequency of the word in a text and the document frequency which is the number of different texts that contain a specific word, on this case the method `TfidfVectorizer` from the library `sklearn.feature_extraction.text`.

Once the vectorization has been made the following classification methods have been used

1. Multilayer perceptron: `MultinomialNB` from library `sklearn.naive_bayes` has been used
2. Logistic Regression: `LogisticRegression` from `sklearn.linear_model`
3. Multinomial Naive Bayes: `MultinomialNB` from `sklearn.naive_bayes`

Finally these methods are compared to analyse which one has performed better in terms of accuracy.

3.Results

3.1)Bag of Words Vectorization:

a) Top 10 common words without removing stop words and keeping them:

Most Common words without removing stop words	Most Common words removing stop words
to 4315	good 1023
the 3904	just 956
it 2439	day 915
you 2410	love 725
my 2369	happy 662
and 2028	like 612
is 1764	work 503
in 1647	today 482
for 1568	going 469
of 1366	lol 458

b) Classification performance

	Logistic Regression	Multinomial Naive Bayes	MLP
Without removing stopwords	78.9	75.7	78.36
Removing Stopwords	77.07	75.03	75.87

c) Confusion Matrix

	negative	neutral	positive
negative	2153	572	180
neutral	594	3102	483
positive	236	404	2511
	negative	neutral	positive

Predicted label

MLP confusion matrix

	negative	neutral	positive
negative	2214	524	167
neutral	585	3166	428
positive	244	398	2509
	negative	neutral	positive

Predicted label

Logistic Regression

	negative	neutral	positive
negative	1708	1038	159
neutral	235	3606	338
positive	61	724	2366
	negative	neutral	positive

Predicted label

Naive Bayes

3.2) Vectorization of Term frequency-inverse document frequency (TF-IDF)

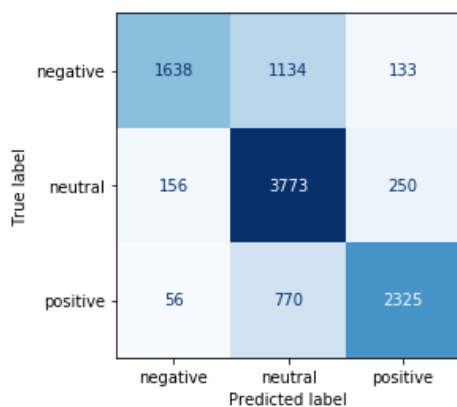
a) Top 10 highest score

Most Common words without removing stop words	Most Common words removing stop words
to 571.526135	good 507.248380
the 500.226164	love 390.717756
good 462.322082	happy 388.901422
you 447.110865	day 268.709957
it 422.019709	thanks 262.030469
happy 370.596851	miss 239.096681
my 367.702183	sad 219.677516
love 357.259020	just 213.615296
is 316.406699	great 210.886488
and 291.280952	fun 202.401850

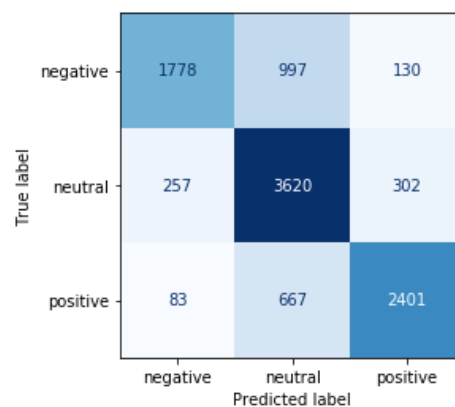
b) Classification Performance

	Logistic Regression	Multinomial Naive Bayes	MLP
Without removing stopwords	80	75.5	77.6
Removing Stopwords	76.2	75.5	77.6

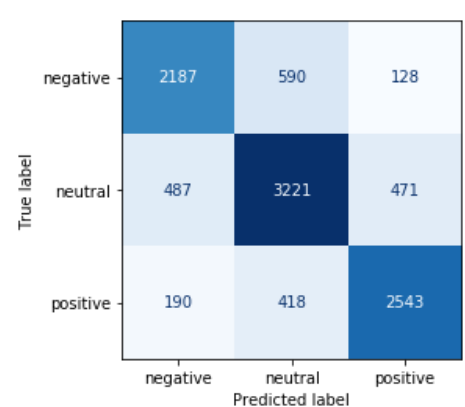
c) Confusion Matrix



Naive Bayes



Logistic Regression



4. Discussion

Regarding Bag of words vectorization , when we see the top 10 most common words on table 1, it is possible to see that when stop words are not considered, the most common words end up being the ones that most of the time do not provide much information at least semantically , with instances such as “to, the, it, is”. The situation changes dramatically when we see the results of the top 10 on table 3.1.a when stop words are considered, where we see examples such as “good, day,love , happy”, which indeed carry a more meaningful information semantically speaking.

Despite the previous, it is relevant to remind that when stop words are considered, that just impacts on the extension of the vocabulary and thus on the vector dimension that represents each of the tweets in this case. With that being said, the use of stop words generate almost the same results in terms of accuracy for each of the 3 models used as is possible to see on table 3.1.b, just logistic regression presents a small detriment but nevertheless , accuracies to identify whether a tweet is positive, negative or neutral are quite good reaching values in the range of 75.03-78.9 % depending on the model used.

Regarding TF -IDF vectorization, once again it's possible to observe this situation ,on table 3.2.a,where not influential words in terms of semantics are included in the top 10 highest tf-idf score on the case when stop-words are not considered, and in the case when they are considered the highest ranked words are meaningful words such as ‘good, love, happy’. Just as the bag of words vectorization, the performance obtained over the 3 models in the scenario of considering and not considering stop words it's almost the same, but there is a slight increase in terms of accuracy in comparison to bag of words, reaching at its highest point an accuracy of 80%.

5. Conclusion

In order to classify texts it is possible to turn them into a vector representation that suits as input of traditional classification methods, as it was possible to see , using the vectorization of bag of words and the tf-idf , each text (in this case tweets) were mapped to vectors of length of the vocabulary produced, one where stop-words were considered and one were not. When considering stop-words, it is possible to see that most repeated or highest ranked words are more meaningful in terms of semantics in comparison when not considering stop-words. Although the latter does not influence in this case that much on the performance of classification methods, this could be due to the fact that tweets use quite particular language distortions thus it would be considerably better to create a particular list of stop-words for tweets. As a next step, it would be interesting to try out vectorization methods that take into consideration stronger relations of the content of the tweets, using concurrent networks.