SEXIST TWEETS CLASSIFICATION
Machine learning for natural language processing 2022

Prisca BAHI
ENSAE, MS Data Science
prisca.bahi@ensae.fr

Matteo CUVELIER
ENSAE, Cycle Ingénieur
matteo.cuvelier@ensae.fr

ABSTRACT

We address the problem of online sexism by building a model capable of detecting in sexist comments. Our approach is as follows: (1) to compare three types of classification models; (2) to compare two multi-label models using LSTM.

I. Problem Framing

A. Context and motivation

Through social media, we can communicate quickly and easily, sharing our experiences, opinions and beliefs. Our conversations and comments can be narrowly targeted or widely distributed to the point of going viral. They are also widely used by abusers who "hide" behind the fact that they are anonymous. Sexism is very common in social media and makes the boundaries of freedom tighter for feminist and female users.

B. Dataset

We have at our disposal a dataset of tweets from Twitter. Each of these tweets have been labeled by human raters. We use the *sexism_data* dataset which has 13631 tweets and 6 columns that are: id, dataset, text, toxicity, sexist, of_id. All tweets are in english

II. Experiments Protocol

A. Data Cleaning

A tweet can be composed of url, hashtag, emoticons and punctuations without forgetting a diversity in terms of character size (upper/lower case). To clean up our data, we decided to standardize our sentences in lower case and to remove urls, hashtags and emoji but also abreviations and stopwords.

To describe the dataset we computed descriptive variables such as the number of capital letters, of emojis...

B. Tokenization and embeddings

We have tokenized the dataset with the **TweetTokenizer** that is well adapted to our study. It results lists of tokens.

We used two types of embeddings. The first one is proper to the Naive Bayes Classifier (the **TFIDFVectorizer**), it gives the tf-idf coefficient for each word t in a document d (tweet), ie : $\log\left(1 + f(t,d) \times (\log \frac{n}{f_D(t)} + 1\right)$, $f(t,d)$ is the frequency of the word d in tweet d and $f_D(t)$ is the number of tweets containing the term t. The second type of embeddings is based on the **word2vec** techniques. Thanks to a ML task, each word obtains a weight (embedding) which is of dimension 300 for the pretrained weights that we selected (fasttext). To have the embeddings over a whole tweet we average the embeddings of all the tokens of the considered tweet.

C. Classification algorithms

To find the best classification algorithm for our topic we decided to implement several different models which are the following ones.

1) Probabilist model:

First, the **Naive Bayes Classifier** which is based on the frequence of appearance of words in the tweet regarding the frequence in the whole dataset. Words specific to sexist tweets then are markers of sexism.

2) Classic ML models:

Second, the **Support Vector Machine** (SVM) puts the data in a high dimension space by doing operations on the data (power, combination...) and then builds frontier between the different classes.

Third, the **K-Nearest Neighbors** (KNN) computes the euclidian distance between the points of the database and the point to be predicted, select the closest points and take the majoritary category to predict (if 5 sexist tweets and 2 non sexists in the closests then it's classified sexist).

Fourth we have the **Random Forest** that consists in a large set of Decision Trees trained with a random part of the dataset. This is an ensemble method in the sense that the prediction will be the one of the voice of the crowd. Each tree will predict the category (sexist or not) and the majoritary category will be the final prediction.

3) Deep learning models:

Fifth, we have the **FastText Supervised Classifier** that is a 1 hidden layer neural network.

Sixth we have the **LSTM** which is a simple neural network with 2 hidden layers that does classification (output of size 2 for the 2 categories of the binary classification).

D. Problem solving strategy

We now have a set of identified potential classifiers and embeddings for our data and problem. They cover a wide set of techniques. We lead the study as follows : we computed the embeddings then for each algorithms we trained the model and evaluated its performance with several metrics. After these two steps we compared the models and conclude on their efficiency regarding our goal : predicting the sexist character of a tweet.

III. Result

A. Results of the data visualization

The data visualization teaches several lessons. First and simple, we have an unbalanced dataset (Appendix A.), 11822 tweets are non sexist and 1809 are sexists (13%). We note that the average tweet is 85 characters long, 15 words long and 1 sentence long. That in average it contains 8 capital letters, 4 punctuation signs, 2 emojis and 0,5 hashtags and 1,5% of tweets are fully in upper case (Appendix B.). To this it's interesting to add that the length distributions (words, characters and sentences) are almost uniform while for capital letters, puctuation, emojis and hashtags the majority of the distribution is close to the little values, respectively for each variable (Appendix C.).

From the graphs in Appendices D. and E. we can note that the length is slightly different between sexists and non sexists tweets (they seem to be a bit longer in average). Also we see that sexists tweets use less capital letters and punctuation (and slightly more emojis).

Finally, with the WordClouds in appendices F. and G., we see that in the sexists tweets some words come more often like the followings : man, men, woman, women, hate, never, football, shit... This is really representative of the sexist content that can be found in those tweets. On the other way, non sexist tweets only show common usage words like the followings : people love, good...

B. Results of the classification algorithms

Please note that all the confusion matrices are available in the python notebook attached. Most of the interpretations under use those matrices.

The interpretations and comments also rely a lot on the metrics computed and available in the comparative table of Section IV.A.

1) Naive Bayes Classifier: The Naive Bayes Classifier proposes a good performance and has a low complexity. We can see the characteristics in the following table. The model still is really imprecise and over the sexist predictions only around 55% really are. Also, around 30% of the truly sexist tweets are not predicted as sexists.

2) SVM, KNN, Random Forest: For those three algorithms we miss most of the sexist tweets (∼60-80% of them are misclassified) and among the sexist predictions we find between 30% and 50% of false positive (non sexist classified as sexists). Theses models are not adapted. We see this in the comparative table that contains the rate of false non sexists predictions over the total number of sexists tweets and with the rate of false sexist predictions over the total number of sexist predictions.

3) FastText Supervised Classifier: This classifier is the second interesting model, its f1-score is close to the Naive Bayes' one. Comparatively the FastText model is better in term of Precision and the Naive Bayes in term of Recall.

4) LSTM: . LSTM also has a close F1-Score to Naive Bayes (our best estimator) and close performances to it in all metrics, being a bit less efficient.

IV. Discussion and conclusion

A. Comparative table

The 2 last lines of the table are rate of false non sexists predictions over the total number of sexists tweets and the rate of false sexist predictions over the total number of sexist predictions, respectively FNSP/S and FSP/SP.

Algorithm	Bayes	SVM	KNN	RForest	FastText	LSTM
Accuracy	0.889	0.888	0.864	0.874	0.895	0.873
Precision	0.556	0.678	0.516	0.631	0.641	0.505
Recall	0.689	0.366	0.294	0.219	0.548	0.664
F1-Score	0.616	0.476	0.375	0.325	0.591	0.574
Train time	0.006s	6.541s	0.004s	18.79s	0.272s	12.43s
FNSP/S	31,1%	63,4%	70,6%	78,1%	45,2%	33,6%
FSP/SP	44,4%	32,2%	48,4%	36,9%	35,9%	49,5%
Complexity	nv	ns^2	$sn\log(n)$	$n_t n\log(n)$ $\times sd$		

With n the number of rows, v is the number of words in the vocabulary, s the size of fasttext's embeddings, n_t the number of trees in the random forest and d the depth of the trees in random forest.

B. Discussion

After all this study we identify two types of models that are interesting : the simple probabilist model of the Naive Bayes Classifier and the deep learning models with FastText embeddings.

The first is really interesting because its computational complexity is really low (number of rows times size of the vocabulary). It's computation is thus extremely fast and it obtains the best performances.

The deep learning models obtain interesting performances but it would be interesting to build our own neural network structure so that it is ad-hoc to our problem. This would require knowledge that we don't have but would be an interesting way of continuing the project.

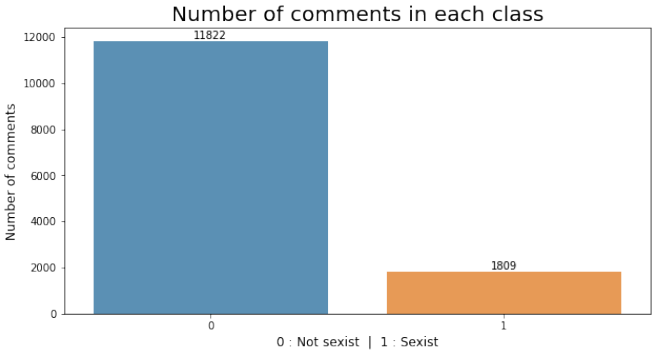
C. Conclusion

Finally, we see that the most efficient solution to predict the sexist character of a tweet is to use the Naive Bayes Classifier. It is based on a count-based embedding of words.

This is interesting because that means that the frequency of appearance of some words is the key to detect the sexist character of a tweet.

APPENDIX

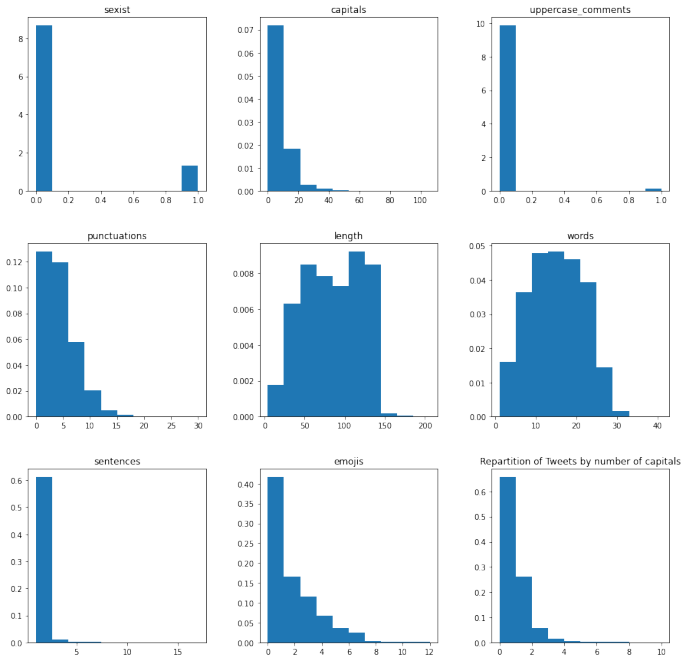
A. Repartition of tweets (sexist and non sexist)



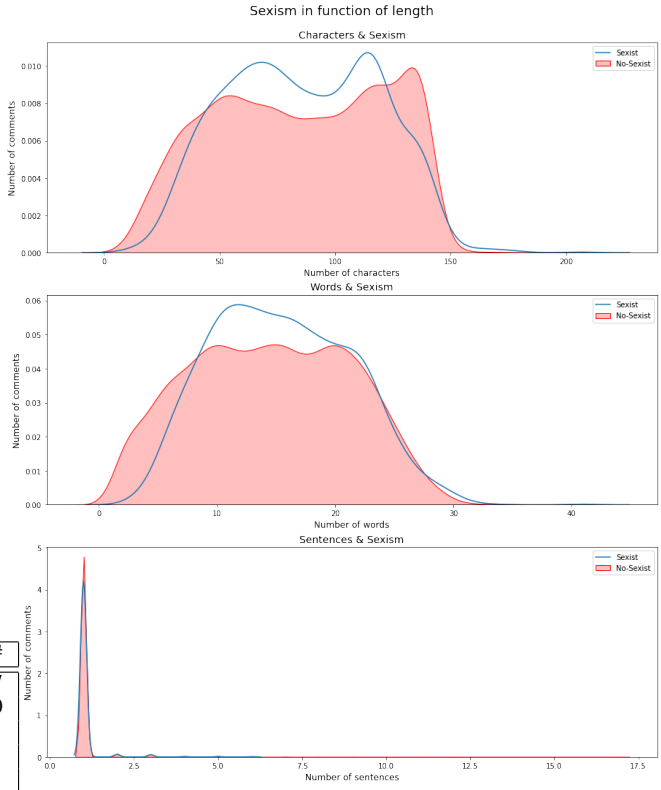
B. Descriptive statistics table

	NbUp	AllUp	NbPn	Len	NbWd	NbSt	NbEm	Nb#
mean	7.678	0.014	3.92	85.2	14.6	1.08	1.85	0.47
std	7.642	0.118	3.06	36.1	6.62	0.51	1.75	0.80
min	0	0	0	4	1	1	0	0
25%	2	0	2	55	9	1	0	0
50%	7	0	3	86	15	1	1	0
75%	10	0	6	117	20	1	3	1
max	106	1	30	206	41	17	12	10

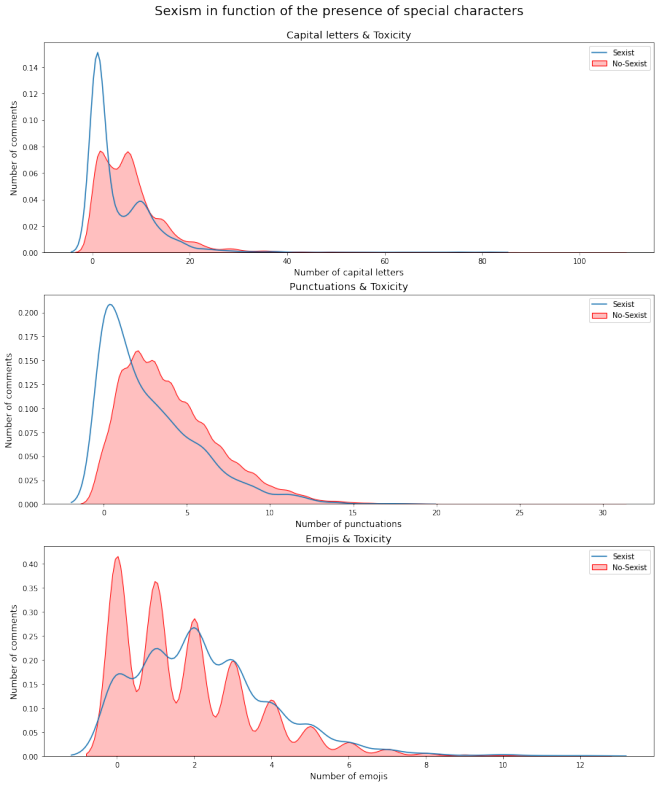
C. Histograms of the descriptive variables



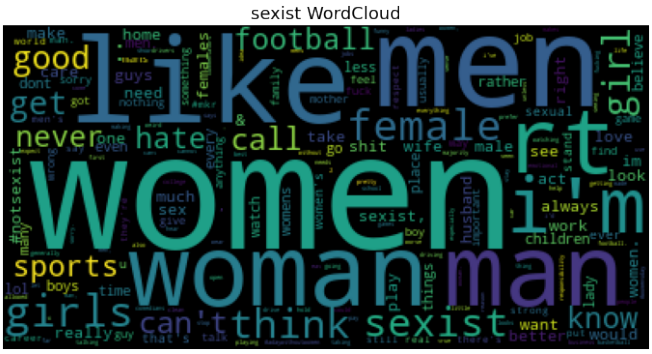
D. Distribution of tweets in function of length



E. Distribution of tweets in function of special characters



F. Sexist Tweets WordCloud



G. Non Sexist Tweets WordCloud