# *Tweets: sentiment analysis, sentiment classification*
# Machine Learning for Natural Language Processing 2022

**Astruc Guillaume**
3A DSSA
guillaume.astruc@ensae.fr

**Peltier Guillaume**
3A DSSA
guillaume.peltier@ensae.fr

## 1 Problem Framing

The purpose of this project is to predict the sentiment of a tweet thanks to its textual content. We were interested in the database, *Tweets emotions*, that we retrieved from kaggle. We are in the context of a supervised problem, a single label classification problem, where the different classes correspond to the possible sentiments of a tweets (anger, fun, sadness ...). We can also approach the problem in the form of sentiment analysis (positive, negative, neutral). Our database contains 40 000 tweets, and 13 sentiments.

## 2 Descriptive analysis and observations

### 2.1 Target variable : sentiment

The classes are strongly unbalanced: the most frequent sentiments are *neutral* and *worry* and they represent 43% of the tweets. The least frequent sentiment, *anger*, represents 0.28% of the tweets (Fig.1,2). It is to be expected that these very infrequent sentiment are very difficult to predict. It may therefore be relevant to consider a subset of feelings for classification. By keeping the 8 most frequent sentiments we keep 92% of the base.

### 2.2 Textual variable : tweets's content

To clean this variable (to make it usable by a model), we first checked if all the tweets were in english. Then, we lowered the text, removed links, numbers, tagged and hashtag symbols, emojis, repeated characters, stop words, punctuations, words with less than two characters, we normalized white spaces, corrected spelling mistakes, relaxed the contractions, lemmatized, and tokenized. This part took us a lot of time because depending on the library used, the order of execution of the different functions, the cleaned text didn't mean anything anymore (too many words were removed). We had to do a lot of random tests to make sure of the quality of the process (essential to have good results).

The average length of the tweets before cleaning is 73 words, while after cleaning it is 43 words. We have calculated word clouds by sentiment, and we observe that they do not allow to easily distinguish the different sentiment (there is not really specific words to sentiment, except for a few keywords) (Fig.3,4). This is interesting because we can therefore anticipate that a bag of words might not work well. Since the sentiments seem difficult to differentiate, we can anticipate that this task of classification will be difficult. Moreover, the zipf law computed on the tweets cleaned shows that 65% of the vocabulary (unique) words represent 95% of the words in the corpus (Fig.5). This means that we have a lot of different words, the words don't repeat well, and this can potentially mean that it will be difficult to identify patterns by feeling.

After cleaning the dataset, we have retained 99.5% of the dataset. By keeping only the tweets represented by the 8 most frequent sentiments, we keep 92% of the dataset.

## 3 Experiments Protocol

Our task is to classify efficiently our sentiments : either by considering the 8 most frequent sentiments, or by considering the types of feelings: positive or negative, setting aside neutral.

The challenges of our project are to :

- Manage unbalanced classes.
- Find embedding methods and efficient models for our task.

To do so, we first try a **classification considering the 8 most frequent sentiments**. We will test two types of models:

- **Baseline model:** we will test a random forest with different embedding techniques: bag of word, and Tf-Idf.
- **Deep learning model :** Fasttext embbedings + LSTM model

In a second step, we will focus on pre-trained model : **DistilBert model for sentiment analysis from HuggingFace.** First we will focus on the **task of sentiment analysis** by classifying positive or negative sentiment, and then we will try to **fine-tune this model for our 8 sentiments classification task**.

# 4 Results

## 4.1 Classification of sentiments

### 4.1.1 Baseline model

The random forest with bag of words and td-idf gives approximately the same performance : an accuracy of 37%. The best predicted classes are happiness, love and worry, with a recall of 45%. This is quite logical given their wordcloud, which is rather distinctive compared to the other sentiments (Fig.6).

### 4.1.2 Deep learning model

The LSTM with Fastext embeddings also gives an accuracy of 37%. The training curves show that we do not overfit (the model was designed to limit this, since in the first versions of our LSTMs, we had big problems with overfitting). The confusion matrix shows us that the classes are much less well predicted than by the random forest: fewer classes are predicted, and the recalls are much lower. The random forest is therefore more efficient: this is quite surprising, but fits well with our hypothesis that this classification task would be difficult given the nature of the texts (Fig.7,8).

## 4.2 DistilBert Model

### 4.2.1 Sentiment analysis task

Here, the task is a simple classification of positive and negative feelings. The classes are balanced. The pre-trained HuggingFace sentiment analysis model gives an accuracy of 70 %. At first glance, for a binary classification, this accuracy is not incredible, especially considering that this kind of model is supposed to be very powerful. However, as we have seen in our previous models, this task on our tweets is very difficult, so we can be satisfied with this accuracy. Here, the pre-train model DistilBert is all the more impressive here, as this accuracy is obtained without the model having been trained on our data. This average accuracy of a supposedly very powerful model, for a very simple task, confirms us in the idea that the task of our project is difficult. Moreover this model ran very quickly, mostly thanks to skipping training phase.

### 4.2.2 Fine-tuning for our sentiments classification task

The model we used was the pipeline *sentiment-analysis* that is composed of a *DistilBert* model and a classification head. We get an accuracy of 39%, only slightly better than for the random forest. The model is trained on 20 epochs, and the loss and accuracy traces show that the model learns well, does not overfit, and could potentially perform better by training it on more epochs. The confusion matrix is interesting since we see that the model has a real tendency to confuse some classes: hapiness tweets are often predicted as neutral, surprise tweets are often predicted as hapiness (Fig.9,10).

# 5 Discussion/Conclusion

Our analyses on wordclouds, Zipf's law, and the results of our models have confirmed that this classification task is particularly difficult. Indeed, we find that the classes of sentiments are not relevant (some classes are too similar). We could have tried to make a k-means to see the tweets that group together, and potentially propose new classes. We could also have studied the proximity between the classes, to potentially group some. The difficulty also comes from the quality of the text to be processed: tweets are difficult formats to make usable by a model. However, we have a relatively satisfactory (binary) sentiment classification model with the Distil-Bert pre-trained model, which shows the power of pre-trained models in some tasks. In addition, we also believe that one way to improve performance is to do data augmentation, since our classes are quite unbalanced. We have started to perform data augmentation by back translation, but due to the computational complexity of this task, we unfortunately did not have the time to train a model on it. Nevertheless, you can find the code of the data augmentation.

## References


1. https://www.kaggle.com/code/ragnisah/text-data-cleaning-tweets-analysis/notebook

2. https://medium.com/mlearning-ai/transfer-learning-example-using-keras-and-distilbert-with-code-e6e725f1fc2d


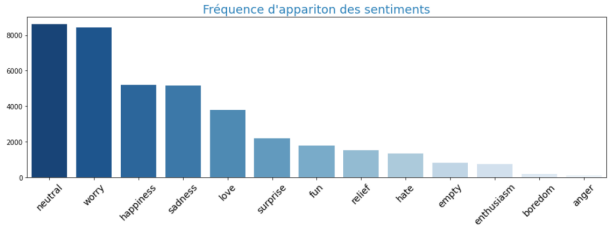


Figure 1: Frequency of feelings

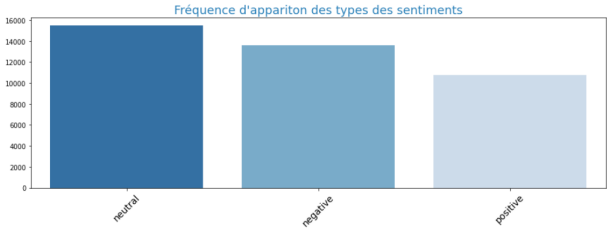


Figure 2: Frequency of positive, negative and neutral sentiments


Figure 3: Word Cloud : Love

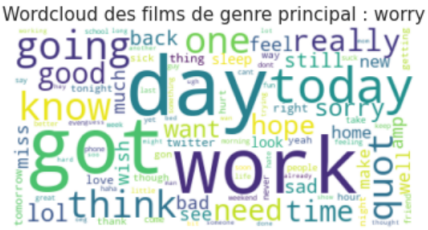


Figure 4: Word Cloud : Worry

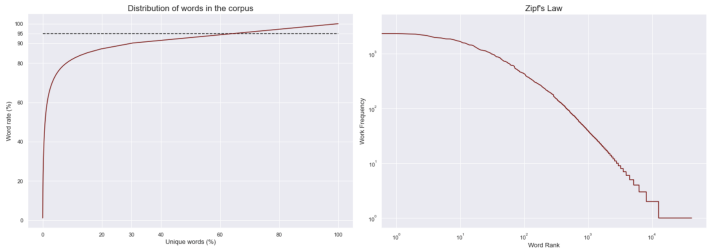


Figure 5: Zipf Law

Figure 6: Random Forest + Tf-Idf : Confusion Matrix

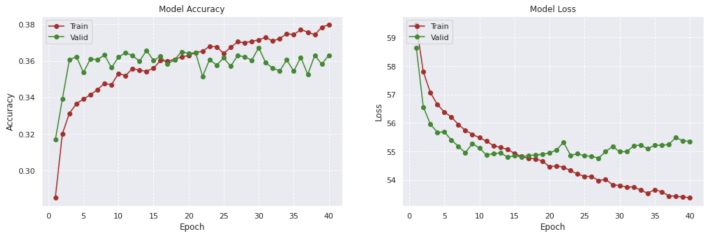


Figure 7: LSTM + Fastext: loss and accuracy plot

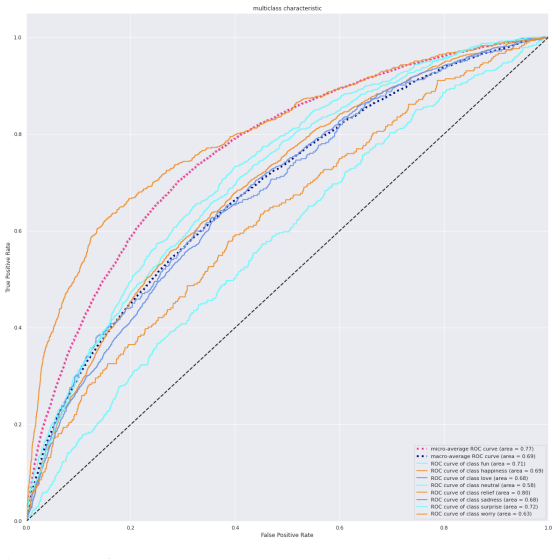


Figure 8: LSTM + Fastext: ROC plot

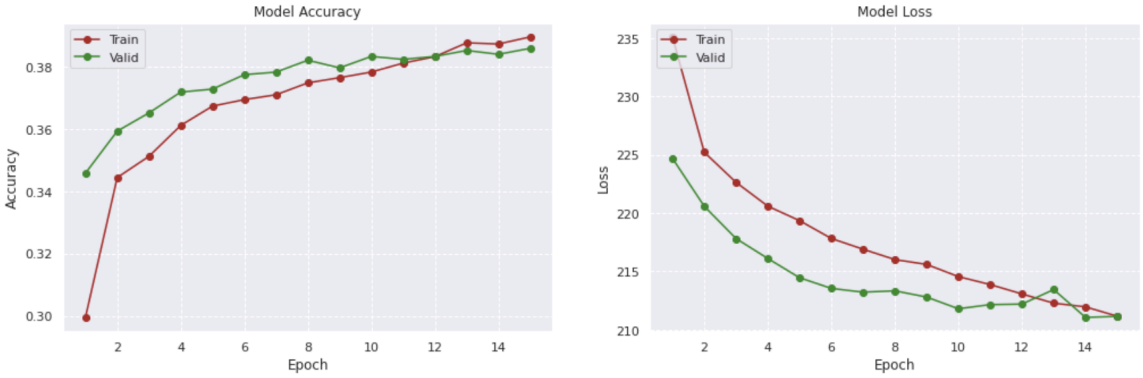


Figure 9: Training curves for DistilBert

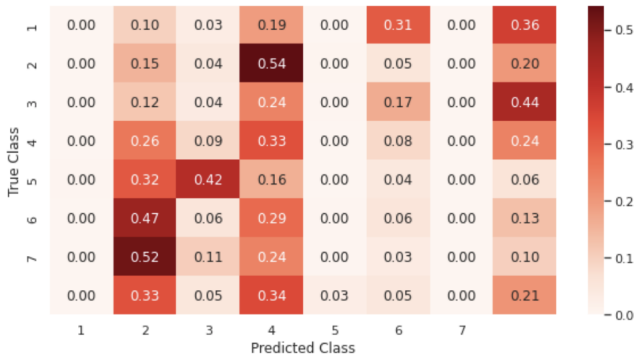


Figure 10: Confusion matrix for DistilBert