Tweets Toxicity Analysis

Text Analytics Project

Author: *Bombino Biancamaria, Fabbri Lucia, Mastrorilli Alessandro, Ricci Davide, Sarbach-Pulicani Vincent*

Professor:	*Lucia Passaro*
Master's degree:	Data Science & Business Informatics — Informatica umanistica
Date:	22 gennaio 2023

Indice

Introduction

Nowadays, social networks play a key role in people's daily lives, that share their experiences online, especially personal ones. The social media considered for the following analysis is *Twitter*: this is a "micro- blogging" system in which people are free to write and share with their friend short texts called 'tweets', free to receive and to comment what the others says in a completely unbounded space.
The aim of this project is to analyze a vaste number of tweets, trying to understand the major topics within the text and their type (negative or positive). In particular, the project will focus and will give more emphasis to the negative comments, which aim is to offend the integrity of a single person from different points of view.
In other words, the main purpose of the project is to highlight the different categories of "toxicity" that recur most frequently within the tweets, with the assumption that the tweet under analysis is targeted as "toxic". By doing this, it will be possible to understand the most specific type of discrimination that people interface with. The work has been carried out and organized as follows:

- **1. Data Understanding & Preparation**: preliminary phase of analysis and understand of the whole working dataset
- **2. Topic Modelling**: detection phase of the toxicity categories
- **3. Classification**: labelling phase of the different tweets
- **4. BERT**: analysis on toxicity using the Bert Transformer
- **5. Advanced Topics**: introduction to the advanced topics in order to find the typical tweets emotions

1. Data Understanding and Preparation

1.1 Data Understanding

The dataset used for developing the study is called *Toxic Tweets Dataset* and it describes a collection of 54313 tweets, put together as a combination of several other datasets with the aim of achieving a balanced set. For further analysis, the whole dataset is available through the following link. This dataset is mainly made up of two features:

- **Toxicity**: it is a binary variable, which takes value 0 if a tweet is *"not toxic"*, value 1 if a tweet is *"toxic"*. As can be seen from the countplot shown below in Figure 1, 30387 of the tweets are labelled as *"not toxic"*, while 23923 are labelled as *"toxic"*. Thus, the data appear to be fairly balanced.
- **Tweet**: it is the variable that defines the comments (= tweets) posted on the social media platform by the users. These sentences are short (max 280 characters) and therefore in the training phase more data will be added or integrated for sure.

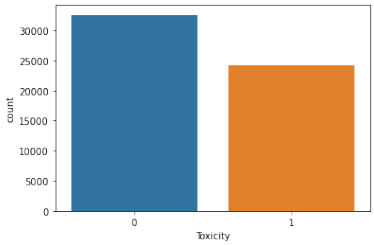

Figura 1: Countplot Toxicity

Since the largest amount of tweets are positive comments, it was decided to analyze the frequency of words in the data as shown in Figure 2 e Figure 3: the first one describes the frequency of a single word among all the tweets; the second one describes the dispersion and the position of a word within the single tweet.

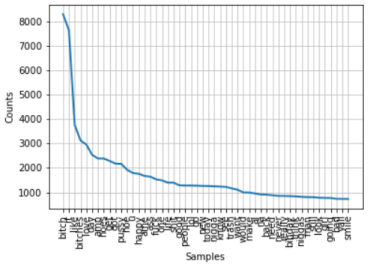

Figura 2: Words Total Frequency

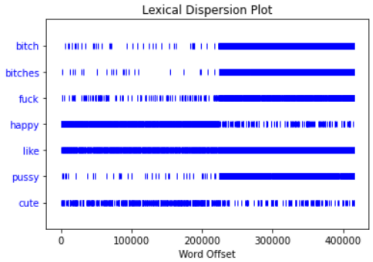

Figura 3: Words Total Dispersion

As can be seen by looking at the two graphs, the word that recurs with the most frequency among the tweets is *"bitch"* and it mainly appears at the end of our corpus. This result can suggests that the most used words on this social media is for sure related to a negative context. The lexical dispersion graph highlights how the different type of words are placed in the text. In fact, more that half of the present words are typically associated to a negative topic and they all recur mainly in the final part of the corpus. On the other side, words that refer to a positive topic are placed mostly at the beginning of the corpus. Finally, neutral words such as *"like"* are consistently present in all tweets.

1.2 Data Preparation

This section analyzes all the methodologies applied to process the data, in order to improve the efficiency of the models build and applied in this project. The processes performed at this stage are depicted in the following Figure 4.

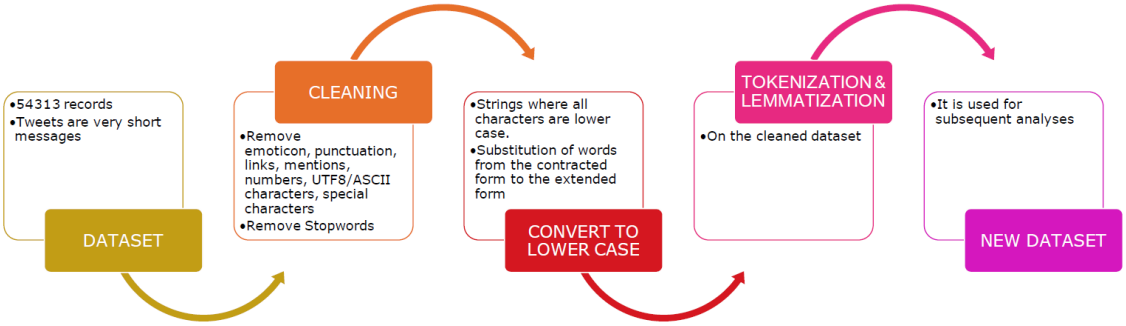

Figura 4: Data Preparation Steps

Starting from the original dataset, a cleanup was performed on the original tweets in order to obtain a cleaner and a more understandable dataset. Punctuation, emoticons, numbers, special characters, references, urls and any other part of speech that will be irrelevant for future analysis and for the task in general have been completely removed from the text. The so pre-processed text was next normalized by converting all characters to lower case, removing stop-words and replacing words in contracted form with their extended form. Finally, the text was split in to sentences through *tokenization* and then split into single words through *lemmatization*. Lemmatization was preferred to stemming because this last one doesn't provide a check on the root produced and so it could generates meaningless words.

2. Topic Modeling

Topic modeling consists of a probabilistic approach to a corpus of documents by analysing the occurrences of words of a given vocabulary within this same corpus, in this case it has been used the LDA model (Latent Dirichlet Alloca-

tion). The use of this method for the project makes sense in order to determine classes of toxicity in our particular dataset. The fact that topic modeling is an unsupervised machine learning technique therefore leads to an empirical interpretation of the results obtained. The display of those results can be done in multiple ways. For this work, it has been chosen to use wordclouds, tables and the notebook from the *pyLDAvis* library that introduces the idea of relevance[1]. Furthermore, it has been used the *gensim* library for our LDA model because it's compatible with *pyLDAvis* and allows to have topic coherence calculation. Several questions rose up when starting the analysis:

- Which type of data to use, lemmatized or raw texts ?
- On which level shall we apply our model, on toxics tweets or all of them ?

2.1 Which dataset should be used ?

In order to perform well, the idea was to test out all the alternatives by playing with the parameters such as the number of k topics or the number of passes of our machine learning model. In this case, 8 topics and 20 passes have been sticked. It's quite difficult to validate a LDA model in an intrinsic way. The "coherence rate" exists but it's not enough to be sure that our model is good, that's why it was necessary to validate it also in an extrinsic way with an empirical aspect : does this topic generated makes sense ? That's why a bag of words approach displayed in wordclouds is important to visualise correctly our topics.

The codes have been divided in two distinct parts. On one hand, there's a module including the code and classes to perform basic text processing, topic modeling and entropy calculation[2] if needed. On the other hand, there's the main jupyter notebook with the analysis. Choosing between a lemmatized dataset or a raw one depends on what we intend to do. The two parts can't be compared so easily by doing top topics comparison using their coherences and most frequent words.

Topic 0	**Topic 1**	**Topic 2**	**Topic 3**	**Topic 4**	**Topic 5**	**Topic 5**
Topic coherence	-3.120201	-4.972535	-10.772315	-11.196076	-13.122129	-13.328698
Word 1	bitch	rt	trash	pussy	bout	new
Word 2	not	hoe	white	yo	nigger	cuz
Word 3	be	nigga	lmao	eat	ill	side
Word 4	get	amp	gotta	cunt	twitter	year
Word 5	like	girl	damn	fat	head	walk
Word 6	fuck	yall	ghetto	shes	night	happy
Word 7	do	think	black	around	first	two
Word 8	as	never	fag	yeah	female	ama
Word 9	shit	take	ho	wear	tonight	high
Word 10	go	still	car	trust	nicca	nobody

Tabella 1: Topic 8 topics coherence and MFW for *lemmatized* corpus

[1] Sievert, Carson and Shirley, Kenneth, "LDAvis: A method for visualizing and interpreting topics.", in : *Proceedings of the workshop on interactive language learning, visualization, and interfaces*, 2014, p. 63-70.

[2] The entropy calculation comes from another project, could have make sense if we wanted to go deeper but it was not very suitable in our case.

Topic 0	Topic 1	Topic 2	Topic 3	Topic 4	Topic 5	Topic 5
Topic coherence	-3.198688	-5.054039	-6.150637	-11.996218	-14.607491	-16.102717
Word 1	bitch	bitches	not	fucking	stop	talking
Word 2	rt	niggas	pussy	love	take	niggah
Word 3	am	trash	do	lmao	ghetto	nigger
Word 4	like	good	can	dick	gone	yeah
Word 5	hoes	really	want	cunt	ho	trust
Word 6	hoe	people	ya	cuz	play	stay
Word 7	ass	see	ame	shes	something	find
Word 8	get	white	even	tonight	ama	nobody
Word 9	shit	wanna	ill	hope	looking	haha
Word 10	nigga	damn	fat	watch	ah	faggots

Tabella 2: Topic 8 topics coherence and MFW for *raw* corpus

From this observation, some elements have been deduced. First, it appears that working on the raw corpus seems to be more efficient in this case. It has been clearly identified a cluster of 3 topics close in their coherence rate in the raw corpus (2). Moreover, the sixth topic found in the lemmatized corpus (1) seems to have gentle words included in its most frequent words list. Now that the dataset has been chosen, let's deal with the issue of what types of tweets to analyze.

2.2 Interpretation of the determined topics

Once the dataset was built, the idea was to analyse the cleaned tweets considered as toxic in order to determine the topics. In order to comprehend this process, it's important to remind a fact. This dataset has been selected in a biased way because 56000 tweets is not much for tweets, so all types of toxicity are not necessarily represented in the corpus. To visualize it, let's take a look at the topics found on the non toxic dataset.

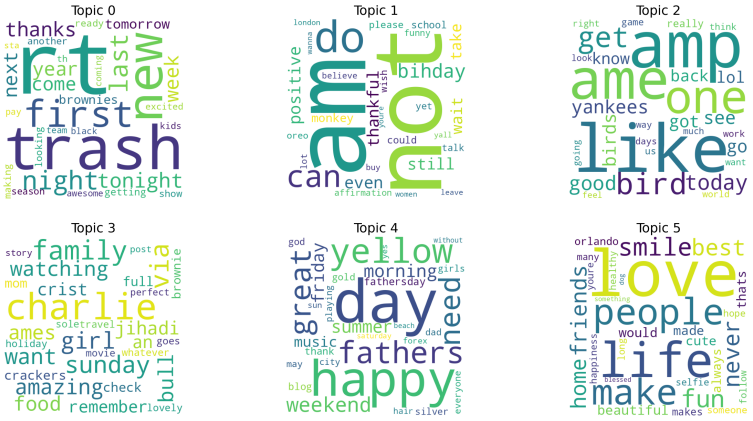

Figura 5: Top 8 topics in the *cleaned* gentle dataset

The figure 5 shows that if some topics seem to be general, the notion of family and especially the father's day appears to be a subject often mentioned in our specific corpus of tweets. Another important thing is that some toxic words ("trash" in topic 0) still remain in the results, meaning that the first classification maybe wasn't perfect.

The figure 6 in the following page allows to understand the type of toxicity present in this specific dataset. Some bad words are present in many topics because they can be used in different contexts, for example "bitch". According to those topics, an empirical classification of the types of toxicity can be done as follows:

- **Cyberbullying** : topics 0, 1 and 5

- **Racism** : topics 2 and 4
- **Misogyny** : topic 3

Figura 6: Top 8 topics in the *cleaned* toxic dataset

The next step would have been the attribution of toxicity type on every tweets considered as toxic. While identifying the most likely subject of a single document is possible with the *gensim* library, the value of this approach is uncertain. The main problem is that topic modeling works with document-term relation probabilities. It is therefore very likely that there are categorisation errors, since tweets are small and relatively short texts. Because if a tweet contains the words "bitch, niggah", it may be categorised as sexism or racism, but that does not necessarily mean it is the reality. It must be the case to understand that topic modeling is useful for two main reasons: first, by determining the global subjects of a corpus of documents, that's this project case; then, by determining the most relevant topic of a document in a corpus, pertinent process in the case of long texts.

3. Simple Classifiers

In this classification phase has been used and tested some of the most traditional supervised learning algorithms, such as: Decision Tree, Naive Bayes (Multinomial), KNearestNeighbor and Linear SVC. In order to correctly apply them, tweets text have been pre-processed by applying the following pipeline:

- **Count Vectorizer**: it's useful for converting text into vector form and then transform it into a term-document matrix. Within the structure and during all the experiments, only unigrams and bigrams was considered and the minimum term frequency taken into account was 5.
- **TfidfTransformer**: it transforma the matrix into a normalized representation of TF/IDF.
- **SelectKBest**: it's useful for performing feature selection based on the chi-square test.

All the algorithms just mentioned will be examined splitting the dataset with a proportion 70-30, thus 70% training set and 30% test set.

3.1 Experiments without hyperparameters's tuning

In this first phase, all the classifiers has been used without tuning the hyperparameters but only passing their default parameters. Different values of k has been passed to the class SelectKBest in order to find the best value leading to the best performance, using as fewer features as possible. The results obtained are listed in the tables below, labelling with class 0 a nontoxic comment and with class 1 a toxic comment:

SVC	Precision	Recall	F1-Score	Support
Class 0	0,91	0,96	0,93	8713
Class 1	0,95	0,88	0,91	6940
macro avg	0,93	0,92	0,92	15653

Naive Bayes	Precision	Recall	F1-Score	Support
Class 0	0,92	0,89	0,90	8713
Class 1	0,87	0,90	0,88	6940
macro avg	0,89	0,90	0,89	15653

Tabella 3: Results obtained by applying SVC and Naive Bayes

KNN	Precision	Recall	F1-Score	Support
Class 0	0,86	0,97	0,91	8713
Class 1	0,95	0,81	0,87	6940
macro avg	0,91	0,89	0,89	15653

Decision Tree	Precision	Recall	F1-Score	Support
Class 0	0,89	0,97	0,93	8713
Class 1	0,95	0,86	0,90	6940
macro avg	0,92	0,91	0,92	15653

Tabella 4: Results obtained by applying Decision Tree and KNN

The results so obtained highlight an excellent performance of our classifiers although it has been used the most simple classifiers with a simplistic study of their hyperparameters.

3.2 Experiments with hyperparameters's tuning

In this section, all the models previously described are tested on the tweets text by tuning the hyperparameters, in particular using the methodology of K-Fold-Cross-Validation with k = 5. The tested parameters are summarized in the table below :

KNN	Naive Bayes	Decision Tree	SVC
selbestk__k: [200, 300, 500, 800, 1000] n_neighbors: list(range(5,28,2)), metric: ["euclidean", "manhattan"]	selbestk__k: [200, 300, 500, 800, 1000]	selbestk__k: [200, 300, 500, 800, 1000] max_depth: [None, 2, 5, 10, 15, 20], min_samples_split: [2, 5, 10, 15, 20], min_samples_leaf: [1, 5, 10, 15, 20]	selbestk__k: [200, 300, 500, 800, 1000] C: [0.01, 0.1, 1, 10, 100]

Tabella 5: Hyperparameters's configuration

For testing all the possible hyperparameters's combinations of the SVC and Naive Bayes has been used a GridSearch. In the other cases has been used a Randomized Search because it's less computationally expensive and it has been set 100 as a maximum number of repetitions. By doing this, the goal was to find the best parameters that maximized the F1 macro's performance. The results obtained are listed in the tables below:

SVC	Precision	Recall	F1-Score	Support
Class 0	0,91	0,96	0,94	8713
Class 1	0,95	0,88	0,91	6940
macro avg	0,93	0,92	0,93	15653

Naive Bayes	Precision	Recall	F1-Score	Support
Class 0	0,92	0,89	0,90	8713
Class 1	0,87	0,90	0,88	6940
macro avg	0,89	0,90	0,89	15653

Tabella 6: Results obtained by applying SVC and Naive Bayes with hyperparameters's tuning

KNN	Precision	Recall	F1-Score	Support
Class 0	0,87	0,95	0,91	8713
Class 1	0,93	0,82	0,88	6940
macro avg	0,90	0,89	0,89	15653

Decision Tree	Precision	Recall	F1-Score	Support
Class 0	0,90	0,96	0,93	8713
Class 1	0,94	0,87	0,90	6940
macro avg	0,92	0,91	0,92	15653

Tabella 7: Results obtained by applying SVC and Naive Bayes with hyperparameters's tuning

The best models obtained by these experiments were the SVC and the Decision Tree. The following two figures below show their confusion matrix: recall values for class 1, respectively 88 for SVC and 87 for DT, are due to a considerable number of records (between 800 and 900) which were erroneously predicted as class 0.

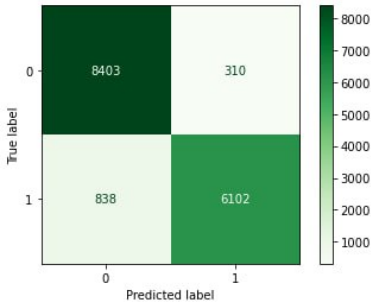

Figura 7: Linear SVC performance

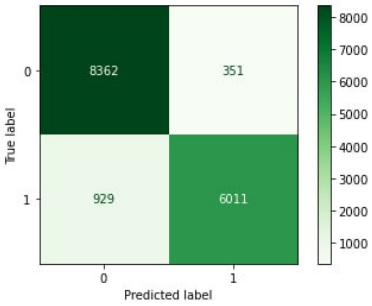

Figura 8: Decision Tree performance

3.3 Conclusions

In both cases, the results obtained were very good, most likely dictated by the balancy of the initial dataset. However, doing the search using the tuning of hyperparameters has slightly increased the performance of the models considered.

4. Neural Network Classifiers

This section provides an analysis of two classification models based on neural networks. Both models are sequential and structured in three levels: Embedding, LSTM and Dense. The middle layer defines the resulting neural network as recurrent in that it allows generating a new output after evaluating the current input along with the output and previous memory, which is very useful for text classification. Finally, the output layer contains a single neuron for making predictions and uses the ”sigmoid activation” function to produce probability output in the range of 0 to 1, which can be easily and automatically converted into crisp class values. For the training phase has been use the logarithmic loss function ”binary crossentropy”, while the ”Adam” optimization algorithm has been used to perform the back-propagation step with the descending gradient method.

4.1 Randomic embedding weight matrix

In this case, there is the definition of a neural network where, at the first level of word embedding, a random matrix of weights is generated and ”adjusted” during the training phase. The performances of the model during both training and validation are shown in the figures below.

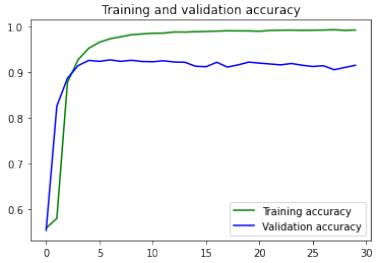

Figura 9: Accuracy Learning Curve

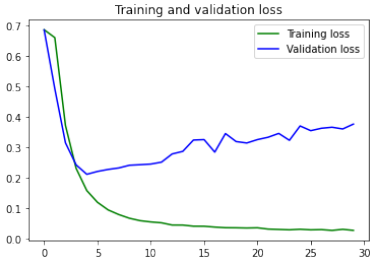

Figura 10: Loss Curve

As can be seen in the previous images, as the epochs increase, the accuracy learning curve for training and validation tends to come close to 1 until the fifth epoch is reached. On the other hand, the Loss function in validation does

not have a logarithmic structure but takes an irregular shape starting from the fifth epoch. The reason could be the continuous updating of the weight matrix for each iteration in the first level of the neural network. The following table shows the results of the model's predictive ability on the target variable "Toxicity": the performances of the model on the test data are excellent with an accuracy of 92%.

	PRECISION	RECALL	F1-SCORE
CLASS 0	0.92	0.93	0.93
CLASS 1	0.91	0.90	0.91
Macro-avg	0.92	0.92	0.92

4.2 Pre-trained embedding weight matrix with GloVe

In this second experiment, the neural network was defined using a pre-trained embedding layer with GloVe(Global Vectors for word representation), which was preferred to Word2Vec because it defines representations based on global word-by-word co-occurrence statistics. In particular, the weight matrix of this layer is no longer randomly generated at the first epoch but it is replaced with the vector representation of the tweets words contained in a specific GloVe file. In this second model, the Adam optimization algorithm is prevented from adjusting the matrix weights at each iteration by setting an appropriate parameter. The results can be seen in the following figures:

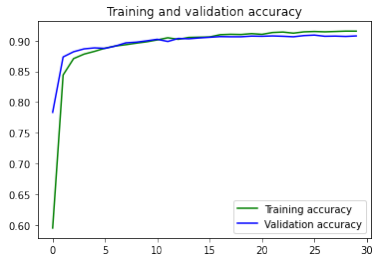

Figura 11: Accuracy Learning Curve

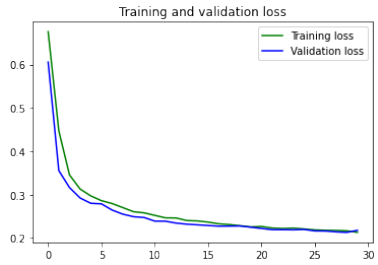

Figura 12: Loss Curve

As shown by the images, the trend of Loss functions in this second experiment remains the same and doesn't suffer of any alterations unlike the first case. The reason is most likely to be found in the use of a pre-trained weight matrix with the embedding vectors contained in the GloVe word file. A good improvement can be also seen in the accuracy graphs during both training and validation, which presumably rules out overfitting situations when classifying new test batteries, because the two curves tend to be coincident.

	PRECISION	RECALL	F1-SCORE
CLASS 0	0.91	0.94	0.93
CLASS 1	0.92	0.89	0.90
Macro-avg	0.92	0.92	0.92

Generally speaking, the results of both experiments coincide and reach approximately the same performances during the the classification phase even using models which aren't based on neural networks. As you can imagine, the results obtained depend very much on the dataset used for the analysis. Since the one used in this project is very balanced, the result obtained was almost expected and it confirmed our hypotheses. [3]

5. BERT

BERT (Bidirectional Encoders Representations from Transformer) is a state-of-the-art natural language understanding model from the Transformers family. Unlike unidirectional models that read incoming text sequentially (left-

[3] GloVe documentation

to-right or right-to-left), BERT makes use of a Transformer encoder that reads the entire sequence of words simultaneously, so it is considered bidirectional. This feature allows the model to learn the context of a word based on everything around it, both on the left and on the right of the word.

5.1 Pre-Processing

The data used for this phase belong to the dataset cleaned and pre-processed during the Data Preparation phase. However, in order to run this model, further processing of the data was carried out through the use of the *BertTokenizer* class. Through the latter, it was possible to use the **encode_plus** method to perform the following steps:

- Tokenization of sentences
- Insertion of the special token [CLS] at the beginning of each sentence
- Insertion of the special token [SEP] at the end of each sentence
- Addition of padding tokens [PAD] to achieve maximum length in each sentence
- Mapping tokens with the *bert-base-uncased* vocabulary indices of the Tokenizer
- Creation of attention masks that differentiate real tokens from padding tokens

The maximum sequence lengths supported by the model are 128 for the train and 512 for the test phase. In this project, it was decided to calculate the maximum length suitable for this task considering the available sentences and for both training and testing the value chosen is 25 tokens: analyzing the distribution present in Figure 13, this number is the one that avoid truncating the sentences, while still remaining far below the defined thresholds and avoiding to computationally burden the model.

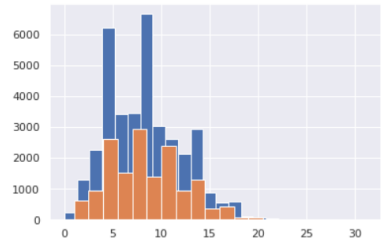

Figura 13: Word Distribution: Training & Test

In the following steps, it's briefly shown an example of how the data are pre-processed:

- Original tweet: "hell wrong humanity would anyone need anything prove"
- Embedding tweet: ['[**CLS**]', 'hell', 'wrong', 'humanity', 'would', 'anyone', 'need', 'anything', 'prove', '[**SEP**]']
- Mapping after padding: [**101**, 3109, 3308, 8438, 2052, 3087, 2342, 2505, 6011, **102**, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
- Attention mask: [1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]

In addition, an iterator was created with the class *DataLoader*. Through this iterator, it is not necessary to load the entire dataset into memory, and in this way it is possible to save memory during the training phases. The model considered is the basic BERT (BertForSequenceClassification), which is useful for being able to perform classification. In conclusion, during the execution of this model, the metrics taken into account are *training loss* and *validation loss*. The latter is used for the validation phase. Through these metrics it was possible to understand how, by increasing the number of epochs, the *training loss* becomes lower than the validation loss. This may indicate that the model is poorly fit. For this reason, an epoch number of 2 was chosen.

5.2 Binary BERT Results

The dataset chosen consists only of the tweets and toxicity. Since the latter variable is binary, only binary BERT could be run with it. The results are shown in the following table:

	PRECISION	RECALL	F1-SCORE	SUPPORT
CLASS 0	0.94	0.96	0.95	8734
CLASS 1	0.94	0.93	0.94	6919
Macro-avg	0.94	0.94	0.94	15653

From the results reported in the table above and considering the *accuracy* value = 0.94, it is evident that the performance of the model on the available data is excellent. Furthermore, comparing the binary BERT with the previous model, the first one leads to slightly better results.

6. Advanced Tasks

6.1 Emotions detection

In this section, an advanced Natural Language Processing model called EmoRoBERTa is used to label each tweet with an emotional type that can take on 28 values. The imported model was pre-trained on a dataset of 58,000 Reddit comments and it is an extension of the basic BERT model. Several experiments were performed with the multiclass Bert model in order to test the predictive ability of the transformer, using the emotions detected in this first phase as the target variable. The Figure below shows the distribution of emotions within the Toxic Tweet Dataset.

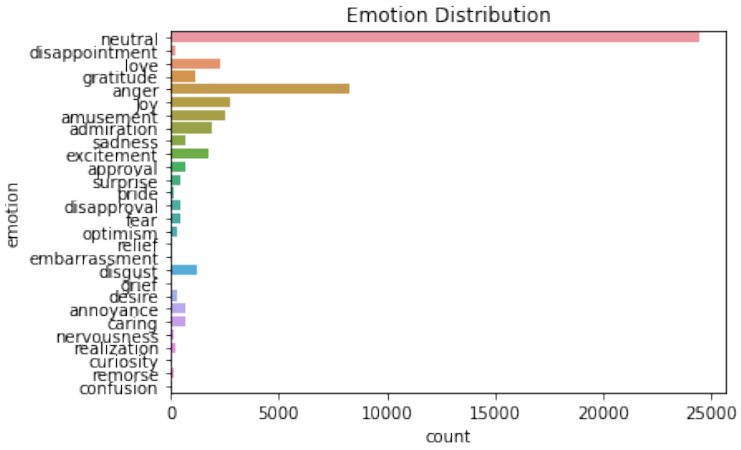

Figura 14: Emotions countplot

As can be seen, about 50 per cent of the records are characterised by a *neutral* emotion. The second highest class by count is *anger*, an expected result given by the large amount of data labelled as toxic comments.[4]

6.2 BERT Multiclass

After searching for the emotions associated with each tweet and adding this information to the dataset, the multiclass BERT was run with this nonbinary target variable. The data pre-processing and methods used in this model are the same as those described for the binary BERT.

In the following table 8 are shown the results obtained with this model: only those emotions that reported an f1_score value different from zero are shown in the table. In fact, having a value of f1_score equal to zero means that some labels in the y_test probably do not appear among those predicted.

[4]EmoRoBERTa documentation

	PRECISION	RECALL	F1-SCORE	SUPPORT
admiration	0.60	0.70	0.64	571
amusement	0.82	0.86	0.84	763
anger	0.74	0.87	0.80	2490
annoyance	0.50	0.00	0.01	201
approval	0.45	0.02	0.04	214
caring	0.51	0.24	0.32	198
desire	0.36	0.32	0.34	96
disappointment	0.80	0.06	0.11	65
disapproval	0.72	0.09	0.16	141
disgust	0.50	0.53	0.51	364
excitement	0.66	0.52	0.58	525
fear	0.25	0.20	0.22	140
gratitude	0.81	0.86	0.84	336
joy	0.67	0.78	0.72	828
love	0.76	0.83	0.80	687
neutral	0.83	0.86	0.84	7328
optimism	0.86	0.07	0.13	84
pride	1.00	0.03	0.05	38
realization	1.00	0.07	0.12	60
remorse	0.81	0.36	0.50	36
sadness	0.44	0.67	0.53	214
surprise	0.56	0.57	0.57	131
Macro-avg	0.52	0.34	0.35	15653

Tabella 8: Results obtained by applying multiclass BERT with emotions

Conclusions

Resuming the steps performed in our research, as the first objective was to perform topic modeling on the corpus of tweets at our disposal. The idea was to determine the types of toxicity present in the dataset by highlighting latent topics and by interpreting their meaning in an empirical approach. After performing several tests by changing the parameters, we found that there are three main types of toxicity inside this specific dataset : cyberbullying, racism and misogyny.

Starting from the original dataset, a classification task was performed taking ”Toxicity” as the target variable and using KNN, Naive Bayes, Decision Tree, Linear SVC and neural networks as main classifiers. The results obtained are very successfully both for the traditional algorithms and neural networks thanks to the balance of the dataset. We decided to complete the classification task through the use of the binary BERT algorithm: the results obtained were found to be better than using traditional classifiers.

Furthermore, for a deeper investigation for emotions in our dataset it was used the EmoRoBERTa algorithm, which provides an already trained dictionary containing emotions associated with certain words. With this application, we then tried to understand the general presence of emotions in the dataset, getting quite unbalanced results because neutral and angry emotions were the most frequent.

Based on the emotions found, we then applied multiclass BERT to perform an additional classification task on the variable ”emotion”(obtained thanks to the EmoRoBERTa algorithm). The results obtained are not as satisfactory as those obtained for the toxicity variable because of the unbalanced nature of the dataset.