# Twitter Sentiment Analysis Classification

Giuseppe Concialdi
*Politecnico di Torino*
Student id: s294666
giuseppe.concialdi@studenti.polito.it

Christian Montecchiani
*Politecnico di Torino*
Student id: s303681
christian.montecchiani@studenti.polito.it

***Abstract*—**

## I. Problem overview

The objective of the competition was to build a model that is able to classify whether a tweet contains positive or negative sentiments. The dataset provided is arranged as follow:

- A **development** set composed by 224,994 records of tweets. Each sample has six different features, including the *sentiment* attribute that is the target of the classification.
- A **evaluation** set consisting of 74,999 samples. Its dimension is one-third of the development set and it does not feature the target variable.

******move***The dataset is quite large, so the time required to manage the operations on the data should be taken into account because it could be not trivial. It is essential to retrieve some meaningful information of the data by exploiting its features. In particular, every sample is characterized by:

- *ids*: the unique identifier of the tweet. It is represented by a progressive integer number that is related to the timestamp of the tweet. The lowest value of the *ids* attribute is 1,467,811,193 while the highest is 2,329,205,038. It is uncertain if the dataset includes only a subset of the actual number of posted tweets or if the increase of the values follows some pattern. In fact, by digging in the past twitter documentation, we found out that the tweet id is generated with a snowflake schema, invented by Twitter for the generation of sequential ids for their tweets.
- *date*: the timestamp of each tweet. The date is encoded as a string not in the ISO 8601 standard [1] but with the format:

  $$weekday \quad month \quad day \quad hour:min:sec \quad tz \quad year$$

  From this schema, the information can be easily extracted and exploited to train the model. The dates in the dataset range from April 6 to June 25 of 2009. Knowing the temporal ranges will help during the preprocessing phase.
- *flag*: a string whose significance is unsure. It is present in the whole dataset with the unique value of `NO_QUERY`. Given its absence of meaning, this feature will be removed without a second thought, but it is possible that it would report the query used to retrieve the tweets when they were extracted using some Twitter APIs.
- *user*: the username of the creator of the tweet. Even though there are almost 225 thousand tweets, there are only 10,647 different usernames. Therefore, on average, the users are very active and they have probably posted several tweets in this period.
- *text*: the text of the tweet. This is the core part of the analysis, it embodies a lot of insights that can be extracted and analyzed to retrieve the overall polarity of the tweet's sentiments. In 2009 the maximum length of a tweet was 140 characters [2]. However 1 shows that some tweets exceed this threshold, so there are likely issues with the encoding of the text extracted.
- *sentiment*: the target variable of the classification. It assumes two possible integer values: 0 and 1 that represent respectively negative and positive sentiments. The dataset is fairly unbalanced, as shown in figure 2, there are more positive sentiments than negative ones.

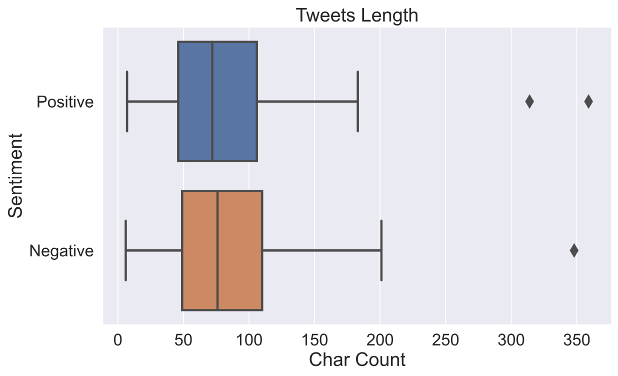

Fig. 1: Boxplot showing the number of chacters of the tweets per sentiment

In the II we will assess what are the most relevant features to carry on the classification and how we will extract the relevant information that lies within each feature.

## II. Proposed approach

*****Move in overview maybe with a table**The dataset does not feature any missing value, but it contains redundant information that is useless for the analysis. We decided to extract the time-related information from the *date* attribute. In this manner, we added new features to the dataset and also some aggregated measures like the time of the day (morning, afternoon, night) and the time of the time (workday, weekend) during which the tweet was posted. At this stage was still

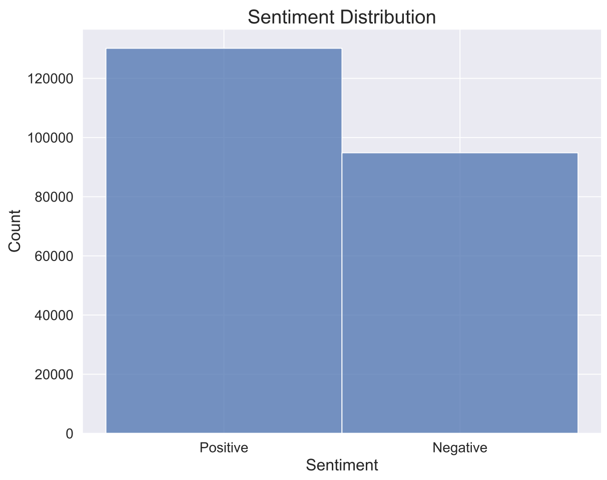

Fig. 2: Distribution of the tweets' sentiments in the training dataset

unknown whether all these attributes extracted from the *date* variable would have been useful or not. That is because the *ids* feature is not useless, but it already encodes the timestamp of the tweet. Nonetheless, we decided to leave those features and figure out later if their relative importance would matter.

Although the *date* attribute could hide some useful insights about the sentiment of the tweet, the major knowledge lies within the *text* attribute. The information extraction from this feature can be performed in different ways and with different processes. We decided to tackle the problem from different points of view and we managed to embed all this data into our model. Firstly, we cleaned up and fix some problems present in the tweets' text, then we exploited the text performing:

- A **sentiment intensity analysis** [] on the text, obtaining new features on the polarization of the sentiments.
- A **tf-df** [] of the text in order to get the words that mainly influence the sentiments of the tweets.
- A **word embedding** [] approach with the FastText [**?**] library to retrieve the morphological relationships between the words in a sentence.

Figure 3 shows a summary of our approach. The last technique resulted in a very powerful tool, that is able, on its own, to perform the classification of the sentiment of the evaluation set with an f1-score [**?**] higher than 0.8. This was our baseline for the development of the model, and we accomplished higher performance by integrating the likelihoods of the FastText supervised learning classification into our dataset.

Finally, we wanted to add the *user* attribute into the equation. We thought that due to the relative low cardinality of this attribute (only 10,000 different usernames) the tweets posted by the same author may reflect its personality, thus it is probable that a sentiment is predominant to the other one. We did not want to encode the information of each user because of the subsequent dimensionality increase of the

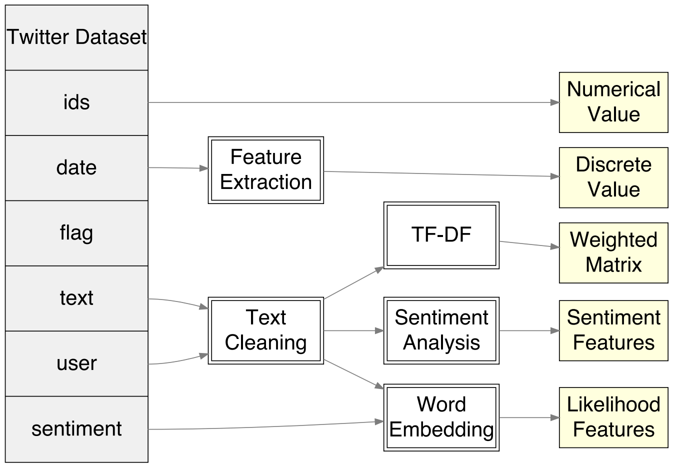

Fig. 3: Overall schema of the problem approach

dataset with an approach like the One-Hot-Encoding [3]. *****to rephrase** So we chose to incorporate the user author of the post at the beginning of the text of the tweet. The text already contains mentioned user whose name is preceded by an at-sign (@). In this way, the word embedding classification will retrieve information about the author of the post like if he was mentioned within it.

Sections II-A and II-B will dig into the details of the three different text mining techniques and some comments about the word embedding can be found in the Discussion section IV of the paper.

## A. *Preprocessing*

The preprocessing step is the core phase of the entire analysis. Here we performed the extraction, transformation and normalization of the features. As discussed in Section II, we extracted from the *date* attribute all the relevant information and we discarded the properties that are redundant like *year*, *timezone* and *flag*. Then we decided to add a new feature based of the number of characters of the tweet. The new *char_count* attribute is useful to troubleshoot the tweets' texts that are longer than 140 characters and it is also valuable for the classification itself.

Afterwards, we addressed the tweet duplication issue. There are three kinds of duplicated tweets:

1) Same text, different id, same sentiment
2) Same text, same id, same sentiment
3) Same text, different sentiment

For the first kind of duplicated tweets, no action is required. Those are very common and eventually short sentences that different authors twitted. They share the same sentiment so it enforces the model to learn that those phases belong to one specific class. The second kind are tweets that have been posted more than once by the same author. This could be a mistake or maybe the user wanted to retweet the same post again. Either way, we decided to keep only one copy of the duplicated post. The most controversial kind of duplication is the last one: regardaless of the author, if the same text is labelled differently, then the classifier will be confused on how to handle samples similar to these ones. For this reason we decided to drop all the records with these conflicting properties.

Then, we tackled the cleaning of the tweets' text. To properly perform this step, we took into consideration a lot of alternatives. Different elements characterize a Twitter post and we tried to draw every possible bit of information from the text. The tweets' length issue led us to inspect the text in search of an encoding-related problem. We found out that the text extracted seems to be encoded in UTF-8 [**?**] but some special symbols were parsed as HTML entities [**?**]. We employed the html library to clean up the text. After that, we lowered the case of the words and we extracted the domain name from the urls contained in the tweets. We exploited two vocabularies[1] to expand into clear words the emoticons and the slang terms. By substituting these pieces of text in addition with some regex to regularize the phrases we obteined a much more coherent and convenient format for the analysis. Ultimately, we applied the lemmatization [**?**] to extract the lemma from each word. We preferred lemmatizing to stemming [**?**] in order to preserve the morphology of the words [**?**]. Figure **??** shows the cleaning pipeline for the *text* attribute.

At this stage, the *text* feature is ready to be processed. We heavily relied on the tools offered by the NLKT library [**?**] to carry on the analysis. We employed the Vader Sentiment Intensity Analyzer [**?**] to gain the relative frequencies of *negative*, *neutral* and *negative* terms. Besides, also the *compound* value is stored, it is a single value summarises the three frequencies. Following, we used the TextBlob library [**?**] to extract additional features: the *subjectivity* score and the *polarity* index. These variables are correlated among them, but they describe the same phenomenon under different point of view and complement each other.

Going forward, we applied a tf-df to the text word with a relative high minimum support. Our aim was to retrieve the words that were present in a lot of tweets and which meaning was heavily polarized toward one class. In this case, we did not use a normal stop words vocabulary, because there are a lot of common term that are very important for the classification, like the negations. So we created a custom vocabulary that contained only conjunctions and some adverbs.

Finally, we leveraged the word embedding approach through Meta's FastText library. The vocabulary was built on top of the tweets of the development set. The autotuning feature of the library allowed us to treat the neural network underneath the facade as a black box. We split the development data into training and validation and the model tuned itself by maximizing the resulting f1-score. One of the strenghts of this technique is that by converting the words into multi-dimensional vectors it does not suffer the presence of unfamiliar terms. The output of the neural network is a prediction and the softmax likelihood for each class. We extracted these probabilities and we integrated them into the dataset. Figure 4 summarises the operations perfomed on the *text* attribute.

[1] *****insert libraries**

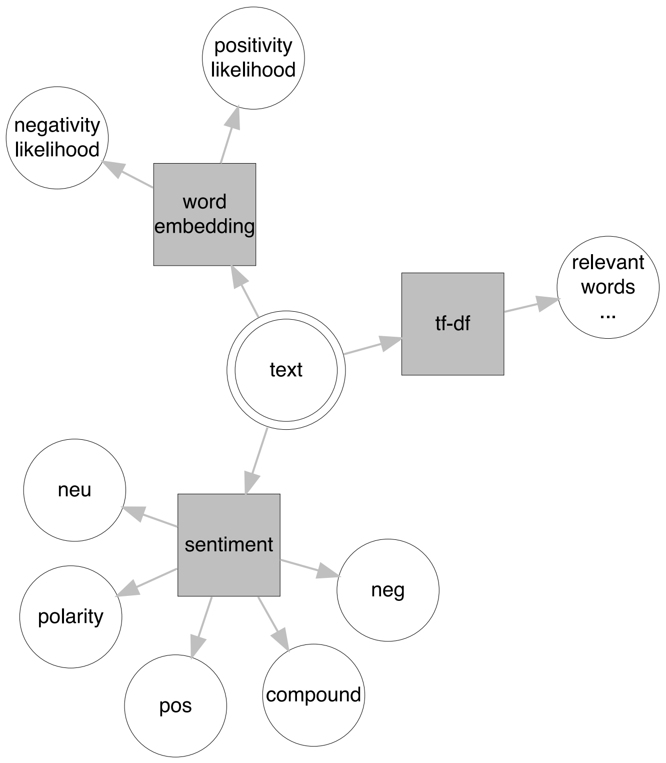

Fig. 4: Feature extraction schema of *text* attribute

The data is not yet ready to be processed by a classification algorithm. The categorical needs to be addressed. We removed the *text* and the *user* features because we have already extracted the information needed and then we proceed to map the remaining features into numerical ones. All the columns are normalized with a MinMax scaler [**?**] and it is ready to go through the model selection pipeline.

### *B. Model selection*

### *C. Hyperparameters tuning*

## III. Results

Here you will present your results (models & configurations selected, performance achieved)

## IV. Discussion

Any relevant discussion goes here.

## References

[1] ISO 8601-1:2019, *Part 1: Basic Rules: Date and time – Representations for information interchange*. ISO, Geneva, Switzerland.

[2] K. Gligorić, A. Anderson, and R. West, "Adoption of twitter's new length limit: Is 280 the new 140?," 2020.

[3] D. Harris and S. Harris, *Digital Design and Computer Architecture*. Engineering professional collection, Elsevier Science, 2013.