Predicting political orientation from Twitter contents

Alberto Bernardi
Gianmarco Cracco
Elena Izzo
Luca Lezzi
Alessandro Manente

May 21, 2020

Contents

1 Introduction

Twitter is a popular social network where people can express their thoughts in the form of the so-called *tweets*, short messages limited to 280 characters. It allows to publish both multimedia and written contents, so that it is frequent that users express their personal opinion. Due to this and to its relatively restrictive privacy policies (e.g. if compared to Facebook or Instagram), Twitter represents a tremendous source of information about people and so it can be really useful for the purposes of sentiment analysis. Indeed, sentiment analysis aims at determining the opinions, attitudes or feelings of a person toward a certain topic and to do so it needs a great quantity of information.

A relevant domain of interest in sentiment analysis has been and is, especially nowadays, the prediction of political orientation; as an example, it suffices to consider the 2016 presidential campaign for the US government led by Donald Trump, that highly exploit this kind of data analysis to identify undecided voters.
As previously said, Twitter can provide us a huge amount of data; however two of the main issues with these data are the non-homogeneity of the topics (they may deal with subjects far apart from politics) and the language used, that is full of colloquialisms, typos and slangs; so it is needed a significant preprocessing phase to allow us to optimize the sentiment analysis.

Machine learning suits very well this kind of problem because it allows to work on enormous databases and extract specific information that would be inconceivable for a human mind. In particular, a convenient way to approach the prediction of political orientation is to consider it as a problem of classification: we have as labels the different parties we want to assign a user to. In the specific case of this project, our final objective is to create a model to predict whether a user supports a left-wing or a right-wing party of the Italian political landscape, so we will distinguish two different labels, *DX* for the right-wing and *SX* for the left-wing. We will eventually consider also a third label, for the Non-Classifiable users (*NC*), as we will state more precisely later on.

We will create our machine learning models through *fastText*, a Natural Language Processing library very fast and efficient powered by *Facebook*; furthermore, we will focus on how different preprocessing and data analysis techniques can overturn the performances of these models.

2 Data Collection and Database Creation

As we have already suggested in the introduction, our aim is to create a model which is able to predict the political orientation from Twitter contents. This result can be achieved by using a machine learning-based library such as *fastText*. Therefore, the first step of this project consisted in the collection of data and the classification of users to build a database of politically labelled profiles that will be used to train the model.

To create the database we selected Twitter profiles with the following criterion: one thousand users chosen randomly between the followers of each of the official Twitter profiles of the five most popular Italian parties: *PD*, *Movimento 5 stelle*, *Lega*, *Fratelli d'Italia* and *Forza Italia*; three thousand users chosen randomly from Italian Twitter users. Hence, the total of the Twitter users selected is around eight thousands.

Each member of the group expressed his thought about each user's profile, classifying them in six main categories:

- Non-Classifiable (0)
- Lega (1)
- Partito Democratico and Italia Viva (2)
- Fratelli d'Italia (3)
- Movimento Cinque Stelle (4)
- Forza Italia (5)

This procedure led to the construction of a dataset like the one in Table 1.

Twitter ID	**Username**	**Sex**	**SID1**	**SID2**	**SID3**	**SID4**
Ruggier98623365	Ruggiero	M	1	1	1	1
EricSalicetti	Eric Salicetti	M	0	0	0	0
MassimoDolore	Massimo Dolore	M	4	4	4	4
Rivalevante	Sergio Stagnaro	M	2	0	2	2
Iolanda07658172	Islanda	U	1	3	1	3
manuelamimosa	Manuela Mimosa	F	2	2	4	2

Table 1: Example of the dataset

Our group, in particular, was asked to extract one thousand users from the followers of the official Twitter profile of *Forza Italia* (*@forza_ italia*) and to classify them in the six main categories just listed.

It is really important to mention that, during the classification of a user, we took into account just the tweets after May 2019. This choice was made essentially because it was the period of the European elections that marked the beginning of the crisis of the *Lega-Movimento 5 Stelle* government, a crisis ended up with the fall of the government and the birth of the new *Movimento 5 Stelle-Partito Democratico* government, the so-called "Conte Bis". Moreover, the Italian political landscape has always been very mutable, with frequent coalition changes and new born parties, therefore it would have been difficult, and maybe even confusing, considering older tweets.

One of the major difficulties we encountered during this first task had been to distinguish between supporters of *Lega* and *Fratelli d'Italia*, because of their similar ideals and their intense cooperation. A similar issue occurred when it came to the classification of a decent group of users that, although clearly against many right-wing ideals, could be associated neither with *PD* nor *M5S*. In this latter case, we chose to classify them as Non-Classifiable, while in the former we managed to classify them more specifically because of the more recognisable ideals of the right-wing parties or by mean of counting the greater number of retweets in support of a specific party.
A different problem came up when we had to classify non-Italian users: we decided not to consider them as Non-Classifiable, but rather to discard them since not engaged in the Italian political talk.

The database was in the end completed by downloading all the tweets of the selected users, a crucial step since all these tweets later became the object of our data analysis.

Recalling our aim of being capable of predicting the political orientation of a Twitter user from the text of his tweets, we focused in particular on determining whether a user was a supporter of right-wing parties (*Lega*, *Fratelli d'Italia* or *Forza Italia*) or left-wing parties (*Partito Democratico* or *Italia Viva*). In order to achieve this result, we operated on the database, distinguishing between left, right, *M5S* and Non-Classifiable users following the procedure outlined below.
The judgment of each member of the group was converted with the following criterion: numbers (1), (3) and (5) were associated with the right-wing classification (*DX*), number (2) with a left-wing one (*SX*), while number (0) and (4) kept their initial meaning (*NC* and *M5S* respectively). In this way, we obtained a database structured like the one in Table 1 with the judgments belonging to {*DX*, *SX*, *M5S*, *NC*}.
At this point two thresholds had been fixed: one at 75% and the other at 100%. These two values represent the minimum level of agreement between the judges to associate a user with one of the four labels. This means that, for example, assuming to work with the 100% threshold, a user can be labeled

as *SX*, *DX* or *M5S* only if all the judgments agreed on the fact he supported left parties, right parties or Movimento Cinque Stelle, respectively. In any case, if the threshold was not reached, the user was classified as *NC*. Table 2 shows the frequency and percentage for each label according to the chosen threshold. It is evident that there is a disparity in the quantity of users classified as *DX* and *SX* (the main objects of interest of our analyses, as we will see later). As a result, efforts had to be made to create models that would compensate for this initial data disparity.

	Threshold = 75%		**Threshold** = 100%	
Label	**Frequency**	**Percentage**	**Frequency**	**Percentage**
DX	3065	39%	2920	37%
SX	1230	16%	1155	15%
M5S	262	3%	236	3%
NC	3332	42%	3479	45%

Table 2: Frequency and percentages for different classifications

3 Preprocessing

The most important pre-analysis phase is the so-called *preprocessing*, phase which consists in cleaning up the data (in this specific case, text of tweets). This step is so important because *fastText* needs to have a specific text formatting in order to work and, in addition, it performs better if the number of different words is reduced to the minimum possible. To understand the relevance of this stage, it suffices to consider how big our dataset is, as Table 3 shows (it does not consider tweets and words associated to user classified as *M5S*). As a consequence, this phase took a long time to be perfected and improved, because we wanted to leave the fewest words without losing any relevant content.

Tweets	**Threshold** = 75%	**Threshold** = 100%
Total	5 625 445	5 643 780
DX	2 474 632	2 404 567
SX	914 992	847 418
NC	2 235 821	2 391 795
Total words	94 289 013	94 599 500

Table 3: Statistics on tweets.

The first step consisted in removing from the analysis all the tweets published prior to May 2019. This choice was dictated by two factors: the former was, as previously highlighted, the highly changeable Italian political framework that entails the difficulty in being able to analyse tweets related to long periods of time; the latter was the fact that starting from May there have been a series of political events strongly cited in the social networks: among these we remember the European elections, the fall of the *Movimento Cinque Stelle-Lega* government, the subsequent formation of the *Partito Democratico-Movimento Cinque Stelle* government, the Umbria regional elections and the birth of the *Sardine* movement.

To obtain a highly cleaned text, first of all we turned all the characters inside the tweets to lowercase in order to avoid case-sensitive distinctions. Secondly we removed links (e.g. *https://t.co/5VwnWaBYk8*) and emojis, punctuation (with the exception of the hashtag symbol # and the *at* symbol @) and words with characters of alphabets different from the Italian one. In addition we cancelled the so-called *stop-words*, a list of words like articles, pronouns, auxiliary verbs, that are not necessary for the analysis (The list of words is shown in Table 4). We also removed all the numbers except 1, 5 and 100; these three numbers were kept because they are linked to groups of words such as *Movimento 5 Stelle* and *Quota 100* or phrases like «Sei il numero 1!» (which is the Italian translation of «You are the number 1!»).

a	rt	uno	tra	dalla	questi	quindi	tuttavia
b'	il	una	fra	dalle	nostri	sicché	altrimenti
e	lo	del	con	dagli	vostri	perché	affinché
o	la	brt	ove	senza	nostro	perchè	allorché
i	le	b'rt	che	anche	vostro	questo	nemmeno
né	un	nel	mio	tanto	oppure	tranne	neanche
se	io	dei	mia	ossia	eppure	ovvero	pertanto
ok	tu	egli	tuo	miei	quello	dunque	neppure
al	di	noi	tua	tuoi	quella	mentre	siccome
da	sua	voi	suo	suoi	quelle	quando	qualora
in	lui	gli	più	loro	quegli	quanto	eccetto
su	lei	dal	tue	alla	questa	appena	meno
ma	mie	per	sue	alle	queste	così	pure
				cioé	tipo	come	essi

Table 4: List of stop-words

Once we did so, we focused on the previously mentioned symbols: # and @ which had not been deleted yet. Hashtags were not removed because they constitute a source of information on Twitter, since they are labels used to aggregate similar topics and consequently, it is useful to find the most frequent hashtags and to consider them in the analysis to keep track of relevant arguments. So we looked for all the words between two hashtag symbols or between a hashtag and a tab, and we estimated the frequency of each of these words through all the tweets available. Then, we kept the 50 most frequent words and discarded all the others. Figure 1 shows the kept words with their frequency.

Similarly, also the @ is an important character on Twitter. In fact, it can be used in a sentence followed by a username when you want to cite users or reply to their tweets. Therefore, analysing usernames became fundamental since the *at* symbol may introduce influential political figures that are essential for our goal of predicting political orientation. Hence, we counted the frequency of each username and kept the fifty most frequent removing all the others. In Figure 2 are shown the kept usernames and their related frequency.

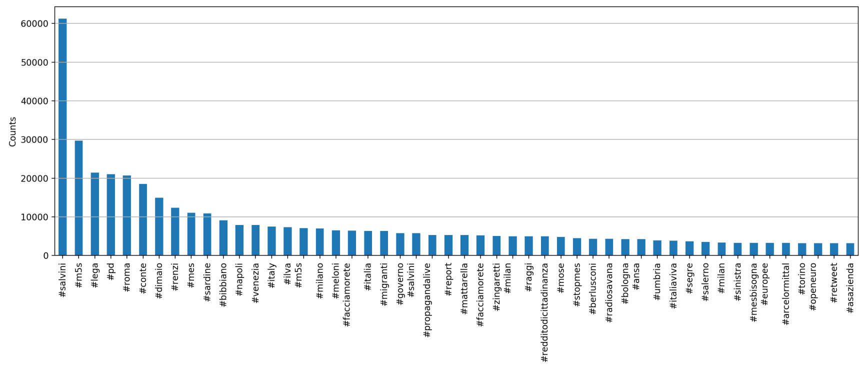


Figure 1: Frequency of the fifty most popular hashtags.

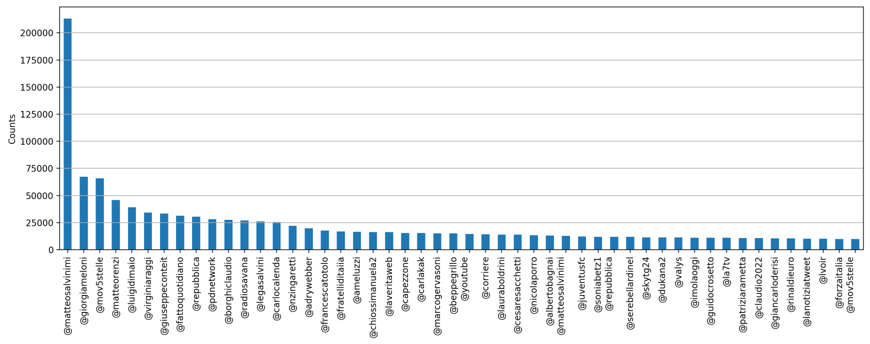


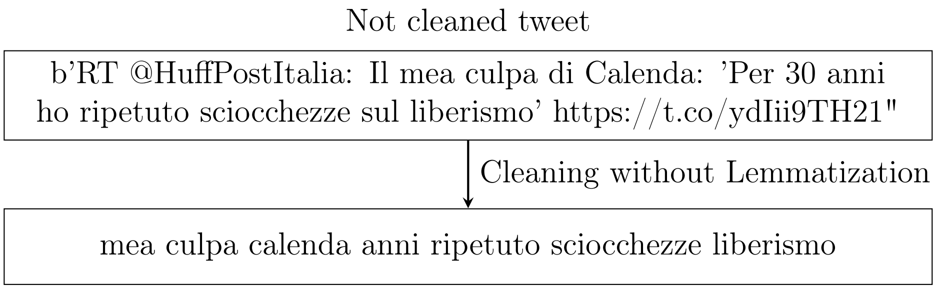
Figure 2: Frequency of the fifty most popular usernames.

At this point, we had cleaned up the text of the tweets. However, the preprocessing phase was not completed yet. To go further, indeed, we counted the frequencies of each word that had not been eliminated yet, and discarded all those repeated less than 10 times.
Moreover, we wanted to solve other issues: the lengthening of words by repeating the final vowel, the presence of laughter like "hahaha" or "ahahah" and the numerous meaningless words made up of one or two letters. To do so, we replaced blocks of repeated vowels with a single one, we removed all forms of laughter and all the words composed of one or two letters.

Another process that can be performed during the preprocessing is the *lemmatization*, a technique that groups together the inflected forms of a word, so that those can be considered as a single one, the so-called *lemma* or *dictionary form*. In this way our NLP algorithm can focus on less words and

give them an appropriate meaning.
The chop of the inflectional part should happen after a morphological analysis of the text, that should make the process the most accurate possible. However, it is clear that this step can be really difficult to implement, even more considering the scarcity of datasets for sentiment analysis for the Italian language. For the English language, by contrast, *lemmatization* is way more developed, so, one immediate possibility could have been translating all the tweets in English and then perform *lemmatization* on this new dataset; anyway, we discarded this option for two main reasons: the former is the high computational cost of this process, the latter is the obvious mistakes that could arise while translating such a huge text file, particularly because it is made up of tweets that, most of the times, are not written in a formal language and come with tons of slangs, colloquialisms and grammatical mistakes.
In the end, we opted for the Italian lemmatization, implementing it through different packages, *Pattern* and *Unidecode*, since *NLTK* by itself gave poorer results, messing up and confusing words.
Conscious of the risks and difficulties mentioned above and wanting to test the performance of *lemmatization* we built different models, some implementing it and some other not, as we will see later on.

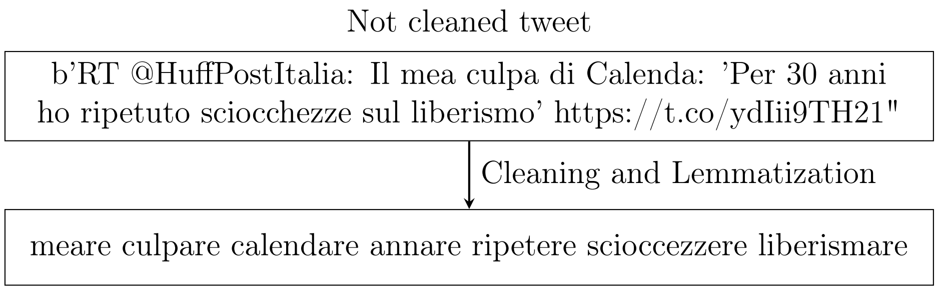
Here follows an example of a tweet before and after preprocessing with or without applying lemmatization.


Not cleaned tweet

b'RT @HuffPostItalia: Il mea culpa di Calenda: 'Per 30 anni ho ripetuto sciocchezze sul liberismo' https://t.co/ydIii9TH21"

↓ Cleaning without Lemmatization

mea culpa calenda anni ripetuto sciocchezze liberismo


Not cleaned tweet

b'RT @HuffPostItalia: Il mea culpa di Calenda: 'Per 30 anni ho ripetuto sciocchezze sul liberismo' https://t.co/ydIii9TH21"

↓ Cleaning and Lemmatization

meare culpare calendare annare ripetere scioccezzere liberismare

4 Model creation

Once we created the dataset and decided how to perform the preprocessing, we faced the problem of how to orient our work, that is, which kind of analysis to perform and what NLP algorithm to apply. We chose to create different models with the same NLP library, *fastText*, developed by *Facebook* and really suitable for our needs. In this section we will describe the different configurations of the models built and the reasons that lead to the choice of *fastText*.

4.1 Data Analysis

Instead of a comparison between the performances of different Natural Language Processing algorithms on our dataset, we preferred to choose a single algorithm and investigate how modifications on the preprocessing features and on the data considered affected the results of the tweets analysis. In particular, we distinguished models varying the following:

- Non-Classifiable users: we built our models on datasets that were alternatively limited to the users of the right-wing and left-wing parties or comprehensive of the Non-Classifiable users;
- Lemmatization: as stated above, the development of lemmatization for the Italian language is not so strong, so we alternatively implemented it or not in order to see how performances of our models would change;
- Label on single user or on single tweet: we built two conceptually different type of models, one labelling each user as a supporter of right or left parties (or even Non-Classifiable if necessary) and considering all their tweets together, the other labelling each tweet of the user with their same orientation; of course, these two models predict different things: the former predicts the orientation of a user, the latter the orientation of a single tweet;
- Splitting of the dataset: we performed two different data-analysis techniques to select the best train and test sets on which to construct our model: *stratification* and *upsampling* (for another example, the *k-fold cross validation*, look at the appendix).

4.2 Choice of *fastText*

The immediate consequence of the previous paragraph is the big number of models required and therefore the need for a high speed Natural Language

Processing algorithm, capable of creating a lot of model in a relatively short time. A reasonable choice, also according to [1], is *fastText*, that assures accuracies comparable to those of other deep-learning algorithms but is many orders of magnitude faster for training and evaluation of a model; moreover, it requires less computational power.

4.3 Train and Test Sets: Overcoming the imbalance of the Dataset

To create a supervised machine learning model, it is fundamental to divide the dataset into two parts: the training set and the test set. It is essential to choose them wisely in order not to have a biased or overfitted model. In particular, this holds in the case of this analysis: our datasets, in fact, both considering labelling at 75% and at 100%, are imbalanced, with the majority of the users classified as right-wing parties supporters (as seen in Table 2); therefore, to reduce the risk of obtaining imbalance train and test sets and to help us to find the best model, we implemented two different techniques: *stratified sampling* and *upsampling*, which are presented below.

4.3.1 Stratified Sampling

Stratified sampling consists in splitting data in different groups for every different label and then select a fixed percentage from each group as training data and the remainder as testing data; merging together training and test sets from the different groups, we obtain a dataset split randomly and in such a way that the initial imbalance is preserved and not worsen in training and test set.

4.3.2 Upsampling

Upsampling (or *over-sampling*) consists of adding new data (for our purpose, to one model we added users, to the other tweets) for the minority class (left-wing parties) to have *DX* and *SX* labels reach an equal number of elements. This procedure may cause overfitting, because the new data we add are just a duplicate of the one in the dataset, but in cases like ours, this should not be an issue since the dataset is very large. [3, 7]

5 Results

We will now deal with the results of this work, analysing performances of the various models we built; but first, let us recall the differences between them. The primary difference is the analysis technique we applied to create our datasets: stratification and upsampling. We will dedicate a separate analysis to each of these two and in particular we will distinguish between other two types: one with a threshold on the agreement fixed at 75%, the other at 100%, as we outlined above; for each of this two values we created 8 models, that combined alternatively three features:

- Considering or not a third type of users and therefore a third label: the Non-Classifiable users; this feature will be denoted as *NC* in our tables;
- Performing or not the *lemmatization*;
- Labelling each user or each of his tweets: as prior highlighted, this distinction produces two conceptually different models, the former predicting persons, the latter predicting single tweets; in the tables, the implementation of this feature will be stored in the *Person* column.

5.1 Stratification

The stratification method consists in maintaining the same ratio, in the train and in the test sets, between the frequencies of the labels. In Table 5 we can see the accuracies of the different models and, in particular, the highlighted ones are those with the highest score. The two best models were both those without the implementation of the lemmatization, that did not consider the *NC* users and that predict the single tweets. The best have both a pretty good accuracy, around the 80%.

Globally analysing the results obtained, the first thing to underline is that the accuracy is greater than 0.59 for all models. Secondly, it is interesting to analyse the variation in accuracy with the variation of the factors taking part in the creation of the model: *NC*, *Lemmatization* and *Person*.

We can note that the presence of lemmatization does not significantly alter the results and, as a matter of fact, the variation between two different models, which have the same features as regard *NC* and Person, is always less than 1%. This means that lemmatization did not bring improvement to the models, as one could have expected, stated the problem we introduced in Section 3. As for the presence of *NC*, we note how this factor significantly decreases the accuracy of a model, with a maximum variation of 13,4%. This is easily due to the fact that the introduction of a new label entails greater difficulty for *fastText* to create an accurate model and consequently leads to worse performances.

Parameters			**Threshold** = 75%	**Threshold** = 100%
NC	**Lemmatization**	**Person**	**Accuracy**	**Accuracy**
✗	✗	✗	0,787	0,790
✗	✗	✓	0,699	0,701
✗	✓	✗	0,781	0,786
✗	✓	✓	0,697	0,699
✓	✗	✗	0,653	0,651
✓	✗	✓	0,627	0,599
✓	✓	✗	0,648	0,646
✓	✓	✓	0,617	0,598

Table 5: Stratification Results

Finally, we observed that the results vary significantly if the analysis is performed by user rather than by their tweets. In particular, the latter case gave better results with an accuracy variation between 10% and 20%. The reason for this finding can be found in various causes: surely, one of the main is that, while the user analysis involves texts of variable length, the analysis by tweets defines a maximum length of the texts equal to the 280 characters imposed by Twitter.

As regards the choice of the threshold, note that the accuracy does not vary modifying the threshold with a model characterised by the same features. This is certainly due to the fact that the variation in threshold did not lead to a significant increase in the number of users classified as "right-wing" or "left-wing", as it can be seen in Table 2.

5.2 Upsampling

The upsampled dataset was balanced with the repetition of some random users (or tweets if the model predicts the single tweet) in order to have the two labels (or three, if *NC* users are considered) equally represented. In Table 6 we can see the accuracies of the different models and, again, the highlighted ones are those with the highest score.

Parameters			Threshold = 75%	Threshold = 100%
NC	Lemmatization	Person	Accuracy	Accuracy
✗	✗	✗	0,706	0,694
✗	✗	✓	0,623	0,576
✗	✓	✗	0,685	0,679
✗	✓	✓	0,650	0,611
✓	✗	✗	0,605	0,595
✓	✗	✓	0,535	0,543
✓	✓	✗	0,596	0,591
✓	✓	✓	0,530	0,533

Table 6: Upsampling Results

Like for the stratified models, the two best models were those without the implementation of the lemmatization, that did not consider the *NC* users and that predict the single tweet: they both have a discrete accuracy, around the 70%, but worse if compared with the previous bests.
Again, similarly to the stratified models, the accuracy is slightly higher, in most of the cases, for models with the threshold at 75%: an higher threshold should be intuitively linked to an higher level of certainty on the orientation of a user (recall the classification performed during the first stage of the data collection), but as already said above, the variation in threshold did not lead to a significant increase of the number of users classified as "right-wing" or "left-wing" (see Table 2).

Also in this case, the implementation of lemmatization gave not the best accuracies; nevertheless, the scores are very close between similar models, and in one case (second and fourth), the lemmatization improved the performances, so we do not suggest to reject it *a priori*.

A recurrent result is also the fact that models predicting tweets are more accurate than those with similar features predicting users.
Finally, it is easy to observe how much the introduction of a third label reduces the accuracy of our models: we lose $\approx 10\%$ between similar models, that is $\approx 15\%$ of the total accuracy, a significant percentage. Anyway, it is a logical consequence of the increase of the variability of our classification problem.

6 Application of the models

In the last sections we have seen all steps that allowed us to create models capable of predicting the political orientation of a Twitter user from the content of his tweets.

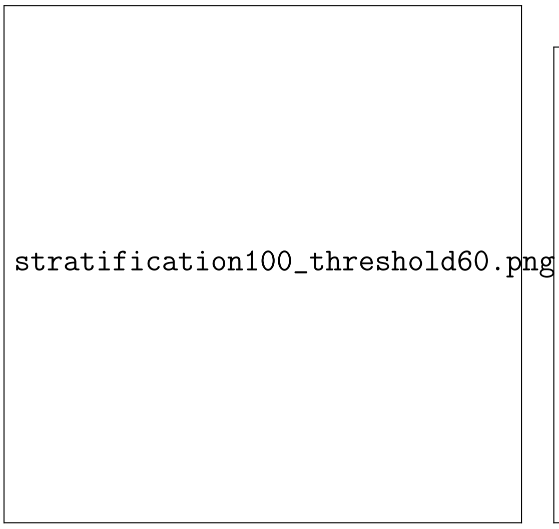
At this point it may be interesting to apply the best models to predict the orientation of new users not present in the dataset used to build the model. In particular, it may be interesting to predict users classified as *M5S* to investigate if they are considered right-wing or left-wing supporters. These predictions were obtained using the four best models (those highlighted in red in Table 5 and Table 6) and are displayed below.

It is important to underline that profiles to be predicted were treated exactly like those used to train and test file of the various models. This means that if lemmatization had been applied, it also had to be applied on the text of which you want to make a prediction; otherwise it had not to be applied. Analogously, if the model had been obtained labelling users' tweets, we predicted tweets and then performed a further analysis to determine users' political bias starting from the results obtained on tweets; instead if the labelling is for user, we had to predict the text including all the tweet of a user.
Finally if the threshold is equal to 100% we predicted users rated *M5S* at 100%, otherwise at 75%.

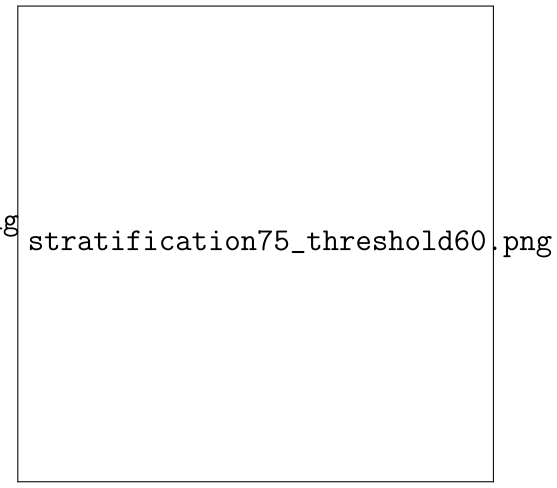
Each of the four models chosen to predict *M5S* users are characterized by the same factors: no *NC* label, no lemmatization and analysis for tweets; therefore, we predicted each tweet of a *M5S* user. To determine whether to classify him as a left-wing or right-wing supporter, we chose to operate as follows: a probability threshold was set equal to 0.60 and all tweets predicted as "right-wing" or "left-wing" with a probability greater than this threshold were kept labeled as such. On the contrary, all the other tweets were deemed as Non-Classifiable and after this step, for each user was counted the number of tweets cataloged with any of the three labels.
At this point, to establish if a user had to be classified with one label or another, we set an additional threshold that represents the percentage of tweets labelled as *DX* or *SX* that a user must have to be classified as such. In all cases where a user did not have the minimum percentage of required tweets, it was considered *NC*. Figure 3**(a)**, Figure 3**(b)**, Figure 3**(c)** and Figure 3**(d)** show the percentages of users predicted as *DX*, *SX* or *NC*, for each of the four models, as a function of this last threshold.

The predictions obtained for each of these four models are a bit destabilising. As a matter of fact, the results obtained applying the stratified models (Figure 3**(a)** and Figure 3**(b)**) show users classified either as *DX* or *NC* (the proportions between the two classifications varies as the threshold on x-axis varies), with an insignificant percentage of left users, really close to zero. On the other hand, the predictions obtained through the upsampled models (Figure 3**(c)** and Figure 3**(d)**) classified all users as left-wing supporters. The complete disagreement among these results suggests that, despite a good accuracy for each model, the choice of how to create the train and test file has a significant impact on future predictions. It would be interesting to increase the database by adding users that can be classified as left-wing supporters in order to reduce the initial imbalance of the data and check if this affects the results or not.

Furthermore, increasing the initial database would be useful in order to have other users to be predicted for checking the reliability of the model.

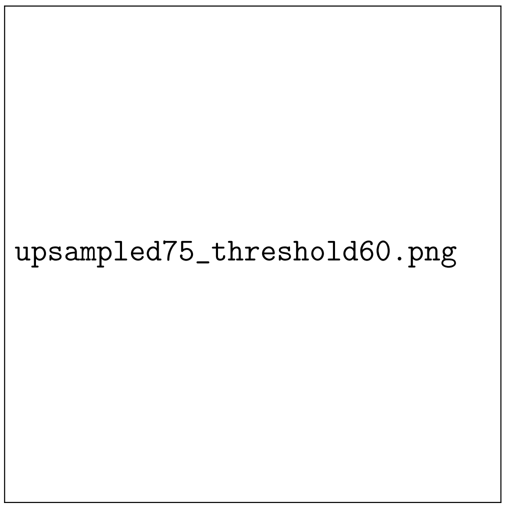


(a) *Stratification with threshold at 100%*



(b) *Stratification with threshold at 75%*

(c) *Upsampling with threshold at 100%*



(d) *Upsampling with threshold at 75%*

Figure 3: Results of predictions: graphs show the percent variation of users for each label (*DX*, *SX* and *NC*) as a function of the percentage of labelled tweets needed to assign a certain label to a user.

7 Conclusions

The first almost obvious outcome of our work was the fall of the accuracy introducing a third label to predict: as we expected and it is easy to understand, the number of labels and the performances of a model are inversely proportional.

As regards lemmatization, there aren't enough benefits to implement it necessarily: we want to stress again, that this also depends on the Italian language and the lexicon used on Twitter, full of mispelled words and abbreviations that makes it really hard to lemmatize in an optimal way.

Another interesting outcome was that the best models were those predicting on single tweets, rather than on single users, because allows the model to focus on fewer words when predicting and consequently to have a better overview of them.

The results on the models we introduced in Section 5 were pretty good and satisfactory, especially those of the overall best model (the one stratified, without lemmatization, without *NC* users and predicting on the single tweets) that, with both the thresholds (75% and 100%), had an accuracy close to 80%. Unfortunately, despite this fairly good results, the analysis on the Movimento Cinque Stelle users was a bit destabilising, since we had opposite outcomes comparing stratified models and upsampled. We expected similar percentages of left and right users, while our best models tend to classify them or close to the right-wing parties or to the left-wing ones.
It is quite hard to understand the deep reasons of this behaviour, but for sure, one of the main causes can be found in the initial imbalance of the dataset: having too few *SX* users gives us models that, even if stratified or upsampled, might misclassify some profiles. In fact, both *stratification* and *upsampling* are techniques to oppose the imbalance, but they can not erase it completely. So, eventually, increasing our data and the percentage of left-wing users may lead to a better model: not just in terms of accuracy, but even in terms of analysis of Movimento 5 Stelle supporters.

As a final conclusion, we want to highlight how much the different techniques applied to split the dataset can influence the performances of our models: it suffices to consider that the bests between those stratified had around 10% of accuracy more than those upsampled, a clear evidence in favor of paying the highest attention possible to the way we create the training and the test set.

Appendix A K-fold Cross Validation

In appendix to our work, we decided to insert a quick overview of other models obtained varying the same features introduced previously, but implementing another splitting technique of the dataset: the *k-fold cross validation*. We decided to put it here in the appendix, because we obtained biased results that can anyway be an interesting example of what may go wrong with an improper split of training and test set.

This method consists in dividing the total dataset into k parts of equal size and, at each of the k steps, the k-th part of the dataset becomes the validation set, while the remaining parts constitutes the training set; in our specific case, k was set equal to 5 in order to have a training set with 80% of the data and a test set with the remaining 20%. In this way, the model is trained for each of the k parts to avoid problems of overfitting, but also of asymmetric sampling of the training dataset, typical of the division of the dataset into two parts.

The k-fold cross-validation method has been applied to the eight different configurations of our dataset linked to the presence or absence of users classified as *NC*, to the execution of the analysis on a single tweet or on the overall tweets posted by a person, and, finally, on the application or not of the lemmatization. Then, these eight different configurations have been applied both to the dataset with an agreement threshold of 100% and 75%. Overall, we trained a total of 80 models, 5 (one for each fold) for every of the 16 possible combinations of our features; however, we reported in the following table just those with the best accuracies varying the three parameters *NC*, *Lemmatization* and *Person*.

Parameters			**Threshold** = 75%	**Threshold** = 100%
NC	**Lemmatization**	**Person**	**Accuracy**	**Accuracy**
✗	✗	✗	0,943	0,801
✗	✗	✓	0,971	0,971
✗	✓	✗	0,899	0,949
✗	✓	✓	0,970	0,970
✓	✗	✗	0,609	0,611
✓	✗	✓	0,648	0,621
✓	✓	✗	0,621	0,623
✓	✓	✓	0,645	0,679

Table 7: k-fold Cross Validation Results

Clearly, the two highest accuracies, highlighted in red, are extremely close to 1 and, because of this, extremely suspicious: it is very hard, in fact, to obtain such a high score. These results can be explained, once again, by the imbalance of the dataset: in fact, it allowed the train and test sets of the various folds to have any kind of percentage of right or left users; hence *fastText* tended to produce models oriented towards one of the two labels. For example, if the train set is made of almost only right-wing supporters, the model will be trained to classify everyone as a right-wing supporter and as a consequence will classify very few left-wing supporters. This last behaviour is exactly the one of our best models when it came to the prediction of *M5S* users: these models, in fact, associated the *DX* label with the 99% of the users for the first model (75% threshold) and with the 96% for the second one (100% threshold); hence only the 1% and the 4% of the users were associated with the *SX* label, respectively.

This simple example shows, once again, how much it is important to have a balanced dataset: even if the data are correctly and highly preprocessed, the results can be overturned by their improper splitting into training and test set; moreover, we want to stress how the accuracy, even if it is for sure the main index of the goodness of a model has not to be overlooked and considered by itself.

References

[1] Armand Joulin, Edouard Grave, Piotr Bojanowski, Tomas Mikolov, *Bag of Tricks for Efficient Text Classification*, arXiv:1607.01759 [cs.CL] (2016).

[2] Jake van der Plas, *Python Data Science Handbook*, O'Reilly Media (2016).

[3] Lemaître, G. Nogueira, F. Aridas, Ch.K., "Imbalanced-learn: A Python Toolbox to Tackle the Curse of Imbalanced Datasets in Machine Learning" in *Journal of Machine Learning Research*, vol. 18, no. 17 (2017), pp. 1-5.

[4] Nàdia F. F. da Silva, Eduardo R. Hruschka, Estevam R. Hruschka Jr., "Tweet Sentiment Analysis with classifier ensembles" in *Decision Support Systems*, vol. 66 (2014), pp. 170-179.

[5] `https://www.clips.uantwerpen.be/pattern`

[6] `https://fasttext.cc/`

[7] `https://towardsdatascience.com/fasttext-sentiment-analysis-for-tweets-a-straightforward-guide-9a8c070449a2`