Predicting Political Orientation From Tweets

Cognitive, Behavioral & Social Data
AY 2019-2020

UNIVERSITÀ DEGLI STUDI DI PADOVA

Prepared by:
Alessandro Padella
Angelica Ayvazian
Giacomo Rotondi
Marco Sommaruga
Matteo Posenato

Abstract

Before the availability of data and the beginning of the era of machine learning processes, social sciences such as sociology, linguistics, politics and psychology relied on observation and human judgment when making decisions which resulted in opposing theories found by specialists in the same field. However, it is important to cross-reference these decisions with some quantitative and statistical methods, and even going further to construct machine learning models in order to have reliable evidence for the inferences made in these fields.

We will apply this approach to the Italian political system: our goal is to be able to classify a Twitter user as belonging to a left political party or a right political party by analyzing their Tweets. In our study, we will start from scratch: we'll collect the data, preprocess it, clean it, plot the training data into algorithms of our choice using Python and analyze the findings. Later, we will apply these algorithms to our test data in order to evaluate the accuracy of each model and choose the one that's most convenient with our purposes.

All the codes used in this report are available in the folder below:
https://drive.google.com/drive/folders/1nopbk9T6MxN37iTnHQjN7PSrZWHlyjWK?usp=sharing

Report Outline:

Introduction

With the increase of people’s presence on social media, there are many studies working towards making statistical inferences about psychological aspects of social media users, in order to use this data for various purposes. These purposes may range from marketing, cultural studies, behavioral studies, detecting psychological disorders, conducting studies on their personalities. The focus of our research will be on using data present on social media - more specifically on Twitter - to classify the political affiliation of Italian citizens.

Although the overall use of Twitter is low in Italy when compared to Europe, this number has increased by 138% from 2010 till 2012(1). According to a 2012 Pew survey, among the Italians who are active on social media platforms, 36% use these placements for political purposes(2).
That is why, we believe that this social media channel is an important placement to collect data that we can use as independent variables in predicting an Italian citizen’s political preference.

While forecasting a range of highly sensitive personal attributes, some people refrain from sharing such information overtly believing that it is an invasion of their privacy. That is why, we will work towards coming up with a Machine Learning Algorithm that classifies whether an Italian citizen leans towards a left or right political party even if they only express it implicitly.
Since the agendas of political parties and their stance on certain social norms convey a certain message as to what their supporters’ persona looks like, the classification of Italians in terms of their political affiliation can be deducted by studying the subjects’ personality traits and their opinions on certain topics.

In general, people may make personal judgments, but the drawback is that there are many biases that might affect the decision of an individual along the way. There are measures that we can take in order to minimize the risks associated with these biases, such as having individuals from different backgrounds making a personal judgment about each user and then comparing it with the rest, which makes up for a collective classification with a certain percentage of agreement. However, this requires a lot of effort and time, and while it diminishes the biases, it doesn’t entirely eliminate them. Which calls for our need for Machine Learning algorithms that can make these decisions about personality and behavior for us, in bulks, with much less time needed and with no human bias whatsoever(3).

Throughout the algorithm modeling process, we considered texts conveyed via Tweets as predictors of personality and behavior(4). Written language is one of the aspects that could predict one’s behaviors and beliefs. Meaning that the vocabulary that people use (whether it consists of just words or a combination of words - this issue will be addressed later in our study) is a stylistic behavior.

With written texts, we don’t have the luxury to resort to body language and tone of voice in order to make implicit predictions on a subject’s statement. Instead, we can resort to detecting patterns in the lexicon chosen by someone concerning various topics (not just the one we’re interested in, which is politics in our case) in order to take into account both implicit and explicit messages and trends in the text that will help us classify them in terms of social and individual matters. It is important to note here that many studies have found that there are patterns in the interaction between members of the same political parties that can be used as benchmarks to classify new subjects, this arises from a phenomenon called “level of party cohesion”(5).

This is why we decided to keep all the tweets in our dataset, even if they were not related to politics, they could provide us with meaningful insight into the subject’s personality and indirectly help us in predicting their political preference.

Another issue to take into account in our study is the language problem. Users on social media express their thoughts using various languages. Many of the Tweets we collected contained words from other languages, which makes it quite difficult to apply tools built using language extraction features only to apply it to another language(6). However, after conducting an analysis of the most used words, we realized that they’re all in Italian, and so were most of the Tweets.

Phase 1: Data Collection

Before going into details about each of the data collection methods, it is important to note that each of the three methods mentioned below has its limitations.
The combination of all three methods creates a basis for a stronger cluster of data, because the risks associated with each technique are decreased with the introduction of the others.

The methods we used to complete the first phase of our study are the following:

- Distributing Online Surveys
- Party’s profiles on Twitter
- Random Twitter Profiles

Distributing Online Surveys

The first method we used to collect data was an explicit data collection method which was a survey we distributed to Italians (via social networks) while diversifying their backgrounds in order to address a sample that correctly represents the Italians. That is why, we tried to expand the survey recipients in terms of:

- Demographics (age & gender): the generation to which users belong to could be highly correlated with their openness concerning some social issues which are amongst the main topics in most of Italian political parties' agendas.
- Geographical locations: the region in which the users were raised in highly affect the cultural and social beliefs which in turn affect the political preference.
- Educational level

Generally, online surveys are the least time-consuming and most flexible data collection methods in which you can ask direct questions or indirect questions from which you can make conclusions. Additionally, the data collection part is a lot simpler(7).
At the end of the questionnaire, we asked respondents to provide us with their Twitter account, we then collected their Tweets in phase 2.
However, after sending out the survey to our acquaintances and posting the questionnaire on politically active social media groups, as well as other groups containing Italians, we still reached only a limited amount of respondents. This data (434 answers, of whom just 25% provided us with a twitter account) is not enough for us to feed our algorithms and make accurate predictions.

Party's profiles on Twitter

The second method we resorted to and which required a lot of effort and time that we put into was collecting data directly from 5000 users who interacted with profiles associated with Political parties.

First, we researched pages associated with each of the five main political parties we took into consideration in our research (Lega, Partito Democratico, Movimento 5 Stelle, Forza Italia, Fratelli d'Italia). The next step was scrolling through the followers of each page and going on their individual profiles. On the individual profiles, we observed the tweets and the interests of each user then classified them on the basis of their account activity, after making a personal judgment.
This judgment was made by 5 different people, which resulted in a dataset with different agreement levels (25% - 100%).
In our study, we only considered the users with 75% or 100% of agreement in order to avoid subjects that are hard to classify and also refrain from feeding our algorithm with possibly wrong data stemming from personal judgment or even bias which cannot be eliminated(3).

Random Twitter Profiles

The third method we used to create our database needed even more time and effort. We went through Twitter profiles and scrolled through the tweets of 3000 Italian users chosen at random.
The issue here was that some of these subjects didn't even have tweets related to political subjects, that is why it was a very critical task that required a lot of background knowledge on the Italian political parties, their agendas and the persona of their usual followers. Typically, it is easy to recognize that this method relies heavily on personal judgment(3).

As with the previous task, five people made a personal judgment as to which political party each user affiliates with. The latter step was critical in overcoming the main drawback of this data collection method.
As mentioned beforehand, we only took into consideration the users for which 4 or 5 of the judges classified the Twitter profiles in the same political party.

Phase 2: Downloading Tweets

After having collected all the users' twitter profiles and classified them, the remaining step before beginning to analyze the data was downloading the tweets. We downloaded the tweets belonging to the labelled users through a Twitter Developer account.

Since we decided to analyze users labelled with an agreement higher than 75%, we only took into consideration the filtered dataset and downloaded their tweets using an algorithm (refer to the code "downloading tweets.ipynb"). This algorithm directly scrolls through every user's profile and downloads all their tweets. During this step, it is important to decide what is the temporal interval to consider for the tweets extraction: in our study we decided to analyze all the tweets available for every profile in order to have the largest amount of data possible. Also, we wanted to include the tweets with the high period of political activity in Italy, meaning the time intervals preceding the elections (2013 & 2018).
Prior to these dates, we didn't have much data, so we decided to leave them anyways.

Some users were excluded in this phase because their accounts weren't recognized or fake.
This minimized the errors in classification which could have arisen from feeding the algorithm with incorrect information and affected the accuracy of our models.

Phase 3: Data Analysis

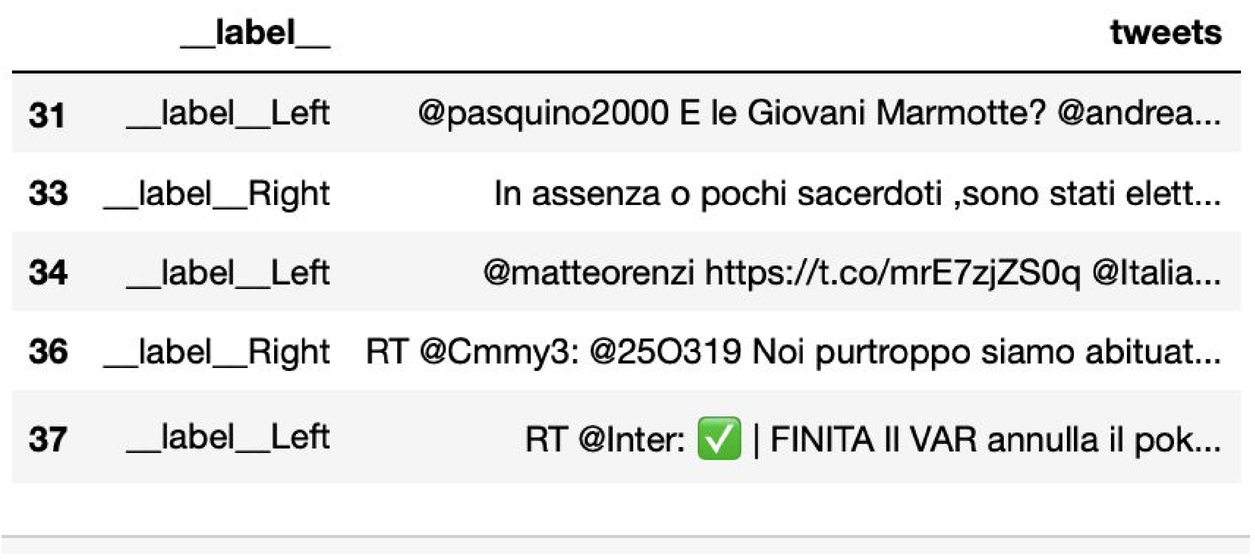
Measures Taken

Building the DataFrame

Once we collected the data, we proceeded with creating a data frame containing all the tweets of the labelled users (refer to the code “Importing tweets.ipynb”) . The first goal is to predict in a supervised method (technique that allows to train an algorithm using labelled data) the political orientation as *‘Right’* (‘Right’ labelled users: Lega, Forza Italia, Fratelli d’Italia supporters) and ‘Left’ (‘Left’ labelled users: PD supporters). So the data structure we created is made of two variables (columns): one contains all the tweets of a user while the other contains the users’ label ‘Right’ or ‘Left’.

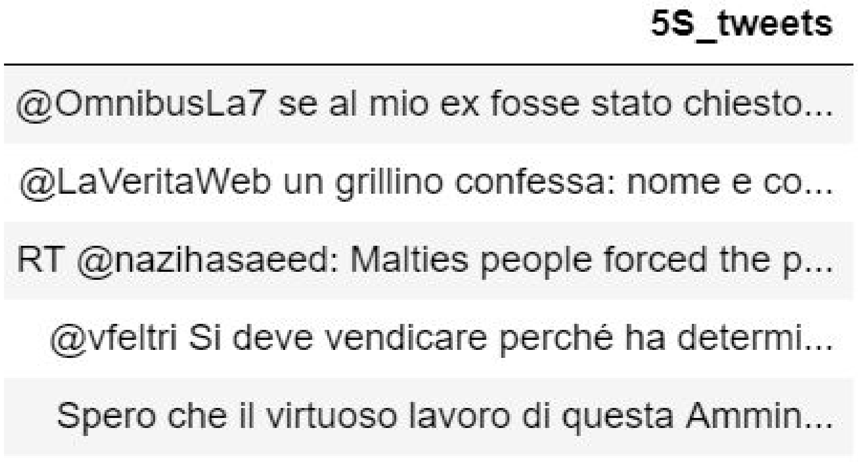
This procedure is repeated twice, once for users whose agreement rate on political orientation is 100% (all 5 members of a group that scrolled the user’s account agree on his political orientation) and once for users whose agreement rate is 75%. This step is made to account for the fact that human judgement can be erroneous (as we will see, the results for these two trials will be comparable).

In order to work with homogeneous data and not to train any model in a biased way, the number of users is class balanced (1100 users labelled “Right” and 1100 labelled “Left”) .



	__label__	tweets
31	__label__Left	@pasquino2000 E le Giovani Marmotte? @andrea...
33	__label__Right	In assenza o pochi sacerdoti ,sono stati elett...
34	__label__Left	@matteorenzi https://t.co/mrE7zjZS0q @Italia...
36	__label__Right	RT @Cmmy3: @25O319 Noi purtroppo siamo abituat...
37	__label__Left	RT @Inter: ✅ | FINITA Il VAR annulla il pok...

The second goal is to classify Movimento 5 Stelle's users as 'Right' or 'Left' user based on the results of the supervised classification; so the data structure for M5S is made of only the column of the tweets.



5S_tweets
@OmnibusLa7 se al mio ex fosse stato chiesto...
@LaVeritaWeb un grillino confessa: nome e co...
RT @nazihasaeed: Malties people forced the p...
@vfeltri Si deve vendicare perché ha determi...
Spero che il virtuoso lavoro di questa Ammin...

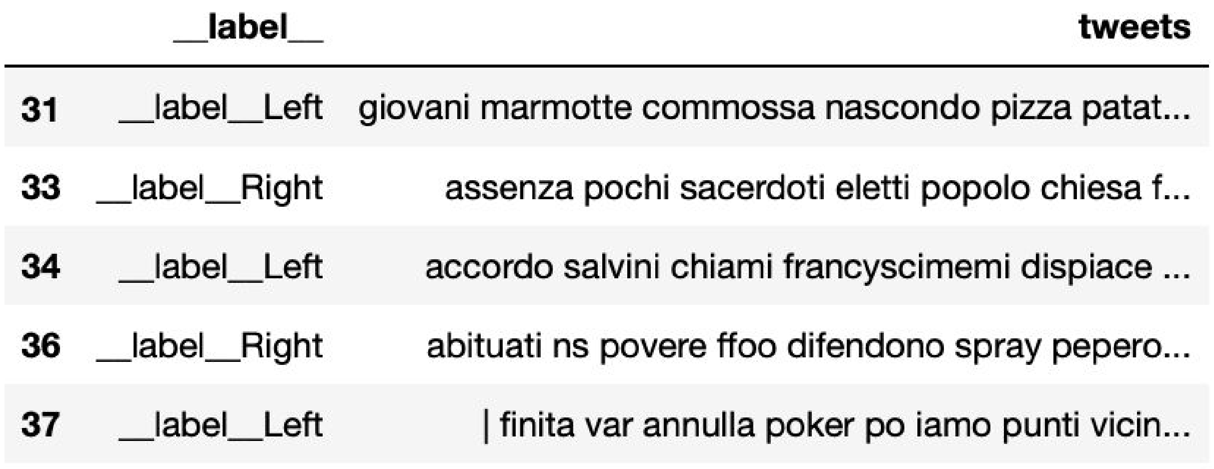
This is quite interesting and meaningful, since Movimento 5 Stelle is a relatively recent party and a subject to many controversial political debates. This party claims not to belong to the typical definition of right and left parties (8).

Text Pre-processing

One of the major parts of the whole analysis is the pre-processing phase: the objective of this process is to get rid of all the noisy parts of the text. This procedure is common to every kind of text analysis and it is especially important if one works with Twitter data. Tweets come in a very messy shape which that negatively affects the learning and prediction ability of a machine learning model(6).

We used the module **nltk**(15) (refer to the code "cleaning the data.ipynb") available on python in order to tokenize (i.e. dividing a string into single words) the tweets and we remove the following texts that we thought would mislead our algorithms:

- HTML and URL links;
- Emoticons;
- Mentions (e.g. @matteosalvinimi) and hashtags;
- Numbers and punctuation.



	__label__	tweets
31	__label__Left	giovani marmotte commossa nascondo pizza patat...
33	__label__Right	assenza pochi sacerdoti eletti popolo chiesa f...
34	__label__Left	accordo salvini chiami francyscimemi dispiace ...
36	__label__Right	abituati ns povere ffoo difendono spray pepero...
37	__label__Left	| finita var annulla poker po iamo punti vicin...

After removing the elements mentioned above, we were left with only words, from which we cut off the so-called ‘stop words’. The ‘stop words’ (stop_words.csv) are terms that do not influence the meaning of a sentence such as pronouns, prepositions and articles. We take into account Italian, English, French, Spanish and German stop-words. After scrolling through the tweets and coming across some glitches that might affect the performance of our models, we conducted a bit of manual work in order to get rid of frequent typos or abbreviations.

After the cleaning task, the dataset was free of input that might have decreased the accuracy of our models and it was ready to be processed.

(In the folder there are available all the data frames. Refer to:
- data_75_cleaned.pkl
- data_75_fasttext.pkl (this is a copy of data_75_fasttext but with a different format of the labels)
- data_100_cleaned.pkl
- data_m5s_cleaned.pkl)

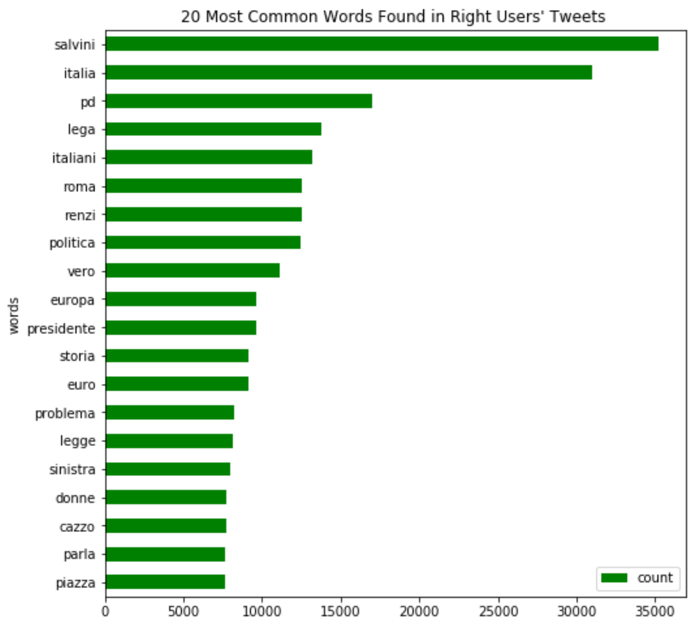
Feature Extraction

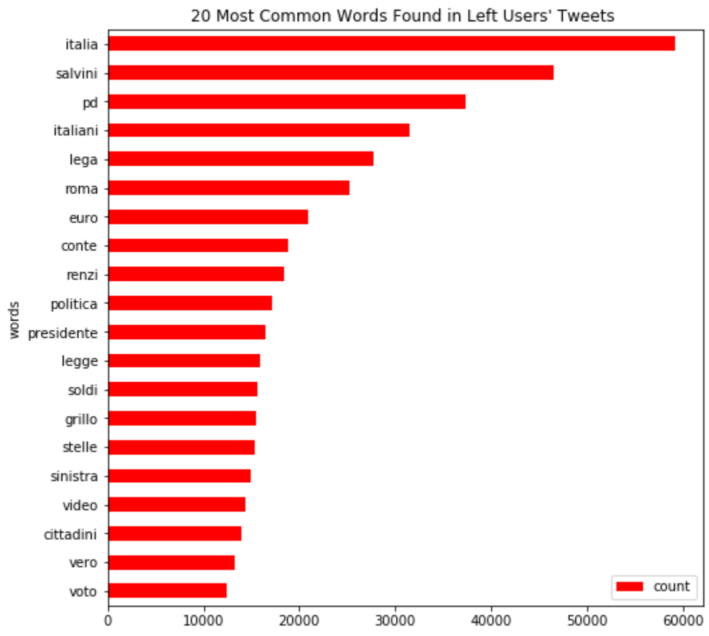
The next step is to rearrange the data in a shape that can be used as input for a statistical model.

The features in a text analysis are clearly words that may or may not be in the tweets.

Roughly speaking, machine learning algorithms run some kind of optimization problem on a set of matrices and vectors. They do not really understand "raw text", which means that it is required to convert the raw text into numerical metrics.

Applying tokenization(16), we get a list of all the words used in the tweets (refer to code “Word Analysis.ipynb”), which constructs our vocabulary, that is the total set of features we may use. We can now visualize which are the most used words in the tweets by the right and left users.



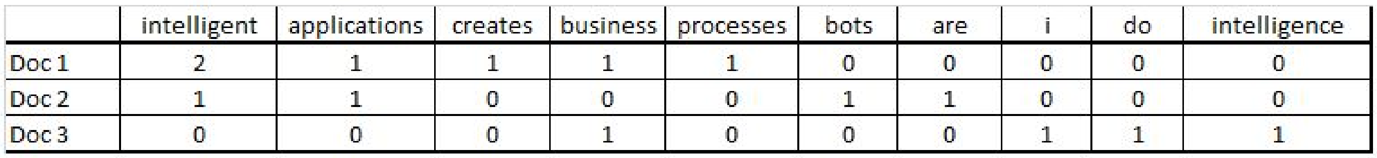


The six most used words are the same for the two groups, pointing out that used words alone, stripped of the context, are not much informative about the political orientation of a Twitter user. A word alone will fail to detect irony, sarcasm or even the meaning of a sentence as a whole (such as negative sentences, questions and so on). (4)

It is interesting though to see that Salvini is the most cited politician not only by right voters, but also by left voters, who probably criticize him. However, this shows that he is at the center of topics debated by the public.

The same word frequency analysis is done for users labelled with 75% of agreement rate and the rank of the most used words is practically the same for both groups.

The structure that we now need to create is the so-called document-term matrix (DTM), which looks like this:



	intelligent	applications	creates	business	processes	bots	are	i	do	intelligence
Doc 1	2	1	1	1	1	0	0	0	0	0
Doc 2	1	1	0	0	0	1	1	0	0	0
Doc 3	0	0	0	1	0	0	0	1	1	1

Each row of the matrix still represents a user and the columns represent all the words of the vocabulary.

The matrix can be filled with different metrics:

- Binary value, representing the presence/absence of the word in the user's tweets;
- Frequency of each word in the user's tweets;
- Term frequency - inverse document frequency (TF-IDF) of the word in the document.

After various studies conducted on TF-IDF, it is proved to be the most reliable statistical metric for text prediction and text analysis in general. It is the product of the following: (17)

- ***TF(w) = (Number of times the term w appears in the document) / (Total number of terms in the document)***
- ***IDF(w)= log(Total number of documents/Number of documents in which term w appears)***

Hence, the TF-IDF score increases with the number of occurrences within a document and with the rarity of the term in the collection, giving less weight to words that are common across the documents.(17)

In Python, TF-IDF scores and the associated document-term matrices are provided by the TfidfVectorize function of the *sklearn.feature_extraction.text* module.

The same approach is used with ‘bi-grams’ features, which consist of sets of 2 consecutive words.

In our codes, we decided to train and test the algorithm using the 5000 n-grams with the highest score given by TF-IDF.

Now that the data is shaped in a proper form, it is ready to be processed with a machine learning algorithm. It is split into two sets: 80% of the data frame is used as training set and its aim is to train the algorithm, the other 20% is the test set, used to assess how accurately the model will be able to predict the political orientation of new users.

After having studied the available machine learning models (11),(12) and looking into both their advantages and disadvantages, we choose the ones we thought were the most aligned with our dataset and our goals.

The models that we used are the following:

- Support Vector Machine
- NaiveBayes
- Logistic Regression

In addition to these, we use a model, called fastText, that carries out text classification.

Machine Learning Models

Support Vector Machine

(refer to the codes:

- SVM_and Naive_Bayes_Monograms 75.ipynb
- SVM_and Naive_Bayes_Monograms 100.ipynb)

Support-vector Machine(SVM) is a supervised learning model with associated learning algorithms that can be used in classification tasks (9). SVM maps each data point in a p-dimensional space where each dimension represents a feature. SVM then fits the (p-1)-dimensional hyperplane that represents the largest separation between the two classes (Right and Left).
SVM is particularly effective in high dimensional spaces, even if the number of dimensions is greater than the number of observations which is true in our case.

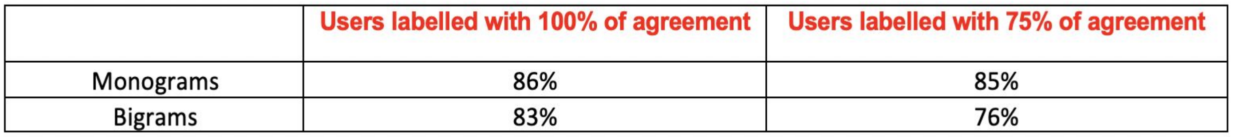
Application

The sklearn Python module provides the functions to apply the SVM model to our data.

A further pre-processing step is done before building the document-term matrix: ‘lemmatization’(16), whose purpose is reducing the inflectional forms of each word into a common base or root.

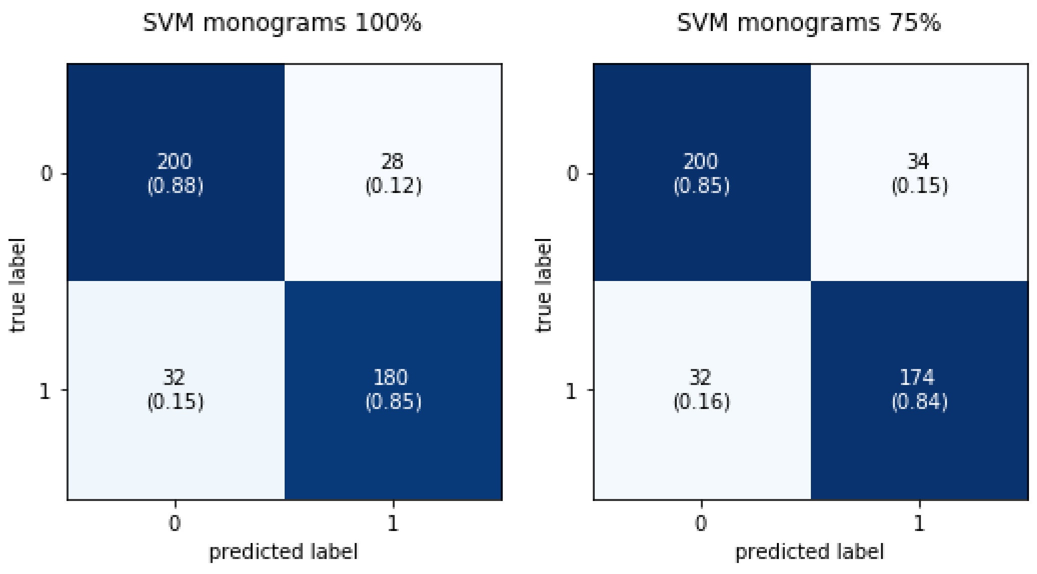
The next step was to apply a grid search to tune the best hyperparameters in order to fit the model to the training data (80% of the total collection of tweets) and later apply it to the remaining test set to see its accuracy in classifying new subjects.

Prediction Accuracy of Support Vector Machine:



	Users labelled with 100% of agreement	Users labelled with 75% of agreement
Monograms	86%	85%
Bigrams	83%	76%

The best prediction accuracy is landed using monograms as features in both cases; one reason for this result could be the fact that tokenizing the sentences in bigrams and not in monograms. We found patterns that are more difficult to classify between left and right, while a single word can give more information about the political orientation of a user. We can take a deeper look into these predictions and plot the so-called confusion matrices for these two methods. This simple structure shows the proportions of correct and wrong predictions within each class.



<u>**Legend:**</u>
0 – “Label Right” users
1 – “Label Left” users

Both the methods classify the right voters more precisely than the left ones.

We can notice that the results are similar, testifying that there aren’t significant differences in the users classified with 100% of agreement and the ones with 75%.

NaiveBayes

(refer to the codes:

- SVM_and Naive_Bayes_Monograms 75.ipynb
- SVM_and Naive_Bayes_Monograms 100.ipynb)

Naive Bayes is a simple probabilistic classifier relying on Bayes' theorem(9):

$$\underset{\text{Posterior Probability}}{P(c \mid x)} = \frac{\overset{\text{Likelihood}}{P(x \mid c)}\,\overset{\text{Class Prior Probability}}{P(c)}}{\underset{\text{Predictor Prior Probability}}{P(x)}}$$

$$P(c \mid \mathrm{X}) = P(x_1 \mid c) \times P(x_2 \mid c) \times \cdots \times P(x_n \mid c) \times P(c)$$

The aim of this model is to estimate the conditional probability of belonging to a labelled class, in our case right or left voter, given a specific combination of the values of the features, in our case TF-IDF values. The predicted class is then the one with the highest estimated conditional probability between the two.
The main drawback of the Naive Bayesian Classifier lies in its strong assumption that within a document, the probability of observing a word is independent of that of others, which is quite implausible in the context of a written text.
To try to overcome the issue reflected by this assumption and moderating the risks presented by it, we applied this model using bi-grams as features as well.

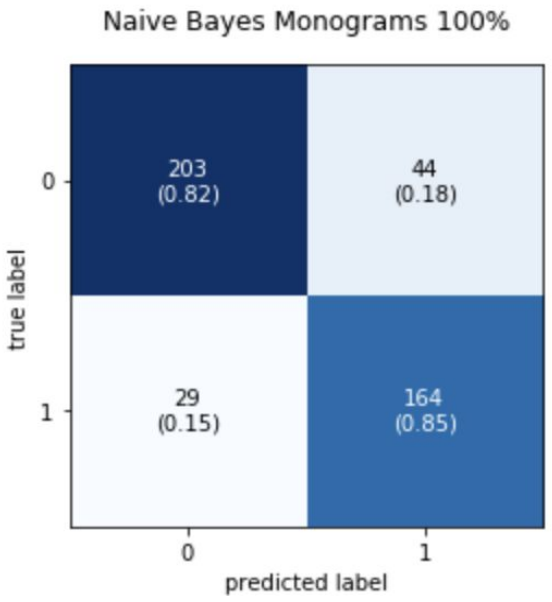
Application

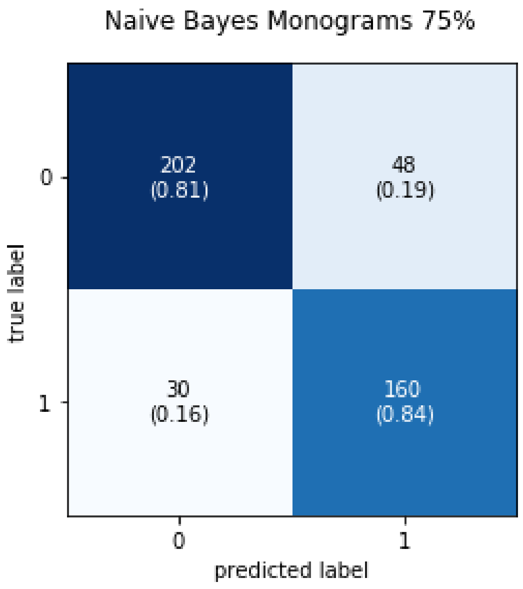
After applying lemmatization, we built the DTM with TF-IDF for both single words and bigrams, fit the simple model to our training set (still made of 80% of the collection of tweets) and used the results to predict the tweets belonging to the test set.

Prediction Accuracy of Naive Bayesian:

	Users labelled with 100% of agreement	**Users labelled with 75% of agreement**
Monograms	83%	82%
Bigrams	79%	79%

Unexpectedly, using the Naive Bayesian classifier, the prediction is more accurate in the dataset characterized by monograms, as it was with SVM. Let us then look into their confusion matrices showcased below:





<u>**Legend:**</u>
0 – "Label Right" users
1 – "Label Left" users

In this case, it can be noticed that the accuracy is higher for the left users, unlike the previous method. Even with the NaiveBayes classifier, it's evident that the results are similar for the two levels of agreement (75% and 100%). We can thus conclude as we did with the SVM method: the classifications with the two different levels of agreement have users that can be similarly labelled.

Logistic Regression

(refer to the code:
- logistic_regression.ipynb)

Logistic Regression is a probabilistic statistical method that uses a logistic function to model a binary dependent variable(10). To do so, it estimates the parameters of a logistic model. A linear relationship between the predictor variables, and the log-odds of the event taking one of the values is assumed: this relationship can be written in the following mathematical form (where ℓ is the log-odds, *b* is the base of the logarithm, and ß□ are the parameters of the model.

$$\ell = \log_b \frac{p}{1-p} = \beta_0 + \beta_1 x_1 + \beta_2 x_2$$

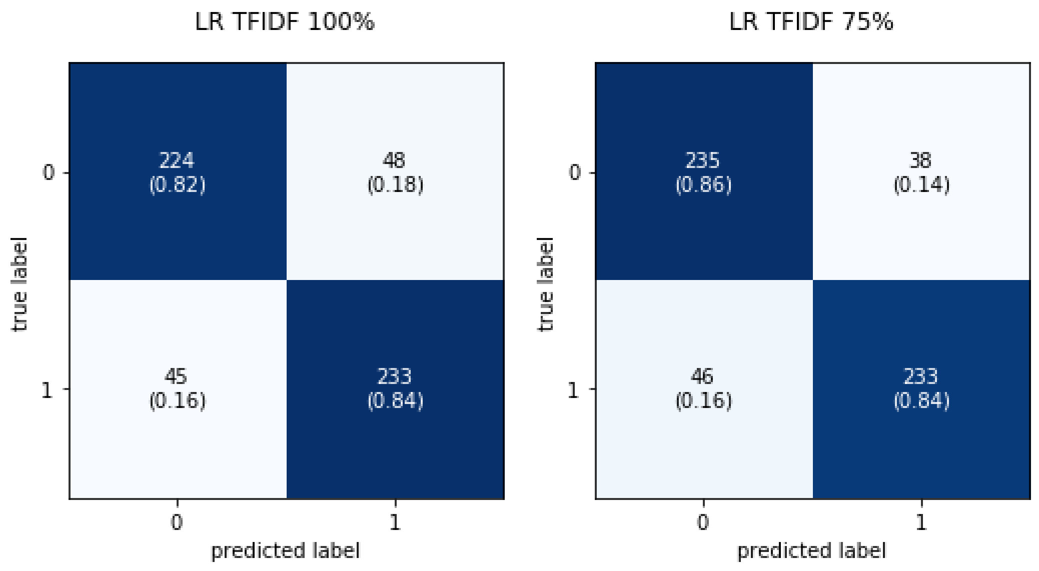
Application

Logistic Regression in our case predicts the probability that a collection of tweets was written by a right or left user using the TF-IDF values of words, binary values expressing the presence or absence of the word in the document or the frequencies of words in the document .
Logistic Regression provided by **scikitlearn** also allows us to penalize the magnitude of the parameters(18).
With a grid search, we tuned the best hyperparameters in terms of future prediction accuracy, then fitted the model using the best combination of hyperparameters.

Prediction Accuracy of Logistic Regression:

	Users labelled with 100% of agreement	Users labelled with 75% of agreement
Binary	82%	82%
Frequency	83%	78%
TF-IDF	83%	85%

Quite surprisingly, running the model using a simple binary variable 0/1 in the 75% dataset performs better than using the frequency of words. As expected, tf-idf features are the ones that land on better accuracy. Below are the TF-IDF confusion matrices providing more insight on this model:



<u>**Legend:**</u>
0 – "Label Right" users
1 – "Label Left" users

Models with TF-IDF values work as better for the two classes and for both datasets.

FastText

(Refer to the codes:
- fasttext_75.ipynb
- fasttext_100.ipynb)

FastText(13) is a model that analyzes textual data, it can also be used for text classification. Although its architecture is quite complex, it uses a Multinomial Logistic Regression model for estimating the probability that a piece of text belongs to two or more classes. Due to this complexity, we have to switch to Google Colab(14). Google Colab is a hosted Jupyter notebook service that requires no setup to use, while providing free access to the computing resources we need.

Application

Like the models illustrated above, fastText maps each piece of text and each label as a vector in space; words are then reduced to low-dimensional vectors called embeddings. Word representations, which are averaged into text representations, are fed into a linear classifier, which is essentially an enhanced linear model.

There are various hyperparameters that need to be set:
- **Epoch:** the number of times fastText takes a look at each data point. Our model performed best with an epoch of 50.
- **Learning rate:** controls how "fast" the model updates during training. This parameter controls the size of the update that is applied to the parameters of the models. We kept the learning rate at a default value which is 0.1.

Prediction Accuracy of fastText:

We ran fastText both with our two datasets: 100% and 75% agreement.
The results obtained are as follows:

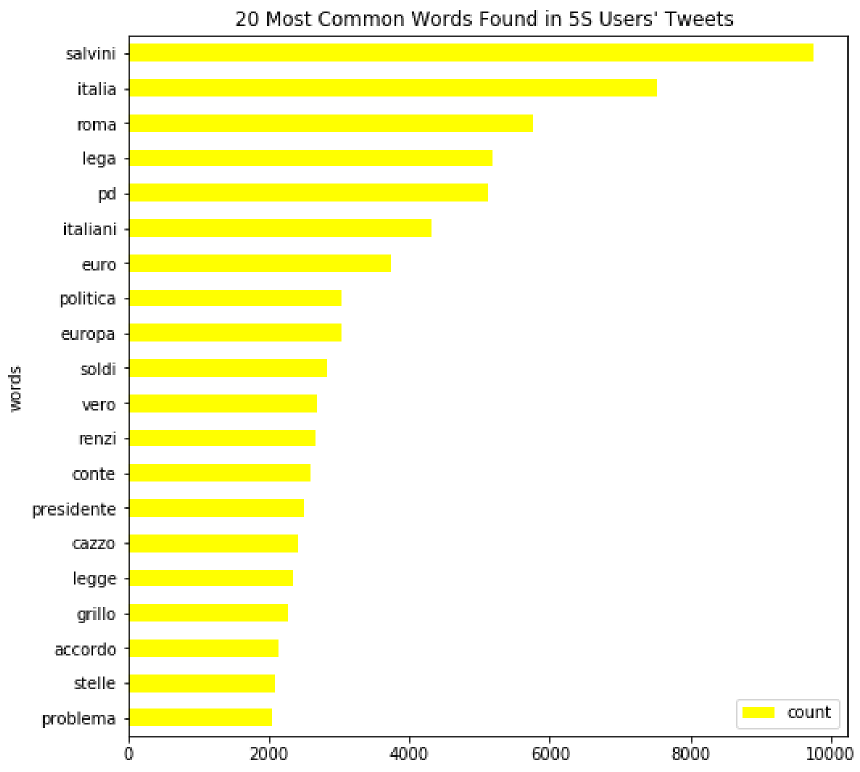
Users labelled with 100% of agreement	Users labelled with 75% of agreement
81%	83%

From the table above, it is confirmed that with the fastText algorithm, we conclude the following: the two dataset are classified with a comparable accuracy.
There can be different reasons to motivate this similarity: one reason is that it is plausible that the users labelled with the 75% of agreement are actually well classified, indeed a 75% of agreement means that these users have a high probability to belong to the left or right party. Another reason is that the vocabulary used by both the groups is similar and, even if the users are classified with uncertainty, they can be classified with the same accuracy by the algorithm between right and left parties.

Further Studies: Classifying Movimento 5 Stelle voters

(Refer to the codes:
- Word Analysis.ipynb
- LogisticRegression_m5s_classifier.ipynb
- SVM_m5s_classifier.ipynb)

The last step of the project was trying to use the results obtained with the right/left dataset to classify the followers of Movimento 5 Stelle.
First of all, we visualized the 20 most frequently used words within their tweets, as done before for right and left voters.



.

The fact that the most used words are the same as left and right voters is quite encouraging for going forward with the classification. Because, as mentioned beforehand, this political party doesn't associate itself as a right party, nor does it associate itself as a left one.

We used the model based on LogisticRegression to try to predict the political orientation of Movimento 5 Stelle users and we found the following results:

- Using Binary method: **50.4%** were classified as left and **49.6%** as right
- Using Frequency method: **63.9%** were classified as left and **36.1%** as right
- Using TFIDF: **100%** were classified as right

It can be seen that the Binary method provides us with a balanced classification of the Movimento 5 Stelle supporters. This result confirms the claim of the supporters of this political party, who say that they don't belong to either political sides.
This result is given creating a vocabulary with each word of the M5S tweets, fixing a binary value (0 or 1) to each. Later, these are compared with the words used to train the algorithm (i.e. the words extracted from the right and left labels classification).

The second method gives a classification that is rather unbalanced, leaning towards the left party. This outcome might be due to the fact that M5S users use words with a frequency that is more similar to the right labelled users rather than the left ones. This hypothesis calls for further studies in order to be verified.

Finally, the TFIDF method classifies all the users as "Label Right".
The problem that arises with this method, and the reason why we obtain such a strange result, is the fact that with the TFIDF method, the algorithm creates a matrix (in our case of 5000 elements) that contains all the words used by both the right and left users. After which, we need to match the words contained in this matrix with the vocabulary used by M5S users in order to have two matrices of the same dimensions that the algorithm is able to compare.

Our decision has been to use the SVM algorithm to reach this result, because it is the algorithm that performed the best with the TFIDF method.
What we decided to do is create a vocabulary given by the TFIDF method that contains all the words of both left and right users in addition to a vocabulary with all the words of M5S users. We then tried to eliminate all the words of M5S users that were not contained in the right and left parties' vocabulary.
However, since the actual classification requires very high computational capacity, unfortunately, we were not able to derive a result. We decided to attach our codes to this article anyways, thinking that they form a satisfactory basis for whoever is interested in classifying tweets and has the means and the capacity to run the code.

Conclusion

Throughout our carefully crafted study, we are able to reach many conclusions based on data that we collected ourselves. We noted that when you're conducting a study from scratch, you have control over the data that you're collecting, how you're collecting it and what independent variable you choose to take into consideration to test out your hypothesis. Our hypothesis is the following: You can classify a user as belonging to a right or left Italian political party by merely analyzing their Tweets.

After preprocessing our dataset and prepping it for the algorithms we choose to test out, we started with the data analysis phase of our project. Each step of this phase was explained in the report above along with the reason behind every decision we made.

All the models that we tested out provided us with a satisfying percentage of accuracy, however, we all agreed that the most promising one was the **Support Vector Machine using monograms.**
We also chose this model given the fact that it is the one presenting the least risks compared to the others while taking into account the assumptions that it is based on. Although we took some measures to counteract each drawback that came with the rest of the models by resorting to different measures we explained beforehand, we were still skeptical about the extent to which we were able to prevent these risks from affecting the accuracy of our models.

Moreover, it is interesting to consider other factors that might also help us make inferences about a person's political affiliation throughout machine learning models. We believe that another hypothesis that's worth being tested is the prediction of whether someone leans towards the right or left party by analyzing the image displayed on their Twitter account through image recognition. There are a lot of other information on Twitter that we did not put into use in our study such as the media, the likes, the personal information (location, age, bio). Further, it would also be intriguing to look into the relationship between followers, the amount of interaction and whether this has anything to say about a user's political preference. After choosing the best performing algorithms in the suggested studies, we can also work on combinations of algorithms, striving for the optimal classifier.

It goes without saying that with the publicly available data growing exponentially, it is becoming easier to understand social, political, marketing and even psychological behavior. By studying the behavior of social media users, and using data that they choose to share, and plotting them into quantitative algorithms, we are making informed decisions in fields that previously relied on observations and human judgment.

References

(1)Cristian Vaccari, Augusto Valeriani, Pablo Barberá, Rich Bonneau, John T. Jost, Jonathan Nagler, Joshua A. Tucker. (2015) Political Expression and Action on Social Media: Exploring the Relationship Between Lower- and Higher-Threshold Political Activities Among Twitter Users in Italy, Journal of Computer-Mediated Communication, Volume 20, Issue 2, 1 Pages 221–239, https://doi.org/10.1111/jcc4.12108

(2)Pew Research Center. (2012) Social Networking Popular Across Globe, https://www.pewresearch.org/global/2012/12/12/social-networking-popular-across-globe/

(3)Wu Youyou, Michal Kosinski, and David Stillwell. (2015) Computer-based personality judgments are more accurate than those made by humans, 112 (4) 1036-1040; https://doi.org/10.1073/pnas.1418680112

(4)Pennebaker, James & King, Laura. (2000) Linguistic styles: Language use as an individual difference. Journal of personality and social psychology. 77. 1296-312. 10.1037//0022-3514.77.6.1296. https://www.researchgate.net/publication/12688664_Linguistic_styles_Language_use_as_an_individual_difference

(5)Antoine Boutet, Hyoungshick Kim, Eiko Yoneki. (2012) What's in Twitter: I Know What Parties are Popular and Who You are Supporting Now! https://www.cl.cam.ac.uk/~ey204/pubs/2012_ASONAM.pdf

(6)Mounica Arroju, Aftab Hassan, Golnoosh Farnadi. (2015) Age, Gender and Personality Recognition using Tweets in a Multilingual Setting, http://ceur-ws.org/Vol-1391/57-CR.pdf

(7)Evans, J. and Mathur, A. (2005), "The value of online surveys", Internet Research, Vol. 15 No. 2, pp. 195-219. https://doi.org/10.1108/10662240510590360

(8) Fabio Bordignon & Luigi Ceccarini (2013) Five Stars and a Cricket. Beppe Grillo Shakes Italian Politics, South European Society and Politics, 18:4, 427-449, DOI: 10.1080/13608746.2013.775720 https://www.tandfonline.com/doi/abs/10.1080/13608746.2013.775720

(9)Gunjit Bed. (2018) "A guide to Text Classification(NLP) using SVM and Naive Bayes with Python", https://medium.com/@bedigunjit/simple-guide-to-text-classification-nlp-using-svm-and-naive-bayes-with-python-421db3a72d34

(10)Kavita Ganesan. "Build your first text classifier in Python with Logistic Regression", https://kavita-ganesan.com/news-classifier-with-logistic-regression-in-python/#.XiQx5y2b7_S

(11)Miguel Fernández Zafra. (2019) "Text Classification in Python",

https://towardsdatascience.com/text-classification-in-python-dd95d264c802

(12)Pranav Dar. (2019) "8 Excellent Pretrained Models to get you Started with Natural Language Processing (NLP)", https://www.analyticsvidhya.com/blog/2019/03/pretrained-models-get-started-nlp/

(13)https://fasttext.cc

(14)https://colab.research.google.com

(15)https://www.nltk.org

(16)Shubham Singh. (2019) "NLP Essentials: Removing Stopwords and Performing Text Normalization using NLTK and spaCy in Python", https://www.analyticsvidhya.com/blog/2019/08/how-to-remove-stopwords-text-normalization-nltk-spacy-gensim-python/

(17)Juan Ramos. (2003) Using TF-IDF to Determine Word Relevance in Document Queries https://www.cs.rutgers.edu/~mlittman/courses/ml03/iCML03/papers/ramos.pdf

(18)Journal of Machine Learning Research 12 (2011) 2825-2830 Scikit-learn: Machine Learning in Python http://www.jmlr.org/papers/volume12/pedregosa11a/pedregosa11a.pdf