Angie Catalina Carrillo Chappe 1796678

Kristyna Dolezalova 1807152

Social and Behaviour networks

Italian Referendum 2016

Introduction

The aim of this project is to perform analysis of twitter data in the time of Italian constitutional referendum. This referendum took place in Italy on 4th December 2016 and the bill was presented by a former prime minister of Italy – Matteo Renzi. Voters were asked whether they approve a constitutional law that amends the Italian constitution to reform the composition and powers of the Parliament of Italy, as well as the divisions of powers between the State, the regions and administrative entities. Since in referendum were two possible choices – yes (support of the bill) and no (do not support), we refer later in this report (and code) the yes supporters as positive and no supporters as negative.

Data analysis is performed over the Twitter data collected in the time period between 26 November 2016 and 6 December 2016. Additionally, a directed graph containing users and their relationships through mentions is used – this graph is later referred to as a provided graph.

This analysis consists of several steps. First part performs a temporal analysis of the obtained data (and with our data directly scraped from twitter). In this part the k-means clustering of the SAX strings is performed for different time granularities.

In second part we are identifying the most important users, influencers using different algorithms.

In third part we are performing analysis of spreading the influence with modified LPA algorithm for identified users in the previous part.

For information on how to run the code to reproduce all the steps of the analysis, please see an attached README.md file. Application code is written for Java JRE 1.7 as a Maven project using twitter4j, Apache Lucene, SAX and stilo.g libraries as additional dependencies.

Part 0 – Temporal Analysis

0.1 Time distribution

List of politicians, journalists and other influencers is obtained semi-manually from the internet. For simplicity, we will refer all influencers as politicians further in the report.

In the end we were able to collect names of 133 negative politicians and 123 positive politicians, which still have a publicly visible Twitter account.

Names were mainly found through Wikipedia and parliament's website and then semi-automatically (with the help of a simple Python script) identified user twitter accounts. Some of the users are too active on Twitter and we had to exclude them from analysis for the part 0.1, as they had no tweets to fetch. This is due to the twitter free API limitations to download only 3200 latest tweets and the referendum is quite old matter already.

This was not a problem with the provided dataset. While working with the whole dataset of scraped tweets, we were using our previously collected list of politicians to decide whether politicians belong to the positive or negative groups of supporters. For further analysis we preserve the information about time (time of the tweet publication), name, id, retweet id and the text of the tweet.

Time stamps were used to create a bar plot to compare the distribution over time. Below you can see both plots – one for our own scraped data and the second one for data from provided dataset. In total, we scraped 231 866 positive tweets and 269 664 negative tweets in the time range from 1.4.2016 till 4.12.2016. For provided data, we have 65 782 tweets from users identified as belonging to the positive group and 36 309 for negative group.

Note that for identification of the group where certain tweet belongs both information about twitter name of the tweet creator same as twitter name of user who was retweeting certain tweet were used. So, in some cases the situation may happen then the opposite team member was retweeting opponent tweet without belonging to his opinion group. The condition first checks the creators name and if he is not in a list, we proceed to the retweeter's name.

The tweets were added to a Lucene index for later usage corresponding to their political beliefs.

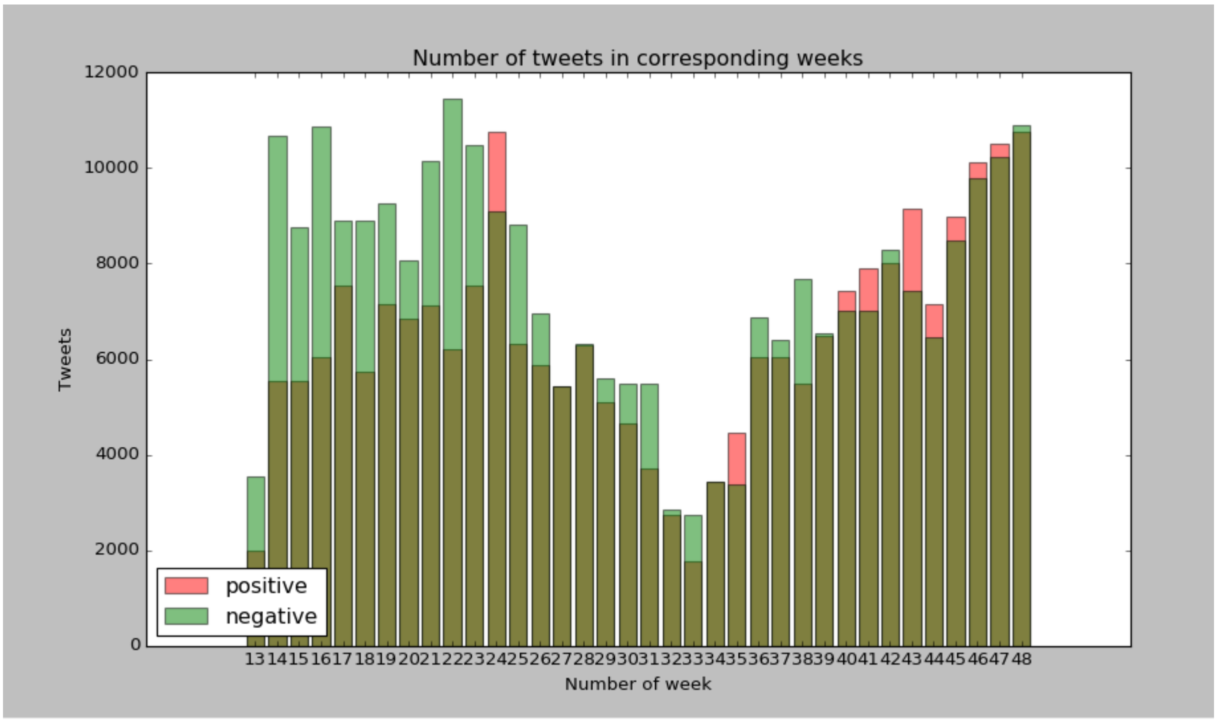

Figure 1 Time distribution of scraped data

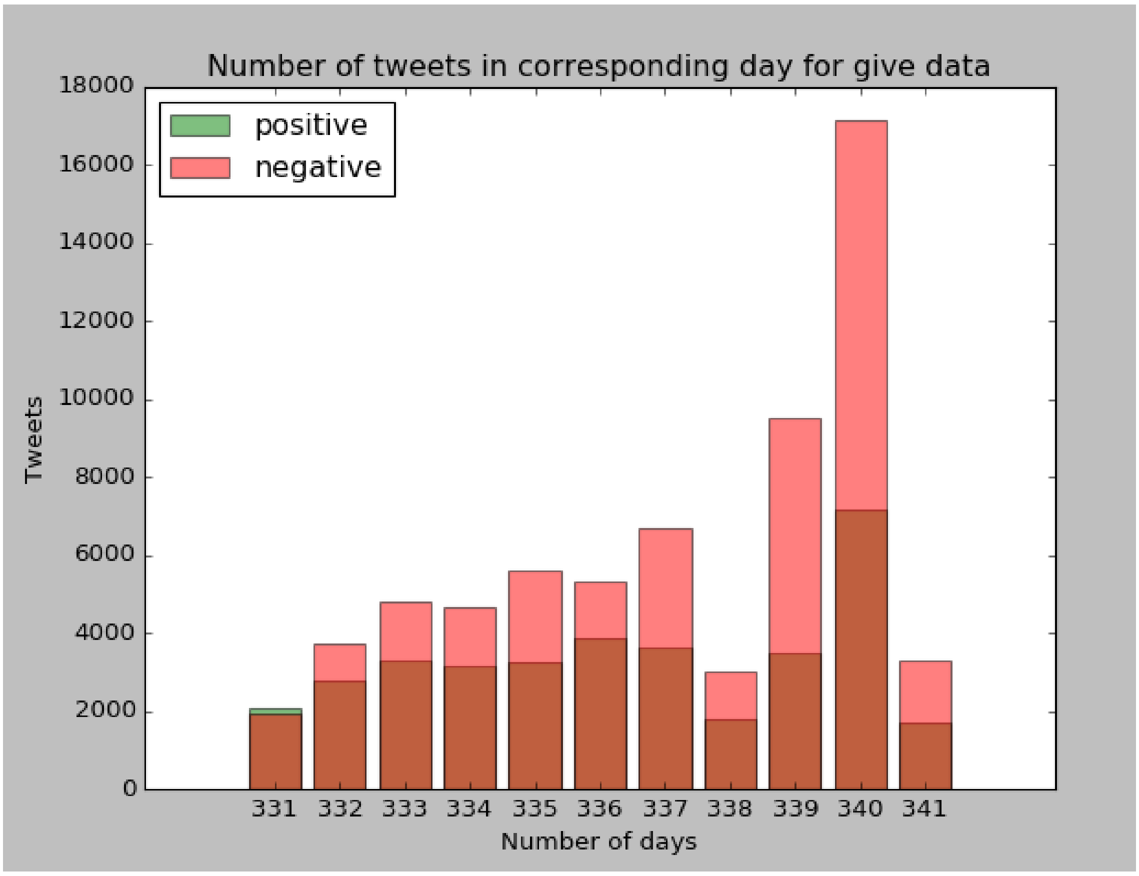

Figure 2 Time distribution for given data

0.2 Clustering

After creating the Lucene index, next step is retrieving the top 1000 most frequent words among the tweets for each sentiment group. From the retrieved words, the time series of the term frequency is built according to the formula:

$$\text{frequency of term} = \frac{\#\ of\ term\ is\ used}{Time\ interval}$$

Time interval – granularity – was set for this case to 12 hours. After obtaining the time frequency, the whole time series is represented as a SAX string using alphabet size equal to 20. To be able to detect terms with a similar behavior over time, we developed the k-means algorithm to cluster terms together. Number of the clusters was set experimentally to 4. The algorithm recomputes the centroids of the clusters as translating between ASCII values of the character and computing the mean of the values.

0.3 Co-occurrence graph

Next step of this analysis is to build an undirected graph for each cluster. In this graph every node is representing different word in the cluster. Two nodes are connected by an edge if they appear in the same tweet. Weight of the edge is computed as the number of times they appear in the tweet together.

For this part of the analysis, mainly tools from Apache Lucene and stilo.g library were used for their performance and suitability for the task at hand. Weights of the edges are normalized (divided by the highest weight) and for the future processing only weights above the threshold are kept. The threshold was set in the main function for part0. Our value was experimentally set as 0.05 which keeps reasonable amount of data.

After graph preprocessing, the Largest connected component is extracted and saved into the result file. The same action was done while we were identifying innermost core using K-core algorithm. As a result, we found out that k-core results are significantly smaller than LCC.

All direct results from the analysis are stored in the resources/part0 section. Below you can find an example of the cluster #1 of positive supporters with k-core algorithm. Also, you can see that the words are usually cut, and a last letter is missing as a result of using the Italian tokenizer as a preprocessing tool. Words displayed below represent positive sentiment towards Matteo Renzi, who was the referendum proposer.

matteorenz graz

matteorenz provat

matteorenz averc

matteorenz arrivederc

matteorenz ciao

provat matteorenz

averc matteorenz

graz matteorenz

arrivederc matteorenz

0.4 Clustering of time series with a different granularity

For this part of the analysis we took tokens calculated in the previous step (0.3) for each cluster and both methods (k-core and LCC) but with a different granularity 3h. Newly created graph is saved. We are comparing the time series of terms for each cluster.

Below you can see examples of the results for different sentiment group and different cluster.

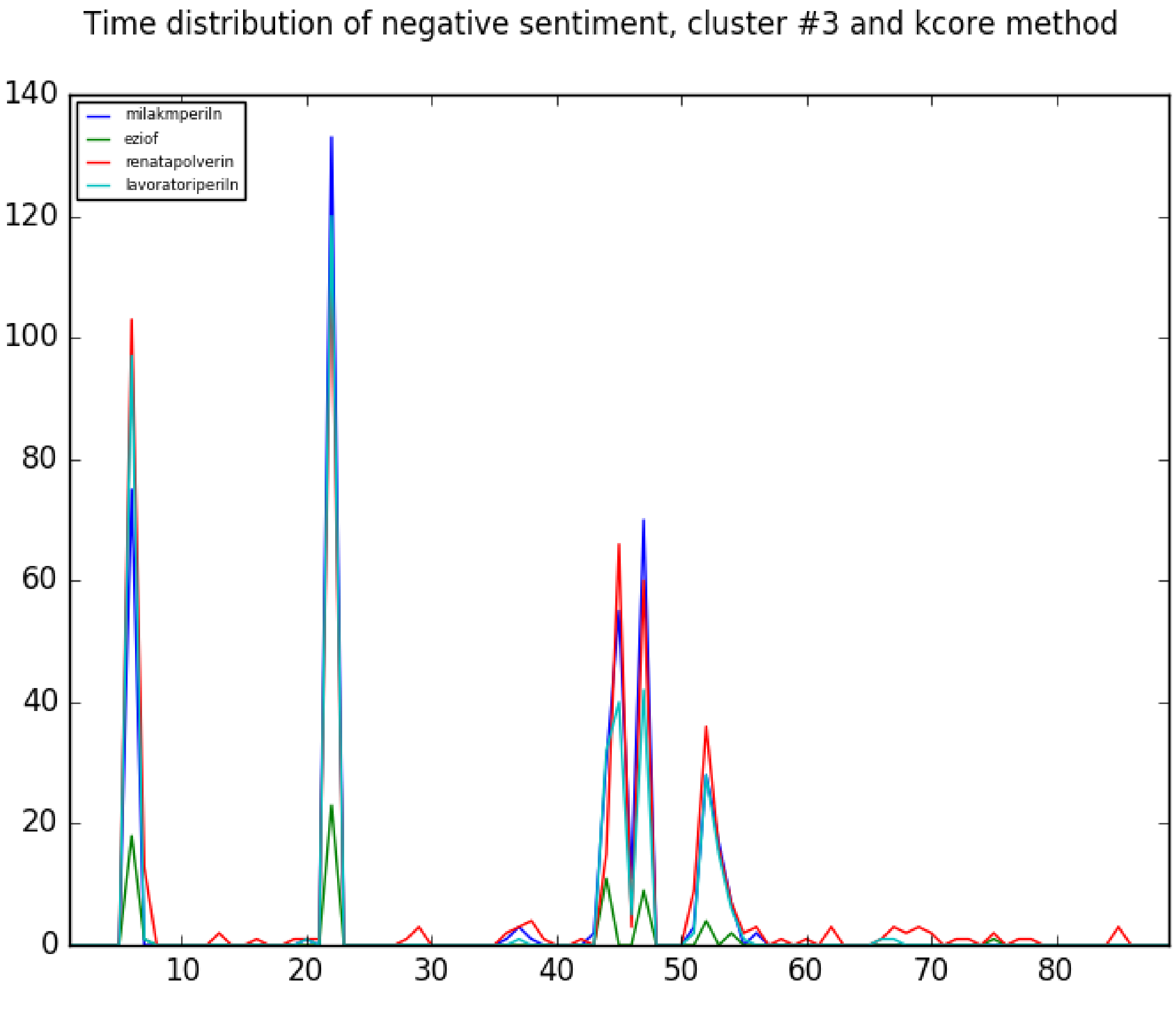

Figure 3 Word frequency of negative group cluster 3

In Figure 3, we can see similar temporal behavior of the terms belonging to the third cluster of negative sentiment group obtained by the k-core method.

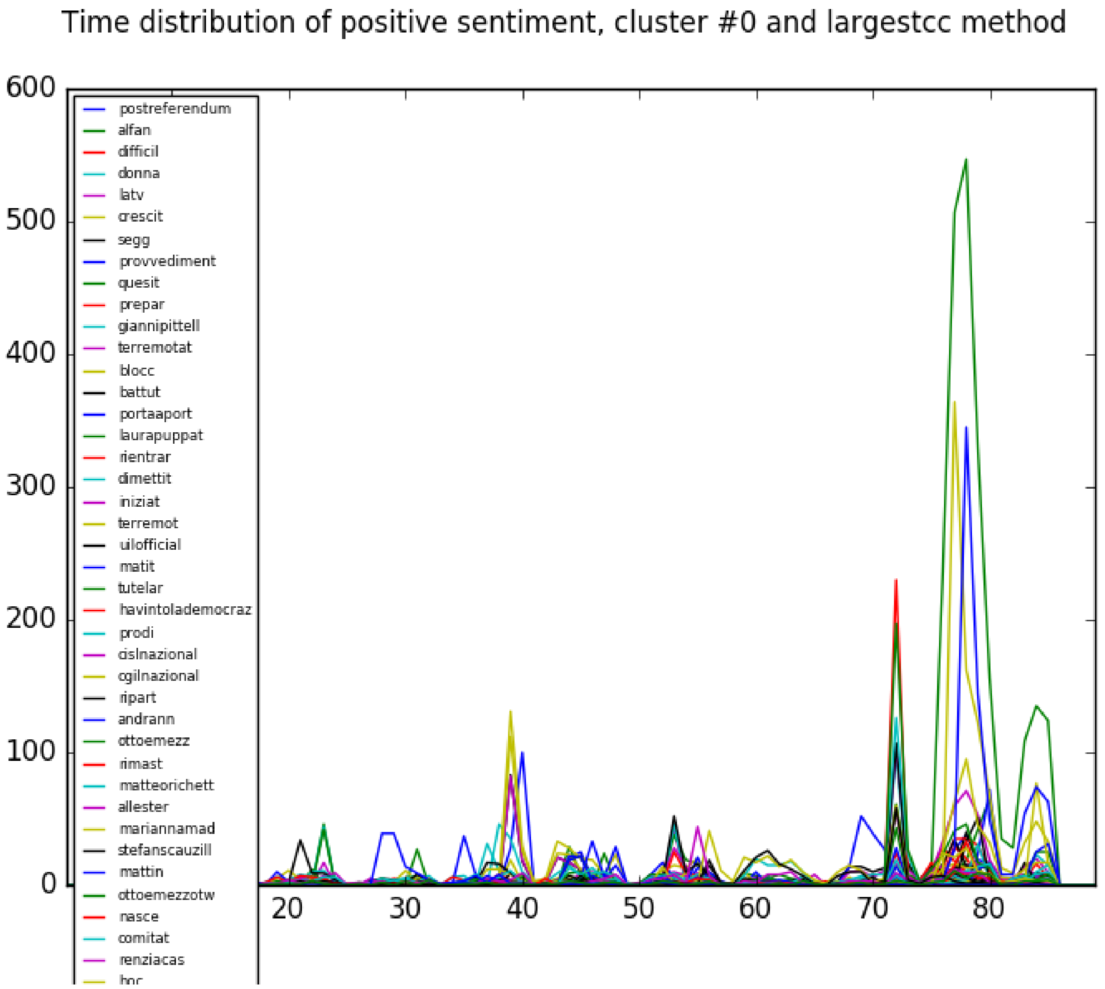

Figure 4 Word frequency of positive group cluster 0

In Figure 4 on the other hand we can observe highly correlated terms in favor of referendum. We can also see a large difference in the number of obtained terms with the largest connected component method.

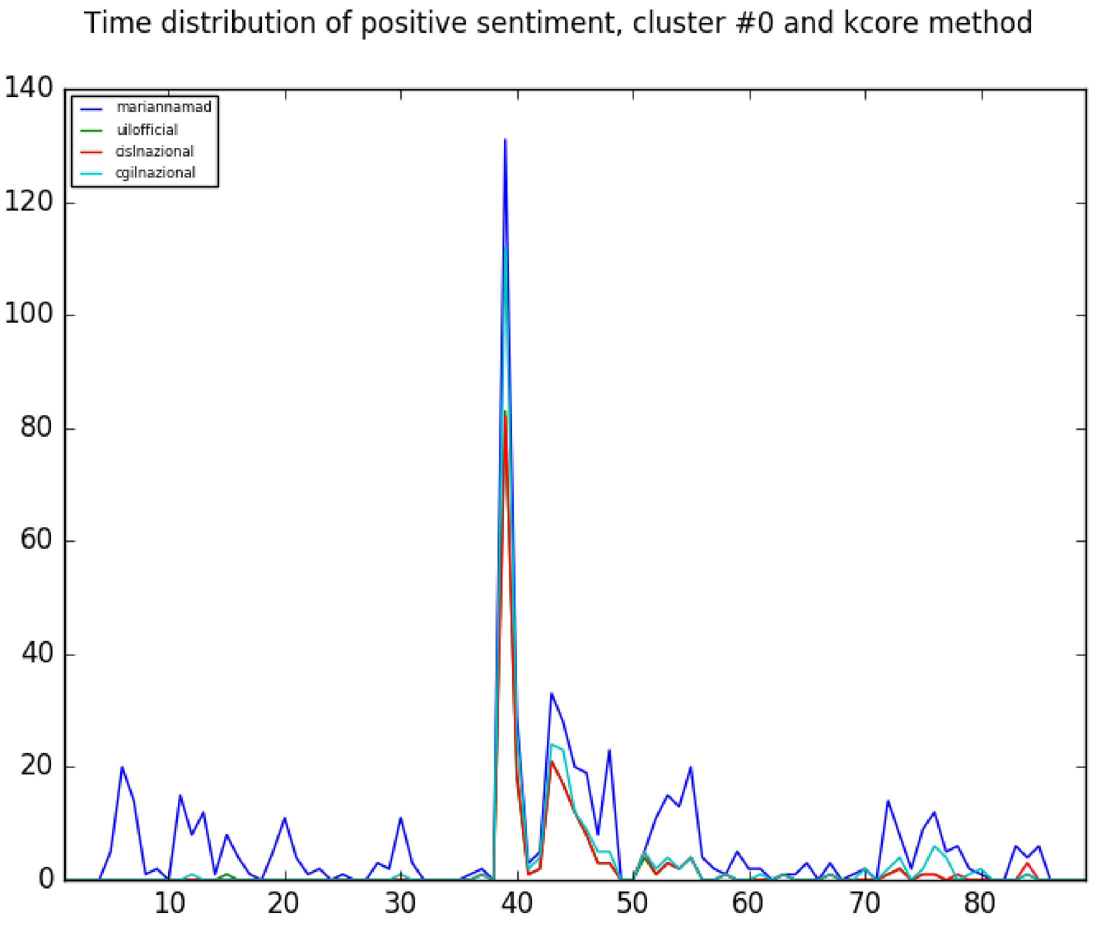

Figure 5 Word frequency of positive group cluster 0

All plots were saved in resources/part0/plots folder. In the above displayed plots, we can see a strong correlation not only between words with similar meaning like voting yes (no) but also between the politicians strongly representing one side.

Part 1 – Identify Supporters

1.1 Identify users that mention the politicians

Using the entire tweets dataset, the users mentioning at least one of the politicians of P retrieved in Part 0 were identified. Due to the size of the dataset, the dataset was split in different parts so the data could be read, and the information could be stored without incurring in memory error.

From each tweet in the provided dataset, it was identified if the user posting that tweet was mentioning any of the politicians in P; if they do, the ID of the user was stored in a text file, as well as a map showing the ID of the user and the IDs of the users that are mentioned in the tweet.

An additional step was performed in this part to distinguish between users supporting YES or NO politicians as follows, defining a variable that keeps count of the sentiment initialized with zero, for each YES politician that the user mentions +1 is summed up to the sentiment variable and for each NO politician that the user mentions -1 is summed up to the sentiment; if the final sentiment for a given user is greater than zero it is considered to support the YES in the referendum, if the sentiment is less than zero then the user is classified as a NO supporter, while if the sentiment is equal to zero, the user is considered as having a “neutral” sentiment or not having sentiment towards the referendum.

When the previous process is completed, the outputs from each division of the dataset are unified in order to get a single integrated output of the users’ IDs mentioning a politician, the map showing all the politicians that they mention, the sentiment of the user and the tweets mentioning a politician.

As result, 736.374 tweets published by 79.196 different users mentioning at least one of the politicians in P were identified, from which 12.286 are NO supporters, 16.782 are YES supporters and the remaining 50.128 users are considered as neutral users, this means that they do not support any YES or NO politician explicitly. In the Figure 6 can be seen the results in relative terms, showing the sentiment distribution among the users mentioning a politician in P.

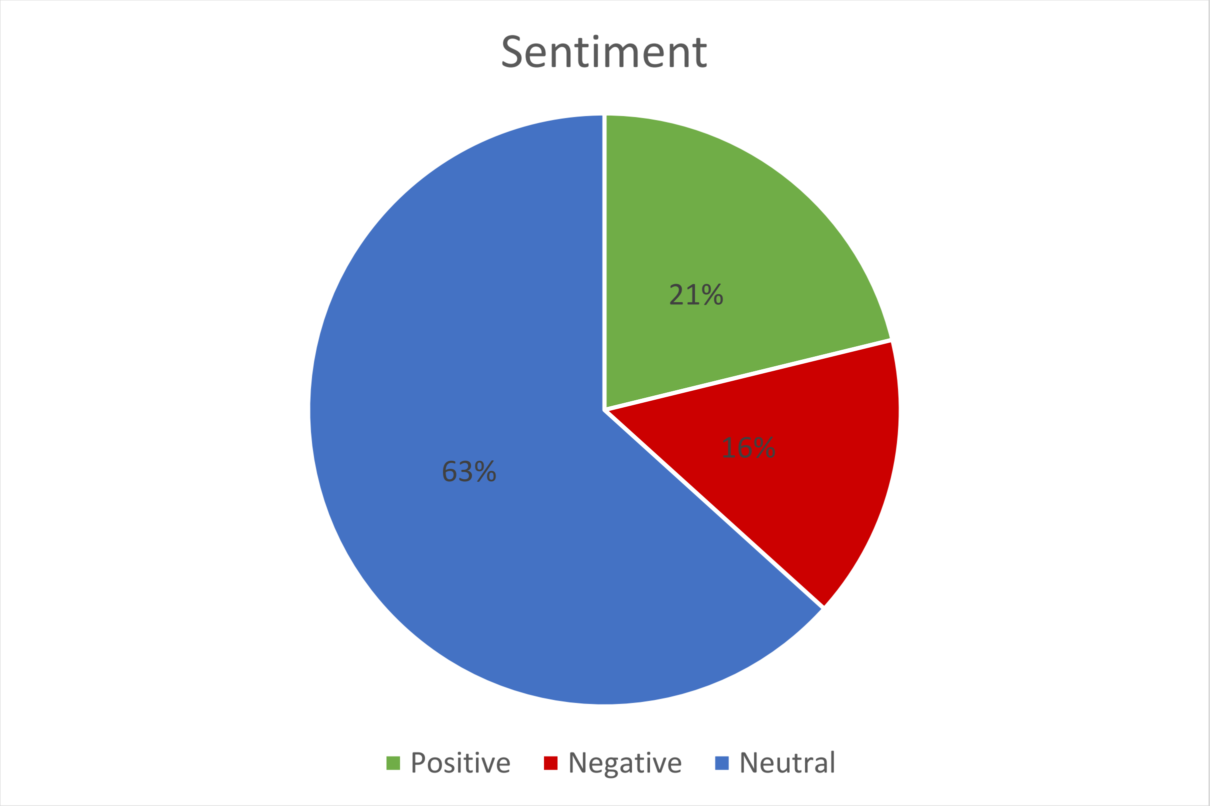

Figure 6 Sentiment distribution among twitter users who mention a politician

1.2 Find the largest connected component and compute HITS

In this part a subgraph was created with the set of users' IDs retrieved in Part 1.1 (excluding the users that were showing a neutral sentiment) using the provided graph "Official_SBN-ITA-2016-Net" by means of the it.sitlo.g.library. The list of IDs had to be cleaned up by removing the special characters such as text quotes appearing in them.

After creating the subgraph with the 29.068 users gathered previously, the Largest Connected Component was found. A connected component is defined as a subrgraph in which any two picked at random are connected and is not connected with any other external node in the supergraph. The resulting Largest Connected Component in this project has 18.059 nodes with 516.012 edges among them. This was done using the it.stilo.g.algo.ConnectedComponents. rootedConnectedComponents. Then the subgraph induced by the Largest Connected Component was created is saved in order to use it for further processes.

Then the HITS algorithm was computed using the subgraph obtained by the Largest Connected Component, a threshold of 0,00001 was set. This was done using the it.stilo.g.algo.HubnessAuthority.compute.

The HITS algorithm (Hypertext Induced Topic Search) calculates the Hub Ranking and the Authority Ranking. In graph theory, an authority is the node having many incoming edges and a hub is the one having many outgoing edges. A good authority points to many good hubs and a good hub points to many good authorities. In this specific case, an authority would be the one being mentioned a high amount of times, for instance, a good authority could be a politician while a hub would be the one mentioning other users a huge amount of times, an example of a good hub would be a journalist or users that are very active in Twitter.

1.3 Find the top 1.000 Hubness Authority users for positive and negative

The aim of this part was to find for both YES and NO, the Top 1.000 most central users using the Hub Score and the number of times they mentioned a politician in P. For this purpose, a combined metric is proposed as follows:

$$SCORE_s = \frac{nm_s + hs_s}{2},$$

Where:

- S contains two sets, the set of users supporting the NO and the set of users supporting YES.
- nm corresponds to the normalized amount of mentions
- hs is the hub score

The normalized amount of mentions nm is calculated as follows:

$$nm = \frac{x - x_{minimum}}{x_{maximum} - x_{minimum}}$$

After calculating the proposed metric, the top 1.000 users were saved for both NO supporters and YES supporters. In Table N°1 there is the list of the top 10 users found after computing the metric for the NO supporters and in Table N°2 can be seen the top 10 users that are YES supporters.

User ID	Username	Score	Total tweets	Days active since account creation	Tweets per day
2898075617	@MPenikas	0,311	595.200	1.765	337,22
499128183	@niemiz1964	0,239	Account suspended		
836271002	@marino29b	0,193	760.500	2.556	297,54
710498404531630080	@lidiapres08	0,145	47.600	1.279	37,22
2361360169	@SirioFerri	0,140	31.900	2.038	15,65
396756630	@gstelluti	0,137	220.700	2.892	76,31
147543162	@forza_italia	0,133	173.300	3.410	50,82
283636579	@MichelePlatoni	0,132	274.100	3.075	89,14
3357308483	@demian_yexil	0,126	598.100	1.552	385,37
4405643841	@lidiera1	0,125	70.500	1.308	53,90

Table N°1 – Top 10 No Supporters

User ID	Username	Score	Total tweets	Days active since account creation	Tweets per day
793787252040142848	@Pimpapallina	0,500	34.000	1.034	32,88
783023868378279940	@monica_bosso	0,447	18.800	1.065	17,65
4885620221	@FedericaBidiVal	0,445	36.000	1.308	27,52
1579008720	@GiusiPlebot	0,407	Account suspended		
2193007701	@mirecannizzaro	0,290	24.700	2.130	11,60
2812859778	@stefanodelisio	0,239	112.100	1.826	61,39
788794627042205696	@AntonellaTrich2	0,219	25.500	1.065	23,94
4851423724	@VittoriaCicu	0,200	132.800	1.339	99,18
722053268587855872	@Sandranino1977	0,189	112.200	1.248	89,90
722050091624833024	@AlessaM1986	0,188	Account deleted		

Table N°2 – Top 10 Yes Supporters

As expected, the top 10 YES and NO supporters are active in the Twitter platform posting from 11,60 tweets per day to 385,37 tweets per day which is much higher than the average tweets per day that a user posts (1.5 tweets per day) according to the most recent information about twitter found by the Worldometers' algorithm.

The Top 1000 users for both YES and NO were saved in a two different text files, one containing the IDs and scores in descending order and the other one just containing the IDs of the supporters.

With this information and the data obtained in the previous sections, there are more YES supporters than NO supporters and they appear to have higher scores in the proposed metrics, taking this into account the expected result of the referendum could be Yes, but the real result was No; this can be explained as the content of the tweets is not taken into account and also because the fact that someone mentions a politician in the tweet it does not mean that the person supports that politician, it could be a tweet expressing their discomfort with the mentioned politician.

1.4 Identify the top 500 k-Players using the KPP-NEG algorithm

The main idea of this part was to find the top 500 k-Players or "brokers" using the KPP-NEG algorithm; the concept behind the KPP-NEG algorithm is to find the set of nodes in a graph such that when removing those nodes, it would maximally disrupt the connectivity in the graph.

In order to perform this, the set of M users found in 1.1 was retrieved as well as the subgraph induced by the Largest Connected Component of Part 1.2, due to the large processing time required to run the KPP-NEG algorithm in the whole graph, a threshold of 100 was set in order to keep the nodes with more than 100 vertices (incoming + outcoming).

The top 500 k-Players were found in both NO and YES supporters sets. In Tables 3 and 4 can be seen the Top 5 k-Players for each set.

User ID	Username	Score
70951142	@Ale_Mussolini_	10572
3004185915	@DeLaMarcus92	10571
2249221297	@N0Euro	7929
2525039696	@concett95520392	7929
2767073855	@ange1914	7929

Table N°3 – Top 5 No Brokers

User ID	Username	Score
420656128	@Lele_Cocco	10572
15178183	@OfficinaTorino	10571
701046240788803584	@chiovaro9	7929
73720086	@mariandola	7929
485719283	@chiarasal93	7929

Table N°4 – Top 5 Yes Brokers

Between these users it can be found politicians such as Alessandra Mussolini (@Ale_Mussolini_) and journalists like Marianna Iandolo (@mariandola). The output files are four, two for the top 500 NO brokers, one with just the IDs and the other one with the IDs and the scores; and the same two files for the top 500 YES brokers.

Part 2 – Spread of Influence Analysis

For solving the problem introduced in this part, we have developed an LPA algorithm which also uses the stilo.g library.

The setup was done as follows:

We are using the whole network and labels of the nodes are decided based on the previously identified users from part 1. Positive group supporters had been set up with positive label, negative supporters with negative label and the rest start with unknown status. In order to be able to better distinguish the label, we assign label **1** to the positive group and label **2** to the negative group. All the unknown nodes get a label value 404. Based on the given instruction, only positive and negative labels can propagate, and LPA algorithm was developed in a standard way. Unknown labels remain the same for the purposes of this task.

Other initial assumption is that seeds identified as positive and negative can propagate and therefore they are not considered as solid. Since we are operating over the directed graph, neighbors are considered based on the incoming edge. Incoming edge to user U from user V means that the user V is following user U on twitter. This determines the way how the labels can be spread.

Terminating conditions for the algorithm run are two possible cases. If all the negative nodes change their label in one step to opposite ones or after 40 iterations (set as a threshold for the algorithm in our case). Th second condition was set in place due to the very small changes over the iterations which could cause long computing time without distinct changes in the results.

Label	Initial count	Final count
Yes	16785	428758
No	12290	172
Unknown	16797510	16387004

Table N°5 - Results of the LPA - M

Label	Initial count	Final count
Yes	1000	428743
No	1000	26
Unknown	16814013	16387165

Table N°6 - Results of the LPA - M'

Label	Initial count	Final count
Yes	500	283711
No	500	145060
Unknown	16815304	16387163

Table N°7 - Results of the LPA - K-Players

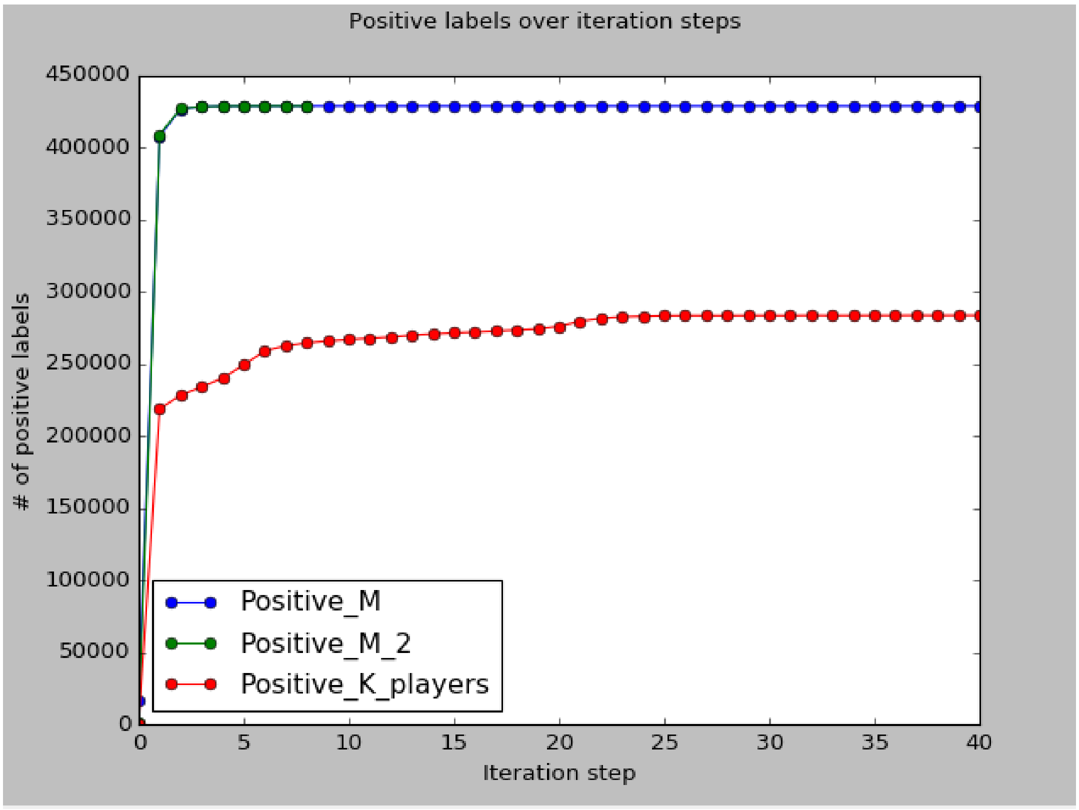

Figure 7 - Results of the LPA - Positive

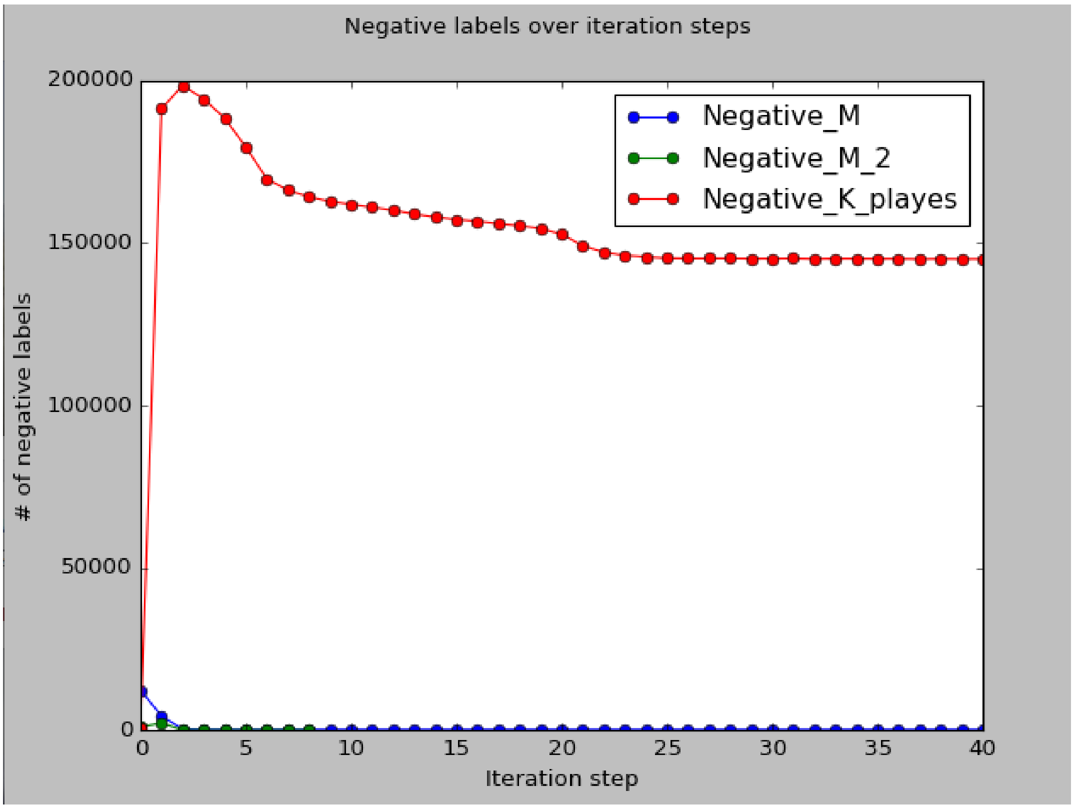

Figure 8 - Results of the LPA – Negative

The algorithm reached the convergence only with M2 dataset and it took only 8 epochs. For both other input data sources, LPA did not reach the convergence after 40 steps. In all cases positive labels propagate much more and faster than the negative ones. In basic M dataset it can be caused by unbalanced data, but as we can see with M2, which is a balanced dataset in terms of numbers, the propagation is similar.