Toulouse School of Economics

M2 - Web Mining

Pierre-Luc Alibert, Bastien Cannard, Kyllian James and Papa Oumar Mbodj

31 March 2023

Contents

1 Introduction

The aims of this project are twofold. First, we have to describe and analyse a social web graph. Then, we will use the social web graph structure, content and usage to make prediction.

The data are Twitter's posts posted during pas crisis events like floods or fires for example. The source of the data is TREC Incident Stream. Those data contains 5 high-level data to make emergency based on data processing:

- Events ⇒ Occured crisis events.
- Event Type/Topic ⇒ Event category.
- User ⇒ Active twitterer.
- Tweet ⇒ Posts by users.
- Tweet Priority ⇒ 4 levels of priority (Low, Medium, High and Critical) to indicate the level of emergency.

2 Part 1 - Describe the Social Network and Mine Actionable Information

2.1 Question 1

We are going to analyze the social graph built upon the user nodes and all the social interactions as edges.

2.1.1 Graph Global Statistics

This is a directed and unweighted graph. There are 53 038 nodes and 23 170 edges. We are going to compute the graph density, transitivity, girth and cohesion.

The graph density is a measure of connectivity defined as the number of edges divided by the number of pairs of vertices. The transitivity of a network tells us how likely it is for two nodes that are connected to a common neighbor to be connected to each other as well. It measures relative frequency of triangles in a graph. The girth is the number of vertices in the shortest cycle. The cohesion is the minimum vertices to remove to disconnect the graph.

Table 1: Twitter Graph Global Statitics

Statistic	Twitter Graph
Density	0
Transitivity	0.173
Girth	1
Cohesion	52970

The graph density is 0, which is very low. This suggests that the graph is not densely connected, and there are relatively fewer edges compared to the maximum possible number of edges. The transitivity of a graph is a measure of how "clustered" the nodes are in the graph. The transitivity of Twitter graph is 0.173, only 17.3% of the possible triangles in the graph actually exist. This suggests that the graph is relatively sparse and there are relatively few connected triples of vertices. This suggests that there are some groups of nodes that are more connected to each other than to the rest of the graph, but that there are also many nodes that are not part of any such group. The shortest cycle in the graph has a length of 1 edges. The cohesion is 52 970, so we need to remove at least 52 970 vertices from the graph to disconnect it. It's a lot compared to the total number of vertices.

2.1.2 Shortest-Path Distribution

The shortest path is the minimum number of edges which are necessary to go from one node to another one. The shortest-path distribution tells us how many pairs of nodes have a certain shortest-path length between them.

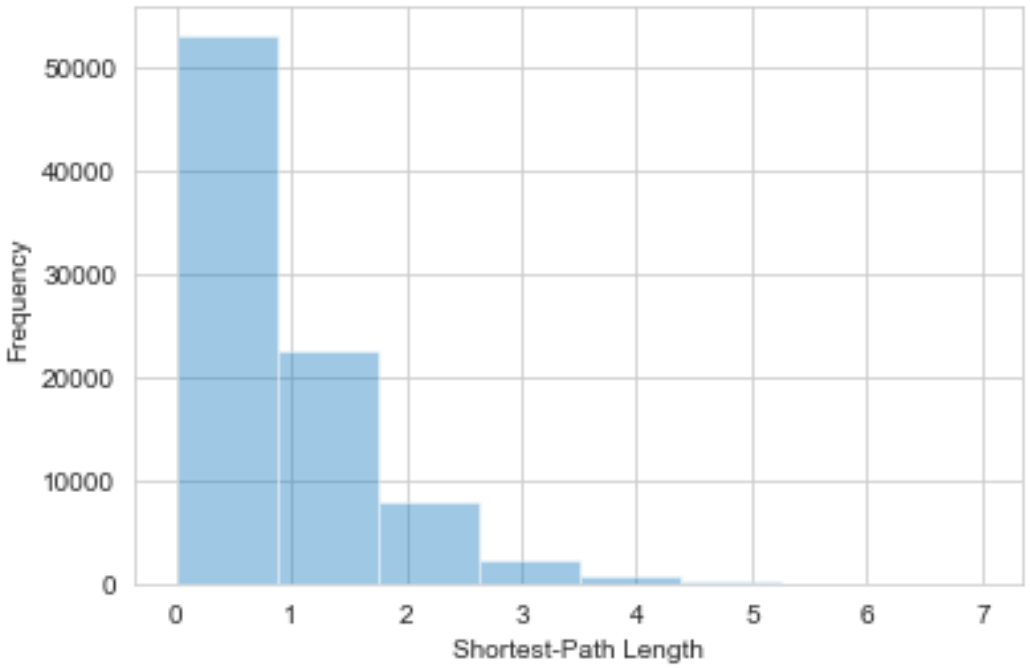

Figure 1: Shortest-Path Distances Distribution

The distribution is skewed to the right, it suggests that the graph has not a homogeneous structure, with different connectivity patterns across all pairs of users. The most frequent shortest-path length between a pair of nodes is 0, it means that there are a few nodes that are highly connected and have short paths to many other nodes, while most nodes have longer paths to reach other nodes in the graph. This could suggest that the Twitter graph is relatively well-connected, with many paths between pairs of nodes that are relatively short. It may also suggest that there are certain users that act as hubs or central points in the graph, connecting many other users to each other, while most nodes have fewer connections.

A highly connected hub node in a network can be critical for its overall robustness and resilience. If such a node is removed or disrupted, it can lead to a breakdown of the network or a loss of connectivity. A skewed shortest path distribution can also indicate a certain level of inequality in a network. Highly connected nodes or hubs may have more control or influence over the flow of information or resources in the network.

However, without further analysis, it is difficult to say for sure what this distribution means for the structure and dynamics of the Twitter graph. It would be useful to compare this distribution to other graph metrics, such as centrality measures, in order to gain a more complete understanding of the relationships between different nodes in the graph.

2.1.3 Centrality Indices

We will now focus on the relative importance of the vertices within the network thanks to centrality metrics. There are several centrality indices that we can compute including degree centrality, in-degree and out-degree centrality, eigenvector centrality, closeness centrality, Katz centrality for example.

The degree centrality shows how important is a node, but this measure not fully asses to nodes importance. Eigenvector defines centrality as an average of neighbors centrality. The closeness centrality is based on shortest path distance. Central vertices have more direct influence on the graph or easy access to other nodes.

First, we compute each centrality and store them in a table. We display the 10 first nodes (users) with their centrality measures.

Table 2: First 10 Nodes/Users - Centrality Indices

Node	Degree	In-Degree	Out-Degree	Eigenvector	Closeness	Katz	Page Rank	Hubs	Authorities
56568	0	0.0	0	1.072058e-21	0	0.0038	0.000012	0	0
56569	0.000019	0	0.000019	1.072058e-21	0	0.0038	0.000012	8.072053e-22	-4.397380e-20
56570	0	0	0	1.072058e-21	0	0.0038	0.000012	0	8.409288e-20
56571	0	0	0	1.072058e-21	0	0.0038	0.000012	0	6.601909e-20
56572	0	0	0	1.072058e-21	0	0.0038	0.000012	0	-8.803174e-21
56573	0	0	0	1.072058e-21	0	0.0038	0.000012	0	4.021454e-20
56574	0.000019	0	0.000019	1.072058e-21	0	0.0038	0.000012	1.562182e-04	9.410185e-20
56575	0.000019	0	0.000019	1.072058e-21	0	0.0038	0.000012	1.845692e-05	-7.641447e-20
56576	0	0	0	1.072058e-21	0	0.0038	0.000012	0	-3.403114e-20
56577	0	0	0	1.072058e-21	0	0.0038	0.000012	0	1.940441e-19

In this table, we can notice that in some centrality indices the value is 0, so those nodes are not important nodes/users in the graph from the point of view of those centrality indices. We also remark that some nodes have the same importance for some centrality measures. We can plot the distribution of each centrality to have a global vision.

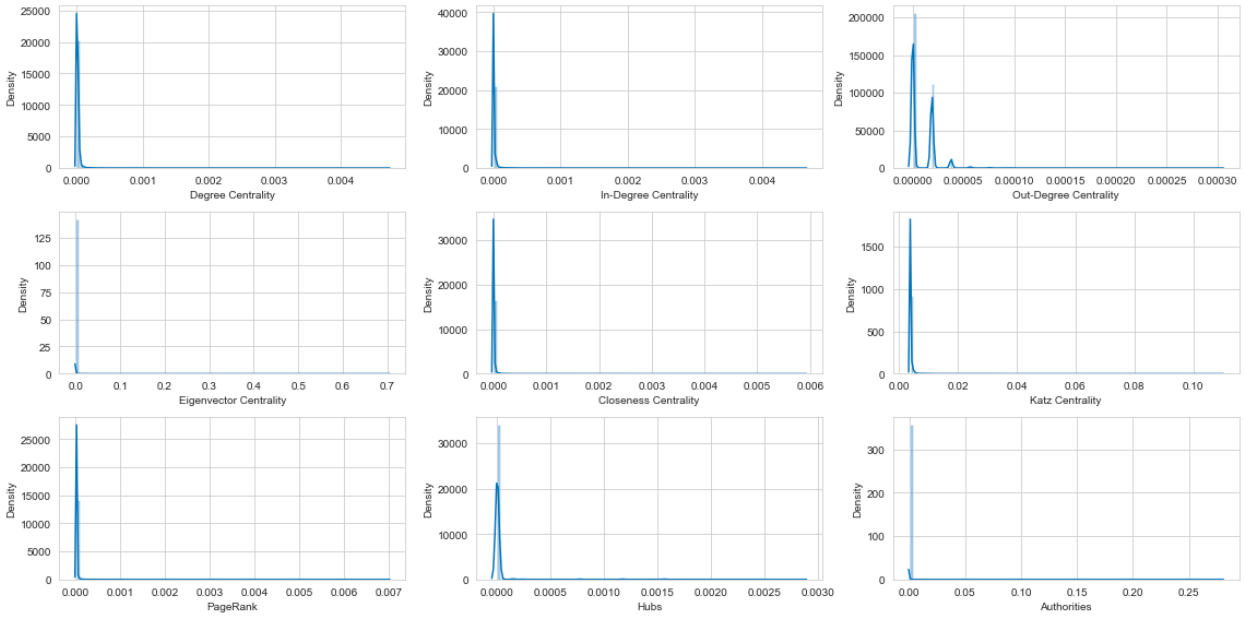

Figure 2: Distribution of each Centrality Indice

The distribution of each centrality metric is right skewed, it means that only a few users are important in the graph from the point of view of those centrality measures. So, only a few users are particularly important and play a disproportionately important role in the network. Some users act like hubs, there is a hierarchical structure in the graph, with a few users at the top of the hierarchy that are more central than others.

We are going to compare previous centrality indices by computing pairwise correlations.

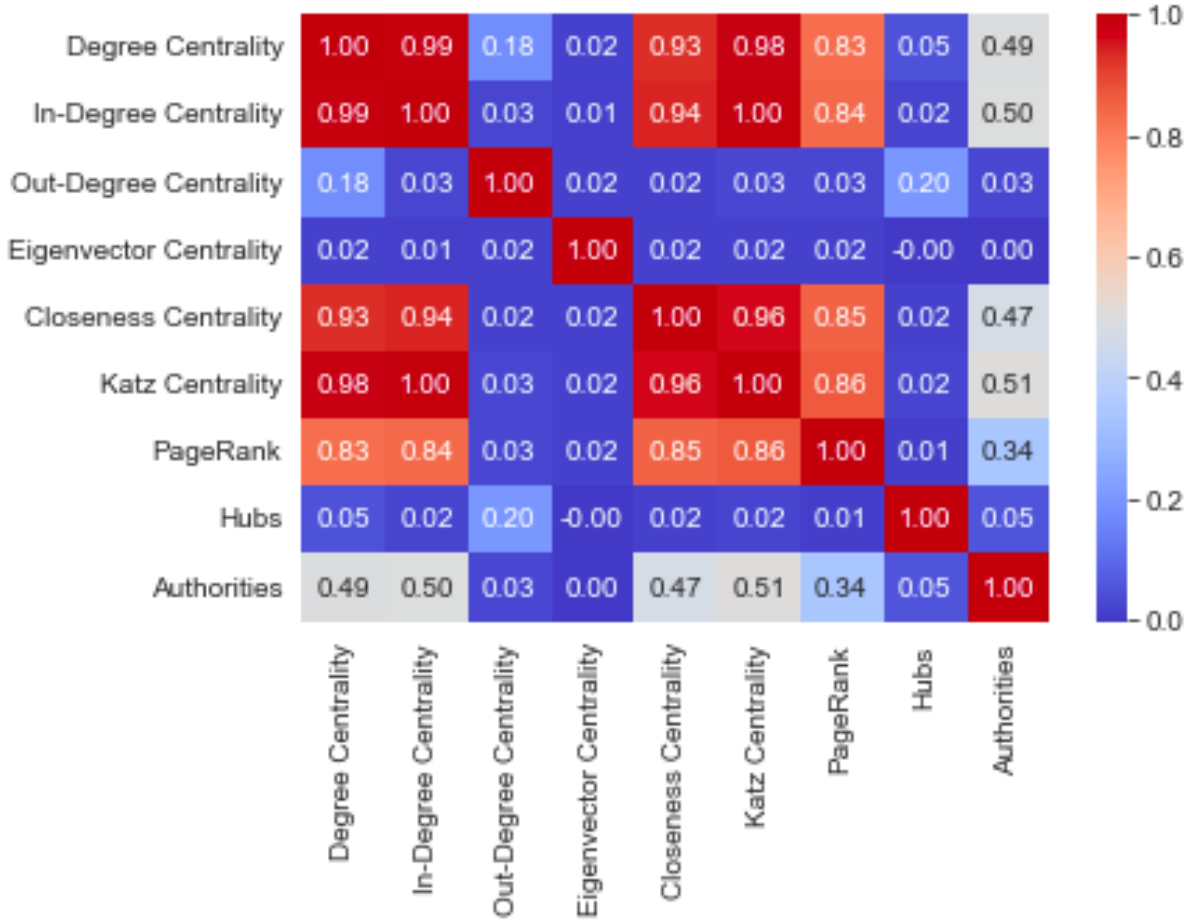

Figure 3: Pairwise Correlations between Centrality Indices

We can see that some centrality metrics are highly correlated like degree, in-degree, closeness, Katz and Page Rank ones. While others are less or not correlated with others.

Now, we have in the two next tables, the top 10 most important users for 2 different measures, which are the out-degree centrality and the Katz one.

Table 3: Top 10 users by Out-Degree Centrality

User	Out-Degree Centrality
29722	0.000302
31999	0.000245
2820	0.000170
29147	0.000151
50615	0.000151
10789	0.000132
28722	0.000132
28951	0.000132
28974	0.000132
50510	0.000132

Table 4: Top 10 users by Katz Centrality

User	Katz Centrality
27280	0.109723
7895	0.107417
31678	0.094892
1933	0.087681
40156	0.081891
7897	0.079173
1749	0.069959
22532	0.068173
2029	0.066029
39558	0.065398

We can notice that most important users from the point of view of a centrality metric is not the same from the point of view of another one. So, we would like to find if there are common important users from the point of view of multiple centrality metrics. That's what we get on the table just below.

Table 5: Common Users - All Centrality Indices

User
22532
39558
1933
2029
27280
1749
5110
7895
7897
40156
31678
31999

Those users are key vertices in the network as they are among the top users for multiple centrality measures.

2.2 Question 2

Since it's a directed graph, there are degree-in and degree-out. So, we have to get the degree-in and degree-out distribution. The degree distribution of a directed network tells us how many nodes have a certain number of connections incoming and outcoming.

Table 6: Descriptive Statistics of Degree-in and Degree-out

Statistic	Degree-in	Degree-out
Minimum	0	0
1st Quantile	0	0
Median	0	0
Mean	0.437	0.437
3rd Quantile	0	1
Maximum	246	16

It appears that the majority of nodes in the graph have a degree-in and degree-out of either 0 or 1. The minimum degree-in and degree-out of 0 suggest that there are nodes in the graph which are not connected to others. On average a user is connected to 0.437 other users. The degree-in third quantile of 0 indicates that at least 75% of users doesn't have a connection incoming from other users. While the degree-out third quantile of 1 indicates that at least 25% of users have one connection outcoming or more to other users. The degree-in maximum is 246, this is the most social interactions incoming to a user. While the most

social interactions outcoming from a user is 16.

Overall, the degree statistics suggest that the graph isn't well-connected, with a majority of nodes having a null degree of connectivity. There are many isolated nodes/users in this graph which are completely disconnected from the rest of the network. However, the right skewed distribution of degree-in and degree-out suggests that there is highly connected hubs.

Now, we can have a look at the histograms of degree-in and degree-out.

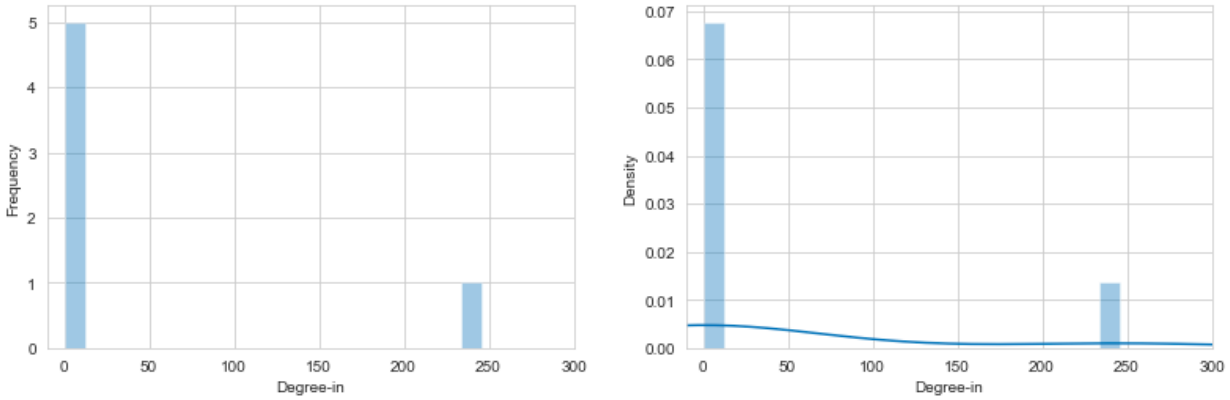

Figure 4: Degree-in Distribution

The histogram confirms what we said just before, the graph is not well-connected. The most frequent degree-in in this Twitter graph is around 0, it means that there are many nodes (users) that have 0 connections (social interaction) with other nodes. This could indicate that people are on Twitter only to get informed and not to interact with others. Moreover, the distribution is right skewed, so there might be the presence of hubs.

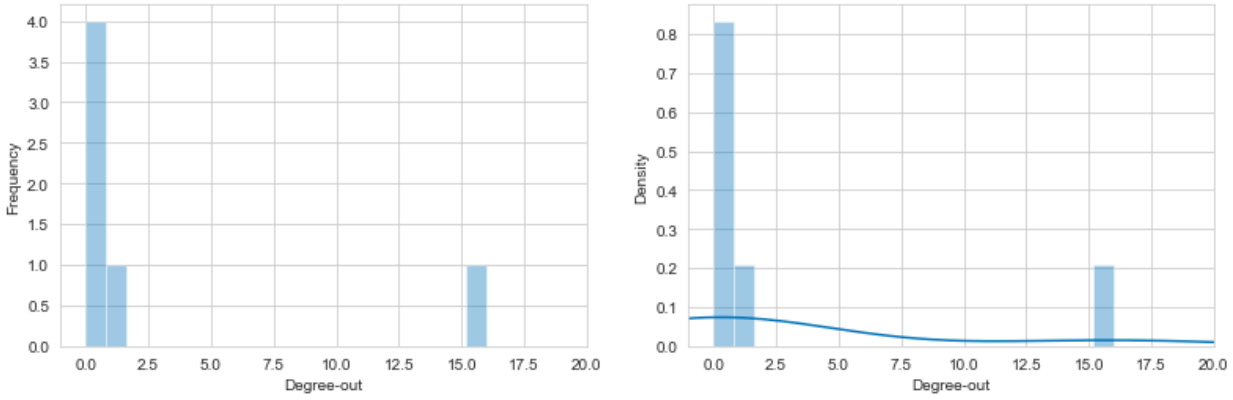

Figure 5: Degree-out Distribution

We can repeat the same comment here for degree-out distribution.

We can also visualize the degree-in and degree-out distributions on the graph, coloring the nodes and letting only the name of vertices with a degree distribution greater than k,

with $k = 200$ for degree-in and $k = 10$ for degree-out. Unfortunately, there are too many nodes in this graph, so we can't draw it.

We wanted to check if the degree-in and degree-out distributions are power-law distributions. But, we an error occurs which is `int too large to convert to float`.

2.3 Question 3

In this question, we are going to identify the 3, 5 and 10 most important users for each event. The notion of important users is based on the Katz centrality here. In the 3 tables below, you will find the top 3, 5 and 10 users for each event. We can remark that the event 22 has no top users. This could indicate that the event is less popular or have a smaller pool of users compared to other events. We also notice that each time the top users of an event are not the same of another event. It suggests that there may be a diverse group of users participating in these events, and that different users may have varying levels of expertise or interest in different topics. This could be a positive sign for the overall engagement and diversity of the community.

Table 7: Top 3 Users for each Event

Event	Top 3
0	[56568, 56569, 56570]
1	[56574, 56717, 57277]
2	[56571, 56586, 56724]
3	[56778, 56869, 58258]
4	[57662, 58604, 58868]
5	[57517, 57594, 59524]
6	[57896, 58174, 58375]
7	[56574, 56677, 57132]
8	[56667, 57377, 57537]
9	[57426, 57678, 57689]
10	[56677, 56742, 56788]
11	[57662, 57784, 58004]
12	[58338, 58533, 58578]
13	[57148, 57395, 57495]
14	[57395, 57469, 57594]
15	[56677, 58175, 64377]
16	[56812, 57537, 57594]
17	[56588, 56742, 56812]
18	[56742, 56911, 57060]
19	[57236, 57594, 58175]
20	[56574, 57129, 57195]
21	[60135, 62719, 64516]
22	
23	[56578, 60102, 61275]
24	[56610, 57278, 57667]
25	[58097, 59612, 75566]
26	[56899, 57692, 58909]
27	[58373, 59388, 60898]
28	[56636, 57426, 58373]
29	[56702, 56742, 57537]
30	[57377, 57441, 57469]
31	[56586, 56686, 57370]
32	[57537, 58506, 59124]
33	[59447, 59580, 64308]

Table 8: Top 5 Users for each Event

Event	Top 5
0	[56568, 56569, 56570, 56571, 56572]
1	[56574, 56717, 57277, 57278, 57279]
2	[56571, 56586, 56724, 56731, 56743]
3	[56778, 56869, 58258, 58259, 58260]
4	[57662, 58604, 58868, 58869, 58870]
5	[57517, 57594, 59524, 59525, 59526]
6	[57896, 58174, 58375, 59124, 59157]
7	[56574, 56677, 57132, 57471, 57665]
8	[56667, 57377, 57537, 58338, 58480]
9	[57426, 57678, 57689, 58058, 59124]
10	[56677, 56742, 56788, 57035, 57908]
11	[57662, 57784, 58004, 59940, 59995]
12	[58338, 58533, 58578, 58597, 58640]
13	[57148, 57395, 57495, 57908, 58058]
14	[57395, 57469, 57594, 57662, 57967]
15	[56677, 58175, 64377, 64378, 64379]
16	[56812, 57537, 57594, 57662, 59420]
17	[56588, 56742, 56812, 57065, 57145]
18	[56742, 56911, 57060, 57372, 57526]
19	[57236, 57594, 58175, 59358, 64494]
20	[56574, 57129, 57195, 57445, 57506]
21	[60135, 62719, 64516, 66982, 76709]
22	[]
23	[56578, 60102, 61275, 62755, 63547]
24	[56610, 57278, 57667, 57762, 58075]
25	[58097, 59612, 75566, 80895, 80896]
26	[56899, 57692, 58909, 58916, 59532]
27	[58373, 59388, 60898, 60917, 60937]
28	[56636, 57426, 58373, 59388, 60416]
29	[56702, 56742, 57537, 57594, 57899]
30	[57377, 57441, 57469, 57700, 57899]
31	[56586, 56686, 57370, 57775, 57818]
32	[57537, 58506, 59124, 59358, 59481]
33	[59447, 59580, 64308, 64463, 64534]

Table 9: Top 10 Users for each Event

Event	Top 10
0	[56568, 56569, 56570, 56571, 56572, 56573, 56574, 56575, 56576, 56577]
1	[56574, 56717, 57277, 57278, 57279, 57280, 57284, 57285, 57286, 57287]
2	[56571, 56586, 56724, 56731, 56743, 56767, 56791, 56965, 57395, 57554]
3	[56778, 56869, 58258, 58259, 58260, 58261, 58262, 58263, 58264, 58265]
4	[57662, 58604, 58868, 58869, 58870, 58871, 58872, 58873, 58874, 58875]
5	[57517, 57594, 59524, 59525, 59526, 59527, 59528, 59529, 59530, 59531]
6	[57896, 58174, 58375, 59124, 59157, 59983, 60148, 60149, 60150, 60152]
7	56574, 56677, 57132, 57471, 57665, 58037, 58479, 60320, 60321, 60322]
8	[56667, 57377, 57537, 58338, 58480, 58595, 58815, 58853, 60429, 60430]
9	[57426, 57678, 57689, 58058, 59124, 60107, 60374, 60376, 60887, 60888]
10	[56677, 56742, 56788, 57035, 57908, 58121, 58899, 59155, 59386, 59388]
11	[57662, 57784, 58004, 59940, 59995, 61832, 62223, 62224, 62225, 62226]
12	[58338, 58533, 58578, 58597, 58640, 58712, 58780, 58848, 59360, 60284]
13	[57148, 57395, 57495, 57908, 58058, 58338, 59218, 59444, 60078, 61592]
14	[57395, 57469, 57594, 57662, 57967, 58174, 58259, 58338, 58597, 58712]
15	[56677, 58175, 64377, 64378, 64379, 64380, 64381, 64382, 64383, 64384]
16	[56812, 57537, 57594, 57662, 59420, 59438, 59485, 59614, 60102, 60420]
17	[56588, 56742, 56812, 57065, 57145, 57146, 57370, 57506, 57565, 57594]
18	[56742, 56911, 57060, 57372, 57526, 57688, 57700, 57851, 57899, 58174]
19	[57236, 57594, 58175, 59358, 64494, 64669, 64901, 64969, 67399, 67553]
20	[56574, 57129, 57195, 57445, 57506, 57594, 57822, 57894, 58021, 58297]
21	[60135, 62719, 64516, 66982, 76709, 78072, 79344, 79345, 79346, 79347]
22	
23	[56578, 60102, 61275, 62755, 63547, 63868, 64494, 64657, 64693, 65170]
24	[56610, 57278, 57667, 57762, 58075, 58199, 58259, 58298, 58370, 58622]
25	[58097, 59612, 75566, 80895, 80896, 80897, 80898, 80899, 80900, 80901]
26	[56899, 57692, 58909, 58916, 59532, 59547, 59887, 59890, 61463, 62682]
27	[58373, 59388, 60898, 60917, 60937, 60974, 60975, 60979, 61029, 61034]
28	[56636, 57426, 58373, 59388, 60416, 60926, 60937, 60967, 60987, 60989]
29	[56702, 56742, 57537, 57594, 57899, 58251, 58252, 58394, 58597, 58698]
30	[57377, 57441, 57469, 57700, 57899, 58310, 58319, 58338, 58376, 58484]
31	[56586, 56686, 57370, 57775, 57818, 58172, 58338, 58969, 59481, 59565]
32	[57537, 58506, 59124, 59358, 59481, 60682, 61189, 62105, 62177, 62755]
33	[59447, 59580, 64308, 64463, 64534, 65156, 66942, 70778, 70851, 73098]

2.4 Question 4

The aim of this section is to get the communities from our graphs given the events.

The first step will be to get these communities in the social graph which depends on

user nodes and their relationship. In the second part of this section we will also try to get communities of users but this time based on their tweet content. This community detection task will require many methods that we will present and detail in next sections.

2.4.1 Question a

Here we used two main methods to do our analysis first we apply a percolation community detection method and on the other hand we try to get communities by maximizing modularity. In the context of community detection, percolation involves gradually removing nodes or edges of a network until the network breaks down into disconnected components. The idea is that nodes or edges that play a critical role in maintaining the connectivity of the network will be removed last, and the resulting clusters will correspond to communities. In our context, we initialize with k (number of minimum users to have a cluster) equals to 3 and we end up with a lot more clusters than needed. So we move to a modularity maximizer based method.

We end up using the modularity maximizer method, but this brings more complexity and with a lot more clusters which is not quite optimal. So let's try now to get these clusters but based on users' tweet content.

2.4.2 Question b

For this question, we need to apply clustering methods like k-means on tweet vectors. The first step would be to create our dataframe of tweets content and user id. To do so, we will need to merge posted and tweet dataset. But the core of this question is in the tokenization of the text and its TF-IDF vectorization.

	Tweet_ID	text	User_ID	text_token
0	582	#colorado. Told you its #amazing http://t.co/6...	56568	[colorado, Told, amazing, httptco6z0Qq7qe]
1	583	RT @northfortynews: Tanker helicopter heads up...	56569	[RT, northfortynews, Tanker, helicopter, heads...
2	584	#Evacuation center Cache La Poudre Middle Scho...	56570	[Evacuation, center, Cache, La, Poudre, Middle...
3	585	20F degrees cooler tomorrow in North Central &...	56571	[20F, degrees, cooler, tomorrow, North, Centra...
4	586	FEMA has authorized the use of federal funds t...	56572	[FEMA, authorized, use, federal, funds, help, ...

Figure 6: Tweet-User Dataset

Now we apply `tfidfvectorizer` to our column text, we can perform our k-means clustering method and get our 32 clusters and their number of users.

Table 10: Clusters' Number of Users

Cluster	Number of Users
0	14746
1	614
2	684
3	623
4	873
5	162
6	1236
7	1718
8	1959
9	1648
10	678
11	581
12	3626
13	940
14	2200
15	574
16	629
17	1457
18	817
19	711
20	968
21	624
22	1610
23	1789
24	561
25	3968
26	1685
27	1018
28	3120
29	1836
30	1205
31	304
32	822

3 Part 2 - Predict the Criticality of a Tweet Post

This part is dedicated to the answer of the following question: given a tweet post, can we predict its level of criticality among the "Low", "Medium", "High" and "Critical" levels?

To carry out this multi-label classification problem, we are going to combine graph struc-

ture analysis and text mining techniques, considering the following methodology:

- Split data into training and testing subsets;
- Definition and implementation of tweet features;
- Learning predictive models;
- Evaluation of the models;
- Model improvement.

Before those steps, some preparation is needed.

3.1 Data preparation

This preliminary step consists in getting a wide ready-to-use table from the original database.

According to the *Graph_DB_description.pdf* document, we perform merges between the following tables: $Tweets$, $User$, $Event$, $Hashtag$; the main table being $Tweets$, because this is the one that contains the column we want to predict, $annotation\_postPriority$.

Finally, the table we are using for that classification work contains 55.414 rows, each one identified as a tweet, and 26 columns, including $annotation\_postPriority$.

3.2 Question 1

The first problem before splitting the dataset is to have data that are of good quality, which means in other words that we need to know if some rows of the table have to be deleted.

Actually, there is not any missing value in the table, which is a very good surprise. On the contrary, when looking at the modalities of the column we want to predict, we can see that some rows will not be useful for the prediction task.

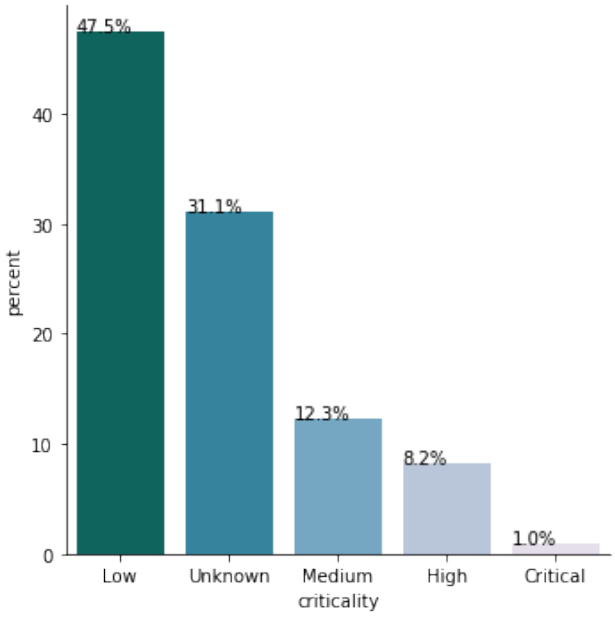

Figure 7: Distribution of the modalities of the *annotation_postPriority* column

Indeed, almost one-third of the rows will have to be deleted because their label is "Unknown", which is not a modality that we want to predict. Thus, our dataset finally has 38.175 rows.

We can now split our dataset between a train set and a test set. We keep 70% of the data for training our models.

3.3 Question 2

In order to have better results in our predictions, we add some columns to our table. These columns can be grouped into two types: columns based on stylometric characteristics, and columns based on content.

3.3.1 Stylometric Characteristics

Such characteristics describe the presence of links in the post, of certain punctuation marks, of emoticons, ...

We choose to add four such columns:

- Is_RT is a Boolean column that indicates whether the tweet contains "RT @", that is to say whether the tweet is a retweet or not, the idea being that a tweet that is a retweet is less important than at least one tweet (the one that is retweeted);
- $Nb\_!$ is a column that indicates the number of exclamation marks in the tweet, the idea being that the more the exclamation marks in the tweet, the higher the criticality;

- *Nb_UpCaseWords* is a column that indicates the number of words that are fully written in uppercase letters (we except the words "I" and "RT"), the idea being that the more the uppercase-written words in the tweet, the higher the criticality;
- *has_link* is a column that indicates the number of links there are in a tweet, the idea being that a tweet that includes links is a tweet that inputs information on the social network from somewhere else on the internet and that is then possibly critical.

3.3.2 Content-Based Features

Such characteristics describe whether the post contains highly frequent words, the popularity of the tweet hashtag, etc.

We choose to add four such columns as well:

- *unique_words* is a column that indicates the number of unique words in a tweet;
- *nb_words* is a column that indicates the nuber of words used in a tweet;
- *ratio_words* is a column that indicates the ratio of unique words used in a tweet, relative to the number of words used in that tweet;
- *len_text* is a column that indicates the length of the tweet after having cleaned it (tokenization, removing of stopwords, lemmatization).

3.4 Question 3

We can now use a supervised algorithm to learn how to predict the value of the target variable, the priority of the tweet. We are going to use the *scikit-learn* library to perform that task. More specifically, we are going to use four algorithms from that library to classify the tweets:

- a Logistic regression model;
- a Gaussian Naive Bayes classifier;
- a Random Forest classifier;
- a Decision Tree classifier.

The dataset we are going to let those algorithms learn on is eventually composed of 19 columns.

3.5 Question 4

The metric we use to evaluate the performance of these four models is the F1 metric. Because we have more than two modalities in the variable we want to predict (Low, Medium, High, Critical and Unknown), the final F1 score of a model will be the average of the F1 score for each class.

The following table gives the results for each classifier.

Table 11: Evaluation of the four classifiers

Model	F1 score
Logisitic Regression	0.680
Gaussian Naive Bayes	0.209
Random Forest	0.868
Decision Tree	0.799

Having a look at the distribution of the different classes in the dataset we use for that prediction task, it seems that the Gaussian Naive Bayes model has a very low F1 score. Indeed, a poor classifier that would predict each tweet as low-critical would have better results. Such a classifier would also have about the same results as the Logisitic Regression. On the contrary, we can say that the Random Forest and Decision Tree models have good results.

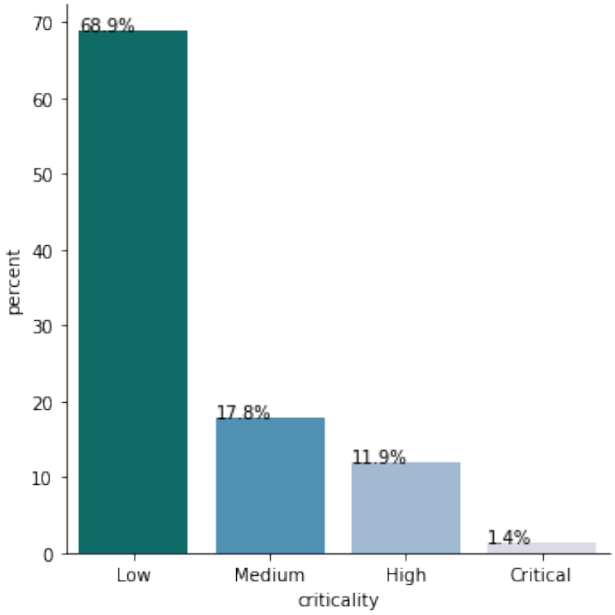

Figure 8: Distribution of the modalities of the *annotation_postPriority* column in the final dataset

3.6 Question 5

Several refinements can be considered to improve the performances of a classifier. Here, we are going to work on the input variables.

This is often the preferred path to significantly increase the quality of a model. Let's have a look at the same models as before, but removing the eight columns we have previously created.

Table 12: Evaluation of the four classifiers with and without eight created variables

Model	F1 score with the eight variables	F1 score without the eight variables
Logisitic Regression	0.680	0.680
Gaussian Naive Bayes	0.209	0.180
Random Forest	0.868	0.833
Decision Tree	0.799	0.777

The F1 score of all the classifiers that we used are about the same with and without the eight variables we have created in Question 2.

4 Conclusion

This study of a Twitter database on specific events has been done on two different aspects: the analysis of a series of tweets as a graph and the prediction of the importance of a tweet.

Concerning the first part, we have seen that the importance of the different tweets is very heterogeneous. Some tweets and users are very important while others can almost be considered as invisible.

In the second part we have seen that it is possible to predict the criticality of a tweet with good performances, using a Random Forest or on on a Decision Tree classifier. Those predictions are based on the characterization of some features of a tweet, that are learnt by the algorithm.

5 References

Graph_DB_description.pdf