Report for Week 4 – 22 April – 26 April

I. Summary

1. I found a solution in order to **reintroduce PCA in the preprocessing pipeline of the t-SNE** process
2. I performed t-SNE with PCA for all timeframes
3. I performed simple **PCA on the dataset (with no t-SNE)**
4. I built the **LDA topics for T13** and plotted all the users and the terms in the same space. (topic space)
5. I performed K-medoids and DBSCAN on Period 13, in the words space (using Tf weights)

II. Actions performed

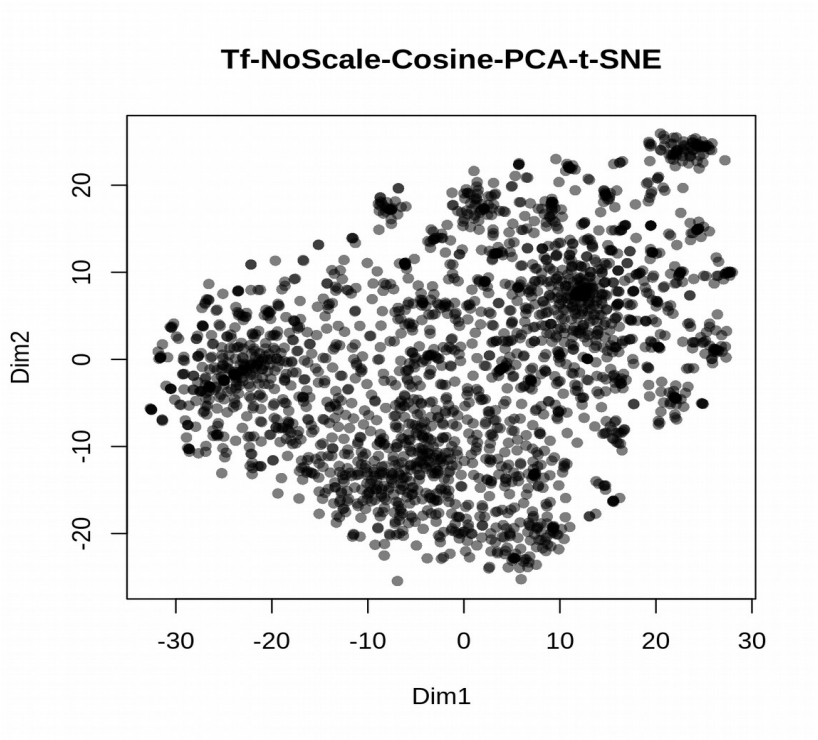
1. Fixing PCA preprocessing step

I discovered the problem with the PCA step which should be done before the t-SNE. The issue is that step of PCA should be done on the Distance Matrix, not on the similarity matrix. By doing this step on the distance matrix, and using the cosine dissimilarity as distance, the results are better and more stable. Please refer to the folder ***t-SNE with PCA vs without PCA*** for images. In all situations, I used as weighting technique **TF** and **cosine dissimilarity**. We can observe that the first 3 images in which PCA step has been used, the results seem to be better contoured and more reproducible. In the last figure, the PCA step has been removed. Performing it several times, the plots differ between successive runs.

Conclusion: **When working in Term Frequency or Tf-Idf, I would suggest sticking to this setup: Tf / Tf-Idf + Cosine + PCA(1-Cosine) + t-SNE**

2. Performing t-SNE with PCA on all timeframes

Once I fixed the problem with the PCA preprocessing performed before t-SNE, I reran the t-SNE algorithm on all timeframes. I did use TF as weightings, no scaling, cosine dissimilarity and PCA reduction (to 50 dimensions) before running the actual t-SNE step. The sparsity of the initial matrix was constant, so, depending on the size of the timeframe, the words space can be bigger or smaller. The results obtained do not only show better contoured clusters, but also are more robust to consequent runnings. In the next figure, there is the 2D representation for timeframe 7. We can see the well defined groups. More figures with all the timeframes can be seen in folder ***t-SNE.***



3. Performing PCA on the dataset, without t-SNE

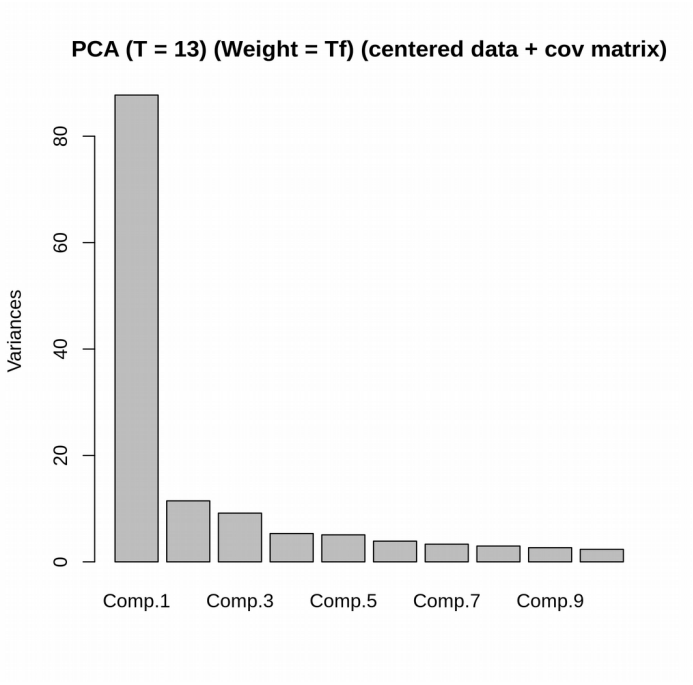
As we discussed, I tried to perform PCA, **with no t-SNE**, on one timeframe. As always, I chose period 13. The tests were done both for TF and for TF-IDF weighting strategies. For this procedure, I did:

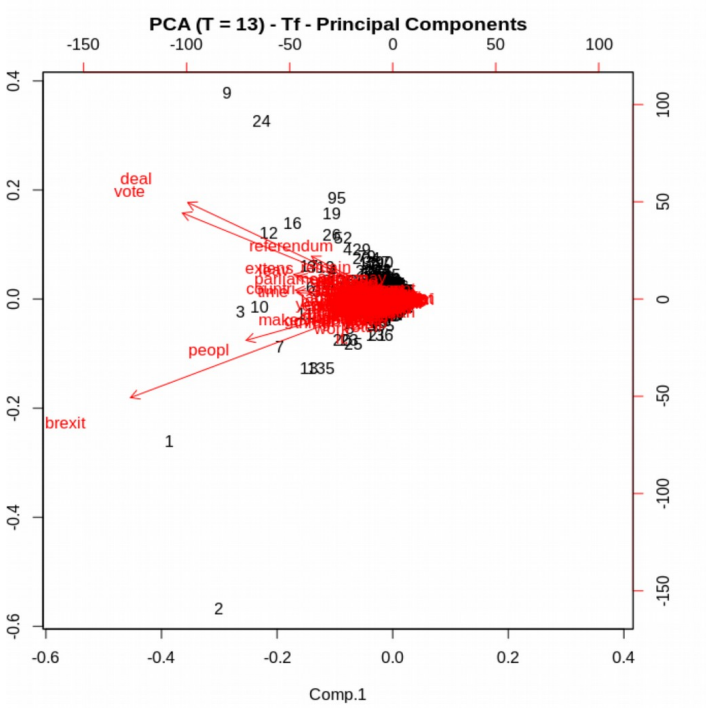
- Create the sparse DTM – the rows are documents (Users in our case) and the columns are terms. The sparsity is controlled so that we have 869 different unique words.
- I center the matrix, by subtracting the column means from the corresponding columns.
- I apply the PCA on the covariance matrix.
- I compute the cummultative explained variance.

These steps have been in turn applied for both Tf and Tf-Idf. Results can be observed in the following set of figures.

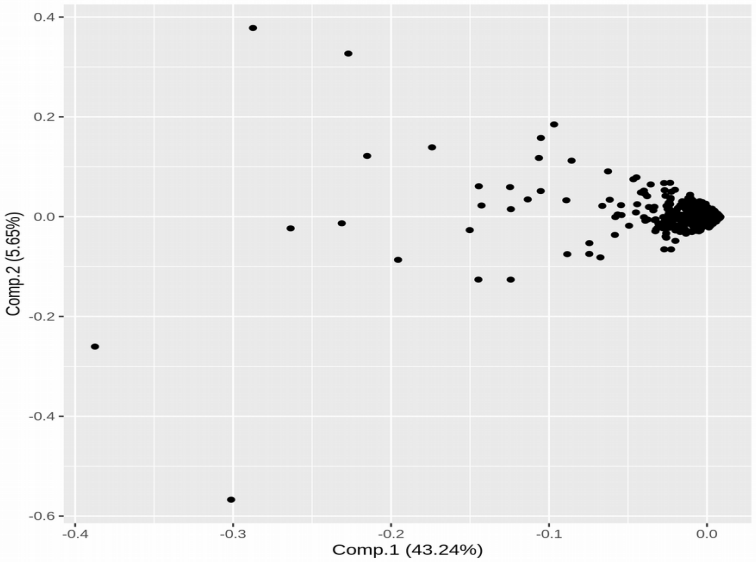
a) TF

	Comp.1	Comp.2	Comp.3	Comp.4	Comp.5	Comp.6	Comp.7	Comp.8	Comp.9	Comp.10
SD	9.3666363	3.38575232	3.02675059	2.30776464	2.25403533	1.97251207	1.82267335	1.72442672	1.63235500	1.53620020
Proportion	0.4323588	0.05649206	0.04514714	0.02624581	0.02503793	0.01917416	0.01637173	0.01465435	0.01313126	0.01162981
Cumulative	0.4323588	0.48885088	0.53399802	0.56024383	0.58528176	0.60445592	0.62082766	0.63548201	0.64861326	0.66024307





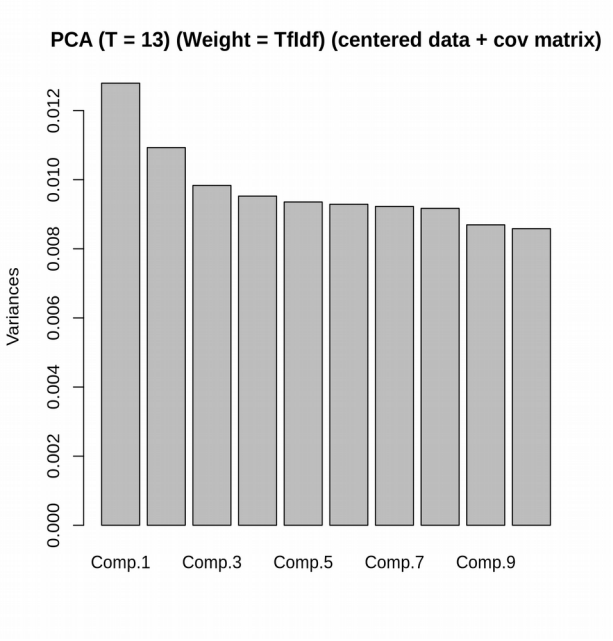
In the above figures we can observe the following. When using TF, most of the variance is explained by the first component (43.23% of the variance is explained by this). From the second row right image we can observe that this component is about **brexit**. The next principal component is by far not so important in terms of explained variance. For instance, it adds only 5.65% explained variance. All in all, when using PCA, we need 10 components to reach 66% of explained variance.

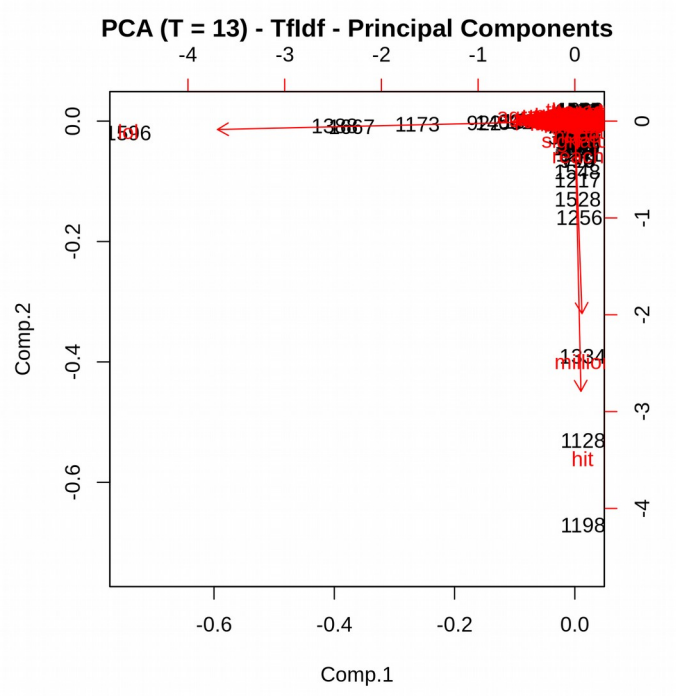


A 2D representation of the resulting data, using only the first 2 principal components is the previous figure.

b) TF-IDF

When using this scheme, the results are a bit worse.



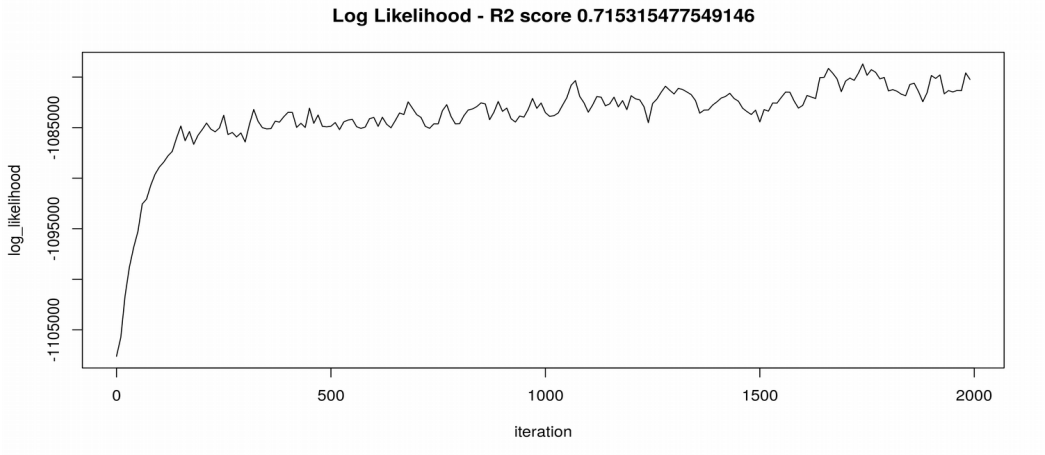


All the figures regarding the Tf-IDF scheme can be observed in folder ***PCA.***

4. Performing LDA topic analysis

As usual, I picked a certain time frame, T13, built the DTM and removed duplicate rows and the ones containing only 0 values. All the analysis was done for TF, as I observed that Tf-Idf resulted in noisy topics.

I ran the LDA algorithm for a range of topics for 10 to 100, with a step of 10. (10, 20, 30, 40 … 100). This algorithm is highly based on expectation maximization, thus, we are interested in the log likelihood maximization. For each topic size, I plotted the log likelihood and the R2 score. R-2 = Explained variation / Total variation. In general, the higher the R-squared, the better the model fits your data.



In the previous figure, there is the log likelihood for **30 topics.** We can see that it grows after a few hundred of iterations, then it plateaus.

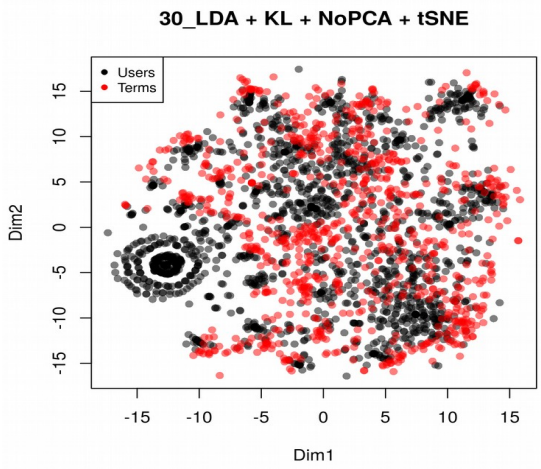
In the next figure, we can see the distribution of terms per topic, in 30 topics scenarion.

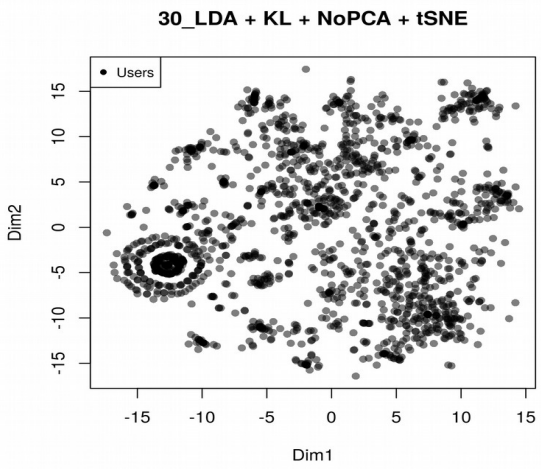
```
     t_1       t_2        t_3      t_4      t_5        t_6       t_7       t_8        t_9        t_10      t_11
[1,] "ampxb"   "europ"    "don"    "peopl"  "britain"  "countri" "think"   "vote"     "petit"    "get"     "agreement"
[2,] "want"    "unit"     "brexit" "brexit" "european" "will"    "brit"    "leav"     "sign"     "will"    "negoti"
[3,] "dont"    "need"     "want"   "dont"   "british"  "just"    "british" "peopl"    "signatur" "compani" "trade"
[4,] "brexit"  "russia"   "like"   "like"   "may"      "like"    "brexit"  "remain"   "just"     "tax"     "deal"
[5,] "ask"     "contin"   "think"  "vote"   "elect"    "power"   "countri" "deal"     "million"  "pay"     "good"
[6,] "believ"  "russian"  "can"    "remain" "extens"   "even"    "dont"    "will"     "can"      "busi"    "state"
[7,] "prevent" "scotland" "doesn"  "want"   "let"      "want"    "now"     "want"     "use"      "trade"   "withdraw"
[8,] "right"   "kingdom"  "thing"  "just"   "brexit"   "problem" "britain" "campaign" "debat"    "buy"     "leav"
     t_12     t_13     t_14       t_15         t_16     t_17     t_18     t_19   t_20         t_21     t_22
[1,] "brexit" "brexit" "war"      "referendum" "brexit" "deal"   "minist" "will" "deal"       "fuck"   "year"
[2,] "peopl"  "will"   "european" "govern"     "mean"   "will"   "brexit" "get"  "may"        "shit"   "live"
[3,] "make"   "polit"  "peac"     "parti"      "post"   "extens" "prime"  "time" "vote"       "say"    "can"
[4,] "like"   "like"   "countri"  "elect"      "word"   "brexit" "said"   "now"  "parliament" "brexit" "peopl"
[5,] "even"   "thing"  "europ"    "vote"       "thank"  "vote"   "may"    "just" "brexit"     "now"    "work"
[6,] "will"   "futur"  "german"   "polit"      "ask"    "may"    "mps"    "yeah" "mps"        "like"   "immigr"
[7,] "nation" "see"    "british"  "labour"     "peopl"  "articl" "govern" "job"  "plan"       "stupid" "say"
[8,] "market" "side"   "well"     "remain"     "use"    "revok"  "call"   "year" "parti"      "just"   "take"
     t_23      t_24         t_25     t_26       t_27     t_28   t_29      t_30
[1,] "brexit"  "parliament" "will"   "ireland"  "think"  "just" "daili"   "vote"
[2,] "peopl"   "law"        "time"   "border"   "just"   "get"  "british" "referendum"
[3,] "may"     "govern"     "extens" "irish"    "dont"   "can"  "express" "peopl"
[4,] "march"   "act"        "may"    "backstop" "like"   "know" "time"    "democraci"
[5,] "remain"  "chang"      "week"   "northern" "say"    "now"  "brussel" "chang"
[6,] "london"  "hous"       "brexit" "deal"     "one"    "hope" "law"     "one"
[7,] "theresa" "rule"       "just"   "custom"   "can"    "like" "want"    "result"
[8,] "day"     "pass"       "day"    "common"   "realli" "tri"  "forc"    "second"
```

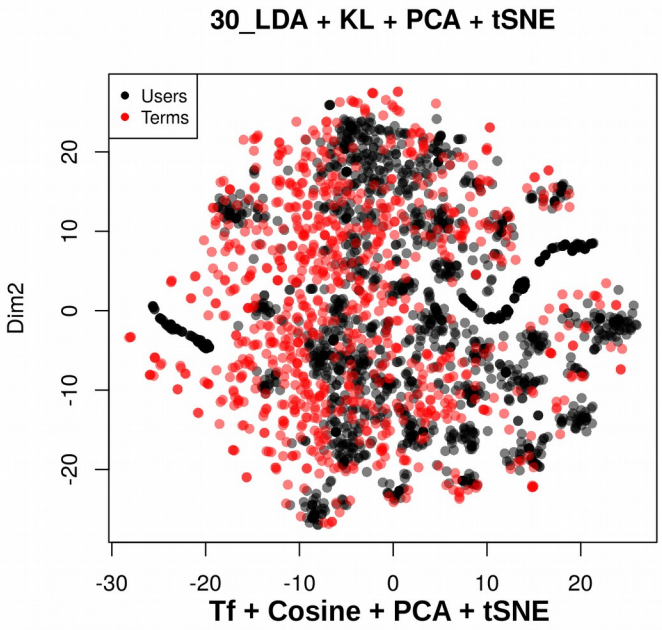
We can observe that the topics are better contoured, for example t_24 talks about the parliament changing the rules, by passing some laws in the house (of lords or commons).

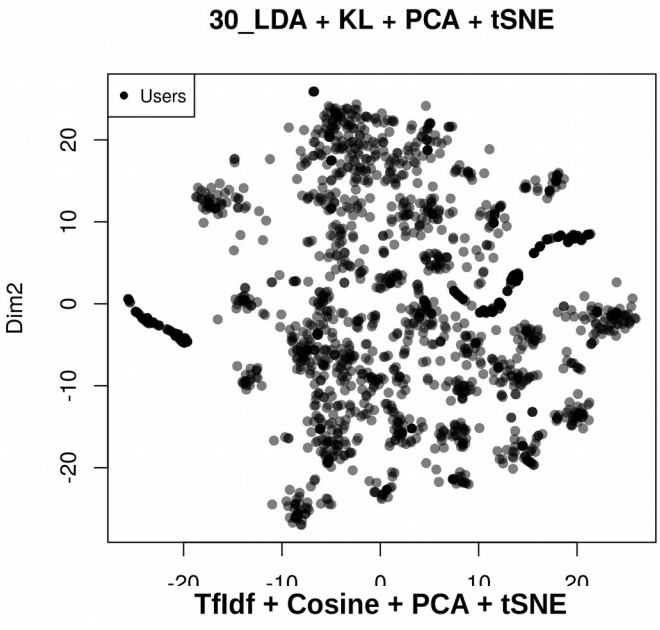
Then I obtained the representations of users and of terms in the topic space, as probability distributions. Using the KL distance between these distributions, I got them plotted after reducing dimensionality (via t-SNE). I also applied in one situation PCA before t-SNE, but results are quite similar.

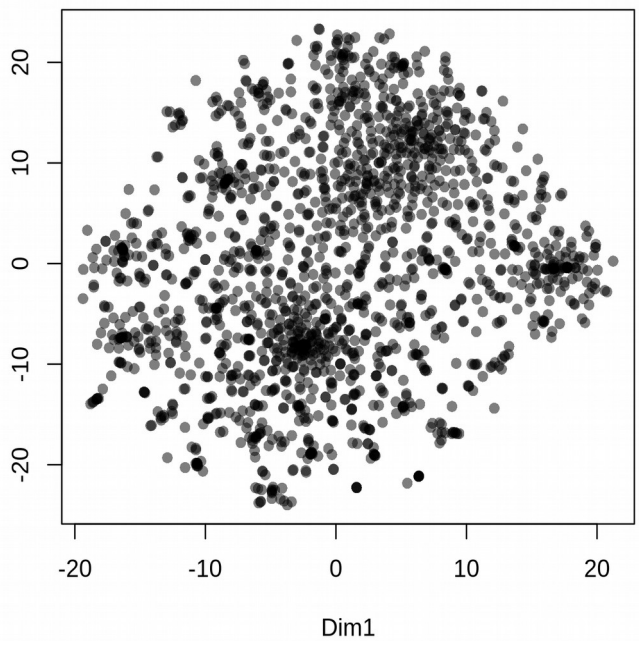
The results can be seen in the following 2 pairs of figures. In the first 2 figures (first row) I did **NOT** applied PCA before the t-SNE. We can observe that strange circle and in the left part of the image. I plotted interactively (will show during the meeting) and in that position we have dump utterances. Basically, there are useless, meaningless replies (either links to youtube etc or meaningless things). This makes me believe that this representation might be accurate. With red, we have the terms and with black the users. We can see that according to the topics (we are in topics dimension), the users are clustered and the terms they use tend to be around them.

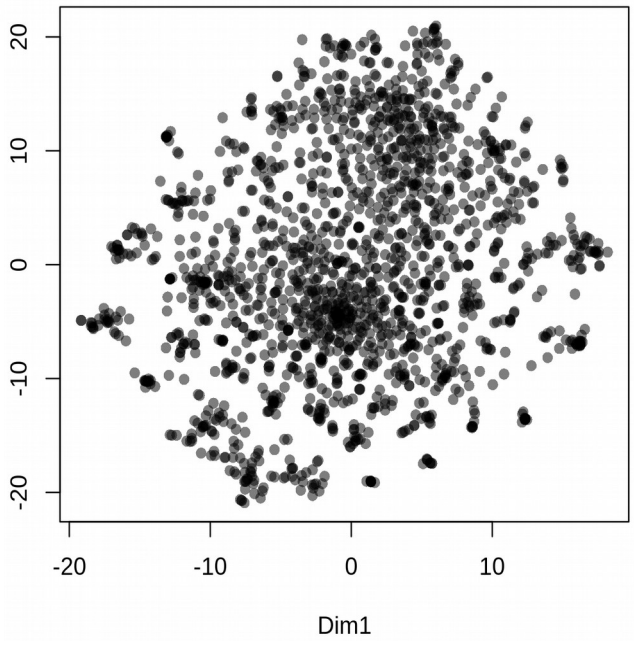

30_LDA + KL + NoPCA + tSNE
Users
Terms
Dim2
Dim1


30_LDA + KL + NoPCA + tSNE
Users
Dim2
Dim1


30_LDA + KL + PCA + tSNE
Users
Terms
Dim2
Tf + Cosine + PCA + tSNE


30_LDA + KL + PCA + tSNE
Users
Dim2
TfIdf + Cosine + PCA + tSNE


Dim1


Dim1

In row 2, we can see the same procedure applied as in the first row, with the slight change of adding PCA preprocessing before applying the t-SNE. Even when we add PCA, we can clearly see clusters of users, in terms of topic space. In the last row (no 3), we can see for the same period of time, the distribution of USERS in the words space. So, the main difference between the last 2 rows is the feature space. In row 2, we are in topics space, while in rows 3, we are in terms space, having Tf, respectively Tf-Idf as weighting strategies.

Images in the topic space of all the topic sizes can be seen in the folder ***LDA.***

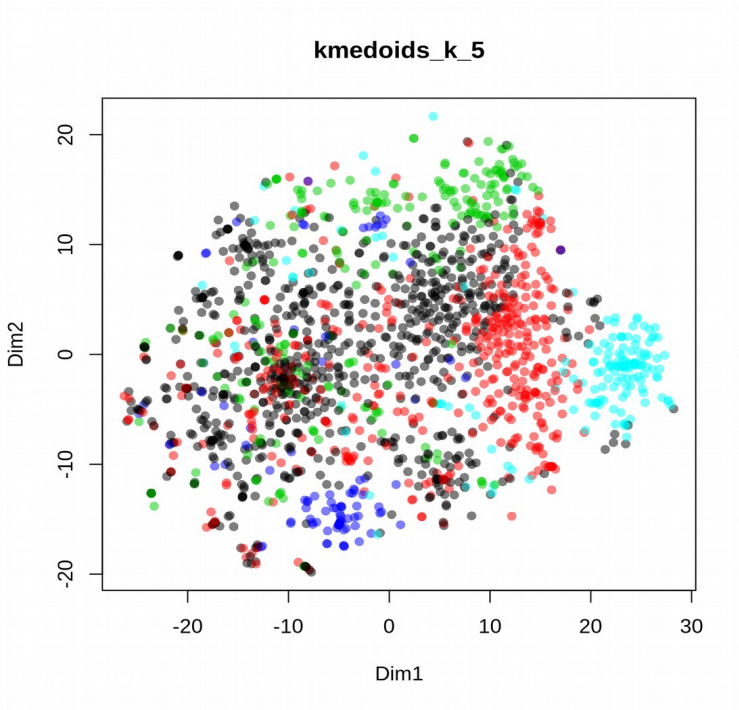
For each topic size, 10 → 100, there is a plot with the log likelihood, and 2 pairs of plots: one without PCA, one with PCA. On the left, side, there are both terms and users in topic space, on the right size, there are only terms in topic space.

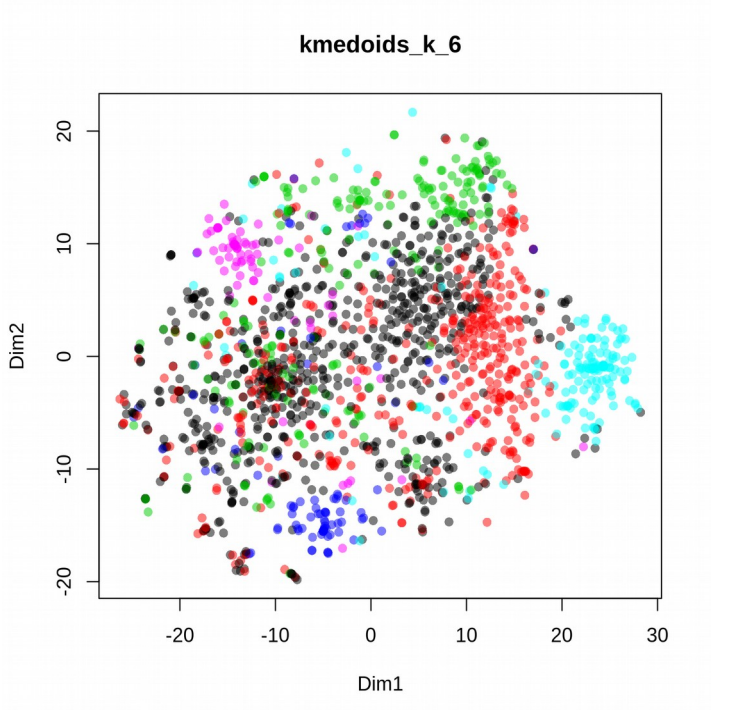
5. Clustering

For the clustering part, I chose the Term Space representation, using TF. I performed the clustering on the well known period 13. I tried both **dbscan** and **k-medoids**. This time, I precomputed the distance matrix, and I use the same distance matrix for both t-SNE and k-meds (the cosine dissimilarity).

a) K-Medoids

I tried different values for K and the results can be observed in folder ***Clustering/K-medoids***. Differently from last clustering try, now, the clusters are more homogenous and for different values of K, they tend to remain the same (increasing K just increasing the cluster number and adds new clusters, but the old ones remain the same). For example, in the next 2 figures, we can see incrementally set of clusters being built.

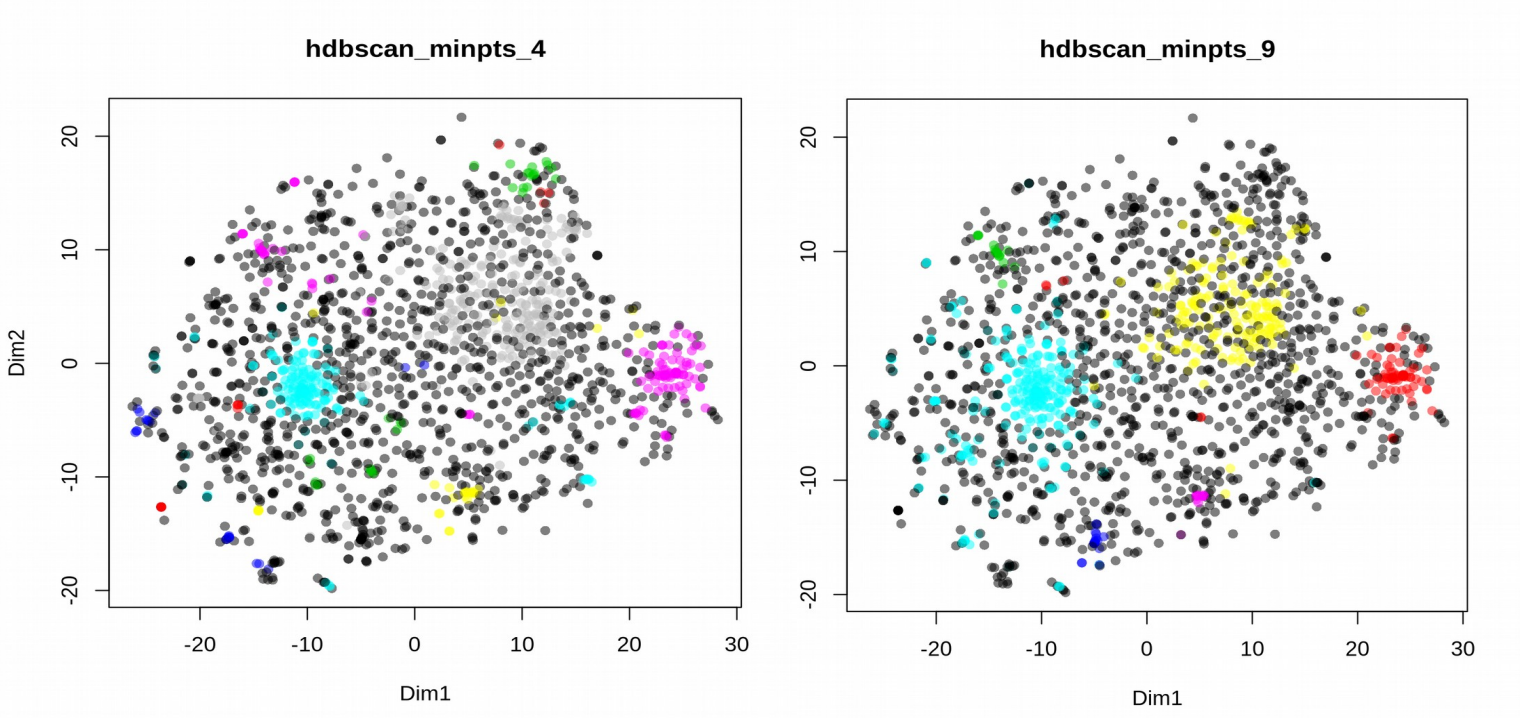




From 5 to 6 clusters, the only difference is that we have one more cluster added, but all the others remain the same. Moreover, we can see the same area of dump messages, non related / useless messages that we saw in the topics representation. (It may be the area where there are no well defined clusters). (middle left zone).

b) DBSCAN

This algorithm takes into account the density of the points, so not necessarily the meaning behind the text each use has posted. For example, 2 users that both post wasteful messages (or web links or whatever) will be clustered together. This can be observed in the following images:



More figures with dbscan can be observed in folder ***Clustering/DBSCAN***.

In folder ***Clustering/K-medoids-Detailed*** I made a thorough analysis of the clusters obtained with the k-Medoids method, by analysing the distribution the set of words from each cluster.