Report 2 for Week 3 – 15 April – 19 April

I. Summary

1. I redesigned the initial 16-cut model, by merging T2 and T3, thus we now have a new cutting model named: **15-cut**
2. I implemented new splitting methods, based on equal periods of time.
3. I changed the default Euclidean distance function used by T-Sne with Cosine similarity for Word Feature Space.
4. Cosine with No-PCA

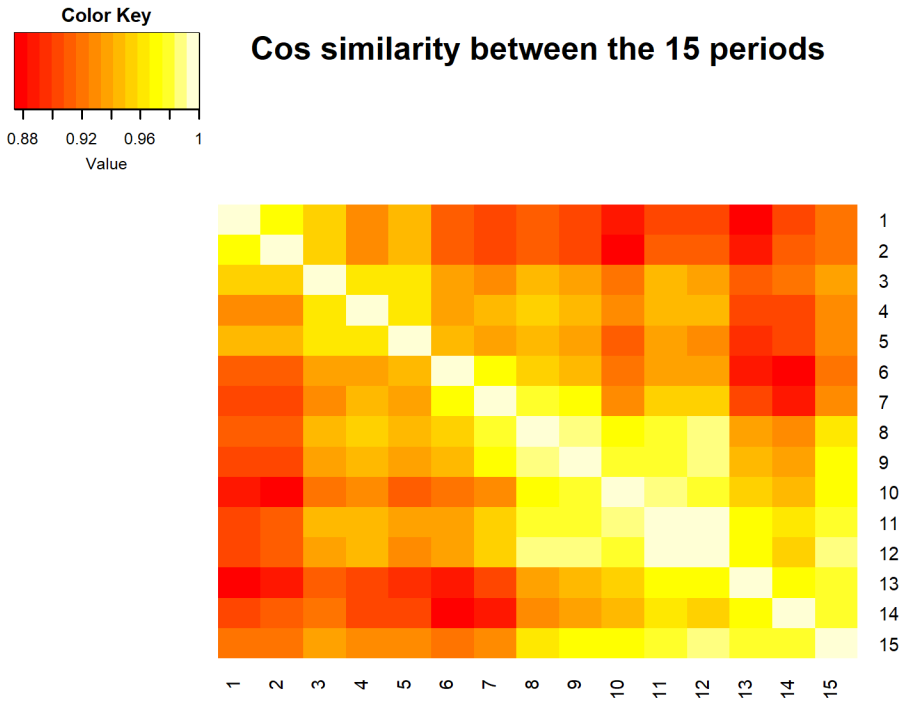
II. Actions performed

1.

Period	Start Date	End Date	Posts	Users
1	2015-11-16	2016-06-24	3367	1268
2	2016-06-25	2016-07-13	6265	1623
3	2016-07-14	2016-12-07	3084	810
4	2016-12-08	2017-01-26	1466	320
5	2017-01-27	2017-03-29	2300	431
6	2017-03-30	2017-06-19	4102	557
7	2017-06-20	2018-07-08	54505	2339
8	2018-07-09	2018-09-21	23067	1479
9	2018-09-22	2018-11-15	15385	1480
10	2018-11-16	2018-11-25	3718	732
11	2018-11-26	2019-01-15	25468	2485
12	2019-01-16	2019-03-14	54850	4489
13	2019-03-15	2019-03-21	9119	1836
14	2019-03-22	2019-03-29	13414	2444
15	2019-03-30	2019-04-05	9509	1840

Having this new model, I recomputed the heatmap for it. To compute this heatmap, I take each Period, aggregate all the utterances from it, and create a DTM matrix, where the D component is the aggregated text for

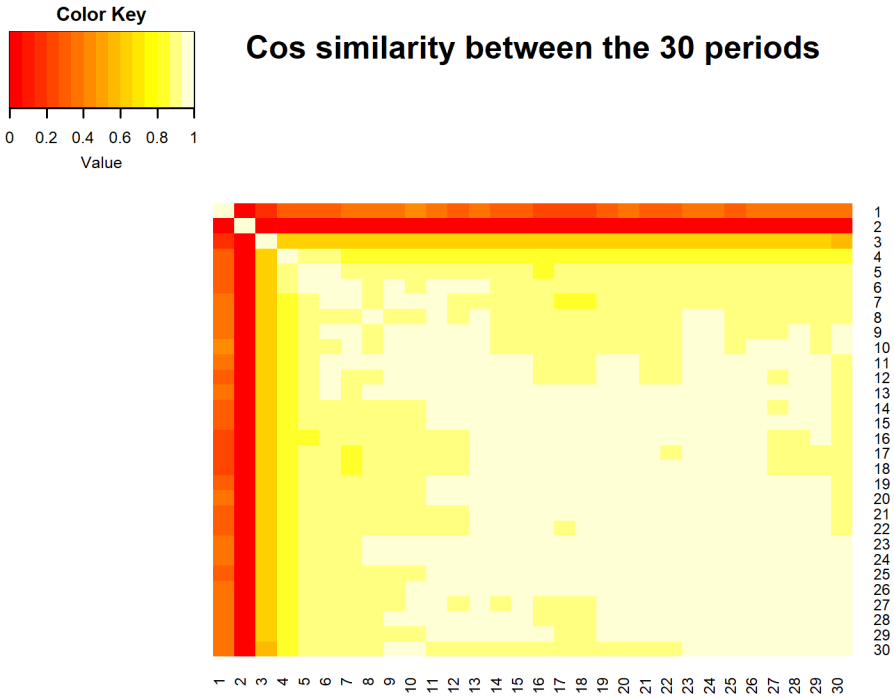
each period. Then, I compute the cosine similarity on rows (how similar / dissimilar is Period I to period II) and create the following heatmap.



We can observe that the problem we had with **Period 3** due to its small size disappeared. Moreover, the gradient structure towards the upper right corner is preserved, meaning that we have a drifting of topics.
The folder **15-cut** contains all the word clouds for each time period (from 1 to 15).

2. **Equal Cut Strategy**

As we discussed, I created a new splitting strategy to check if there are artifacts within the previous selected time frames. Thus, after performing the same steps for obtaining the heatmaps, I get the following heatmaps for 30 equal intervals. Figures for 10 or 40 intervals can be seen in folder equal-cut.



We can observe a strange outlying period, in case of 2. After checking manually the posts, in that period, there is a small number of posts (ie. 2) and they are talking about Nigel Farage, a populist, Eurosceptic politician.

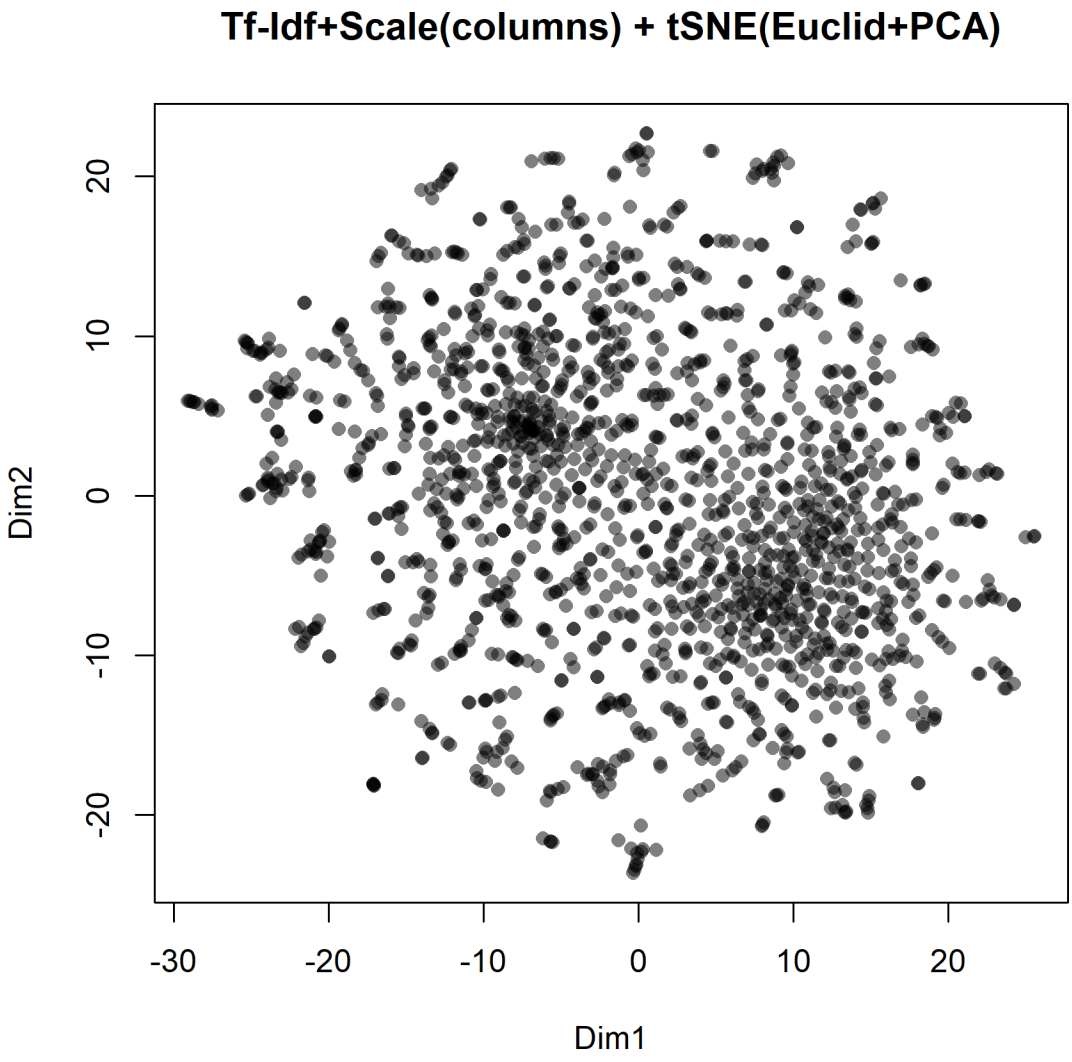
3. **Changing Euclidean to Cosine Similarity**

   This part was particularly difficult because of the R-Tsne function in R. Therefore, I did not know for sure which representation is correct (clearly the first one that has been presented until now is using Euclidean distance, thus it's not appropriate). However, when using the cosine similarity, clusters would now appear, initially. To check if it is ok not to have 2 well defined clusters or not, we define the following cases to be tested, and gradually determine where is the problem:

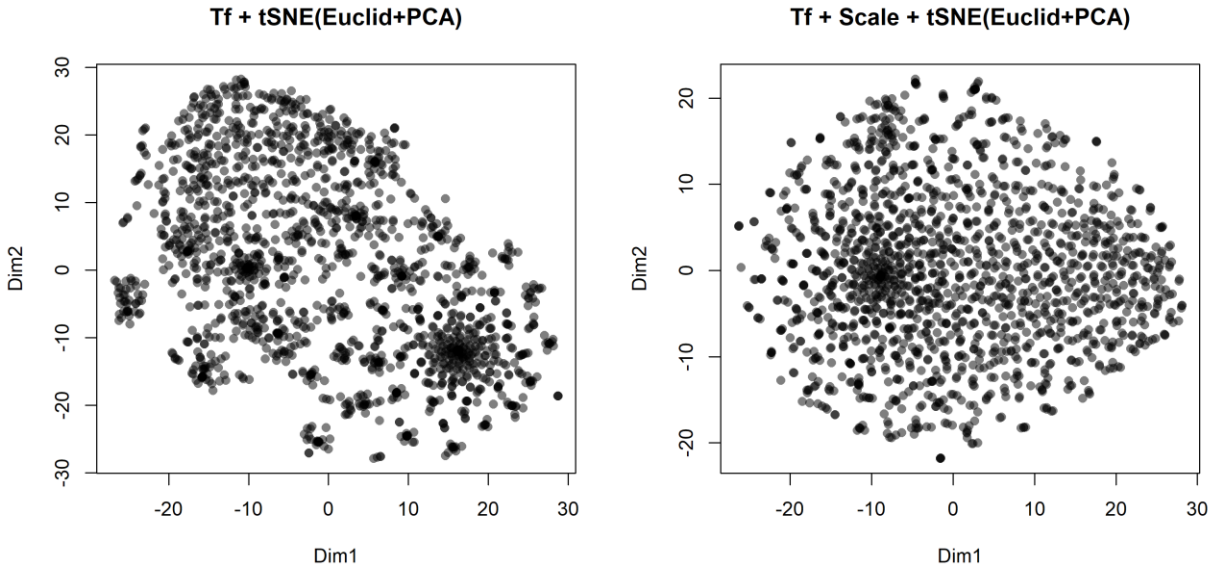
   - **V1**: Tf-Idf + Scale(column-wise) + t-SNE_*{Euclidean + PCA + t-SNE}*
   - **V2**: Tf + Scale(column) + t-SNE_*{Euclidean + PCA + t-SNE}*
   - **V3**: Tf + Euclidean + PCA + t-SNE
   - **V4**: Tf + Cosine + PCA + t-SNE
   - **V5**: Tf-Idf + Cosine + PCA + t-SNE

Explainations:

- **V1** represents the version we have had until now. It is taking a certain period of time, aggregates all the utterances per user, and builds a DTM matrix, with weights Tf-Idf. Then, it eliminates duplicate rows and performs a scaling on columns! This may affect the quality of the outcome, because we are missing the relative difference between different words (different columns). Therefore, all further tests, will be done both with scaling and without to check the results. **After the scaling, I apply directly t-SNE. IMPORTANT! T-SNE has a built-in EUCLIDEAN and PCA preprocessing step.** Only then, the actual dimensional lowering is performed.

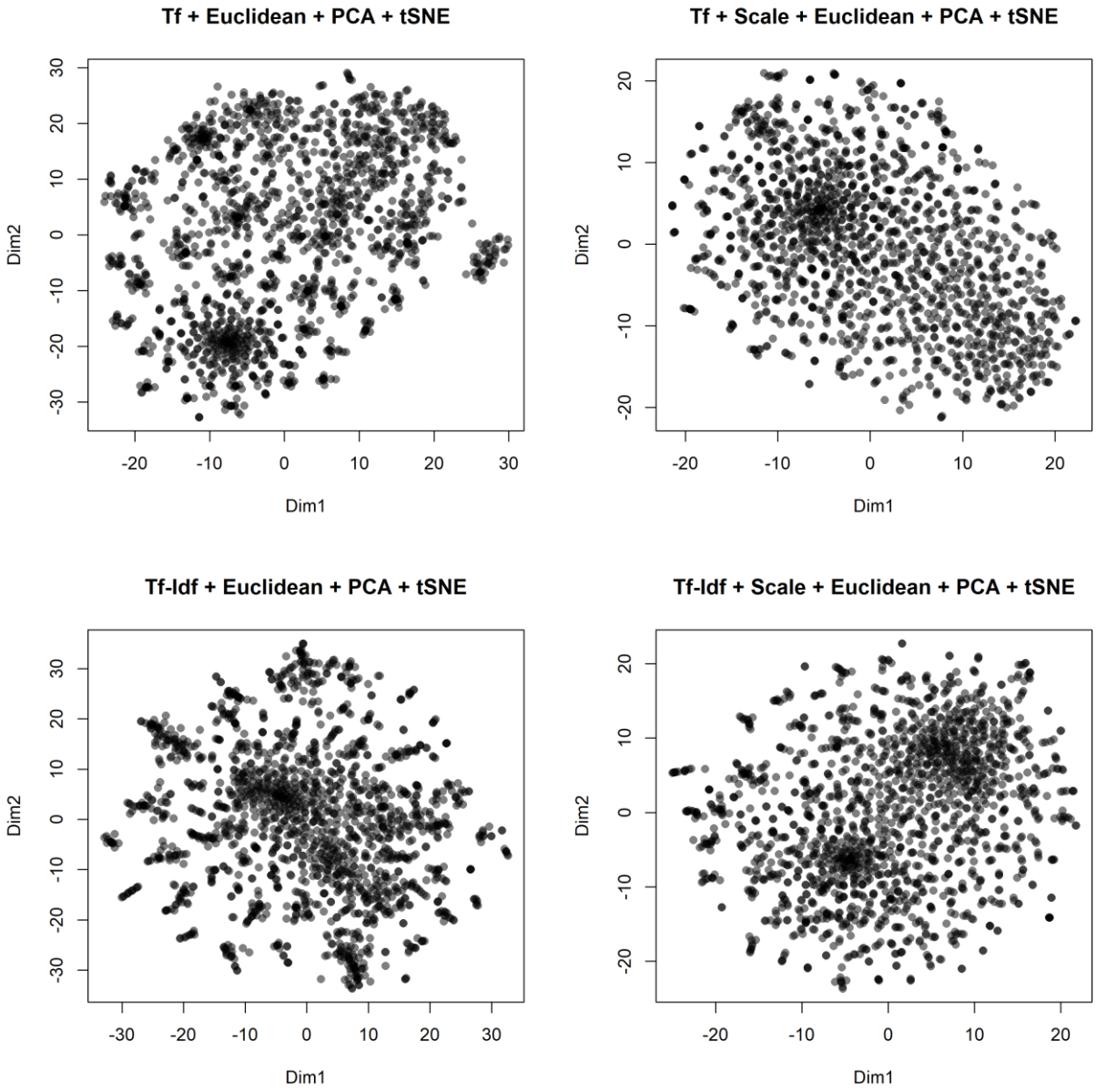


- **V2**: In version 2, we want to check if the simple Tf weighting has the same effect after applying scaling on columns as the Tf-Idf. Thus, we perform exactly the same steps, with the exception of the weights of the matrix. We claim that: **scale(Tf_Idf) = scale(Tf)**



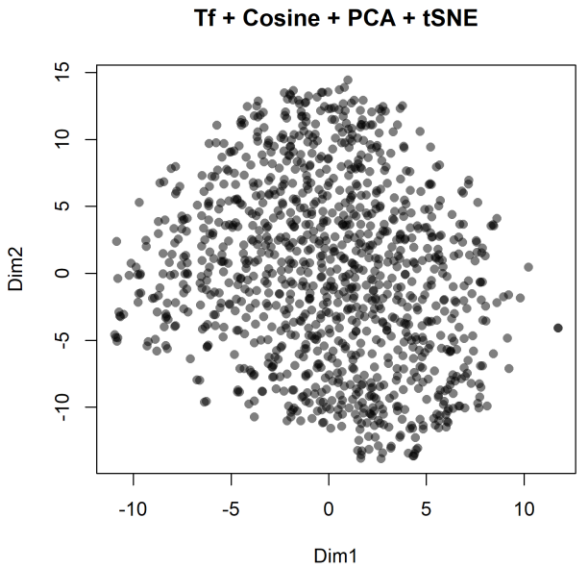
I did both with scaling and without scaling tests. In both situation, we can observe that the initial claim does not hold. **Scale(Tf_Idf) != scale(Tf)**. Moreover, we can see that even if we are using Euclidean distance, when using only the term frequency, we can see more clusters than in case of Tf-Idf (where we have only one cluster).

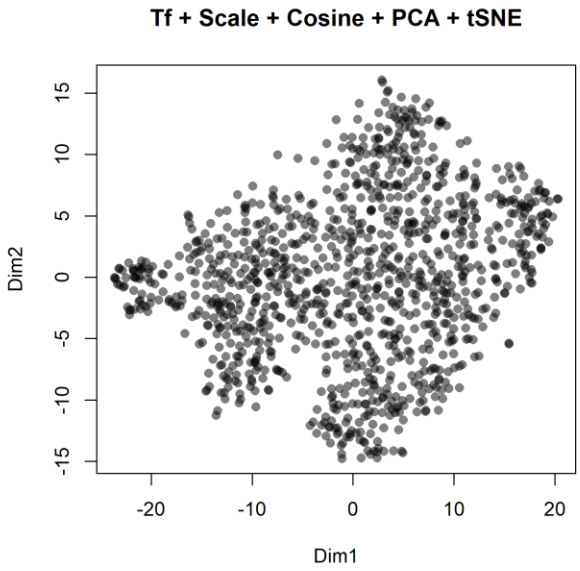
- **V3**: The final goal is to replace Euclidian distance (default) with cosine similarity. When we do the replacement, the R function will automatically cut the PCA preprocessing, as it has no effect on distance matrices. For them, it should be used Multi-dimensional scaling (MDS). Basically, I am trying to externalize the PCA step. However, for V3, we still keep Euclidean distance, in order to be able to compare with V1, to make sure that the externalization was correct.



The results here show that we managed to successfully externalize the PCA step so we are ready for replacing the Euclidean distance with the Cosine sim.

- **V4:** This version proposes for the first time the Cosine similarity. Basically, V4 is V3 in which we replace the Euclidean distance matrix with the Cosine similarity.

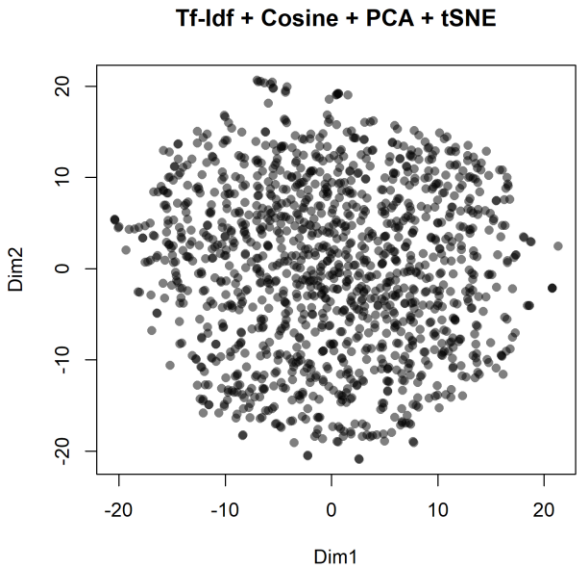


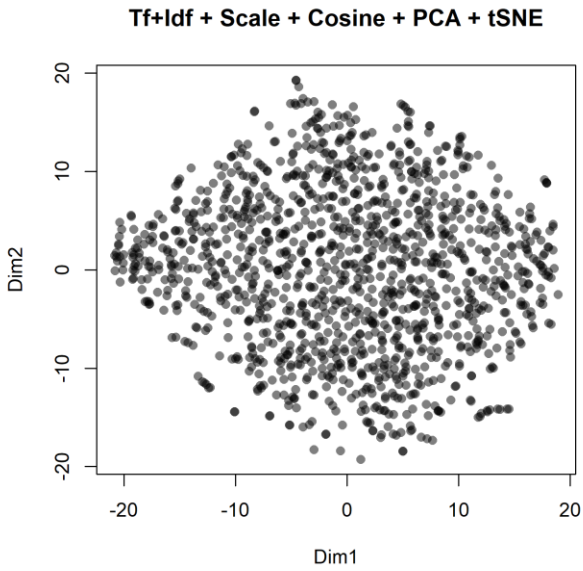


-

Disappointing! When using cosine similarity, we kind of miss all the clustering information. There are hardly any clusters.

- **V5:** We replace Tf with Tf-Idf in order to see if we get better results with the Tf-Idf weighting.



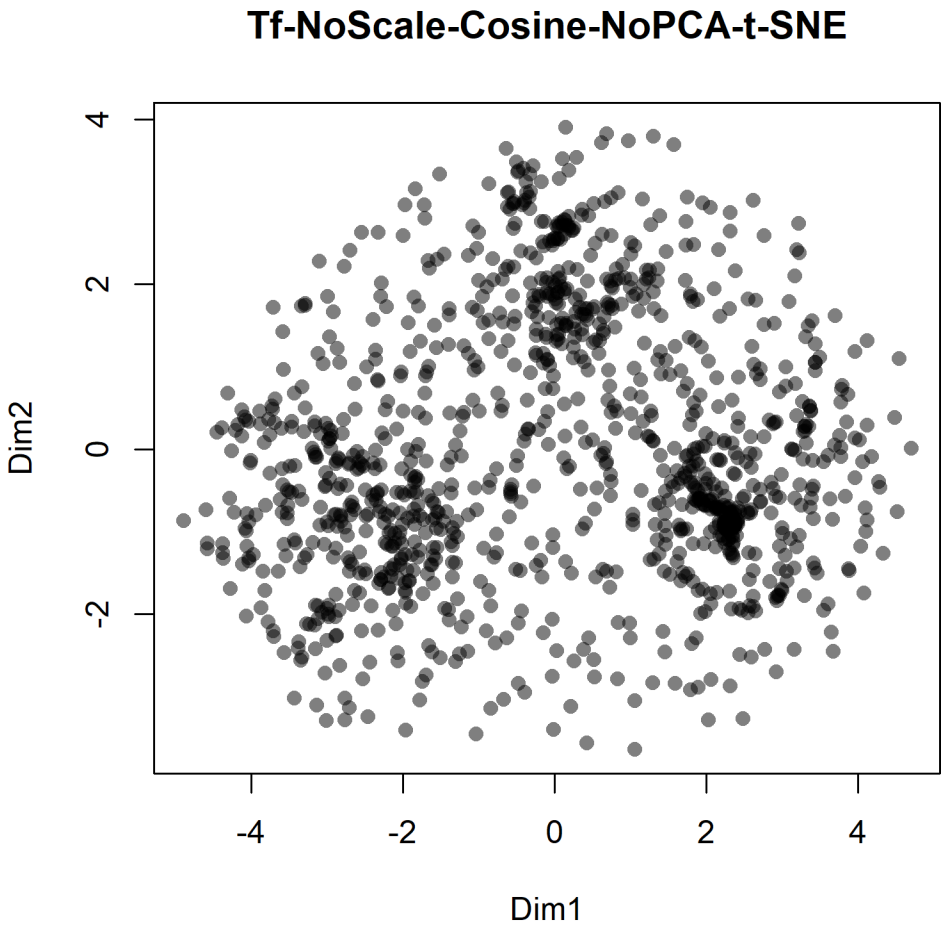


Both when using Tf and Tf-Idf weights, no clusters can be observed.

A possible solution I found is to externalize the distance matrix, but skip the initial PCA step. I did this trick of removing the PCA step because, in a lot of situations, I reach a point where the **mds** complains that there are not even 50

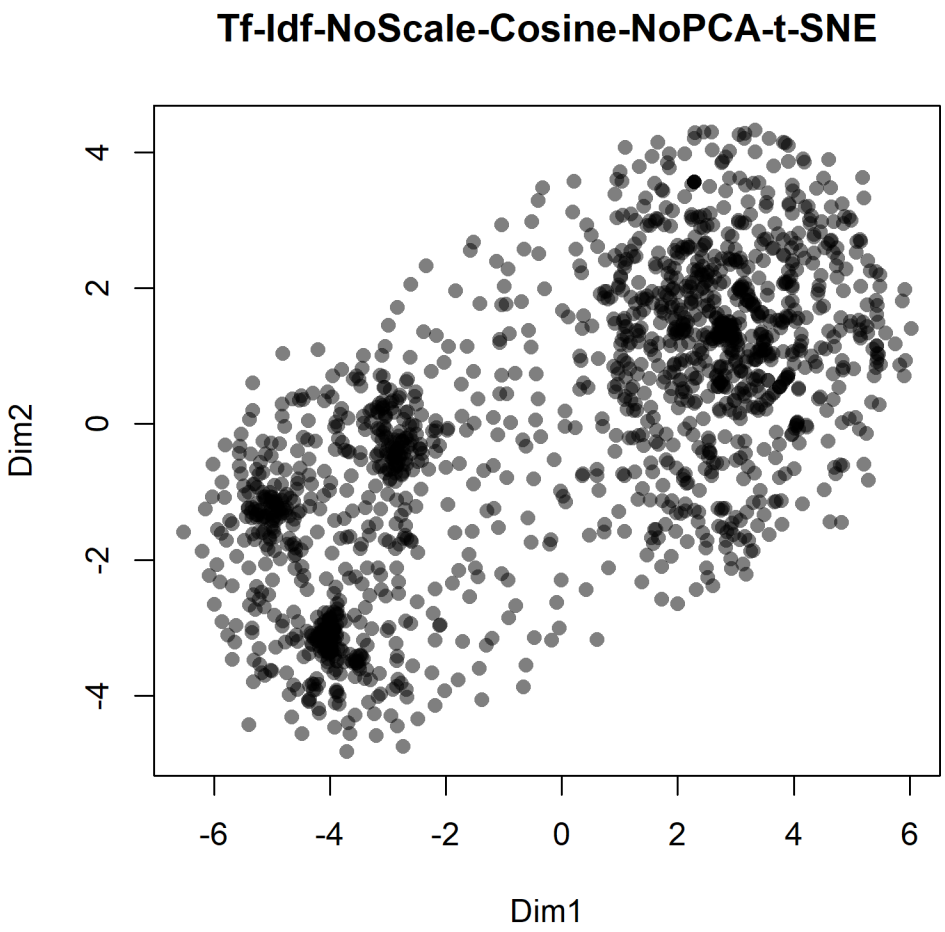
positive eigenvalues, hence less than 50 dimensions resulted after PCA. Moreover, most of the time, all the values resulted after applying PCA are really really small... eg: 3*10^(-13)

TimeFrame 13 (same as last time) – Tf – No PCA



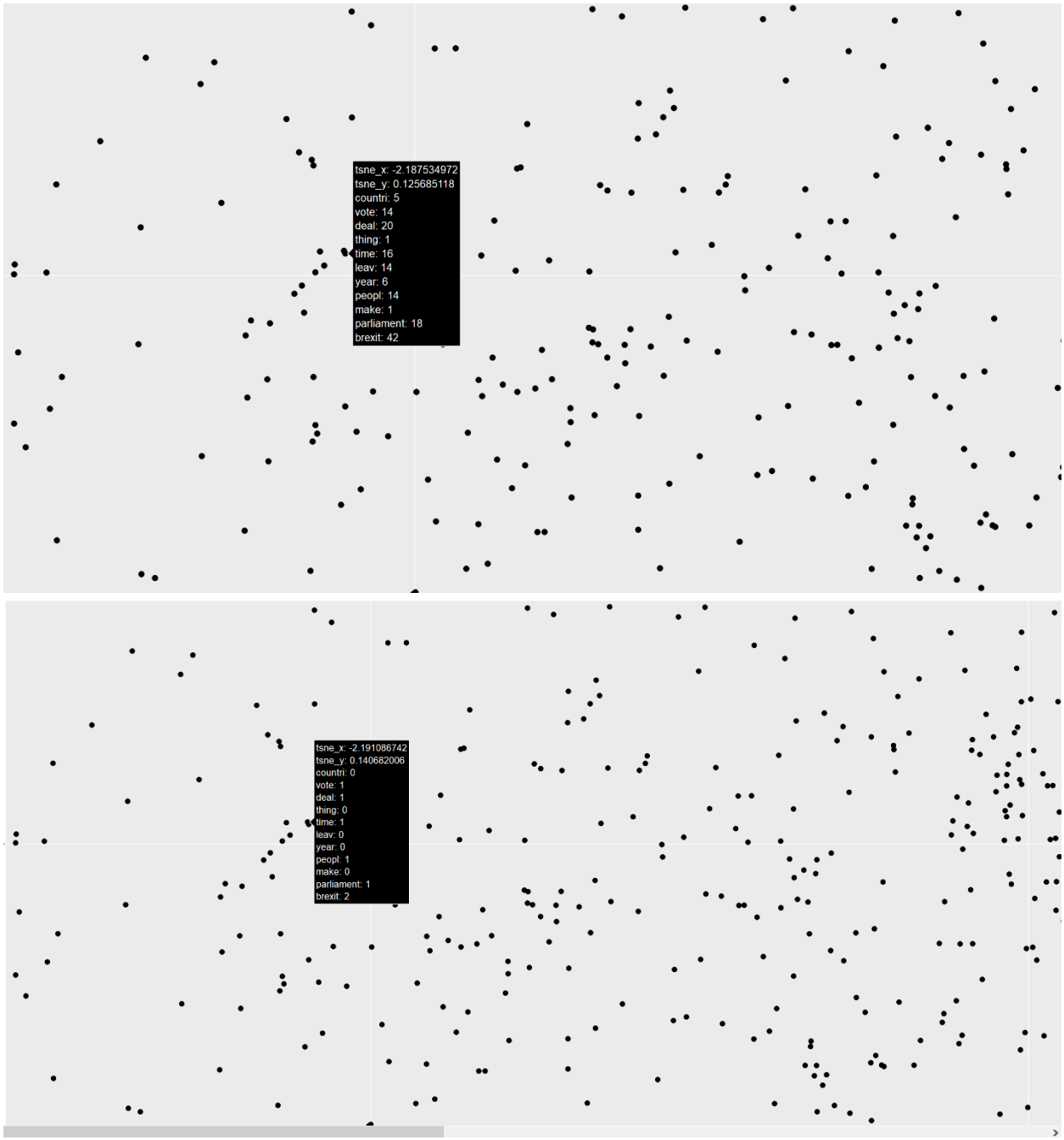
More pictures can be seen in folder: **T13 – Tf – No PCA**. (different word embedding sizes, multiple runs etc).

TimeFrame 13 – Tf-Idf – No PCA



More picture in folder: **T13 - Tf-Idf - No PCA**

To check the correctness, I've performed the same experiment, this time tuning the sparsity so that I have 11 words embedding. Then I did the same trick. Unique rows, no zeros, cosine similarity, r-tsne (using the distance matrix). Then I plot using plotly so that I can hover and see the features.



In the above pictures, I hover on two really close elements. The feature vectors seem quite similar wrt cosine similarity.