EE219 Project 2

Clustering

Winter 2018

Xinyi Gu 205034975

Yutan Gu 005034976

Introduction

K-means is a clustering algorithm that partitions a set of points into k clusters such that the points in each cluster tend to close to each other. K-means is an unsupervised algorithm since the clusters are unknown before the training. In this project, we worked on the clustering on the datasheet “20 Newsgroups” using K-means. We will also try different preprocessing methods to increase the performance of the K-means.

Problem (1)

Similar with project 1, to deal with the text information, we first need to extract features from the documents. We use the TF-IDF vectors as the features of the documents. For this part, we only focus on the dataset of eight subclasses which are shown below.

Computer Technology	Recreational Activity
comp.graphics	rec.autos
comp.os.ms-windows.misc	rec.motorcycles
comp.sys.ibm.pc.hardware	rec.sport.baseball
comp.sys.mac.hardware	rec.sport.hockey

In detail, we remove the stop words and the punctuations, without doing the stem. We set the minimum document frequency of vocabulary term to 3. Finally, we get the dimension of TFxIDF, which is (7882, 18174), indicating we have 7882 documents and 18174 terms in total.

Problem (2)

We then perform the K-mean clustering algorithm to the derived TFxIDF vectors. There are basically two classes, indicating k should be 2. Sklearn provides a function that we can use it to implement this clustering. The result is showing below:

Contingency Table (Confusion Matrix):

[[1353 2550]

[3935 44]]

Homogeneity Score: 0.40871

Completeness Score: 0.44714

V-measure: 0.42706

Adjusted Rand Score: 0.41663

Adjusted Mutual Info Score: 0.40866

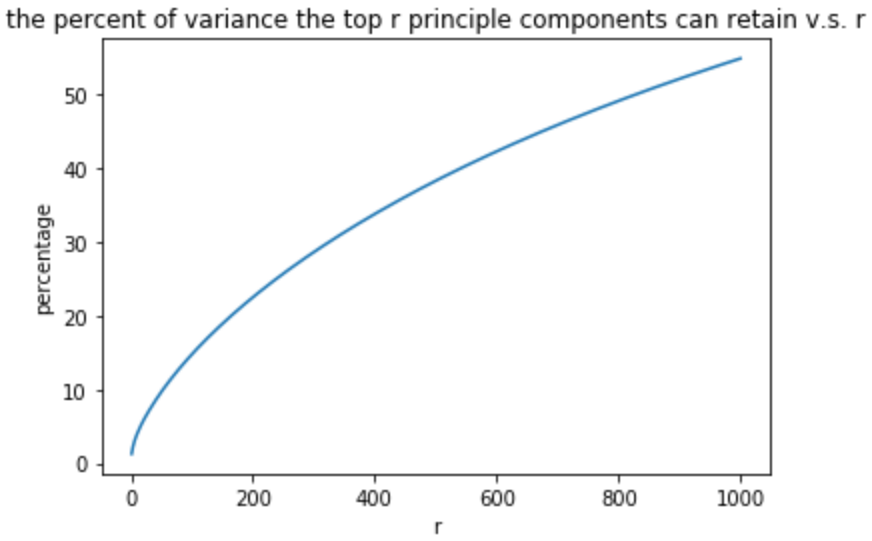
As we can see in the results, this simple process does not provide a good result. For the confusion matrix, an ideal result should be a diagonal matrix. However, the result of the first row is not really well. For the other scalar measurements, they stand for different measures of purity. All the measurements are around 40% percents.

Problem (3)

1. As we discussed above, the high dimensional TFxIDF vector does not provide a good result. This is mainly because of that in the high-dimensional space, the euclidean distance is no longer a good representation of the difference between each cluster.

Thus, we are now trying to do more during the preprocessing stage to improve the feature we extract from the document. Similar to what we have done in project 1, we use Latent Semantic Indexing (LSI) and Non-negative Matrix Factorization(NMF) to reduce the dimension of the matrix. Firstly, we have to decide how many top singular values of the TFxIDF matrix we will preserve during the while calculating the truncated SVD. Thus, we plot a figure which represents the percent of variance the top r principal

components can retain v.s. r. The idea of doing this is that we want to find how much variance of the original data can retain after the dimensionality reduction, while at the same time, we do not want to keep too much principal components as we discussed above.



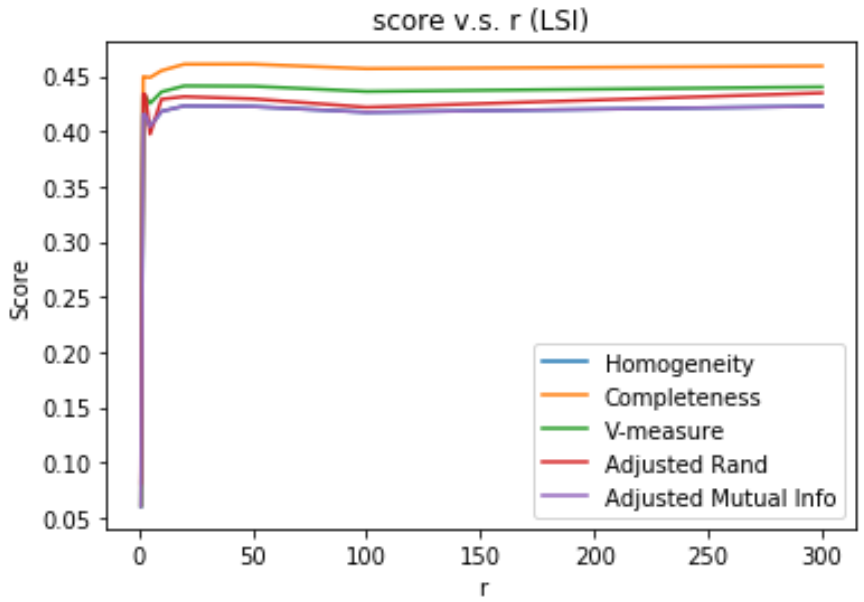
From the plot, we can see that the growth rate is decreasing. At the point of r=1000, we will keep around 60% variance after using the truncated SVD.

2. Next, we will try several different r values to find out which one provides the best clustering result. Also, we will try the two different dimension reduction methods. Note that in the coding part, we set the hyperparameter n_init to 30 as we want to find the best result of each KMeans with its relevant r value. The results are shown as follow:

r	Homogeneity	Completeness	V-measure	Adjusted Rand	Adjusted Mutual Info	Contingency Table
1	0.06012	0.06112	0.06061	0.08094	0.06003	[1688 2215] [2848 1131]
2	0.41564	0.44956	0.43194	0.43417	0.41558	[1285 2618] [3920 59]

3	0.41503	0.44926	0.43146	0.43283	0.41497	[2613 1290] [58 3921]
5	0.40422	0.44881	0.42535	0.39752	0.40417	[1432 2471] [3955 24]
10	0.41815	0.45503	0.43581	0.42917	0.41810	[2588 1315] [44 3935]
20	0.42313	0.46090	0.44121	0.43150	0.42308	[2588 1315] [37 3942]
50	0.42259	0.46095	0.44094	0.42950	0.42254	[2580 1323] [35 3944]
100	0.41735	0.45684	0.43620	0.42188	0.41729	[1347 2556] [3945 34]
300	0.42287	0.45923	0.44030	0.43484	0.42282	[2604 1299] [43 3936]

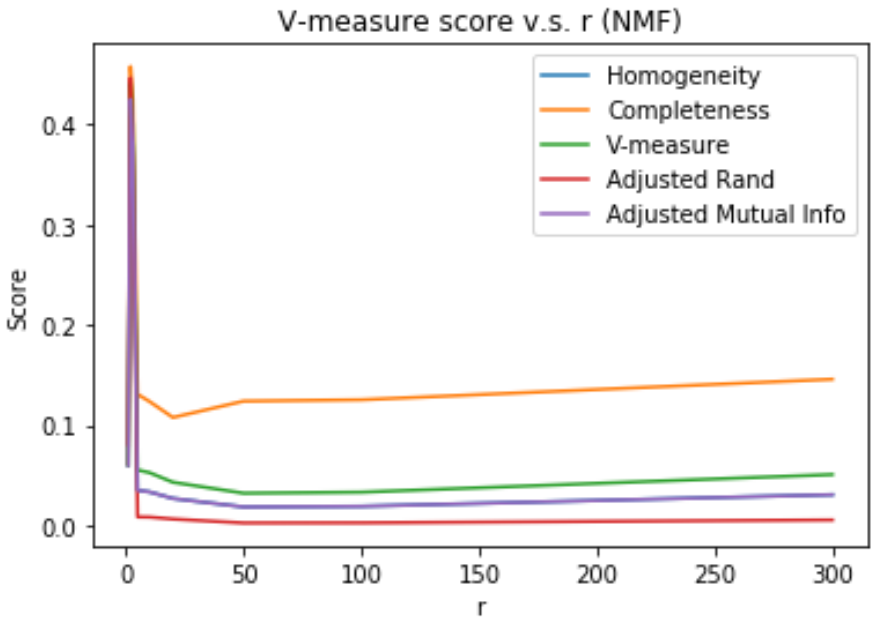
Thus, we plot the figure of scores v.s. r.



In our experiment, r=20 provides the best scores. However, there are just slight differences between the results of 2 to 300 in this LSI method.

For the NMF method, the results are as follows:

r	Homogeneity	Completeness	V-measure	Adjusted Rand	Adjusted Mutual Info	Contingency Table
1	0.06012	0.06112	0.06061	0.08094	0.06003	[1688 2215] [2848 1131]
2	0.42543	0.45808	0.44115	0.44663	0.42538	[2654 1249] [58 3921]
3	0.37800	0.42469	0.39999	0.36893	0.37795	[1514 2389] [3946 33]
5	0.03504	0.13087	0.05528	0.00843	0.03495	[3560 343] [3962 17]
10	0.03348	0.12375	0.05270	0.00829	0.03339	[3559 344] [3958 21]
20	0.02689	0.10771	0.04304	0.00633	0.02680	[3600 303] [3954 25]
50	0.01845	0.12402	0.03212	0.00243	0.01836	[3739 164] [3975 4]
100	0.01909	0.12539	0.03314	0.00256	0.01900	[169 3734] [4 3975]
300	0.03070	0.14589	0.05072	0.00539	0.03061	[258 3645] [4 3975]

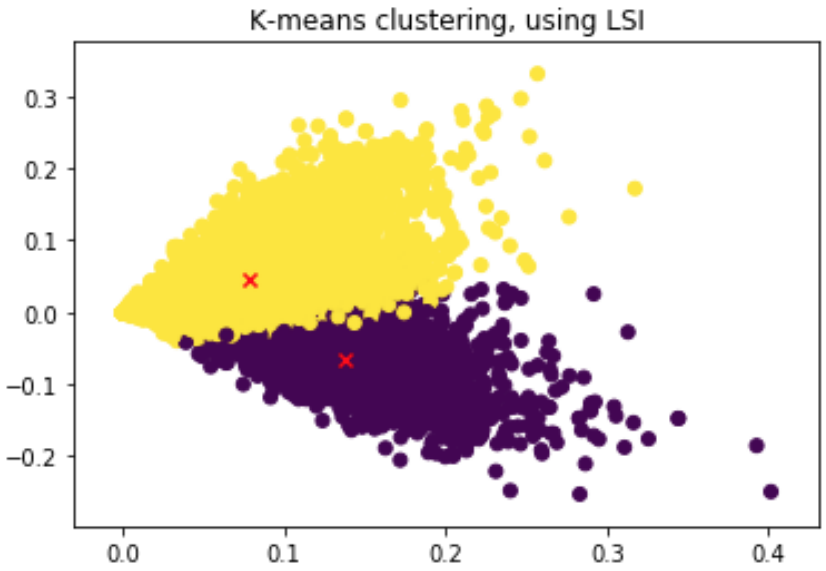


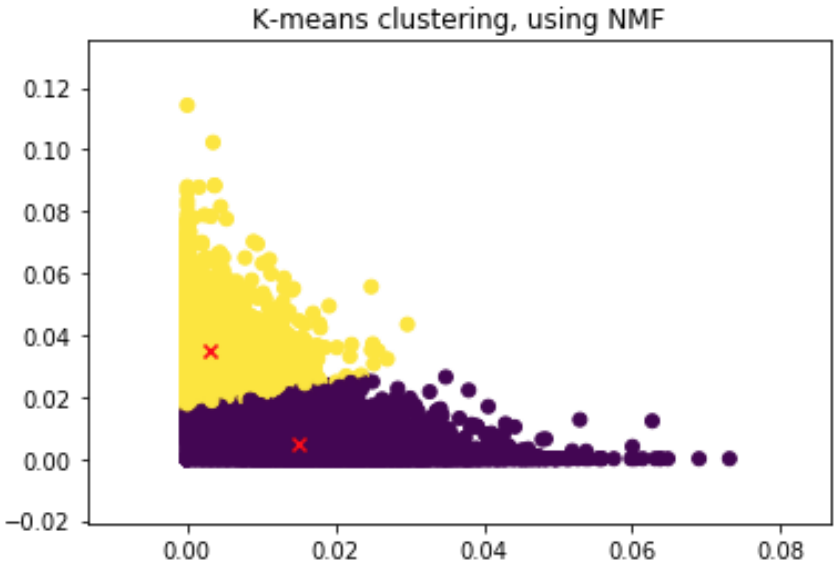
With the NMF method, we can see that there is a peak at r=2. All other options of r give really poor results. From the contingency matrix, it is clear that the clustering process mis-clustered most data points to one cluster, leaving the other cluster almost empty.

It seems that as r increases, the measurement scores behaves non-monotonically. This is because of that there are mainly two factors that make contribution to the final cluster result accuracy. One is the matrix variance retained after dimensionality reduction, which will increase as r increases. The other one is related to the K-means algorithm. As we discussed above, as r increases, the euclidean distance performs worse and worse in the high-dimensional space. Thus these two factors make the non-monotonicity of the measurement scores.

Problem (4)

From optimization in problem 3, we choose the reduction dimension for LSI to be 20, and the best reduction dimension for NMF is 2. To see the result of the clustering better, we make a scatter plot for both two methods. The visualization results are shown in the following figures. The cluster center are marked as red “X”.





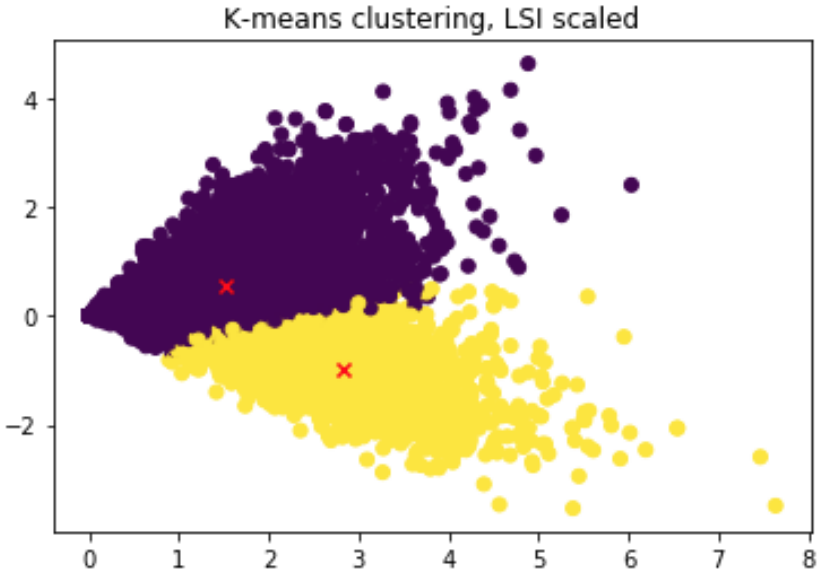
To increase the performance of the clustering, we make normalization and non-linear transformation on the preprocessing data. First, we normalize the data so that all data has the unit variance. We plot the confusion matrix and the visualization result:

LSI with normalization:

Contingency Table (Confusion Matrix) :

[[2086 1817]

[16 3963]]

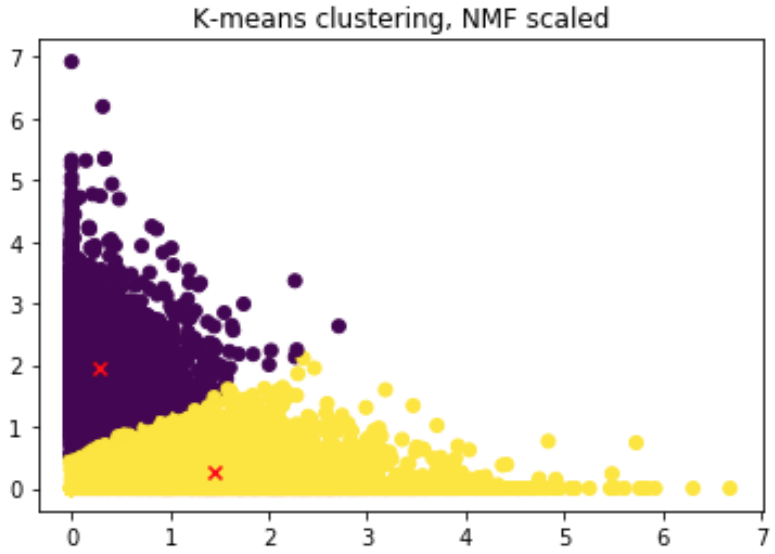


NMF with normalization:

Contingency Table (Confusion Matrix):

[[2972 931]

[97 3882]]



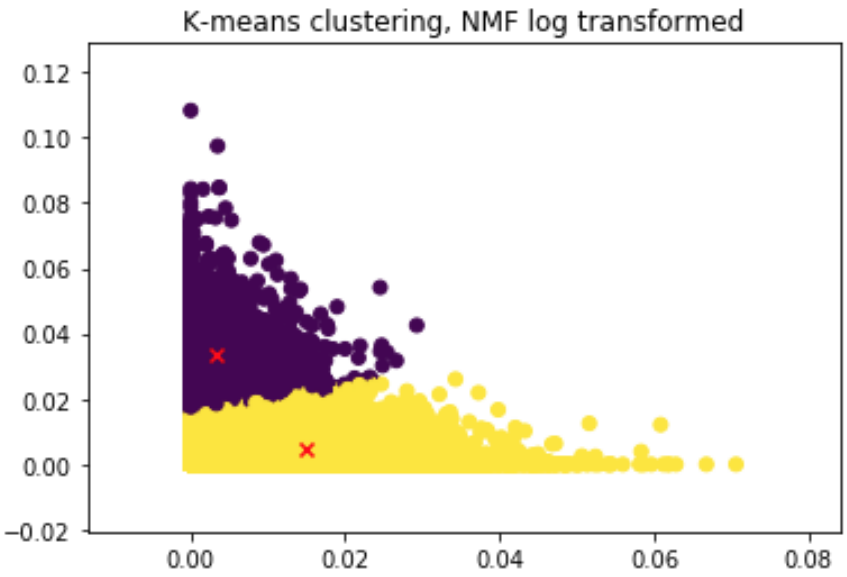
Then we apply the logarithmic transformation as the non-linear transformation to the data after NMF. Similar to the previous part, the confusion matrix and the visualization is shown below. Because k-means does not normalize the data, so the clusters will be strongly influenced by the magnitude of points. Log transformation will make the influence of the large magnitude points less important, and the small magnitude points more important. From the visualization plot, we can see the log transformation help balance the clusters, therefore the result after transformation is more accurate.

NMF with log transformation:

Contingency Table (Confusion Matrix):

[[2682 1221]

[62 3917]]



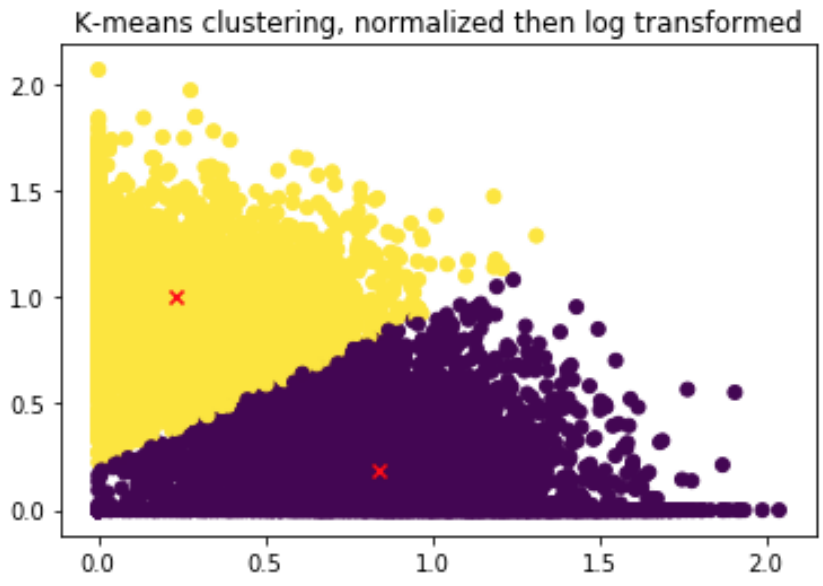
Finally, we combine two methods together with different orders:

NMF with 1) normalization and 2) log transformation:

Contingency Table (Confusion Matrix):

[[694 3209]

[3815 164]]

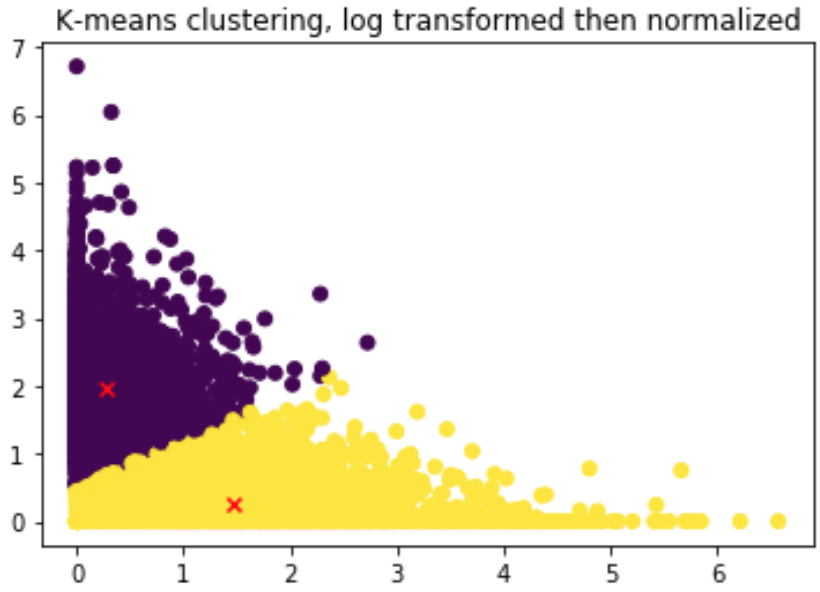


NMF with 1) log transformation and 2) normalization:

Contingency Table (Confusion Matrix):

[[2990 913]

[103 3876]]

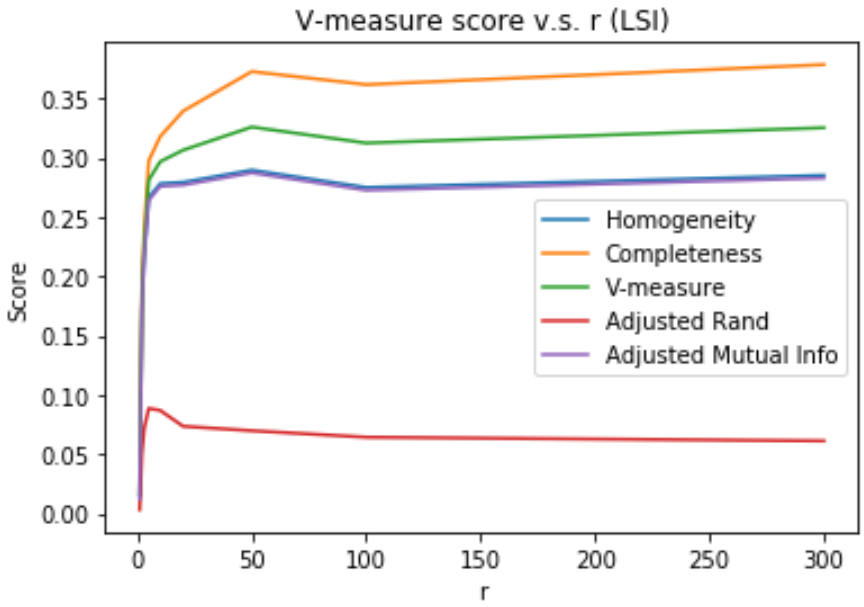


Problem 5

In this problem, we expand the dataset from 8 categories to 20 categories. Similar as the previous parts, we first extract documents features using TFxIDF methods. Since TFxIDF matrix is sparse and has high dimension. We apply truncated SVD(LSI) and NMF methods to the matrix in order to reduce the dimension. Then we try different dimensions for both truncated SVD and NMF methods in order to find the best dimensions for each. The plots of V-mesure score vs. dimensions for each methods are shown below. From the pictures below, we can conclude that the best r for LSI is 50, and the best r for NMF is 10 in this problem.

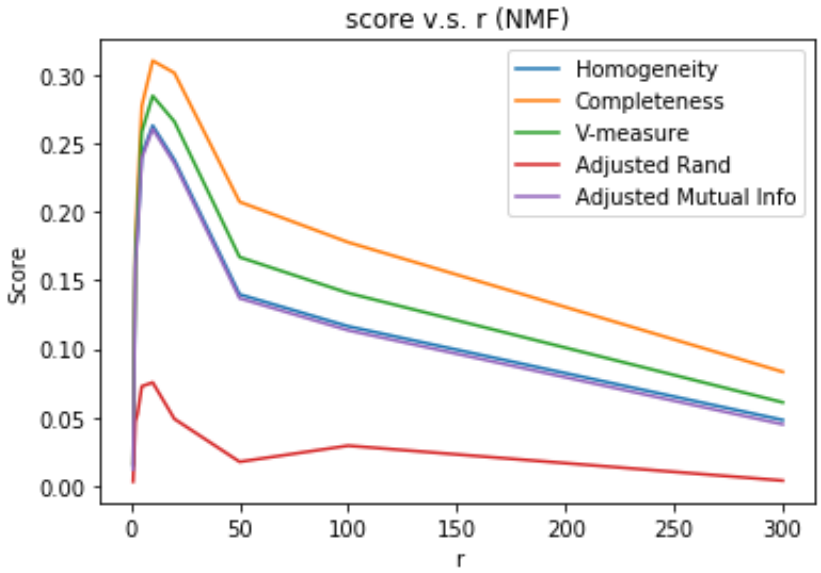
LSI

r	Homogeneity	Completeness	V-measure	Adjusted Rand	Adjusted Mutual Info
1	0.01524	0.01648	0.01584	0.00313	0.01204
2	0.17450	0.18700	0.18054	0.05086	0.17183
3	0.21854	0.23728	0.22753	0.07192	0.21601
5	0.26657	0.29760	0.28123	0.08859	0.26419
10	0.27824	0.31799	0.29679	0.08693	0.27590
20	0.27937	0.33943	0.30648	0.07352	0.27702
50	0.28964	0.37279	0.32600	0.06978	0.28732
100	0.27502	0.36176	0.31248	0.06430	0.27265
300	0.28523	0.37864	0.32536	0.06119	0.28288



NMF:

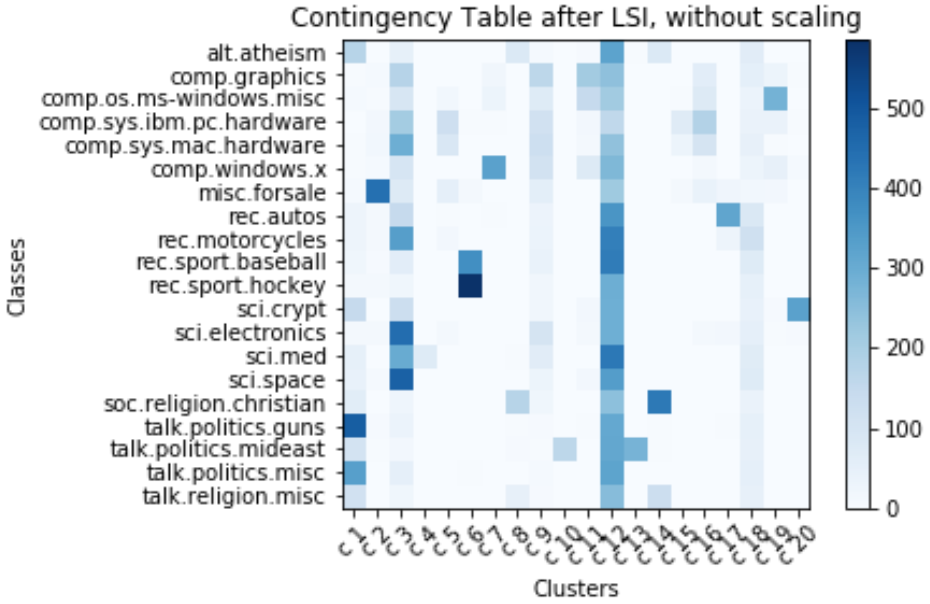
r	Homogeneity	Completeness	V-measure	Adjusted Rand	Adjusted Mutual Info
1	0.01525	0.01650	0.01585	0.00313	0.01205
2	0.16252	0.17309	0.16764	0.04691	0.15980
3	0.18564	0.21148	0.19772	0.05247	0.18300
5	0.24168	0.27714	0.25820	0.07262	0.23922
10	0.26274	0.31010	0.28446	0.07540	0.26034
20	0.23769	0.30114	0.26568	0.04883	0.23521
50	0.13965	0.20712	0.16682	0.01759	0.13682
100	0.11637	0.17765	0.14063	0.02932	0.11345
300	0.04806	0.08315	0.06091	0.00387	0.04491

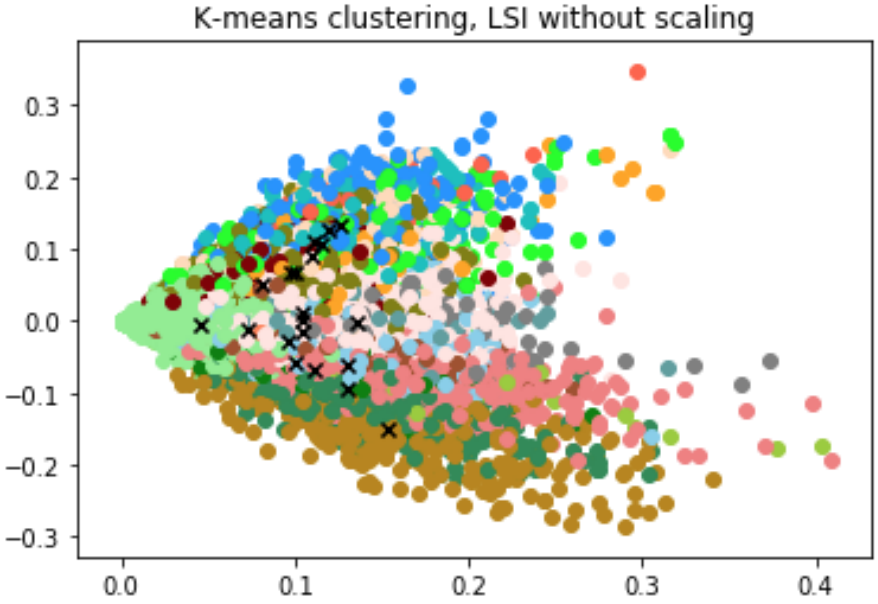


LSI visualization:

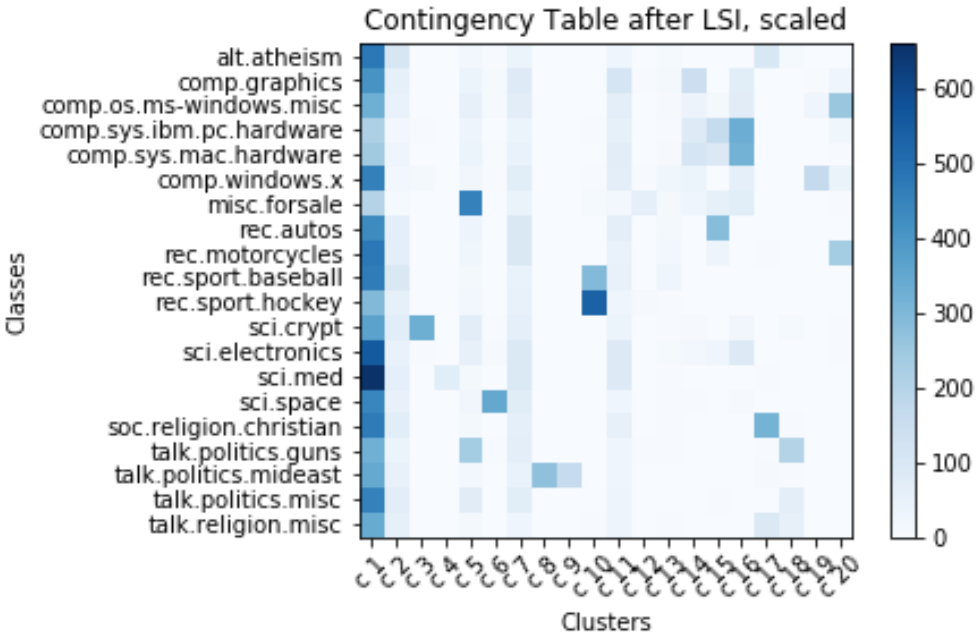
Then we first visualize the contingency table and the clustering result of truncated SVD. To improve the clustering performance, we then normalize the data such that it has unit variance. The visualizations of contingency table and clustering result are shown below as well.

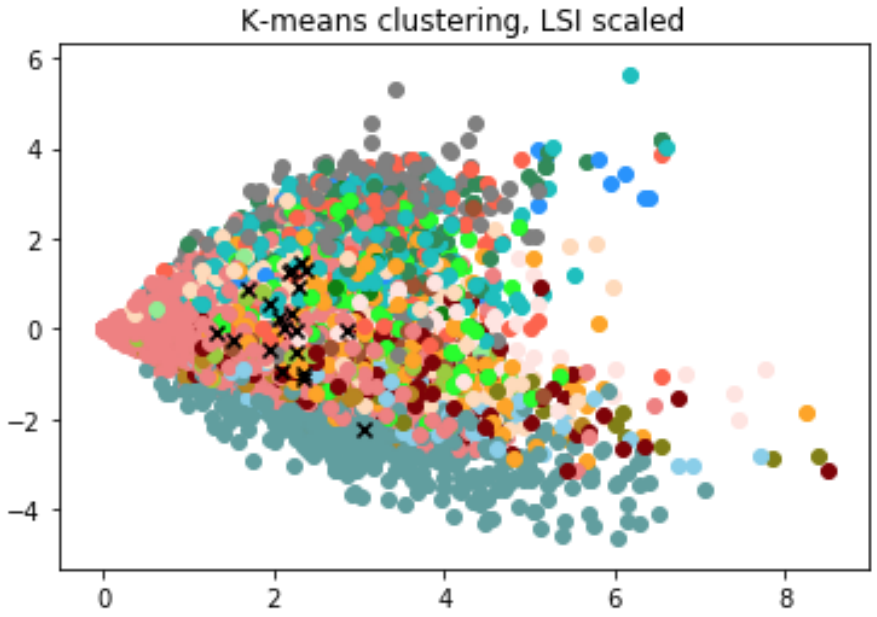
Without normalization:





With normalization:



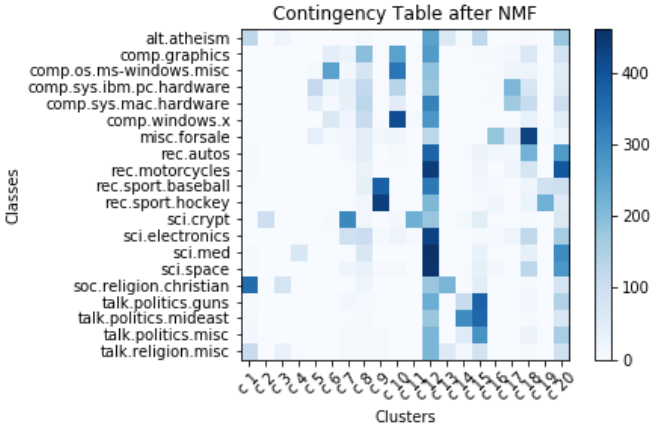


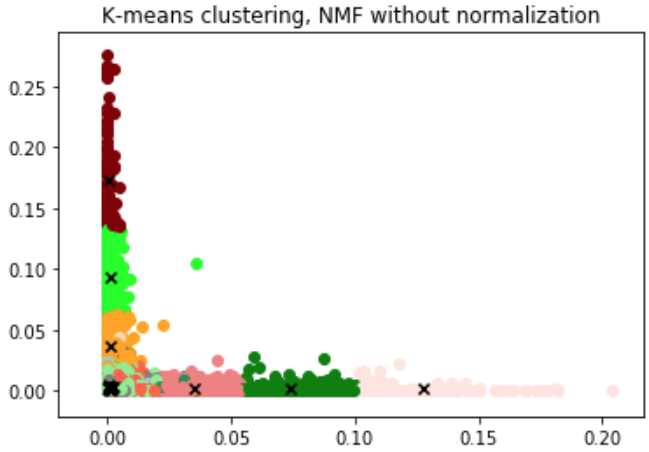
NMF visualization:

Finishing the visualization of the LSI part, we apply the same process to NMF part. The first two pictures show the contingency table and the clustering result. To improve the clustering performance, we then normalize the data such that it has unit variance. The visualizations of contingency table and clustering result are shown below as well.

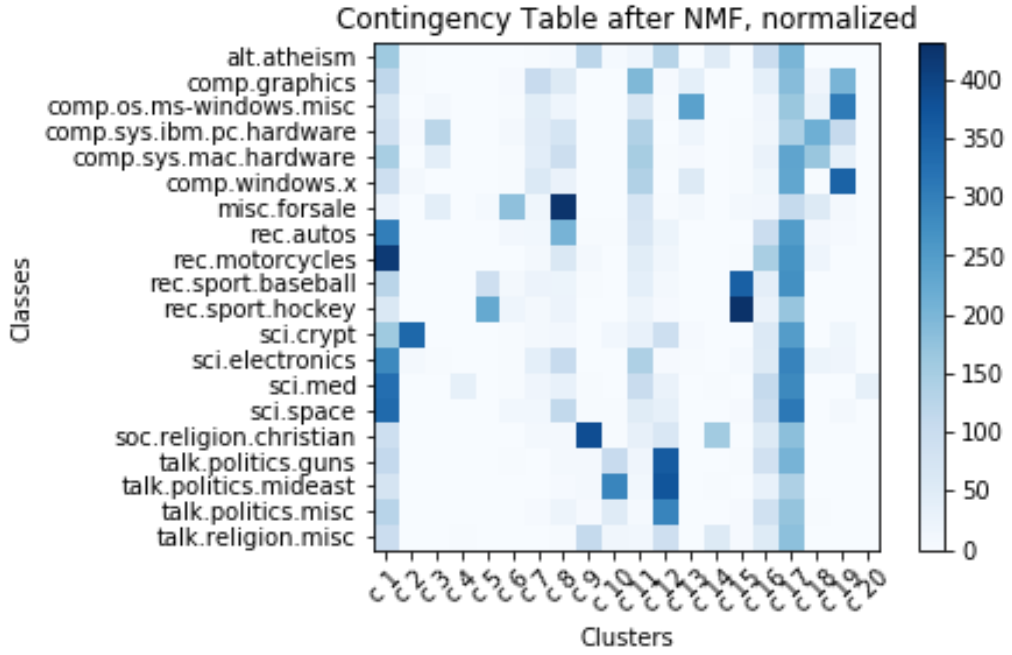
Different from the LSI part, we apply the log transformation to the data after NMF and show its result. Moreover, we also try the different combination of two scaling methods and plot the result in the last four pictures.

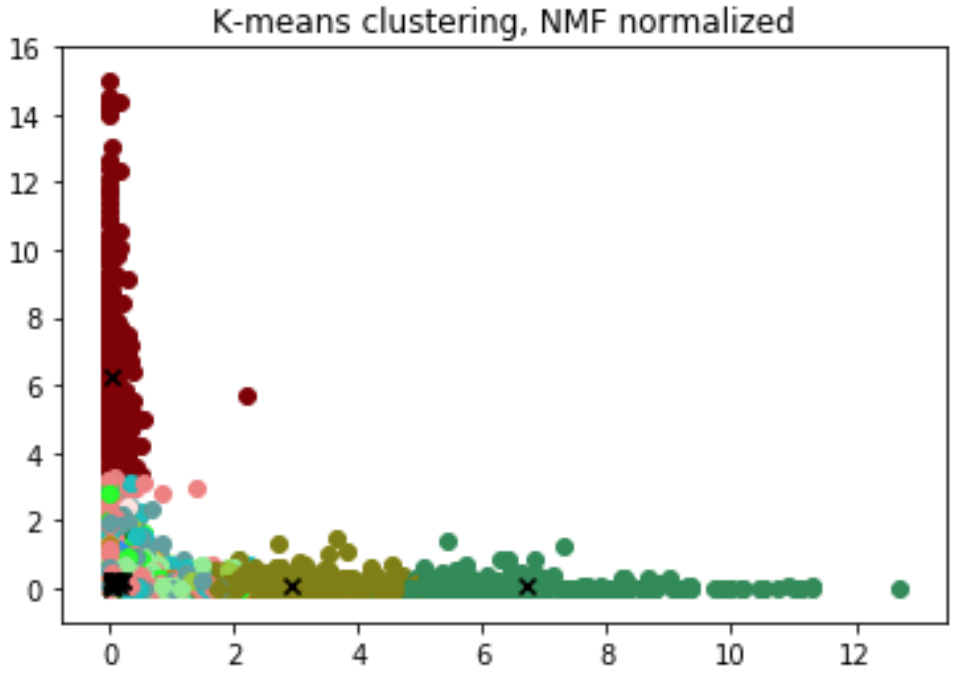
Without transformation:



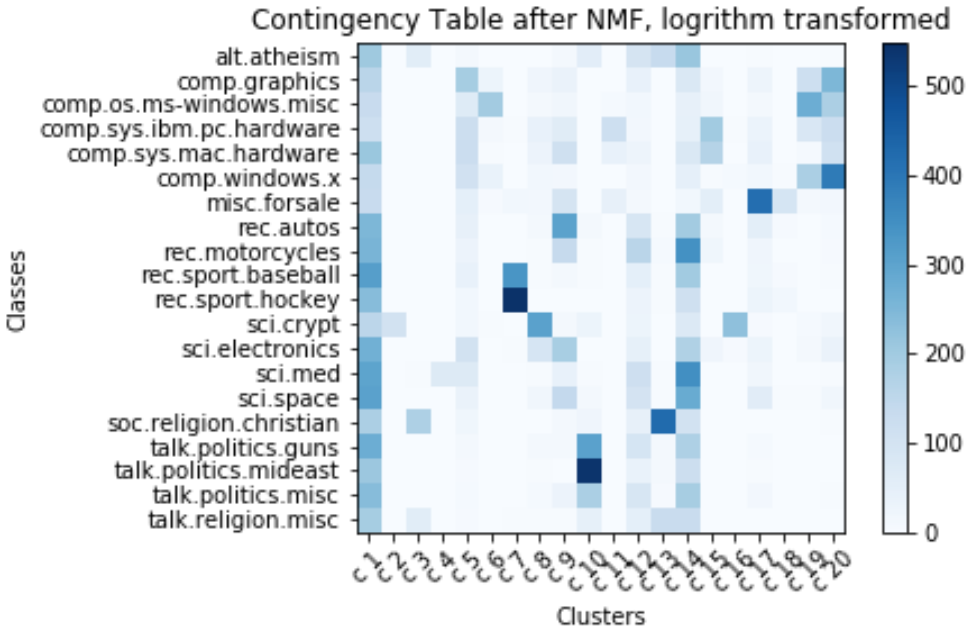


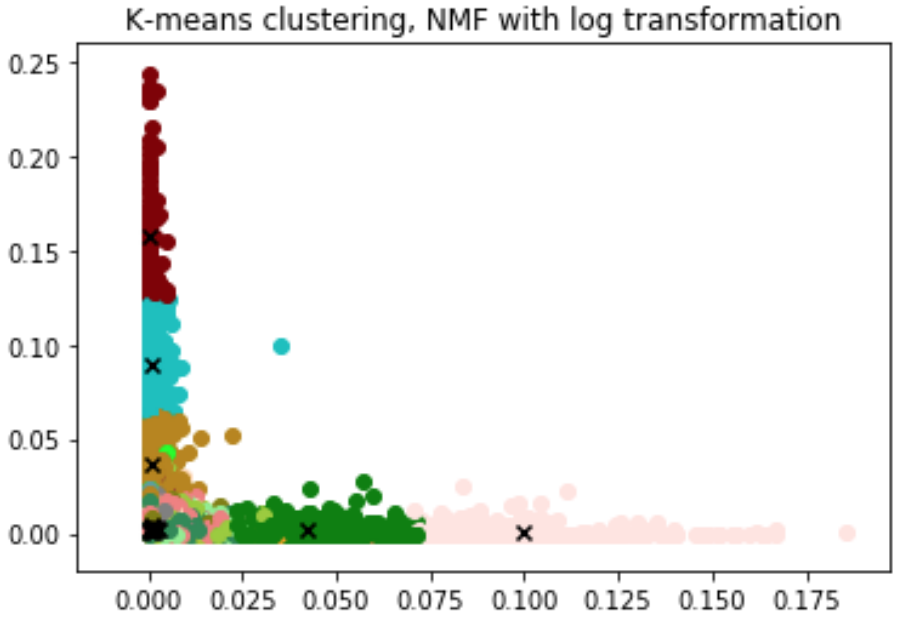
With normalization:



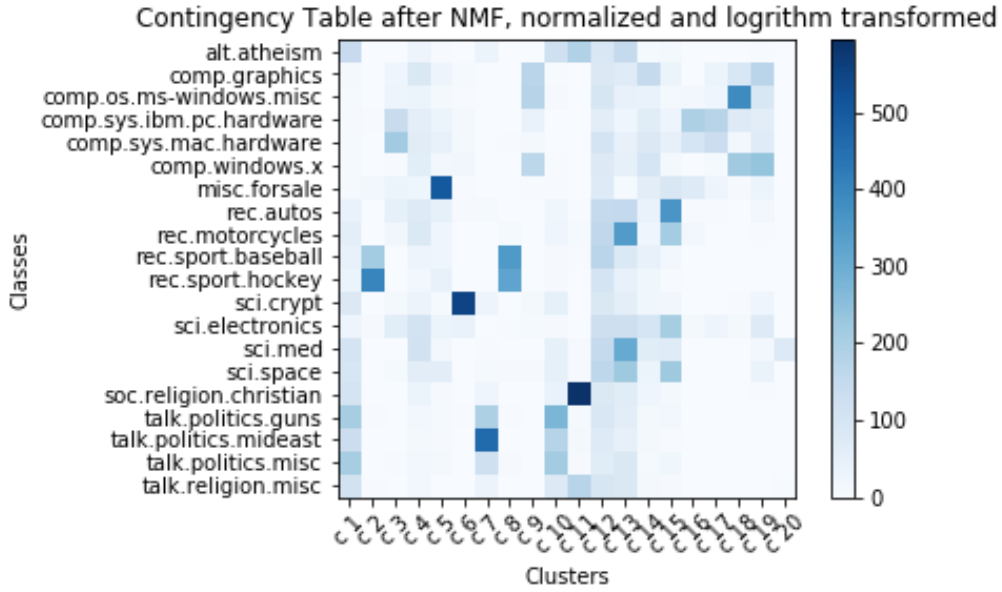


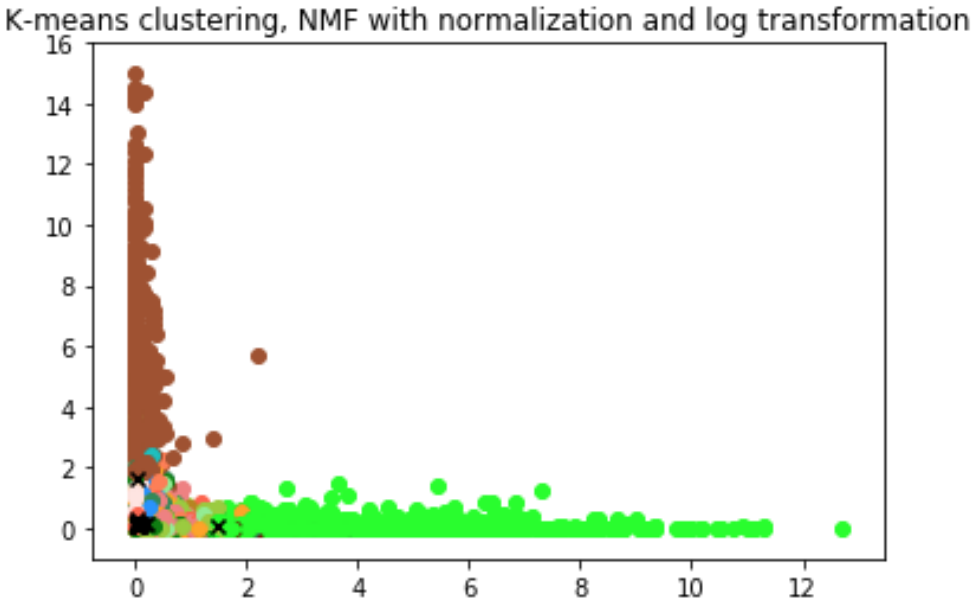
With log transformation:





With 1) normalization, and 2) log transformation:





With 1) log transformation and 2) normalization