EE 219 WINTER 2017
Project 4
Clustering

TEAM MEMBERS

Kaiyi Wu 004761345 kaiyiwu@ucla.edu

Xueyin Yu 504741703 soniayu@ucla.edu

Ruoxi Zhang 404753334 zhangruoxi@ucla.edu

1. Introduction

In previous project, we have already examined methods to solve classification problems. In this project, we are going to explore ways to deal with clustering problems.

Clustering is also a category of common problems when we talk about data analysis, Clustering algorithms are unsupervised methods for finding groups of data point that have similar representations in a proper space. The difference between clustering and classification is that in clustering we don't have priori knowledge about the label of data points even don't know how many categories these data points should be grouped into.

In this project, we are still going to use "20 Newsgroup" dataset. Our goals include finding a proper representation of the data, evaluating the performance of clustering algorithms.

2. Preprocessing of Dataset

As we mentioned above, the dataset we use in this project is "20 Newsgroup" dataset. It is a collection of 20000 documents, which are partitioned evenly into 20 different groups corresponding to different topics.

First, we start from a simple clustering problems, cluster the data points into 2 categories, the data points we use is showed in below figure, which includes 2 major classes "Computer Technology" and "Recreational Activities". In each major class, there are also 4 subclasses.

Class 1: Computer technology	Class 2: Recreational activity
comp.graphics comp.os.ms-windows.misc comp.sys.ibm.pc.hardware comp.sys.mac.hardware	rec.autos rec.motorcycles rec.sport.baseball rec.sport.hockey

Figure 2-1 Category of dataset

Even though we already know the classification of these data points, but in the process of clustering, we should pretend that we don't have the priori knowledge and to see how accurately our algorithms can assign these data points back to the class where they originally belong to.

The texture data cannot be used for clustering directly. Thus, before we start the clustering, we need to do some preprocessing work on the datasets, which is basically the same process we did in project 2. Here, we are not going to talk about these techniques in details but just go over the procedure.

First, we load the dataset direct from the Sklearn package and extract the classes we needed. What's need to mention is that the dataset in this project is the training dataset. Then, to increase the accuracy and avoid the noise, we excluded the influence caused by stop words and words with the same stems through TFxIDF technique. Now we get a vectorized sparse matrix which can be used for clustering problems. Below figure shows some basic information about the dataset and matrix.

```
Size of data:
4732
Size of target:
4732
```

```
Number of features before excluding stopwords and stemmers
#samples: 4732, #features: 79218

Number of features after excluding stopwords and stemmers
#samples: 4732, #features: 70065
```

Figure 2-2 Basic information about the dataset and matrix

From the figure, we can see that the dataset we selected has 4732 document samples. Before we exclude the stop words and words with same stems, the dataset has 79218 features, after that process this number decreases to 70065, which means that about 11.5% of feature has been filtered. Thus, the final matrix is of size 4732 rows and 70065 columns.

3. K-means Clustering Analysis

In this part, we will use K-Means algorithm to cluster the data points. K-Means clustering is a method of vector quantization, originally from signal processing, that is popular for cluster analysis in data mining. K-Means clustering aims to partition n observations into k clusters in which each observation belongs to the cluster with the nearest mean, serving as a prototype of the cluster.

First, we start from a simple condition where we are only going to cluster the data points into 2 groups – "Computer Technology" and "Recreational Activities". To achieve this, we need to select the label in the matrix and find which group they belong to, then we denote "Computer Technology" as 0 and "Recreational Activities" as 1 and append them into a new array. This new array will be used to examine the accuracy and purity in the following process.

After importing K-Means section into our script, we can start clustering using code below:

```
y_pred = KMeans(n_clusters=2).fit_predict(X_train)
y_true = target_train
```

Figure 3-1 Code for K-Means clustering

In the code, the *y_pred* is the predicted labels after K-Means clustering, *y_true* is the ground truth label we already know from the original data.

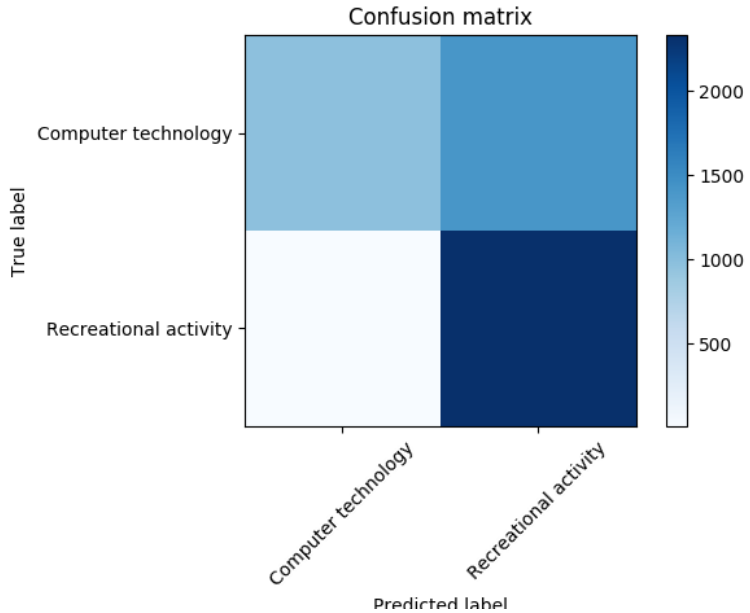
In order to examine the effect of K-Means, we analyzed the confusion matrix and 4 different scores (homogeneity score, completeness score, adjusted rand score and the adjusted mutual info score). First let's look at the confusion matrix between ground truth and predicted label. Confusion matrix is a direct approach to see the accuracy and the distribution among different categories. The confusion matrix is generated through the below code:

```
def plot_confusion_matrix(cm, title='Confusion matrix', cmap=plt.cm.Blues):
    plt.imshow(cm, interpolation='nearest', cmap=cmap)
    plt.title(title)
    plt.colorbar()
    tick_marks = np.arange(2)
    plt.xticks(tick_marks, ['Computer technology', 'Recreational activity'], rotation=45)
    plt.yticks(tick_marks, ['Computer technology', 'Recreational activity'])
    plt.tight_layout()
    plt.ylabel('True label')
    plt.xlabel('Predicted label')

cm = confusion_matrix(y_true, y_pred);
np.set_printoptions(precision=2)
print("")
print('Confusion matrix, without normalization')
print(cm)
plt.figure()
plot_confusion_matrix(cm)
plt.show()
```

Figure 3-2 Code for generating confusion matrix

The confusion matrix we generated and the statistical data are listed below:

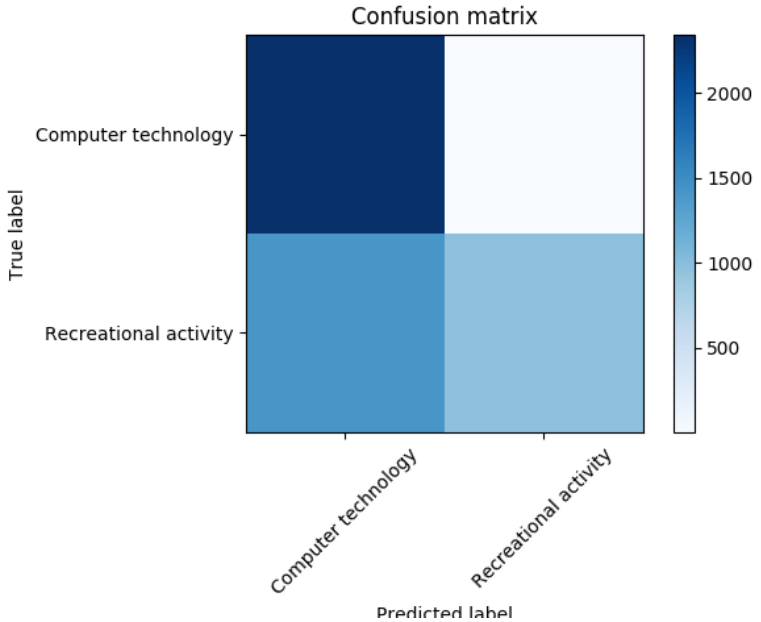



```
Confusion matrix, without normalization
[[ 926 1463]
 [   2 2341]]
```

Figure 3-3 Confusion matrix

From the matrix, we found that the cluster of recreational activities is almost accurate but this clustering mode can easily classify some computer technology document into recreational activity. Only less than 40% of computer technology documents can be correctly clustered into correct category.

Then we run our code again and get another confusion matrix, surprisingly, in this confusion matrix, the accuracy of computer technology and recreational activity are flipped. But after observation, we found even though the areas are flipped, but the pattern remains the same, which means that we can arrange the rows and get the similar pattern as before. Considering K-Means method is unsupervised, thus in the clustering process, it may not assign label as we did during the creation of ground truth method, so I think we cannot be sure whether this model has higher accuracy on computer technology or on recreational activities, what we can be sure is that the result has the same pattern and the same result distribution regardless of clustering labels.




```
Confusion matrix, without normalization
[[2341    2]
 [1416  973]]
```

Figure 3-4 Generate Confusion Matrix for the 2nd time

After we examined the confusion matrix, what's next is to calculate the 4 above mentioned score to determine how good or bad this clustering result is. These 4 scores all follow the same pattern, which is the higher the value is, the better the clustering model is. The value of scores we calculated are listed below:

```
The homogeneity score of KMeans cluster is:
0.217954830694

The completeness score of KMeans cluster is:
0.308386096177

The adjusted rand score of KMeans cluster is:
0.139452817528

The adjusted mutual info score of KMeans cluster is:
0.217835520637
```

Figure 3-5 the value of the 4 different scores

The first one is homogeneity score. A clustering result satisfies homogeneity if all of its clusters contain only data points which are members of a single class. This metric is independent of the absolute values of the labels: a permutation of the class or cluster label values won't change the score value in any way. The homogeneity score is in the range from 0 to 1. The higher this score is, the better the clustering result is. However, in our result, the homogeneity is only around 0.21795, and it's not a good result, which makes sense because from the confusion matrix we can see that we had a bad clustering result in one of the categories.

The second one is completeness score, a clustering result satisfies completeness if all the data points that are members of a given class are elements of the same cluster. This metric is independent of the absolute values of the labels: a permutation of the class or cluster label values won't change the

score value in any way. From the absolute value of this score, it's also not good enough for the same reason.

The third one is adjusted rand score, which computes a similarity measure between two clustering by considering all pairs of samples and counting pairs that are assigned in the same or different clusters in the predicted and true clustering. It's still quite low, only around 0.1394.

The last score is adjusted info score. Adjusted Mutual Information (AMI) is an adjustment of the Mutual Information (MI) score to account for chance. It accounts for the fact that the MI is generally higher for two clustering with a larger number of clusters, regardless of whether there is actually more information shared. In our calculation, the value is around 0.2178. Considering there are only 2 clustering in our model, it's reasonable this value is low.

The major reason of unsatisfying result of our clustering mode is that we carried out the clustering without doing the feature selection, which means that the vectors are still in a very high-dimensional space. Considering that clustering is a distance-based model, high dimensional space will greatly diminish the outcome of such model.

4. Better Representation for Data

4.1 Dimension Reduction

In this part, we will use Latent Semantic Indexing (LSI) and Non-negative Matrix Factorization(NMF) for dimensionality reduction.

In order to get a good initial guess for an appropriate dimensionality to feed in the K-means algorithm, I try to find the effective dimension of the data through inspection of the top singular values of the TF-IDF matrix. After calculation, I found that around 50 of them are significant in reconstructing the matrix with the truncated SVD representation.

To better validate the effectiveness of dimension reduction, I use LSI and NMF respectively by sweeping over the dimension parameter. Then I compare their corresponding results in terms of clustering purity metrics. The four scores are listed below:

Table 4-1 Dimensionality and Corresponding Clustering Scores after LSI

	2	3	20	50	100	150	200
Homogeneity	0.241	0.221	0.223	0.223	0.221	0.225	0.219
Completeness	0.324	0.310	0.312	0.313	0.311	0.314	0.309
Adjusted rand	0.168	0.142	0.145	0.146	0.143	0.147	0.141
Adjusted mutual info	0.241	0.220	0.223	0.223	0.221	0.225	0.219

Table 4-2 Dimensionality and Corresponding Clustering Scores after NMF

	2	3	20	50	100	150	200
Homogeneity	0.232	0.217	0.210	0.112	0.021	0.007	0.002
Completeness	0.319	0.308	0.302	0.219	0.098	0.116	0.096
Adjusted rand	0.156	0.138	0.131	0.145	0.009	0.006	0.006
Adjusted mutual info	0.232	0.217	0.210	0.112	0.019	0.007	0.002

From the tables above, we can easily tell that, while using LSI, we can get the best clustering performance when dimension parameter is 50, which validates our initial guessing for the number of significant top singular values. However, by using NMF, we had better choose small dimension parameter.

4.2 Data Representation

We can easily find that the clustering scores shown in 3.1 are far from satisfactory. Therefore, we first try to find a better representation of the data by normalizing features. To better show the results, here I use LSI as dimension reduction method and I set the dimension parameter as 50.

The purity metrics are greatly improved and are listed as below:

Homogeneity = 0.784
Completeness = 0.785
Adjusted rand = 0.867
Adjusted mutual info = 0.784

Thus, we can draw the conclusion that by normalizing features, we can have a better representation of the data and improve the clustering performance.

What's more, we want to apply some non-linear transformation on the data vectors after reducing dimensionality. Still, I use LSI as dimension reduction method and I set the dimension parameter as 50.

Firstly, we try to translate the matrix and take the square root of the matrix to see if it may be a good representation for the data. The process is shown below:

```
#Transit the matrix
min = X_svd2.min()
difference = 0 - min
for i in range(num_samples_svd):
    for j in range(num_features_svd):
        X_svd2[i][j] = difference + X_svd2[i][j]
#Take square root
X_sqrt = np.sqrt(X_svd2)
```

Figure 4-1 Non-linear Transformation: Square Root

The results are listed as follow:

Homogeneity = 0.238
Completeness = 0.323
Adjusted rand = 0.161
Adjusted mutual info = 0.237

Since the results improve only a little from previous ones, we can tell that the square root may not be a good candidate for the data.

Then, we try to translate the matrix and take the logarithm of the matrix. The process is shown below:

```
X_svd3 = svd.fit_transform(X_train)
num_samples_svd, num_features_svd = X_svd3.shape
#Transit the matrix
min = X_svd3.min()
difference = 0.005 - min
for i in range(num_samples_svd):
    for j in range(num_features_svd):
        X_svd3[i][j] = difference + X_svd3[i][j]
#Take logarithm
X_log2 = np.log(X_svd3)
```

Figure 4-2 Non-linear Transformation: Logarithm

Here, when we have the smallest element in the matrix be translated to 0.005, we can get the best performance of the non-linear transformation. The results are listed as follow:

Homogeneity = 0.784

Completeness = 0.784

Adjusted rand = 0.866

Adjusted mutual info = 0.784

Therefore, we can draw the conclusion that compared to square root, logarithm is a better candidate of non-linear transformation for the data.

Here, we can find out that when use LSI as dimension reduction method and 50 as the dimension parameter, both normalization and logarithm can have the best final representation of the data.

4.3 Plot of the Result

To get a visual sense on the NMF embedding of the data, we try applying NMF to the data matrix with ambient parameter 2 and plot the resulting points. Also, we plot the actual labels to have a comparison with the predicting ones, which help us choose the appropriate non-linear transformation. The figure is shown below:

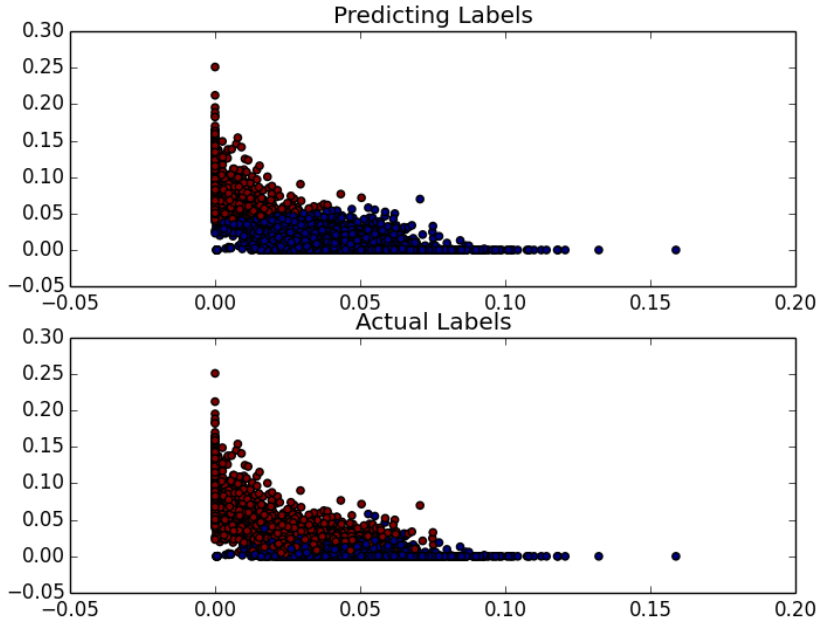


Figure 4-3 Predicting Labels vs. Actual Labels

From the figure above, we can find that all points (data) are likely to be distributed around 0, which tend to be in a very small area and hard to be separated from each other. Also, there may be some outliers which have much larger values than others. To solve these problems, we can take the logarithm of the data, so that for points among 0-1, they can be scaled to a larger range, which can be more separable and thus, easier for clustering. So logarithm is a good candidate for my TF*IDF data.

5. Visualization of the Clustering

In this part, I try to visualize the performance of my clustering by projecting final data vectors onto 2 dimensions and color-coding the classes. Here we try NMF and LSI respectively and then taking the logarithm.

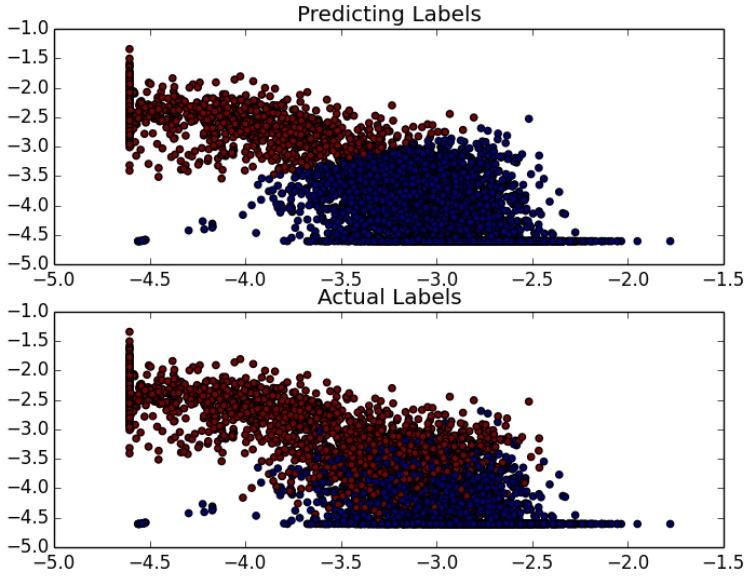


Figure 5-1 Predicting Labels vs. Actual Labels (NMF + Logarithm)

The figure above is the result of clustering with NMF and Logarithm. Compared with figure 3-3, we can find that the performance is greatly improved by adding linear transformation.

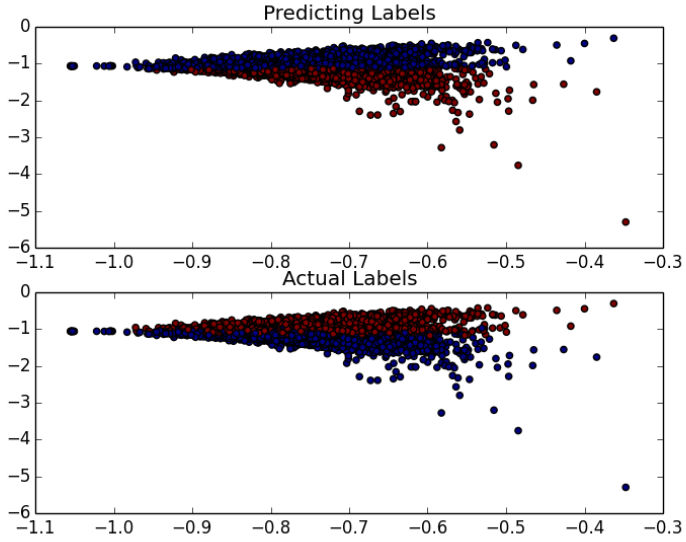


Figure 5-2 Predicting Labels vs. Actual Labels (LSI + Logarithm)

Also, when we combine LSI with Logarithm, the results are better than the ones (reported in 4.1) merely using LSI.

Therefore, non-linear transformation is quite useful for clustering.

6. Clustering Analysis with 20 Categories

In this part, we load the whole dataset of 20 categories and we want to examine how purely we can retrieve all the original 20 sub-class labels with clustering. First, by setting k=20, just the same as class number, we use truncated SVD and NMF methods to do the dimension reduction, and also, we use normalization and logarithm techniques to find better representations.

For each dimension reduction method, meaning, truncated SVD and NMF, we combine

normalization and logarithm methods, to form four combination techniques to find good representation, with trying different ambient space dimensions. Adding original truncated SVD and NMF methods, we have six tables to show how we find the best representation of each technique. Since here, we set number of clusters = k = 20 = number of classes, so we can just focus on homogeneity score which evaluates the performance. We have emphasized the best representation line in each table.

Table 6-1 Different ambient space dimension for truncated SVD (class=20, k=20)

Dimension	Homogeneity Score	Completeness Score	Adjusted Rand Score	Adjusted Mutual Info Score
2	0.228	0.247	0.074	0.225
5	0.329	0.360	0.130	0.327
10	0.352	0.399	0.141	0.350
20	0.307	0.411	0.093	0.305
30	0.301	0.434	0.091	0.298
40	0.320	0.394	0.108	0.318
50	0.313	0.406	0.104	0.310
100	0.300	0.410	0.087	0.297

Table 6-2 Different ambient space dimension for truncated SVD with normalization (class=20, k=20)

Dimension	Homogeneity Score	Completeness Score	Adjusted Rand Score	Adjusted Mutual Info Score
2	0.243	0.250	0.074	0.241
5	0.381	0.389	0.196	0.379
10	0.429	0.432	0.269	0.427
20	0.405	0.423	0.262	0.403
30	0.388	0.396	0.241	0.386
40	0.368	0.381	0.231	0.366
50	0.387	0.408	0.224	0.385
100	0.370	0.408	0.188	0.368

Table 6-3 Different ambient space dimension for truncated SVD with logarithm (class=20, k=20)

Dimension	Homogeneity Score	Completeness Score	Adjusted Rand Score	Adjusted Mutual Info Score
2	0.232	0.253	0.077	0.229
5	0.330	0.376	0.125	0.328
10	0.356	0.416	0.154	0.354
20	0.326	0.400	0.128	0.324
30	0.250	0.362	0.078	0.248
40	0.288	0.372	0.080	0.286

50	0.271	0.369	0.078	0.269
100	0.310	0.401	0.106	0.307

Table 6-4 Different ambient space dimension for NMF (class=20, k=20)

Dimension	Homogeneity Score	Completeness Score	Adjusted Rand Score	Adjusted Mutual Info Score
2	0.197	0.214	0.057	0.195
5	0.282	0.312	0.104	0.279
10	0.336	0.401	0.131	0.333
20	0.284	0.387	0.089	0.281
30	0.306	0.409	0.092	0.304
40	0.191	0.273	0.044	0.189
50	0.167	0.287	0.025	0.164
100	0.153	0.228	0.027	0.150

Table 6-5 Different ambient space dimension for NMF with normalization (class=20, k=20)

Dimension	Homogeneity Score	Completeness Score	Adjusted Rand Score	Adjusted Mutual Info Score
2	0.202	0.234	0.086	0.199
5	0.319	0.335	0.160	0.317
10	0.386	0.404	0.247	0.384
20	0.333	0.365	0.195	0.331
30	0.385	0.409	0.216	0.383
40	0.337	0.353	0.185	0.335
50	0.311	0.324	0.169	0.309
100	0.234	0.279	0.077	0.232

Table 6-6 Different ambient space dimension for NMF with logarithm (class=20, k=20)

Dimension	Homogeneity Score	Completeness Score	Adjusted Rand Score	Adjusted Mutual Info Score
2	0.209	0.219	0.066	0.207
5	0.318	0.330	0.141	0.316
10	0.382	0.397	0.210	0.380
20	0.343	0.374	0.199	0.341
30	0.402	0.439	0.215	0.400
40	0.360	0.385	0.195	0.358
50	0.336	0.364	0.165	0.334
100	0.304	0.331	0.135	0.302

From the data of truncated SVD and NMF dimension reduction methods, we can know that truncated SVD with normalization at dimension equals 10 is a good combination, with homogeneity score equals to 0.429, and NMF with logarithm at dimension equals 30 is another good combination, with homogeneity score equals to 0.402. In the next step, we take these two combinations and try to find appropriate parameter k in K-means clustering.

In table 6-7 and table 6-8, we list the four scores of different k value in certain representation. Considering the definitions of four scores, we know that when k < (number of classes), we should focus on completeness score and when k > (number of classes), we should focus on homogeneity score. The limitation situation is that when k =1, completeness score = 1, and when k = number of data, homogeneity score = 1. However, we cannot find the best k value only per homogeneity score and completeness score. So here, we pay attention to adjusted rand score, which is just like accuracy measure computing similarity between the clustering labels and ground truth labels.

Table 6-7 Different k values in k-means
(Truncated SVD with normalization at dimension=10)

k	Homogeneity Score	Completeness Score	Adjusted Rand Score	Adjusted Mutual Info Score
2	0.128	0.578	0.058	0.127
5	0.244	0.465	0.140	0.243
10	0.360	0.483	0.229	0.359
15	0.400	0.455	0.259	0.398
20	0.432	0.436	0.283	0.430
25	0.455	0.430	0.273	0.427
30	0.466	0.415	0.255	0.413
50	0.490	0.381	0.201	0.377
100	0.521	0.344	0.131	0.337

Table 6-8 Different k values in k-means
(NMF with logarithm at dimension=30)

k	Homogeneity Score	Completeness Score	Adjusted Rand Score	Adjusted Mutual Info Score
2	0.081	0.745	0.018	0.081
5	0.274	0.600	0.124	0.273
10	0.352	0.500	0.182	0.351
15	0.383	0.459	0.226	0.382
20	0.413	0.455	0.209	0.411
25	0.435	0.438	0.230	0.433
30	0.435	0.406	0.214	0.403
50	0.465	0.376	0.185	0.372
100	0.514	0.345	0.126	0.337

From table 6-7 and table 6-8, we find best k value of these two techniques. We have emphasized the

line with best adjusted rand score, k equals to 20 and 25, respectively, for truncated SVD and NMF.

7. Evaluate the Performance of Clustering in the Top-wise Classes

In this part, our goal is to evaluate the performance in retrieving the topic-wise classes. To do that, we try to find a proper representation through dimensionality reduction and feature transformation. Also, six tables corresponding to six technique combination are shown below. Here, the number of classes is equal to number of clusters, which is 6, so we can focus on homogeneity score to evaluate the performance of each technique.

Table 7-1 Different ambient space dimension for truncated SVD (class=6, k=6)

Dimension	Homogeneity Score	Completeness Score	Adjusted Rand Score	Adjusted Mutual Info Score
2	0.262	0.276	0.167	0.262
5	0.333	0.395	0.179	0.333
10	0.292	0.448	0.146	0.292
20	0.257	0.417	0.149	0.257
30	0.323	0.401	0.136	0.323
40	0.186	0.388	0.142	0.185
50	0.309	0.453	0.251	0.309
100	0.234	0.374	0.153	0.233

Table 7-2 Different ambient space dimension for truncated SVD with normalization (class=6, k=6)

Dimension	Homogeneity Score	Completeness Score	Adjusted Rand Score	Adjusted Mutual Info Score
2	0.354	0.346	0.238	0.345
5	0.408	0.398	0.344	0.398
10	0.378	0.369	0.281	0.368
20	0.373	0.390	0.314	0.373
30	0.366	0.383	0.234	0.366
40	0.314	0.310	0.195	0.309
50	0.362	0.363	0.252	0.361
100	0.281	0.279	0.244	0.279

Table 7-3 Different ambient space dimension for truncated SVD with logarithm (class=6, k=6)

Dimension	Homogeneity Score	Completeness Score	Adjusted Rand Score	Adjusted Mutual Info Score
2	0.252	0.275	0.145	0.252
5	0.290	0.335	0.195	0.290
10	0.289	0.357	0.154	0.289
20	0.296	0.353	0.245	0.296

30	0.287	0.450	0.178	0.286
40	0.264	0.372	0.142	0.264
50	0.288	0.457	0.200	0.288
100	0.248	0.416	0.208	0.248

Table 7-4 Different ambient space dimension for NMF (class=6, k=6)

Dimension	Homogeneity Score	Completeness Score	Adjusted Rand Score	Adjusted Mutual Info Score
2	0.197	0.214	0.057	0.195
5	0.273	0.300	0.103	0.271
10	0.331	0.382	0.141	0.329
20	0.280	0.376	0.088	0.274
30	0.319	0.424	0.084	0.317
40	0.210	0.319	0.043	0.207
50	0.184	0.281	0.037	0.181
100	0.151	0.273	0.021	0.149

Table 7-5 Different ambient space dimension for NMF with normalization (class=6, k=6)

Dimension	Homogeneity Score	Completeness Score	Adjusted Rand Score	Adjusted Mutual Info Score
2	0.300	0.327	0.244	0.300
5	0.323	0.338	0.262	0.323
10	0.347	0.351	0.317	0.347
20	0.267	0.265	0.189	0.265
30	0.316	0.342	0.184	0.316
40	0.142	0.159	0.055	0.142
50	0.131	0.160	0.030	0.131
100	0.121	0.144	0.035	0.121

Table 7-6 Different ambient space dimension for NMF with logarithm (class=6, k=6)

Dimension	Homogeneity Score	Completeness Score	Adjusted Rand Score	Adjusted Mutual Info Score
2	0.275	0.267	0.174	0.267
5	0.307	0.318	0.271	0.307
10	0.429	0.466	0.399	0.428
20	0.308	0.356	0.235	0.307
30	0.347	0.354	0.212	0.347
40	0.216	0.234	0.100	0.216

50	0.126	0.159	0.049	0.125
100	0.084	0.106	0.024	0.084

From table 7-1 to table 7-6, we can get same result as part 6 when class number is 20, which means, the truncated SVD and normalization is a good combination and NMF and logarithm is another good combination for proper representation. For truncated SVD and normalization pair, the proper dimension is around 5, and for NMF and logarithm pair, the proper dimension is around 10. We then try more different space dimensions around them with smaller interval. Results are shown in table 7-7 and table 7-8.

Table 7-7 Different ambient space dimension for truncated SVD with normalization (class=6, k=6)

Dimension	Homogeneity Score	Completeness Score	Adjusted Rand Score	Adjusted Mutual Info Score
2	0.354	0.344	0.235	0.344
3	0.399	0.393	0.308	0.393
4	0.437	0.434	0.380	0.434
5	0.408	0.398	0.344	0.398
6	0.394	0.398	0.359	0.394
7	0.378	0.387	0.340	0.378
8	0.391	0.382	0.339	0.382
9	0.399	0.411	0.355	0.398
10	0.327	0.325	0.242	0.324

Table 7-8 Different ambient space dimension for NMF with logarithm (class=6, k=6)

Dimension	Homogeneity Score	Completeness Score	Adjusted Rand Score	Adjusted Mutual Info Score
5	0.325	0.328	0.231	0.325
6	0.302	0.333	0.260	0.302
7	0.305	0.329	0.247	0.305
8	0.364	0.379	0.320	0.364
9	0.415	0.452	0.394	0.414
10	0.403	0.426	0.337	0.403
11	0.416	0.461	0.349	0.415
12	0.389	0.383	0.324	0.382
13	0.353	0.375	0.264	0.353
14	0.345	0.363	0.271	0.344
15	0.339	0.357	0.267	0.338

From table 7-7 which is corresponding to truncated SVD with normalization, we can find that when dimension equals 4, the score is the highest, therefore, the performance of the clustering is pretty good in retrieving the 6 classes. From table 7-8, which is corresponding to NMF with logarithm, we

can see that when dimension equals 9 or 11, the performance is good, by higher score.
In conclusion, for better retrieving the top-wise classes, our good representation combination is as table 7-9.

Table 7-9 Technique combination of good representation

Truncated SVD & Normalization	Dimension	4
	Cluster number	6
	Final homogeneity score	0.437
NMF & Logarithm	Dimension	9
	Cluster number	6
	Final homogeneity score	0.415
NMF & Logarithm	Dimension	11
	Cluster number	6
	Final homogeneity score	0.416

8. Conclusion

In this project, there're three main processes. Each step takes its own role and every difference in each step will bring changes to results.
The first one is "preprocessing of data" is to preprocess the raw data, including deal with the stop words, and vectorization based on project 2. Our team have done simple preprocess on the dataset, which brings better performance in the final scores. However, theoretically, if we keep doing more detailed preprocess on the data, we can gain better results. Actually, it's a good point we can focus on in the future work.
The second step is "dimension reduction and feature transformation". In this project, we follow the spec and choose truncated SVD and NMF as dimension reduction techniques and try normalization and logarithm as feature transformation techniques. We find in certain case, certain combination of dimension reduction method and feature transformation method can bring good results. For example, in retrieving the original 20 sub-class part, truncated SVD performs well with normalization, and NMF well with logarithm. In the general, by trying different technique and parameters, we find proper representation.
The final step is "k-means clustering". Based on the proper representation we have found, we do k-means clustering and then evaluate their performance, by seeing four scores. Also, we have tried different k values to make the results better. The thing need to pay attention to is, when k value changes, the evaluation score should be correspondingly changes. This is for well evaluating the k-means clustering.