Large Scale Data Mining: Models and Algorithms: Project #2

Due on February 5st, 2020 at 11:59pm

Professor Roychowdhury, Vwani

Wang, Yin, He

1. Introduction

According to recent lectures, we have learnt how to use cluster in Python. In this project, we are going to build a cluster to distinguish the target data like text or imagine. The first part is to cluster the text data; the second part is to cluster our own dataset; and the last part is to cluster the color of imagine. More detailed descriptions and discussions on each term will be presented in the following section throughout the report. In this project, the goal includes:
1. To find proper representations of the data, such that the clustering is efficient and gives out reasonable results.
2. To perform K-means clustering on the dataset, and evaluate the result of the clustering.
3. To try different preprocessing methods which may increase the performance of the clustering.

2. The Dataset

In this part, we still use the "20 Newsgroups" dataset have been used in the Project 1. This time, we need to divide the data into 2 groups according to their similarity. That means the original labels of classes, "Computer Technology" and "Recreational Activity" will be ignored. To evaluate our cluster algorithm, we will compare our groups with the ground truth. The purity of groups represents the accuracy of our algorithm.

3. Part I

Question 1

We are going to use the similar filtering method adapted in Project 1 without lemmatization to vectorize our documents. Moreover, we will remove the headers and footers of the documents to achieve the better clustering performance.
The TF-IDF score is defined as below:

$$tf - idf(d,t) = tf(t,d) \times idf(t) \tag{1.1}$$

where tf(t,d) represents the frequency of term t in document d, and the inverse document frequency is defined as below:

$$idf(t) = log(\frac{n}{df(t)}) + 1 \tag{1.2}$$

where n is the total number of documents and df(t) is the document frequency. Finally, we can get the size of TF-IDF matrix as 4732×15926.

Question 2

The parameters for the Kmeans method are set as Table 1. The contingency matrix for the Kmeans method based on the parameters set above is shown as 2.
This means 1904 entries of computer technology combined with 36 recreation documents are clustered together while another cluster contains 439 entries of computer technology and 2353 texts concerning recreation.

Parameters	Value
n_cluster	2
random state	0
max_iter	1000
n_init	30

Table 1: Parameter Setting for the Kmeans clustering

		Clustered	
		Cluster 0	Cluster 1
Actual	Technology	1904	439
	Recreation	36	2353

Table 2: The contingency matrix for the trained clustering model

Question 3

The metric above is not straight forward to estimate the performance of clustering. In the question 3, we are going to use five other measures to value the result, which are homogeneity score, completeness score, V-measure, adjusted Rand score and adjusted mutual info score.

Homogeneity score: if a cluster contains only one category of samples, then homogeneity is satisfied. In fact, it can also be considered as the correct rate (the proportion of the correctly classified samples in each cluster to the total number of samples in the cluster):

$$p = \frac{1}{k}\sum_{i=1}^{k}\frac{N(C_i == K_i)}{N(K_i)} \tag{3.1}$$

Completeness score: when all samples of the same category are classified into the same cluster, the integrity is satisfied; the sum of the proportion of correctly classified samples in each cluster to the total number of samples of this type:

$$r = \frac{1}{k}\sum_{i=1}^{k}\frac{N(C_i == K_i)}{N(C_i)} \tag{3.2}$$

V-measure, the weighted average of homogeneity and completeness:

$$V_\beta = \frac{(1+\beta^2)\times pr}{\beta^2 \times p + r} \tag{3.3}$$

Adjusted Rand Index (ARI) is derived from the conception, Rand index, which can be defined as

$$RI = \frac{a+b}{c_n^2} \tag{3.4}$$

where a represents the logarithms of elements of the same category, b represents the logarithms of elements of different categories, and c_2^n represents the data logarithms. ARI is a variant of RI trying to measure the consistency of the clustering result with the real situation:

$$ARI = \frac{RI - E[RI]}{max(RI) - E[RI]} \tag{3.5}$$

Adjusted Mutual Information (AMI), which is similar to ARI, uses the entropy information internally. Using the equation above, the 5 metrics for the K-means clustering results is obtained as Table 8.

Method	Homogeneity	Completeness	V measure	adjusted rand	adjusted mutual info
K-means	0.5750	0.5888	0.5818	0.6387	0.5749

Table 3: Clustering performance based on Kmeans without dimensional reduction.

Question 4

There are more than one reason that leads K means method does not performance well. For example, the limitation of Euclidean distance, the clusters are not isotopically shaped, and the clusters do have unequal variances. To solve these problem, we are going to use TF-IDF, which we have used in Project 1, to reduce the dimension of data.

The percent of variance retaining varying with the top r principle components is shown as Figure 1.

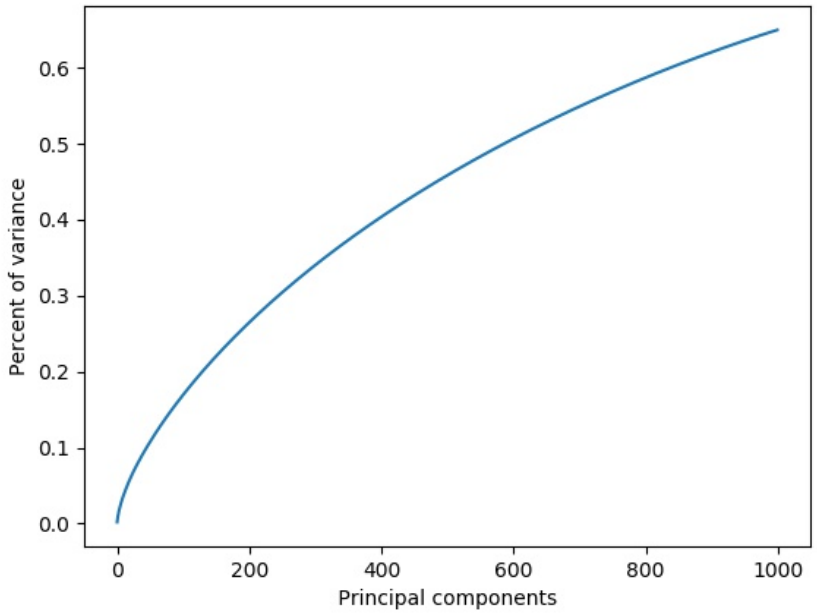

Figure 1: The trend of percent of variance with the top principle components included.

Question 5

The impacts of the choice of r on the clustering performance for the SVD and NMF method are illustrated as Figure 6 and Figure 7. The pictures can only show the trend that the 5 metrics values vary with the choice of r. For clarity, we decide to organize the metrics obtained as three part tables shown as Table 4 and Table 5. Considering both the performance of clustering and the reservation of the information, we decide to choose r as 50 for the SVD method while r equals to 10 for achieving the trade-off between information preservation and clustering performance using NMF method.

r	Homogeneity	Completeness	V measure	adjusted rand	adjusted mutual info
1	0.0202	0.0206	0.0204	0.0274	0.0200
2	0.5410	0.5548	0.5478	0.6080	0.5409
3	0.5319	0.5471	0.5394	0.5955	0.5318
5	0.5066	0.5289	0.5175	0.5501	0.5065
10	0.5643	0.5779	0.5710	0.6300	0.5642
20	0.5604	0.5744	0.5673	0.6253	0.5604
50	0.5867	0.5971	0.5918	0.6612	0.5866
100	0.5840	0.5960	0.5899	0.6530	0.5839
300	0.5777	0.5903	0.5839	0.6455	0.5776

Table 4: Clustering performance based on SVD method with different r.

r	Homogeneity	Completeness	V measure	adjusted rand	adjusted mutual info
1	0.0202	0.0206	0.0204	0.0274	0.0200
2	0.5084	0.5284	0.5182	0.5589	0.5083
3	0.5577	0.5655	0.5616	0.6434	0.5576
5	0.4758	0.5031	0.4891	0.5077	0.4758
10	0.0951	0.2160	0.1320	0.0360	0.0949
20	0.0598	0.1827	0.0901	0.0119	0.0597
50	0.0519	0.1746	0.0801	0.0088	0.0518
100	0.0308	0.1495	0.0511	0.0028	0.0306
300	0.0239	0.1393	0.0408	0.0015	0.0238

Table 5: Clustering performance based on NMF method with different r.

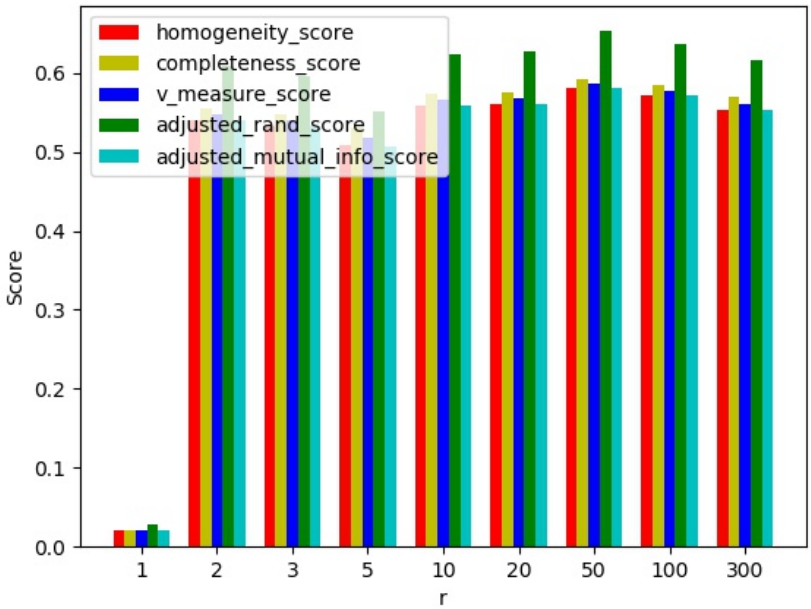

Figure 2: Metrics evaluating the performance of clustering based on SVD method for different r.

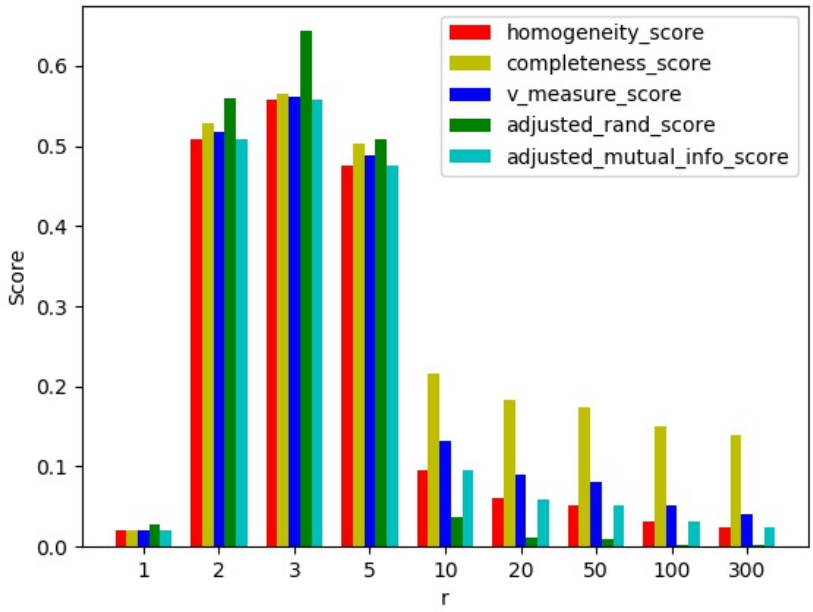

Figure 3: Metrics evaluating the performance of clustering based on NMF method for different r.

Question 6

At first, when r is small, there are too less information contains in the singular value matrix, so the performance of the cluster could not be well. As the r increase, the increased information helps the accuracy increased. Therefore, the performance of the five metrics increases as the number of dimensions increases. However, this increase is not monotonic. After reaching the apex, as the dimension increases, a large amount of useless data will pollute the Euclidean distance, causing the performance of the measurement to decline instead. The overall characteristics are non-linear.

Question 7

According to the Question 5, we are going to choose r for the SVD method as 50 while the best r for the NMF method is chosen as 10. To visualize the clustering result, we will build a new SVD model with $n\_components = 2$. The results of clustering based on two different dimensional reduction methods are shown as Figure 4, 5 while the ground truth is painted as a comparison as well. The colors of dots for clustering result and ground truth images may have different corresponding labels but it does not affect the evaluation of the clustering performance.

Question 8

There are 6 figures we are going to show here. Specifically, the figure 6 refers to the TruncatedSVD method without transformation while the figure 7 refers to the SVD method applying scaling transformation. The Figure 8 refers to the vanilla NMF method without any transformation. The figure 9 shows the ground truth and clustering result of the NMF method with scaling transformation. The figure 10 illustrates the corresponding result adapting the logarithm transformation while the figure 11 is the result applying all the

Question 8 continued on next page...

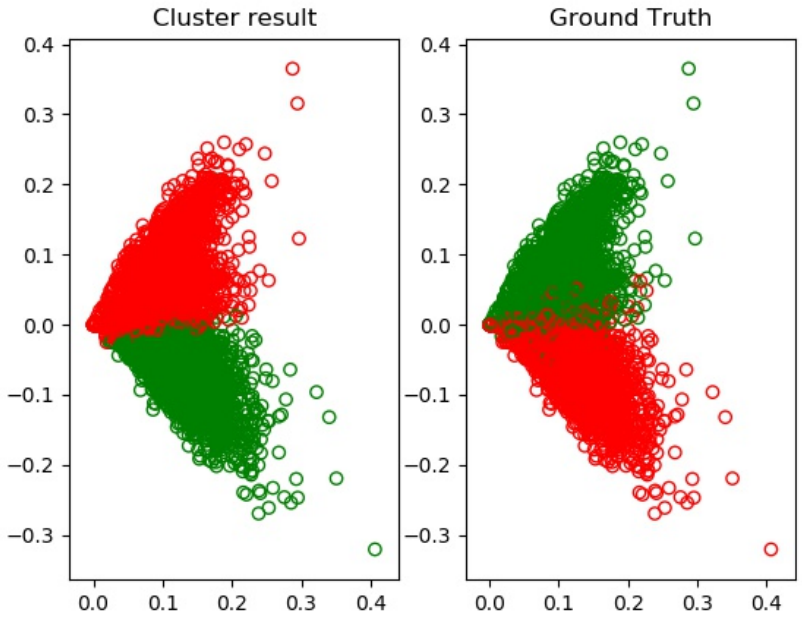

Figure 4: Visualization of the clustering result and ground truth for the SVD method

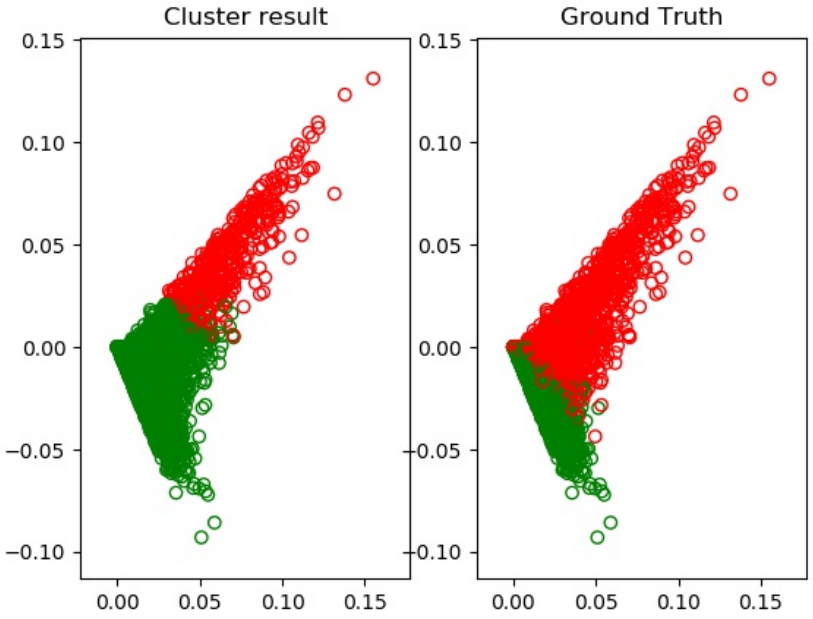

Figure 5: Visualization of the clustering result and ground truth for the NMF method

transformation mentioned between the stem. It is notable that if we do not applying any transformation to the SVD or NMF method, which refers to the figure 6 and figure 8, it is nominally the same as the result we have shown at Question 7.

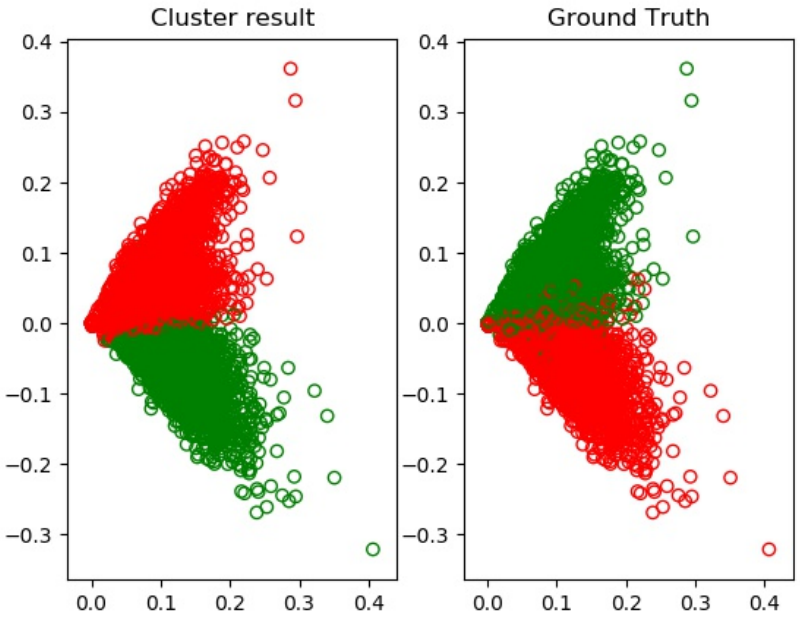

Figure 6: Visualization of the result and truth without any transformation for the SVD method

Question 9

If we inspect the six different figures we have demonstrated at Question 7, we can easily find that the 2-D projection of the data dimensionally reduced by the NMF method are much more compact than the SVD method: the span of x and y shown at Figure 5 are less than the span of x,y shown at Figure 4. Thus, a reasonable guess about the factor impacting the performance of clustering is the compactness of the data. If we can make entries of different classes distributes separately, then it is much easier for the clustering method to obtain the proper result.

For logarithm transformation, it is often used to create monotonic data transformations. Its main role is to help stabilize the variance. Logarithmic transformation tends to stretch the range of independent variable values that fall in the lower amplitude range, and compress or reduce the range of independent variable values in the higher amplitude range. This makes the tilt distribution as close to the normal distribution as possible. The values of the originally dense intervals are dispersed as much as possible, and the values of the originally dispersed intervals are aggregated as much as possible.

Question 10

The five metrics impacted by the transformation we apply to the input of the clustering model can refer to Table 6 and Table 7. There is an interesting phenomenon needed to be illustrate here. The metrics we get without any transformation is different with the results shown in Question 5 while we expect that they

Question 10 continued on next page...

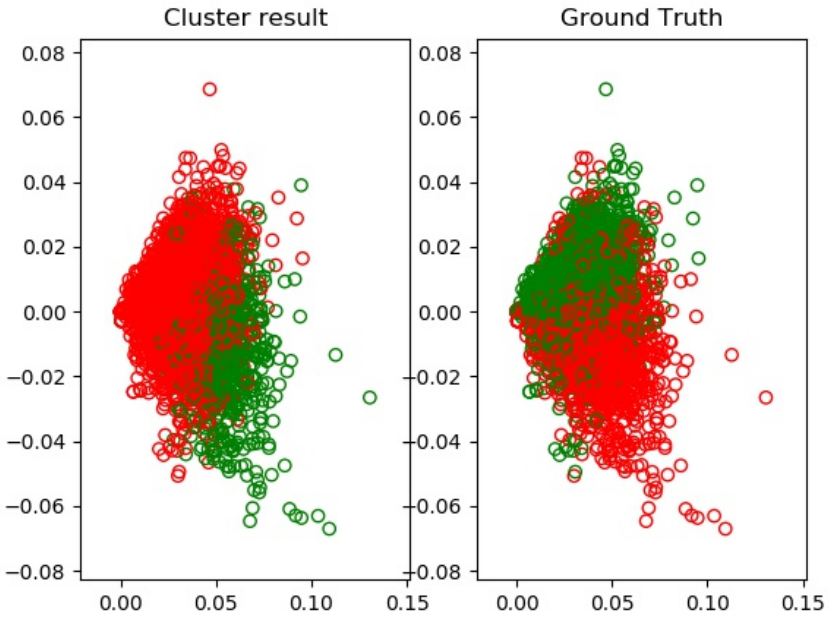

Figure 7: Visualization of the result and truth with scaling transformation for the SVD method

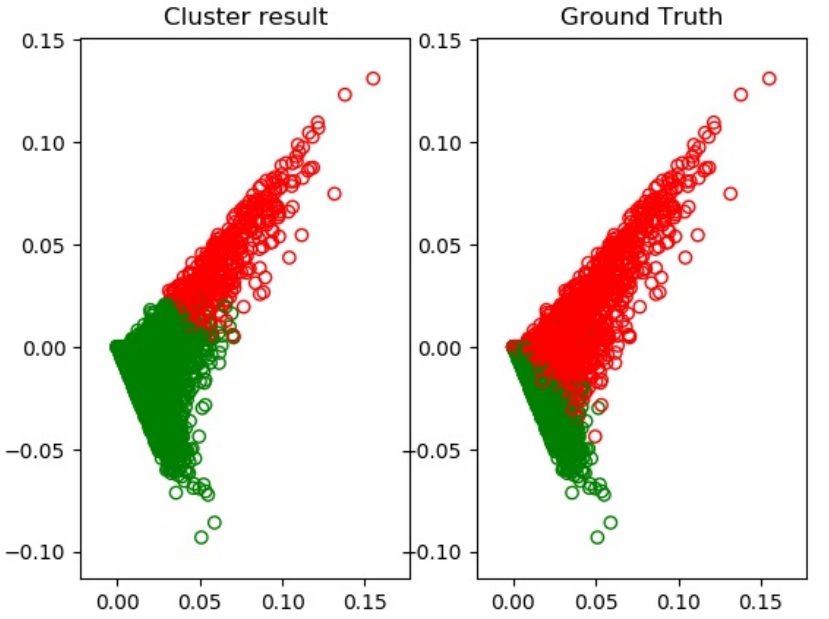

Figure 8: Visualization of the result and truth without any transformation for the NMF method

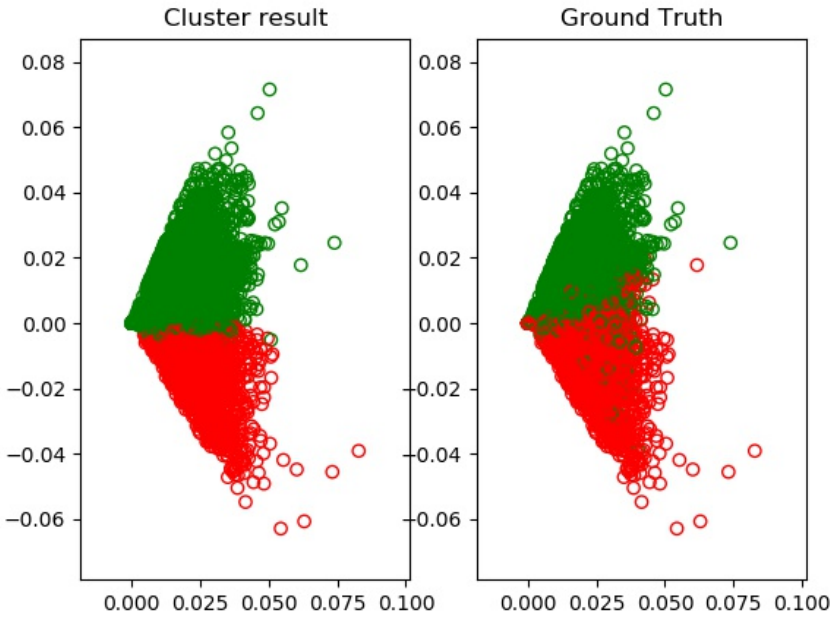

Figure 9: Visualization of the result and truth with Scaling transformation for the NMF method

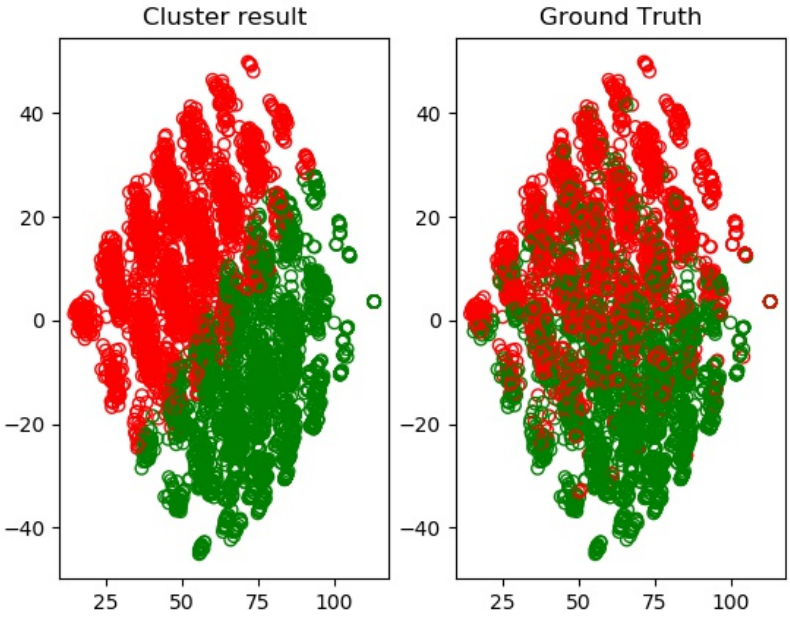

Figure 10: Visualization of the result and truth with logarithm transformation for the NMF method

Question 10 continued on next page...

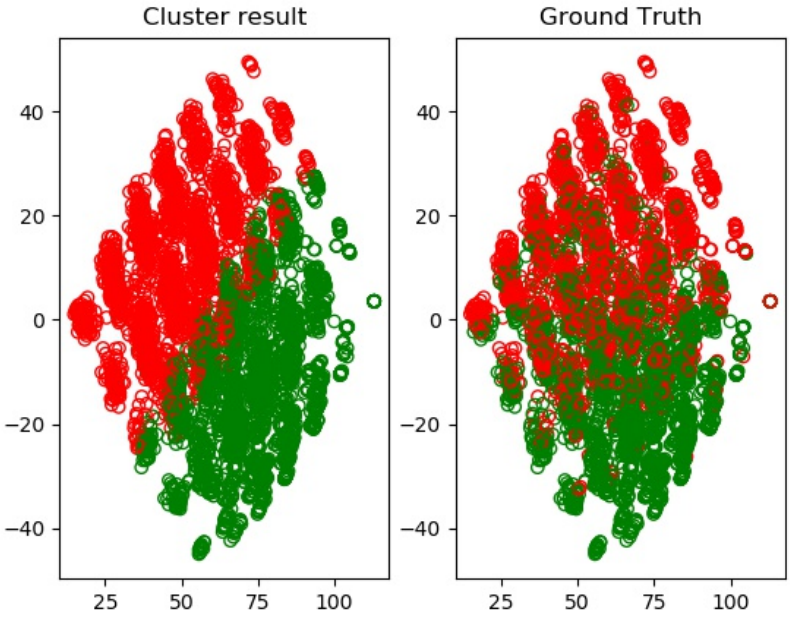

Figure 11: Visualization of the result and truth applying all transformation mentioned for the NMF method

should be the same. This may be caused by the randomness of clustering algorithm: we add a transformer before clustering which changes the random number generated for the model even though the transformer itself does not do anything. Regardless of this subtle difference, the result without any transformation is consistent with the original one. The results in Table 7 do live up to our expectation that logarithm may improve the performance of clustering while several contradicting situations are found here. If we do not choose the optimal r as 10 for the NMF method, but just considering the clustering performance and choose r as 3 or 5, then the logarithm transformation deteriorate the performance of clustering. Specifically, we illustrate the clustering result when choosing r as 5 with and without logarithm transformation as Figure 12 and Figure 13 respectively. Obviously, the clustering result without logarithm transformation do cluster more entries from the same class. This may result from the logarithm destroys good boundary of the data entries between two different classes as logarithm itself is a non-linear transformation.

Transformation	Homogeneity	Completeness	V measure	adjusted rand	adjusted mutual info
None	0.5783	0.5903	0.5842	0.6482	0.5783
Scaling	0.1638	0.2364	0.1935	0.1203	0.1637

Table 6: Clustering performance based on SVM method with different transformation applied

Part II

Rather than clustering the documents, we are going to apply this unsupervised learning algorithm to another data type we always use, pictures. The classic data set for picture classification is MNIST, which can be used for clustering as well. This data set contains about 70000 entries of manually written integers. That is to say, the data set has 10 different classes, satisfying the demand on the complexity. We will use just a part of

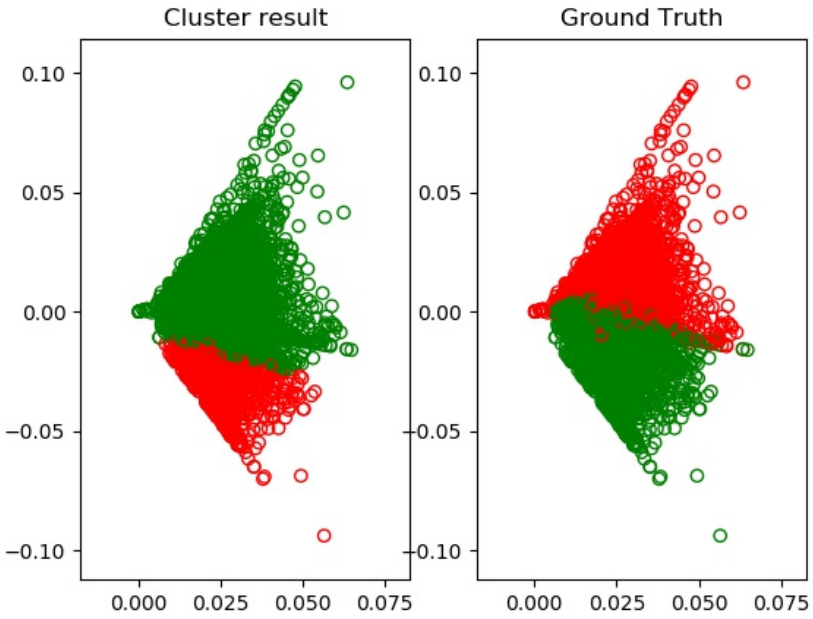

Figure 12: The clustering result for the NMF method with top 5 components chosen.

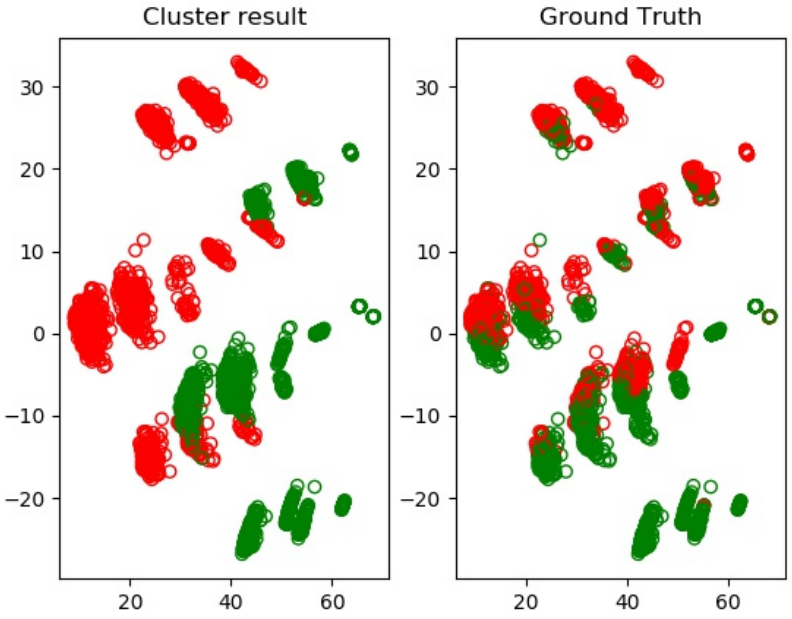

Figure 13: The clustering result when applying logarithm transformation with top 5 components chosen.

Transformation	Homogeneity	Completeness	V measure	adjusted rand	adjusted mutual info
None	0.0951	0.2160	0.1320	0.0360	0.0949
Scaling	0.5092	0.5173	0.5132	0.5955	0.5091
Logarithm	0.1490	0.1493	0.1492	0.1990	0.1483
Scaling&Logarithm	0.1499	0.1503	0.1501	0.2002	0.1498

Table 7: Clustering performance based on NMF method with different transformation applied

the MNIST data set to train our clustering model. Thanks to the evenly distribution of the original data set, sampling from the data set will not induce the imbalance size of classes. To apply the clustering method to our data, we have to derive representative features from the raw data. Before the feature extraction, we will introduce some preprocessing methods to augment our pictures. First, we apply the Gaussian filter to our original graphs in order to eliminate the high frequency noise. Then we will adapt Gamma adjustment to the filtered image. After preprocessing, we can derive the hog feature from the processed images by specifying the cell size and block size. The output of the hog function imported from scikit-image package is a vector that can be directly transmitted to the Kmeans model. The metrics evaluating the performance of the final result is shown below as Table while the cluster result and corresponding ground truth are visualized through T-SNE as Figure 14 and Figure 15. The scores and visualization shown are relatively good considering there are 10 different classes without any prior knowledge given. To improve the performance, we may derive more histograms of gradient from the image, namely, decrease the size of cells and blocks. However, this may result in the higher computer resource demands and over-fitting. Hence, the setting about the cell size and block size should be fine-tuned by parameter grid searching and cross-validation method mentioned in project 1. Because of the time it may cost, we do not implement here.

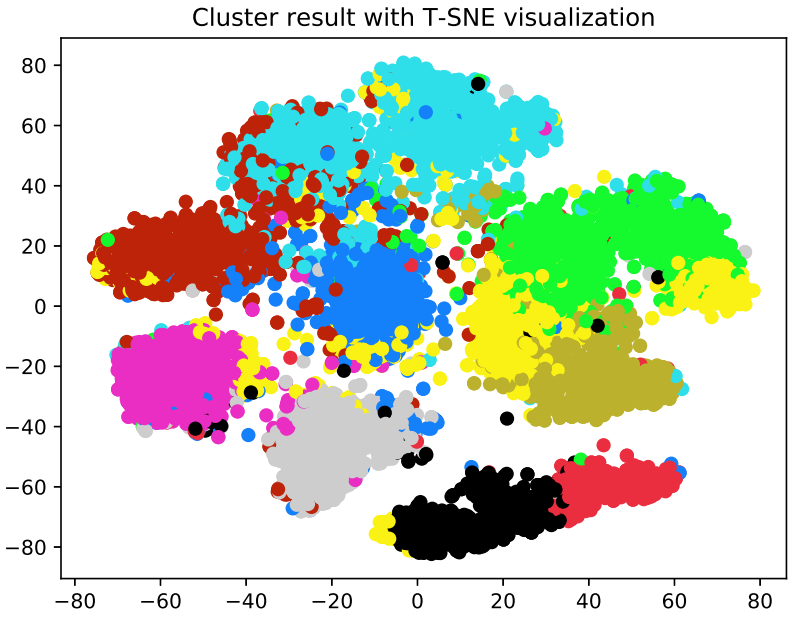

Figure 14: Visualization of the cluster result with T-SNE method.

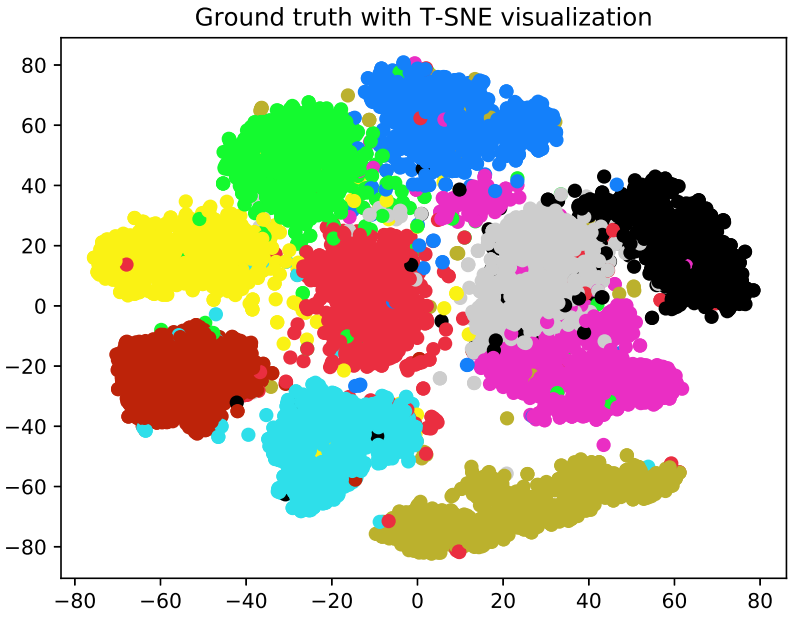

Figure 15: Visualization of the ground truth with T-SNE method.

Method	Homogeneity	Completeness	V measure	adjusted rand	adjusted mutual info
HOG	0.6149	0.6290	0.6219	0.5065	0.6212

Table 8: Clustering performance based on Kmeans without dimensional reduction.

Part III

In this part, we cluster the data of RGB of a real picture. In particular, we first transform the original pixel RGB information to normalized RG space by doing the following:

$$R = \frac{R}{R+G+B}, \; G = \frac{G}{R+G+B} \tag{11.1}$$

Note that by doing so, we are discarding the Blue channel. And we set $K = 3$, the resulting picture is shown below as Figure 16

Question 11

There is a method to choose an appropriate k: given a suitable cluster indicator, such as the average radius or diameter. As long as the number of clusters we assume is equal to or higher than the number of real clusters, the index will rise slowly, and once we try to get less than the true number of clusters, the index will rise sharply. Where the diameter of a cluster is the maximum distance between any two points in the cluster, and the radius of the cluster is the maximum distance from all points in the cluster to the center of the cluster.

There are two other methods for selecting the k initial cluster points: 1) Select k points that are as far away from each other as possible. 2) First cluster the data using a hierarchical clustering algorithm or Canopy

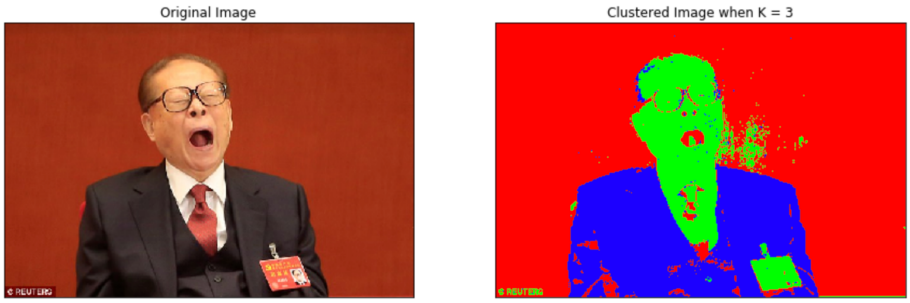

Figure 16: Original picture and color clustering result.

algorithm to obtain k clusters. Select a point in each cluster, this point can be the center of the cluster, or the point closest to the center of the cluster.