EE 219 Project IV - Report

Muralidharan, Vignesh UID: 904729596
Muthappan, Chidambaram UID: 704774938
Jeyakumar, Jeya Vikranth UID: 404749568

Abstract

Cluster analysis is defined as the process of grouping a set of objects/data in such a way that objects in the same group or cluster are more similar (in some sense or another) to each other than to those in other groups (clusters). One of the most popular type of clustering algorithm is Centroid-Based clustering. Clustering is one of the most used unsupervised learning algorithms for classification.

One of the most popular Centroid-Based clustering is K-Means. K-Means clustering can be defined as an optimization problem to find the k cluster centers and assign the objects to the nearest cluster center, such that the squared distances from the cluster are minimized.

The purpose of this project is to find proper representations of the data and evaluate the performance of clustering algorithms.

1 Question 1

The 20 Newsgroup dataset is imported and all the data from the following eight categories are loaded. Clustering does not require separate training and testing data as it is an unsupervised learning algorithm.

- 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

However the raw data, a sequence of symbols cannot be fed directly to the algorithms themselves as most of them expect numerical feature vectors with a fixed size rather than the raw text documents with variable length. We will use CountVectorizer to "convert text into a matrix of token counts"
Features and samples are defined as follows:

- Each individual token occurrence frequency (normalized or not) is treated as a feature.
- The vector of all the token frequencies for a given document is considered a multivariate sample.

A corpus of documents can thus be represented by a matrix with one row per document and one column per token (e.g. word) occurring in the corpus. We call vectorization the general process of turning a collection of text documents into numerical feature vectors. This specific strategy (tokenization, counting and normalization) is called the Bag of Words or "Bag of n-grams" representation. Documents are described by word occurrences while completely ignoring the relative position information of the words in the document.

We defined our tokenizer such that has only letters by using Regular expressions filter and also eliminated the characters with Ascii values greater than 128 by using the ord function. After removing all the punctuations and symbols we did stemming to remove the words with the same meaning by using the Snow-Ball Stemmer. All the common words were also removed from the vocabulary by declaring them as stop words.Terms that occur in less than four documents(very rarely occurring words) were also removed by setting the parameter min df = 2 Then we did the TF-IDF transformation to get the TF-IDF vector representation. TF-IDF stands for "Term Frequency, Inverse Document Frequency". It is a way to score the importance of words (or "terms") in a document based on how frequently they appear across multiple documents.

```
Number of terms Extracted: 30299
Dimensions of TF-IDF vector: (7882, 30299)
```

2 Question 2

For this particular task, we applied k-means clustering with k = 2. k-means clustering, as we know it, is a method of vector quantization, originally from signal processing, that is popular for cluster analysis in data mining. Basically, it 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. This results in a partitioning of the data space into "Voronoi" cells.

In order to make a concrete comparison of different clustering results, there are various measures of purity a given partitioning of the data points with respect to the ground truth. The measures we examine in this project are homogeneity score, completeness score, adjusted rand score and the adjusted mutual info score.

- Homogeneity score is a measure of how purely clusters contain only data points that belong to a single class.
- Clustering result satisfies completeness if all of its clusters contain only data points that belong to a single class.

Both of these scores span between 0 and 1;
where 1 stands for perfect clustering.

- Rand Index is similar to accuracy measure, which computes similarity between the clustering labels and ground truth labels.
- Adjusted Mutual Info measures mutual information score measures mutual information between the cluster label distribution and the ground truth label distributions.

After transforming the documents into TF-IDF vectors, we tried to do k-means clustering and thereby compare the clustering results with the known class labels. We initialized "$n\_clusters$" to 2, "n_init" (Number of time the k-means algorithm will be run with different centroid seeds) was set to its default value - 10, "max_iter" was again set to its default value - 300.

The following values were obtained without any transformations:

```
Homogeneity: 0.246
Completeness: 0.331
V-measure: 0.282
Adjusted Rand-Index: 0.169
Adjusted Mutual Info: 0.238
```

2.1 Permutation of rows for "diagonal" matrix

Yes, there exists a permutation of rows which makes the confusion matrix look diagonal.

```
These are the different permutations for the
obtained confusion matrix
[[ 317 3662]
 [3663  240]]

[[3663  240]
 [ 317 3662]]
```

3 Question 3

In order to find a "better" representation tailored to how the clustering algorithm works. Since in K-means clustering within-cluster distances are minimized in Euclidean l2-norm sense, and it turns out that we need to reduce the dimension of the data properly.

We initially started on with TF-IDF with Singular Value Decomposition (SVD). The reason for choosing SVD is because the idea is to do dimensionality reduction. The range of 'k' was varied from 1 to 6 and a particular value for which it is optimal.

SVD is actually useful in many tasks. Let us consider two examples and explain why it is useful. First, the rank of a matrix A can be read off from its SVD. This is useful when the elements of the matrix are real numbers that have been rounded to some finite precision. Before the entries were rounded the matrix may have been of low rank but the rounding converted the matrix to full rank. Original rank can be determined by the number of diagonal elements of D not exceedingly close to zero. Second, for a square and invertible matrix A, the inverse of A can be found out easily.

In order to gain insight into the SVD, we can treat the rows of any n x d matrix A as n points in a d-dimensional space and considered the problem of finding the best k-dimensional subspace with respect to the set of points. Here, best means minimize the sum of the squares of the perpendicular distances of the points to the subspace.

To get a better idea of the whole system, we tried to analyze the system by incrementing threshold over a scale of 0.5 with 1 as our beginning point and 5 as our end point. There are several advantages in analyzing the threshold on a scale of 0.5 than 1. Firstly, it will be easier to see the the number of entries in our system where the user would like the movie. Based on precision and accuracy values obtained, it will become easier to see which threshold value can be fixed.

The different values that we obtained are:

Table 1: K-means purity measures after Truncated SVD

Dimensions	Homogeneity	Completeness	V-Measure	ARI*	AMI**
1	0.019	0.020	0.020	0.026	0.018
2	0.629	0.629	0.629	0.732	0.633
3	**0.652**	**0.652**	**0.652**	**0.755**	**0.658**
4	0.232	0.318	0.268	0.155	0.232
5	0.232	0.317	0.268	0.156	0.233

ARI* - Adjusted Rand Index; AMI - Adjusted Mutual Index**

Form the above table, it is clearly evident that the most optimum value of the different features is obtained for $\mathbf{k = 3}$.

To visualize it better, we tried to plot it and see if the clustering purity is satisfactory. Since it was not satisfactory, we tried to find a better representation by applying normalizing features and applying some non-linear transformation on the data vectors.

To get a better sense, we initially performed LSI for 2 components followed by LSI for 3 components. The values obtained were:

ARI* - Adjusted Rand Index; AMI - Adjusted Mutual Index**

From Table 6, it is evident that Rand Index value is better when Number of Components is 3. Now, we tried to apply non-linear transformation on the data vectors and made the negative and zero values in transformed tf-idf to a very small value, ie,. $1e^{-6}$. Thereby, we applied logarithm and tried to notice the Rand Index values.

3.1 Applying non-linear transformation after reducing dimensionality

Following are the values we obtained:

Table 2: K-means purity measures after applying log transformation on Truncated SVD

Dimensions	Homogeneity	Completeness	V-Measure	ARI*	AMI**
2	**0.652**	**0.652**	**0.652**	**0.756**	**0.632**
3	0.651	0.651	0.651	0.755	0.658

ARI* - Adjusted Rand Index; AMI - Adjusted Mutual Index**

From Table 7, it is clear that the most optimum value that we obtained was for for k = 2. It is evident that applying some non-linear transformation on the data vectors after reducing dimensionality makes the clustering purity give satisfactory results.

3.2 Visual sense on NMF embedding of data

Table 3: K-means purity measures after NMF

Dimensions	Homogeneity	Completeness	V-Measure	ARI*	AMI**
1	0.020	0.021	0.020	0.026	0.018
2	**0.582**	**0.585**	**0.583**	**0.681**	**0.368**
3	0.041	0.156	0.065	0.009	0.041
4	0.225	0.312	0.261	0.150	0.217
5	0.227	0.314	0.263	0.151	0.233

ARI* - Adjusted Rand Index; AMI - Adjusted Mutual Index**

We used **l2 norm** as the type of normalization. We also tried 'l1' norm and 'none'. But better results were obtained in l2 for the following reasons. Also, we used **logarithmic** nonlinear transformation. We explained the reasons for why l2 normalization and logarithmic nonlinear transformation are better in subsections 3.3 and 3.4.

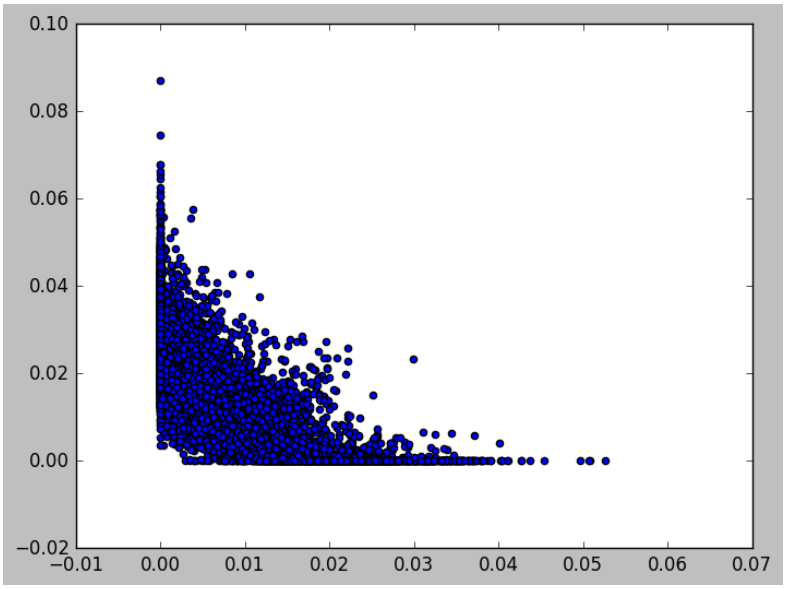

Figure 1: NMF with ambient parameter 2

3.3 Reason for using l2 normalization

Advantages of L2 over L1 norm:

- Derivations of L2 norm can be easily computed. Therefore it is also easy to use gradient based learning methods.
- L2 regularization optimizes the mean cost (whereas L1 reduces the median explanation) which is often used as a performance measurement. This is especially good when we don't have any outliers and you want to keep the overall error small.
- Solution is more likely to be unique. This ties in with the previous point: While the mean is a single value, the median might be located in an interval between two points and is therefore not unique.
- L2 is invariant under rotation. When we had a dataset consisting of points in a space and you apply a rotation, we could still get the same results.

3.4 Justification for logarithm is good for TF-IDF

From Figure 1, it can be seen that the **data is aligned more near the axis**. In other words, NMF with ambient parameter 2 is aligned more towards the axis and not spread out. So, the graph is similar to an asymptotic curve with values clustered along the X and Y axes.

TF-IDF is not a single method, but a class of techniques where similarity between queries and documents is measured via the sum of term frequency-like numbers (TFs) multiplied by terms' importance. The term importance is

frequently expressed via the IDF (the inverse document frequency frequency tf–idf). Actually, it is the logarithm of IDF that is used in practice).

Typically, the more frequently the term occur in a document the larger is the TF coefficient. It is the reverse for the term importance coefficients, which are larger for terms that occur in fewer documents, i.e., more important. Thus, to compute TF*IDF, you need to know the number of term occurrences. TF-IDF methods differ in details with respect to computing the TF and IDF part.

Let us assume there are N documents in the collection, and that term t_i occurs in n_i of them. (What might constitute a 'term' is not very important, but we may assume that terms are words, or possibly phrases or word stems. 'Occurs in' is taken as shorthand for 'is an index term for', again ignoring all the difficulties or subtleties of either automatic indexing from natural language text, or human assignment of index terms.) From a few reference papers that we referred to, we understood that :

$$idf(t_i) = log(\frac{N}{N_i}) \tag{1}$$

So, it can be seen that logarithm is a good candidate by being the **inverse of an exponential function** and thereby **spreading out the data little bit away from the X and Y axes**.

3.5 Applying Non-Linear Transformation on NMF

```
The K-means purity measures after applying log transformation
(7882, 2)
number of components: 2
Homogeneity: 0.627
Completeness: 0.628
V-measure: 0.628
Adjusted Rand-Index: 0.731
Adjusted Mutual-Index: 0.627
Confusion Matrix:
[[3621  358]
 [ 214 3689]]
```

We can clearly see that applying Logarithmic transformation on NMF data gives superior purity values for K-means clustering.

3.6 Reasons why SVD is better than NMF

Both NMF and SVD represent a set of vectors in a given basis. The basis in NMF is composed of vectors with positive elements while the basis in SVD can have positive or negative values.

The difference then is that NMF reconstructs each vector as a positive summation of the basis vectors, in other words you take a little of each vector in the basis to reconstruct your data.

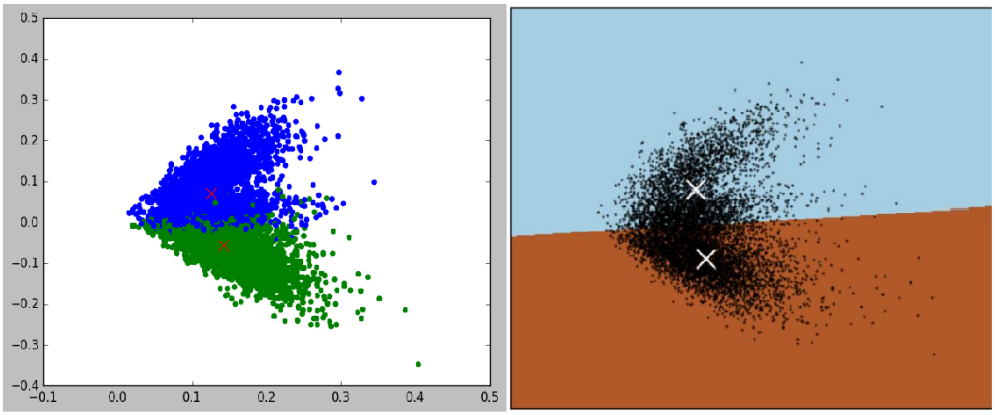

Figure 2: Clustering on Truncated SVD for 2 dimensions

In SVD the data is modeled as a linear combination of the basis you can add or substract vectors as needed.

Another important difference is that in SVD the importance of each vector in the basis is relative to the value of the singular value associated with that vector this usually means that the first vector of the basis dominates and is the most used vector to reconstruct data, then the second vector and so on, so the basis in SVD has an implicit hierarchy and that doesn't happen in NMF. Again which factorization makes sense depends on your application.

4 Question 4

4.1 Visual sense of SVD with plots

In order to visualize the performance of our clustering, we plotted the TruncatedSVD for k = 2 and k = 3, which are shown as in Figures 2 and 3. Particularly, to get a better visualization effect along with the centroid, we initially plotted the graph on 2 dimensions and thereafter proceeded with a 3 dimensional representation.

4.2 Visual sense of NMF with plots

As shown clearly in the case of NMF, after applying log transformation for tf-idf, we observed better plots with cluster spread over a wide range. WE also plotted the centroids in both the cases to visualize it better. We used **l2 norm**

as the type of normalization. We also tried 'l1' norm and 'none'. But better results were obtained in l2 for the following reasons. Also, we used **logarithmic** nonlinear transformation. We explained the reasons for why l2 normalization and logarithmic nonlinear transformation are better in subsections 3.3 and 3.4.

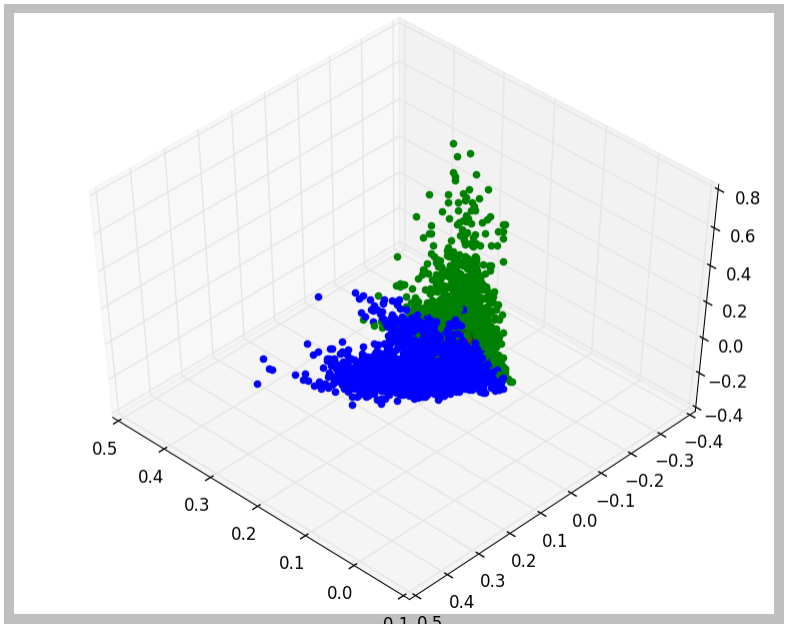

Figure 3: Clustering on Truncated SVD for 3 dimensions

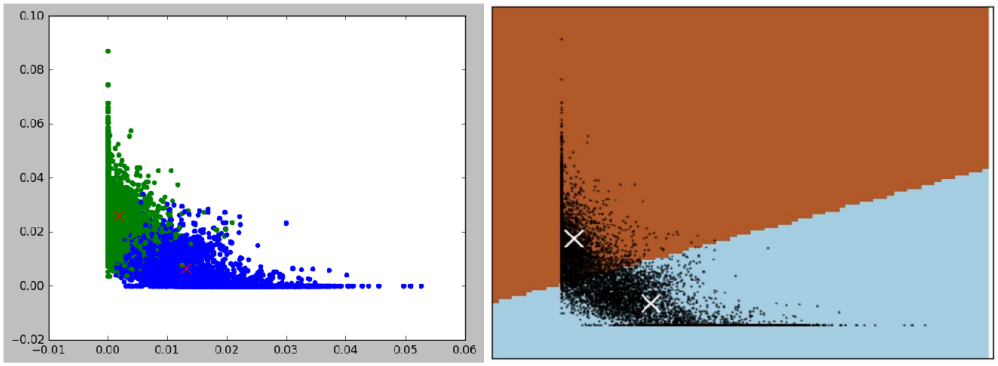

Figure 4: Clustering on NMF for 2 dimensions

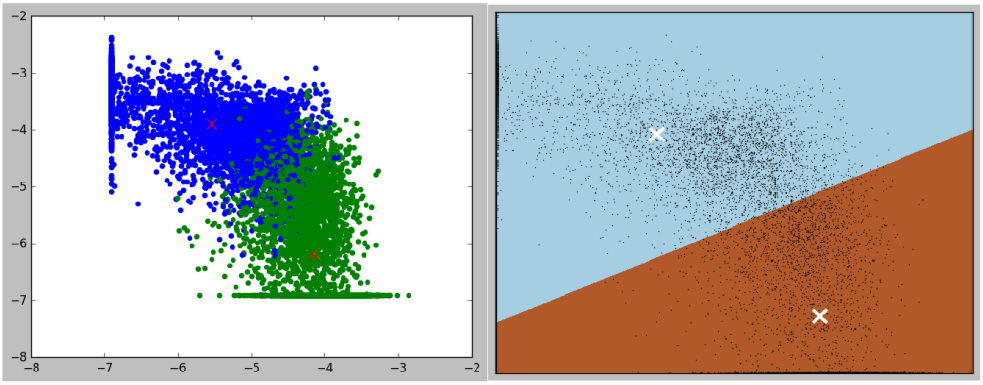

Figure 5: Clustering after Applying log on NMF

4.3 Reasoning for when a non-linear transform is useful

From the table, it can be seen that non-linear transformation is particularly useful in cases when there is a necessity for increasing the linear relationship between variables.

A linear transformation neither increases nor decreases the linear relationship between variables; it preserves the relationship. A nonlinear transformation is used to increase the relationship between variables. The most effective transformation method depends on the data being transformed. In some cases, a logarithmic model may be more effective than other methods; but it other cases it may be less effective. Non-random patterns in a residual plot suggest a departure from linearity in the data being plotted.

When a residual plot reveals a data set to be nonlinear, it is often possible to "transform" the raw data to make it more linear.

Consider the example of a **logarithm** function, which can be explained as a type of non-linear transformation. This is the one which we used.

Generally, in most of the transformations which are non-linear, the following would typically be useful:

- **Symmetry**: First, many statistical techniques work best with data that are single-peaked and symmetric (symmetry).

  A non-linear transformation will reduce positive skewness because it compresses the upper end (tail) of the distribution while stretching out the lower end.

- **Homoscedasticity**: Second, when comparing different groups of subjects, many techniques work best when the variability is roughly the same within each group (homoscedasticity).

A non-linear transformation will often make the within-group variability more similar across groups.

- **Linearity**: Third, it is easier to describe the relationship between variables when it's approximately linear (linearity).

5 Question 5

To to examine how purely we can retrieve all the 20 original sub-class labels with clustering, we included all the documents and their corresponding terms in the data matrix and find proper representation through reducing the dimension of the TF-IDF representation.

We used **l2 norm** as the type of normalization. We also tried 'l1' norm and 'none'. But better results were obtained in l2 for the following reasons. Also, we used **logarithmic** nonlinear transformation. We explained the reasons for why l2 normalization and logarithmic nonlinear transformation are better in subsections 3.3 and 3.4.

We looped the number of dimensions from 1 and observed the different purity measures for both SVD and NMF, which were as follows:

Table 4: K-means purity measures after SVD

Dimensions	Homogeneity	Completeness	V-Measure	ARI*	AMI**
(18846,10)	0.387	0.433	0.409	0.151	0.385
(18846,11)	0.401	0.454	0.426	0.162	0.399
(18846,12)	0.397	0.447	0.421	0.157	0.395
(18846,13)	0.404	0.458	0.429	0.169	0.402
(18846,14)	**0.414**	**0.459**	**0.435**	**0.174**	**0.412**
(18846,15)	0.410	0.456	0.432	0.170	0.408

ARI* - Adjusted Rand Index; AMI - Adjusted Mutual Index**

The following were the values for NMF transformation:

Table 5: K-means purity measures after NMF

Dimensions	Homogeneity	Completeness	V-Measure	ARI*	AMI**
(18846,8)	0.371	0.412	0.390	0.139	0.369
(18846,9)	0.346	0.396	0.369	0.127	0.344
(18846,10)	0.377	0.437	0.405	0.150	0.375
(18846,11)	**0.429**	**0.469**	**0.448**	**0.187**	**0.427**
(18846,12)	0.348	0.399	0.372	0.151	0.346
(18846,13)	0.367	0.427	0.395	0.146	0.365
(18846,14)	0.391	0.459	0.422	0.157	0.389

In doing so, we tried different effective ambient space dimension for both truncated SVD and NMF dimensionality reduction techniques and the different transformations of the obtained feature vectors.

```
After Taking Non-Linear Log Transformation on SVD
Best Dimension : 14
Homogeneity: 0.439        Completeness: 0.476
V-measure: 0.456        Adjusted Rand-Index: 0.227
Adjusted Mutual-Index: 0.437
```

```
After Taking Non-Linear Log Transformation on NMF
Best Dimension : 11
Homogeneity: 0.474        Completeness: 0.498
V-measure: 0.486        Adjusted Rand-Index: 0.252
Adjusted Mutual-Index: 0.473
```

6 Question 6

We reduced the 20 classes used in the previous question to 6 by grouping the 'similar' topic-wise classes. After grouping it that way, we observed the values:

We used **l2 norm** as the type of normalization. We also tried 'l1' norm and 'none'. But better results were obtained in l2 for the following reasons. Also, we used **logarithmic** nonlinear transformation. We explained the reasons for why l2 normalization and logarithmic nonlinear transformation are better in subsections 3.3 and 3.4.

Tables 10, 11 and 12 show the values for SVD, NMF and NMF applied with log respectively.

Table 6: Values obtained for SVD after topic-wise clustering

Dimensions	Homogeneity	Completeness	V-Measure	ARI*	AMI**
(18846,1)	0.013	0.014	0.014	0.008	0.013
(18846,2)	0.169	0.172	0.170	0.098	0.169
(18846,3)	0.208	0.227	0.217	0.105	0.207
(18846,4)	**0.237**	**0.278**	**0.256**	**0.119**	**0.237**
(18846,5)	0.202	0.246	0.221	0.063	0.201

Table 7: Values obtained for NMF after topic-wise clustering

Dimensions	Homogeneity	Completeness	V-Measure	ARI*	AMI**
(18846,9)	0.227	0.288	0.254	0.103	0.227
(18846,10)	0.170	0.221	0.192	0.052	0.170
(18846,11)	**0.261**	**0.345**	**0.297**	**0.140**	**0.261**
(18846,12)	0.212	0.272	0.238	0.087	0.212
(18846,13)	0.166	0.263	0.204	0.093	0.166
(18846,14)	0.161	0.215	0.184	0.095	0.160

```
After Taking Non-Linear Log Transformation on SVD
Best Dimension : 14
Homogeneity: 0.439        Completeness: 0.476
V-measure: 0.456        Adjusted Rand-Index: 0.227
Adjusted Mutual-Index: 0.437
```

```
After Taking Non-Linear Log Transformation on NMF
Best Dimension : 11
Homogeneity: 0.474        Completeness: 0.498
V-measure: 0.486        Adjusted Rand-Index: 0.252
Adjusted Mutual-Index: 0.473
```