ECE219 Project 2: Clustering

Firnaz Ahamed (104943091) and Larsan Aro Brian (904943186)

February 11, 2018

1 Introduction

Clustering is a common form of unsupervised learning where there is no a priori labeling of the data points. Instead, the algorithm tries to find a structure in the data points by grouping the points that lie close together in a multidimensional space. Clustering algorithms find a lot of application in real-life, such as finding groups of customers with similar preferences based on their buying records. In this project, we attempt to do clustering on textual data. We implement dimensionality reduction that will best represent the data for good clustering results. We also experiment with different preprocessing techniques that help us to achieve better results.

2 Dataset

The dataset which was used for this project was the '20 Newsgroups' dataset. The dataset had a total of 18846 newsgroup documents which were classified evenly across 20 classes. The dataset was imported directly from the scikit-learn library. For a major part of this project, a subgroup of this dataset corresponding to 8 classes were used. The dataset corresponding to these 8 classes had 4732 training documents and 3150 testing documents.

3 Cleaning the Raw Text

In any text based mining problem, be it classification or clustering, the first task at hand is to clean the data. This means that we must convert the raw text into a form in which we would be able to extract useful inferences to begin our clustering problem. The major tasks in this process would be to extract words from the raw text by eliminating punctuations and stop words. In the cleaning process, first all the punctuations and numbers were removed using regular expression. All the words were converted to lower case, stop words were removed.

4 Term-Frequency Inverse Document Frequency (TF-IDF)

TF-IDF refers to a commonly used statistical measure that determines how important a word is to a document in a collection of documents or corpus. The importance is determined by a weight that is given based on the two components called term frequency and inverse document frequency. The basic idea behind tf-idf is that when a term occurs more frequently, it is more important since it carries the bulk of the information in the document. It is important as to how much information the word carries and contributes to the document, and based on this the weight should be adjusted. Thus tf-idf weight consists of the following components:

Term frequency The frequency of occurrence of a particular term in a document is called term frequency. The words with a higher term frequency will be given a higher weight.

Inverse document frequency This gives the measure of how important the word is to the document and how much information it provides to the document. It is found by taking the negative logarithm of the fraction of documents that contain the particular word out of all the documents in the corpus. The weight of the words that occur in most of the documents will be reduced, while the less commonly occurring words' weights will be increased.

5 Creating the TFIDF matrix

Once the text was cleaned, the TFIDF matrix was created using the TfidfVectorizer from scikit-learn package. It first creates a bag of words model and then applies tfidf to obtain the tfidf weights. The bag of words model refers to the matrix that represents the frequency of occurrence of each word in each document of the corpus. Each column in the matrix refers to an unique word from the documents and each row represents the document itself. The value present in the matrix refers to the frequency of occurrence of the word in that document.The TfidfTransformer object takes the bag of words model matrix as input and returns a sparse matrix where the columns correspond to the words in the data and the rows correspond to the tfidf weights. The min_df parameter is used to determine the minimum number of occurrence of a term in a document for it be considered in the matrix. We used min_df as 3 for the remainder of the project.

min_df	Number of terms
2	32653
3	23135
5	15454

6 K-Means Clustering

The K-means clustering is vector quantization method used in cluster analysis. K-means clustering is used mainly as a feature learning step in unsupervised and semi-supervised learning. The K-means clustering algorithm partitions a set of data points into k clusters, such that the sum of the squares of the distances between the data points and the center of the cluster is minimized. Each data point belongs to only one cluster.

6.1 Clustering Results

We applied K-means clustering on the given dataset with 2 clusters using sklearn's KMeans class and using K-Means++ algorithm for initialization. The clustering results can be evaluated using the contingency matrix and several measures of purity.

Contingency matrix

		Cluster	
		0	1
Class	0	3556	347
	1	63	3916

Clustering metrics

	Clustering with k=2
Homogeneity score	0.7217
Completeness score	0.7251
V-measure	0.7234
Adjusted Rand score	0.8027
Adjusted mutual info score	0.7216

7 Dimensionality Reduction

Since we are dealing with high dimensional matrices, dimensionality reduction can be applied as part of preprocessing for better data representation. The high dimensional vectors are mapped to lower dimensions that will best explain the variance in the data. Singular Value Decomposition (SVD) plays a significant role in the dimensionality reduction process. In this project, we have used two different dimensionality reduction techniques, namely Latent Semantic Indexing (LSI) and Non-Negative Matrix Factorization (NMF) to map the high dimensional vectors to lower dimensions. First, we have plotted how much variance the top r principal components can retain in the case of Truncated SVD.

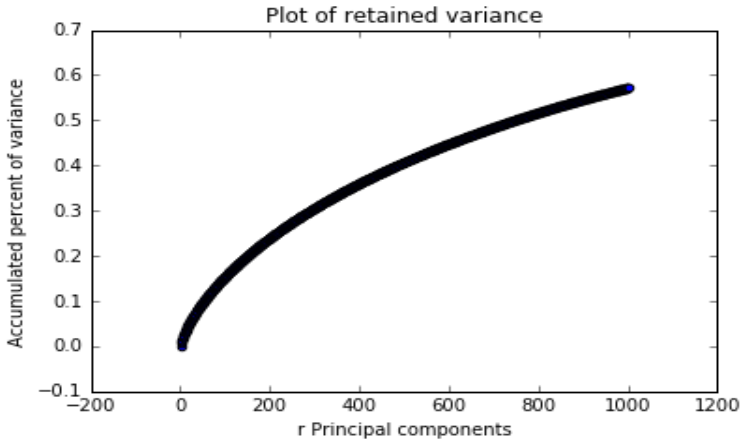

Figure 1: Plot of the variance for different number of principal components

Inference As seen from the plot above, the accumulated percentage of retained variance increases steeply initially and then slows down. This shows that the first few principal components, which correspond to the largest singular values, retain a larger percentage of the variance than those corresponding to the smaller singular values.

8 Finding the best r

In this section, we try to determine the best value of the number of dimensions to map the data points on to, by examining the various clustering metrics for different values of r. This was done using both LSI and NMF as the dimensionality reduction technique.

8.1 Truncated SVD

The different measures for different values of r ranging from 1 to 300 were plotted and the contingency matrices were found as well. Since we were plotting for unevenly spaced values of r, we decided to use scatter plot instead of line plot.

8.1.1 Homogeneity

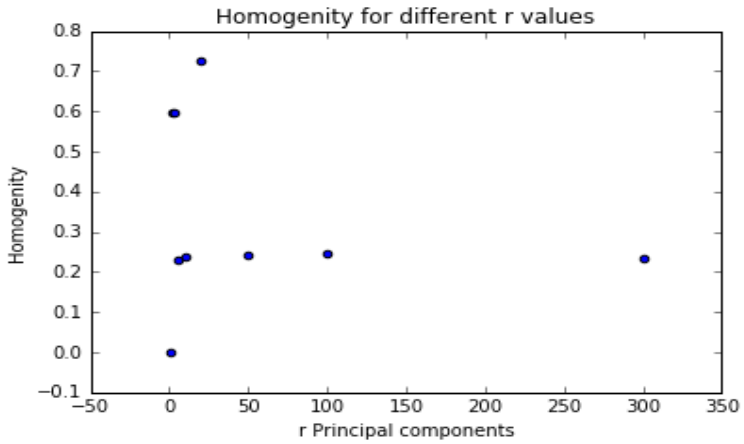

Figure 2: Plot of the homogeneity for different number of principal components

8.1.2 Completeness

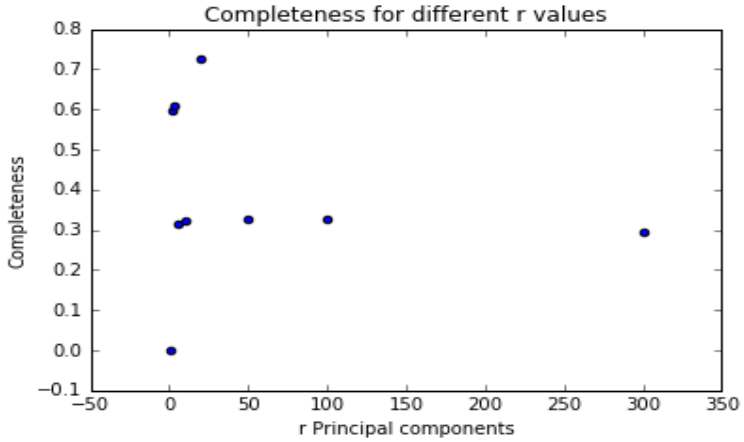

Figure 3: Plot of the completeness for different number of principal components

8.1.3 V-Measure

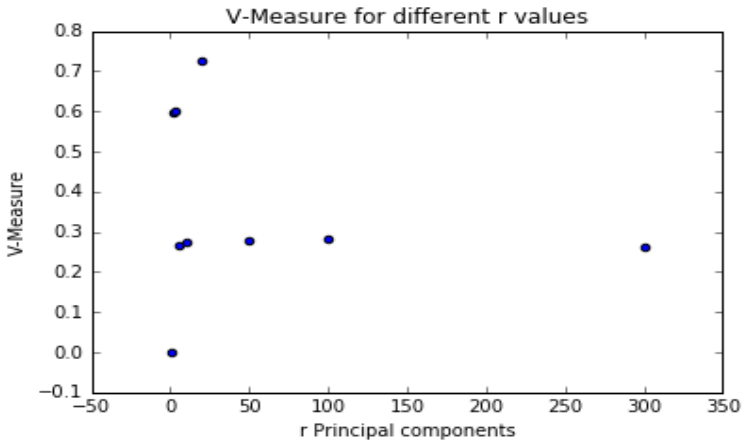

Figure 4: Plot of the V-Measure for different number of principal components

8.1.4 Adjusted Rand index

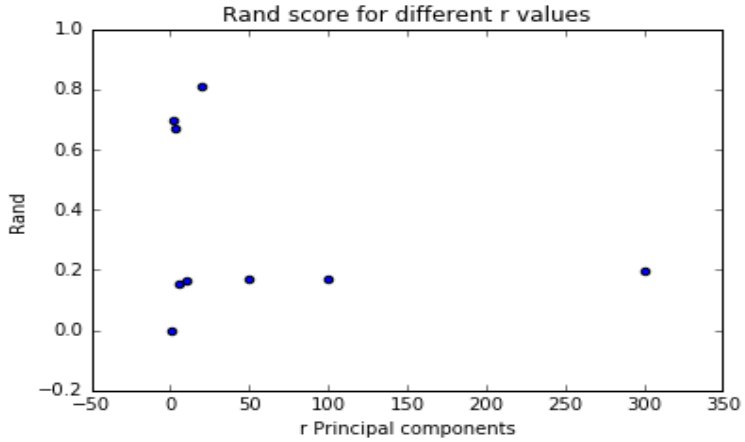

Figure 5: Plot of the adjusted Rand index for different number of principal components

8.1.5 Adjusted mutual info score

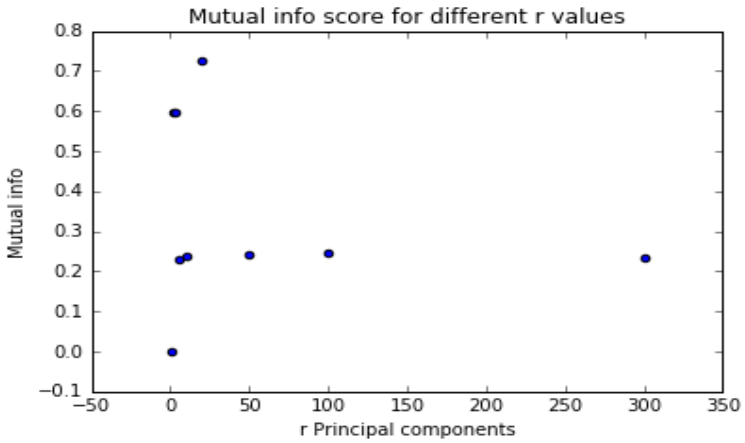

Figure 6: Plot of the adjusted mutual info score for different number of principal components

8.1.6 Contingency matrices

r=1

		Cluster	
		0	1
Class	0	2167	1736
	1	2294	1685

r=2

		Cluster	
		0	1
Class	0	454	3449
	1	3781	198

r=3

		Cluster	
		0	1
Class	0	639	3264
	1	3905	74

r=5

		Cluster	
		0	1
Class	0	3898	5
	1	2397	1582

r=10

		Cluster	
		0	1
Class	0	3898	5
	1	2336	1643

r=20

		Cluster	
		0	1
Class	0	319	3584
	1	3901	78

r=50

		Cluster	
		0	1
Class	0	4	3899
	1	1662	2317

r=100

		Cluster	
		0	1
Class	0	3899	4
	1	2309	1670

r=300

		Cluster	
		0	1
Class	0	66	3837
	1	1848	2131

8.1.7 Best value of r

After running the program and looking at all the plots, it was seen the $r = 20$ resulted in the best metric values. For all the measures, $r = 20$ gave the highest score.

8.2 Non-negative Matrix Factorization (NMF)

NMF was fitted for different values of r and the different measures were plotted and the contingency matrices were noted. In the case of NMF, for each value of r, a new NMF matrix was formed unlike the case of LSI where the top principal components were taken.

8.2.1 Homogeneity

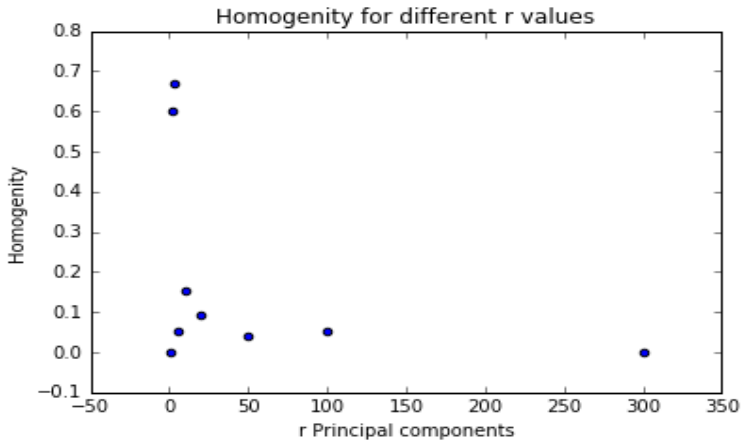

Figure 7: Plot of the homogeneity for different number of principal components

8.2.2 Completeness

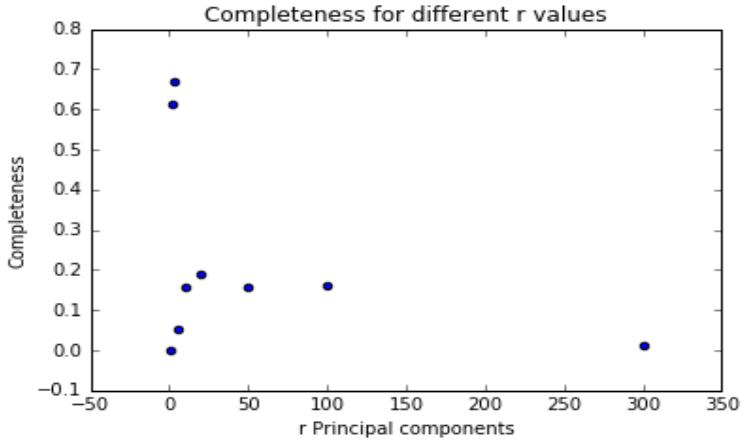

Figure 8: Plot of the completeness for different number of principal components

8.2.3 V-Measure

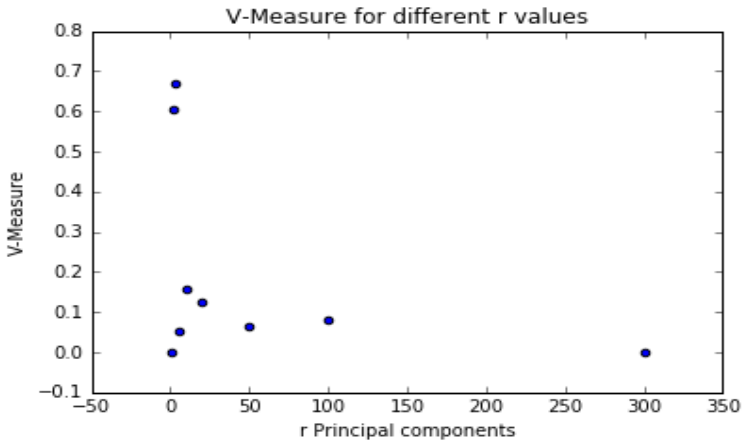

Figure 9: Plot of the V-Measure for different number of principal components

8.2.4 Adjusted Rand index

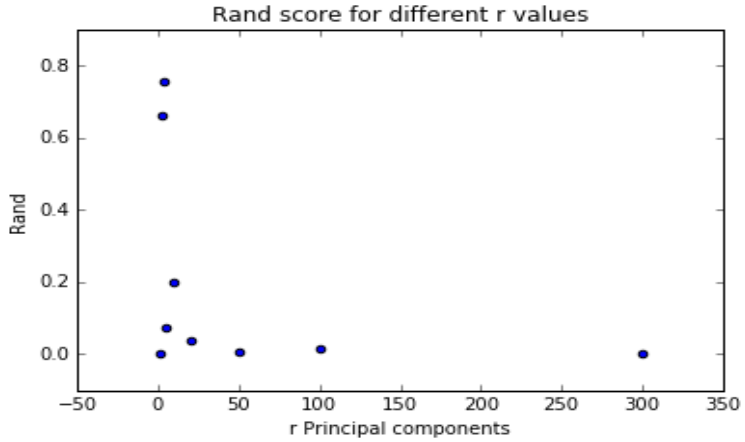

Figure 10: Plot of the adjusted Rand index for different number of principal components

8.2.5 Adjusted mutual info score

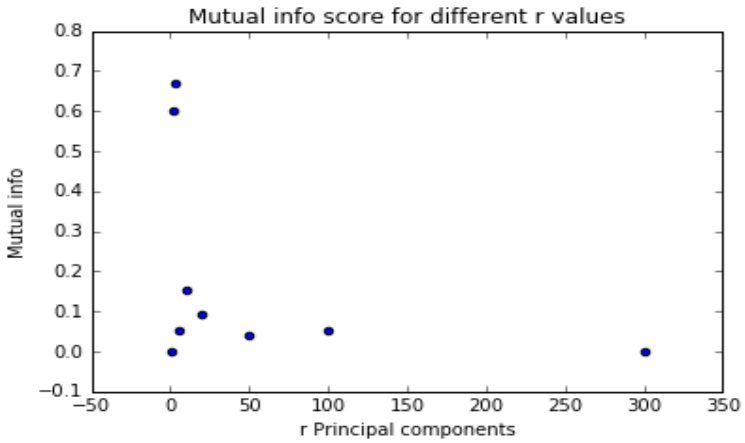

Figure 11: Plot of the adjusted mutual info score for different number of principal components

8.2.6 Contingency matrices

r=1

		Cluster	
		0	1
Class	0	2167	1736
	1	2294	1685

r=2

		Cluster	
		0	1
Class	0	3213	690
	1	46	3933

r=3

		Cluster	
		0	1
Class	0	3488	415
	1	99	3880

r=5

		Cluster	
		0	1
Class	0	2601	1302
	1	1574	2405

r=10

		Cluster	
		0	1
Class	0	1417	2486
	1	3218	761

r=20

		Cluster	
		0	1
Class	0	3875	28
	1	3149	830

r=50

		Cluster	
		0	1
Class	0	336	3567
	1	5	3974

r=100

		Cluster	
		0	1
Class	0	3453	450
	1	3968	11

r=300

		Cluster	
		0	1
Class	0	3899	4
	1	3969	10

8.2.7 Best value of r

After running the program and looking at all the plots, it was seen the $r = 3$ resulted in the best metric values. For all the measures, $r = 3$ gave the highest score for the NMF reduced matrix.

8.2.8 Non-monotonic behavior

As we have seen from the graphs above, there is a non-monotonic behavior in the variation of the measures as r increases and it can also be observed that the best value of r is usually a small number. There are two opposing forces which result in such a behavior. Firstly, the clustering results become poor in high dimensions since the data points become very spaced out and the notion of distance loses its meaning in such high dimensions. This is called the curse of dimensionality. Once the number of dimensions become large, it becomes hard and meaningless to quantify distance between two points in such high dimensions. This is one of the reasons why dimensionality reduction becomes extremely important in certain machine learning algorithms like K-means clustering and KNN.

So one might think that reducing the dimensions as much as possible might be effective. However, there is the issue of losing information as the number of dimensions are reduced. Each dimension or each column in the matrix is a feature. By using principal component analysis we are trying to find the dimensions which retain the

maximum information or variance in the data, but we effectively lose some information by neglecting the other dimensions. Also, by mapping all the data points to a single dimension or two dimensions, we are clamping all the data points together and dividing the data points into clusters become hard. This issue is even more pronounced if we are restricted to linear separation.

Hence, there are these two opposing forces which affect the clustering results differently as the number of dimensions vary. Effectively, this results in a non-monotonic behavior of measures as the number of dimensions vary and results in a small value of r, though not the smallest value possible, to be the best value.

9 Visualizing the best case

By mapping the results of the best case of r onto a two dimensional plane and color coding the data points based on the cluster to which it belongs, the following plots were obtained for the two cases of LSI and NMF.

9.1 LSI

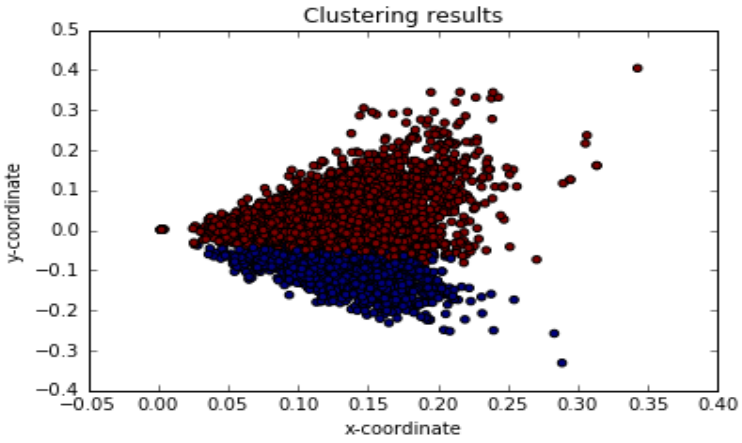

Figure 12: Clustering results for $r = 20$ in LSI

9.2 NMF

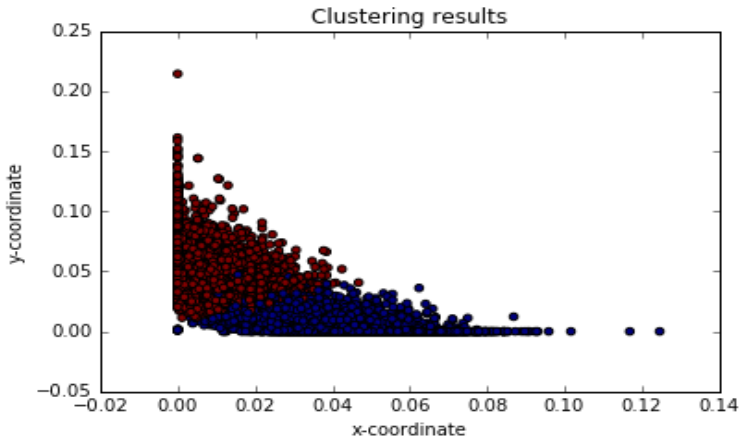

Figure 13: Clustering results for $r = 3$ in NMF

10 Normalizing the features

Normalization was implemented on each of the features by using StandardScaler from sklearn library which ensures that each feature has unit variance. This is achieved by dividing each column by its standard deviation. Normalization for both LSI and NMF, and the following results and measures were obtained.

10.1 LSI

The visualization, contingency matrix and measures for the LSI reduced matrix are as follows.

10.1.1 Plot

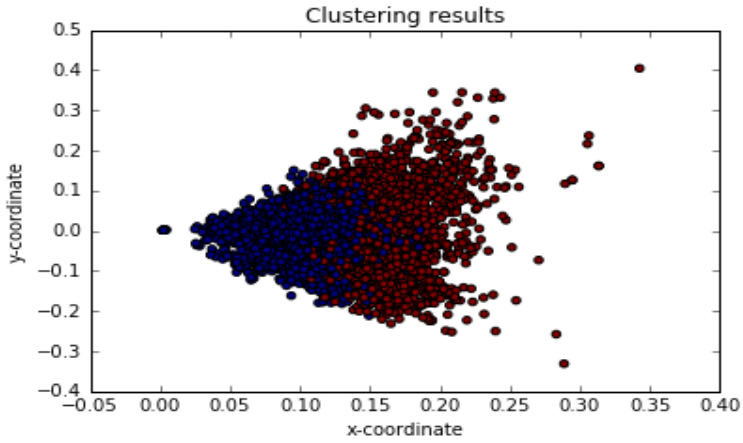

Figure 14: Clustering results after normalization in LSI

10.1.2 Contingency matrix

		Cluster	
		0	1
Class	0	1808	2095
	1	3674	305

10.1.3 Measures

	LSI with normalization
Homogeneity score	0.1964
Completeness score	0.2215
V-measure	0.2082
Adjusted Rand score	0.2150
Adjusted mutual info score	0.1963

10.1.4 Inference

The clustering results become poor after normalizing the data. As seen from the plot, there is quite a lot of overlap between the red and blue data points in the middle. There is good clustering only at the sides. The LSI reduced data matrix contains both positive and negative values and the normalization only results in smaller variance in the positive and negative entries.

10.2 NMF

The visualization, contingency matrix and measures for the NMF reduced matrix are as follows.

10.2.1 Plot

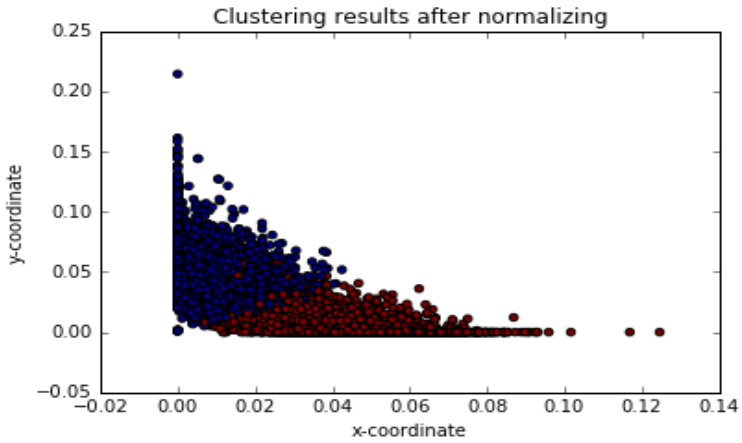

Figure 15: Clustering results after normalization in NMF

10.2.2 Contingency matrix

		Cluster	
		0	1
Class	0	3613	290
	1	160	3815

10.2.3 Measures

	NMF with normalization
Homogeneity score	0.6846
Completeness score	0.6854
V-measure	0.6850
Adjusted Rand score	0.7828
Adjusted mutual info score	0.6846

10.2.4 Inference

Normalization has reduced the clustering results by a small margin. In the case of LSI, normalization resulted in very poor results. However, in NMF reduced data the results are not as bad. The reason for this might be due to the fact the NMF data matrix does not have any negative values and hence the normalization yields better variance in the data than that of the LSI reduced data.

11 Transformations

Non-linear transformations such as log transformation was applied to examine the difference in clustering results. The following results were obtained by applying the mentioned transformations on a NMF reduced matrix.

11.1 Log transformation

In the NMF reduced matrix, a value of 1 was added to each element before taking log on the matrix.

11.1.1 Plot

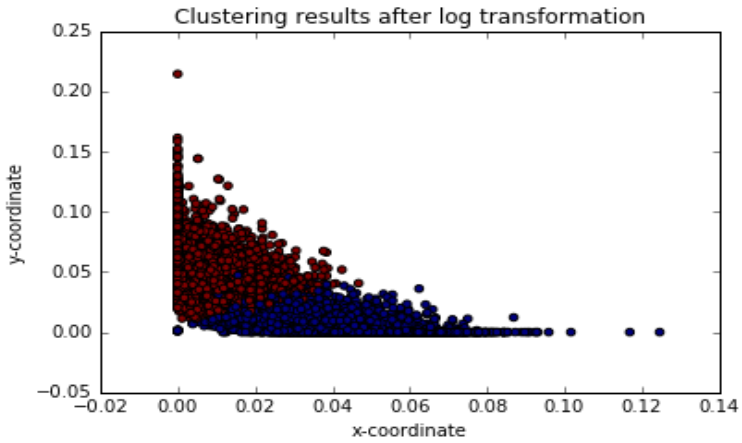

Figure 16: Clustering results after log transformation in NMF

11.1.2 Contingency matrix

		Cluster	
		0	1
Class	0	3497	406
	1	101	3878

11.1.3 Measures

	NMF with log transform
Homogeneity score	0.6699
Completeness score	0.6736
V-measure	0.6717
Adjusted Rand score	0.7592
Adjusted mutual info score	0.6699

11.1.4 Inference and importance of log transformation

In general, log transformation is able to achieve pretty good results. Log transformation can be used to nullify the effect of outliers. Some data points which are far away can be brought closer by taking the log transformation and helps to achieve better clustering results. Log transformations are also specifically useful for skewed data. It helps to reduce the variability in the data and conform more closely to the normal distribution.

11.2 Normalization and log transformation

The NMF reduced matrix was first normalized. After normalization, the matrix will contain both positive and negative values. Hence, before taking the log transformation, the minimum value is subtracted so that the minimum becomes zero and then a value of 1 is added to all the elements as well.

11.2.1 Plot

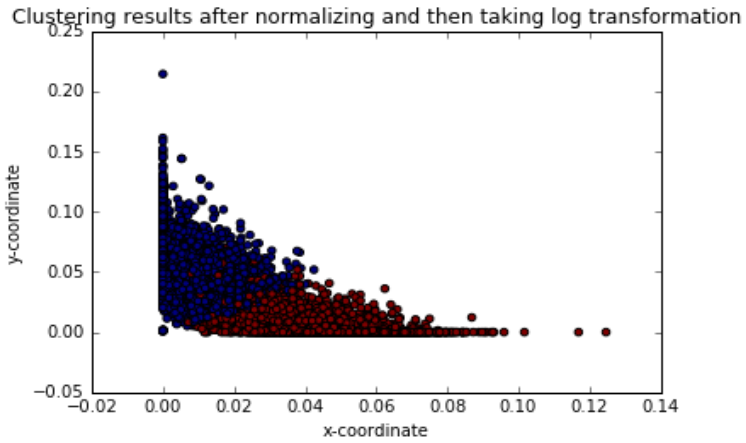

Figure 17: Clustering results after normalization and log transformation in NMF

11.2.2 Contingency matrix

		Cluster	
		0	1
Class	0	3556	347
	1	155	3824

11.2.3 Measures

	NMF with normalization and log transform
Homogeneity score	0.6634
Completeness score	0.6650
V-measure	0.6642
Adjusted Rand score	0.7614
Adjusted mutual info score	0.6634

11.2.4 Inference

The results achieved by taking normalization followed by log transformation are pretty good. The non-linear transformations have helped to achieve good results.

11.3 Log transformation and normalization

A log transformation was first applied on the NMF reduced matrix after taking shifting all the elements by 1.Then the matrix was normalized using StandardScaler.

11.3.1 Plot

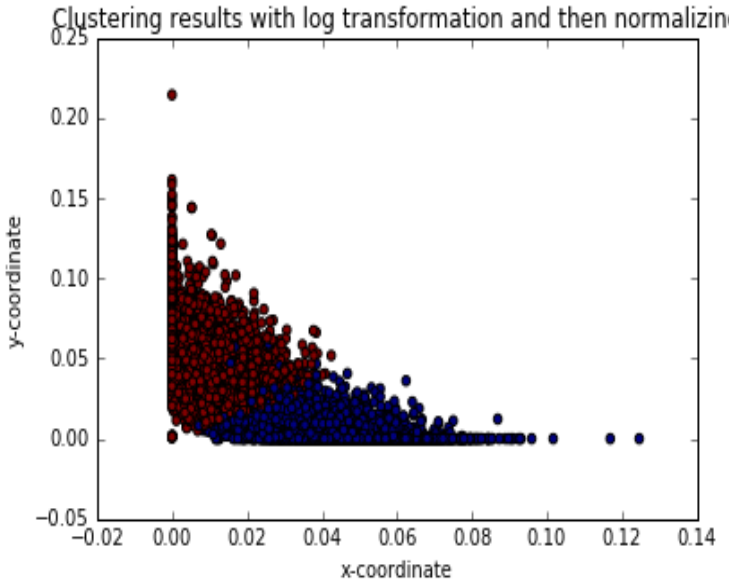

Figure 18: Clustering results after log transformation and normalization in NMF

11.3.2 Contingency matrix

		Cluster	
		0	1
Class	0	3611	292
	1	161	3818

11.3.3 Measures

	NMF with log transform and normalization
Homogeneity score	0.6854
Completeness score	0.6862
V-measure	0.6858
Adjusted Rand score	0.7833
Adjusted mutual info score	0.6853

11.3.4 Inference

The results achieved by taking log transformation followed by normalization are pretty good as well. The non-linear transformations have helped to achieve good results.

12 20 Classes with LSI

Now we expand the number of classes to 20 to include the entire dataset. We'll apply both LSI and NMF as dimensionality reduction techniques to make comparisons. In this section, we have used LSI as the dimensionality reduction technique. The TFIDF matrix that was created earlier was used for this.

12.1 Finding the best r

Once again, we'll try to find the best r in the same range that was used in the previous case. The following are the accuracy measures and contingency matrices that were obtained.

12.1.1 Purity measures

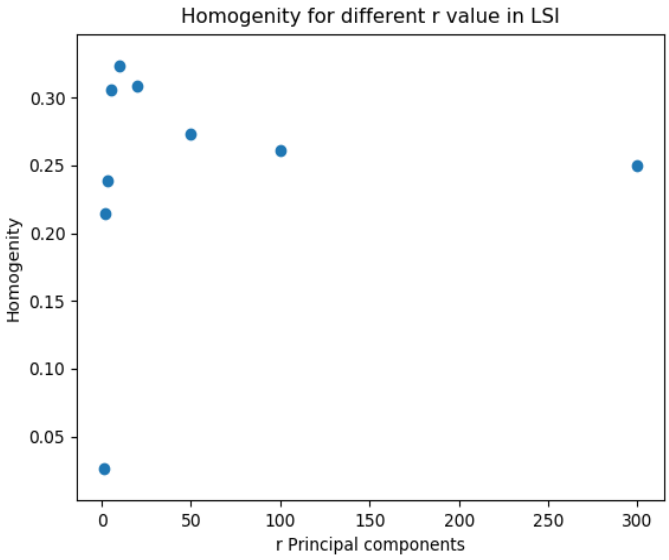

Figure 19: Plot of Homogeneity for different values of r in LSI

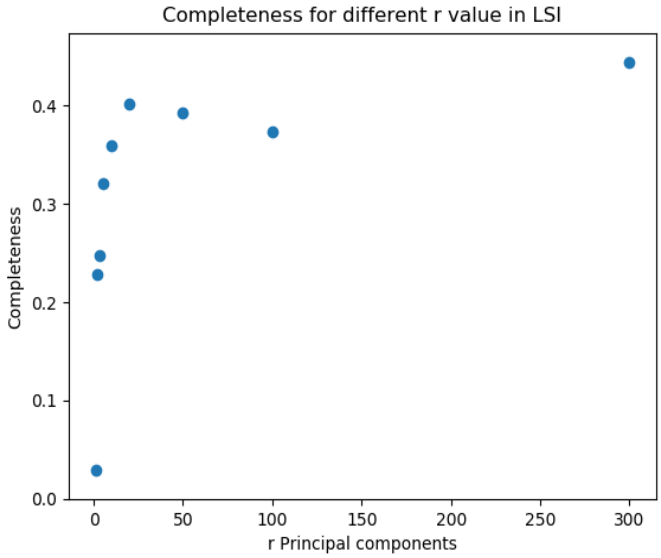

Figure 20: Plot of completeness for different values of r in LSI

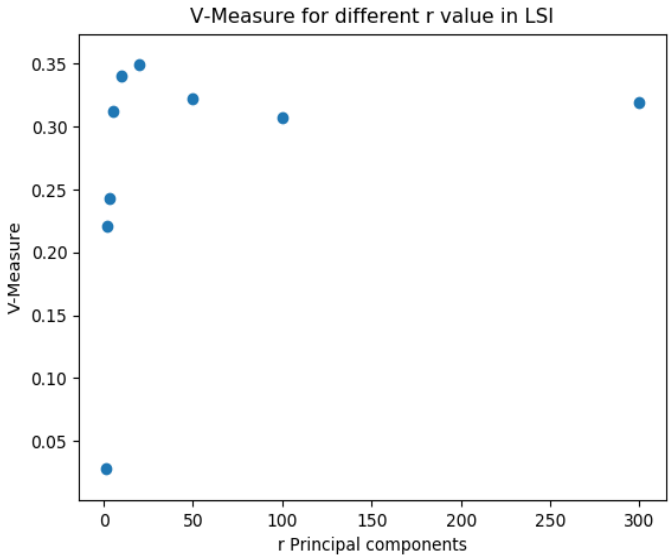

Figure 21: Plot of V-measure for different values of r in LSI

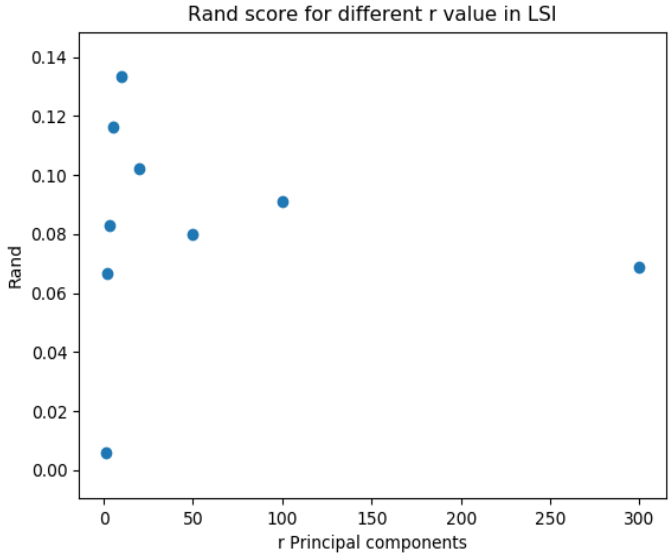

Figure 22: Plot of Rand for different values of r in LSI

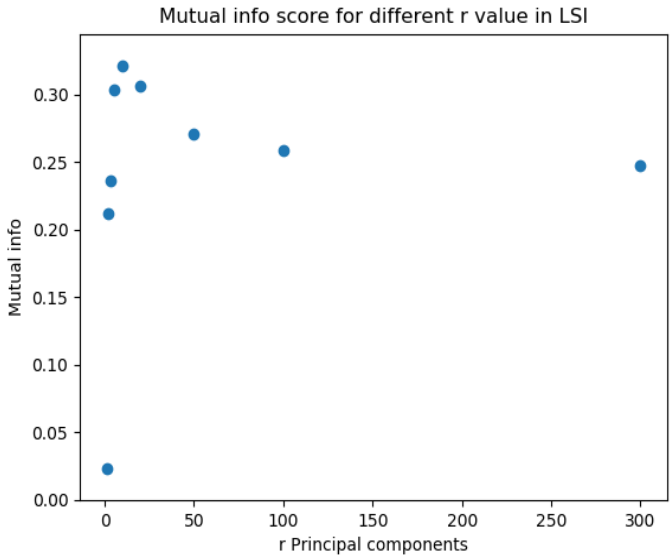

Figure 23: Plot of mutual info score for different values of r in LSI

12.1.2 Contingency matrices

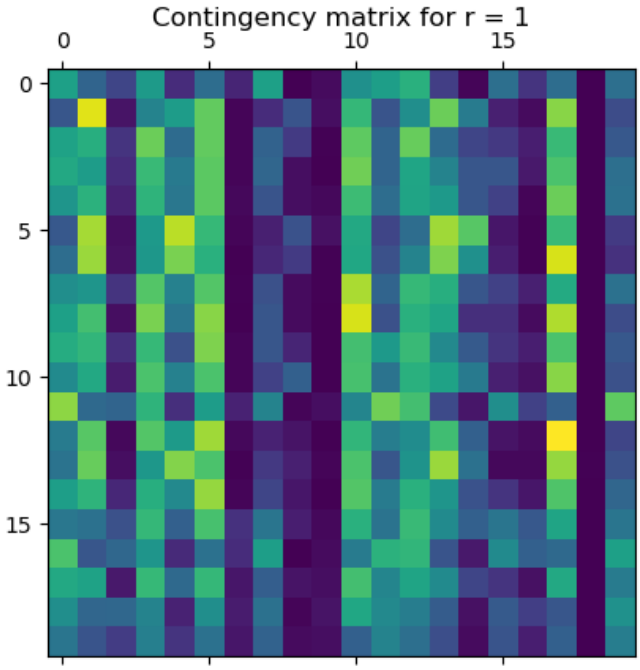

Figure 24: Contingency matrix for r=1

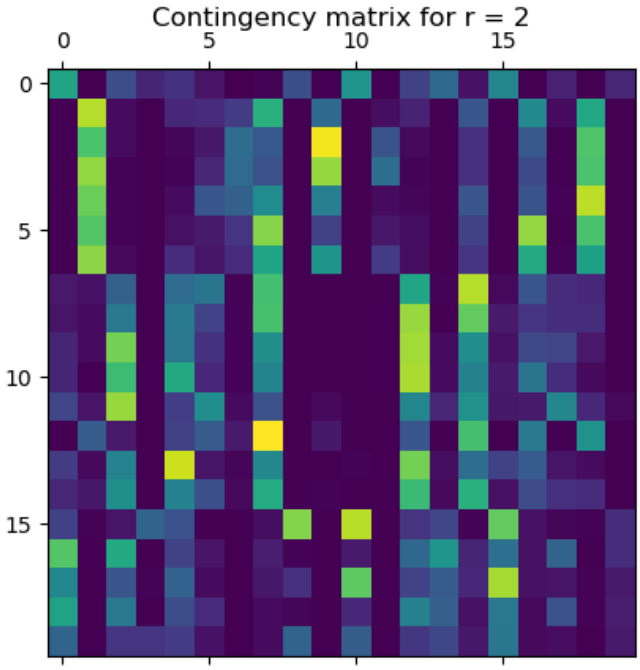

Figure 25: Contingency matrix for r=2

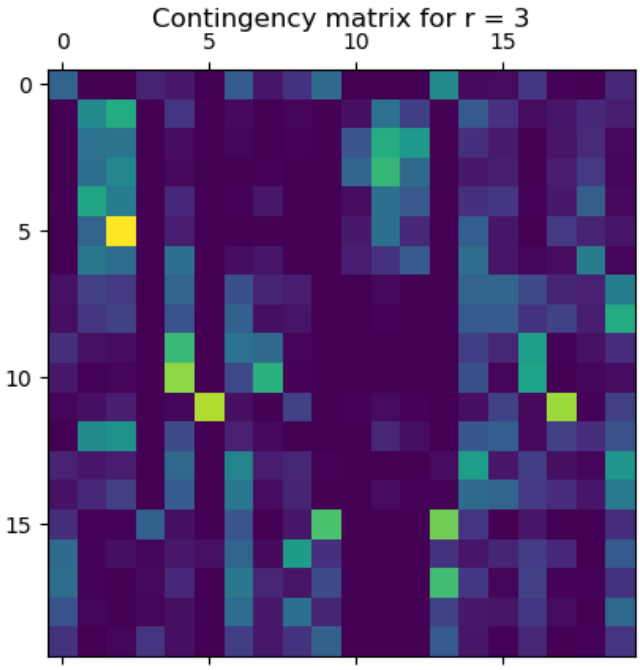

Figure 26: Contingency matrix for r=3

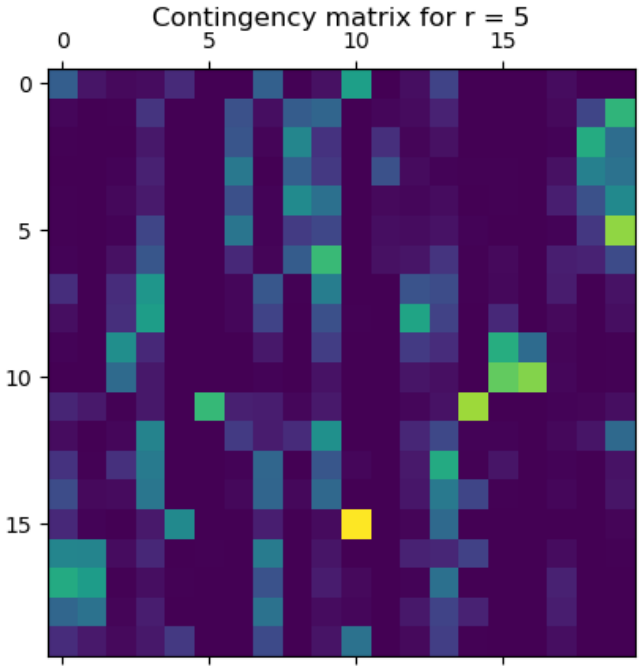

Figure 27: Contingency matrix for r=5

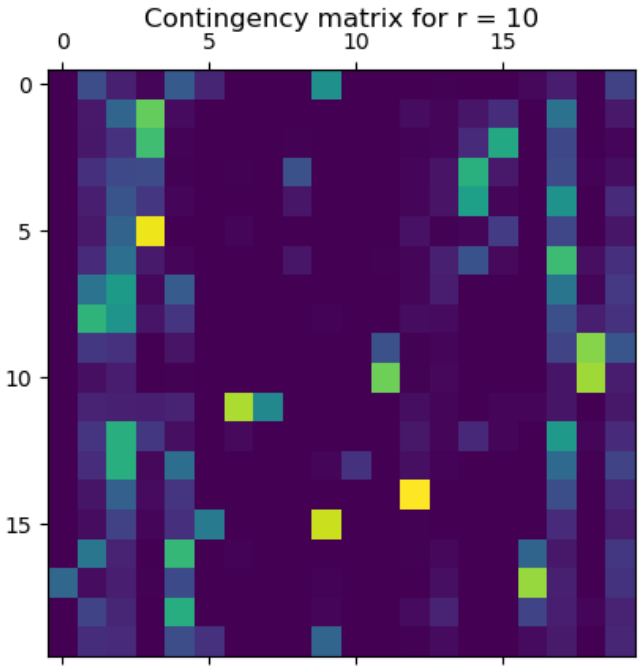

Figure 28: Contingency matrix for r=10

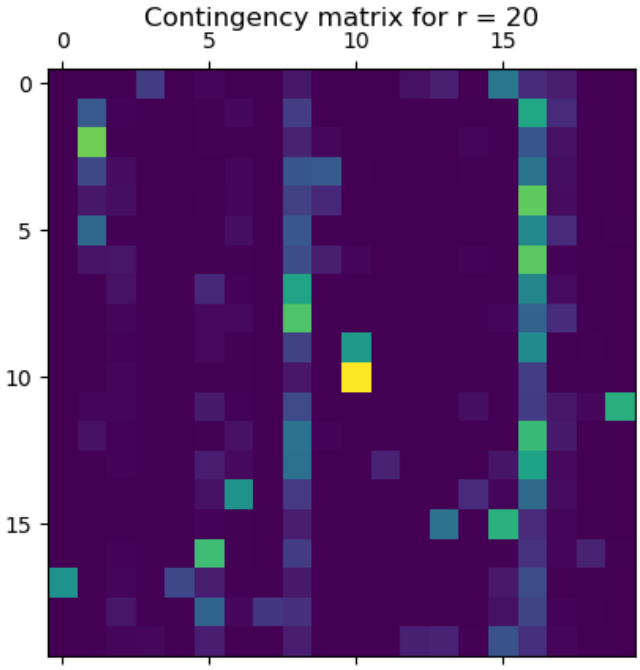

Figure 29: Contingency matrix for r=20

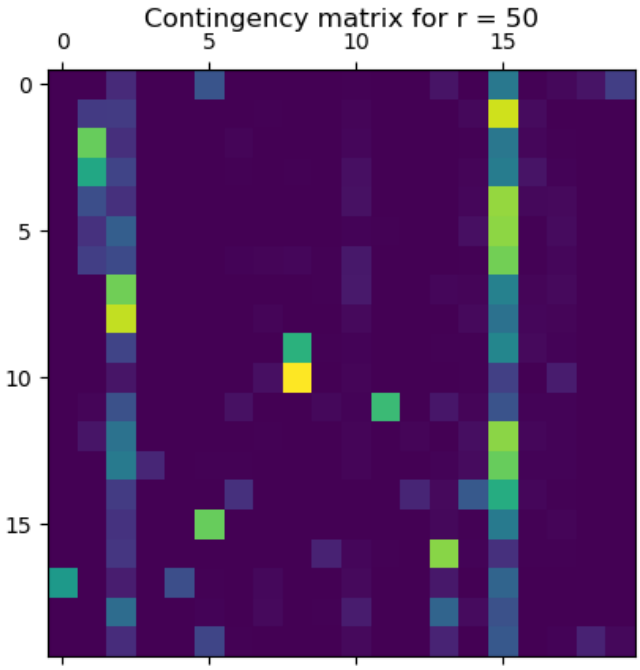

Figure 30: Contingency matrix for r=50

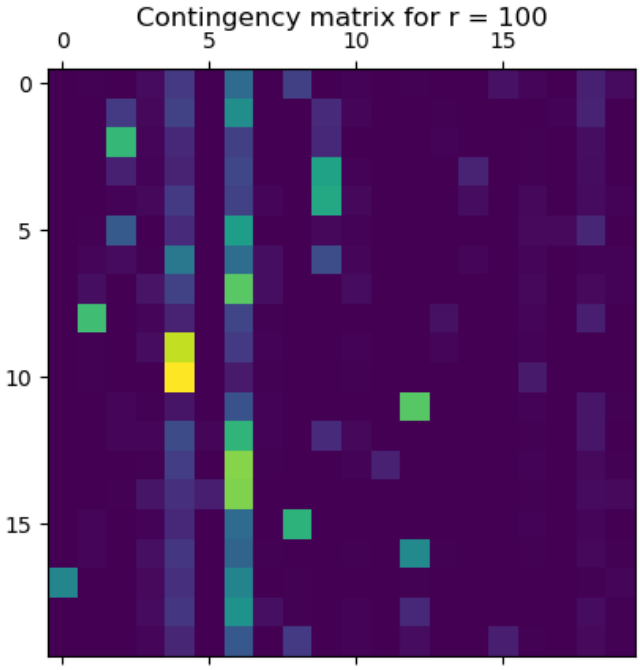

Figure 31: Contingency matrix for r=100

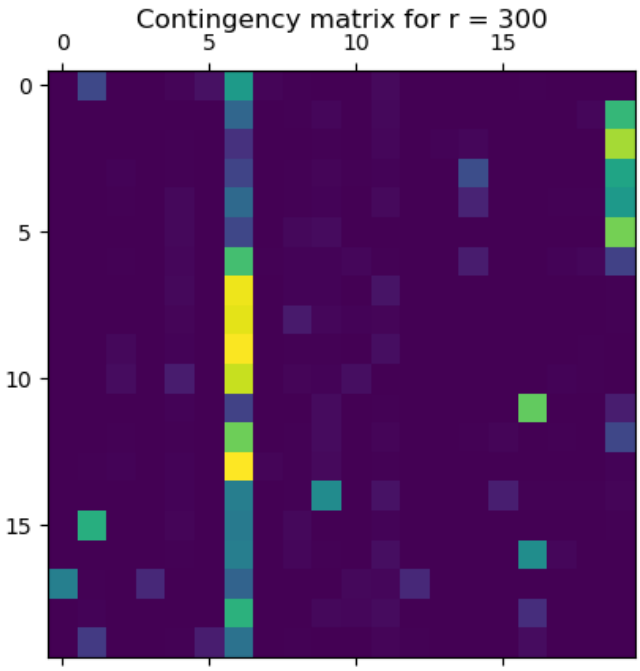

Figure 32: Contingency matrix for r=300

12.1.3 Best value of r

After running the program and looking at all the plots, it was seen the $r = 10$ resulted in the best metric values. For all the measures, $r = 10$ gave the highest

score.

12.2 Visualizing for the best r

Now we'll visualize how the clustering result looks like by mapping the clustered data onto two dimensional plane.

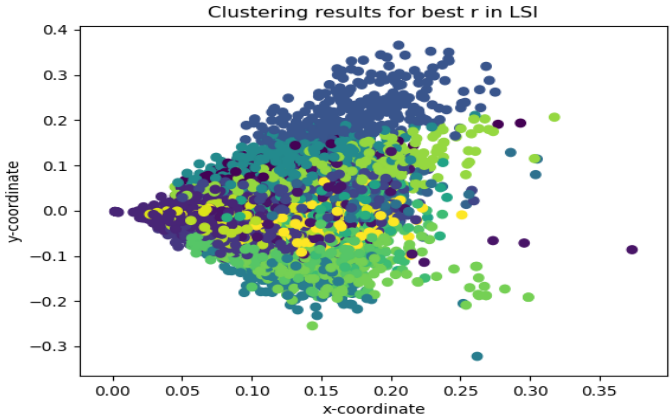

Figure 33: Clustering results for the best r in LSI

Inference We can see 20 different clusters but there is some overlap especially in the middle. This is due to the fact that we are trying to visualize the data that we predicted in 10 dimensions onto 2 dimensions. Near the top and at the sides we can see some good distinct clustering.

12.3 Applying normalization

Next we try to experiment with transformations on the data. The features are normalized to obtain zero mean and unit variance. Then clustering into 20 clusters is done based on this matrix that we obtain after normalization. The clustering results after normalizing are shown below.

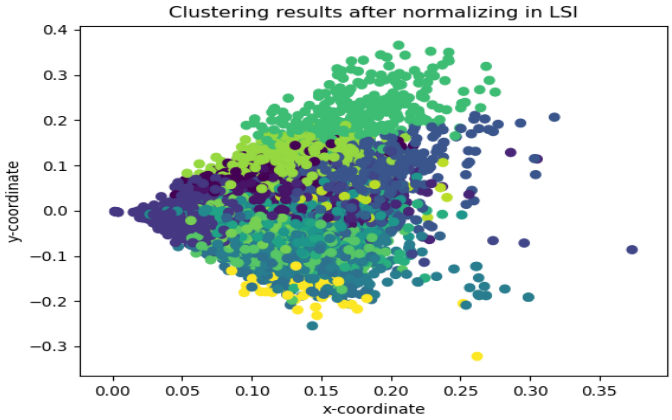

Figure 34: Clustering results after normalizing in LSI

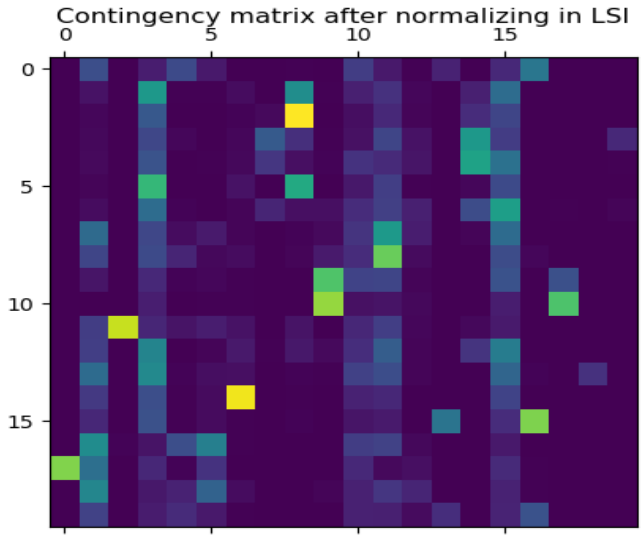

Figure 35: Contingency matrix after normalizing in LSI

	LSI with normalization
Homogeneity score	0.2944
Completeness score	0.3256
V-measure	0.3092
Adjusted Rand score	0.1199
Adjusted mutual info score	0.2921

12.3.1 Inference

In the case of 20 classes with 20 clusters, the purity measures are around the 30 percentage mark. This remains around the same with and without normalization. Unlike the case of 2 clusters where the measures were around 70 percentage, here with 20 clusters, the clustering results are not that great. This is expected since we are trying to identify more clusters and the probability of mis-clustering increases exponentially.

13 20 Classes with NMF

Similar to what was done in the previous section, we expand the dataset to the entire 20 Newsgroups dataset and experiment with different representations and transformations to analyze the effects. In this section, we use the matrix that is obtained after using NMF for dimensionality reduction.

13.1 Finding the best r

Once again, we'll try to find the best r in the same range that was used in the previous case. However, here we have excluded 300 from the range due to computational time and the observations from previous sections that the best r is usually a smaller number. The following are the accuracy measures and contingency matrices that were obtained.

13.1.1 Purity measures

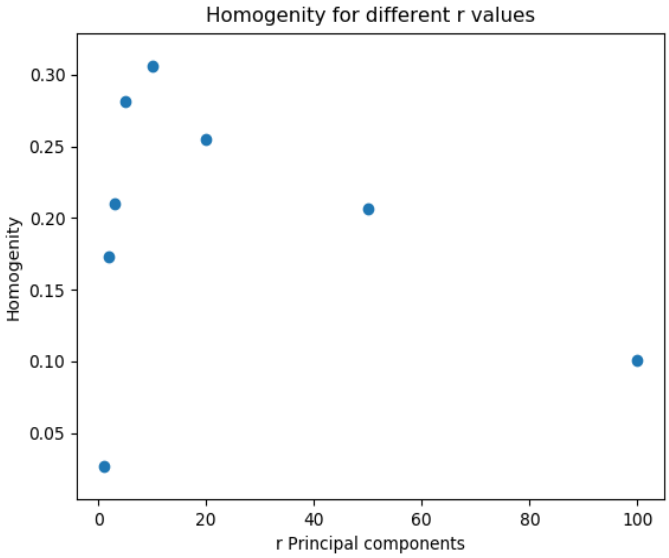

Figure 36: Plot of Homogeneity for different values of r in NMF

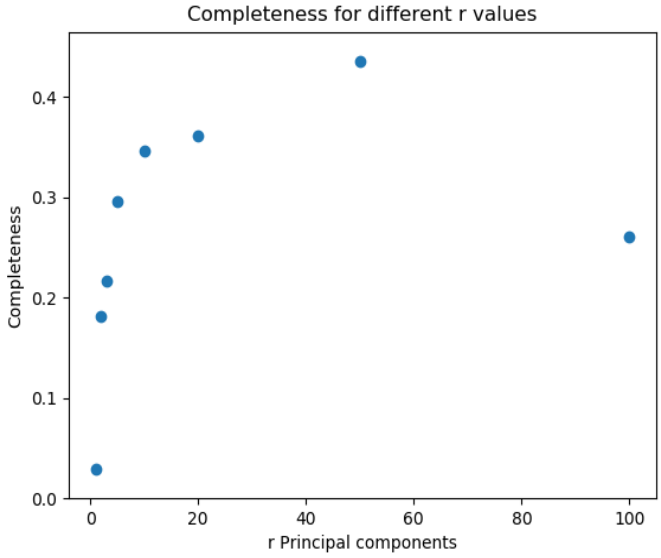

Figure 37: Plot of completeness for different values of r in NMF

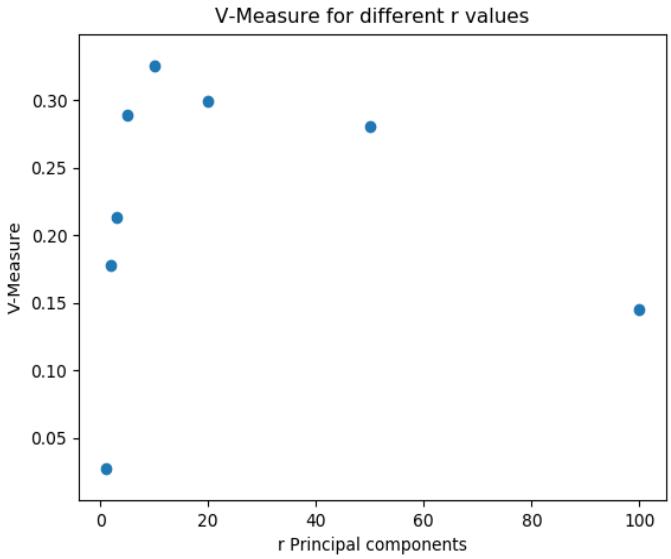

Figure 38: Plot of V-measure for different values of r in NMF

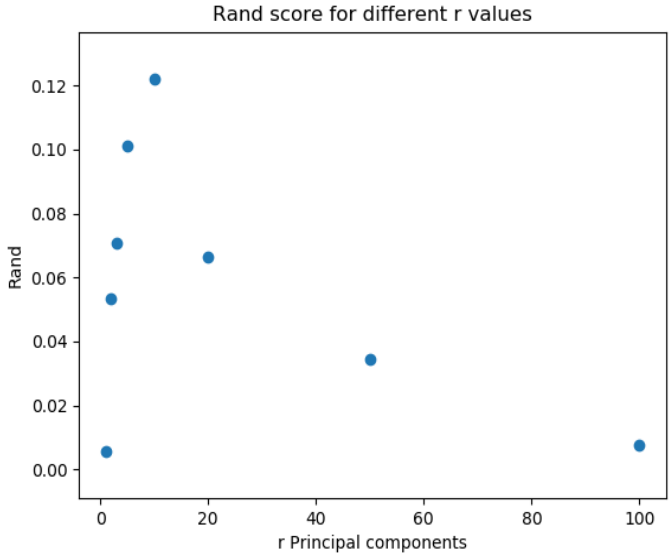

Figure 39: Plot of Rand for different values of r in NMF

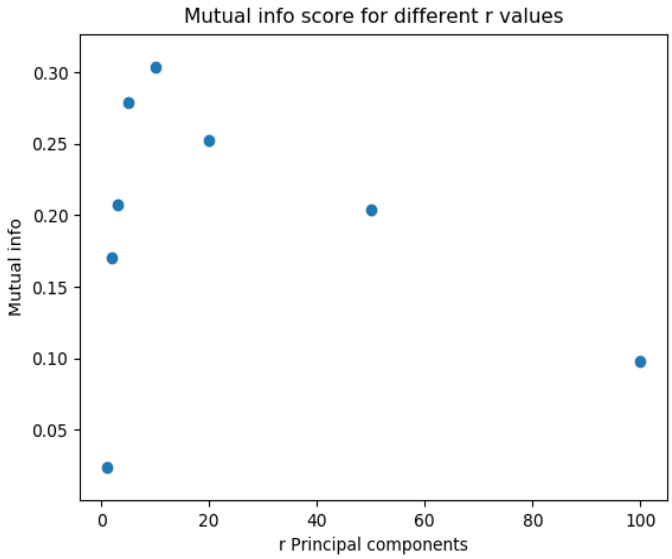

Figure 40: Plot of mutual info score for different values of r in NMF

13.1.2 Contingency matrices

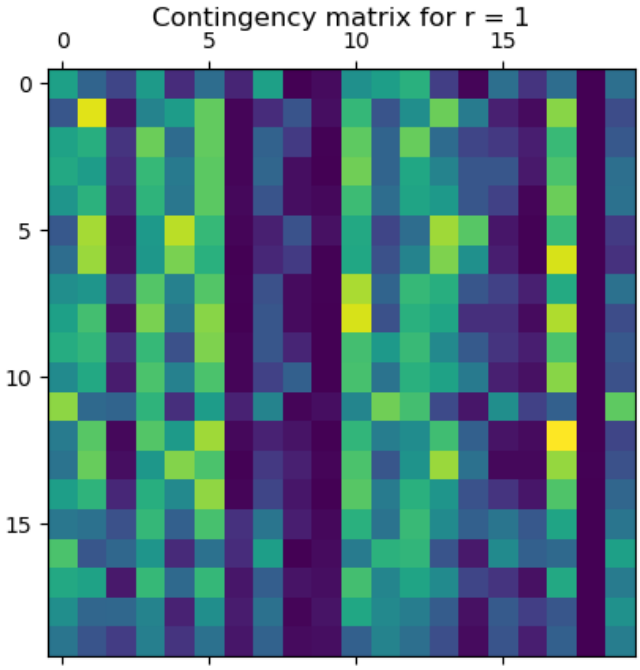

Figure 41: Contingency matrix for r=1

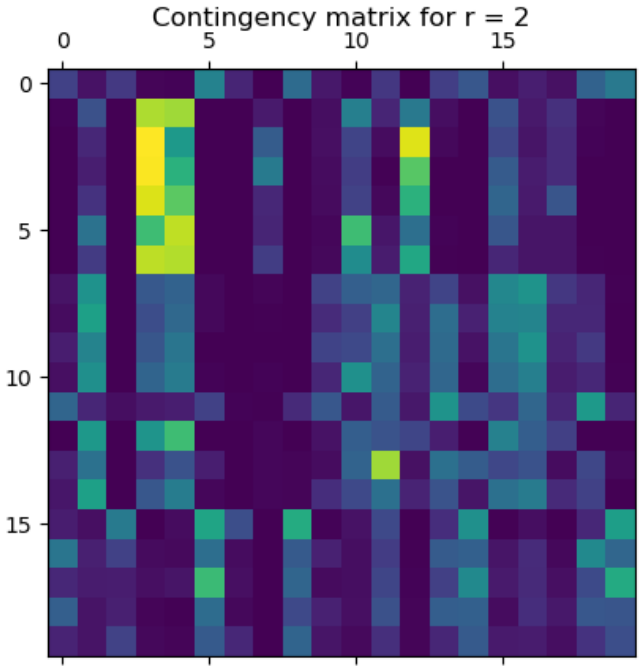

Figure 42: Contingency matrix for r=2

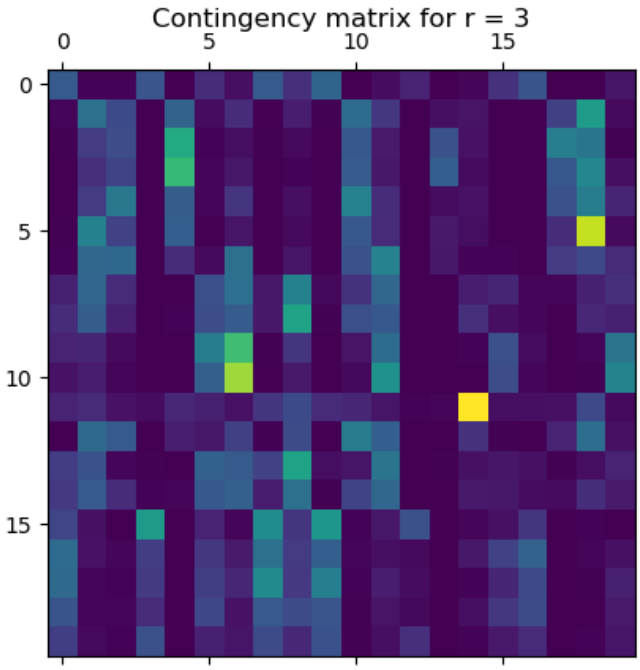

Figure 43: Contingency matrix for r=3

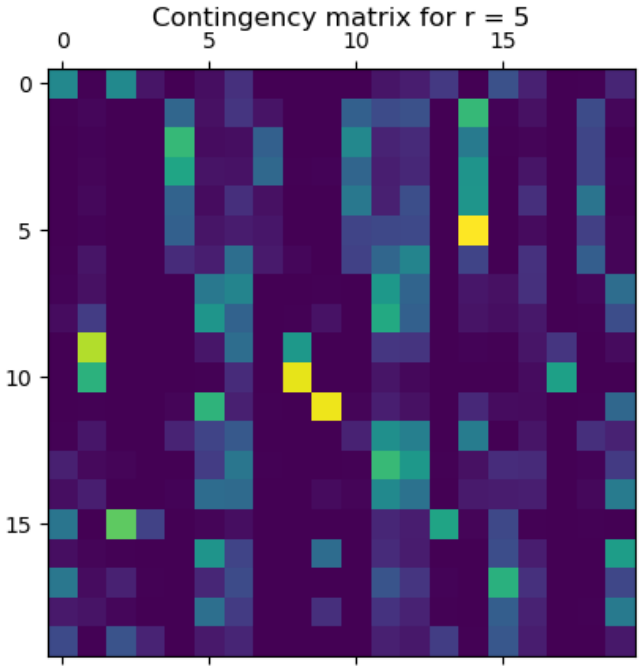

Figure 44: Contingency matrix for r=5

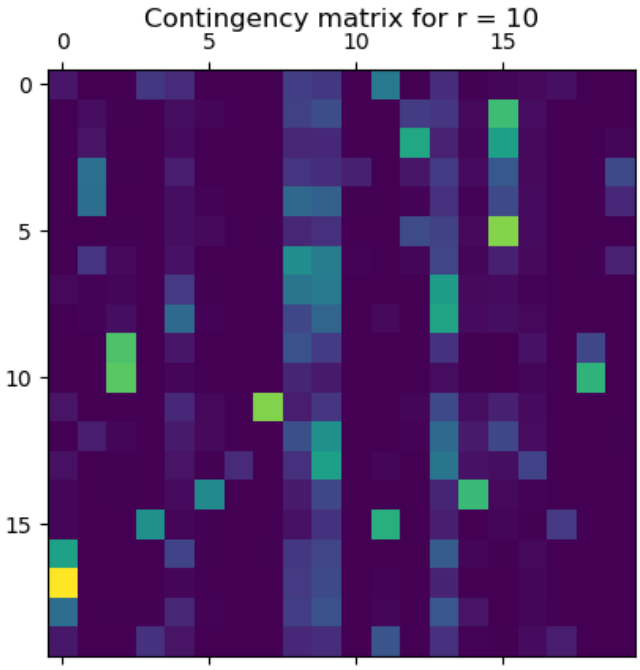

Figure 45: Contingency matrix for r=10

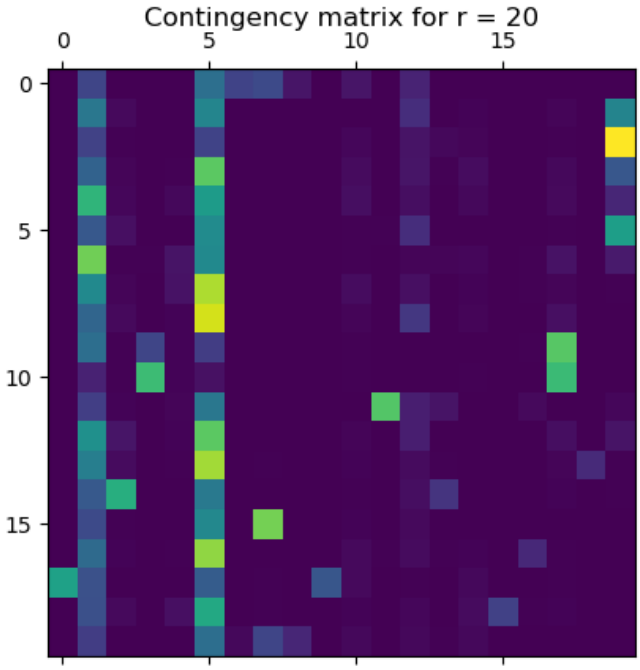

Figure 46: Contingency matrix for r=20

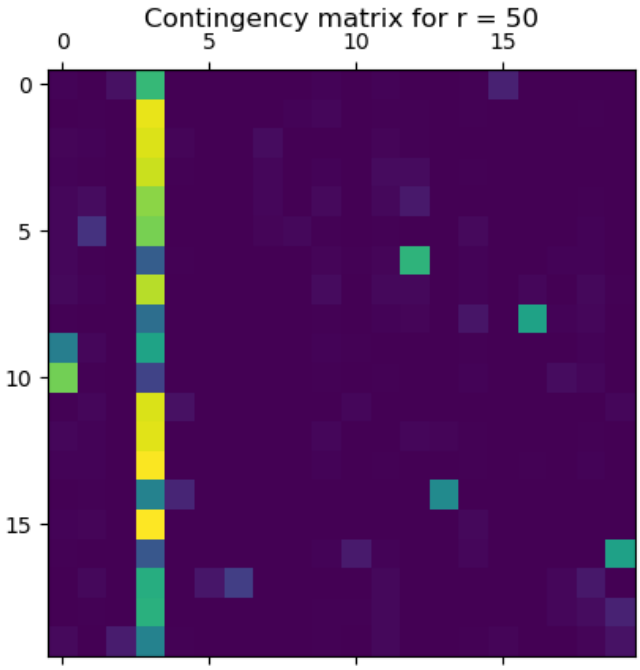

Figure 47: Contingency matrix for r=50

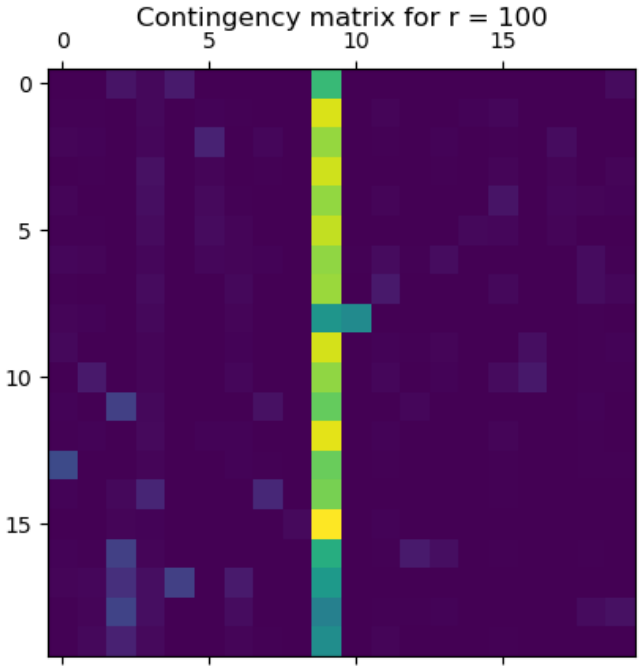

Figure 48: Contingency matrix for r=100

13.1.3 Best value of r

After running the program and looking at all the plots, it was seen the $r = 10$ resulted in the best metric values. For all the measures except completeness, $r = 10$ gave the highest score.

13.2 Visualizing for the best r

Now we'll visualize how the clustering result looks like by mapping the clustered data onto two dimensional plane.

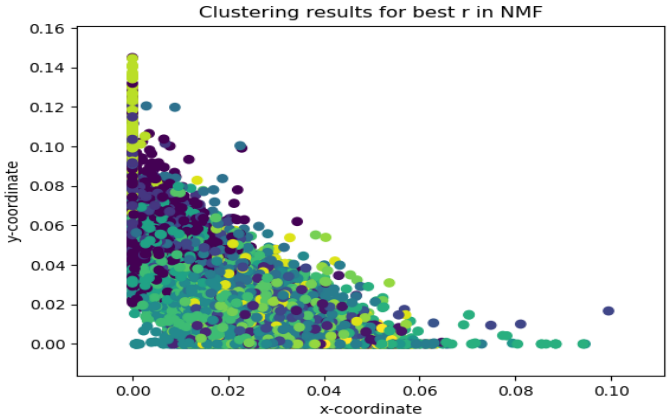

Figure 49: Clustering results for the best r in NMF

13.3 Transformations

Now let's apply a few transformations on this data to see how the clustering results are affected.

13.3.1 Normalization

The features are normalized to obtain zero mean and unit variance. Then clustering into 20 clusters is done based on this matrix that we obtain after normalization. The clustering results after normalizing are shown below.

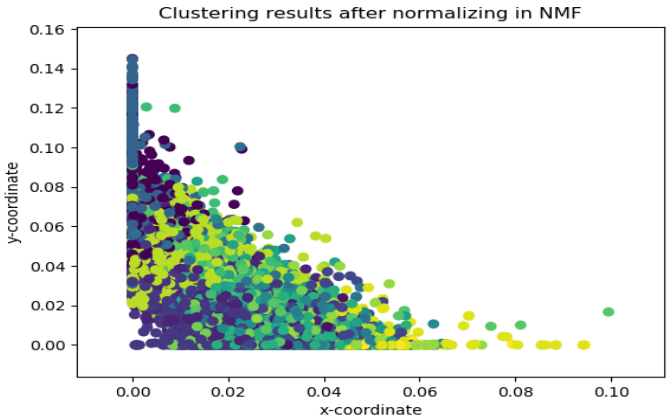

Figure 50: Clustering results after normalizing in NMF

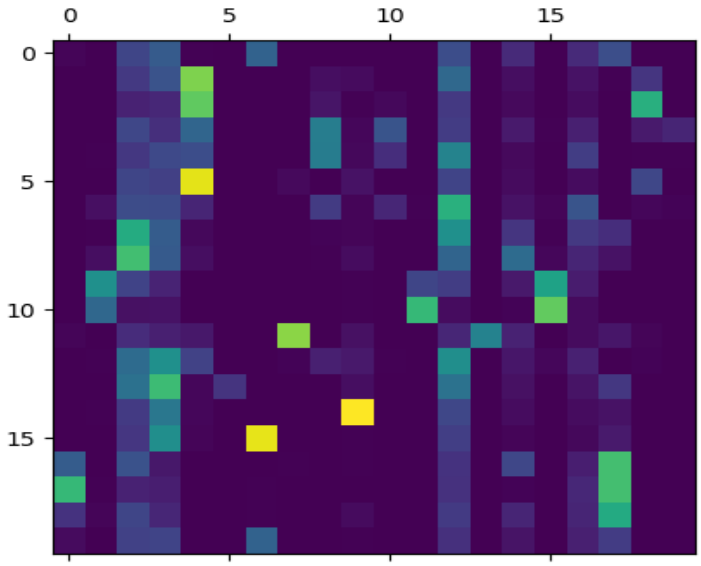

Figure 51: Contingency matrix after normalizing in NMF

	NMF with normalization
Homogeneity score	0.3054
Completeness score	0.3414
V-measure	0.3224
Adjusted Rand score	0.1178
Adjusted mutual info score	0.3031

13.3.2 Log transformation

Log transformation is applied on the dimensionality reduced matrix that we have. Then clustering into 20 clusters is done based on this matrix that we obtain after log transformation. The clustering results after log transform are shown below.

Figure 52: Clustering results after log transformation in NMF

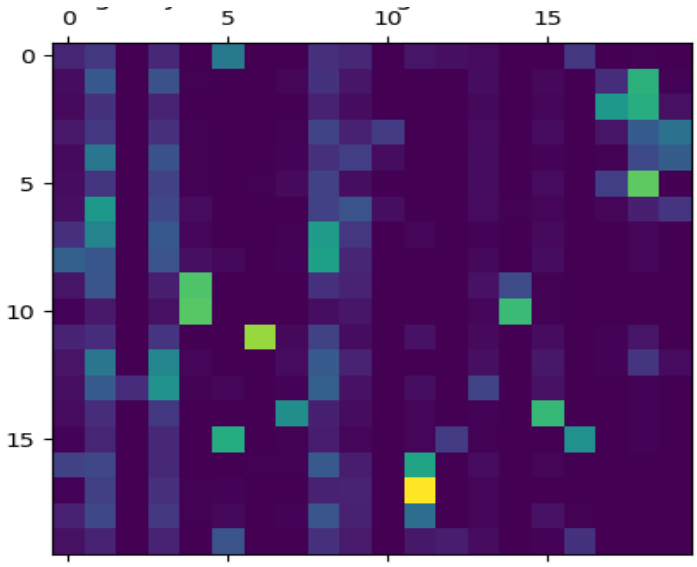

Figure 53: Contingency matrix after log transformation in NMF

	NMF with log transformation
Homogeneity score	0.3002
Completeness score	0.3322
V-measure	0.3158
Adjusted Rand score	0.1227
Adjusted mutual info score	0.2979

13.3.3 Normalization and then log transformation

The features are normalized to obtain zero mean and unit variance. After normalization, log transformation is applied. Then clustering into 20 clusters is done based on this matrix that we obtain. The clustering results after are shown below.

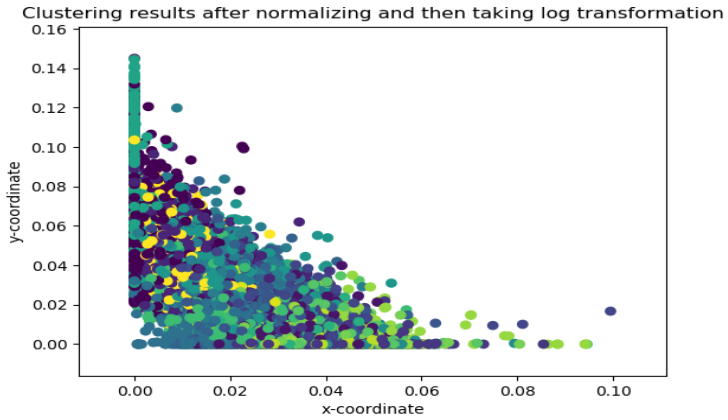

Figure 54: Clustering results after normalizing and log transformation in NMF

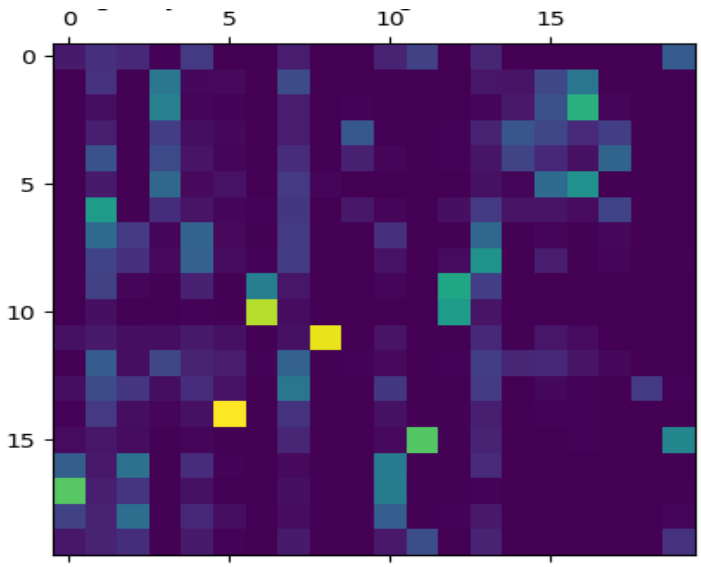

Figure 55: Contingency matrix after normalizing and log transformation in NMF

	NMF with normalization and log transform
Homogeneity score	0.3415
Completeness score	0.3556
V-measure	0.3484
Adjusted Rand score	0.1592
Adjusted mutual info score	0.3394

13.3.4 Log transformation and normalization

Log transformation is applied on the dimensionality reduced matrix that we have. After that the data is normalized to have zero mean and unit variance. Then clustering into 20 clusters is done based on this matrix that we obtain. The clustering results after are shown below.

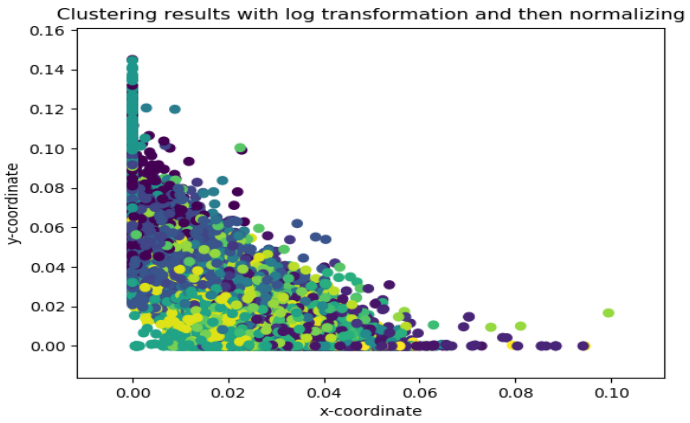

Figure 56: Clustering results after log transformation and normalization in NMF

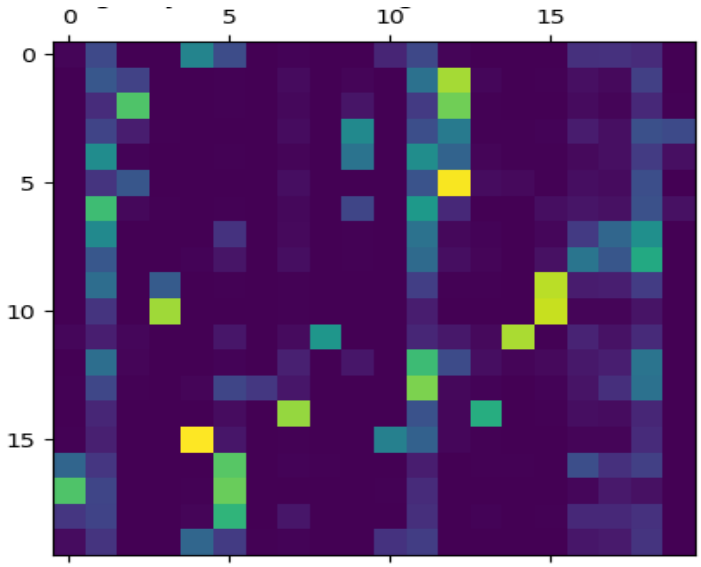

Figure 57: Contingency matrix after log transformation and normalization in NMF

	NMF with log transformation and normalization
Homogeneity score	0.3157
Completeness score	0.3493
V-measure	0.3317
Adjusted Rand score	0.1242
Adjusted mutual info score	0.3145

13.3.5 Inference

For all the different types of transformations that we tried, there was no big variation in the clustering results. The clustering measures were close to 30 percentage without any transformation and even after trying different transformations. In the case of 20 clusters, it has become hard to improve the clustering results since the number of clusters is large.

14 Conclusion

In this project, we have applied clustering, which is an unsupervised learning technique to cluster the 20 Newsgroups dataset. We learned how to represent the how dimensional data in the best number of dimensions to obtain good clustering results. We also experimented with different transformations to understand the effect they had on clustering results. We extended all these ideas from lower number of classes to the entire dataset and analyzed the results obtained.