Clustering

University of California, Los Angeles
ECE 219 Project 2

Joshua Hannan (UID: 805221851)
James (Zach) Harris (UID: 105221897)
Dennis Wang (UID: 105229662)
Akash Deep Singh (UID: 405228294)

`jhannan@g.ucla.edu`
`jzharris@g.ucla.edu`
`dmwang626@g.ucla.edu`
`akashdeepsingh@g.ucla.edu`

Introduction

In the last project, we did classification of a large data set using supervised learning algorithms. However, not all datasets are well labeled (especially the ones that are old and predate the computer itself). When class labels are not available, unsupervised learning methods provide a great way of finding similarities and dissimilarities between different documents.

In this project, we use K-means clustering which is a very popular unsupervised learning algorithm for clustering.

Apart from clustering, we analyze the data with a metric known as TF-IDF to process the words into words that have more significance to us. We further process the data by performing dimensionality reduction to circumvent the Curse of Dimensionality. We repeat this for different values of r and report the results.

Finally, we expand the dataset to 20 categories and repeat all the steps again and report the results.

Getting familiar with the dataset

As with the last project, the data set we will be working with in this project is the "20 Newsgroups" data set. We first analyze 8 of the 20 newsgroups, described below, and then we move on cluster the documents from all 20 newsgroups.

The 8 newsgroups we will analyze first are:

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

The spread of these 8 categories is shown in Fig. 1

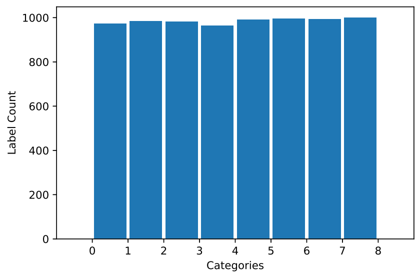


Fig. 1. Spread for 8 news groups from the dataset

Now, we combine them into 2 well-separable classes as shown in table 1:

Class 1	**Class2**
comp.sys.ibm.pc.hardware	rec.autos
comp.graphics	rec.motorcycles
comp.sys.mac.hardware	rec.sport.baseball
comp.os.ms-windows.misc	rec.sport.hockey

Table 1. Two well-separated classes

Now, we take the documents present in these two classes and perform unsupervised clustering on them to find if it is possible to separate the documents in 2 distinct classes without the knowledge of their labels.

1 Building the TF-IDF Matrix

The frequency of each word used in the document is analyzed using the "Term Frequency-Inverse Document Frequency (TF-IDF)" metric. This metric measures the number of times a word is used in a single document, normalized over a function of how many times the word appears in the entire corpus. Therefore the TF-IDF will attempt to quantify which words are unique for each document. The most unique words can be used to identify that document. If enough documents have the same set of words, it is straightforward to conclude that they are of the same class of document. Classification of documents can be performed in this way. The specific equations used in this project are

$$\text{tf-idf}(d,t) = \text{tf}(t,d) * \text{idf}(t)$$

$$\text{idf}(t) = \log\left(\frac{n+1}{\text{df}(t)+1}\right) + 1$$

Where $\text{tf}(t,d)$ is the term frequency of term t in document d and $\text{idf}(t)$ is the inverse document frequency of term t across the corpus. The inverse document frequency metric is calculated by taking the logarithm of the total number of documents, n, divided by the number of documents that contain term t, $\text{df}(t)$. Intuitively, $\text{idf}(t)$ gives larger weight to unique terms and smaller weight to terms that are frequent among many documents. A trailing 1 is added to the terms in the quotient of $\text{idf}(t)$ to prevent zero division from occurring.

Question 1 The following specs were used during transformation of the document into TF-IDF vectors:

– Use the "english" stopwords of the CountVectorizer

- Use min_df = 3 for the CountVectorizer

The resulting dimensions of the TF-IDF matrix are:

(7882, 27768)

2 K-Means Clustering

K-means clustering is a popular clustering algorithm used in data mining. It works best when the data is not labeled and is used as an unsupervised technique to group similar data points into clusters.

Each data point belongs to only one cluster and hence the 'K' in K-means stands for the total number of clusters that you want to group your data into. In its essence, K-means performs the following two tasks repeatedly until convergence:

1. Calculate the distance of the data point to the centers of all the K clusters and assign/reassign it to the cluster whose center is the closest.
2. After all the data points are assigned a cluster, recalculate the position of the center of each cluster. The center should be the mean of all the data points in the cluster.

Initially, the centers are initialized randomly. Since this algorithm continuously iterates over the data points until convergence, it takes a long time to finish.

Mathematically, the algorithm works as follows [1]:

If we define μ_k as the center of the k^{th} cluster, then

$$r_{nk} = \begin{cases} 1, & \text{if } x_n \text{is assigned to cluster k} \\ 0, & \text{otherwise} \end{cases} (n = 1, ..., N)(k = 1, ..., K) \quad (1)$$

where K is the total number of clusters, N is the total number of data points in the dataset and x_n is the n^{th} point in the dataset.

Then, our goal is find r_{nk}'s and μ_k's such that they minimize,

$$J = \sum_{n=1}^{N} \sum_{k=1}^{K} r_{nk} ||x_n - \mu_k||^2 \quad (2)$$

In this portion of the project, we apply K-means clustering using the **Kmeans** class from **sklearn** library with the following parameters:

- k = 2
- random_state = 0
- max_iter ≥ 1000
- n_init ≥ 30

2.1 Contingency Matrix

"Contingency matrix reports the intersection cardinality for every true/predicted cluster pair. The contingency matrix provides sufficient statistics for all clustering metrics where the samples are independent and identically distributed and one doesn't need to account for some instances not being clustered."

— https://scikit-learn.org/stable/modules/clustering.html#contingency-matrix

In simple terms, if we have 2 classes, the contingency matrix A can be used to find out how many data points predicted by K-means clustering actually belong to their true class and how many don't. In the matrix A, A_{ij} represents the number of data points that belong to the Class C_i and Cluster K_j.

Question 2 The contingency matrix for K-means clustering with $k = 2$ is reported below in figure 2:

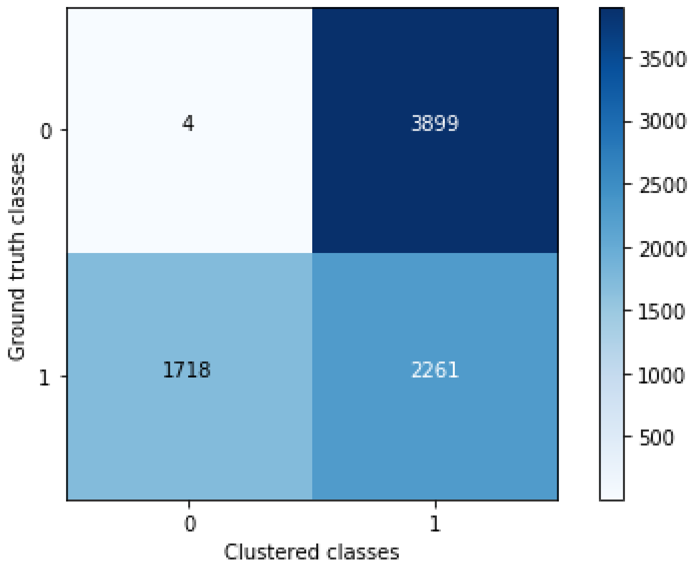


Fig. 2. Contingency matrix for K-means clustering

Question 3 Evaluation of clustering results is done using the parameters **Homogeneity Score**, **Completeness Score**, **V-measure**, **Adjusted Rand Index (Rand Index)**, and **Adjusted Mutual Information Score (Mutual Info)**. These results for our K-means clustering with $k = 2$ are reported in table 2.

Homogeneity:	0.2536
Completeness:	0.3348
V-measure:	0.2886
Rand Index:	0.1808
Mutual Info:	0.2535

Table 2. Reporting K-means clustering results for k=2

3 Dimensionality Reduction

It is often the case that clustering does not yield good results when our data points are very high-dimensional. One major reason is that the Euclidean distance, which the K-means clustering algorithm relies on, is not a good metric in higher dimensions. Also, since K-means uses the Euclidean distance, it naturally tends to cluster data that is already grouped in balls, as opposed to any other possible shape. Thus we want to reduce the data to a dimension that will hopefully transform the data into round-shaped clusters to best allow the K-means clustering algorithm to find potential centroids.

3.1 Explained Variance Ratio

Question 4 Figure 3 shows the percentage the top r principle components can explain, up to $r = 1000$. The slope in the graph is steepest for low r's; this is because in SVD, the first few principle components give the most information about the original matrix. Thus it makes sense that as r increases, we gain less and less information. In fact, in some cases it can be detrimental to our classification scheme to keep too many principle components, as described in the following section.

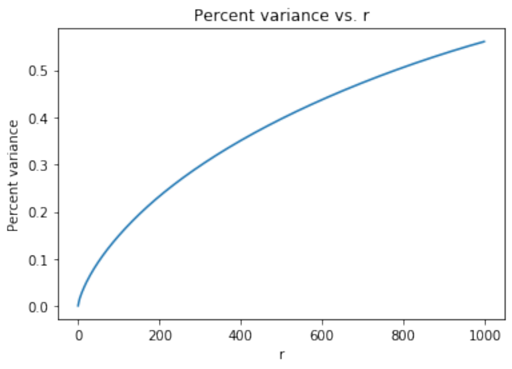


Fig. 3. Percent of variance for top r principle components

3.2 Effects of Dimensionality Reduction

Question 5 Tables 3 and 4 show the different scores when we reduced our data using SVD and NMF, respectively, for varying values of r. From these tables, along with figures 4 and 5, we see that there is a clear spike in each of the scores when $\mathbf{r = 2}$, allowing us to determine that this is the "best" number of components for both SVD and NMF reduced data.

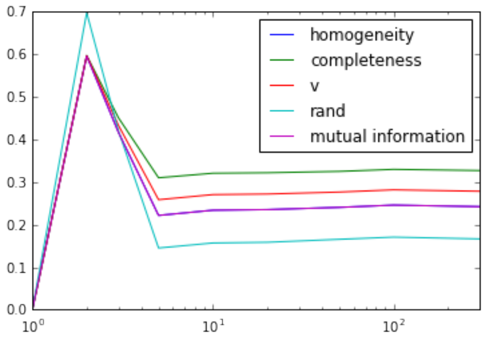


Fig. 4. Metrics for **SVD**-reduced data

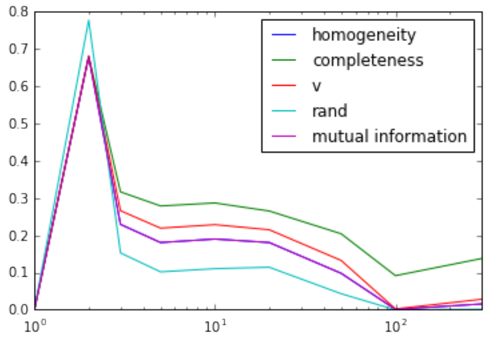


Fig. 5. Metrics for **NMF**-reduced data

$r =$	1	2	3	5	10	20	50	100	300
Homogeneity:	0.0003	0.5959	0.4164	0.2127	0.2339	0.2353	0.2405	0.2459	0.2423
Completeness:	0.0003	0.5970	0.4506	0.3100	0.3207	0.3218	0.3249	0.3297	0.3270
V-measure:	0.0003	0.5965	0.4329	0.2585	0.2705	0.2718	0.2765	0.2817	0.2784
Rand Index:	0.0003	0.6973	0.4199	0.1452	0.1570	0.1586	0.1656	0.1708	0.1666
Mutual Information:	0.0002	0.5959	0.4164	0.2216	0.2338	0.2353	0.2405	0.2458	0.2422

Table 3. Scores for varying number of principle components in SVD

$r =$	1	2	3	5	10	20	50	100	300
Homogeneity:	0.0003	0.679	0.2293	0.1806	0.1900	0.1806	0.0982	0.0013	0.0159
Completeness:	0.0003	0.6801	0.3165	0.2787	0.2867	0.2651	0.2042	0.0917	0.1377
V-measure:	0.0003	0.6796	0.2660	0.2192	0.2285	0.2149	0.1326	0.0025	0.0285
Rand Index:	0.0003	0.7770	0.1528	0.1020	0.1106	0.1143	0.0434	0.0001	0.0016
Mutual Information:	0.0002	0.6790	0.2293	0.1806	0.1899	0.1806	0.0981	0.0012	0.0158

Table 4. Scores for varying number of principle components in NMF

Question 6 It is evident that as r moves away from 1, the scores become larger. They rise until a specific point, at which time they change direction and either oscillate or decrease in value. This non-monotonic behavior of the measures as r increases can be explained by a trade-off between the dimension reduction algorithm and K-means algorithm.

By having too few dimensions to represent the data with, the K-means clustering algorithm cannot make accurate clusters around the data. K-means requires there to be enough valuable information about the corpus to cluster with. K-means therefore will improve at the beginning since necessary information is being introduced. However, as more dimensions are added, the K-means clustering algorithm must work with a higher dimensional space. K-means uses Euclidean distances, as was shown by equation 2. The Euclidean distance has less effect when in a higher dimensional space. This phenomenon is otherwise known as the curse of dimensionality. This phenomenon causes there to be a general decrease in scores as r increases in value.

4 Visualization

We visualize the clustering results by plotting the reduced-dimension data points onto a two-dimensional graph. We color the points according to the class labels dictated by clustering methods and several transformation methods. Visualizations are provided for ground truth class labels and clustering labels.

4.1 Clustering Results

Question 7 Figure 6 visualizes the clustering results for SVD with its best r, when $r = 2$ and Figure 7 visualizes clustering results for ground truth class labels.

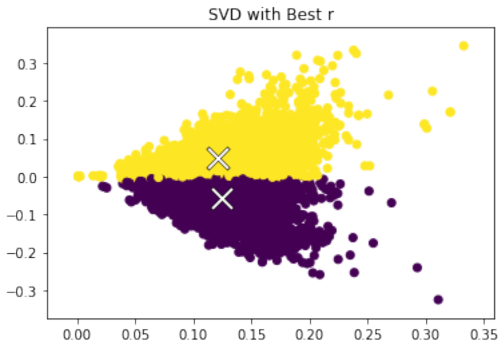


Fig. 6. SVD, $r = 2$.

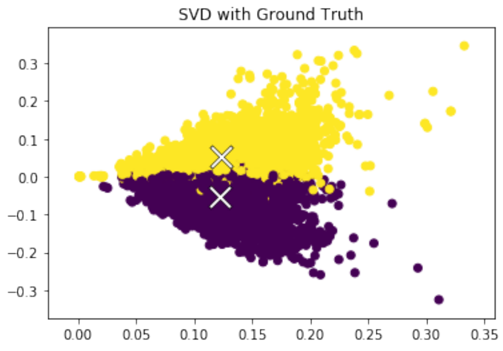


Fig. 7. SVD, Ground Truth

Figure 8 visualizes the contingency table for clustering results for SVD with its best r, when $r = 2$ and Figure 9 visualizes contingency table for clustering results for ground truth class labels.

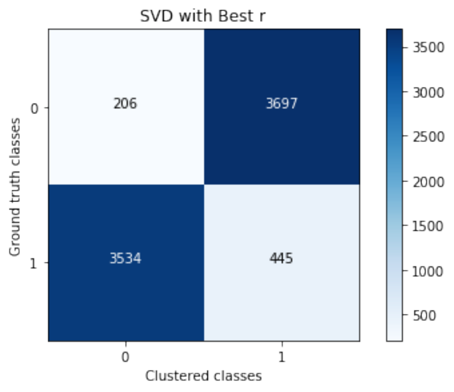


Fig. 8. SVD, $r = 2$.

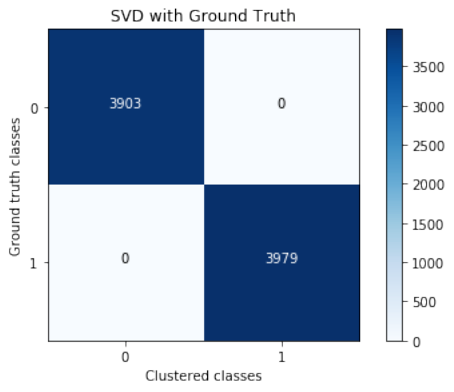


Fig. 9. SVD, Ground Truth

The results for our K-means clustering with $k = 2$ for SVD when $r = 2$ are reported in table 5.

Homogeneity:	0.5917
Completeness:	0.5929
V-measure:	0.5923
Rand Index:	0.6926
Mutual Info:	0.5916

Table 5. Clustering Measures for SVD with $r = 2$

Figure 10 visualizes the clustering results for NMF with its best r, when $r = 2$ and Figure 11 visualizes clustering results for ground truth class labels.

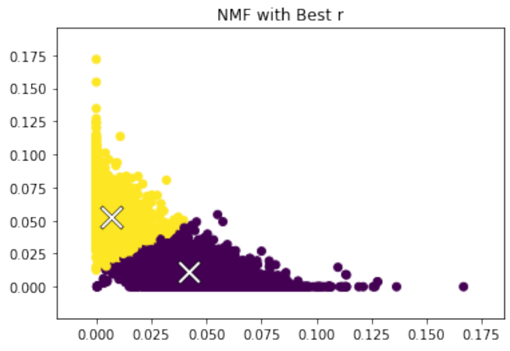


Fig. 10. NMF, $r = 2$.

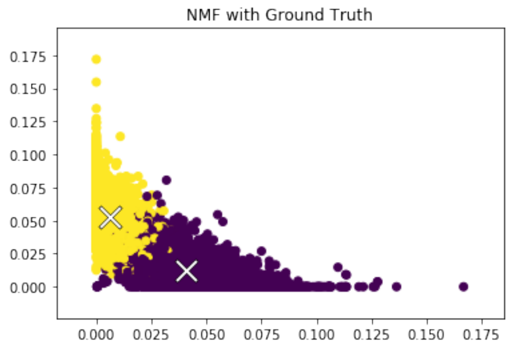
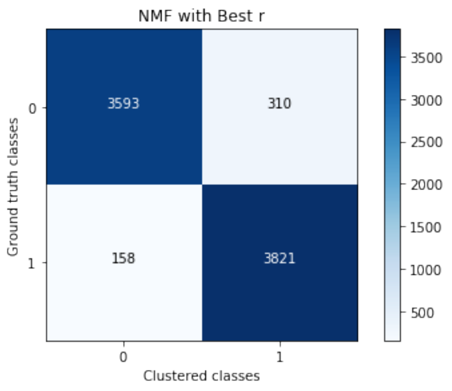


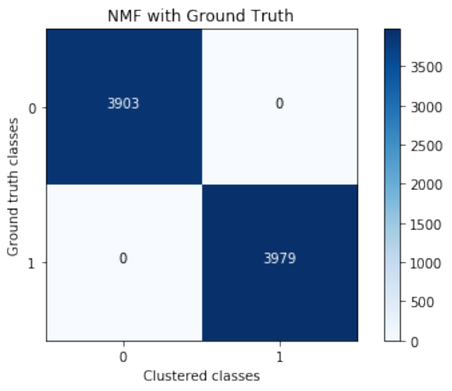
Fig. 11. NMF, Ground Truth

Figure 12 visualizes the contingency table for clustering results for NMF with its best r, when $r = 2$ and Figure 13 visualizes contingency table for clustering results for ground truth class labels.



	0	1
0	3593	310
1	158	3821

Fig. 12. NMF, $r = 2$.



	0	1
0	3903	0
1	0	3979

Fig. 13. NMF, Ground Truth

The results for our K-means clustering with $k = 2$ for NMF when $r = 2$ are reported in table 6.

Homogeneity:	0.679
Completeness:	0.6801
V-measure:	0.6796
Rand Index:	0.777
Mutual Info:	0.679

Table 6. Clustering Measures for NMF with $r = 2$

4.2 Data Transformation Methods

We perform eight transformation methods on SVD-reduced data and NMF-reduced data using the best r values from previous parts.

We implement a unit variance scaling transform, such that each feature has unit variance (each column of the reduced-dimensional data matrix has unit variance). The following is a code snippet that shows how we implemented the unit variance scaling transform:

```python
# Scaling features such that each feature has unit variance

def scaling (features): # Each column of the reduced dimension
↪  data has unit variance
return (features - np.mean(features,axis=0)) / np.std(features,
↪  axis=0)
```

We also apply the following non-linear logarithm transformation on the data:

$$\mathbf{f(x)} = \mathbf{sign(x)} \cdot (\log(|\mathbf{x}| + c) - \log(c)),\ \ (\mathbf{sign(x)})_i \equiv \begin{cases} -1 & \text{if } x_i < 0 \\ 0 & \text{if } x_i = 0 \\ 1 & \text{if } x_i > 0 \end{cases} \tag{3}$$

where $c = 0.01$.

The following is a code snippet that shows how we implemented the logarithmic transform:

```python
# Logarithm transformation

def logarithm_trans (features):

c_const = 0.01
return np.sign(features) * (np.log(np.abs(features) + c_const) -
↪  np.log(c_const))
```

We implemented the following four transformations on both the SVD-reduced data and the NMF-reduced data:

1. unit variance scaling transformation
2. logarithmic transformation
3. logarithmic transformation then scaling transformation
4. the scaling transformation then logarithmic transformation

Question 8 The following visualizations represent the aforementioned transformations on both the SVD-reduced data and the NMF-reduced data.

Transformations on SVD-Reduced Data:

Figure 14 visualizes the clustering results for SVD with unit variance scaling.

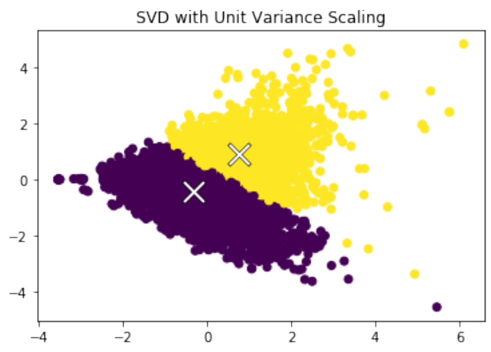


Fig. 14. SVD, Unit Variance Scaling

Figure 15 visualizes the clustering results for SVD with logarithmic transformation and Figure 16 visualizes clustering results for ground truth class labels with logarithmic transformation.

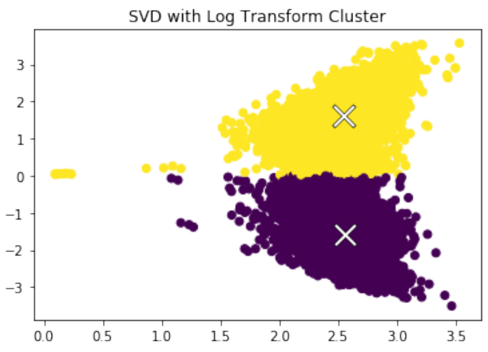


Fig. 15. SVD, Logarithmic Transformation

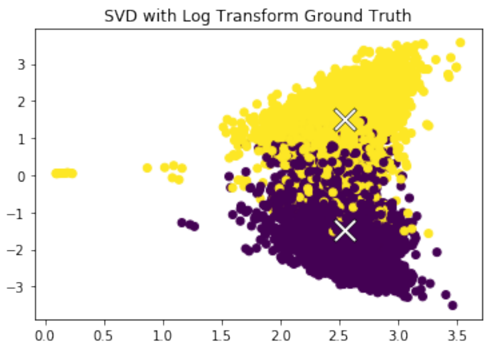


Fig. 16. SVD, Logarithmic Transformation, Ground Truth

Figure 17 visualizes the clustering results for SVD with logarithmic transformation then unit variance scaling and Figure 18 visualizes clustering results for ground truth class labels with logarithmic transformation then unit variance scaling.

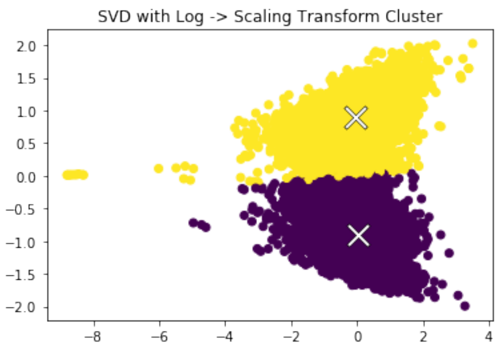


Fig. 17. SVD, Logarithmic then Scaling

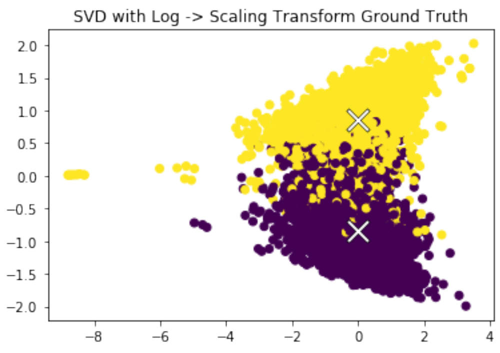


Fig. 18. SVD, Logarithmic then Scaling, Ground Truth

Figure 19 visualizes the clustering results for SVD with unit variance scaling then logarithmic transformation and Figure 20 visualizes clustering results for ground truth class labels with unit variance scaling then logarithmic transformation.

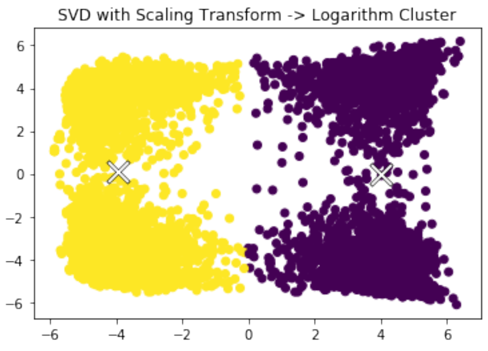


Fig. 19. SVD, Scaling then Logarithmic

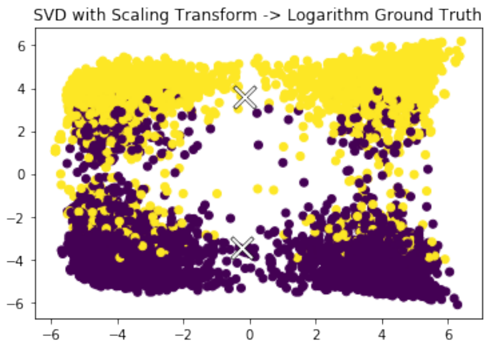


Fig. 20. SVD, Scaling then Logarithmic Ground Truth

Transformations on NMF-Reduced Data:

Figure 21 visualizes the clustering results for NMF with unit variance scaling.

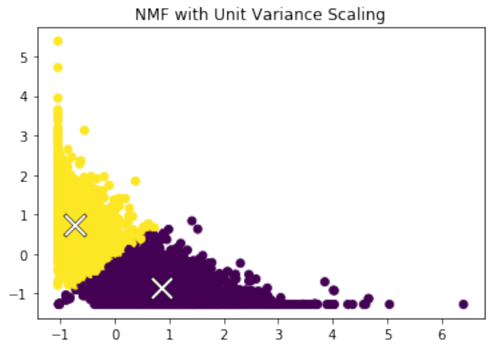


Fig. 21. NMF, Unit Variance Scaling

Figure 22 visualizes the clustering results for NMF with logarithmic transformation and Figure 16 visualizes clustering results for ground truth class labels with logarithmic transformation.

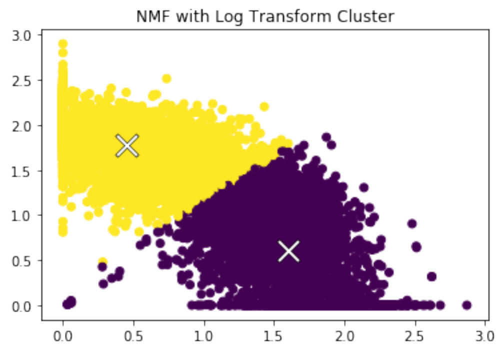


Fig. 22. NMF, Logarithmic Transformation

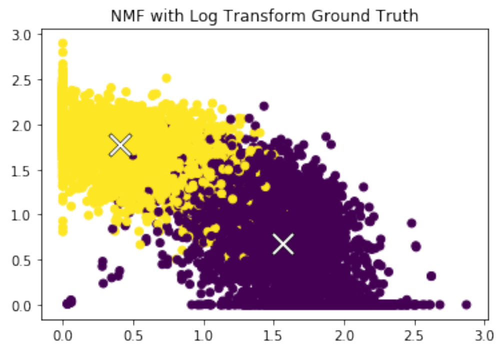


Fig. 23. NMF, Logarithmic Transformation, Ground Truth

Figure 24 visualizes the clustering results for NMF with logarithmic transformation then unit variance scaling and Figure 25 visualizes clustering results for ground truth class labels with logarithmic transformation then unit variance scaling.

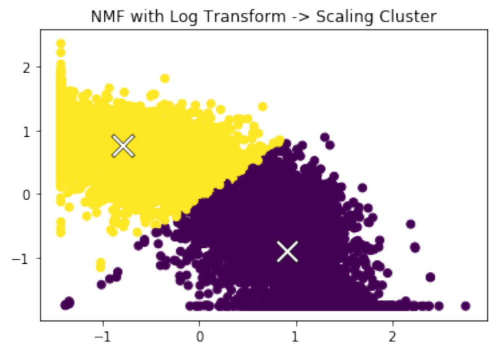


Fig. 24. NMF, Logarithmic then Scaling

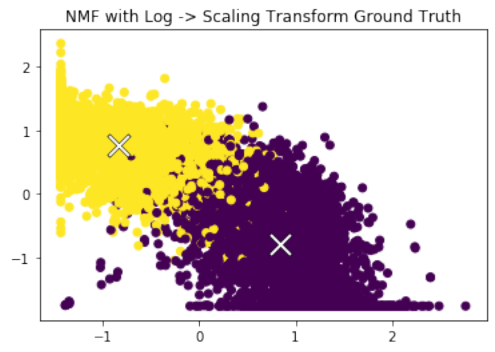


Fig. 25. NMF, Logarithmic then Scaling, Ground Truth

Figure 58 visualizes the clustering results for NMF with unit variance scaling then logarithmic transformation and Figure 59 visualizes clustering results for ground truth class labels with unit variance scaling then logarithmic transformation.

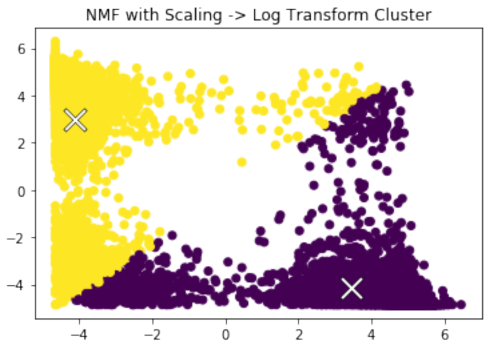


Fig. 26. NMF, Scaling then Logarithmic

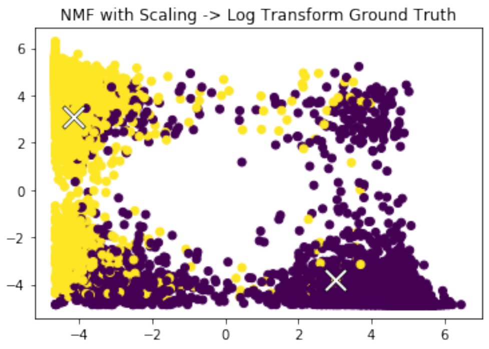


Fig. 27. NMF, Scaling then Logarithmic Ground Truth

Question 9 The aforementioned logarithmic transformation can be implemented to make highly skewed data distributions less skewed. Unlike a linear scale for example, which looks at the differences between data values (the change from 2 to 3 is considered the same as the change from 5 to 6), logarithmic scales analyze the change between values based on their ratio (the change from 1 to 2 is considered the same as the change from 4 to 8). Consequently, **a logarithmic transformation "squeezes" together larger data values and "stretches" smaller data values.** Thus, **logarithm transformations can be useful when clustering data that shows outliers at the high end.** In general, logarithmic transformations can be implemented to spread out clumped data and to make skewed data symmetric. **Logarithmic transformation is particularly useful for data with strongly skewed positive values or data with a relationship that is close to exponential.**

Question 10 The following are the five clustering measures for the clustering results of the transformed data for both the SVD and NMF:

The results for our K-means clustering with $k = 2$ for SVD with unit variance scaling are reported in table 7.

Homogeneity:	0.2274
Completeness:	0.2569
V-measure:	0.2413
Rand Index:	0.244
Mutual Info:	0.2274

Table 7. Clustering Measures for SVD with unit variance scaling

The results for our K-means clustering with $k = 2$ for SVD with logarithm transform are reported in table 8.

Homogeneity:	0.6112
Completeness:	0.6112
V-measure:	0.6112
Rand Index:	0.7182
Mutual Info:	0.6112

Table 8. Clustering Measures for SVD with log transformation

The results for our K-means clustering with $k = 2$ for SVD with logarithm transform then unit variance scaling are reported in table 9.

Homogeneity:	0.6077
Completeness:	0.6077
V-measure:	0.6077
Rand Index:	0.7148
Mutual Info:	0.6077

Table 9. Clustering Measures for SVD with log transformation then unit variance scaling

The results for our K-means clustering with $k = 2$ for SVD with unit variance scaling then logarithm transform are reported in table 10.

Homogeneity:	0.0001
Completeness:	0.0001
V-measure:	0.0001
Rand Index:	0.0
Mutual Info:	0.0

Table 10. Clustering Measures for SVD with unit variance scaling then log transformation

The results for our K-means clustering with $k = 2$ for NMF with unit variance scaling are reported in table 11.

Homogeneity:	0.6828
Completeness:	0.6856
V-measure:	0.6842
Rand Index:	0.7734
Mutual Info:	0.6828

Table 11. Clustering Measures for NMF with unit variance scaling

The results for our K-means clustering with $k = 2$ for NMF with logarithm transform are reported in table 12.

Homogeneity:	0.6757
Completeness:	0.6791
V-measure:	0.6774
Rand Index:	0.765
Mutual Info:	0.6757

Table 12. Clustering Measures for NMF with log transformation

The results for our K-means clustering with $k = 2$ for NMF with logarithm transform then unit variance scaling are reported in table 13.

Homogeneity:	0.6864
Completeness:	0.6891
V-measure:	0.6877
Rand Index:	0.777
Mutual Info:	0.6863

Table 13. Clustering Measures for NMF with log transformation then unit variance scaling

The results for our K-means clustering with $k = 2$ for NMF with unit variance scaling then logarithm transform are reported in table 14.

Homogeneity:	0.69
Completeness:	0.6928
V-measure:	0.6914
Rand Index:	0.7797
Mutual Info:	0.6899

Table 14. Clustering Measures for NMF with unit variance scaling then log transformation

5 Expand Dataset to 20 Categories

The task of binary classification may simplify the results of clustering. How purely can K-means cluster all 20 categories of the news groups datasets? We will answer this question by evaluating the performance of clusters using the best combination of clustering methods previously defined. All twenty categories are defined in the Appendix. The spread of these 20 categories is shown in Fig. 28. Although the spread of all classes may be disproportionate, all of the data is included to produce the best clustering results.

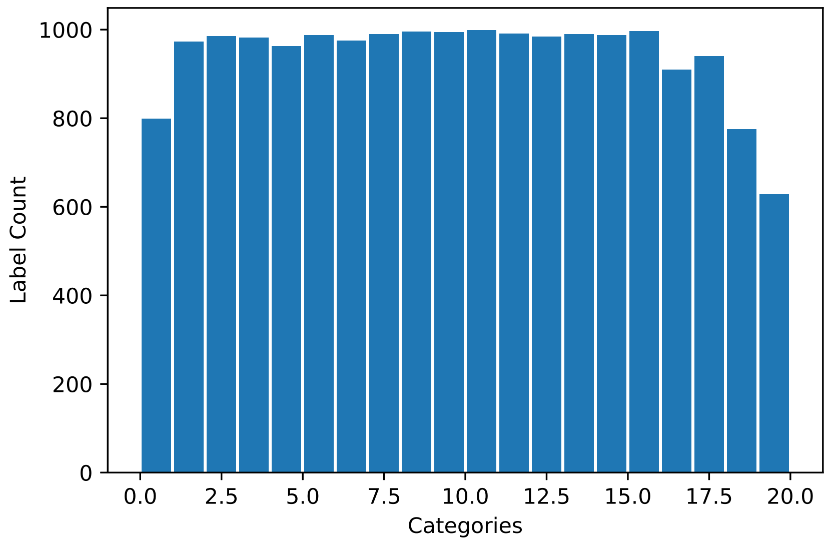


Fig. 28. Spread for 20 news groups from the dataset

Question 11 The following specs were used during transformation of the document into TF-IDF vectors:

– Use the "english" stopwords of the CountVectorizer
– Use min_df = 3 for the CountVectorizer

The resulting dimensions of the TF-IDF matrix are:

(18846, 52295)

Table 15 shows the **scores after clustering was performed**. The measures are far from optimal. The clusters are only able to capture approximately $\frac{2}{5}$ of the data, and the clusters that were used are not homogenous. By reducing and transforming the dimensions of the TF-IDF matrix, we may be able to improve this result.

Homogeneity:	0.3594
Completeness:	0.4511
V-measure:	0.4001
Rand Index:	0.1366
Mutual Information:	0.3573

Table 15. Reporting K-means clustering results for 20 categories

The contingency matrix of this clustering result is another way of analyzing the performance of using the original TF-IDF matrix. Fig. 29 shows the contingency matrix. A perfectly clustered dataset will only have a single element in every row and every column. In comparison, the columns of Fig. 29 show that points from many different classes are mixed into the clusters. The rows of Fig. 29 show that each class contains many different clusters. By viewing the contingency matrix in this way, we can qualitatively verify where the homogeneity and completeness measures need improvement.

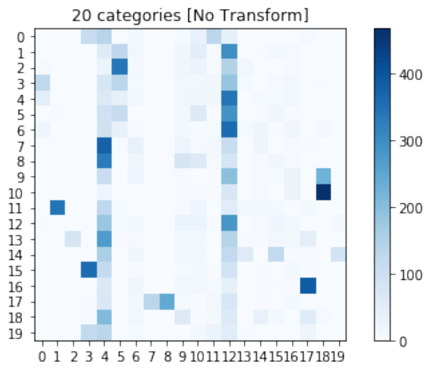


Fig. 29. Contingency matrix from TF-IDF clustering

Question 12 Truncated SVD and NMF were performed on the 20 category dataset. The transformations discussed in Section 4 are applied to the reduced dataset using varying r's. The best r was calculated using the Adjusted Random Index score, which gave a clear indication of when the new dimensions negatively impacted the K-means clustering algorithm. The plots showing all scores for varying r can be found in the Appendix, Section 7.3.

This method was performed for each combination of transformations, and the best combinations are summarized in Table 16. These combinations achieved a higher Adjusted Random Index score than the original TF-IDF matrix, however the improvements are lower than expected. The best improvement from the original scores was by using the SVD dimension reduction technique with $r = 100$ and an Adjusted Random Index score of 0.2327. This improved the original TF-IDF Random Index score by 70%. It is clear from the contingency matrices that the K-means clustering from the best combination is able to successfully cluster $\frac{11}{20}$ of the classes.

Undesirable methods were also explored. Table 17 shows examples of undesirable transformation combinations. The absolute worst result that was achieved was by using the SVD dimension reduction technique with $r = 300$ and an Adjusted Random Index score of 0.0733. This diminished the original TF-IDF Random Index score by 79%. The contingency matrices show that the worst combinations caused the K-means to cluster many classes into a single cluster, which greatly reduced the Homogeneity measures.

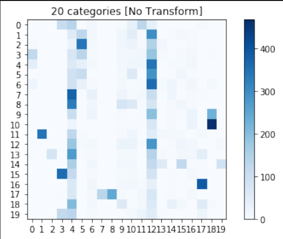
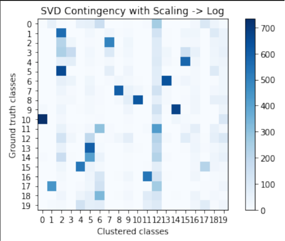
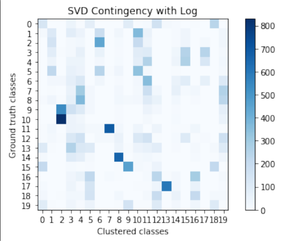
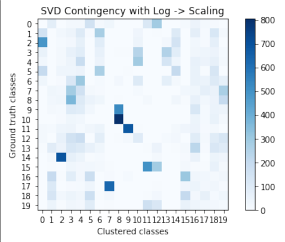
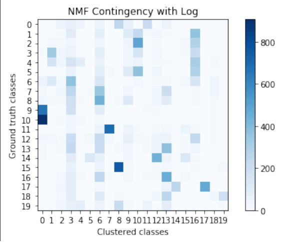
Transforms	Best r	Homogeneity	Completeness	V-measure	Rand	Mutual Information	Contingency Table
TF-IDF	NA	0.3594	0.4511	0.4001	0.1366	0.3573	
SVD + Scale then Log	100	0.4102	0.4369	0.4231	0.2327	0.4083	
SVD + Log	10	0.394	0.4027	0.3983	0.2199	0.392	
SVD + Log then Scale	10	0.3696	0.3759	0.3727	0.2127	0.3675	
NMF + Log	20	0.3606	0.4002	0.3794	0.1945	0.3586	

Table 16. Scores for best combinations of transformations

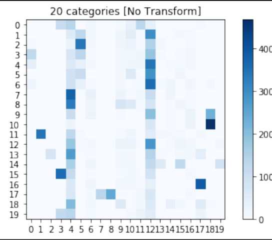
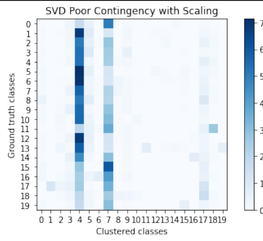
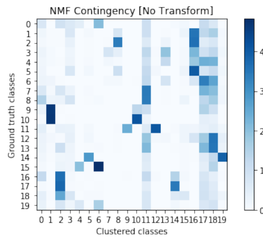
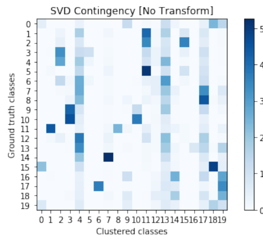
Transforms	Best r	Homogeneity	Completeness	V-measure	Rand	Mutual Information	Contingency Table
TF-IDF	NA	0.3594	0.4511	0.4001	0.1366	0.3573	
SVD + Scale	300	0.0764	0.145	0.1001	0.0184	0.0733	
NMF	10	0.3089	0.3438	0.3254	0.119	0.3066	
SVD	10	0.3323	0.3735	0.3517	0.1313	0.3301	

Table 17. Score for worst combinations of transformations

6 Conclusion

Based on our results, we found that the “best” number of components for dimensionality reduction was $r = 2$, particularly with NMF-reduced data. We saw that performing various transformations could affect the performance both positively and negatively. In the end, we saw that logarithmic transformation then normalization improved the SVD-reduced data the most, and normalization then logarithmic transformation improved the NMF-reduced data the most. However, other combinations such as just normalization or just logarithmic transformation, also improved the data and had scores very close to the ones just described.

For the 20 Category Data, some best and worst combinations were presented. By performing normalization and then logarithmic transformation on the SVD-reduced data, the K-means was able to achieve better scores than when K-means was clustered on the TF-IDF data. The worst result achieved unusable results, and K-means was unable to distinguish any of the classes from one another.

Dimension reduction and data transformation can improve clustering results, however this method is proven to be a double-edged sword that must be fine-tuned for every dataset.

7 Appendix

7.1 20 categories used by section 5

'alt.atheism', 'comp.graphics', 'comp.os.ms-windows.misc', 'comp.sys.ibm.pc.hardware', 'comp.sys.mac.hardware', 'comp.windows.x', 'misc.forsale', 'rec.autos', 'rec.motorcycles', 'rec.sport.baseball', 'rec.sport.hockey', 'sci.crypt', 'sci.electronics', 'sci.med', 'sci.space', 'soc.religion.christian', 'talk.politics.guns', 'talk.politics.mideast', 'talk.politics.misc', 'talk.religion.misc'

7.2 Contingency Matrices

Here are the contingency matrices we got at various parts in the iPython Notebook.

SVD for Different Values of r

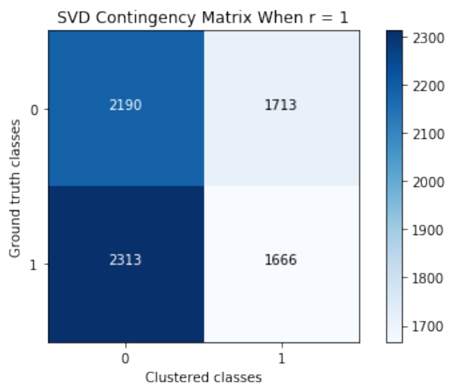


Fig. 30. SVD, $r = 1$

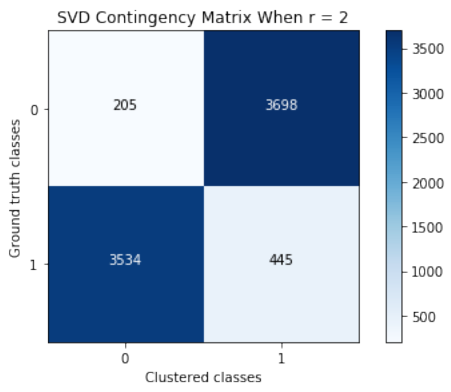


Fig. 31. SVD, $r = 2$

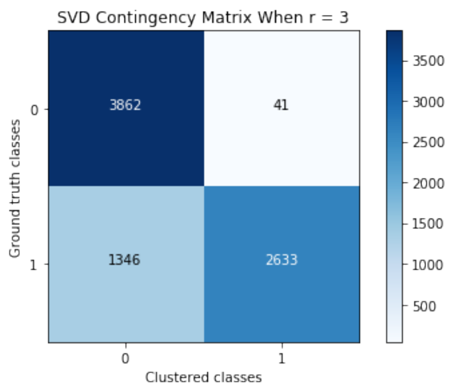


Fig. 32. SVD, $r = 3$

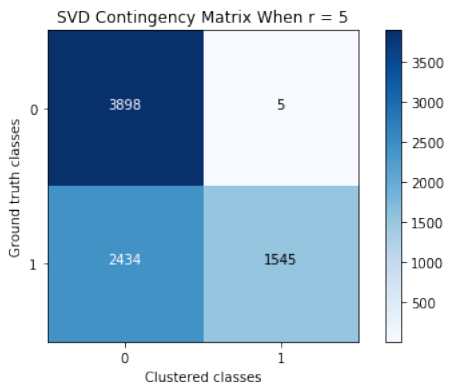


Fig. 33. SVD, $r = 5$

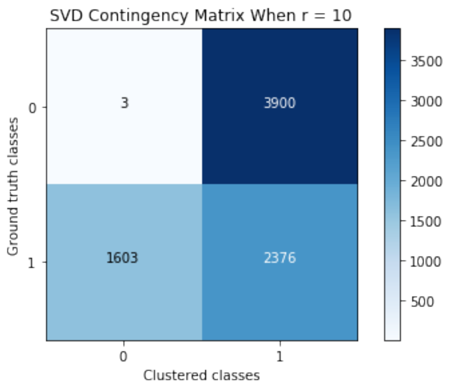


Fig. 34. SVD, $r = 10$

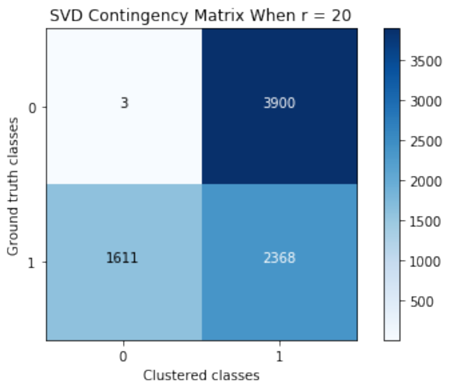


Fig. 35. SVD, $r = 20$

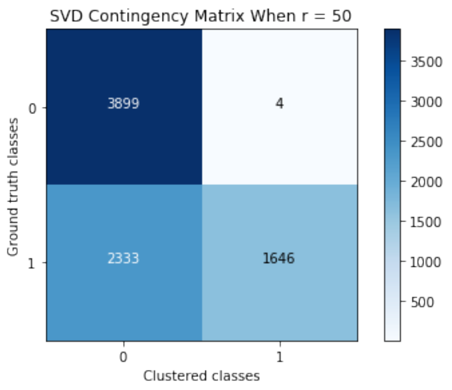


Fig. 36. SVD, $r = 50$

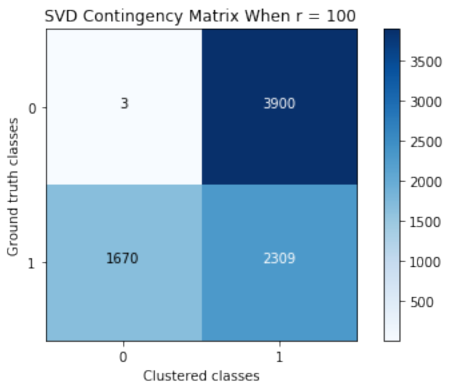


Fig. 37. SVD, $r = 100$

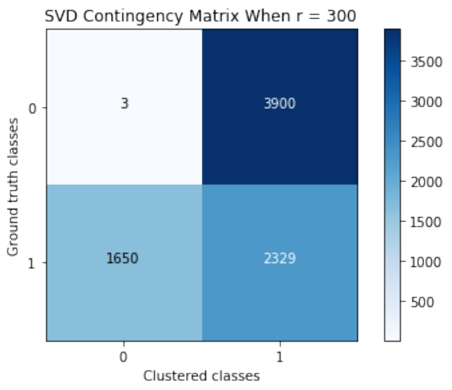


Fig. 38. SVD, $r = 300$

NMF for Different Values of r

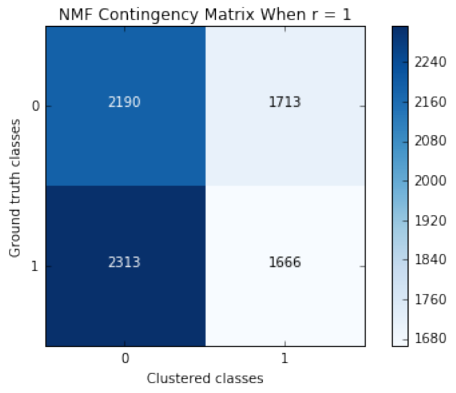


Fig. 39. NMF, $r = 1$

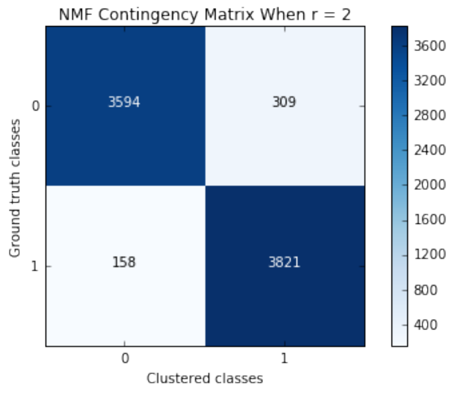


Fig. 40. NMF, $r = 2$

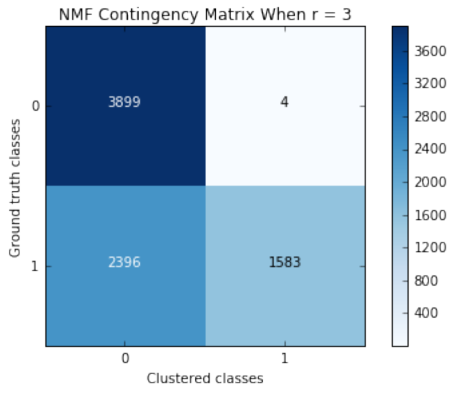


Fig. 41. NMF, $r = 3$

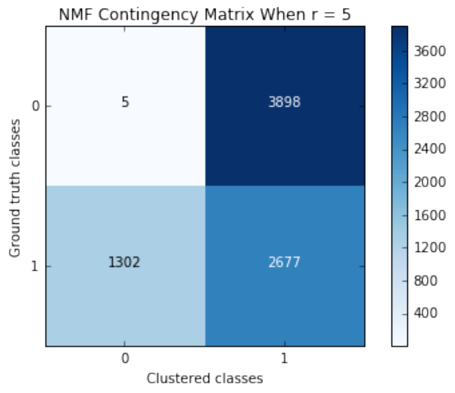


Fig. 42. NMF, $r = 5$

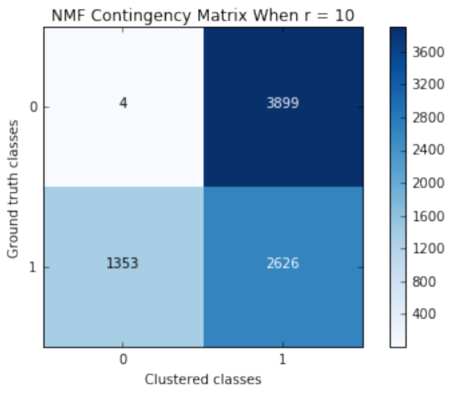


Fig. 43. NMF, $r = 10$

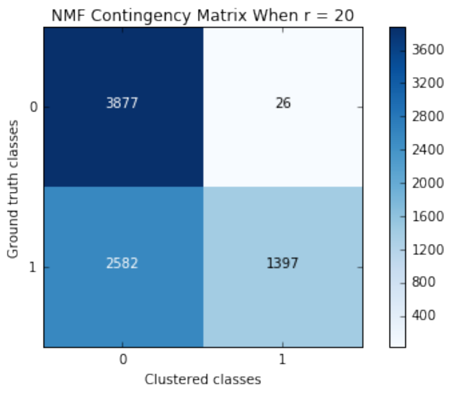


Fig. 44. NMF, $r = 20$

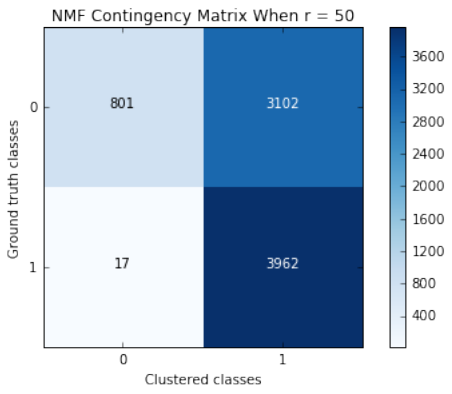


Fig. 45. NMF, $r = 50$

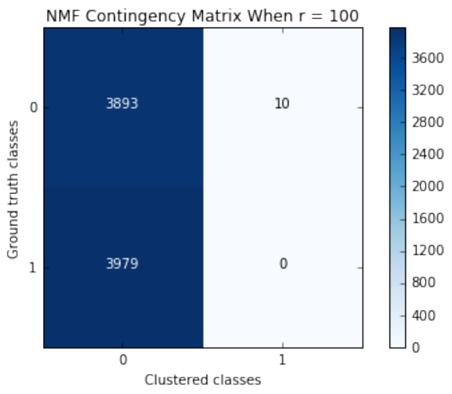


Fig. 46. NMF, $r = 100$

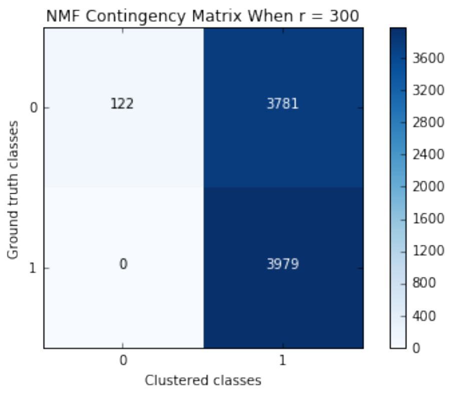


Fig. 47. NMF, $r = 300$

Contingency Matrices for Various Transformations

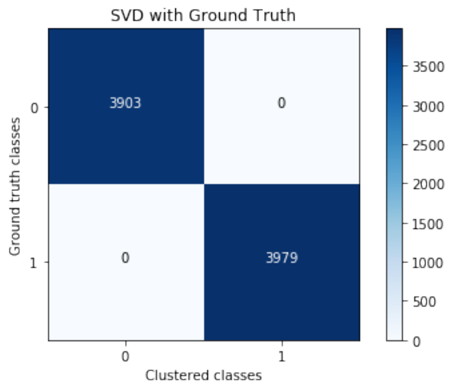


Fig. 48. SVD, Ground Truth

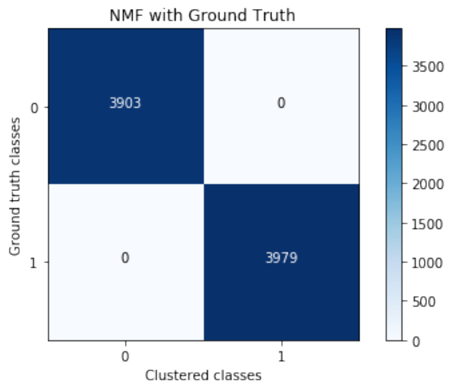


Fig. 49. NMF, Ground Truth

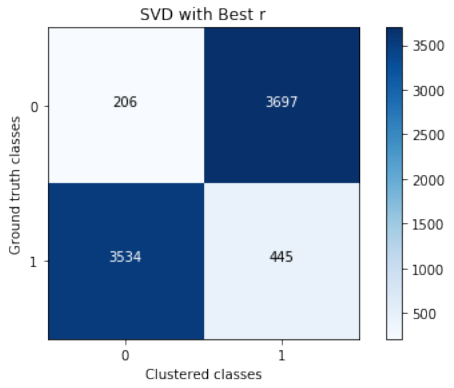


Fig. 50. SVD, Best r

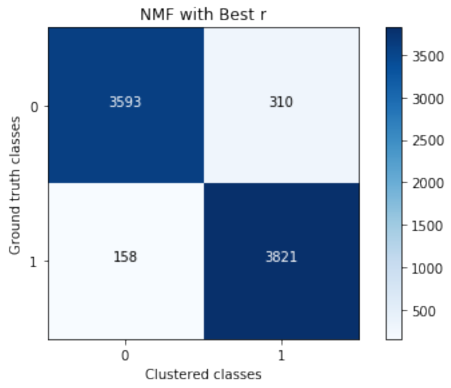


Fig. 51. NMF, Best r

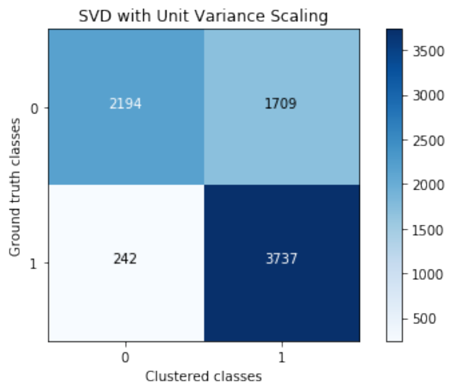


Fig. 52. SVD, Scaled

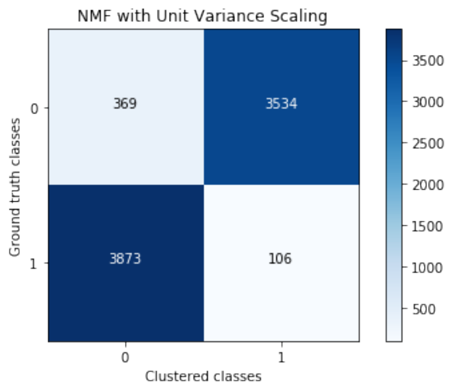


Fig. 53. NMF, Scaled

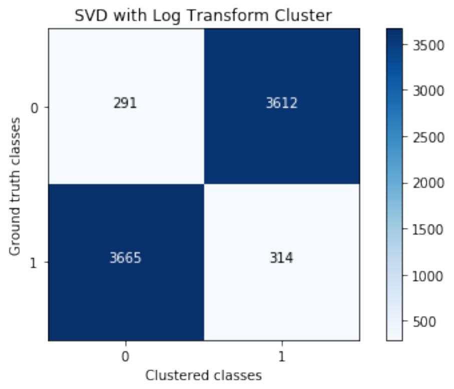


Fig. 54. SVD, Log Transformation

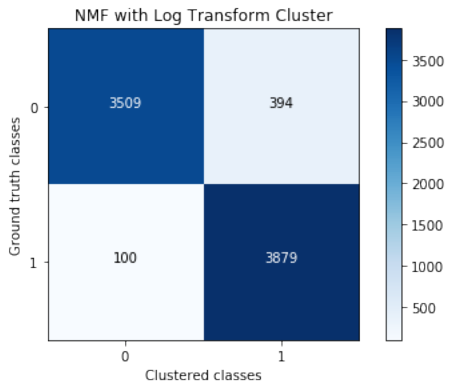


Fig. 55. NMF, Log Transformation

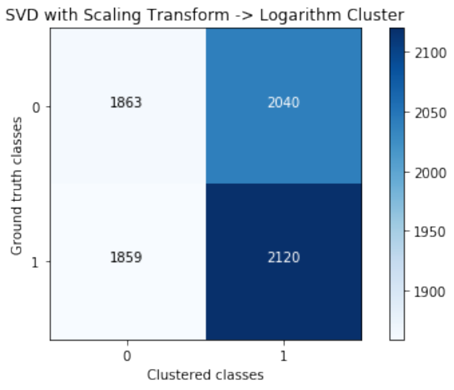


Fig. 56. SVD, Scaled then Log

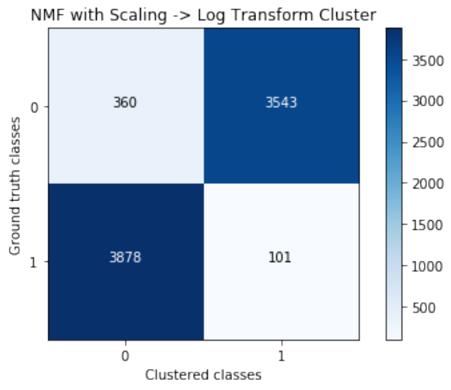


Fig. 57. NMF, Scaled then Log

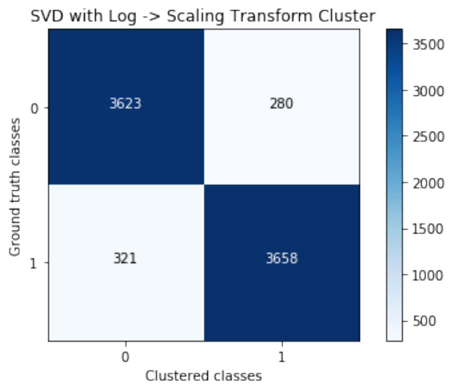


Fig. 58. SVD, Log then Scaled

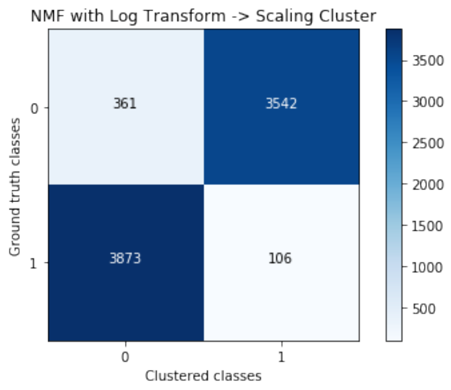


Fig. 59. NMF, Log then Scaled

7.3 Transformations on 20 Category Data

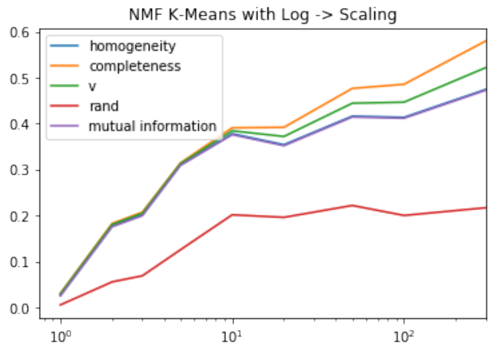


Fig. 60. 20-class scores from NMF with Log then Scaling

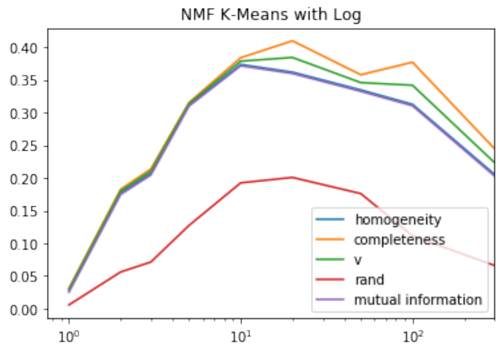


Fig. 61. 20-class scores from NMF with Log

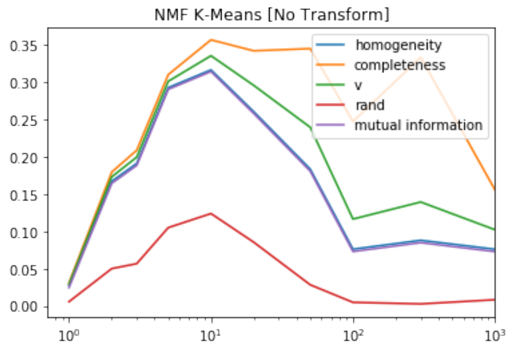


Fig. 62. 20-class scores from NMF with No Transforms

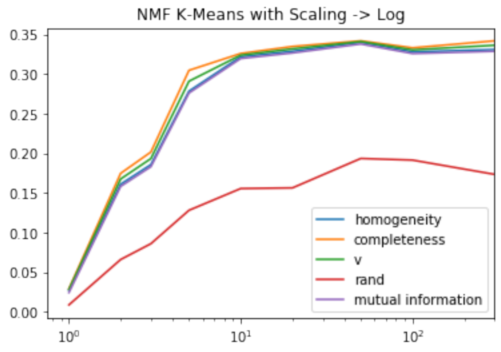


Fig. 63. 20-class scores from NMF with Scaling then Log

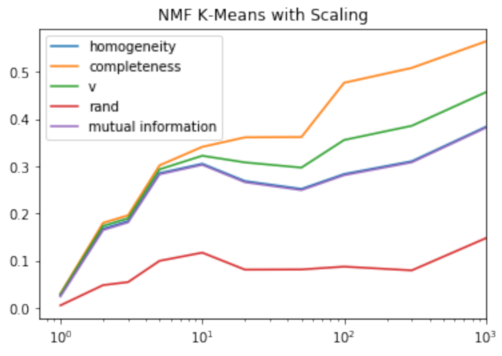


Fig. 64. 20-class scores from NMF with Scaling

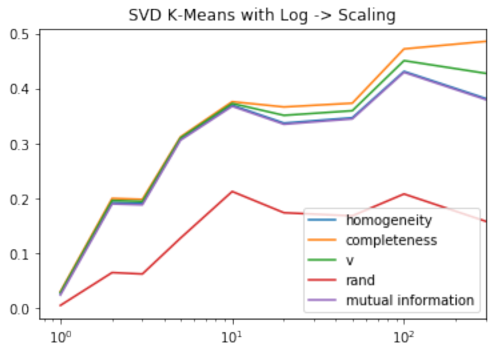


Fig. 65. 20-class scores from SVD with Log then Scaling

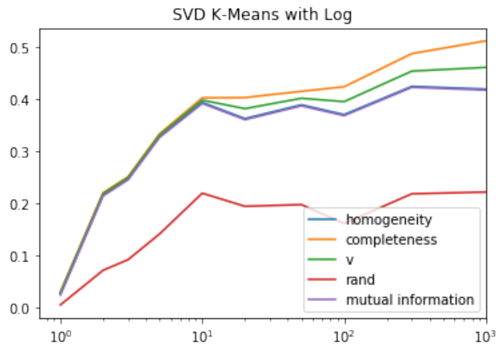


Fig. 66. 20-class scores from SVD with Log

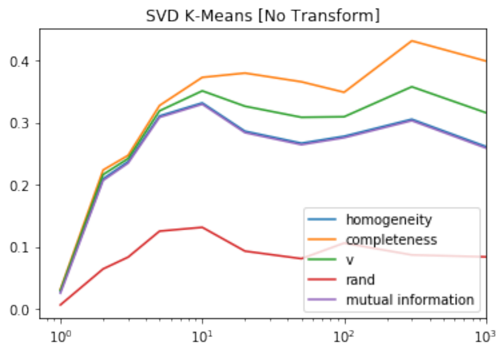


Fig. 67. 20-class scores from SVD with No Transforms

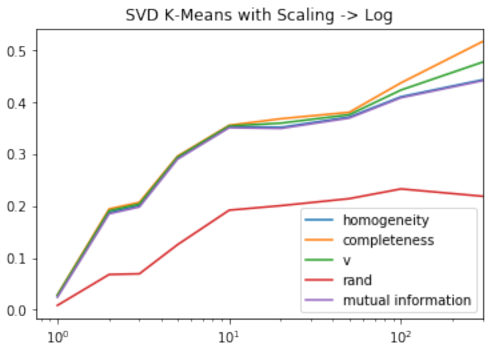


Fig. 68. 20-class scores from SVD with Scaling then Log

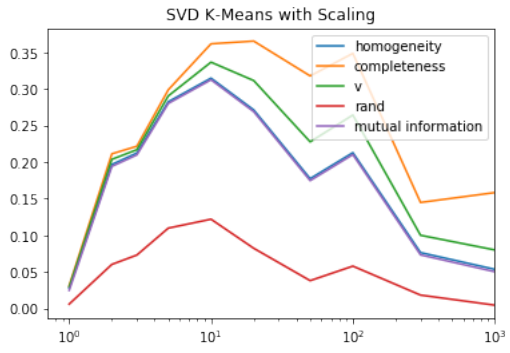


Fig. 69. 20-class scores from SVD with Scaling

Bibliography

[1] V. Roychowdhury, “Project 2: Clustering,” University of California, Los Angeles, 2019.

[2] “1.4. support vector machines - scikit-learn 0.19.2 documentation, 2.3 clustering.” `https://scikit-learn.org/stable/modules/clustering.html#contingency-matrix`.