Large-Scale Data Mining: Models and Algorithms
ECE 219 Winter 2023
Prof. Vwani Roychowdhury
UCLA, Department of ECE

Project 2: Data Representations and Clustering

Due Feb 12, 2023 by 11:59 AM

Yaman Yucel - 605704529
Ozgur Bora Gevrek - 505846360
Eray Eren - 006075032

Introduction:

In this project, data representations which are also known as feature extraction methods are used on clustering algorithms. Clustering algorithms are a class of unsupervised machine learning methods, where an algorithm tries to label data without the knowledge of ground truth labels.

The dataset used in this project is available at sklearn.datasets and is named as fetch_20news groups.

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

The dataset is divided into two classes to perform 2 classes k-means. For other parts, all classes are used. Note that, this dataset includes texts, so TF-IDF transformation was required to obtain a numerical feature extracted matrix.

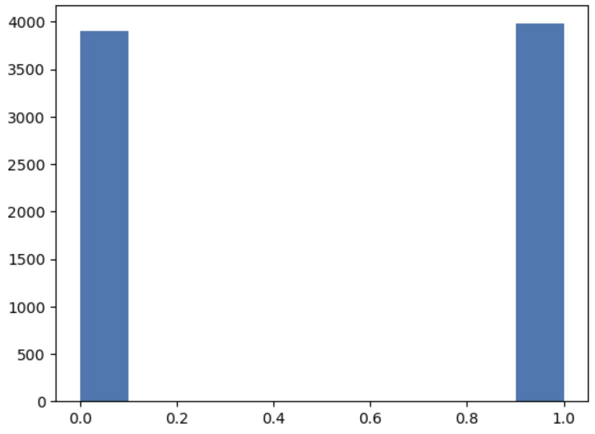


Figure 1: Distribution of classes, Class 1: 0, Class 2: 1

Note that classes are balanced.

Question 1: Report the dimensions of the TF-IDF matrix you obtain.

Dimensions of TF-IDF matrix: 7882,19198. 7882 samples and 19198 features(vocabulary size).

Question 2: Report the contingency table of your clustering result. Does the contingency matrix have to be square-shaped?

For this part K-means clustering with the following parameters are applied.
n_clusters = 2
random_state = 0
max_iter = 2000
n_init = 40

Using the train data for prediction, following contingency table is obtained:

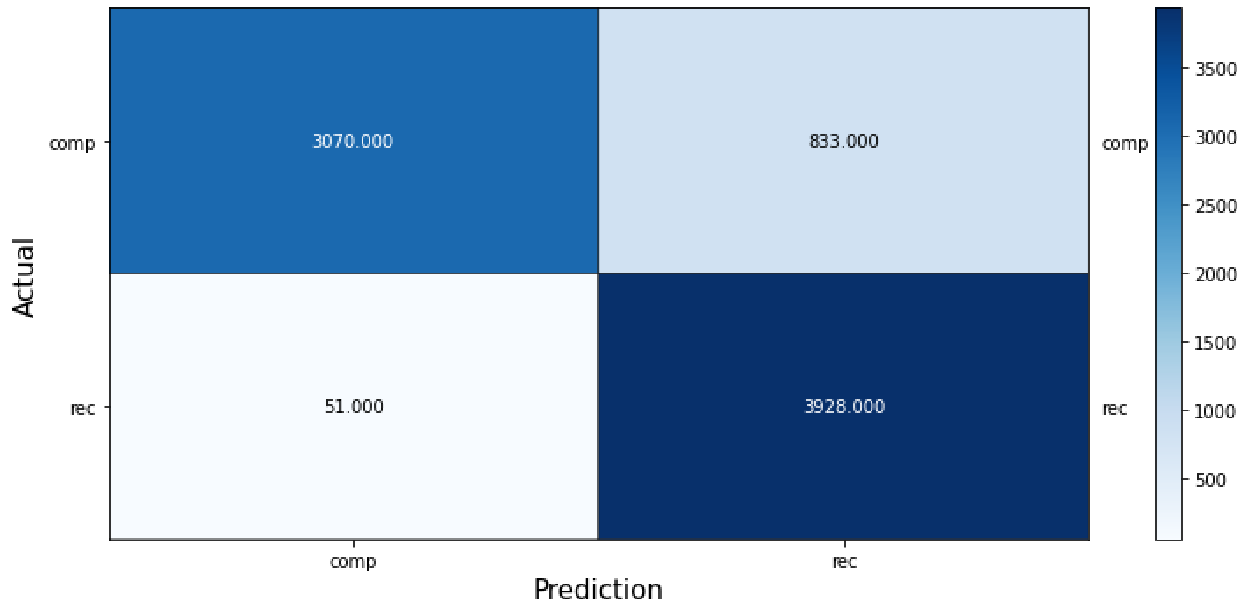


Figure 2: Contingency Table for K-means clustering with 2 clusters.

Linear sum assignment is used to correctly match label number with cluster number.

- We have acquired a **square matrix** since our number of classes and number of clusters match. However, one may choose the number of clusters different from the number of classes, then a rectangular contingency table will be obtained. For example, if we selected n_clusters = 3, there will be three columns for prediction, one for each cluster.

Question 3: Report the 5 clustering measures explained in the introduction for K-means clustering.

	Homogeneity	V-measure	Completeness	Adjusted rand	Adjusted Mutual Info
Score	0.5482	0.5569	0.5660	0.6016	0.5569

Table 1: Clustering metrics for K-means on TFIDF-matrix with 2 clusters.

There is room for improvement since these scores are not perfect. We can use truncated SVD or NMF for feature extraction.

Question 4: Report the plot of the percentage of variance that the top r principal components retain v.s. R, for r = 1 to 1000.

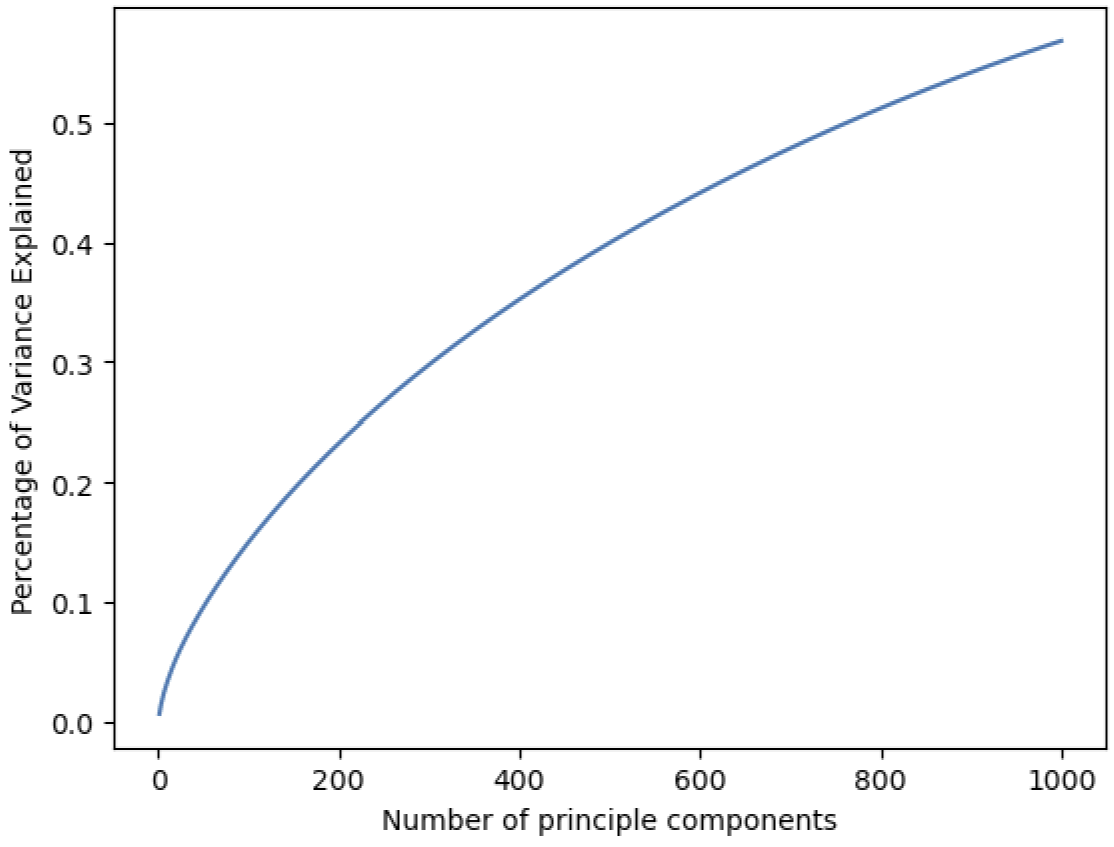


Figure 3: Percentage of variance vs number of principal components

The relationship is concave which indicates that the introducing the latest principal components have less effect than the first principal component. Adding a new principal component for analysis does not increase as much as adding the first principal component.

Question 5: Plot the 5 measure scores vs r for both SVD and NMF. Report a good choice of r for SVD and NMF respectively.

Following plots are obtained for 5 measure scores when we have used Truncated SVD:

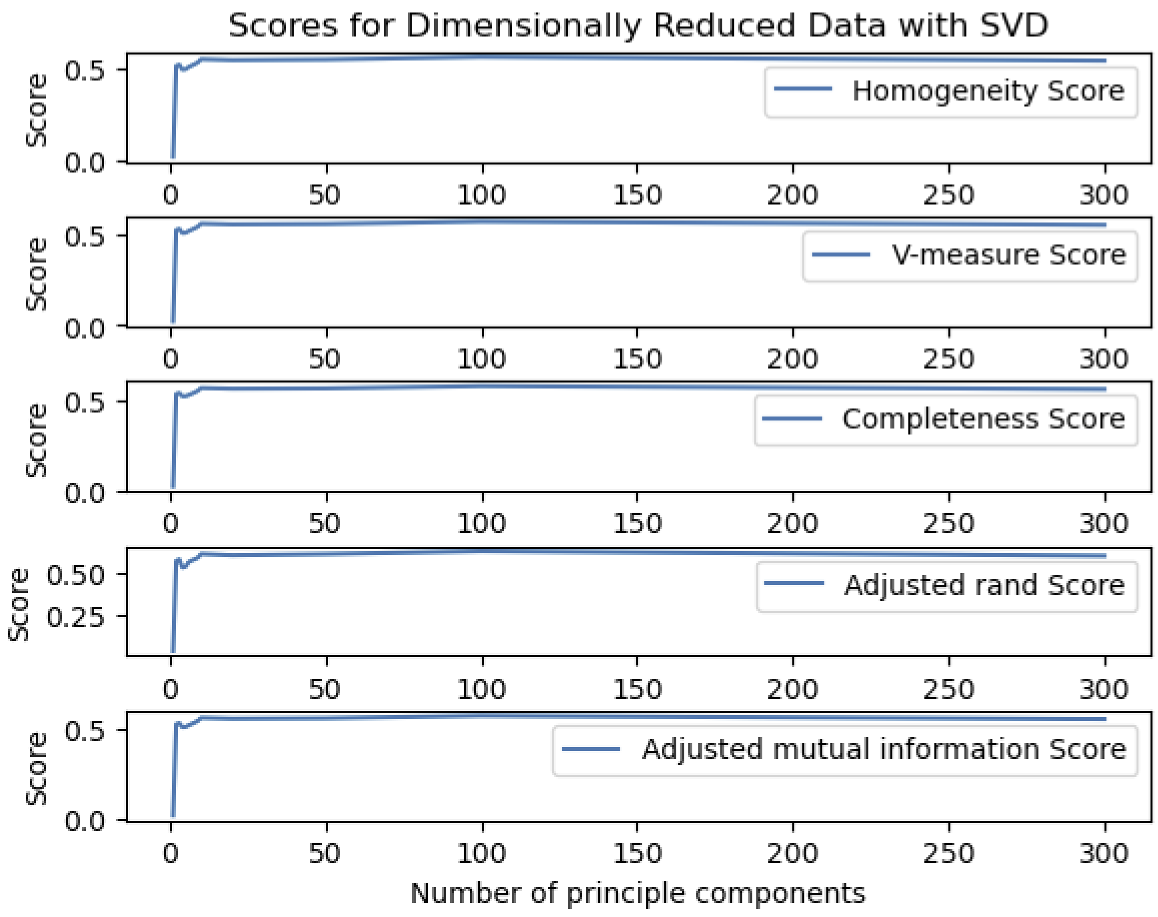


Figure 4: Scores for Dimensionally Reduced Data with SVD vs number of principal components

When we examined the five plots, all scores were maximized when the number of principal components was **100**.

Following plots are obtained for 5 measure scores when we have used NMF:

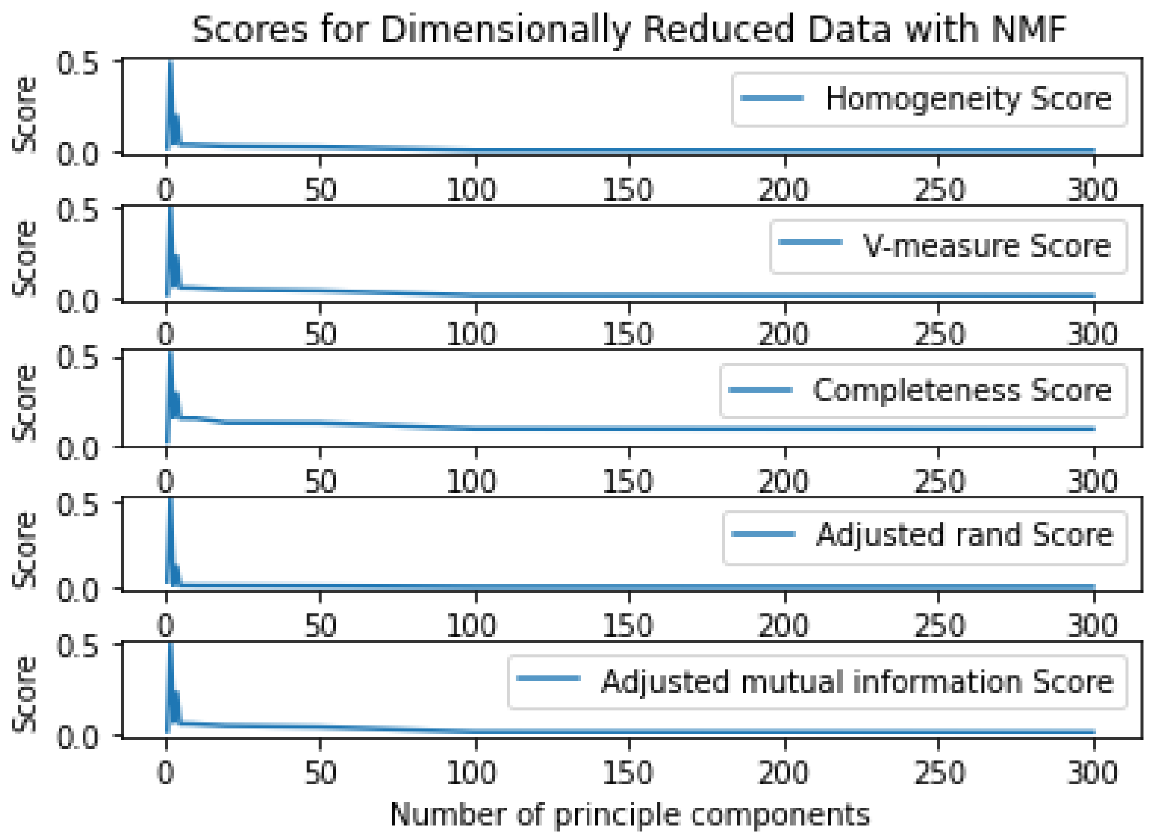


Figure 5: Scores for Dimensionally Reduced Data with NMF vs number of principal components

When we examined the five plots, all scores were maximized when the number of principal components was **2**.

Question 6: How do you explain the non-monotonic behavior of the measures as r increases?

It is easy to observe that score graphs are non-monotonic. When r is too low, we require more information to cluster data points more accurately. The number of features are inadequate for the K-means algorithm to work. However, when we use too many principal components, we also get low scores. The main reason behind this phenomenon is the meaning of the euclidean distance or any distance measure. When the feature space is highly dimensional, distance between data points becomes meaningless. Note that the data is sparse, therefore our clusters include huge gaps inside. We have only 7882 samples, which is not enough to cover all space. Therefore, using too many principal components not only introduces error into clustering, but also disrupts the algorithm's performance. Therefore, the r parameter should be not too small or too high. Also for NMF, we believe that as dimensionality increases NMF can not converge to a local minima.

Question 7: Are these measures on average better than those computed in Question 3?

	Homogeneity	V-measure	Completeness	Adjusted rand	Adjusted Mutual Info
Average SVD	0.4938	0.5027	0.5120	0.5404	0.5027
Average NMF	0.077	0.0969	0.1673	0.0539	0.0968

Table 2: The average scores for SVD and NMF

On average, all measures perform significantly worse than the results at Question 3.

	Homogeneity	V-measure	Completeness	Adjusted rand	Adjusted Mutual Info
Score(Q3)	0.5482	0.5569	0.5660	0.6016	0.5569
Ideal SVD	**0.5653**	**0.5729**	**0.5807**	**0.6246**	**0.5728**
Ideal NMF	0.5162	0.5260	0.5361	0.5663	0.5260

Table 3: The ideal scores for SVD and NMF.

Higher scores are obtained when we do dimensionality reduction using SVD. However, the score of NMF is lower than the score at Question 3. Another advantage of using dimensionality reduction is that data becomes easier to be processed and used for algorithms.

Question 8: Visualize the clustering results

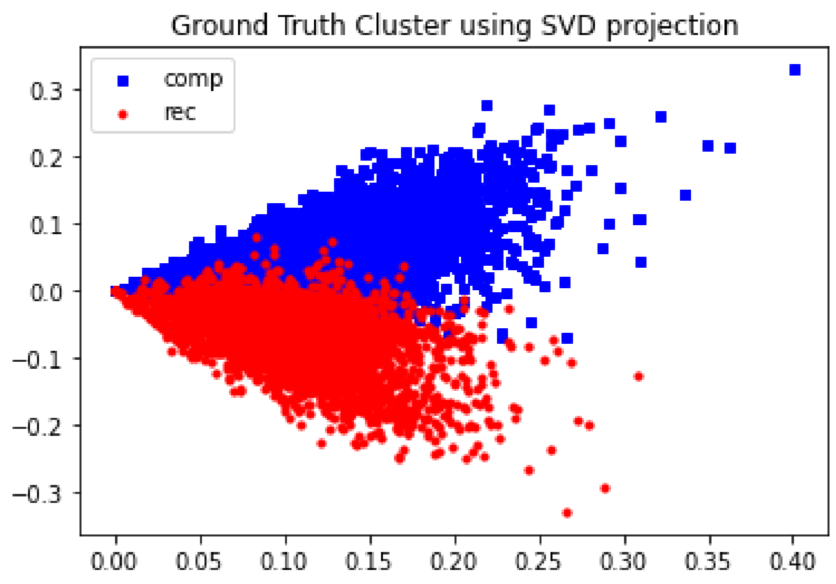


Figure 6: Ground Truth Cluster Visualization using SVD Projection

Firstly, the TF-IDF data matrix is visualized using the SVD projection feature onto 2D. Note that two classes are nearly separable from each other in 2D. Then, the data matrix is reduced to 100 dimensions using SVD. With that data, k-means clustering is made and cluster labels are found. Then, the data is again projected to 2D using SVD. Notice that clusters are separated by a linear line in 2D, and that line is similar to the separation at the ground truth cluster.

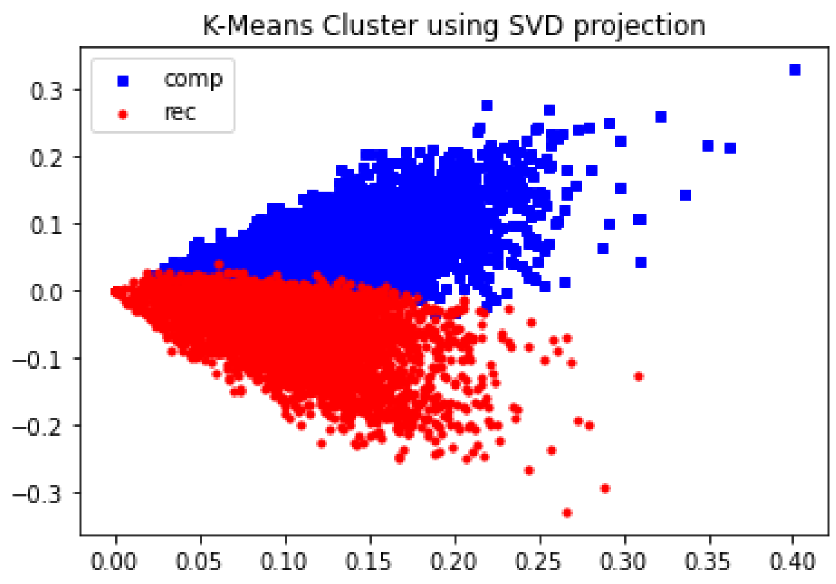


Figure 7: K-Means Clustering after Dimensionality Reduction with SVD Visualization using SVD Projection

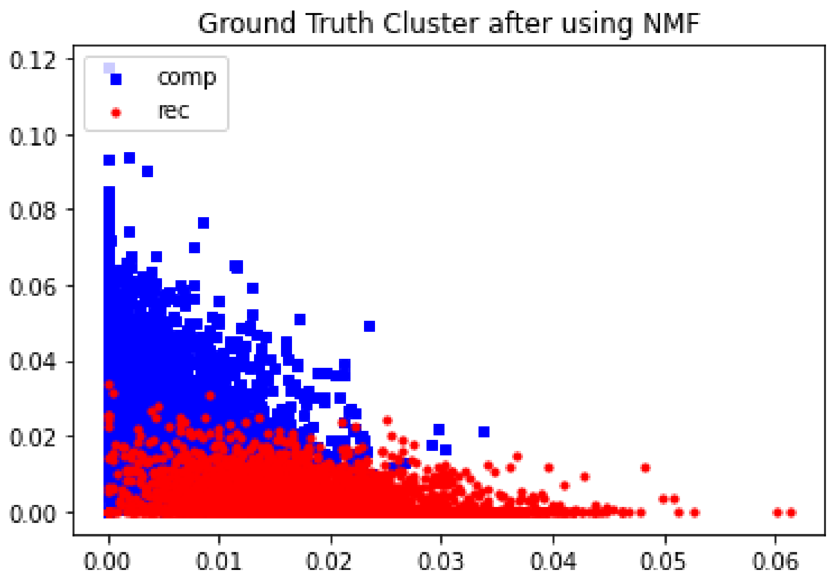


Figure 8: Ground Truth Cluster after Dimensionality Reduction with NMF

Firstly, the TF-IDF data matrix is visualized in 2D. For NMF, we do not need to project our data onto 2D since the best r value is 2 for NMF. Note that two classes are nearly separable from each other in 2D. After ground visualization, we have acquired clustering labels using the k-means algorithm and plotted our dimensionality reduced data using the cluster labels. Notice that clusters are separated by a linear line in 2D, and that line is similar to the separation at the ground truth cluster.

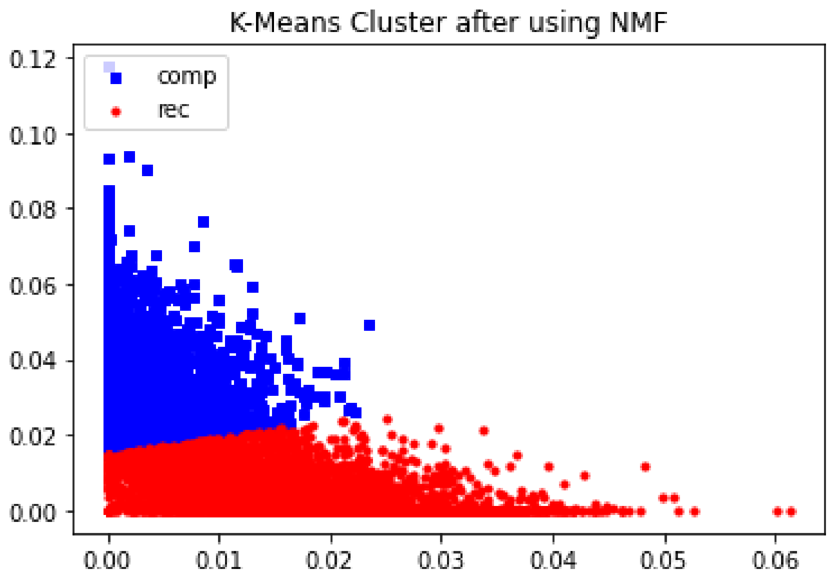


Figure 9: K-Means Clustering after Dimensionality Reduction with NMF

Question 9: Visualize the clustering results

When we have observed ground truth figures for both SVD and NMF, we can easily see that the data is separable in 2 dimensions. There is not much interference between different labels, therefore k-means clustering can be used for this dataset. Finding a good cluster center is essential for k-means clustering to work, therefore from the ground truth clusters we can see that centers are easily found. Our observation is validated with NMF and SVD. After reducing the dimension of the data, data remains to be used for the K-means algorithm to use. Model can cluster data points accurately even after dimensionality reduction.

Question 10: Construct the TF-IDF matrix, reduce its dimensionality using BOTH NMF and SVD (specify settings you choose and why), and perform K-Means clustering with k=20 . Visualize the contingency matrix and report the five clustering metrics (DO BOTH NMF AND SVD).

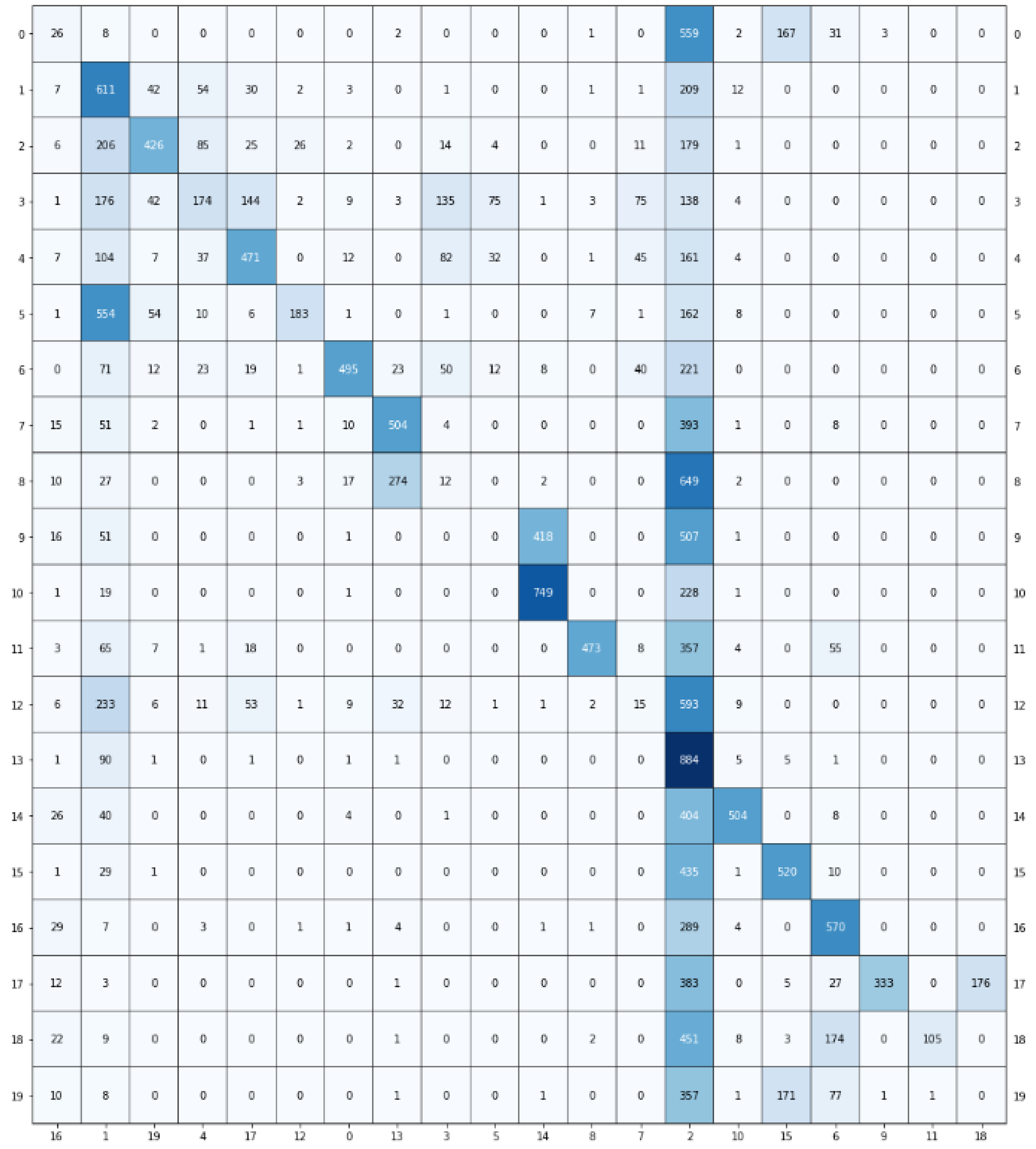
We used the same data extraction as the first part. Since we loaded the whole 20 categories, our columns (tf-idf features) increased to 38627. We still have the same 18846 samples or rows.

In order to see the effects of different numbers of components for SVD and NMF, we conducted experiments for the number of components = {5,20,200}. For each setting we fitted k-means clustering and evaluated the metrics. Results are shown in Table 4. Although k-means with these settings did not perform ideally, we can observe from the metrics that SVD with the number of components = 200 performed the best.

	Homogeneity	Completeness	V-measure	Adjusted rand	Adjusted Mutual Info	Avg. of Scores
SVD (n_comp=5)	0.31	0.34	0.32	**0.12**	0.32	0.28
SVD (n_comp=20)	0.32	0.37	0.34	0.11	0.34	0.29
SVD (n_comp=200)	**0.37**	**0.49**	**0.42**	0.10	**0.42**	**0.36**
NMF (n_comp=5)	0.23	0.26	0.25	0.07	0.24	0.21
NMF (n_comp=20)	0.35	0.42	0.38	0.10	0.38	0.33
NMF (n_comp=200)	0.14	0.19	0.16	0.03	0.16	0.13

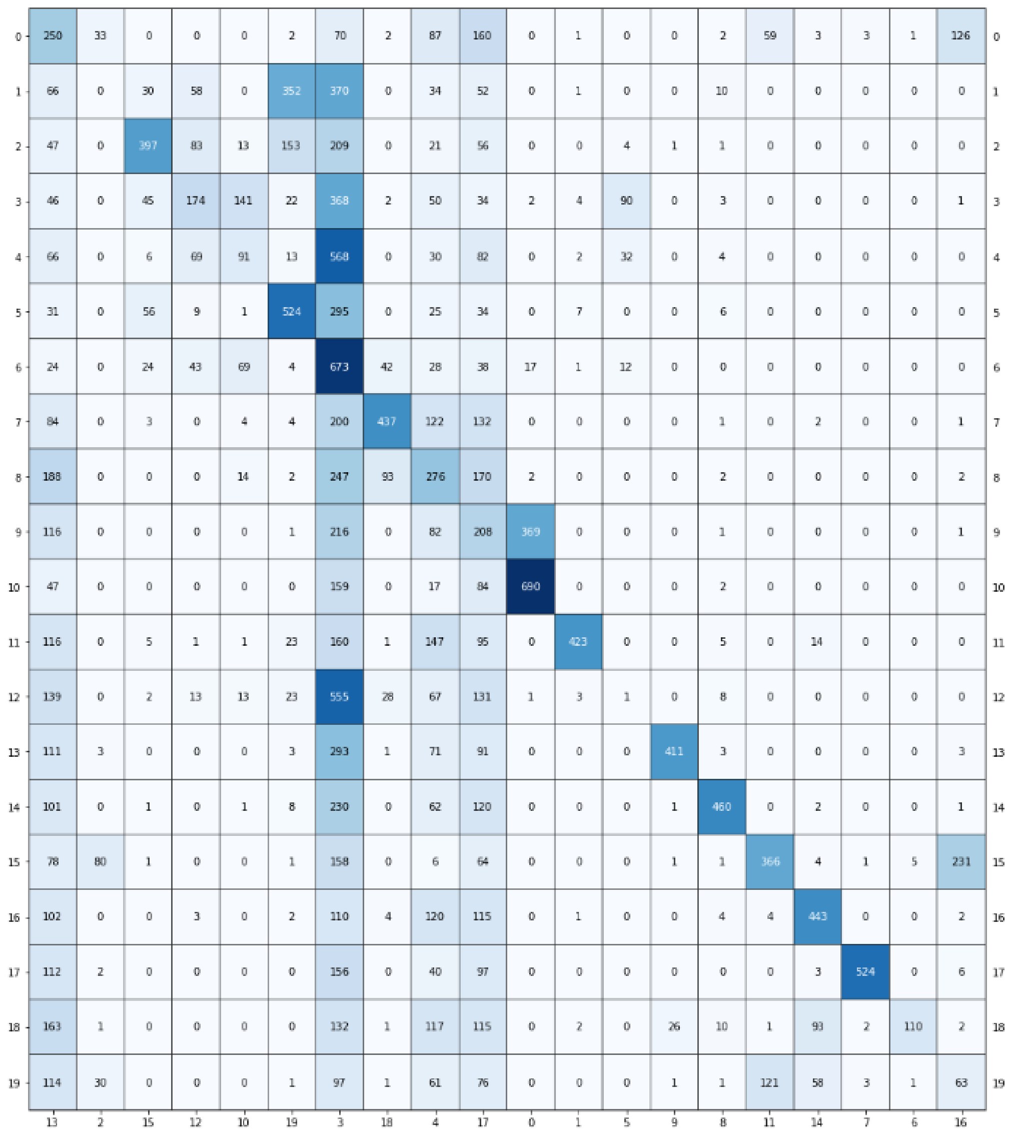
Table 4: SVD vs NMF clustering (k-means) metrics for different n_component (5, 20, 200).

Since the soft labels by clustering does not necessarily correspond to true labels, we reordered the contingency matrix with best matching cluster-class pairs. We used the code given by the question. The contingency matrix for the best clustering with SVD is shown in Figure 10, and the contingency matrix for the best clustering with NMF is shown in Figure 11.



	16	1	19	4	17	12	0	13	3	5	14	8	7	2	10	15	6	9	11	18	
0	26	8	0	0	0	0	0	2	0	0	0	1	0	559	2	167	31	3	0	0	0
1	7	611	42	54	30	2	3	0	1	0	0	1	1	209	12	0	0	0	0	0	1
2	6	206	426	85	25	26	2	0	14	4	0	0	11	179	1	0	0	0	0	0	2
3	1	176	42	174	144	2	9	3	135	75	1	3	75	138	4	0	0	0	0	0	3
4	7	104	7	37	471	0	12	0	82	32	0	1	45	161	4	0	0	0	0	0	4
5	1	554	54	10	6	183	1	0	1	0	0	7	1	162	8	0	0	0	0	0	5
6	0	71	12	23	19	1	495	23	50	12	8	0	40	221	0	0	0	0	0	0	6
7	15	51	2	0	1	1	10	504	4	0	0	0	0	393	1	0	8	0	0	0	7
8	10	27	0	0	0	3	17	274	12	0	2	0	0	649	2	0	0	0	0	0	8
9	16	51	0	0	0	0	1	0	0	0	418	0	0	507	1	0	0	0	0	0	9
10	1	19	0	0	0	0	1	0	0	0	749	0	0	228	1	0	0	0	0	0	10
11	3	65	7	1	18	0	0	0	0	0	0	473	8	357	4	0	55	0	0	0	11
12	6	233	6	11	53	1	9	32	12	1	1	2	15	593	9	0	0	0	0	0	12
13	1	90	1	0	1	0	1	1	0	0	0	0	0	884	5	5	1	0	0	0	13
14	26	40	0	0	0	0	4	0	1	0	0	0	0	404	504	0	8	0	0	0	14
15	1	29	1	0	0	0	0	0	0	0	0	0	0	435	1	520	10	0	0	0	15
16	29	7	0	3	0	1	1	4	0	0	1	1	0	289	4	0	570	0	0	0	16
17	12	3	0	0	0	0	0	1	0	0	0	0	0	383	0	5	27	333	0	176	17
18	22	9	0	0	0	0	0	1	0	0	0	2	0	451	8	3	174	0	105	0	18
19	10	8	0	0	0	0	0	1	0	0	1	0	0	357	1	171	77	1	1	0	19

Figure 10: x-axis is the predicted labels by the best k-means clustering with SVD(n_component=200). Y-axis is the true labels.



	13	2	15	12	10	19	3	18	4	17	0	1	5	9	8	11	14	7	6	16	
0	250	33	0	0	0	2	70	2	87	160	0	1	0	0	2	59	3	3	1	126	0
1	66	0	30	58	0	352	370	0	34	52	0	1	0	0	10	0	0	0	0	0	1
2	47	0	397	83	13	153	209	0	21	56	0	0	4	1	1	0	0	0	0	0	2
3	46	0	45	174	141	22	368	2	50	34	2	4	90	0	3	0	0	0	0	1	3
4	66	0	6	69	91	13	568	0	30	82	0	2	32	0	4	0	0	0	0	0	4
5	31	0	56	9	1	524	295	0	25	34	0	7	0	0	6	0	0	0	0	0	5
6	24	0	24	43	69	4	673	42	28	38	17	1	12	0	0	0	0	0	0	0	6
7	84	0	3	0	4	4	200	437	122	132	0	0	0	0	1	0	2	0	0	1	7
8	188	0	0	0	14	2	247	93	276	170	2	0	0	0	2	0	0	0	0	2	8
9	116	0	0	0	0	1	216	0	82	208	369	0	0	0	1	0	0	0	0	1	9
10	47	0	0	0	0	0	159	0	17	84	690	0	0	0	2	0	0	0	0	0	10
11	116	0	5	1	1	23	160	1	147	95	0	423	0	0	5	0	14	0	0	0	11
12	139	0	2	13	13	23	555	28	67	131	1	3	1	0	8	0	0	0	0	0	12
13	111	3	0	0	0	3	293	1	71	91	0	0	0	411	3	0	0	0	0	3	13
14	101	0	1	0	1	8	230	0	62	120	0	0	0	1	460	0	2	0	0	1	14
15	78	80	1	0	0	1	158	0	6	64	0	0	0	1	1	366	4	1	5	231	15
16	102	0	0	3	0	2	110	4	120	115	0	1	0	0	4	4	443	0	0	2	16
17	112	2	0	0	0	0	156	0	40	97	0	0	0	0	0	0	3	524	0	6	17
18	163	1	0	0	0	0	132	1	117	115	0	2	0	26	10	1	93	2	110	2	18
19	114	30	0	0	0	1	97	1	61	76	0	0	0	1	1	121	58	3	1	63	19

Figure 11: x-axis is the predicted labels by the best k-means clustering with NMF(n_component=20). Y-axis is the true labels.

Question 11: UMAP. settings: n components = [5, 20, 200], metric = ”cosine” vs. ”euclidean”. If ”cosine” metric. Report the permuted contingency matrix and the five clustering evaluation metrics for the different combinations (6 combinations).

We conducted experiments for the UMAP reduction method with different distances and the number of components. We observed that using cosine distance instead of euclidean distance can significantly improve all metrics as shown in Table 5.

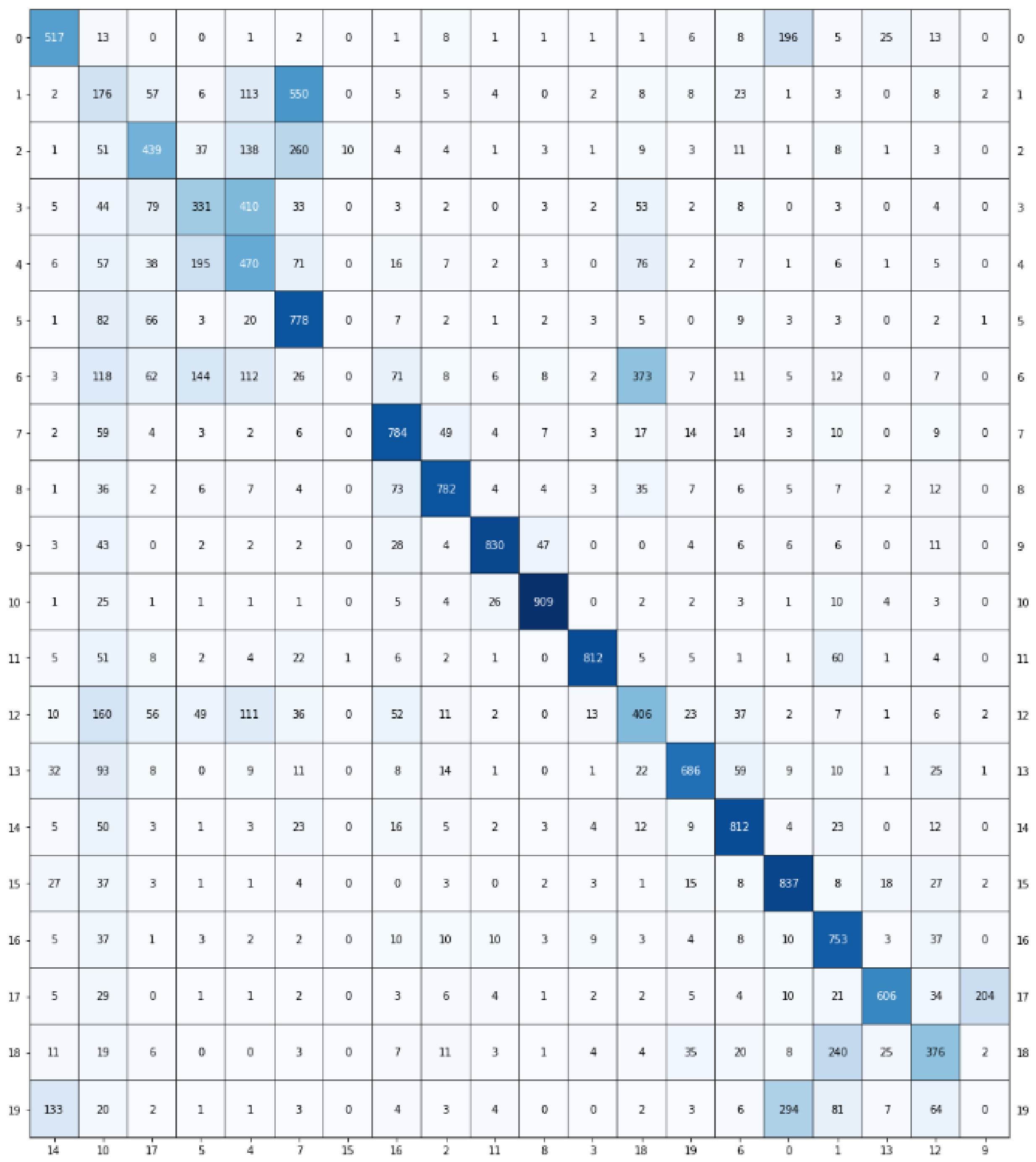
This is due to the fact that cosine distance can bypass the length of documents, which may negatively affect the results as in euclidean case. Furthermore, we observed that the best number of components for cosine is 20, which gives an average score of 0.56.

UMAP	Homogeneity	Completeness	V-measure	Adjusted rand	Adjusted Mutual Info	Avg. of Scores
cos./ n_comp=5	0.56	0.58	0.57	0.44	0.57	0.55
cos./ n_comp=20	**0.57**	**0.60**	**0.59**	**0.46**	**0.58**	**0.56**
cos./ n_comp=200	0.56	0.60	0.58	0.43	0.58	0.55
euc../ n_comp=5	0.007	0.007	0.007	0.001	0.004	0.005
euc../ n_comp=20	0.006	0.006	0.006	0.001	0.002	0.004
euc../ n_comp=200	0.006	0.006	0.006	0.001	0.003	0.005

Table 5: K-means clustering with UMAP. Different number of components are denoted with n_comp, distance metric is denoted with cos.(cosine) or euc.(euclidean).

From Figure 11,12, and 13, we can observe that the contingency matrix (CM) for cosine distance is superior to euclidean, which is shown in Figure 14, 15, 16. We can see that CMs for euclidean distance are poor because not only clusters cannot distinguish between different labels, but also models tend to predict the same cluster for different labels. On the other hand, CMs for cosine show that clusters are well-separated for different classes.

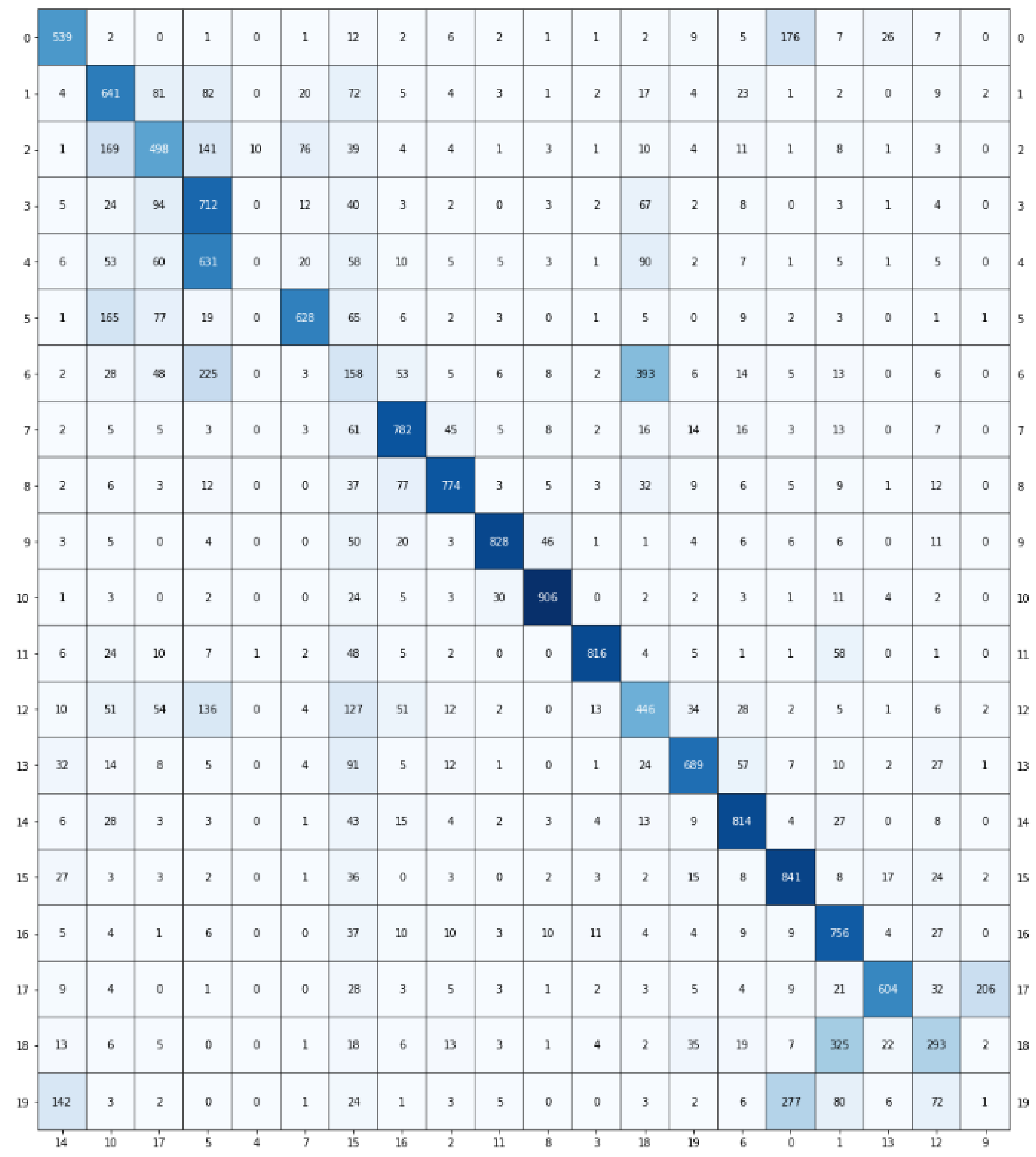
```
*** Permuted Conf. Mat. for n component: 5, dist: cosine : ***
```



	14	10	17	5	4	7	15	16	2	11	8	3	18	19	6	0	1	13	12	9	
0	517	13	0	0	1	2	0	1	8	1	1	1	1	6	8	196	5	25	13	0	0
1	2	176	57	6	113	550	0	5	5	4	0	2	8	8	23	1	3	0	8	2	1
2	1	51	439	37	138	260	10	4	4	1	3	1	9	3	11	1	8	1	3	0	2
3	5	44	79	331	410	33	0	3	2	0	3	2	53	2	8	0	3	0	4	0	3
4	6	57	38	195	470	71	0	16	7	2	3	0	76	2	7	1	6	1	5	0	4
5	1	82	66	3	20	778	0	7	2	1	2	3	5	0	9	3	3	0	2	1	5
6	3	118	62	144	112	26	0	71	8	6	8	2	373	7	11	5	12	0	7	0	6
7	2	59	4	3	2	6	0	784	49	4	7	3	17	14	14	3	10	0	9	0	7
8	1	36	2	6	7	4	0	73	782	4	4	3	35	7	6	5	7	2	12	0	8
9	3	43	0	2	2	2	0	28	4	830	47	0	0	4	6	6	6	0	11	0	9
10	1	25	1	1	1	1	0	5	4	26	909	0	2	2	3	1	10	4	3	0	10
11	5	51	8	2	4	22	1	6	2	1	0	812	5	5	1	1	60	1	4	0	11
12	10	160	56	49	111	36	0	52	11	2	0	13	406	23	37	2	7	1	6	2	12
13	32	93	8	0	9	11	0	8	14	1	0	1	22	686	59	9	10	1	25	1	13
14	5	50	3	1	3	23	0	16	5	2	3	4	12	9	812	4	23	0	12	0	14
15	27	37	3	1	1	4	0	0	3	0	2	3	1	15	8	837	8	18	27	2	15
16	5	37	1	3	2	2	0	10	10	10	3	9	3	4	8	10	753	3	37	0	16
17	5	29	0	1	1	2	0	3	6	4	1	2	2	5	4	10	21	606	34	204	17
18	11	19	6	0	0	3	0	7	11	3	1	4	4	35	20	8	240	25	376	2	18
19	133	20	2	1	1	3	0	4	3	4	0	0	2	3	6	294	81	7	64	0	19

Figure 11: x-axis is the predicted labels by the k-means clustering with UMAP(n_component=5, cosine distance). Y-axis is the true labels.

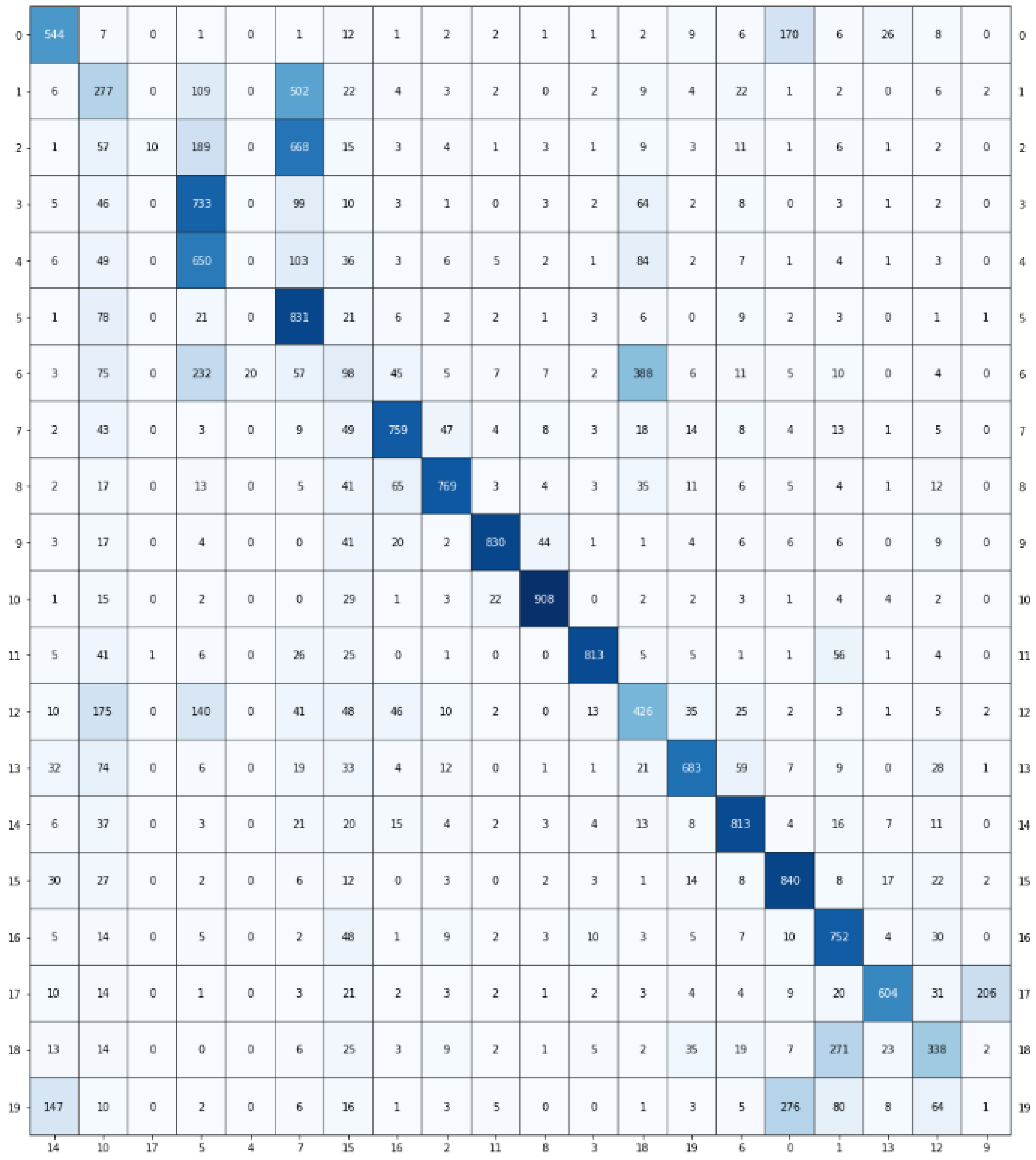
```
*** Permuted Conf. Mat. for n component: 20, dist: cosine : ***
```



	14	10	17	5	4	7	15	16	2	11	8	3	18	19	6	0	1	13	12	9	
0	539	2	0	1	0	1	12	2	6	2	1	1	2	9	5	176	7	26	7	0	0
1	4	641	81	82	0	20	72	5	4	3	1	2	17	4	23	1	2	0	9	2	1
2	1	169	498	141	10	76	39	4	4	1	3	1	10	4	11	1	8	1	3	0	2
3	5	24	94	712	0	12	40	3	2	0	3	2	67	2	8	0	3	1	4	0	3
4	6	53	60	631	0	20	58	10	5	5	3	1	90	2	7	1	5	1	5	0	4
5	1	165	77	19	0	628	65	6	2	3	0	1	5	0	9	2	3	0	1	1	5
6	2	28	48	225	0	3	158	53	5	6	8	2	393	6	14	5	13	0	6	0	6
7	2	5	5	3	0	3	61	782	45	5	8	2	16	14	16	3	13	0	7	0	7
8	2	6	3	12	0	0	37	77	774	3	5	3	32	9	6	5	9	1	12	0	8
9	3	5	0	4	0	0	50	20	3	828	46	1	1	4	6	6	6	0	11	0	9
10	1	3	0	2	0	0	24	5	3	30	906	0	2	2	3	1	11	4	2	0	10
11	6	24	10	7	1	2	48	5	2	0	0	816	4	5	1	1	58	0	1	0	11
12	10	51	54	136	0	4	127	51	12	2	0	13	446	34	28	2	5	1	6	2	12
13	32	14	8	5	0	4	91	5	12	1	0	1	24	689	57	7	10	2	27	1	13
14	6	28	3	3	0	1	43	15	4	2	3	4	13	9	814	4	27	0	8	0	14
15	27	3	3	2	0	1	36	0	3	0	2	3	2	15	8	841	8	17	24	2	15
16	5	4	1	6	0	0	37	10	10	3	10	11	4	4	9	9	756	4	27	0	16
17	9	4	0	1	0	0	28	3	5	3	1	2	3	5	4	9	21	604	32	206	17
18	13	6	5	0	0	1	18	6	13	3	1	4	2	35	19	7	325	22	293	2	18
19	142	3	2	0	0	1	24	1	3	5	0	0	3	2	6	277	80	6	72	1	19

Figure 12: x-axis is the predicted labels by the k-means clustering with UMAP(n_component=20, cosine distance). Y-axis is the true labels.

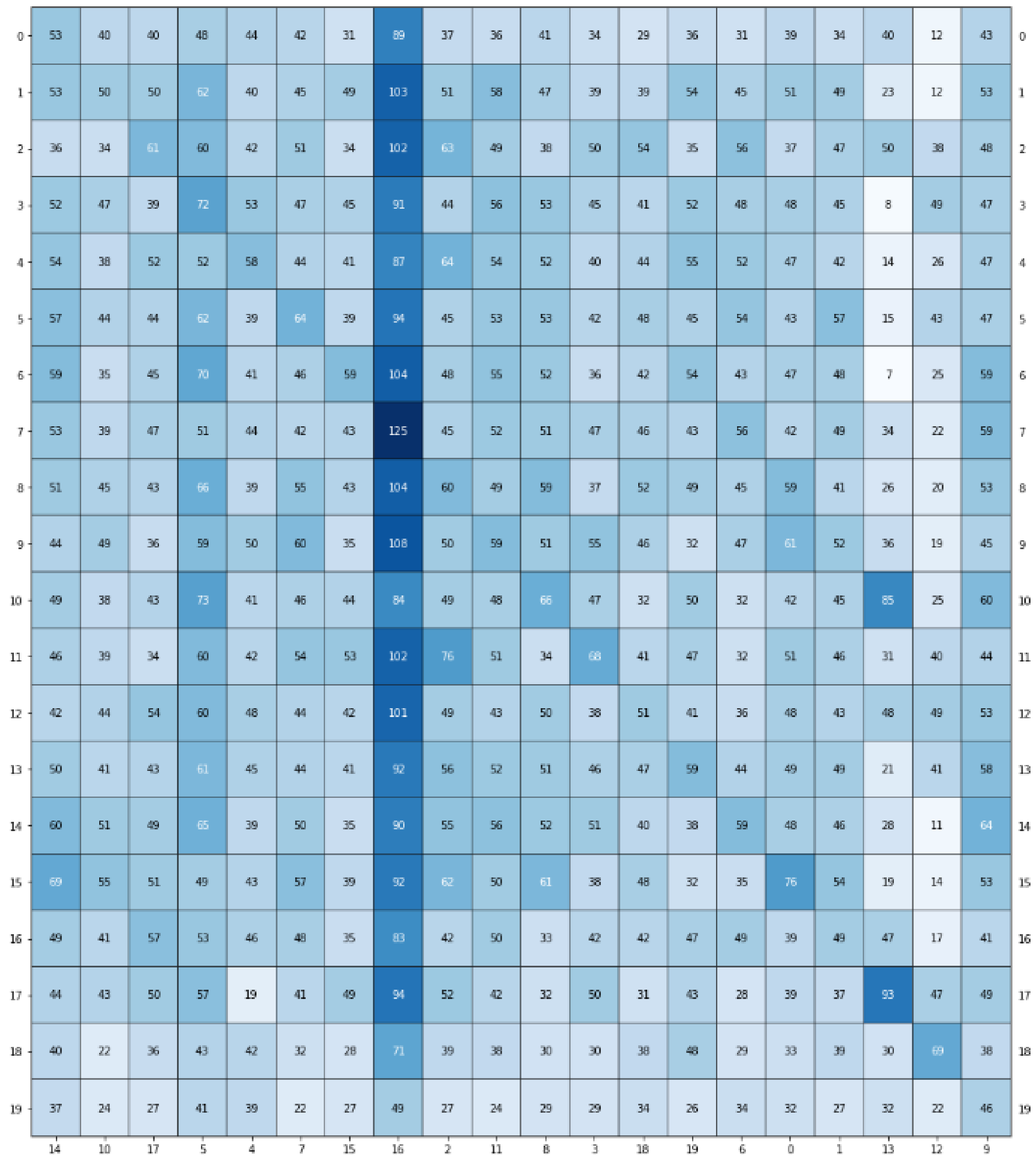
```
*** Permuted Conf. Mat. for n component: 200, dist: cosine : ***
```



	14	10	17	5	4	7	15	16	2	11	8	3	18	19	6	0	1	13	12	9	
0	544	7	0	1	0	1	12	1	2	2	1	1	2	9	6	170	6	26	8	0	0
1	6	277	0	109	0	502	22	4	3	2	0	2	9	4	22	1	2	0	6	2	1
2	1	57	10	189	0	668	15	3	4	1	3	1	9	3	11	1	6	1	2	0	2
3	5	46	0	733	0	99	10	3	1	0	3	2	64	2	8	0	3	1	2	0	3
4	6	49	0	650	0	103	36	3	6	5	2	1	84	2	7	1	4	1	3	0	4
5	1	78	0	21	0	831	21	6	2	2	1	3	6	0	9	2	3	0	1	1	5
6	3	75	0	232	20	57	98	45	5	7	7	2	388	6	11	5	10	0	4	0	6
7	2	43	0	3	0	9	49	759	47	4	8	3	18	14	8	4	13	1	5	0	7
8	2	17	0	13	0	5	41	65	769	3	4	3	35	11	6	5	4	1	12	0	8
9	3	17	0	4	0	0	41	20	2	830	44	1	1	4	6	6	6	0	9	0	9
10	1	15	0	2	0	0	29	1	3	22	908	0	2	2	3	1	4	4	2	0	10
11	5	41	1	6	0	26	25	0	1	0	0	813	5	5	1	1	56	1	4	0	11
12	10	175	0	140	0	41	48	46	10	2	0	13	426	35	25	2	3	1	5	2	12
13	32	74	0	6	0	19	33	4	12	0	1	1	21	683	59	7	9	0	28	1	13
14	6	37	0	3	0	21	20	15	4	2	3	4	13	8	813	4	16	7	11	0	14
15	30	27	0	2	0	6	12	0	3	0	2	3	1	14	8	840	8	17	22	2	15
16	5	14	0	5	0	2	48	1	9	2	3	10	3	5	7	10	752	4	30	0	16
17	10	14	0	1	0	3	21	2	3	2	1	2	3	4	4	9	20	604	31	206	17
18	13	14	0	0	0	6	25	3	9	2	1	5	2	35	19	7	271	23	338	2	18
19	147	10	0	2	0	6	16	1	3	5	0	0	1	3	5	276	80	8	64	1	19

Figure 13: x-axis is the predicted labels by the k-means clustering with UMAP(n_component=200, cosine distance). Y-axis is the true labels.

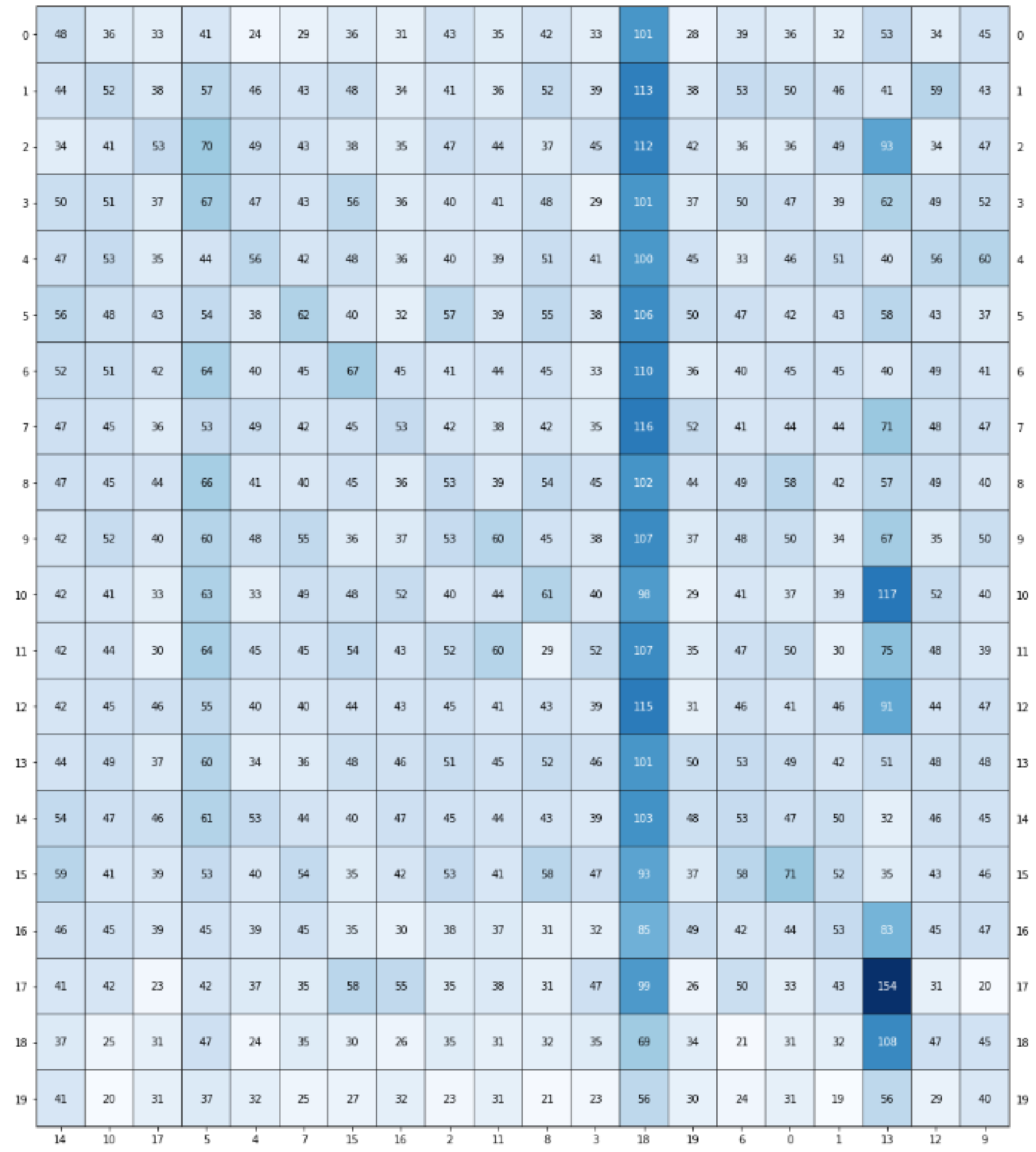
```
*** Permuted Conf. Mat. for n component: 5, dist: euclidean : ***
```



	14	10	17	5	4	7	15	16	2	11	8	3	18	19	6	0	1	13	12	9	
0	53	40	40	48	44	42	31	89	37	36	41	34	29	36	31	39	34	40	12	43	0
1	53	50	50	62	40	45	49	103	51	58	47	39	39	54	45	51	49	23	12	53	1
2	36	34	61	60	42	51	34	102	63	49	38	50	54	35	56	37	47	50	38	48	2
3	52	47	39	72	53	47	45	91	44	56	53	45	41	52	48	48	45	8	49	47	3
4	54	38	52	52	58	44	41	87	64	54	52	40	44	55	52	47	42	14	26	47	4
5	57	44	44	62	39	64	39	94	45	53	53	42	48	45	54	43	57	15	43	47	5
6	59	35	45	70	41	46	59	104	48	55	52	36	42	54	43	47	48	7	25	59	6
7	53	39	47	51	44	42	43	125	45	52	51	47	46	43	56	42	49	34	22	59	7
8	51	45	43	66	39	55	43	104	60	49	59	37	52	49	45	59	41	26	20	53	8
9	44	49	36	59	50	60	35	108	50	59	51	55	46	32	47	61	52	36	19	45	9
10	49	38	43	73	41	46	44	84	49	48	66	47	32	50	32	42	45	85	25	60	10
11	46	39	34	60	42	54	53	102	76	51	34	68	41	47	32	51	46	31	40	44	11
12	42	44	54	60	48	44	42	101	49	43	50	38	51	41	36	48	43	48	49	53	12
13	50	41	43	61	45	44	41	92	56	52	51	46	47	59	44	49	49	21	41	58	13
14	60	51	49	65	39	50	35	90	55	56	52	51	40	38	59	48	46	28	11	64	14
15	69	55	51	49	43	57	39	92	62	50	61	38	48	32	35	76	54	19	14	53	15
16	49	41	57	53	46	48	35	83	42	50	33	42	42	47	49	39	49	47	17	41	16
17	44	43	50	57	19	41	49	94	52	42	32	50	31	43	28	39	37	93	47	49	17
18	40	22	36	43	42	32	28	71	39	38	30	30	38	48	29	33	39	30	69	38	18
19	37	24	27	41	39	22	27	49	27	24	29	29	34	26	34	32	27	32	22	46	19

Figure 14: x-axis is the predicted labels by the k-means clustering with UMAP(n_component=5, euclidean distance). Y-axis is the true labels.

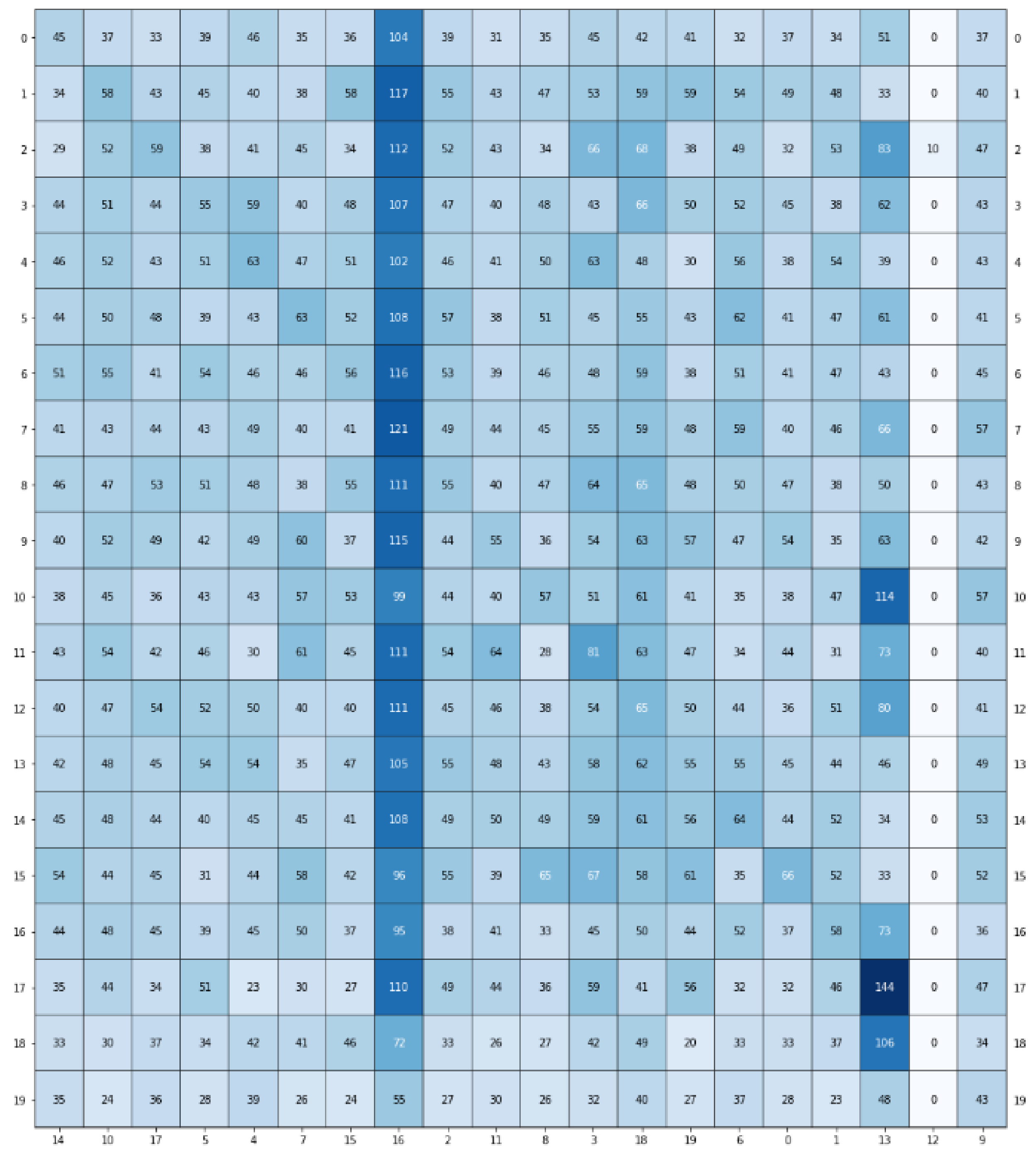
```
*** Permuted Conf. Mat. for n component: 20, dist: euclidean : ***
```



	14	10	17	5	4	7	15	16	2	11	8	3	18	19	6	0	1	13	12	9	
0	48	36	33	41	24	29	36	31	43	35	42	33	101	28	39	36	32	53	34	45	0
1	44	52	38	57	46	43	48	34	41	36	52	39	113	38	53	50	46	41	59	43	1
2	34	41	53	70	49	43	38	35	47	44	37	45	112	42	36	36	49	93	34	47	2
3	50	51	37	67	47	43	56	36	40	41	48	29	101	37	50	47	39	62	49	52	3
4	47	53	35	44	56	42	48	36	40	39	51	41	100	45	33	46	51	40	56	60	4
5	56	48	43	54	38	62	40	32	57	39	55	38	106	50	47	42	43	58	43	37	5
6	52	51	42	64	40	45	67	45	41	44	45	33	110	36	40	45	45	40	49	41	6
7	47	45	36	53	49	42	45	53	42	38	42	35	116	52	41	44	44	71	48	47	7
8	47	45	44	66	41	40	45	36	53	39	54	45	102	44	49	58	42	57	49	40	8
9	42	52	40	60	48	55	36	37	53	60	45	38	107	37	48	50	34	67	35	50	9
10	42	41	33	63	33	49	48	52	40	44	61	40	98	29	41	37	39	117	52	40	10
11	42	44	30	64	45	45	54	43	52	60	29	52	107	35	47	50	30	75	48	39	11
12	42	45	46	55	40	40	44	43	45	41	43	39	115	31	46	41	46	91	44	47	12
13	44	49	37	60	34	36	48	46	51	45	52	46	101	50	53	49	42	51	48	48	13
14	54	47	46	61	53	44	40	47	45	44	43	39	103	48	53	47	50	32	46	45	14
15	59	41	39	53	40	54	35	42	53	41	58	47	93	37	58	71	52	35	43	46	15
16	46	45	39	45	39	45	35	30	38	37	31	32	85	49	42	44	53	83	45	47	16
17	41	42	23	42	37	35	58	55	35	38	31	47	99	26	50	33	43	154	31	20	17
18	37	25	31	47	24	35	30	26	35	31	32	35	69	34	21	31	32	108	47	45	18
19	41	20	31	37	32	25	27	32	23	31	21	23	56	30	24	31	19	56	29	40	19

Figure 15: x-axis is the predicted labels by the k-means clustering with UMAP(n_component=20, euclidean distance). Y-axis is the true labels.

```
*** Permuted Conf. Mat. for n component: 200, dist: euclidean : ***
```



	14	10	17	5	4	7	15	16	2	11	8	3	18	19	6	0	1	13	12	9
0	45	37	33	39	46	35	36	104	39	31	35	45	42	41	32	37	34	51	0	37
1	34	58	43	45	40	38	58	117	55	43	47	53	59	59	54	49	48	33	0	40
2	29	52	59	38	41	45	34	112	52	43	34	66	68	38	49	32	53	83	10	47
3	44	51	44	55	59	40	48	107	47	40	48	43	66	50	52	45	38	62	0	43
4	46	52	43	51	63	47	51	102	46	41	50	63	48	30	56	38	54	39	0	43
5	44	50	48	39	43	63	52	108	57	38	51	45	55	43	62	41	47	61	0	41
6	51	55	41	54	46	46	56	116	53	39	46	48	59	38	51	41	47	43	0	45
7	41	43	44	43	49	40	41	121	49	44	45	55	59	48	59	40	46	66	0	57
8	46	47	53	51	48	38	55	111	55	40	47	64	65	48	50	47	38	50	0	43
9	40	52	49	42	49	60	37	115	44	55	36	54	63	57	47	54	35	63	0	42
10	38	45	36	43	43	57	53	99	44	40	57	51	61	41	35	38	47	114	0	57
11	43	54	42	46	30	61	45	111	54	64	28	81	63	47	34	44	31	73	0	40
12	40	47	54	52	50	40	40	111	45	46	38	54	65	50	44	36	51	80	0	41
13	42	48	45	54	54	35	47	105	55	48	43	58	62	55	55	45	44	46	0	49
14	45	48	44	40	45	45	41	108	49	50	49	59	61	56	64	44	52	34	0	53
15	54	44	45	31	44	58	42	96	55	39	65	67	58	61	35	66	52	33	0	52
16	44	48	45	39	45	50	37	95	38	41	33	45	50	44	52	37	58	73	0	36
17	35	44	34	51	23	30	27	110	49	44	36	59	41	56	32	32	46	144	0	47
18	33	30	37	34	42	41	46	72	33	26	27	42	49	20	33	33	37	106	0	34
19	35	24	36	28	39	26	24	55	27	30	26	32	40	27	37	28	23	48	0	43

Figure 16: x-axis is the predicted labels by the k-means clustering with UMAP(n_component=200, euclidean distance). Y-axis is the true labels.

Question 12: Analyze the contingency matrices. Which setting works best and why? What about for each metric choice?

The best number of components for cosine is 20, which gives an average score of 0.56 when we observe the metrics. When we observe the matrices (Figure 14,15,16), we can again confirm that k-means clustering with euclidean distance fails for UMAP feature reduction. When we observe different matrices for cosine distance, we can see that there is not much difference between different numbers of components, all performed similarly good and match clusters with classes.

We know that cosine distance can bypass the length of documents, which may negatively affect the results as in the euclidean case. For different length, but similar documents, the tf-idf values will be different, euclidean distance cannot make up for this magnitude difference even though documents are similar.

Question 13: TF-IDF representation, PCA-reduced, NMF-reduced, UMAP-reduced
Compare and contrast the clustering results across the 4 choices, and suggest an approach that is best for the K-Means clustering task on the 20-class text data. Choose any choice of clustering metrics for your comparison.

From earlier sections, we observe that the best number of components for SVD is 200, while NMF is 20. It is 20 for UMAP, but since there is not much difference between clustering metrics (we can take average of scores as a metric) of 5 and 20, we used 5 for computation concerns. We can observe from Table 6 that UMAP significantly outperforms other feature methods.

	Homogeneity	Completeness	V-measure	Adjusted rand	Adjusted Mutual Info	Avg. of Scores
TF-IDF	0.36	0.41	0.38	0.13	0.38	0.33
SVD or PCA (n_comp=200)	0.37	0.49	0.42	0.10	0.42	0.36
NMF (n_comp=20)	0.35	0.42	0.38	0.10	0.38	0.33
UMAP(cos.) (n_comp=5)	**0.56**	**0.58**	**0.57**	**0.44**	**0.57**	**0.55**

Table 6: K-means clustering metrics for 4 different input methods.

Question 14: UMAP to reduce the dimensionality properly, and perform Agglomerative clustering with n_clusters=20 . Compare the performance of “ward” and “single” linkage criteria. Report the five clustering evaluation metrics for each case.

All the experiments that we conducted used the k-means clustering method. However, this technique’s capacity is limited since it assumes Gaussian distribution. Another method of clustering is agglomerative clustering, in which hierarchical clustering is performed. Each observation starts with one cluster and they are successively merged into bigger clusters until all clusters are merged.

Here, we examine two different linkage or merging criteria: ward and single. While single linkage utilizes the minimum of the distances between all observations of the two sets, ward linkage minimizes the variance of the clusters being merged. As indicated before UMAP (cosine distance) with n_components=20 performed the best, but we use similarly performing n_components=5 here.

Results are shown in Table 7, and it reveals that Ward linkage clustering significantly outperforms single linkage clusters. This is because single linkage performs poorly when the dataset, though it can perform better with non-globular data. Ward can also work and perform well with noisy data unlike single linkage.

Linkage	Homogeneit y	Completeness	V-measure	Adjusted rand	Adjusted Mutual Info	Avg. of Scores
Ward	**0.55**	**0.58**	**0.57**	**0.42**	**0.57**	**0.54**
Single	0.02	0.39	0.03	0.00	0.03	0.09

Table 7: Agglomerative clustering metrics for 2 different linkage criteria with UMAP (n_component=5, distance=cosine) reduction.

Question 15: HDBSCAN on UMAP-transformed 20-category data. Vary the min cluster size among 20, 100, 200 and report your findings in terms of the five clustering evaluation metrics.

In order to evaluate the clustering metrics for HDBSCAN we analyzed two other important parameters other than min. cluster size: min. sample size and epsilon for cluster selection. Min. sample size is the number of samples in a neighborhood for a point to be considered a core point, and epsilon is a distance threshold (clusters below this value will be merged). Additionally, unlike the other clustering methods we have covered so far, this method does not require the number of clusters. Unlike distance-based k-means clustering, density-based HDBSCAN can cluster varying density data distributions, and it can handle noise. Unlike DBSCAN, HDBSCAN is a hierarchical clustering.

Full parameters are shown below. We analyze the clustering metrics for each pair of parameters, and we experimentally found that for min. cluster size of 20, epsilon of 0.5 and min. sample size of 40 gives the best outcome. For min. cluster size of 100 and 200, epsilon of 0.05 and min. sample size of 20 gives the best outcome.

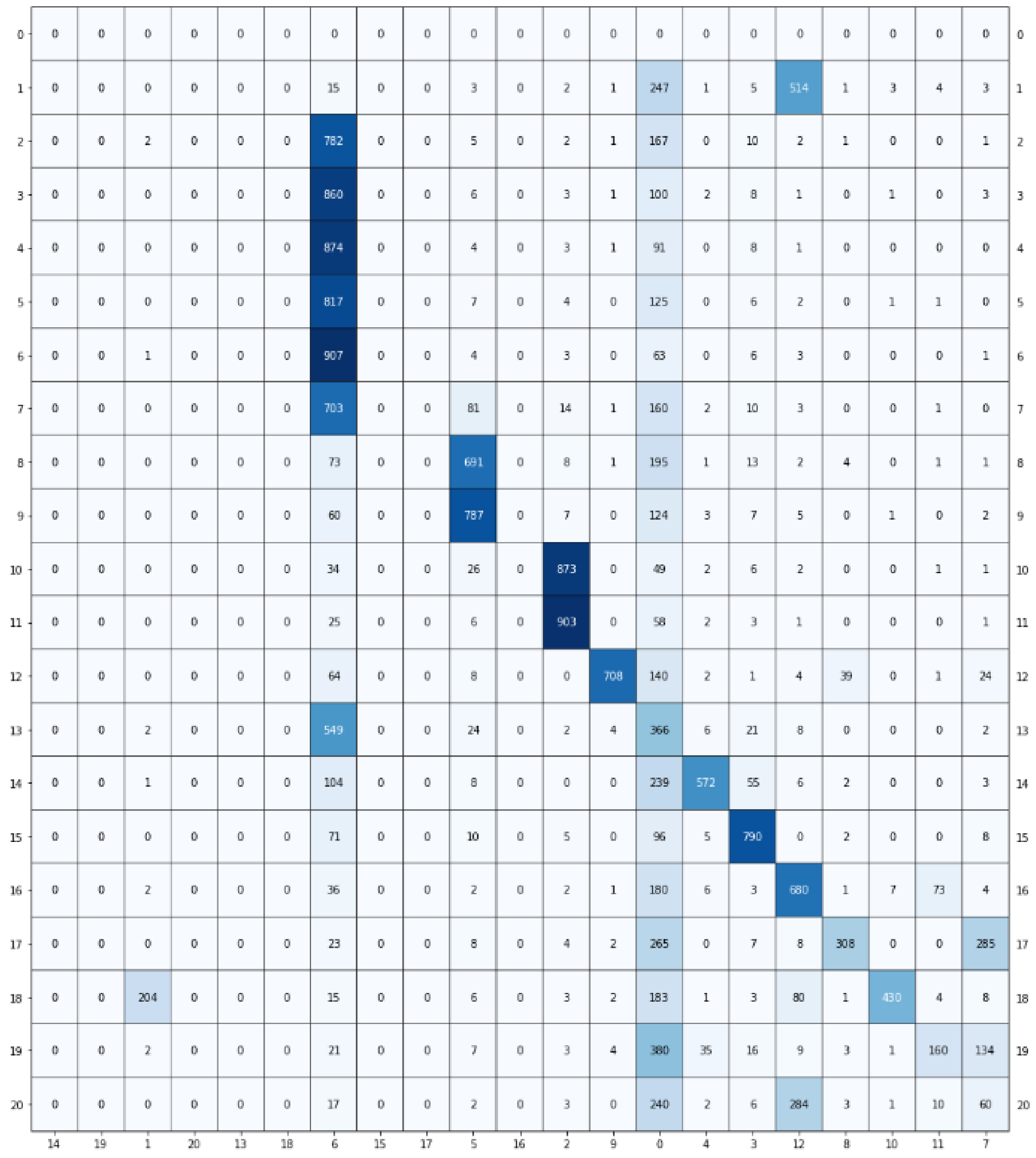
```
hdbscan_min_cluster_size_list = [ 20, 100, 200 ]
hdbscan_min_sample_list = [5, 20, 30, 40, 50, 60, 70, 80, 160, 500]
hdbscan_epsilon_list = [0.05, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 1.0,
2.0, 5]
```

Min. cluster size	Homogeneity	Completeness	V-measure	Adjusted rand	Adjusted Mutual Info	Avg. of Scores
20 (eps=0.5, min_sample=40)	0.41	0.59	0.49	**0.22**	0.48	0.440
100 (eps=0.05, min_sample=20)	**0.42**	**0.60**	**0.49**	0.21	**0.49**	**0.442**
200 (eps=0.05, min_sample=20)	**0.42**	**0.60**	**0.49**	0.21	**0.49**	**0.442**

Table 8: Agglomerative clustering metrics for 2 different linkage criteria with UMAP (n_component=5, distance=cosine) reduction.

Question 16: Plot the contingency matrix for the best clustering model from Question 15. How many clusters are given by the model? What does “-1” mean for the clustering labels? Interpret the contingency matrix considering the answer to these questions.

We can observe from Table 8 that there is no difference between min cluster size of 100 and 200 for the clustering metrics, and they slightly outperform min. cluster size of 20. For the min. Cluster size of 100 (best), contingency matrix is shown in Figure 17. It performed moderately good, while performing worse than k-means with UMAP. This may be due to the fact that there are -1 labels in the cluster labels. This -1 label means noise points in HDBSCAN algorithm, and they do not belong to any cluster. There are 12 unique clusters predicted by the best HDBSCAN clustering, and there are 3468 noise points.



	14	19	1	20	13	18	6	15	17	5	16	2	9	0	4	3	12	8	10	11	7	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	15	0	0	3	0	2	1	247	1	5	514	1	3	4	3	1
2	0	0	2	0	0	0	782	0	0	5	0	2	1	167	0	10	2	1	0	0	1	2
3	0	0	0	0	0	0	860	0	0	6	0	3	1	100	2	8	1	0	1	0	3	3
4	0	0	0	0	0	0	874	0	0	4	0	3	1	91	0	8	1	0	0	0	0	4
5	0	0	0	0	0	0	817	0	0	7	0	4	0	125	0	6	2	0	1	1	0	5
6	0	0	1	0	0	0	907	0	0	4	0	3	0	63	0	6	3	0	0	0	1	6
7	0	0	0	0	0	0	703	0	0	81	0	14	1	160	2	10	3	0	0	1	0	7
8	0	0	0	0	0	0	73	0	0	691	0	8	1	195	1	13	2	4	0	1	1	8
9	0	0	0	0	0	0	60	0	0	787	0	7	0	124	3	7	5	0	1	0	2	9
10	0	0	0	0	0	0	34	0	0	26	0	873	0	49	2	6	2	0	0	1	1	10
11	0	0	0	0	0	0	25	0	0	6	0	903	0	58	2	3	1	0	0	0	1	11
12	0	0	0	0	0	0	64	0	0	8	0	0	708	140	2	1	4	39	0	1	24	12
13	0	0	2	0	0	0	549	0	0	24	0	2	4	366	6	21	8	0	0	0	2	13
14	0	0	1	0	0	0	104	0	0	8	0	0	0	239	572	55	6	2	0	0	3	14
15	0	0	0	0	0	0	71	0	0	10	0	5	0	96	5	790	0	2	0	0	8	15
16	0	0	2	0	0	0	36	0	0	2	0	2	1	180	6	3	680	1	7	73	4	16
17	0	0	0	0	0	0	23	0	0	8	0	4	2	265	0	7	8	308	0	0	285	17
18	0	0	204	0	0	0	15	0	0	6	0	3	2	183	1	3	80	1	430	4	8	18
19	0	0	2	0	0	0	21	0	0	7	0	3	4	380	35	16	9	3	1	160	134	19
20	0	0	0	0	0	0	17	0	0	2	0	3	0	240	2	6	284	3	1	10	60	20

Figure 17: x-axis is the predicted labels by the HDBSCAN clustering with UMAP (n_component=5, distance=cosine) reduction. Y-axis is the true labels.

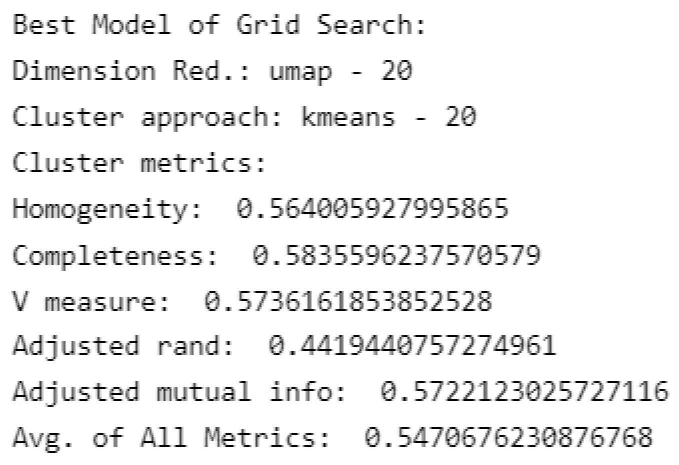
Question 17: Based on your experiments, which dimensionality reduction technique and clustering methods worked best together for 20-class text data and why?

We conducted a grid search with the parameters given in Table 9. The best performing system is evaluated with the average of all metrics. Figure 18 reveals that the best clustering method with our data is k-means with 20 clusters, and the best reduction method is UMAP with n_components=20. Best average score is 0.55 after the search.

These results are no surprise because we already discussed the effectiveness of the cosine distance metric of UMAP compared to SVD and NMF. Tf-idf matrix by itself cannot perform as the best due to the curse of dimensionality. Since the data has 20 different true classes, we expect that dataset inherently may have the number clusters close to 20. This can be the reason why k-means with 20 clusters performed better than other cluster numbers. There may be several reasons why k-means performed better than other clustering methods. Underlying complexity of the dataset can be low and the inductive biases such as Gaussian assumption of k-means, similar density of data points can be present in the data.

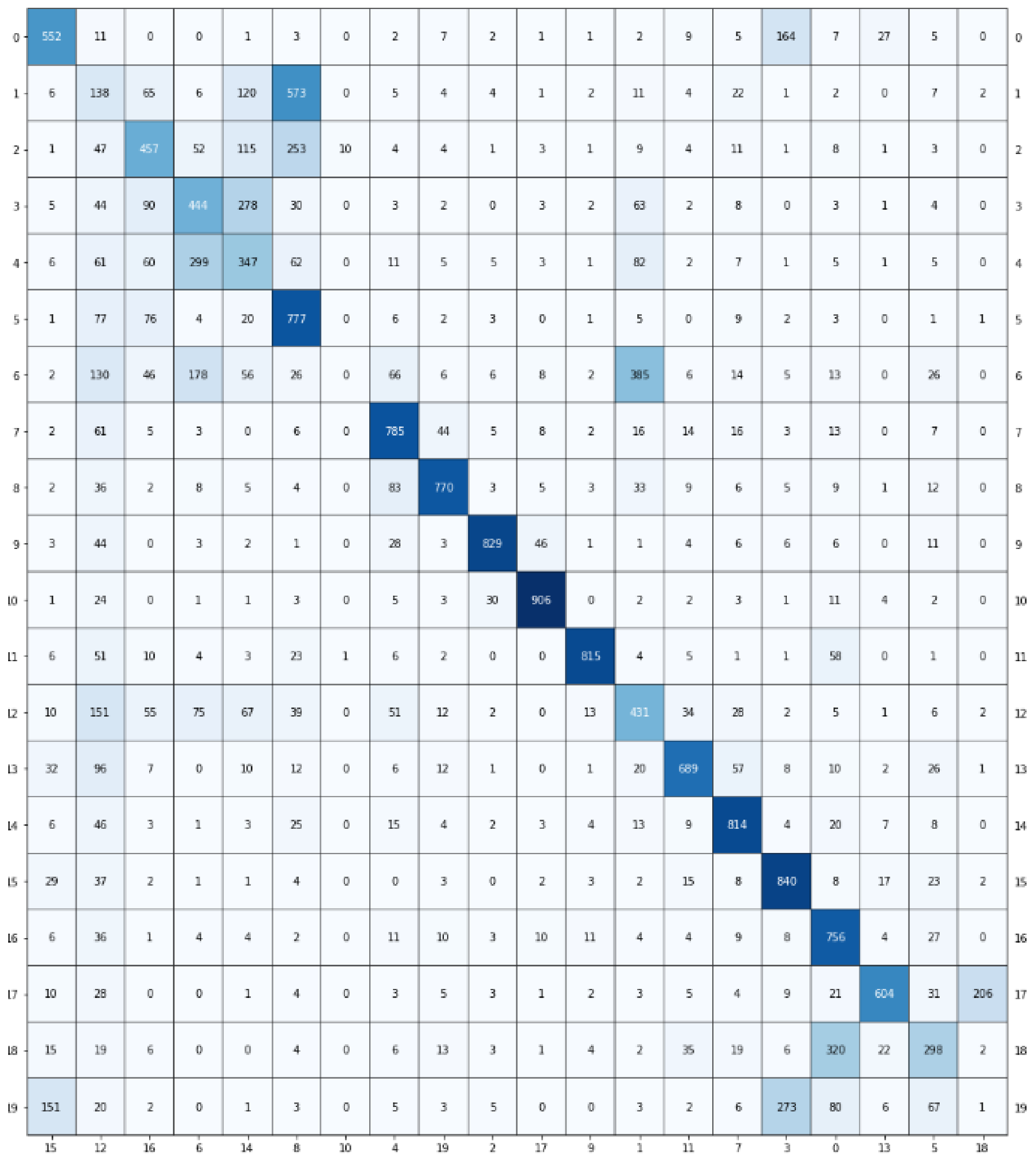
Module	**Alternatives**	**Hyperparameters**
Dimensionality Reduction	None	N/A
	SVD	r = [5,20,200]
	NMF	r = [5,20,200]
	UMAP	n_components = [5,20,200]
Clustering	K-Means	k = [10,20,50]
	Agglomerative Clustering	n_clusters = [20]
	HDBSCAN	min_cluster_size = [100,200]

Table 9: Hyperparameter grid for clustering of question 17.



```
Best Model of Grid Search:
Dimension Red.: umap - 20
Cluster approach: kmeans - 20
Cluster metrics:
Homogeneity:  0.564005927995865
Completeness:  0.5835596237570579
V measure:  0.5736161853852528
Adjusted rand:  0.4419440757274961
Adjusted mutual info:  0.5722123025727116
Avg. of All Metrics:  0.5470676230876768
```

Figure 18: Best performing (avg. score of all metrics) system and its metrics after grid search.



	15	12	16	6	14	8	10	4	19	2	17	9	1	11	7	3	0	13	5	18	
0	552	11	0	0	1	3	0	2	7	2	1	1	2	9	5	164	7	27	5	0	0
1	6	138	65	6	120	573	0	5	4	4	1	2	11	4	22	1	2	0	7	2	1
2	1	47	457	52	115	253	10	4	4	1	3	1	9	4	11	1	8	1	3	0	2
3	5	44	90	444	278	30	0	3	2	0	3	2	63	2	8	0	3	1	4	0	3
4	6	61	60	299	347	62	0	11	5	5	3	1	82	2	7	1	5	1	5	0	4
5	1	77	76	4	20	777	0	6	2	3	0	1	5	0	9	2	3	0	1	1	5
6	2	130	46	178	56	26	0	66	6	6	8	2	385	6	14	5	13	0	26	0	6
7	2	61	5	3	0	6	0	785	44	5	8	2	16	14	16	3	13	0	7	0	7
8	2	36	2	8	5	4	0	83	770	3	5	3	33	9	6	5	9	1	12	0	8
9	3	44	0	3	2	1	0	28	3	829	46	1	1	4	6	6	6	0	11	0	9
10	1	24	0	1	1	3	0	5	3	30	906	0	2	2	3	1	11	4	2	0	10
11	6	51	10	4	3	23	1	6	2	0	0	815	4	5	1	1	58	0	1	0	11
12	10	151	55	75	67	39	0	51	12	2	0	13	431	34	28	2	5	1	6	2	12
13	32	96	7	0	10	12	0	6	12	1	0	1	20	689	57	8	10	2	26	1	13
14	6	46	3	1	3	25	0	15	4	2	3	4	13	9	814	4	20	7	8	0	14
15	29	37	2	1	1	4	0	0	3	0	2	3	2	15	8	840	8	17	23	2	15
16	6	36	1	4	4	2	0	11	10	3	10	11	4	4	9	8	756	4	27	0	16
17	10	28	0	0	1	4	0	3	5	3	1	2	3	5	4	9	21	604	31	206	17
18	15	19	6	0	0	4	0	6	13	3	1	4	2	35	19	6	320	22	298	2	18
19	151	20	2	0	1	3	0	5	3	5	0	0	3	2	6	273	80	6	67	1	19

Figure 19: Best performing system’s contingency matrix after grid search.

Question 19

The main task of a generalized neural network is to make sense of the common features of each class. The VGG Neural Network is trained with a vast amount of data and what it really learns is not the data itself but its features. These features are not restricted to a specific class but they are the representations of a family of distributions. Since the dataset on which VGG is trained and the custom dataset have similar underlying data distribution it is expected that the VGG can capture these features.

Question 20

The Feature Extractor first passes the input vector x from a set of VGG features (predetermined weights of VGG) and then it compresses the obtained latent representation with pooling operations. Finally, the flattened latent vector is subject to the last fully connected layer of VGG in order to obtain more accurate results from the VGG operations.

Question 21

All of them have 224x224x3 RGB images but the feature extractor module compresses the features into a single dimensional vector with length = 4096

Question 22

The extracted features have pooling and convolution operations thus they contain information from several pixels. In a sense, they are the non-linear combination of these pixels. We checked that there are no zero values on the features thus they are dense. On the other hand, TF-IDF vectors were sparse.

Question 23

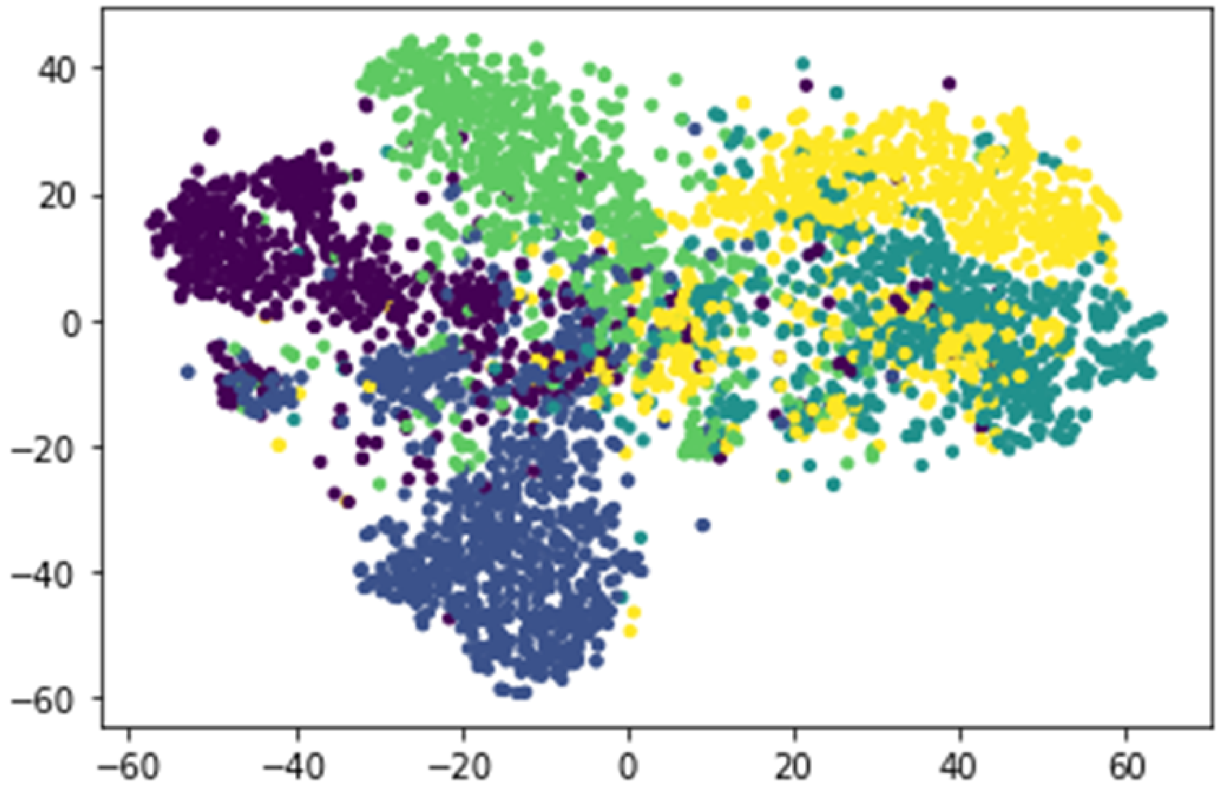


Figure 20: The 2D reduced distributions of the 5 different classes of flowers.

As clearly visible from the figure above, the classes form a cluster, however, it should be noted that the non-linearity in the cluster formation is observed.

Question 24

Rand Score/Feature Extractor	**None**	**SVD**	**UMAP**	**Autoencoder**
K Means	0.1873	0.1897	**0.3968**	0.1943
Agglomerative Clustering	0.1885	0.2136	**0.3716**	0.1838

HDBSCAN	0.015(Cluster Size = 10, Min Sample = 1)	0.023(Cluster Size = 5, Min Sample = 2)	0.031(Cluster Size = 10, Min Sample = 1)	0.0254(Cluster Size = 5, Min Sample = 2)

Table 10: Experiments on different clustering schemes and dimension reduction techniques

We should note that UMAP captures the best dependencies and relations between the samples thus it consistently outperforms in all of the clustering schemes. The performance of the autoencoder is highly dependent upon its structure and its training details. The SVD technique is inherently weak in the non-linear classification tasks as in the flowers dataset that we can observe from the cluster formation in Question 23.

Question 25

The dataset is split into 3000/670 training and test samples. The Multi Layer Perceptron was employed for the whole set of features and the resulting output is reduced to 1 dimensional vector by classifying each subject with the most probable output among five class weights or probabilities. From this scheme we obtain **0.839** adjusted rand score. The same scheme was employed for the autoencoder outputs. This compressed input has a slightly worse performance but still outperforms the clustering schemes with a score of **0.774** for the adjusted rand score.