Project 2: Data Representations and Clustering

Team Members:
Rachel Menezes: 905508013
Diplav : 605627748
Evyn Chiappe : 605201321

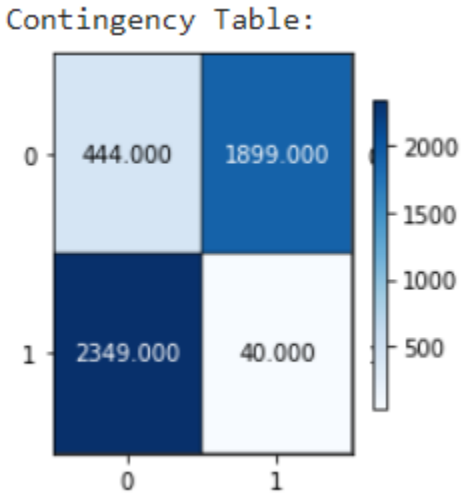
Part 1 - Clustering on Text Data

Clustering with Sparse Text Representations

QUESTION 1: Report the dimensions of the TF-IDF matrix you obtain.
ANS: The TF-IDF matrix has 4732 rows (documents) and 17131 columns (features/words).

QUESTION 2: Report the contingency table of your clustering result. You may use the provided plotmat.py to visualize the matrix. Does the contingency matrix have to be square-shaped?

ANS: The contingency table is reported below.



The contingency table does not have to be square-shaped. The shape depends on the number of unique class labels and the number of unique predicted cluster labels. If there are n unique class labels and m unique predicted cluster labels, the contingency table will have n rows and m columns and if $n \neq m$, the contingency table will be rectangular. In the above case $n = m$ hence it is a square matrix.

QUESTION 3: Report the 5 clustering measures explained in the introduction for K-Means clustering.

ANS:

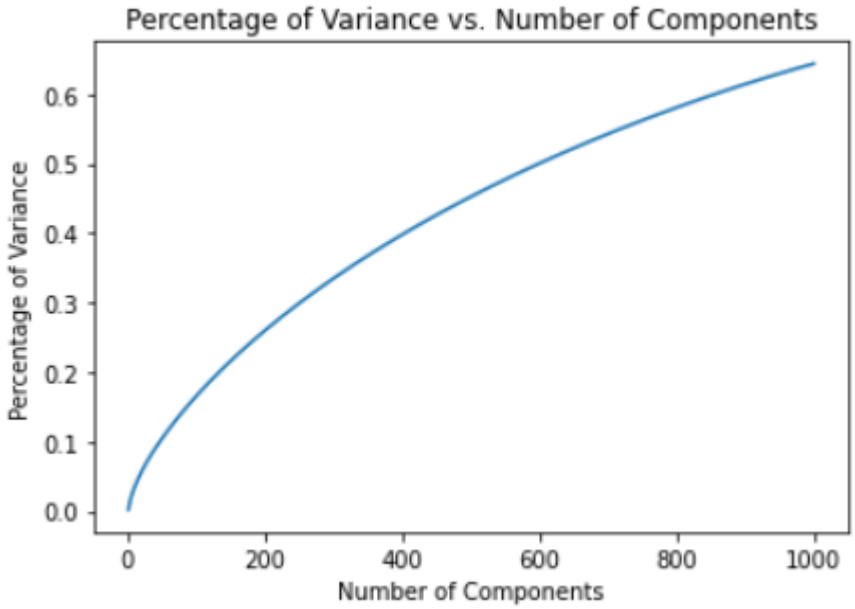
Clustering Measures:

* Homogeneity: 0.590
* Completeness: 0.601
* V-Measure: 0.595
* Adjusted Rand Index: 0.660
* Adjusted Mutual Information: 0.595

Clustering with Dense Text Representations

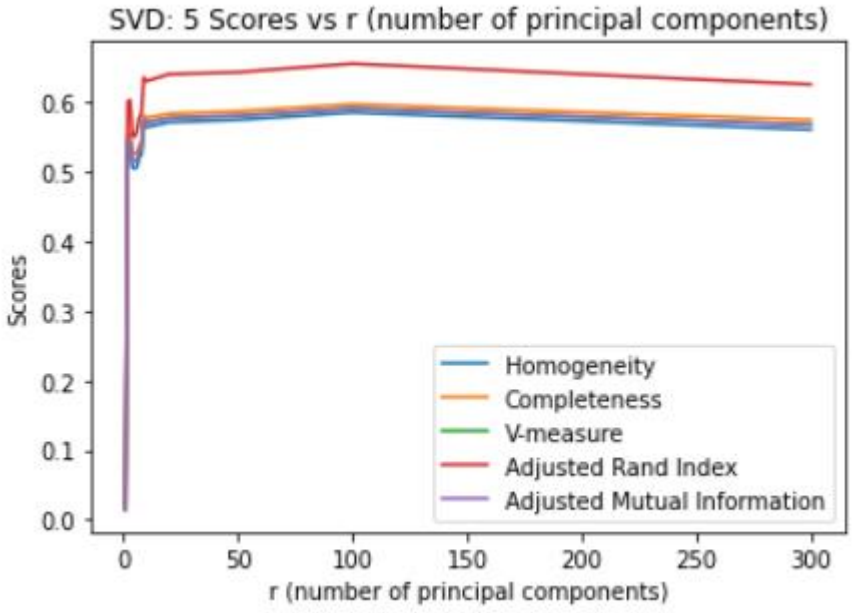
QUESTION 4: Report the plot of the percentage of variance that the top r principle components retain v.s. r, for r = 1 to 1000.

ANS:

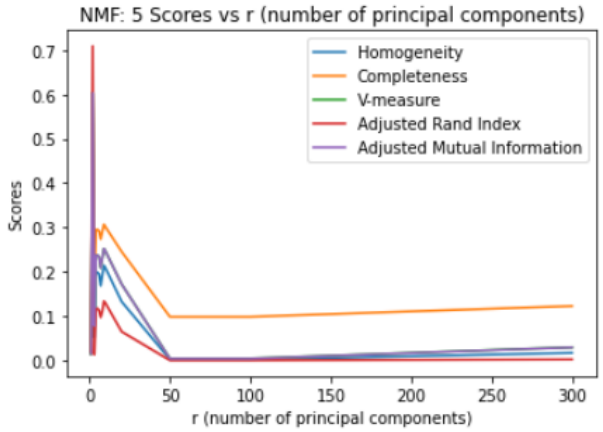


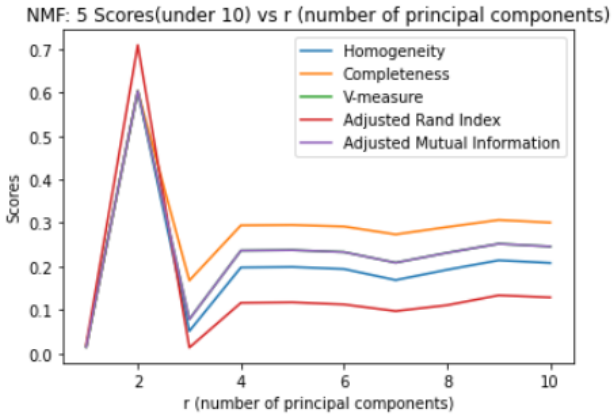
QUESTION 5: Let r be the dimension that we want to reduce the data to (i.e. n components). Try r = 1 – 10, 20, 50, 100, 300, and plot the 5 measure scores v.s. r for both SVD and NMF. Report a good choice of r for SVD and NMF respectively. Note: In the choice of r, there is a trade-off between the information preservation, and better performance of k-means in lower dimensions.

ANS: From the graph for SVD: 5 Scores vs r (number of principal components), the best choice for r is 100. As we can see the scores are nearly the same (horizontal plateau)



From the graph for NMF: 5 Scores vs r (number of principal components), the best choice for r is 2 (magnified the graph on the right to identify r = 2





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

ANS: A non-monotonic behavior is observed in both the graphs as r increases. The scores first increase, then fall and plateau off in the end as r increases. As the number of components increases, the dimensions from which k-means needs to perform

clustering increases and k-means suffers due to the well known curse of dimensionality. It makes it difficult to cluster points as the euclidean distance is no longer a good metric because the ratio of euclidean distances between nearest and farthest point from the cluster center approaches 1. Thus, increasing the number of features beyond the elbow point does not add new information in the clustering task and the measures remain constant and eventually plateau.

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

ANS: The results without data compression and after data compression using SVD and NMF with n_components as 100 and 2 respectively shows that NMF performs better than SVD. The results of SVD with r-100 are similar to those of TFIDF

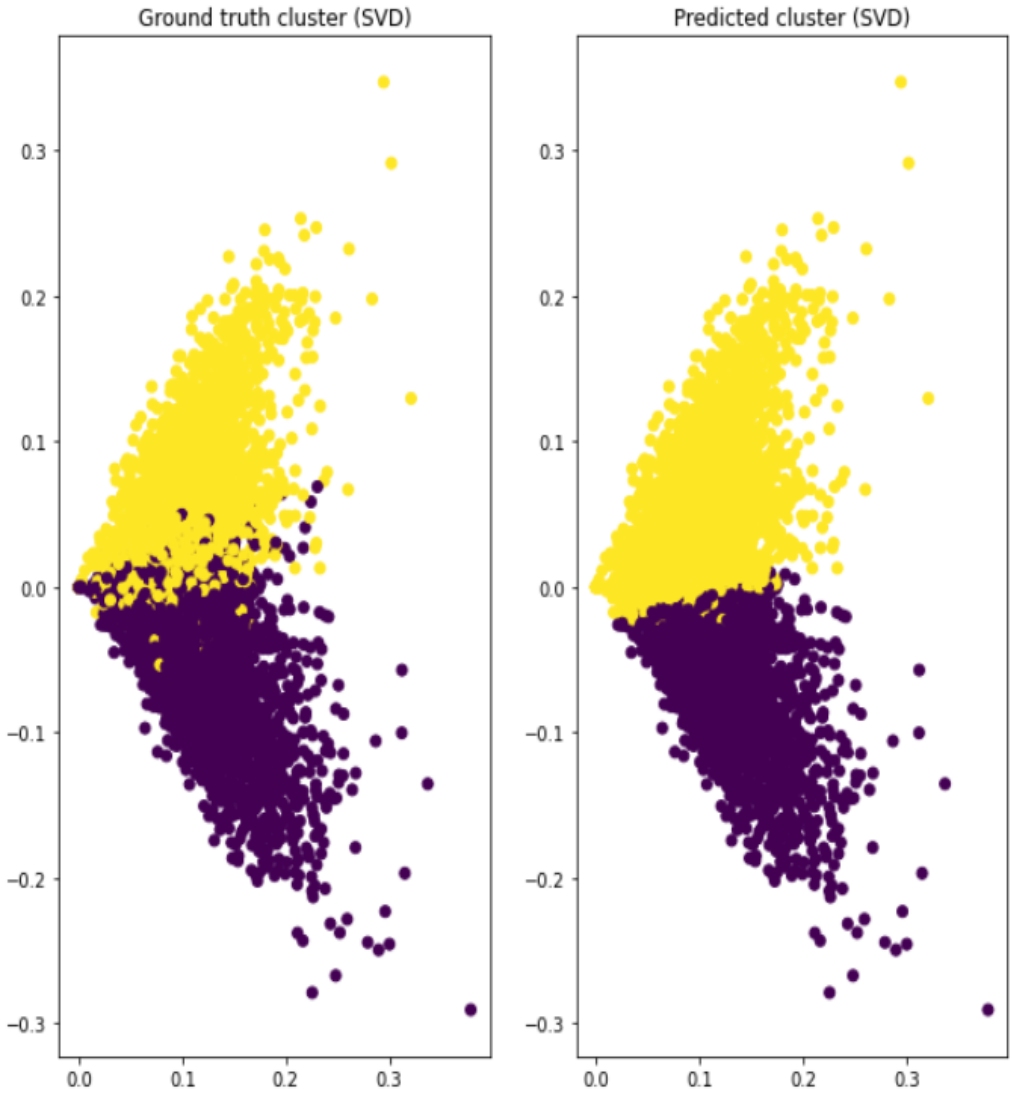
	Homogeneity	**Completeness**	**V-Measure**	**Adjusted Rand Index**	**Adjusted Mutual Information**
TF-IDF	0.590	0.601	0.595	0.660	0.595
SVD r=100	0.573	0.586	0.580	0.641	0.579
NMF r=2	0.603	0.604	0.603	0.709	0.603

QUESTION 8: Visualize the clustering results for:

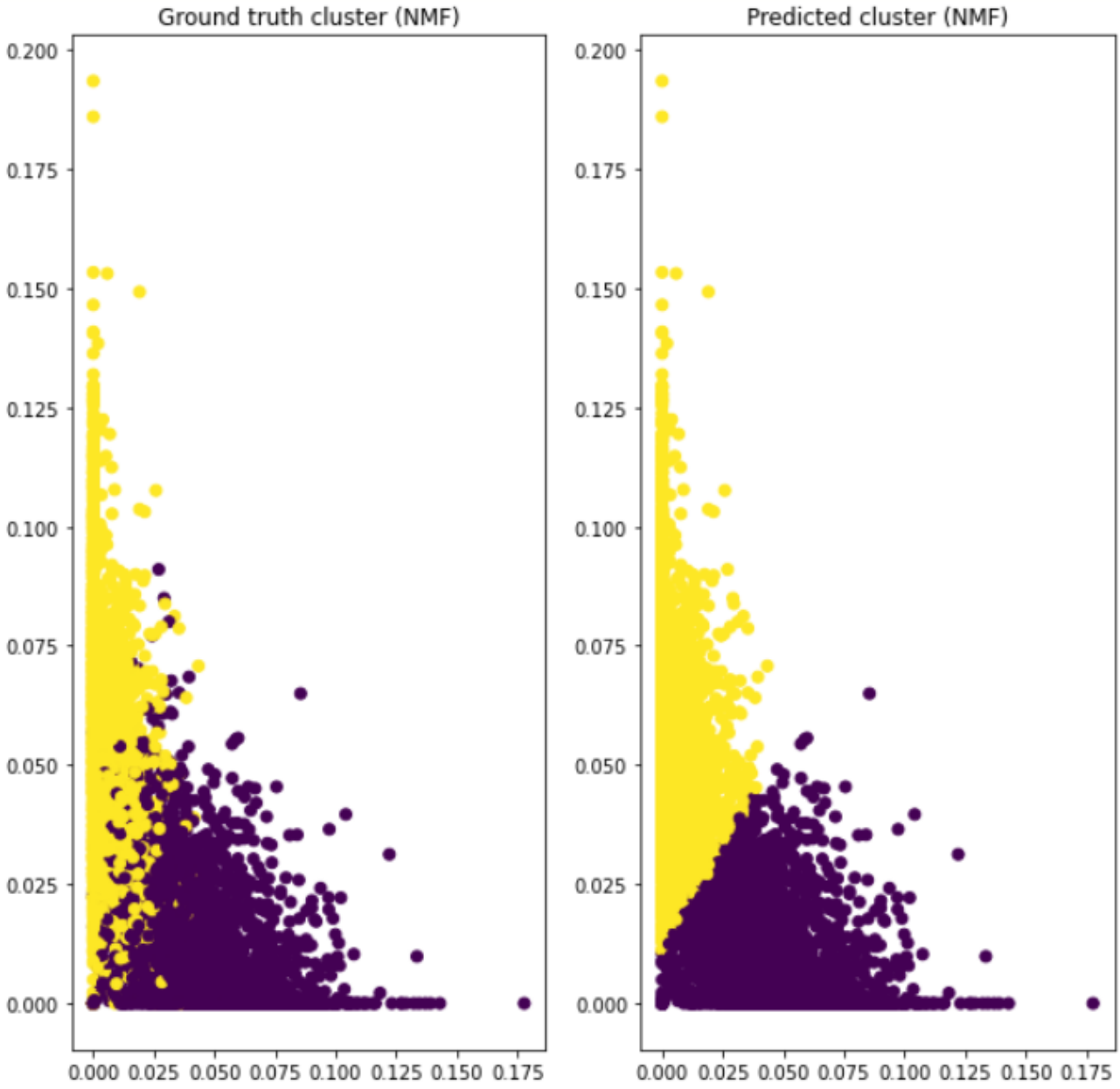
- SVD with your optimal choice of r for K-Means clustering;
- NMF with your choice of r for K-Means clustering.

ANS: The 2 graphs are shown below.

- SVD with your optimal choice of r (100) for K-Means clustering



- NMF with your choice of r (2) for K-Means clustering.



QUESTION 9: What do you observe in the visualization? How are the data points of the two classes distributed? Is distribution of the data ideal for K-Means clustering?

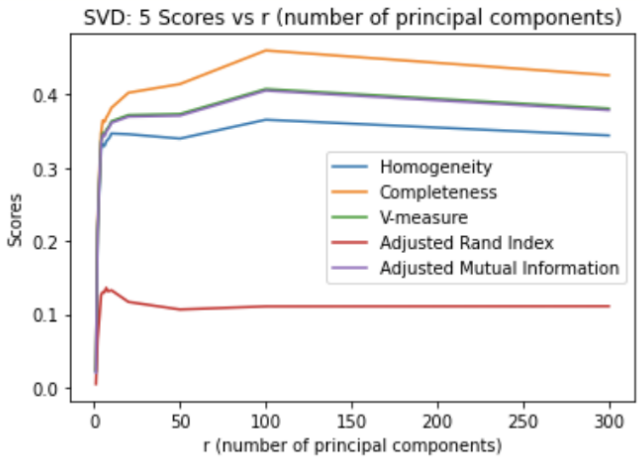
ANS: In both the visualizations, the data is linearly separable & the k-means model performs well on the given data. I believe the distribution is ideal for k-means clustering.

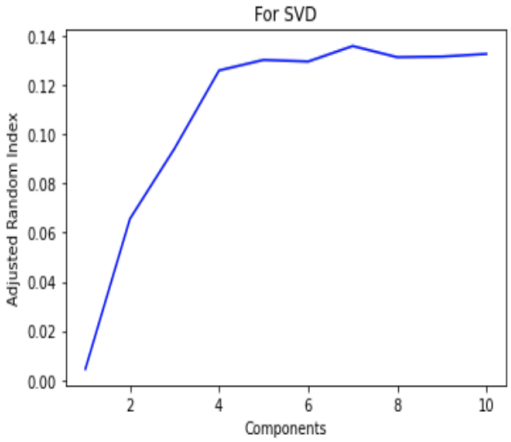
3. Clustering of the Entire 20 Classes

QUESTION 10: Load documents with the same configuration as in Question 1, but for ALL 20 categories. 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).

ANS:

For SVD: From the graph for SVD: 5 Scores vs r (number of principal components), the best choice for r is 100. The cluster scores are first increasing and then decreasing after r = 100. The Adjusted Random index takes its best value at r = 7 but the value is low for other scores.

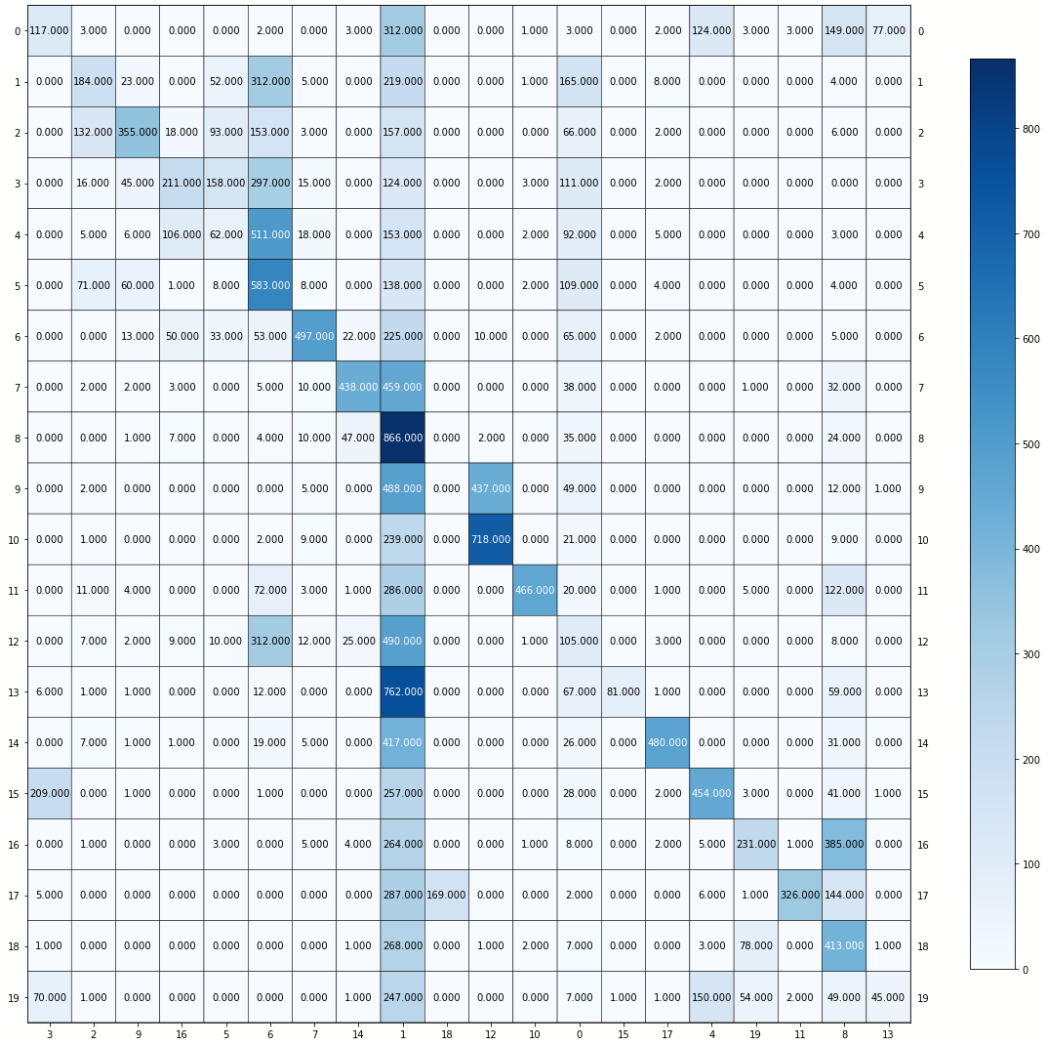




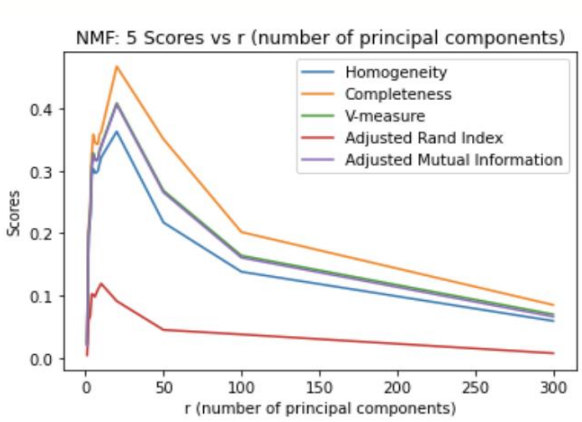
5 clustering and contingency matrix for r = 100 is mentioned below

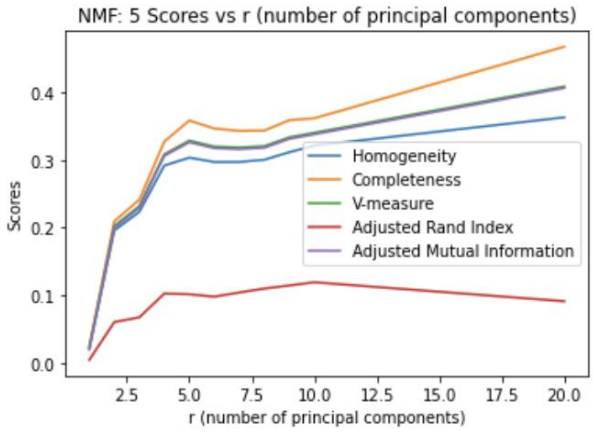
Clustering Measures:

* Homogeneity: 0.3657
* Completeness: 0.4604
* V-Measure: 0.4076
* Adjusted Rand Index: 0.110
* Adjusted Mutual Information: 0.4055



For NMF: 5 Scores vs r (number of principal components), the best choice for r is 20 (magnified the graph on the right to identify r = 20)

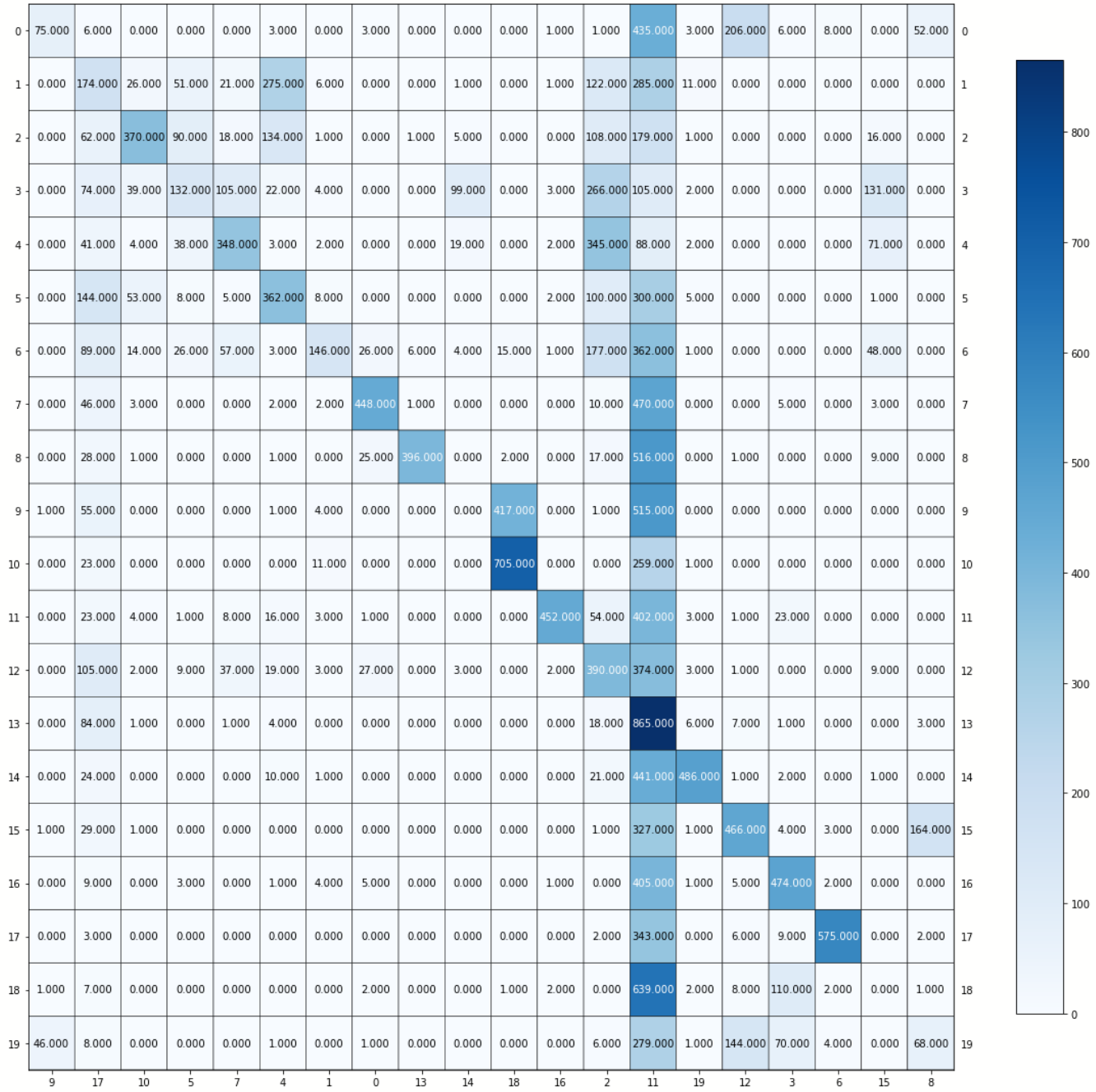




5 clustering and contingency matrix for r = 20 is mentioned below

Clustering Measures:

* Homogeneity: 0.362
* Completeness: 0.4673
* V-Measure: 0.408
* Adjusted Rand Index: 0.0911
* Adjusted Mutual Information: 0.4063



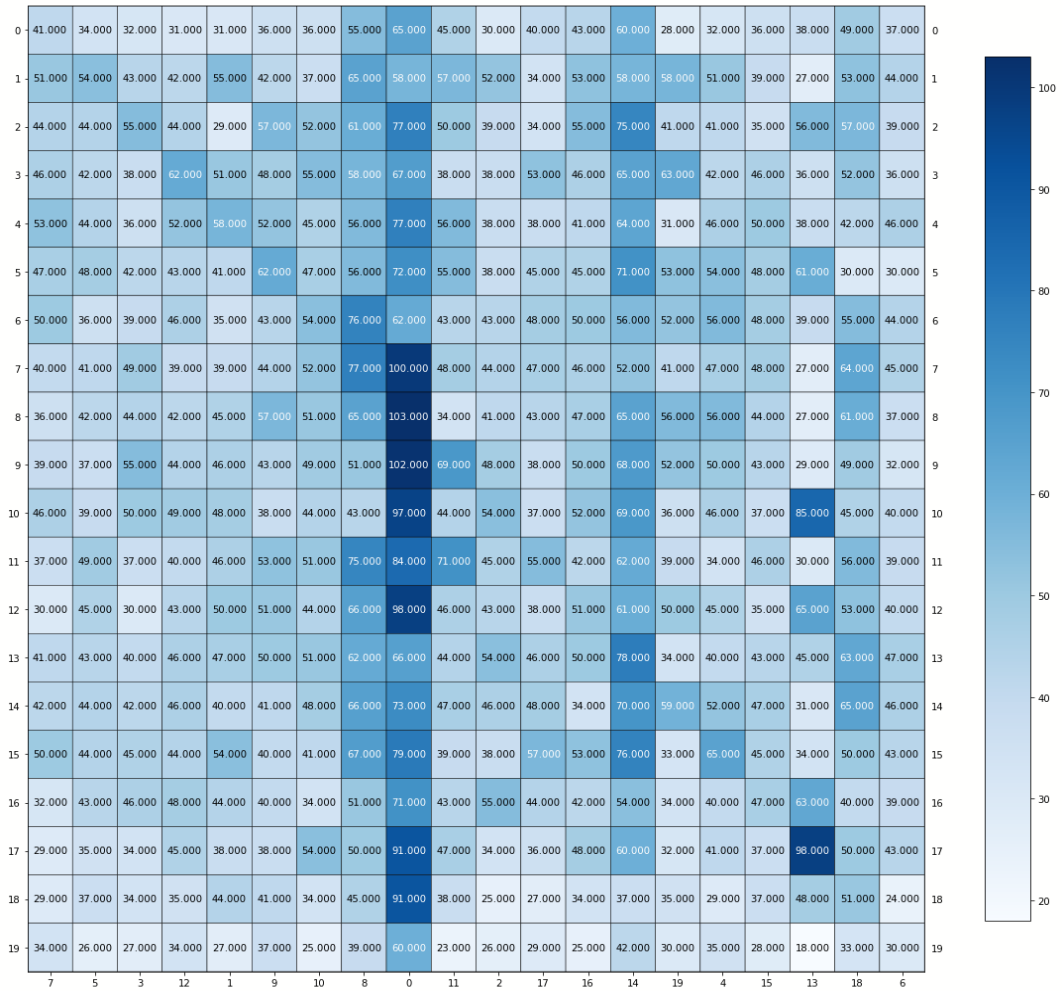
4. UMAP

QUESTION 11: Reduce the dimension of your dataset with UMAP. Consider the following settings: n components = [5, 20, 200], metric = ”cosine” vs. ”euclidean”. If ”cosine” metric fails, please look at the FAQ at the end of this spec.Report the permuted contingency matrix and the five clustering evaluation metrics for the different combinations (6 combinations).

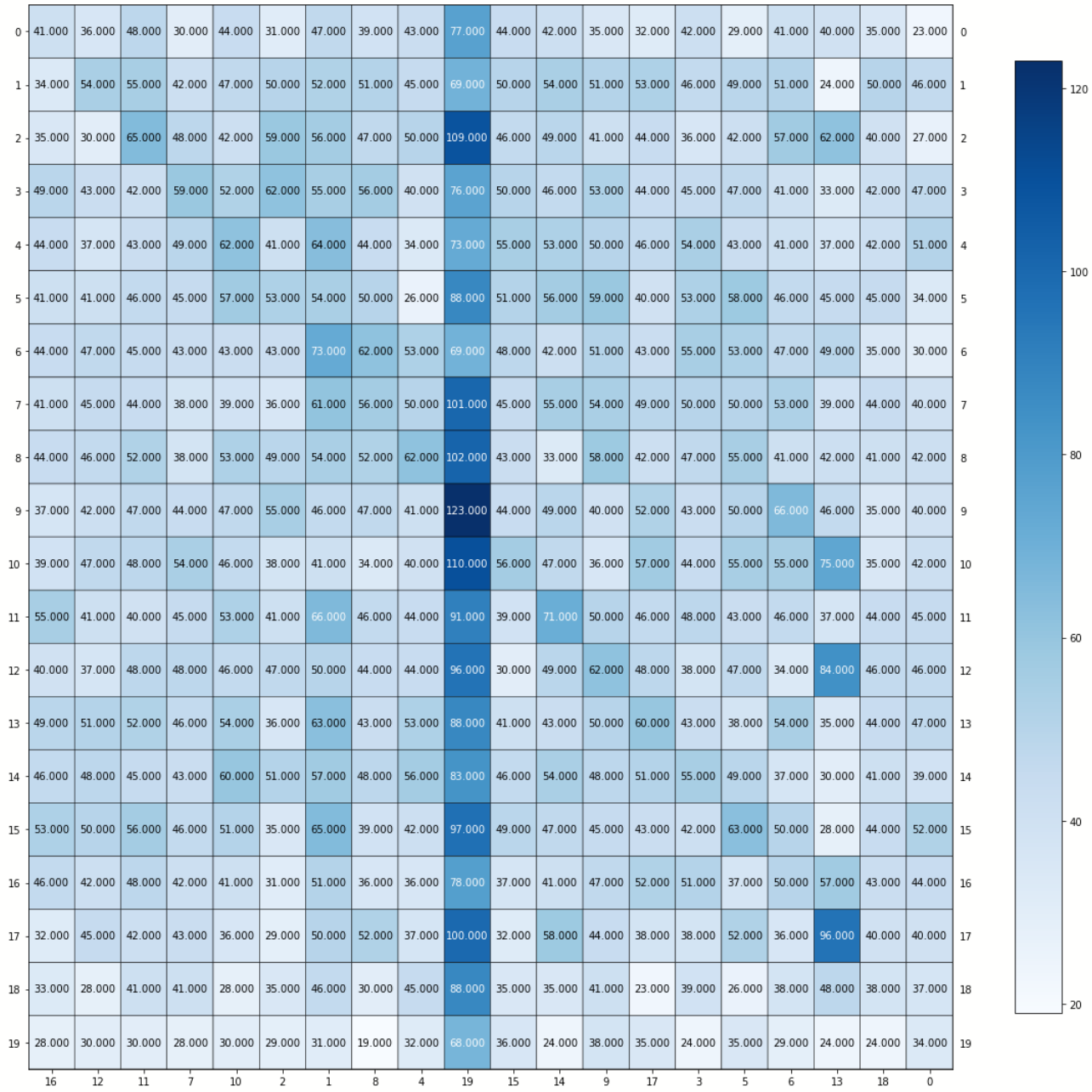
ANS: For Euclidean: 5 Scores for Umap with Euclidean metric for different number of components

	Num of Components	Homogeneity	Completeness	Vmeasure	ARI	AMIS
0	5.0	0.005090	0.005111	0.005101	0.000662	0.001884
1	20.0	0.004786	0.004808	0.004797	0.000553	0.001578
2	200.0	0.005313	0.005532	0.005420	0.000662	0.002144

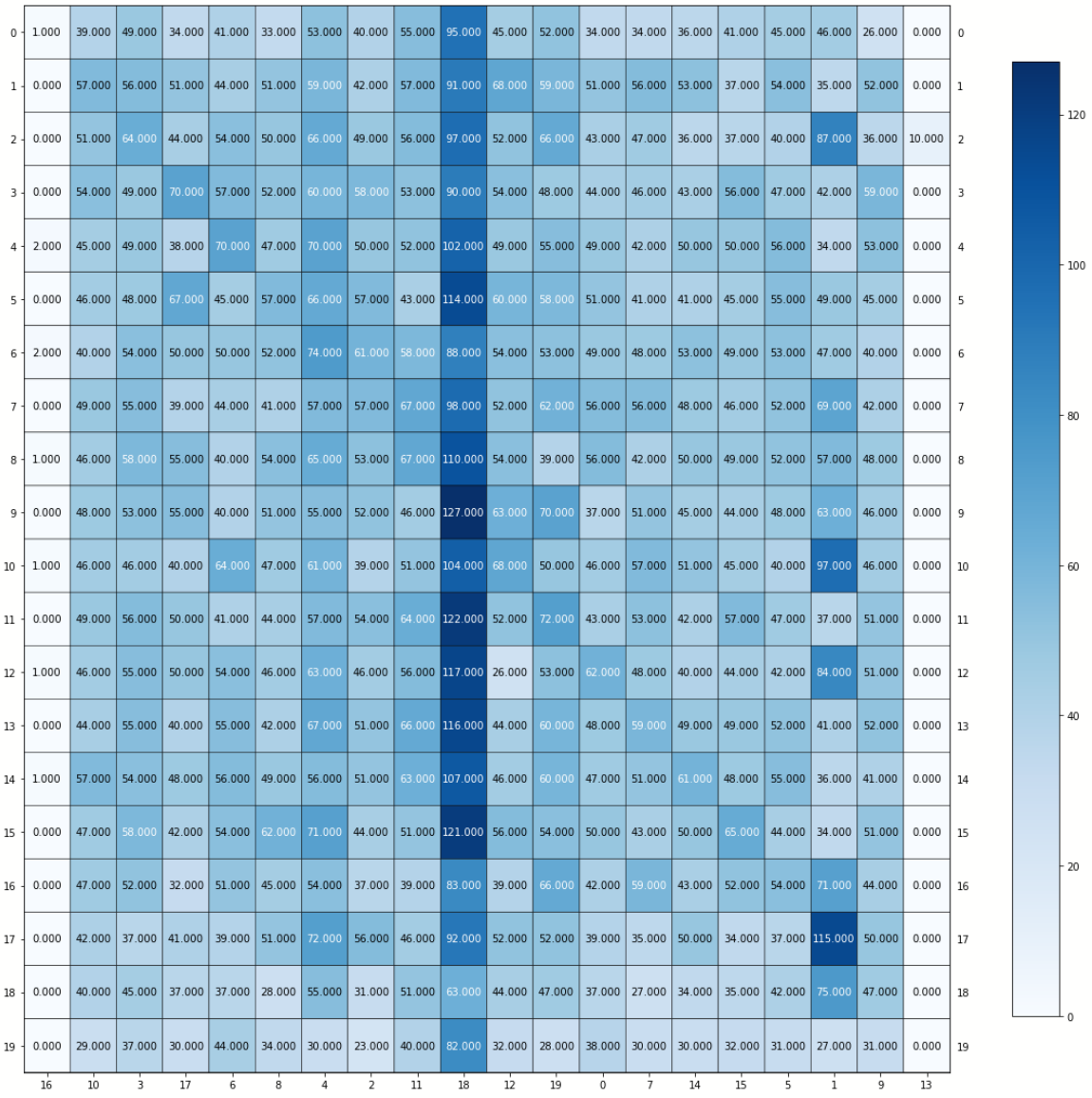
CM for # Comp = 5:



CM for # Comp = 20:



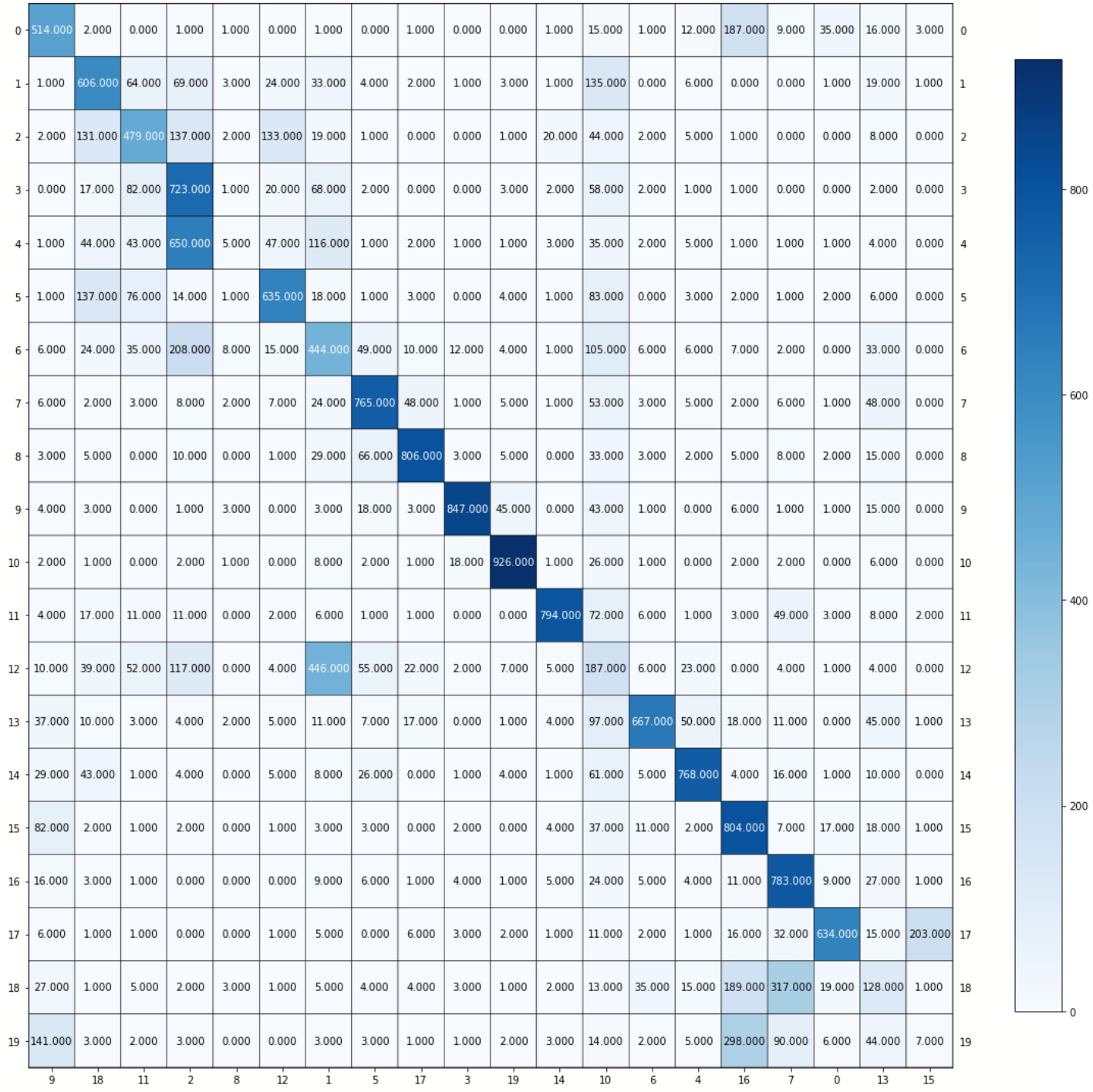
CM for # Comp = 200:



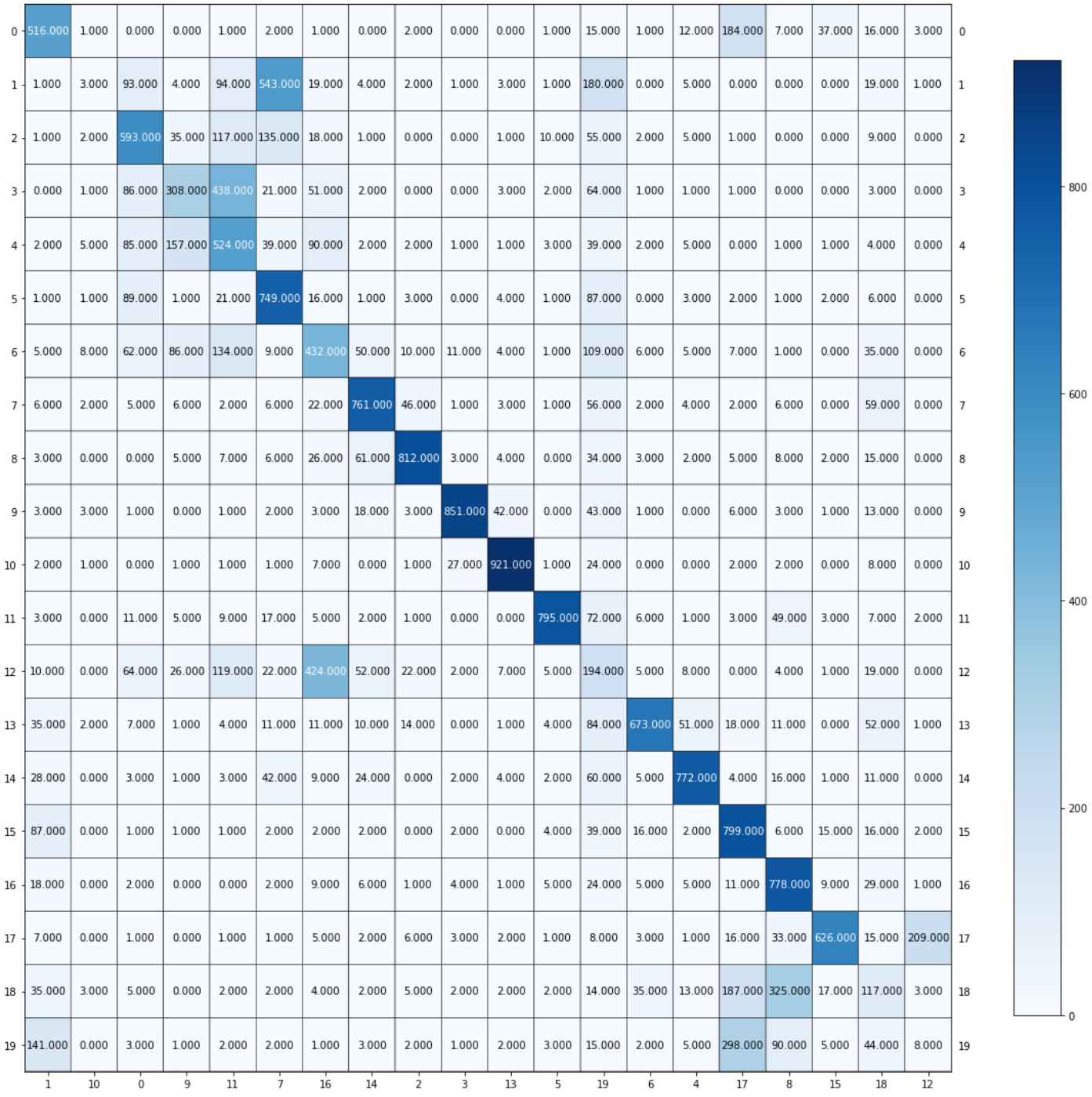
For Cosine Metric: 5 Scores for Umap with Cosine Metric for different number of components

	Num of Components	Homogeneity	Completeness	Vmeasure	ARI	AMIS
0	5.0	0.578146	0.599978	0.588860	0.451638	0.587496
1	20.0	0.573813	0.594536	0.583991	0.443968	0.582612
2	200.0	0.579453	0.600915	0.589989	0.454331	0.588630

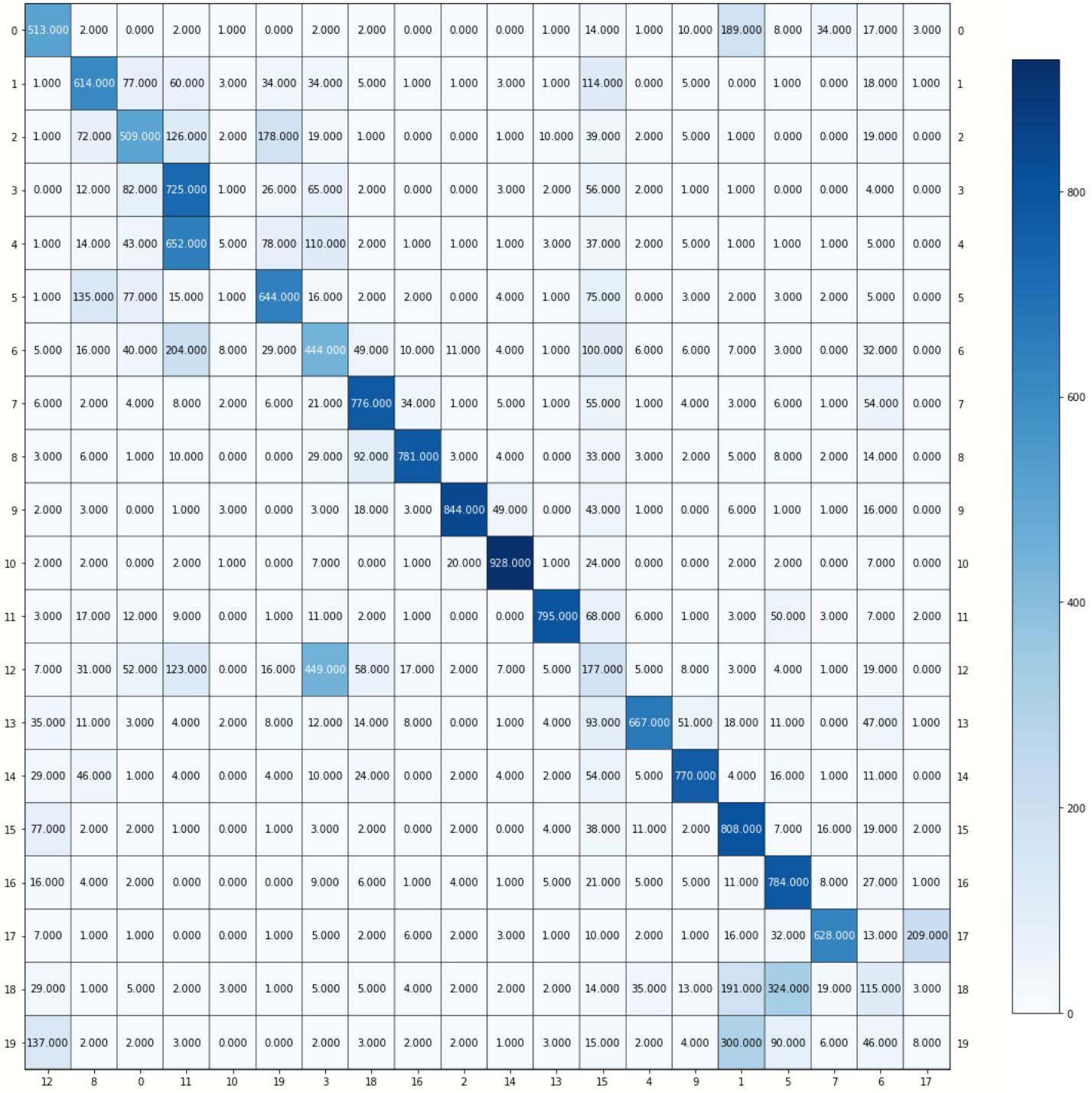
CM for # Comp = 5:


0 514.000 2.000 0.000 1.000 1.000 0.000 1.000 0.000 1.000 0.000 0.000 1.000 15.000 1.000 12.000 187.000 9.000 35.000 16.000 3.000 0
1 1.000 606.000 64.000 69.000 3.000 24.000 33.000 4.000 2.000 1.000 3.000 1.000 135.000 0.000 6.000 0.000 0.000 1.000 19.000 1.000 1
2 2.000 131.000 479.000 137.000 2.000 133.000 19.000 1.000 0.000 0.000 1.000 20.000 44.000 2.000 5.000 1.000 0.000 0.000 8.000 0.000 2
3 0.000 17.000 82.000 723.000 1.000 20.000 68.000 2.000 0.000 0.000 3.000 2.000 58.000 2.000 1.000 1.000 0.000 0.000 2.000 0.000 3
4 1.000 44.000 43.000 650.000 5.000 47.000 116.000 1.000 2.000 1.000 1.000 3.000 35.000 2.000 5.000 1.000 1.000 1.000 4.000 0.000 4
5 1.000 137.000 76.000 14.000 1.000 635.000 18.000 1.000 3.000 0.000 4.000 1.000 83.000 0.000 3.000 2.000 1.000 2.000 6.000 0.000 5
6 6.000 24.000 35.000 208.000 8.000 15.000 444.000 49.000 10.000 12.000 4.000 1.000 105.000 6.000 6.000 7.000 2.000 0.000 33.000 0.000 6
7 6.000 2.000 3.000 8.000 2.000 7.000 24.000 765.000 48.000 1.000 5.000 1.000 53.000 3.000 5.000 2.000 6.000 1.000 48.000 0.000 7
8 3.000 5.000 0.000 10.000 0.000 1.000 29.000 66.000 806.000 3.000 5.000 0.000 33.000 3.000 2.000 5.000 8.000 2.000 15.000 0.000 8
9 4.000 3.000 0.000 1.000 3.000 0.000 3.000 18.000 3.000 847.000 45.000 0.000 43.000 1.000 0.000 6.000 1.000 1.000 15.000 0.000 9
10 2.000 1.000 0.000 2.000 1.000 0.000 8.000 2.000 1.000 18.000 926.000 1.000 26.000 1.000 0.000 2.000 2.000 0.000 6.000 0.000 10
11 4.000 17.000 11.000 11.000 0.000 2.000 6.000 1.000 1.000 0.000 0.000 794.000 72.000 6.000 1.000 3.000 49.000 3.000 8.000 2.000 11
12 10.000 39.000 52.000 117.000 0.000 4.000 446.000 55.000 22.000 2.000 7.000 5.000 187.000 6.000 23.000 0.000 4.000 1.000 4.000 0.000 12
13 37.000 10.000 3.000 4.000 2.000 5.000 11.000 7.000 17.000 0.000 1.000 4.000 97.000 667.000 50.000 18.000 11.000 0.000 45.000 1.000 13
14 29.000 43.000 1.000 4.000 0.000 5.000 8.000 26.000 0.000 1.000 4.000 1.000 61.000 5.000 768.000 4.000 16.000 1.000 10.000 0.000 14
15 82.000 2.000 1.000 2.000 0.000 1.000 3.000 3.000 0.000 2.000 0.000 4.000 37.000 11.000 2.000 804.000 7.000 17.000 18.000 1.000 15
16 16.000 3.000 1.000 0.000 0.000 0.000 9.000 6.000 1.000 4.000 1.000 5.000 24.000 5.000 4.000 11.000 783.000 9.000 27.000 1.000 16
17 6.000 1.000 1.000 0.000 0.000 1.000 5.000 0.000 6.000 3.000 2.000 1.000 11.000 2.000 1.000 16.000 32.000 634.000 15.000 203.000 17
18 27.000 1.000 5.000 2.000 3.000 1.000 5.000 4.000 4.000 3.000 1.000 2.000 13.000 35.000 15.000 189.000 317.000 19.000 128.000 1.000 18
19 141.000 3.000 2.000 3.000 0.000 0.000 3.000 3.000 1.000 1.000 2.000 3.000 14.000 2.000 5.000 298.000 90.000 6.000 44.000 7.000 19
9 18 11 2 8 12 1 5 17 3 19 14 10 6 4 16 7 0 13 15
800
600
400
200
0

CM for # Comp = 20:



CM for # Comp = 200:



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

ANS:

The Cosine Metric with Number of component (k) = 200 performs best with formed cluster having proper segregation and allocation having following Clustering Measures:

* Homogeneity: 0.579
* Completeness: 0.60
* V-Measure: 0.589
* Adjusted Rand Index: 0.454
* Adjusted Mutual Information: 0.588

On analyzing the contingency matrices, we get to observe that Euclidean matrices doesn't do well with clustering, cluster formed are quite random and are indistinguishable.
The clusters for the cosine matrix appear to be well formed, with distinct segregations and allocations. A few points cannot be assigned to a cluster well, and other clusters have few assignments, demonstrating the importance of managing Kmeans outliers carefully.

QUESTION 13: So far, we have attempted K-Means clustering with 4 different representation learning techniques (sparse 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?

ANS:

Based on Adjusted Random Index and contingency metrices
Umap with cosine metrics and with number of components as 200 performed the best having Adjusted Random Index around 0.45 much higher than SVD or NMF which ranged between (0.1, 0.15) and (0.08, 0.1) respectively. It even has highest value for other 4 metrics too.

Agglomerative Clustering

QUESTION 14: Use 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.

ANS:

For Ward Linkage: 5 Scores for Agglomerative Clustering with Ward Linkage for different number of components and number of cluster = 20

	Num of Components	Homogeneity	Completeness	Vmeasure	ARI	AMIS
0	5.0	0.555949	0.580151	0.567792	0.410522	0.566354
1	20.0	0.555371	0.590477	0.572386	0.416165	0.570950
2	50.0	0.560211	0.605615	0.582029	0.413929	0.580613
3	100.0	0.541147	0.583006	0.561297	0.401055	0.559814
4	200.0	0.554767	0.578442	0.566357	0.407881	0.564926

For Single Linkage: 5 Scores for Agglomerative Clustering with Single Linkage for different number of components and number of cluster = 20

	Num of Components	Homogeneity	Completeness	Vmeasure	ARI	AMIS
0	5.0	0.097752	0.724632	0.172266	0.019885	0.168658
1	20.0	0.019735	0.356318	0.037398	0.000585	0.032366
2	50.0	0.007081	0.284159	0.013817	0.000011	0.009153
3	100.0	0.017823	0.392942	0.034100	0.000471	0.029320
4	200.0	0.008757	0.291097	0.017003	-0.000028	0.012261

Ward linkage is performing much better as compared to single linkage criteria with having best Adjusted Random Index around 0.413 with 50 number of components.

HDBSCAN

QUESTION 15: Apply HDBSCAN on UMAP-transformed 20-category data.
Use min_cluster_size=100. Vary the min cluster size among 20, 100, 200 and report your findings in terms of the five clustering evaluation metrics - you will plot the best contingency matrix in the next question. Feel free to try modifying other parameters in HDBSCAN to get better performance.

ANS:
The Hyper parameter experimented for HDBSCAN with Umap are:

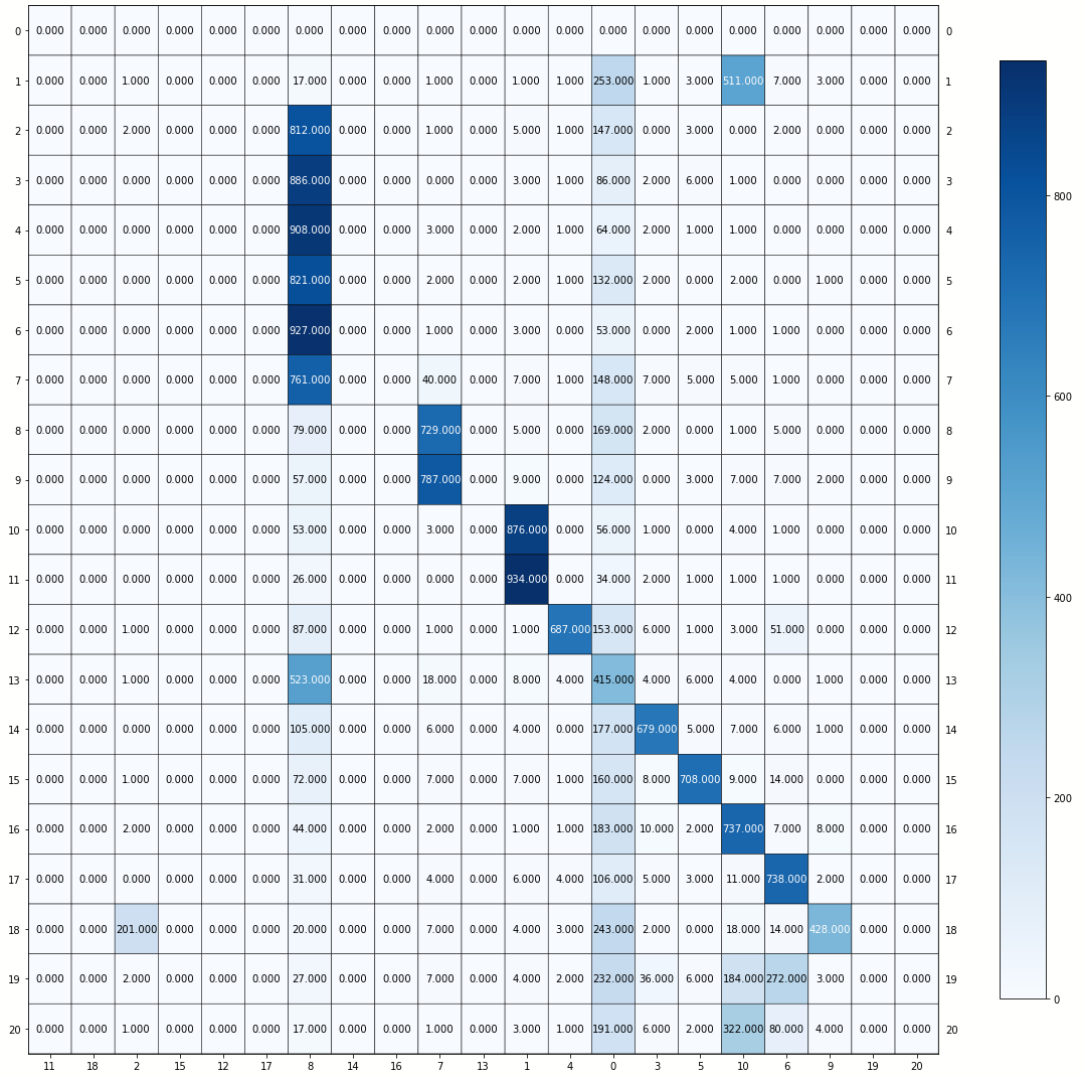
* Min cluster sizes = [20,100, 200]
* Min samples = [20,100, 200]
* Epsilon values = [0.5, 0.8, 5]
* Number of components = [5,20,200]

The best Hyper parameter and corresponding 5 scores are following:
* Min cluster sizes = 20
* Min samples = 100
* Epsilon values = 0.5
* Number of components = 5

* Homogeneity: 0.428
* Completeness: 0.631
* V-Measure: 0.510
* Adjusted Rand Index: 0.222
* Adjusted Mutual Information: 0.5096

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.

ANS:
Contingency Matrix for best model for Q15 is plotted below
The data points that are regarded as outliers or noisy samples by the clustering algorithm are indicated by the clustering labels "-1". They are not a part of any cluster.
Total 21 clusters are formed. Although certain data points are ignored by the algorithms because they are deemed outliers.



QUESTION 17: Based on your experiments, which dimensionality reduction technique and clustering methods worked best together for 20-class text data and why? Follow the table below. If UMAP takes too long to converge, consider running it once and saving the intermediate results in a pickle file.

ANS:

For KMeans Clustering: Best hyperparameter found and corresponding 5 scores are

* Best Number of Clusters: 20
* Best Reduction Method: UMAP with 5 number of components

* Homogeneity: 0.5757
* Completeness: 0.6034

* V-Measure: 0.589
* Adjusted Rand Index: 0.4460
* Adjusted Mutual Information: 0.587

For Agglomerative Clustering: Best hyperparameter found and corresponding 5 scores are

* Best Number of Clusters: 20
* Best Reduction Method: UMAP with 5 number of components

* Homogeneity: 0.559
* Completeness: 0.594
* V-Measure: 0.5761
* Adjusted Rand Index: 0.4270
* Adjusted Mutual Information: 0.5746

For HDBSCAN: Best hyperparameter found and corresponding 5 scores are

* Best min clusters size: 200
* Best Reduction Method: UMAP with 5 number of components

* Homogeneity: 0.4327
* Completeness: 0.6385
* V-Measure: 0.5155
* Adjusted Rand Index: 0.2247
* Adjusted Mutual Information: 0.5145

From above results, the best combinations in decreasing order of average clustering measures are as follows:

1). KMeans + UMAP
2). Agglomerative clustering + UMAP
3). HDBSCAN + UMAP

QUESTION 18: Extra credit: If you can find creative ways to further enhance the clustering performance, report your method and the results you obtain.
ANS: Two approaches were taken:

In first, we used a combination of SVD and UMAP. The 5 dimensionally reduced features were taken from both and multiplied together element wise. We also tried

creating new feature by concatenating both. The idea was that feature space created might map data point far apart creating better clusters.

This approach didn’t give better results.

Score on multiplying the SVD and UMAP feature map
* Homogeneity: 0.2834
* Completeness: 0.3147
* V-Measure: 0.2982
* Adjusted Rand Index: 0.101
* Adjusted Mutual Information: 0.295

Score on concatenating the SVD and UMAP feature map
* Homogeneity: 0.352
* Completeness: 0.442
* V-Measure: 0.392
* Adjusted Rand Index: 0.1066
* Adjusted Mutual Information: 0.3902

In the second approach we made changes in the data itself.

A) Data without removing header and footer

This was tested on best two combinations from previous results i.e., UMAP with K-means and SVD with K-Means were tried.

* Number of Components for SVD and UMAO: 50
* Number of clusters: 20
* min_df: 3 with TF-IDF representation.

1. UMAP + KMeans:
* Homogeneity: 0.5187
* Completeness: 0.54338
* V-Measure: 0.53077

* Adjusted Rand Index: 0.53077
* Adjusted Mutual Information: 0.52921

2. SVD + Kmeans:

* Homogeneity: 0.5205
* Completeness: 0.5431
* V-Measure: 0.531
* Adjusted Rand Index: 0.417
* Adjusted Mutual Information: 0.5300

B. Using min_df = 5 with TF-IDF representation.

* Homogeneity: 0.53253
* Completeness: 0.5512
* V-Measure: 0.54173
* Adjusted Rand Index: 0.43478
* Adjusted Mutual Information: 0.5402

b). SVD + Kmeans:

* Homogeneity: 0.3531
* Completeness: 0.44197* V-Measure: 0.54173
* Adjusted Rand Index: 0.3926
* Adjusted Mutual Information: 00.390

The Adjusted Random index for this approach is similar to the results without including the header and footer, but the results of all metrics were improved slightly on average.

Part 2 – Deep Learning and Clustering of Image Data

QUESTION 19: In a brief paragraph discuss: If the VGG network is trained on a dataset with perhaps totally different classes as targets, why would one expect the features derived from such a network to have discriminative power for a custom dataset?

Ans: The VGG network would hopefully be able to recognize features that are relatively universal from the other dataset. Universal features might be things like colors, shapes, or common objects. These features would be applicable to a different dataset even if it contains different classes of images.

QUESTION 20: In a brief paragraph explain how the helper code base is performing feature extraction.

Ans: The helper code crops and normalizes the images when loading them to make for a more uniform dataset of original images. It then transforms the data according to a feature extraction module on every image in the dataset and saves the result to a feature vector. This feature extraction module copies the model from the pretrained model by duplicating the neural network feature layers, pooling layer, and fully connected layer. The original data does not modify the model as it is the model run in evaluation mode when the data is processed.

QUESTION 21: How many pixels are there in the original images? How many features does the VGG network extract per image; i.e. what is the dimension of each feature vector for an image sample?

Ans: The original images are loaded in as 244x244 pixel objects (where each pixel has 3 values corresponding to R, G, and B). This would represent 59,539 different pixels each with three different RGB values. Each image's feature vector has 4096 values after processing.

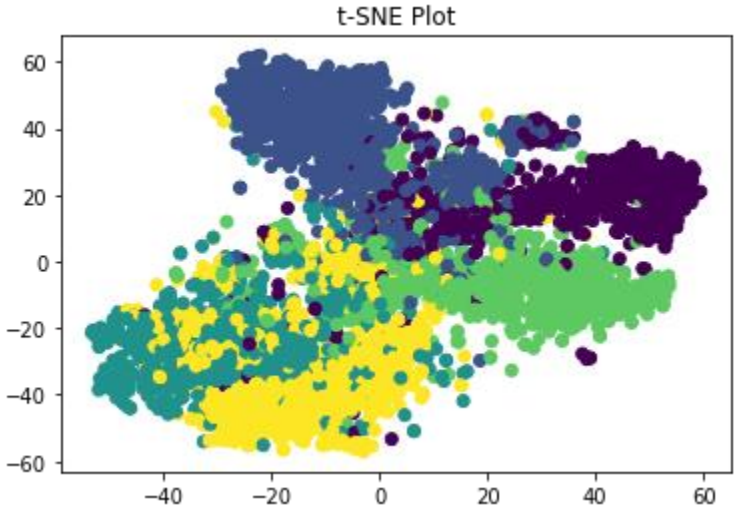
QUESTION 22: Are the extracted features dense or sparse? (Compare with sparse TF-IDF features in text.)

Ans: The extracted features are dense, especially in comparison to TF-IDF which is sparse. This makes sense that it is less sparse than TF-IDF as that simply

represents whether a word appears in a document, while whether an image has features that would be more nuanced.

QUESTION 23: In order to inspect the high-dimensional features, t-SNE is a popular off-the-shelf choice for visualizing Vision features. Map the features you have extracted onto 2 dimensions with t-SNE. Then plot the mapped feature vectors along x and y axes. Color-code the data points with ground-truth labels. Describe your observation.

Ans:



For the most part, the t-SNE analysis is able to contain the different labels to different regions. This shows that objects of the same label have statistical similarities to each other that can be exploited. However, there are also large regions that have multiple different labels blended together. This could lead to issues when trying to separate the data through clustering mechanisms.

QUESTION 24: Report the best result (in terms of rand score) within the table below. For HDBSCAN, introduce a conservative parameter grid over min cluster size and min samples.

Ans:

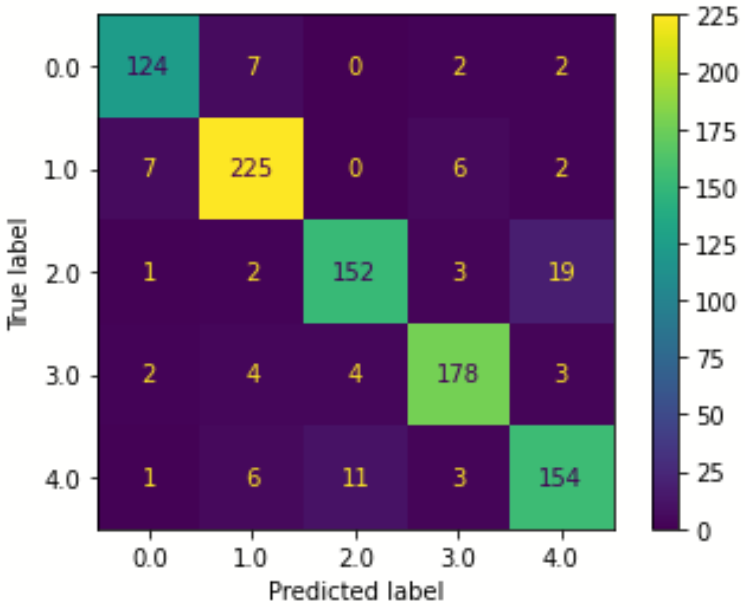
The best overall combination was UMAP with 50 features combined with K-Means with 5 clusters. This had an adjusted rand score of 0.42.

For HDBSCAN, a parameter grid of [5, 50, 500] was used for both min cluster size and min samples. The best result that used HDBSCAN had a minimum cluster size of 500, minimum samples of 5, and UMAP feature reduction to 50 features. This resulted in an adjusted rand score of 0.28.

QUESTION 25: Report the test accuracy of the MLP classifier on the original VGG features. Report the same when using the reduced-dimension features (you have freedom in choosing the dimensionality reduction algorithm and its parameters). Does the performance of the model suffer with the reduced-dimension representations? Is it significant? Does the success in classification make sense in the context of the clustering results obtained for the same features in Question 24.

Ans: The test accuracy of the MLP classifier on the original VGG features is 0.91, and the adjusted rand score is 0.79. Using UMAP to reduce the dimensions to 100, the accuracy becomes 0.83, and the adjusted rand score becomes 0.63. The performance of the model does significantly suffer with the reduced-dimension representation, but it is still a lot better than a random guess. It is a bit surprising that the classification works significantly better than the clustering, but at the same time it makes sense since the pretrained model likely did not extract the most optimal features.

CM for Original VGG Features



CM for UMAP Reduced Original VGG Features