Large-Scale Data Mining: Models and Algorithms ECE 219 MSOL Spring 2022

Prof. Vwani Roychowdhury

UCLA, Department of ECE

Team Members:
Rohan Mehta, UID: 205871841

Project 2: Data Representations and Clustering

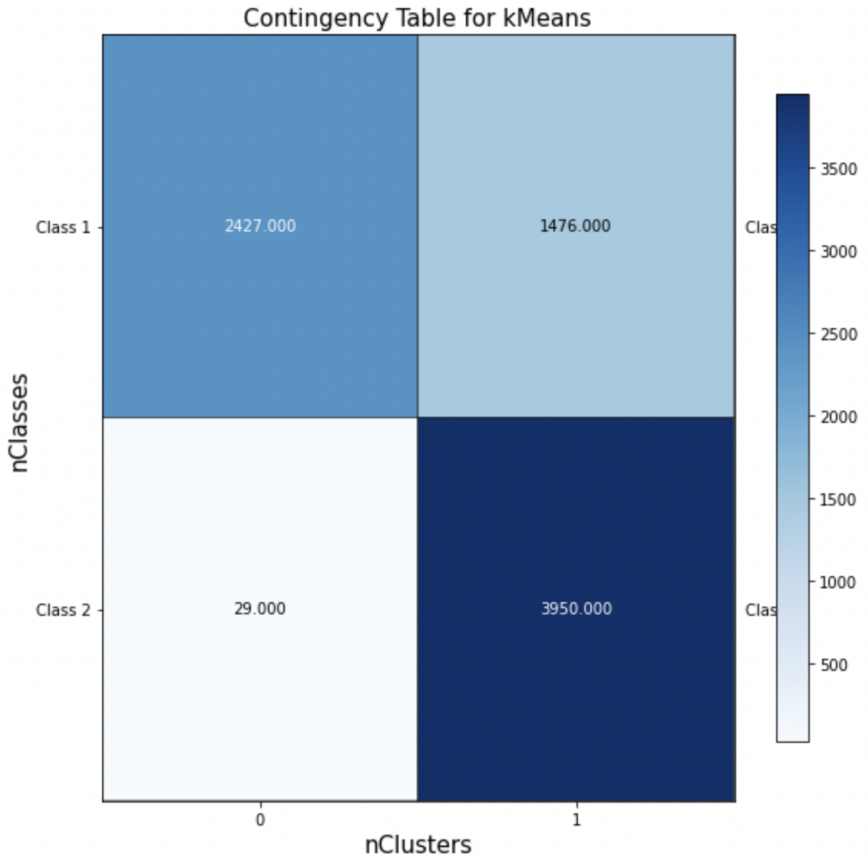
1. Generate sparse TF-IDF representations:

Question 1:

Table 1: TF-IDF Representation of Data	
Dimensions of All Data	(7882, 15164)

2. Clustering

Question 2:



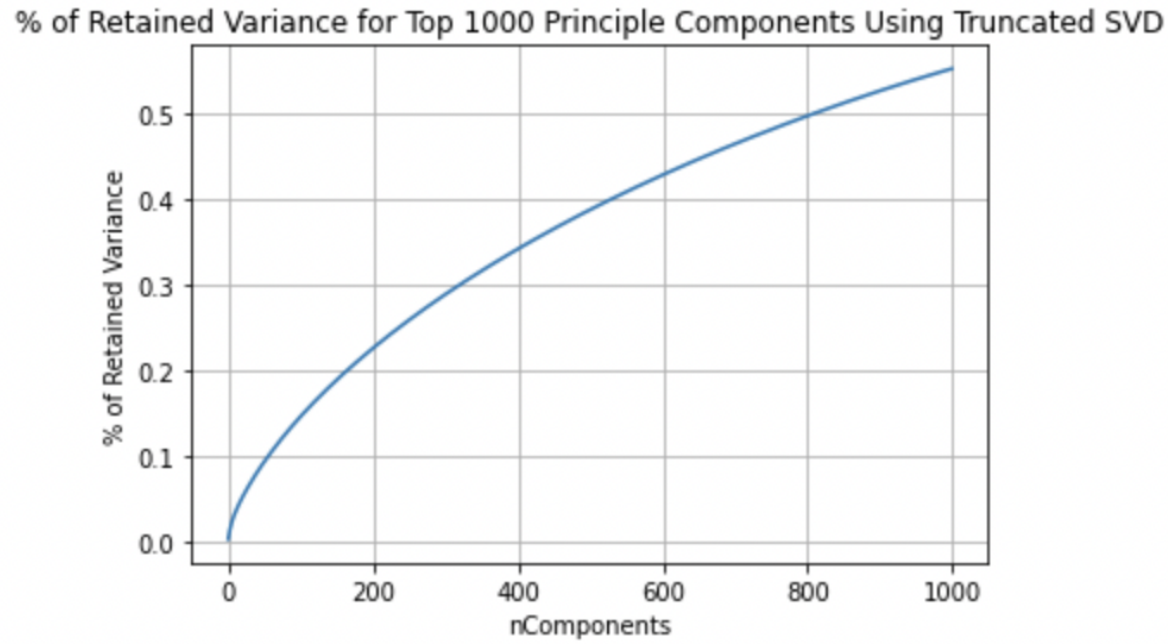
Although for this particular problem the number of classes and clusters chosen are the same, these parameters can fluctuate resulting in a non-square contingency matrix.

Question 3:

Table 2: Common Clustering Evaluation Metrics	
Homogeneity	0.436
Completeness	0.390
V-measure	0.411
Adjusted Rand Index	0.382
Adjusted mutual information score	0.411

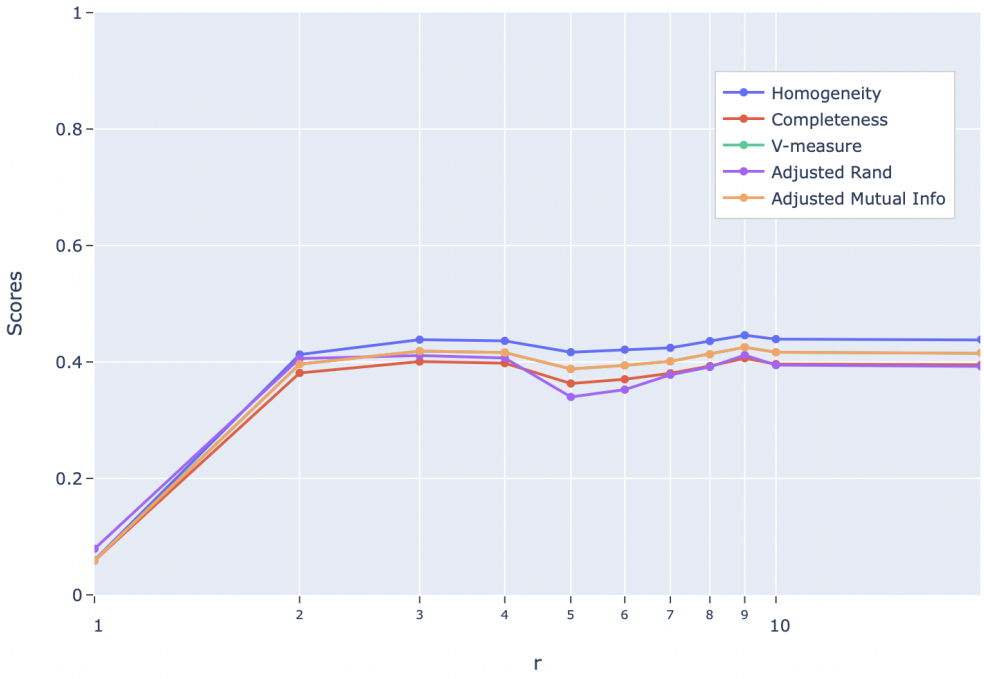
1. Generate dense representations for better K-Means Clustering

Question 4:



Question 5:

SVD nComponents vs Classification Scores



NMF nComponents vs Classification Scores

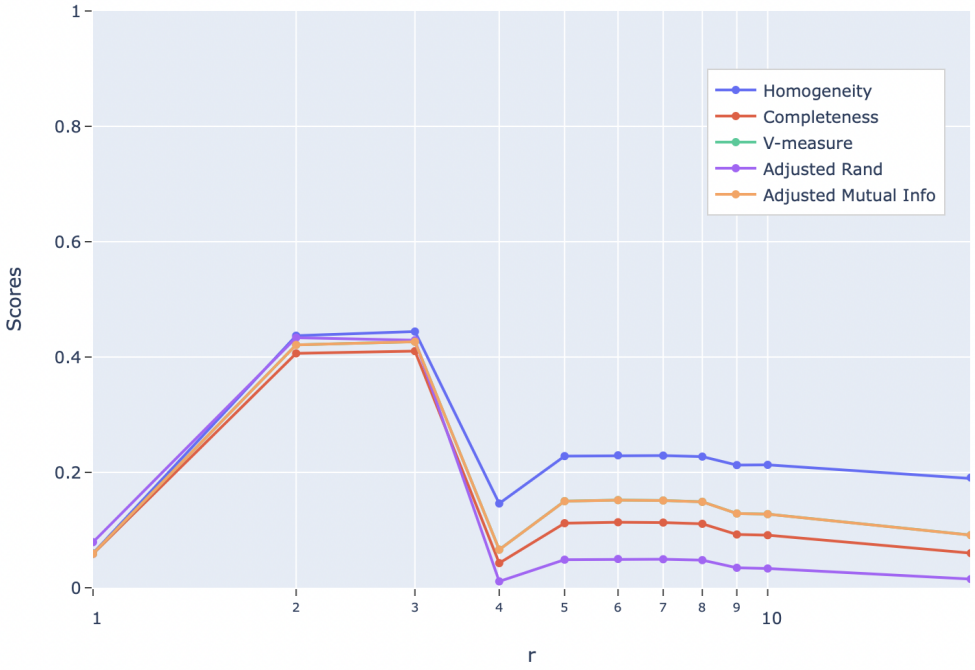


Table 4: Chosen nComponents After Performing Sweep	
SVD	nComponents = 3
NMF	nComponents = 3

Question 6:

In general, both NMF and the TruncatedSVD techniques are unsupervised dimensionality reduction techniques. As a result, there could be unwanted data loss which is essential to describing the dataset.

The TruncatedSVD finds linear correlations between data which is undesirable if more complex relationships are better suited to describe the dataset. As the number of components increased in the figure, the performance did not improve for the various clustering scores showing a cap in performance with linear techniques.

NMF on the other hand, fails to consider the geometric structure of the data being represented. The output of the NMF method is usually two matrices with smaller dimensions than the original data; this is a compressed representation of the original matrix but there is information loss on data structure.

Question 7:

Table 5: Average Performance Across nComponents SVD/NMF (removed nComponent = 1)					
	Homogeneity	Completeness	V-measure	Adjusted Rand Index	Adjusted mutual information score
SVD	.433	.390	.410	.388	.410
NMF	.232	.126	.155	.090	.155
Question 3	.436	.390	.411	.382	.411

The purity score for SVD performs slightly worse on average than when no dimensionality reduction is used. In general, the performance is not significantly affected by dimensionality reduction with SVD, however with NMF the performance worsens (especially if convergence was not optimal). The best method between SVD and NMF is clearly SVD.

2. Visualize the clusters

Table 6: SVD with Optimal Component Choice (r = 3)

Table 7: NMF with Optimal Component Choice (r = 3)

Question 9:

In the SVD visualization with the optimal nComponent parameter, there appears to be a clear boundary between both clusters. The points are distributed with a high density at the origin (0,0) but spread outwards equally about the x-axis in both the positive and negative y-direction. The class (ground-truth) colored plot and cluster colored plot are inverses but we do not expect a one-to-one correspondence between class number and cluster number. Overall, the distribution of data points is not ideal for clustering because there is not a clear separation boundary between both classes; it is densely populated where the transition occurs between clusters.

Similar analysis can be made about the NMF visualizations. The class and cluster labels are inverses, there is a separation boundary

geometrically between both classes however it is densely populated where the transition occurs.

3. Clustering of Entire 20 Classes

Question 10:

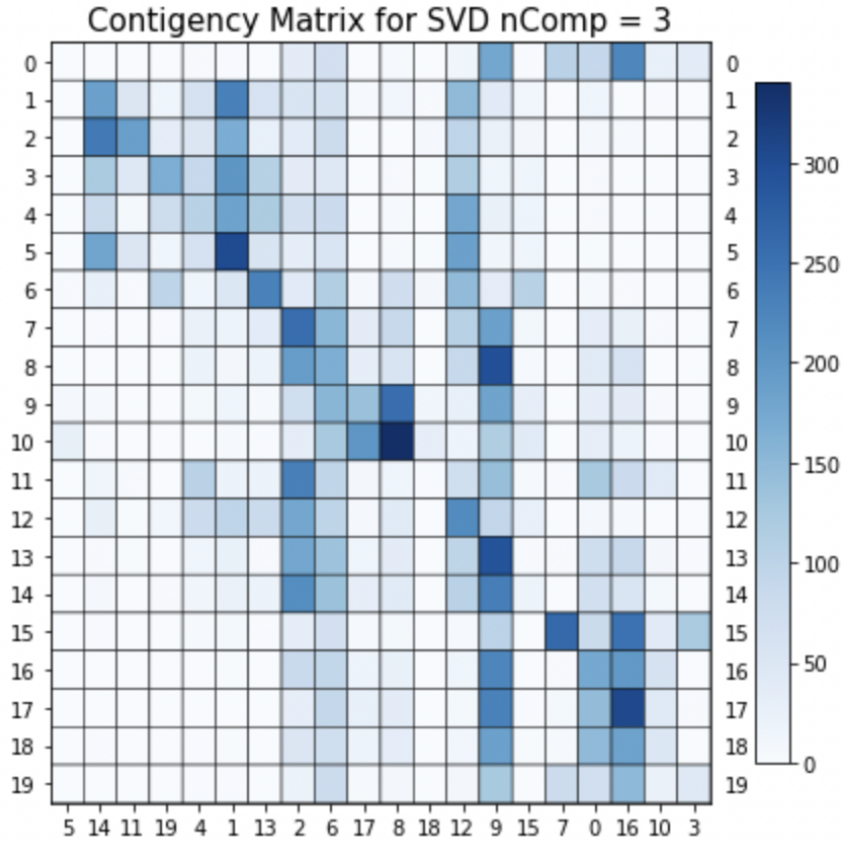
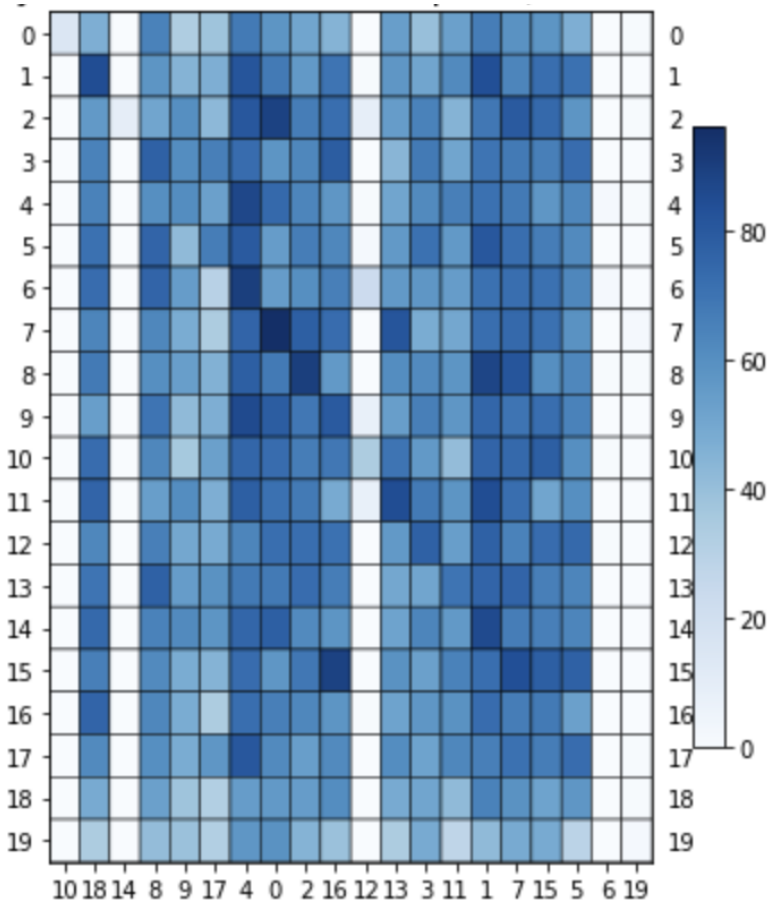


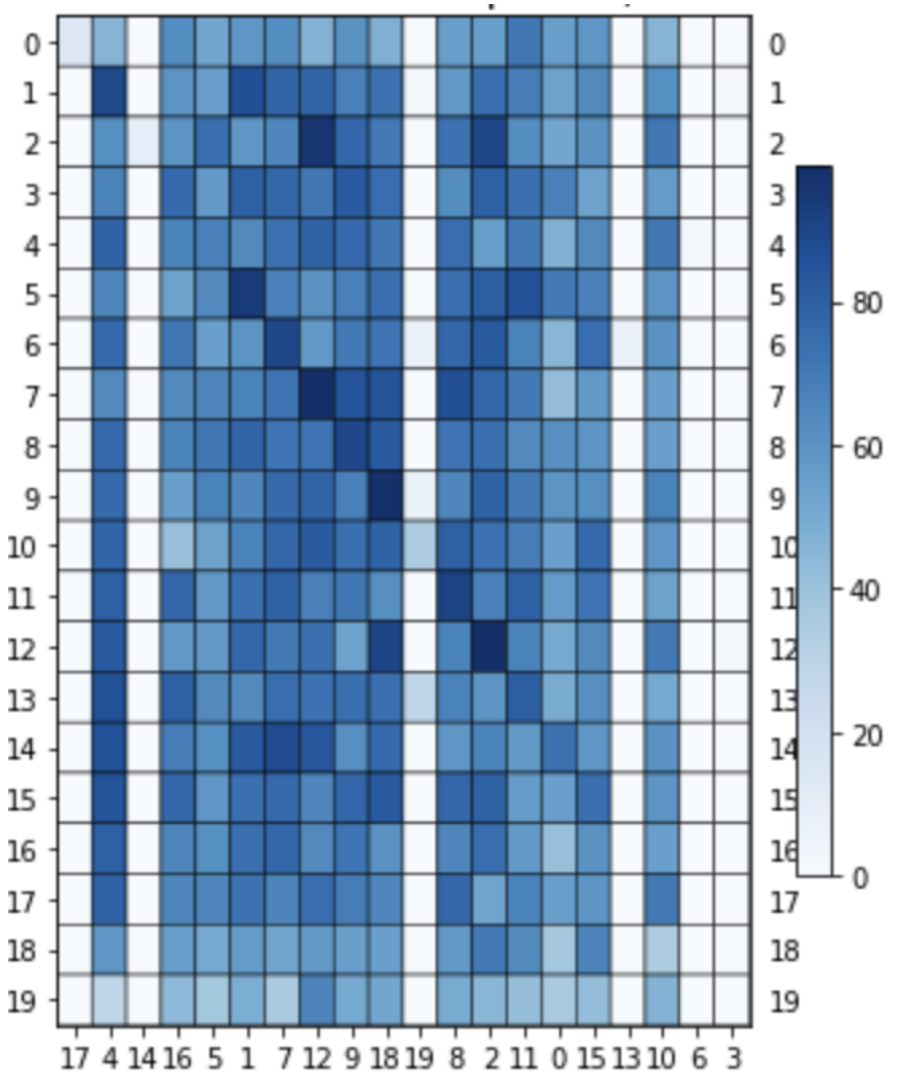
Table 6: Clustering Metrics	
Homogeneity	0.215
Completeness	0.195
V-measure	0.204
Adjusted Rand Index	0.062
Adjusted mutual information score	0.202

4. UMAP

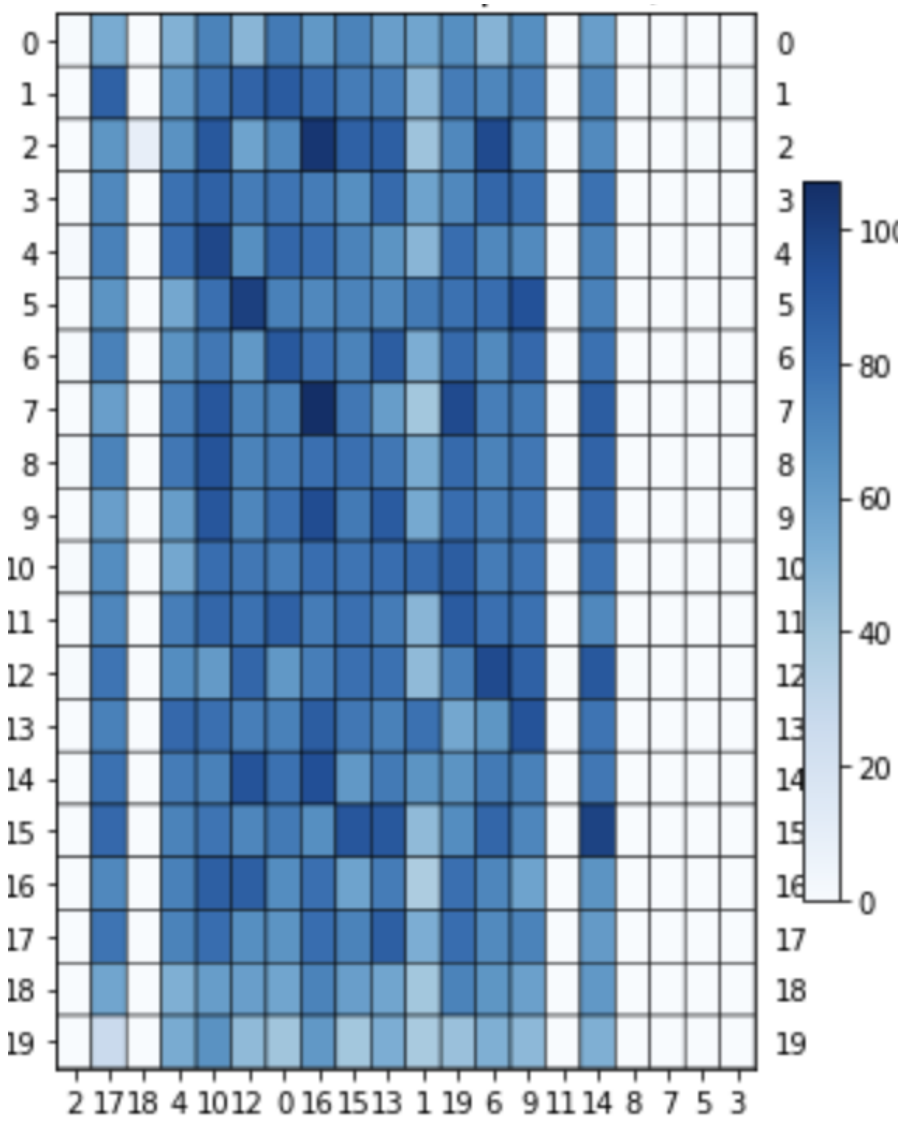
Table 7: UMAP Clustering Contingency Matrices



metric= “Euclidean”, nComp = 5



metric= “Euclidean”, nComp = 20

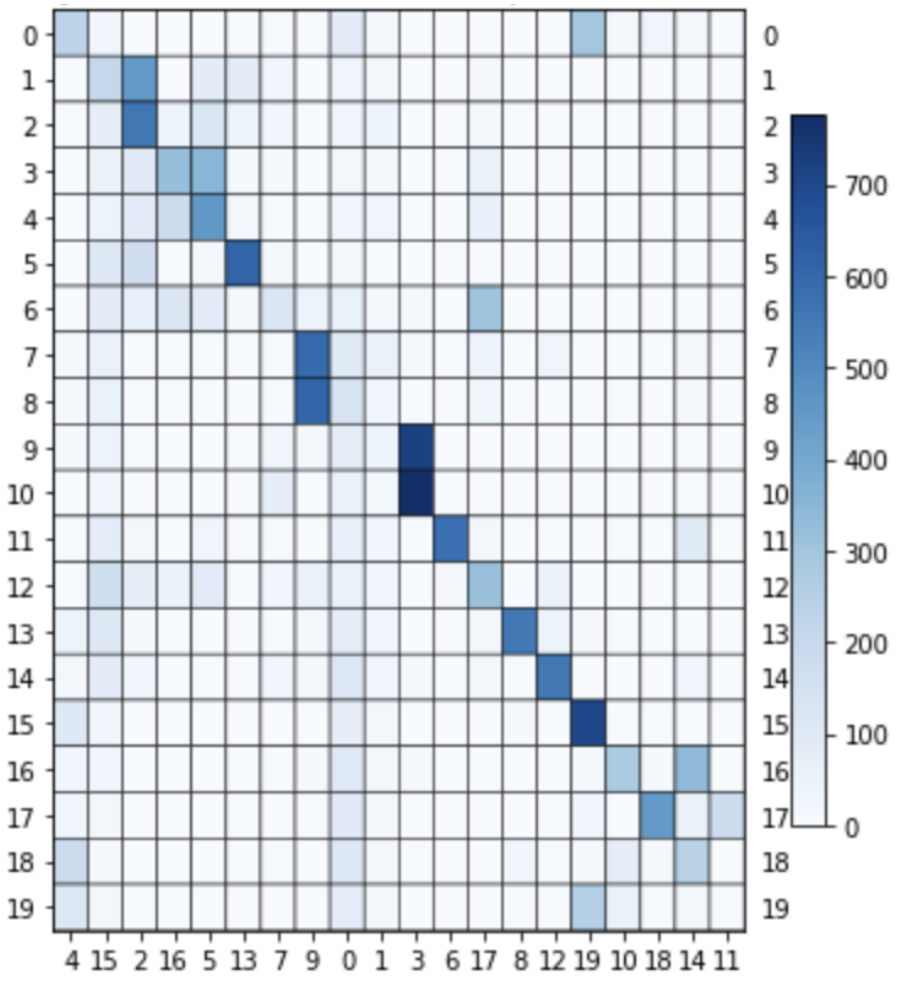


metric= “Euclidean”, nComp = 200

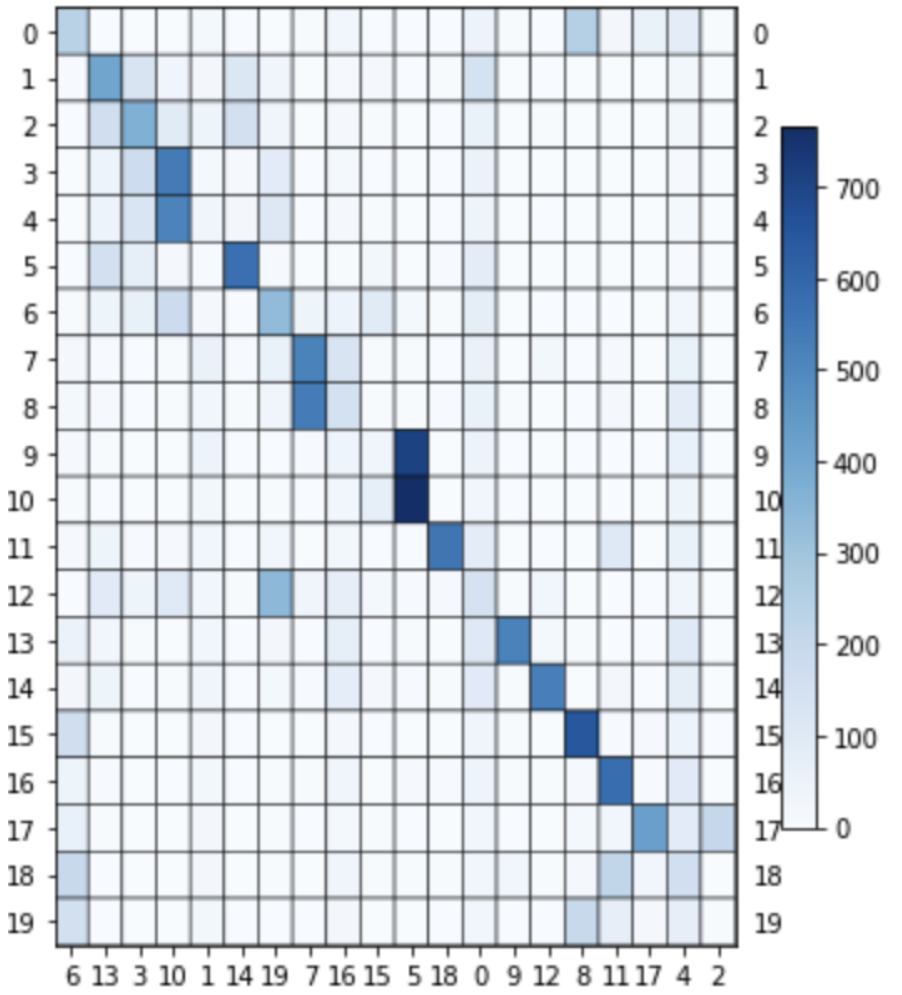
Table 8: UMAP, Distance Metric = “Euclidean”, nComp = 5, 20, 200

	Homogeneity	Completeness	V-measure	ARI	AMI
nComp = 5	.007	.006	.007	0.00	.003
nComp = 20	.008	.007	.007	0.00	.004
nComp = 200	.004	.003	.004	0.00	.001

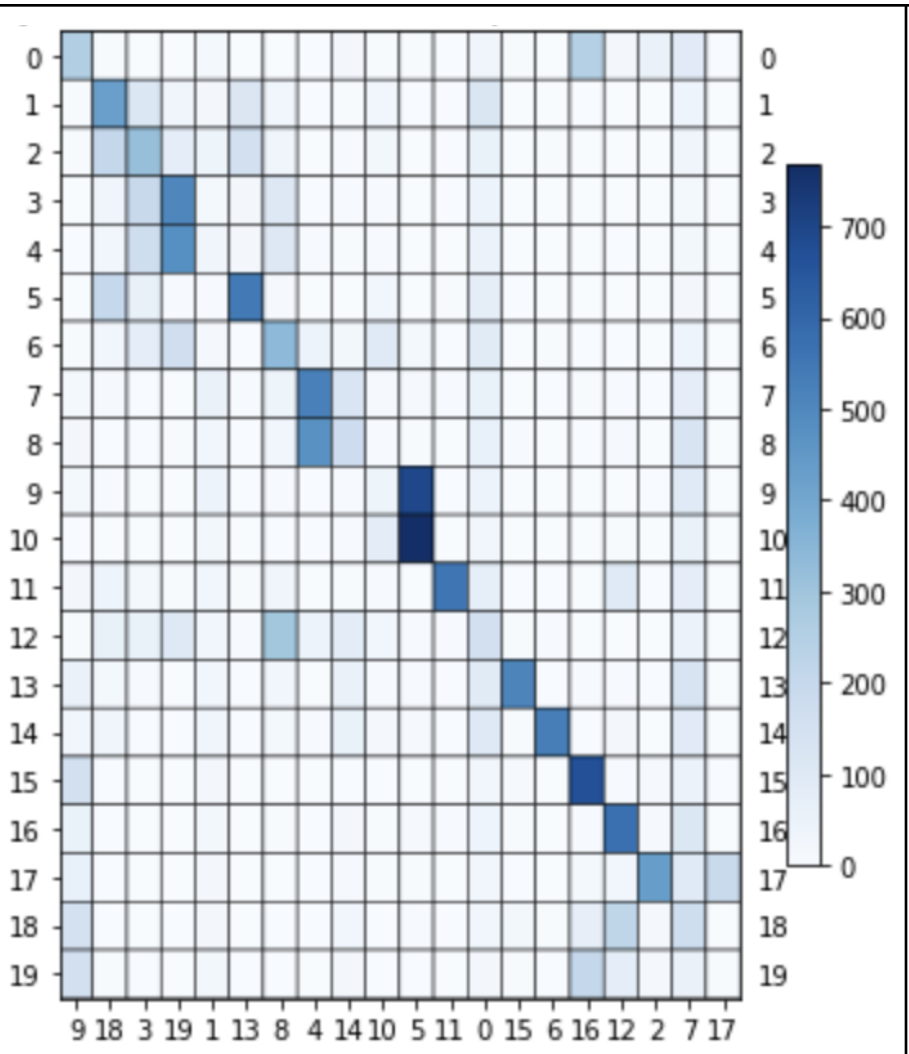
Table 9: UMAP Clustering Contingency Matrices



metric= “Cosine”, nComp = 5



metric= “Cosine”, nComp = 20



metric= “Cosine”, nComp = 200

Table 10: UMAP, Distance Metric = “Cosine”, nComp = 5, 20, 200

	Homogeneity	Completeness	V-measure	ARI	AMI
nComp = 5	.430	.416	.423	.272	.421
nComp = 20	.419	.408	.414	.264	.412
nComp = 200	.418	.407	.413	.256	.411

Question 12:

The contingency matrix is a tool to determine the effectiveness of a clustering algorithm. In this case, there are 20 classes and the KMeans classifier is initialized to have 20 clusters. Hence, an ideal clustering result should have the largest “presence” or “coloration” along the diagonal; this ideal result indicates that all examples of the same ground truth class are clustered together. We would expect an ideal feature vector to be separable by ground truth class which would lead to better clustering results.

In the case of many of the contingency matrices shown after performing UMAP, the performance is far from ideal. Especially with euclidean distance as the metric, the dimensionality reduction did not do a good job of separating by class as is shown by the large presence along many columns indicating that several ground truth classes were dispersed among many clusters. The better metric is the cosine distance for UMAP dimensionality reduction; this calculation is prone to producing NaN values however so an additional step was performed of imputing NaN values with the mean of the column prior to KMeans clustering.

Question 13:

Among the four different representation techniques performed in the prior sections, the one which intuitively and practically performed the best was UMAP with the cosine similarity as the “distance metric”.

Table 8 and Table 10 provide the clustering metrics of using euclidean distance and cosine distance respectively; the different magnitudes of TF IDF vectors seem to bias the euclidean distance whereas cosine distance can determine similarity despite differences in magnitude. Table 6, on the other hand, provides the clustering metrics of the best performing method from the previous 2 class experiment. The performance is much worse when applied to a larger set of classes. When comparing NMF, SVD, and the sparse representation, there appears to be little difference in performance between SVD and the sparse representation as seen in Table 5. Based on these clustering metrics gathered in the previous sections, it

can be concluded that UMAP with cosine similarity is the best approach for the KMeans clustering task.

a. Agglomerative Clustering

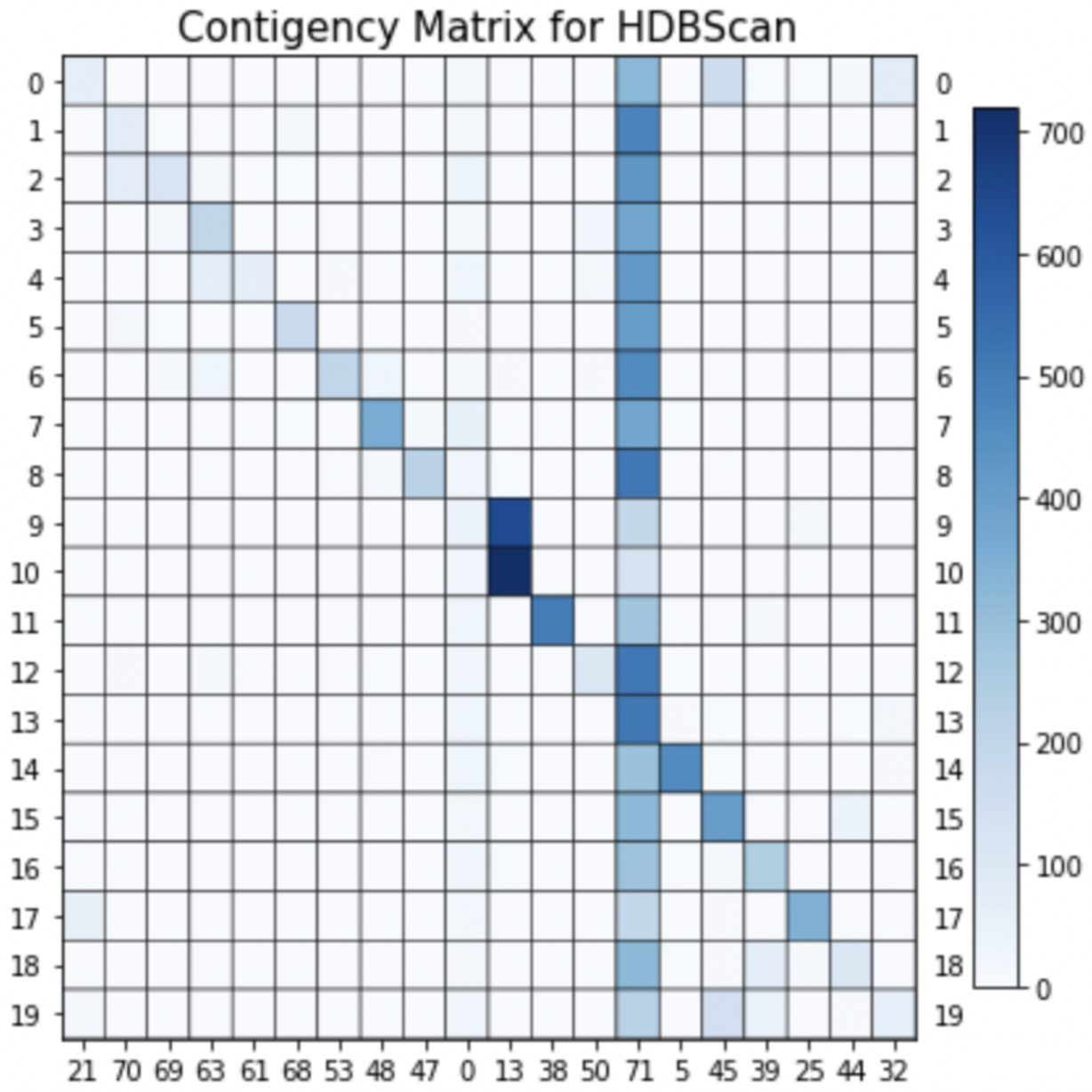
Question 14:

Table 11: Agglomerative Clustering					
Linkage	Homogeneity	Completeness	V-measure	ARI	AMI
Ward	.430	.414	.422	.280	.420
Single	.120	.008	.015	0.00	.010

Question 15:

Table 12: HDBSCAN, alpha=1.0, min_samples=1, leaf_size = 40, metric='l1', min_cluster_size = 42				
Homogeneity	Completeness	V-measure	ARI	AMI
.350	.355	.352	.070	.344

Question 16:



For this model there are 72 defined clusters, but not all are displayed after performing linear sum assignment. From the contingency matrix it can be seen that cluster 72 had the most spread across the different classes for data included. Additionally, not shown above is the cluster label “-1” which corresponds to all points which were considered noise and not considered as part of the cluster after calling the fit function.

Question 17:

For the 20-class text data, UMAP dimensionality reduction when paired with either K-Means or Agglomerative (“ward” linkage) clustering were optimal choices. The optimal trained UMAP embedding had n_components = 5 and the optimal K-Means clustering had k=20 (Agglomerative was n_clusters = 20 only). Tables 10 and 11 display the clustering metrics calculated between each method. When measured by purity score, both Agglomerative and K-Means clustering have a homogeneity score of .430.

As mentioned previously, the UMAP dimensionality reduction technique with the cosine metric is more immune to changes in the magnitude of the TF IDF vectors which vary greatly as the number of classes trained on increases. This could be a large contributing factor as to why using UMAP resulted in equal performance across multiple clustering methods.

Question 19:

The ability to take a trained model on a dataset and apply the same model to another task with different target labels is a technique referred to as transfer learning in machine learning literature. This technique helps alleviate the resource demand which comes with training models for every new task one may encounter. VGG will be trained on a task with labeled data and apply its learned trends to the new set of labels; this technique works well when the new task is in a similar environment, or is somehow related to the original task. It would be ideal if VGG was trained on a general task, and the new task was more specific.

Question 20:

The input layer to the final max pooling layer in the VGG architecture is regarded as the feature extraction portion of the network, however in the helpers class the feature extraction is done until the first fully connected

layer. In the helpers notebook, the feature extractor class loads a pretrained VGG model and performs average pooling to meet the total kernel size of the FC layer, then flattens in order to be the one dimensional required length. This process is done for a single image so in order to extract all features for the entire dataset, all features are stacked vertically into an array.

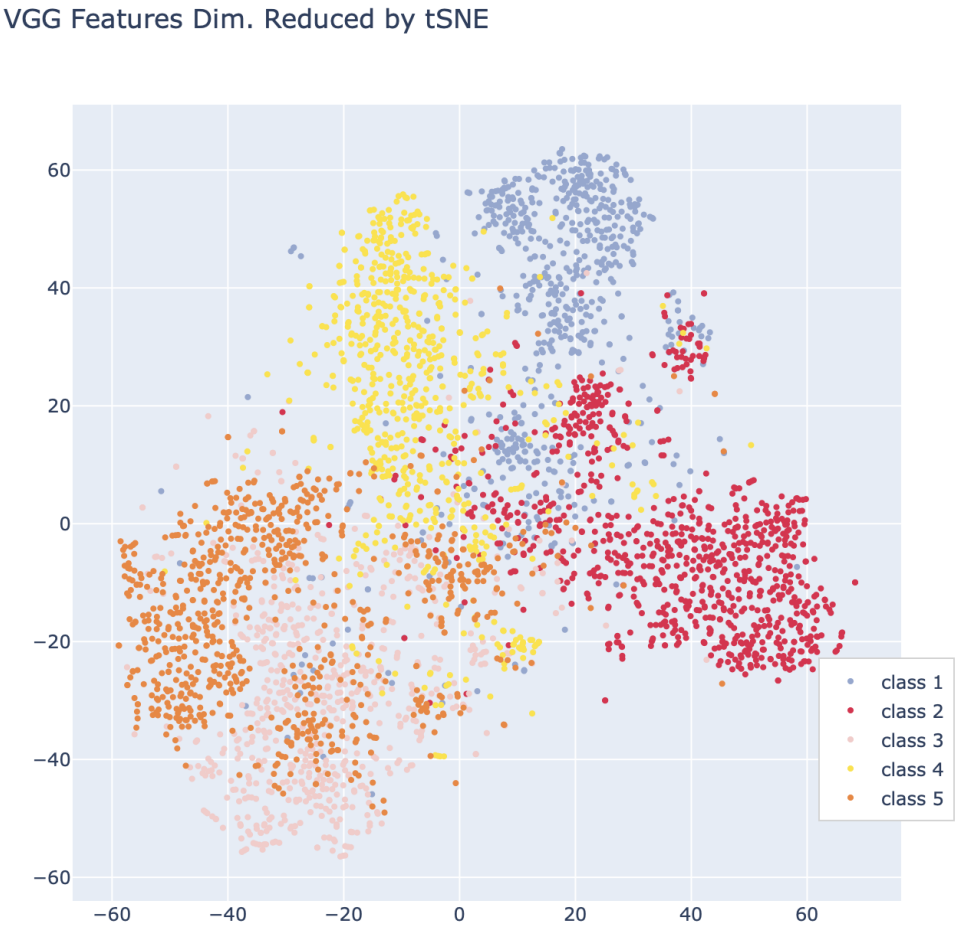
Question 21:

The sizes of each of the images in the flower dataset are variable, but must be resized to 224x224x3 in order to be compatible with the VGG network architecture. Thus, the input pixel count into the network is 150528. Per image, the VGG network extracts features of size (1,4096) which when vertically stacked for each image in the dataset results in a (3670,4096) matrix.

Question 22:

Compared with the TF IDF vectors computed in previous sections, these features are dense because the values in the feature array are predominantly non-zero.

Question 23:



There are five classes present in the flower data and the clusters colored by class depict five noisy clusters which appear to be separable. Classes 3 and 5 appear to be in very similar regions of the feature map; if this was fed into a classifier, then I would expect more misclassifications of those two labels.

Question 24:

Table 13: Best Clustering Result Measured by Adjusted Rand. Index		
	Adjusted Rand. Index	Dimensionality Reduction Technique

Agg. Clustering	0.4599	UMAP (n_comp = 50)
KMeans	0.4681	UMAP (n_comp=50)
HDBSCAN (min_cluster = 5, min_samples=1)	0.1912	UMAP(n_comp=50)

Based on Table 13 displayed above, the best clustering technique is the K-Means clustering technique when paired with UMAP (metric = cosine) dimensionality reduction. The Adjusted Random Index score was the clustering metric used to determine the best result.

Question 25:

	Accuracy
VGG No Dim. Reduction	0.6877
VGG Dim. Reduction	0.4726

The model suffers by 21% accuracy with the inclusion of dimensionality reduction. This is a significant performance drop. For the best clustering model (K-Means) in Question 24, the Adjusted Rand. Index, a metric similar to accuracy, is 0.4681 which is in the vicinity of 0.4726, the test accuracy of the similarly pre-processed VGG model.