Project 2: Data Representations and Clustering

EECS219 Winter 2023

Fengyuan Heying, Alice Lu, Dadian Zhu

Question 1:

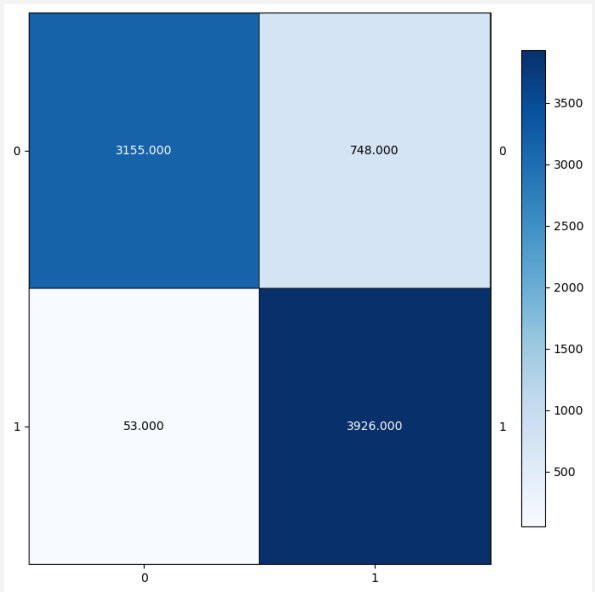
There are **7882 rows** and **23522 columns** in the TF-IDF matrix we obtained.

Question 2:

```
contingency_matrix(labels,kmeans.labels_)
```

```
array([[3155,  748],
       [  53, 3926]])
```

Contingency Table



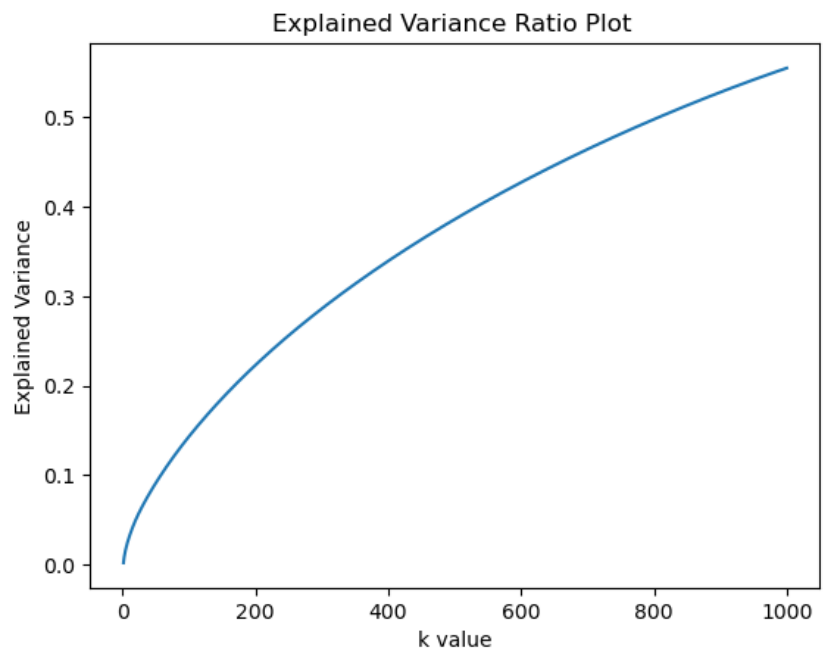
Visualize Contingency Matrix

Contingency matrix does **not** have to be square-shaped because it could be generated by different sampling methods. It should be a rectangle or square shape.

Question 3:

Clustering Measures	Score (round 4 decimals)
Homogeneity	0.5880
Completeness	0.6018
V-measure	0.5948
Adjusted Rand Index	0.6498
Adjusted Mutual Information	0.5947

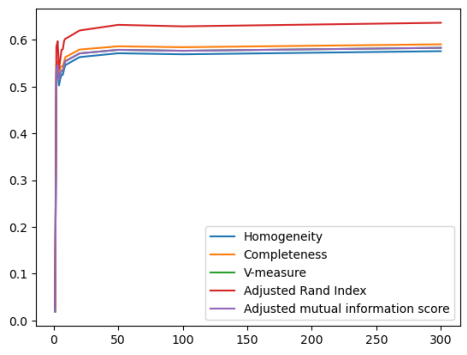
Question 4:



Explained Variance Ratio Plot

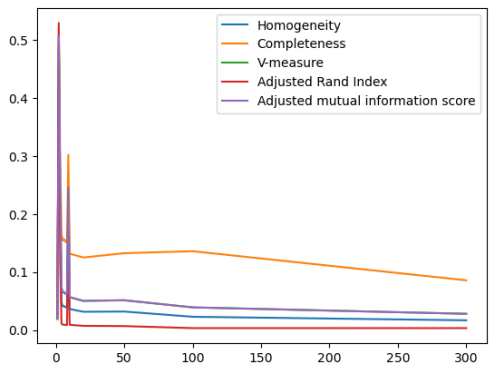
Based on the plot, we can see that as r increases, explained variance also increases.

Question 5:



5 measure scores for SVD

Based on the five measure scores for SVD, we choose r=300 as the best dimension. When r=300, it has highest measure scores which means keeping most information and better performance.



5 measure scores for NMF

According to the plot, r=2 is the best dimension because as r increases, all measure scores go down. When r=2 has the best balance between information preservation and better performance of k-means in lower dimensions.

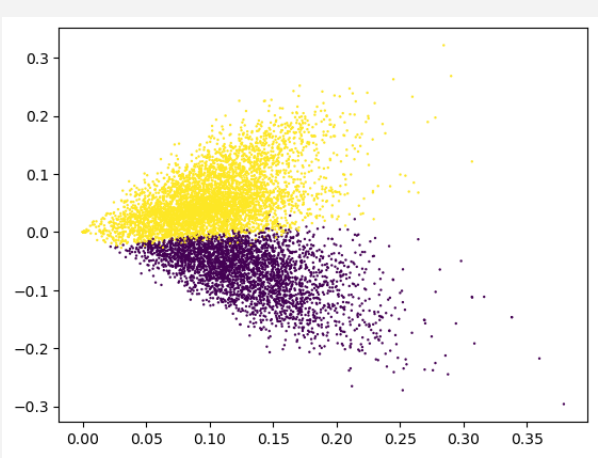
Question 6:

SVD yields better results in giving our interested Eigenvectors of input matrices. It gives more insight into data. NMF performs better in sparse matrices because it has missing-value assumptions inbuilt. It gives more insight into how much information eigenvectors hold. Although both of them are dimension reduction methods, they have different output metrics. They may preserve noise or not that useful information. So, when we apply k-means to two methods, it performs non-monotonically. We cannot predict how accuracy changes based on a certain set of principal components.

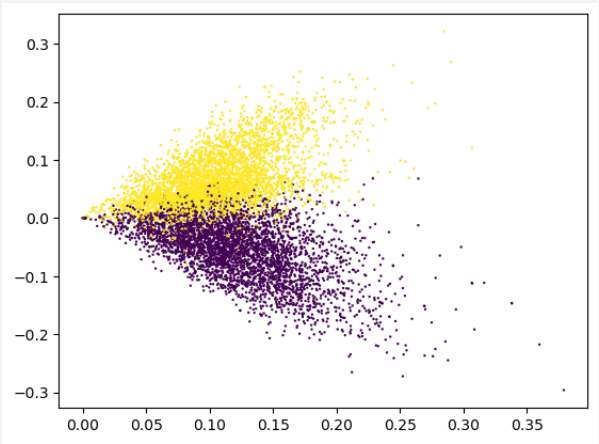
Question 7:

Compared to Question 3, their results are very close. But Question 3 will be a little bit better because it has more information and without dimension reduction. For Question 5, after dimension reduction, we only keep very useful information to cluster. Their measurement sores are also pretty good because they provide close measurement though they lost a lot of information.

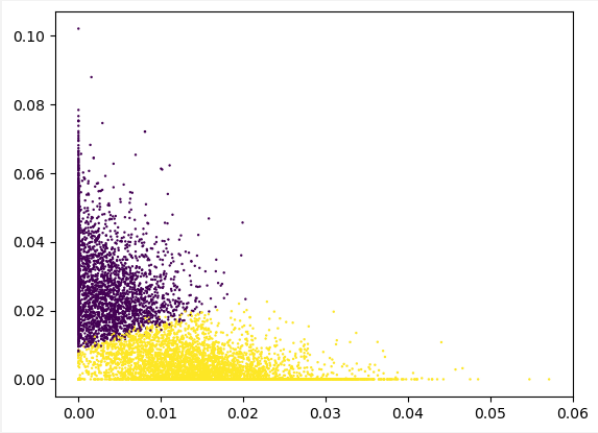
Question 8:



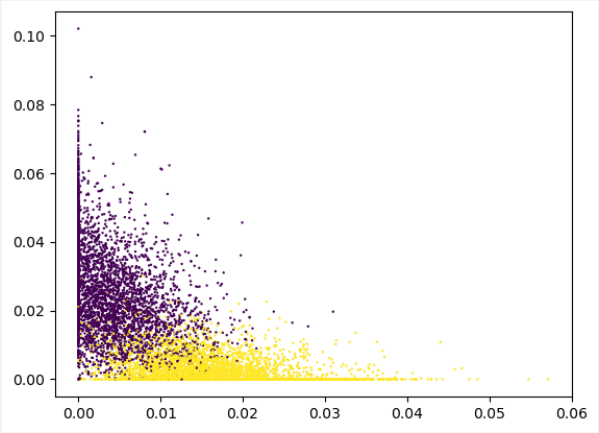
SVD clustering label



SVD Ground truth class label



NMF clustering label



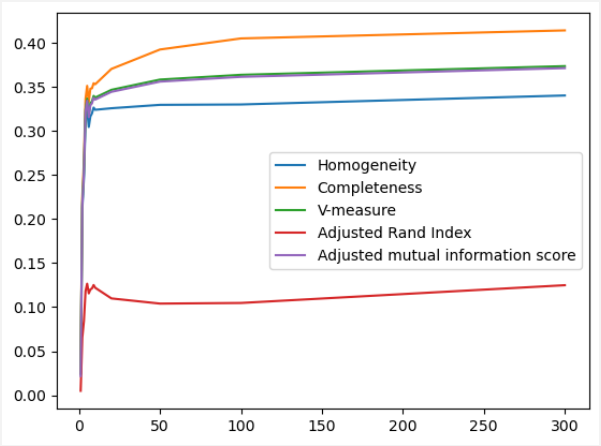
NMF Ground truth class label

Question 9:

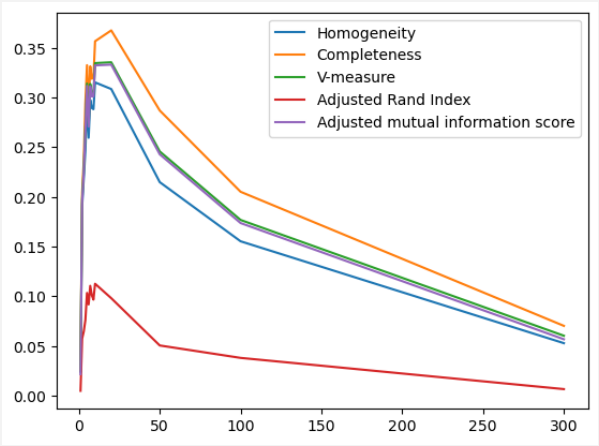
- PCA could be negative, but NMF only is positive. So the PCA plot has a negative axis, NMF does not. K-means can cluster labels with an obvious linear relationship.
- Data points of two labels distributed discreetly. NMF labels seem to have a geometric distribution. PCA labels seem to have a linear trend in the plot because PCA derived variables formed as a linear combination of the original variables that explains the most variance.

- It is kind of ideal for K-means clustering because K-means works better in geometric shapes like spherical-like shape is the best. NMF and PCA both have relatively not complicated geometric shapes.

Question 10:

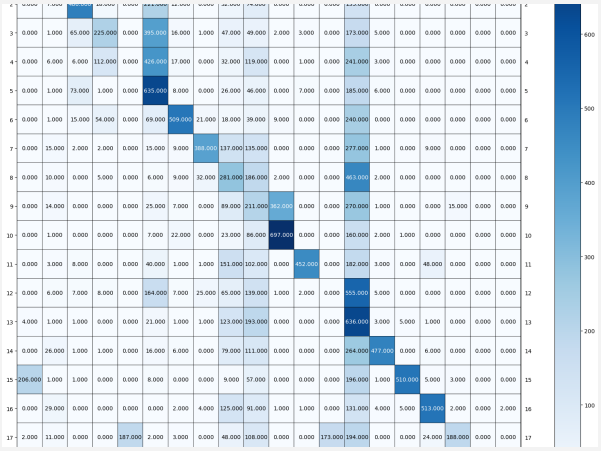


SVD scores for 20 labels

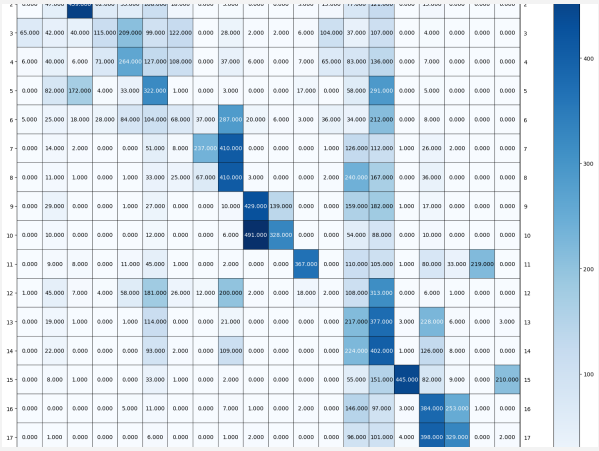


NMF scores for 20 labels

SVD best dimension	NMF best dimension
r=300	r=10



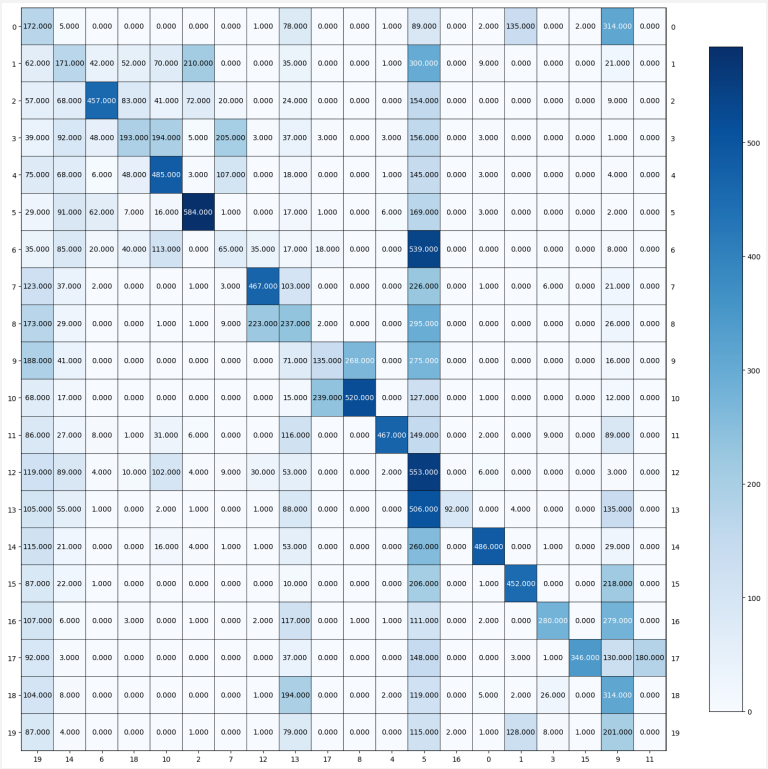
SVD matrix for 20 labels



NMF matrix for 20 labels

Compare with No Dimension Reduction:

Clustering Measures	Score (round 4 decimals)
Homogeneity	0.3479
Completeness	0.3968
V-measure	0.3707
Adjusted Rand Index	0.1221
Adjusted Mutual Information	0.3686
Average	0.3212

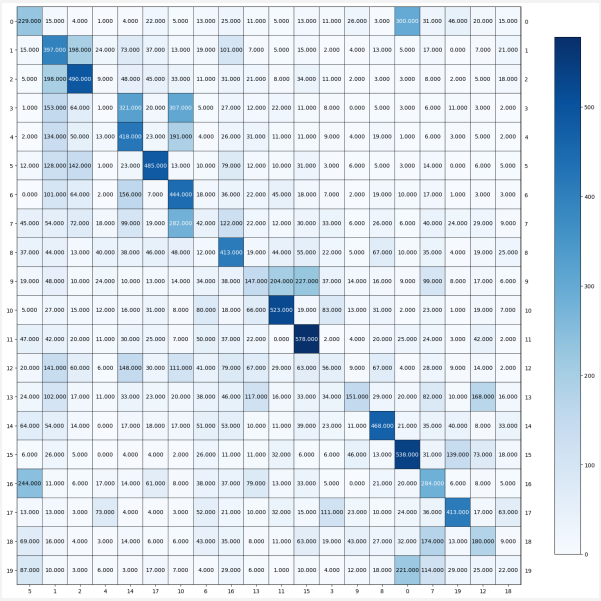


5 measure scores with no dimension reduction

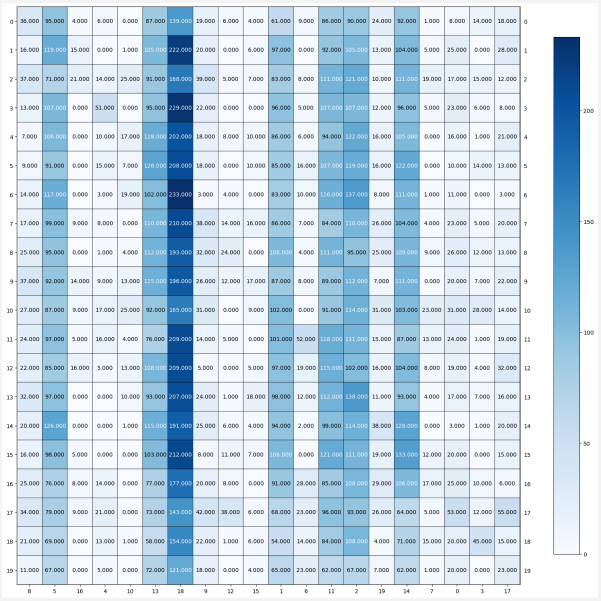
Compared with the original train data without dimension reduction, we found that PCA performs a little better than original data and NMF loses a little accuracy but not that much. It shows that dimension reduction based on both methods explained the most useful information.

Question 11:

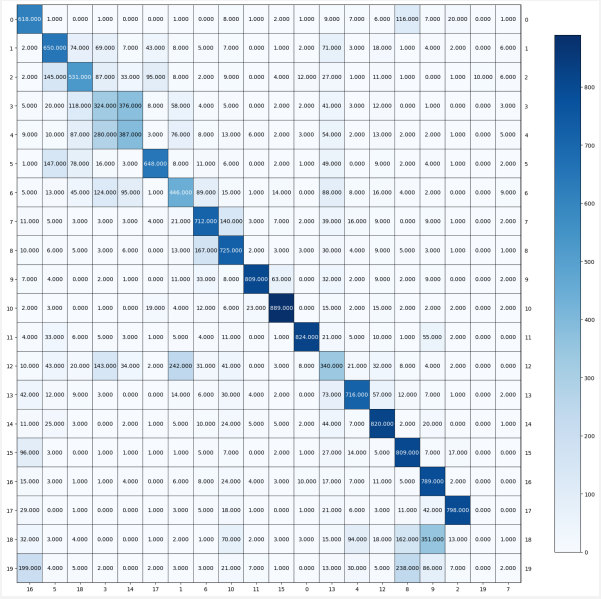
Contingency Matrix



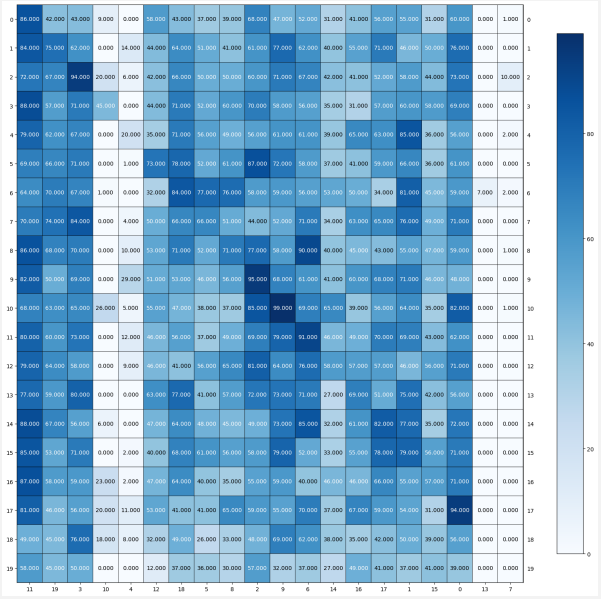
N_components = 5, metric = “cosine”



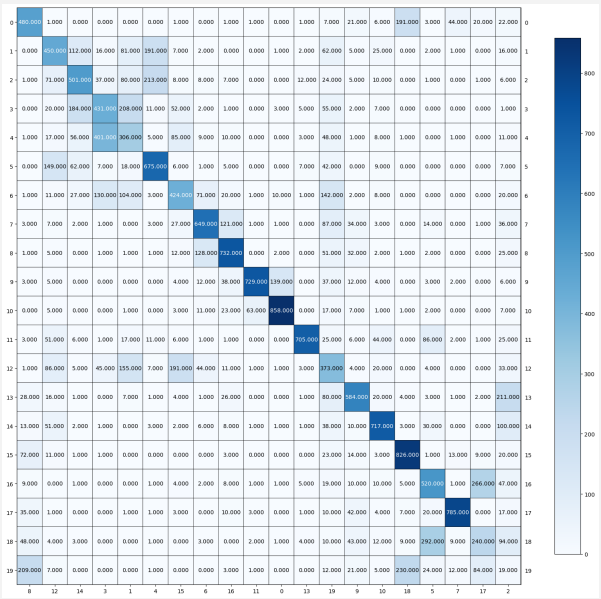
N_components = 5, metric = “euclidean”



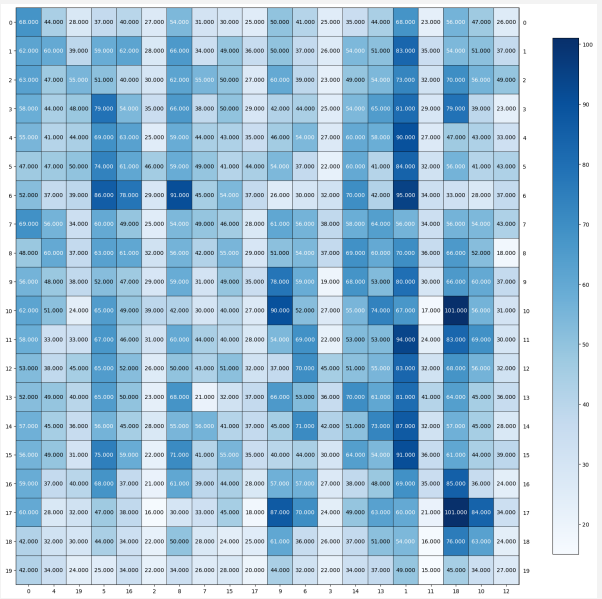
N_components = 20, metric = “cosine”



N_components = 20, metric = “euclidean”



N_components = 200, metric = “cosine”



N_components = 200, metric = “euclidean”

Evaluation Metrics

```
Homogeneity:  0.269849618855689
Completeness:  0.2789383088099403
V-measure:  0.2743187031711635
Adjusted Rand Index:  0.15665527898741563
Adjusted mutual information score:  0.2719352290367982
Average:  0.25033942777220136
```

N_components = 5, metric = “cosine”

```
Homogeneity:  0.019617738013073673
Completeness:  0.023749540632678836
V-measure:  0.02148681128315963
Adjusted Rand Index:  0.001467821635725807
Adjusted mutual information score:  0.01799063444585558
Average:  0.016862509202098707
```

N_components = 5, metric = “euclidean”

```
Homogeneity:  0.5675204514312507
Completeness:  0.586776457379573
V-measure:  0.5769878398519761
Adjusted Rand Index:  0.4537327765120024
Adjusted mutual information score:  0.5755893219320715
Average:  0.5521213694213747
```

N_components = 20, metric = “cosine”

```
Homogeneity:  0.010802118457913798
Completeness:  0.01147887995543727
V-measure:  0.011130221253326376
Adjusted Rand Index:  0.001163079430442902
Adjusted mutual information score:  0.007826418180672319
Average:  0.008480143455558533
```

N_components = 20, metric = “euclidean”

```
Homogeneity:  0.5648023709225738
Completeness:  0.5662550226299147
V-measure:  0.5655277639336692
Adjusted Rand Index:  0.4096891710780805
Adjusted mutual information score:  0.564124079534736
Average:  0.5340796816197948
```

N_components = 200, metric = “cosine”

```
Homogeneity:  0.006638457414285602
Completeness:  0.00671448489577333
V-measure:  0.006676254716666776
Adjusted Rand Index:  0.001176056028020816
Adjusted mutual information score:  0.0034520426464795585
Average:  0.004931459140245217
```

N_components = 200, metric = “euclidean”

Question 12:

From the contingency matrix, we can see that K-Means using euclidean as a metric randomly cluster these 20 categories no matter how many components we use when reducing the

features with UMAP. While K-Means using cosine as metric is able to correctly cluster these 20 categories as we can see from the diagonal of the matrix. From the matrix, we can see that K-Means using UMAP with n_components of 20 and with a metric of cosine works the best since under this setting, more data points are correctly clustered to its own category since each class is heavily clustered into one cluster, which is unlike other setting. Data points from the same class are clustered into more than one cluster.

From the metric, we can see that under the setting of 20 n_components and with a metric of cosine, we have the highest adjusted rand score, highest homogeneity, completeness, V-measure, and Silhouette Coefficient. This means that not only this setting can correctly cluster data points of the same category, it also can cluster most data points of the same class into the same cluster according to completeness among all these settings, the cluster under this setting also contains most data points from a single class.

Question 13:

UMAP with dimension=20 and metric = cosine works the best for K-Means clustering task on the 20-class text data.

Best dim	Best metric
20	‘cosine’

Question 14:

Evaluation metrics with 4 linkage criteria

```
agglomerative clustering with single as linkage
Homogeneity:  0.018518066200508162
Completeness:  0.3784553878324676
V-measure:  0.03530846637034055
Adjusted Rand Index:  0.0004658489726512295 7
Adjusted mutual information score:  0.030335846120322786
Average:  0.09261672309925809
```

single

```
agglomerative clustering with ward as linkage
Homogeneity:  0.5454727582859332
Completeness:  0.5860841782559056
V-measure:  0.5650496991835158
Adjusted Rand Index:  0.414728873671653
Adjusted mutual information score:  0.563581383525731
Average:  0.5349833785845477
```

ward

```
agglomerative clustering with average as linkage
Homogeneity:  0.49786642279858495
Completeness:  0.6122221908347892
V-measure:  0.5491541276352487
Adjusted Rand Index:  0.35527965289559293
Adjusted mutual information score:  0.5475130100562092
Average:  0.512407080844085
```

average

```
agglomerative clustering with complete as linkage
Homogeneity:  0.5253430484642387
Completeness:  0.5879411059108111
V-measure:  0.5548821865159991
Adjusted Rand Index:  0.39929994582505396
Adjusted mutual information score:  0.5533309061404217
Average:  0.5241594385713049
```

complete

Agglomerative clustering with ward linkage has a higher accuracy compared to agglomerative clustering with complete linkage while with complete linkage the completeness is higher than using ward linkage.

Question 15:

Evaluation metrics with different min_cluster_size

```
Homogeneity:  0.4014796237127924
Completeness:  0.5965591103800566
V-measure:  0.47995397167729903
Adjusted Rand Index:  0.1826151459051232
Adjusted mutual information score:  0.47868226442277634
Average:  0.42785802321960953
```

min_cluster_size=20

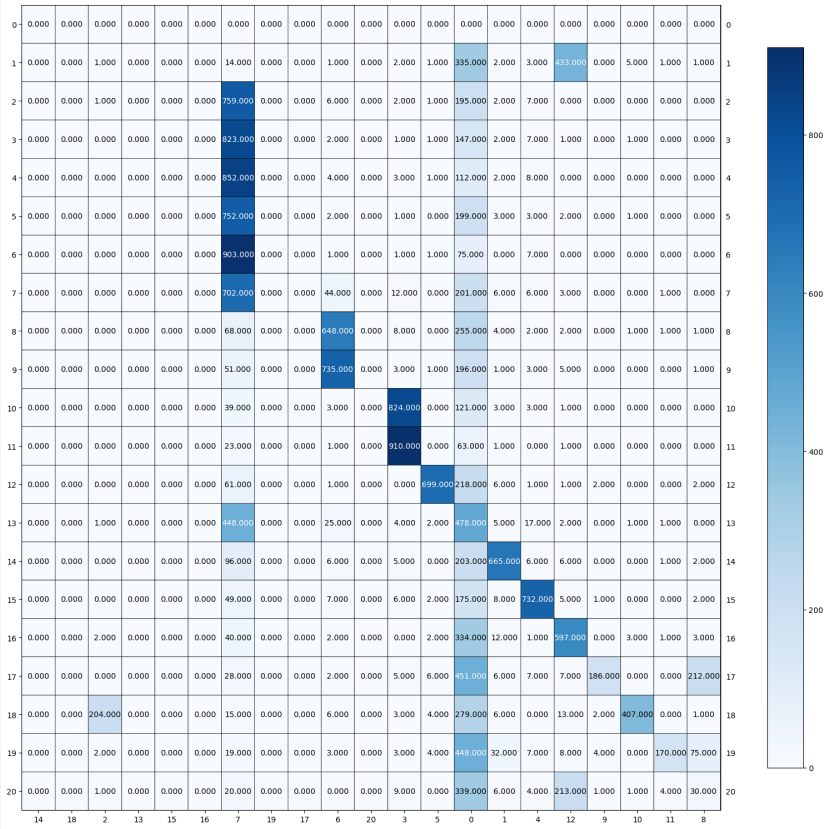
```
Homogeneity:  0.4014796237127924
Completeness:  0.5965591103800566
V-measure:  0.47995397167729903
Adjusted Rand Index:  0.1826151459051232
Adjusted mutual information score:  0.47868226442277634
Average:  0.42785802321960953
```

min_cluster_size=100

```
Homogeneity:  0.4029942212998046
Completeness:  0.5995602021644008
V-measure:  0.48200734272102636
Adjusted Rand Index:  0.19626501027391074
Adjusted mutual information score:  0.4809551993595842
Average:  0.43235639516374536
```

min_cluster_size=200

Question 16:



HDBSCAN with min_cluster_size of 100, min_samples of 100

- 100 clusters;
- 16 plot the best one's contingency matrix;
- -1 indicates data points that are not assigned to any cluster.

Question 17:

Module	Methods	Hyperparameters
Dimensionality Reduction	UMAP	r=200
Clustering	K-Means	k=20

```
HDBSCAN with min_cluster_size=200 with a reducer UMAP on n_components 20
Homogeneity:  0.4149355581855768
Completeness:  0.6083158841492249
V-measure:  0.4933526218476096
Adjusted Rand Index:  0.21382233586627958
Adjusted mutual information score:  0.4923297948229103
```

Metrics with best module

Question 19:

We can use a VGG network that's trained on a dataset with different classes as targets because we can transfer the learning from the pretrained dataset to the custom dataset and fit on the custom dataset on a higher layer. That is why we can use a pretrained VGG model to have discriminative power for a custom dataset.

Question 20:

Load data in batch and extract features from input image extract features from the pretrained VGG model. Perform average pooling on the input image of pretrained VGG model to downsample input features flatten features that are downsampled through pooling to map features of the image into one-dimensional features.

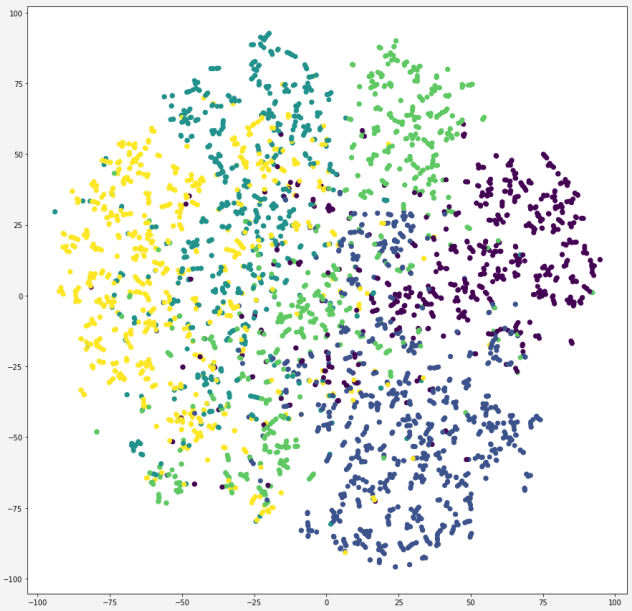
Question 21:

- The shape of a batch of extracted features is (64, 4096), which indicates that for each image 4096 features are extracted.
- The shape of a batch of images is ([64, 3, 224, 224]) which means there are 64 images in a batch with 3 color channels, and each image has 224 pixels in height and 224 pixels in width.
- Therefore, there are 224*224=50176 pixels per image in the original image.

Question 22:

The extracted features are pretty dense with most of the entries non-zero compared with sparse TF-IDF where most entries are 0.

Question 23:



t-SNE scatter plot of Vision features

t-SNE measures similarity between pairs of points in both high dimension and low dimension and then tries to use the set of similarity measured in low dimension to reflect the set of similarity measured in high dimension as best as possible. As we can see, with t-SNE the data points with different ground truth labels are well separated, while with PCA, data points with different labels are squeezed together and overlapping with each other. This happens because the high dimensional similarity is not well represented in one dimension with PCA, while with t-SNE, high dimensional similarity is reflected in a low dimension pretty well by minimizing the KL divergence.

Question 24:

Module	Methods	Hyperparameters
Dimensionality Reduction	UMAP	n_components=50
Clustering	K-Means	k=5
Score (around 4 decimal points): 0.3951		

Question 25:

MLP classifier	Original VGG	Reduced-Dimension VGG
Test Accuracy	0.9114	0.8896

The performance went down from 0.911 to 0.890. The performance drop is not that significant. This indicates that reducing dimension using autoencoder well reflects the original features. The success in classification makes sense in the context of clustering results because in this task, we are given labels and we train the classifier to learn from data, and classifiers we better learn about the data. However, clustering algorithms learn data without given labels. Thus, it doesn't perform as well as the classifier.