# **Question 1:**

The dimensions of the TF-IDF matrix are (4732, 17131). (We only fetched the ‘train’ part per Piazza's replies )

# **Question 2:**

# Contingency table of the clustering result:

# [1940 403]

# [ 45 2344]



# The contingency table does not have to be square-shaped. It can have various shapes, depending on the number of classes being compared.

# **Question 3:**

The contingency matrix for K-means clustering:

| Homogeneity score | 0.5852566172247848 |
| --- | --- |
| Completeness score | 0.5964216999698374 |
| V-measure score | 0.5907864119779943 |
| Adjusted rand score | 0.6570834245592161 |
| Adjusted mutual info score | 0.5907234072569751 |

# 

# 

# **Question 4:**

The plot of the percentage of variance for γ = 1 to 1000:

# 

# **Question 5:**

# 

# 

The optimal γ for SVD is: 300

Average score is: 0.5993380312319259.

The optimal γ for NMF is: 2

Average score is: 0.5531579294756476.

It is worth noting that the optimal γ is chosen based on the average value of the 5-measures, indicating that each measure score is weighted equally. The optimal γ will change if we weigh the 5-measures differently.

# 

# **Question 6:**

For SVD, the 5-measures increase → decrease → increase → remain constant. This non-monotonic behavior can be interpreted as the following. A low γ value provides limited information, hence lower scores. A medium γ value provides more information and a relatively low dimension, therefore, higher scores. A high γ value provides more information but makes Euclidean distance measuring inefficient, thus lower scores. It is worth noting that the scores are still high at γ = 300, but the scores could drop as we extend the γ range to 1000 or above.

For NMF, the 5-measures reach the peak value at a low γ value. Then, the 5-measures slightly fluctuate and keep decreasing as the γ value increases. This is because an increase in γ means more components in the NMF, hence higher dimensions. K-mean measure suffers from higher dimensions as the distance measuring method becomes inefficient, hence lower scores.

# **Question 7:**

| Scores | TF-IDF (Q3) | SVD (γ=300) | NMF (γ=2) |
| --- | --- | --- | --- |
| Homogeneity | 0.5852566172247848 | 0.5806135302171918 | 0.4950453839087971 |
| Completeness | 0.5964216999698374 | 0.5923574831989338 | 0.5174355469228464 |
| V-measure | 0.5907864119779943 | 0.586426715642444 | 0.5059928956174533 |
| Adjusted rand | 0.6570834245592161 | 0.6509294230254073 | 0.5395035409494571 |
| Adjusted mutual info | 0.5907234072569751 | 0.5863630040756529 | 0.505915879956075 |

# 

These measures on average are actually not better than those computed in Question 3. However, the measures calculated in SVD are almost identical to those in Question 3. This is a surprising result. The measures are expected to be higher as the dimension goes down. The potential explanation is that SVD and NMF functions treat the clusters as a whole, resulting in a loss in the local data structure. Therefore, the performance decreases.

Furthermore, the shape of the clustering is not isotropically shaped after SVD and NMF reduction, as we can see in Question 8. Also, we can argue that the optimal γ is not in the range of 1-300, and the measures of SVD or NMF could be improved.

# 

# 

# 

# 

# **Question 8:**

# 

# **Question 9:**

From the visualization, we can see that there are two triangular-shaped clusterings, and two clusterings are very close to each other.

The data points are distributed in a way that class 1 is stacked on class 2. It is worth noting that the NMF method does not contain any negative values. Therefore, numerous points are squeezed at zero boundaries.

The data distribution is not ideal for K-Means clustering because the shape of the two clusterings is not round, especially in NMF. Therefore, the K-means algorithm may fail.

# 

# **Question 10:**

Setting for SVD is TruncatedSVD(n\_components=8, random\_state=0). The n\_components is chosen based on the highest measures score on average by iterations from 1 to 300.

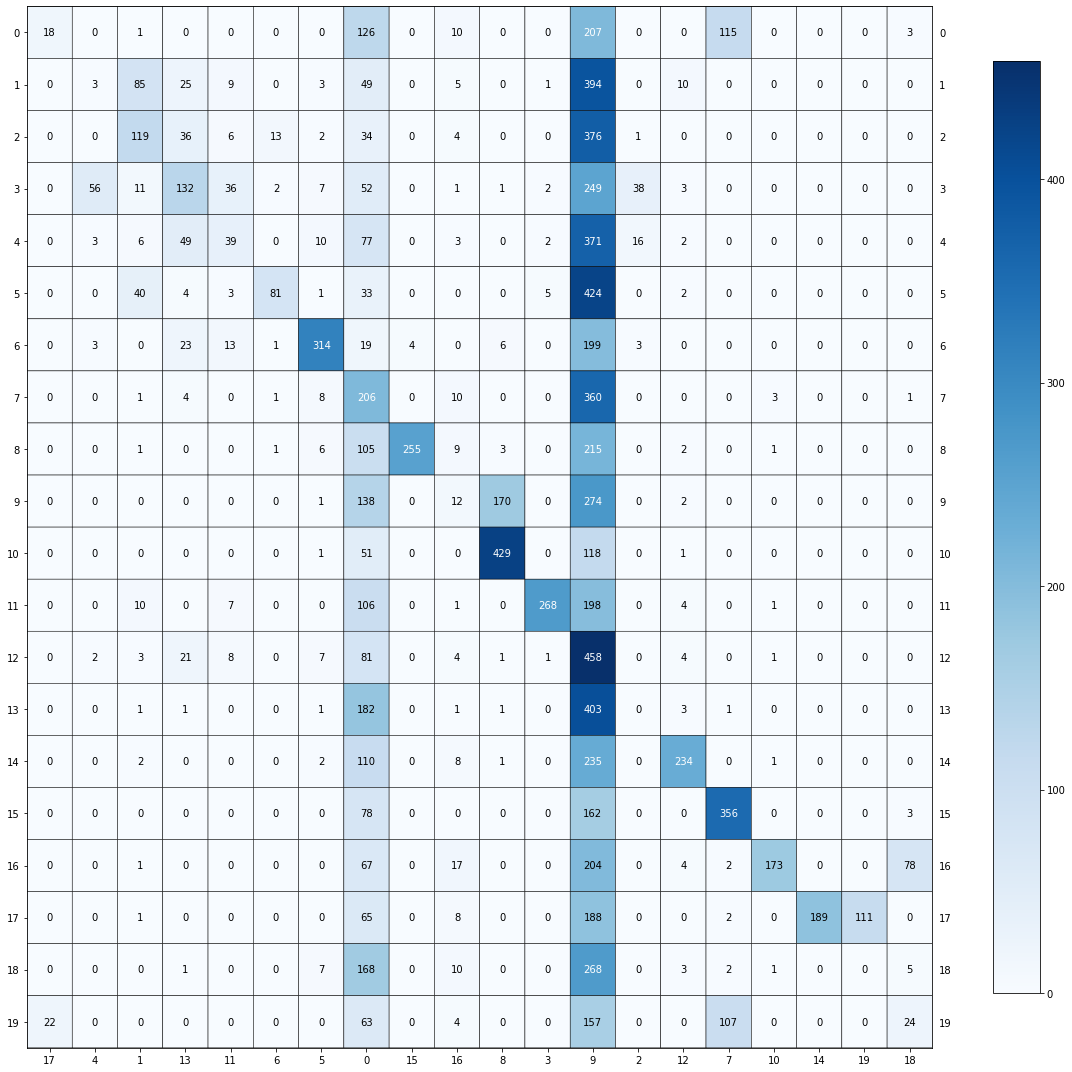
# 

# 

| Homogeneity score | 0.34356185740291545 |
| --- | --- |
| Completeness score | 0.3752679359730747 |
| V-measure score | 0.3587156522858008 |
| Adjusted rand score | 0.13936178731937282 |
| Adjusted mutual info score | 0.3550675876242833 |

# 

Setting for NMF is NMF(n\_components=50, init = 'random', random\_state=0, max\_iter=500). The n\_components is chosen based on the highest measures score on average by iterations.



# 

| Homogeneity score | 0.2760010367197732 |
| --- | --- |
| Completeness score | 0.42097610555511633 |
| V-measure score | 0.33341076635089767 |
| Adjusted rand score | 0.05119369194370699 |
| Adjusted mutual info score | 0.32892904653461336 |

# 

# 

# 

# 

# **Question 11:**

# **n\_components = 5, metric = “cosine”**

# 

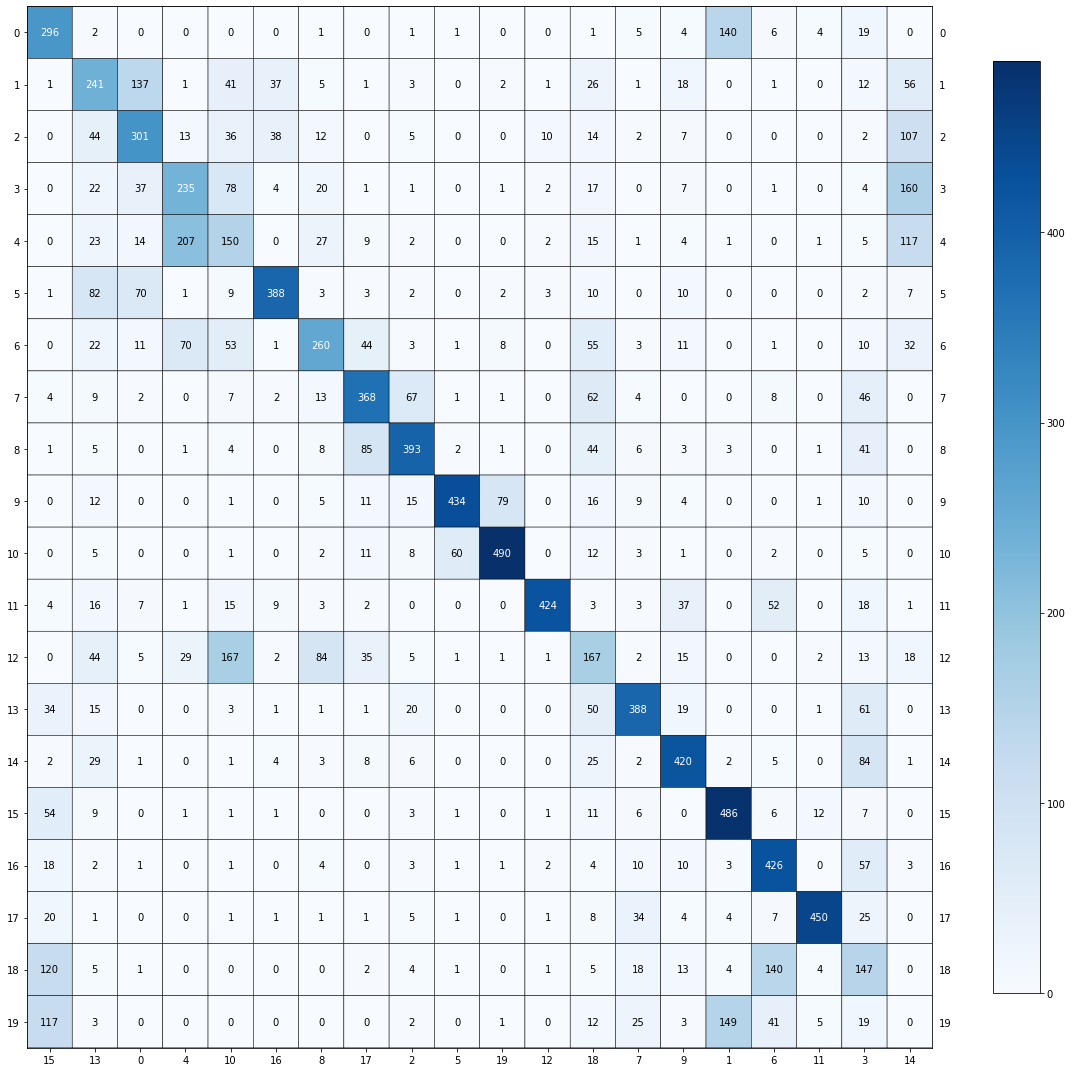
| Homogeneity score | 0.31324948075378223 |
| --- | --- |
| Completeness score | 0.3218999728331573 |
| V-measure score | 0.31751581860045097 |
| Adjusted rand score | 0.2023259975755401 |
| Adjusted mutual info score | 0.3137708115046873 |

# **n\_components = 20, metric = “cosine”**

# 

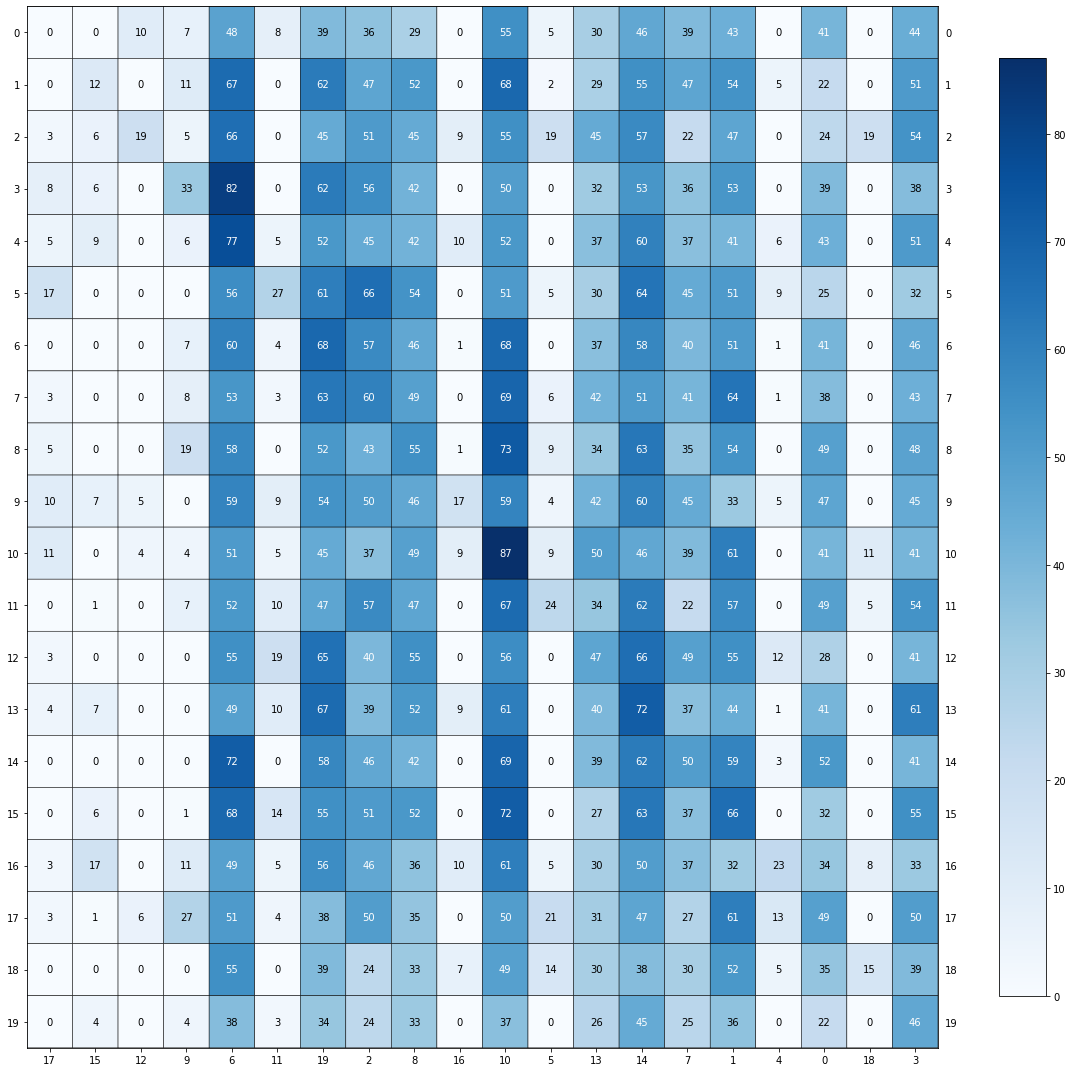
| Homogeneity score | 0.57526983707966 |
| --- | --- |
| Completeness score | 0.5825063235461496 |
| V-measure score | 0.5788654650880622 |
| Adjusted rand score | 0.44236683204571786 |
| Adjusted mutual info score | 0.5765737177023393 |

# **n\_components = 200, metric = “cosine”**



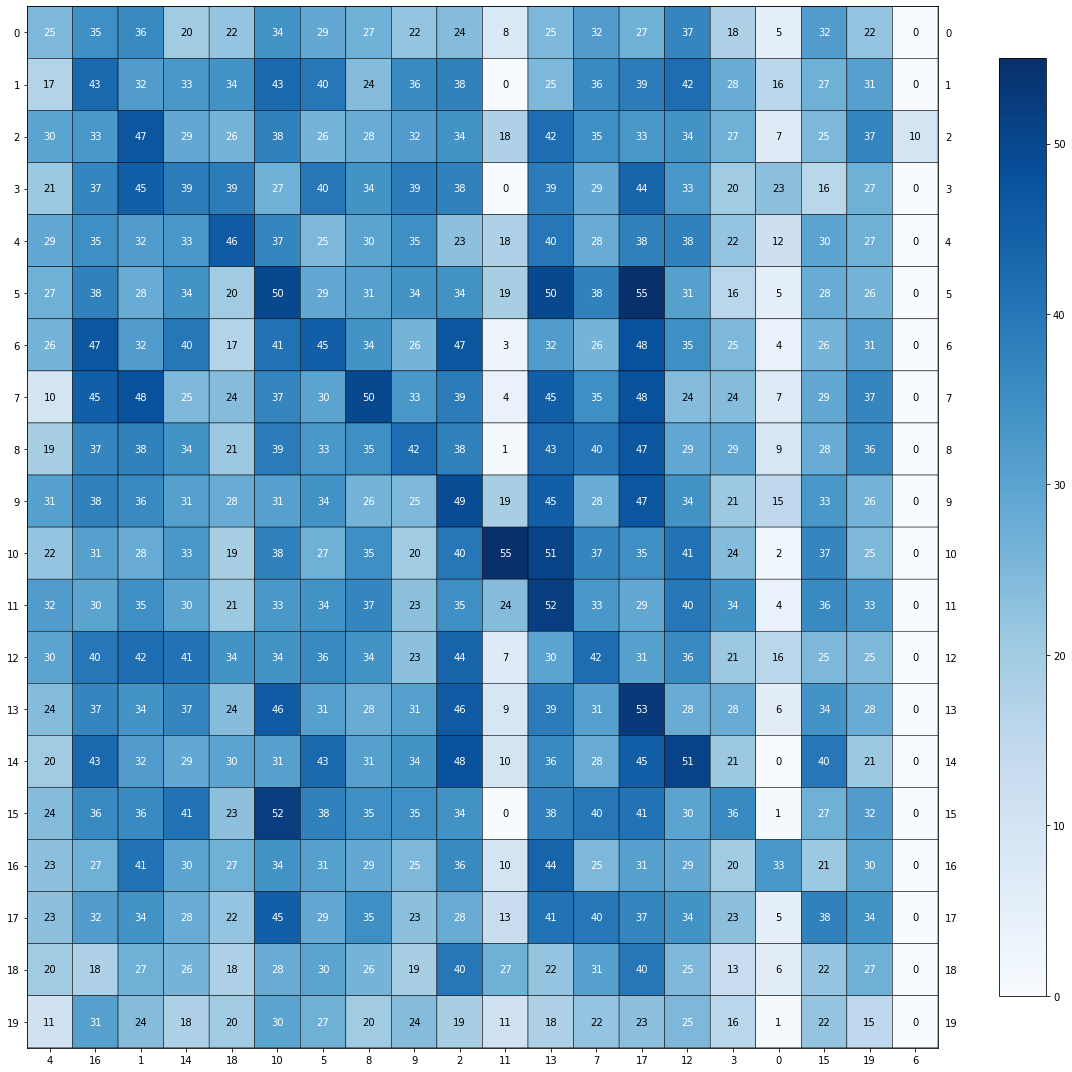
| Homogeneity score | 0.5526910631698617 |
| --- | --- |
| Completeness score | 0.5535286111520253 |
| V-measure score | 0.5531095200961513 |
| Adjusted rand score | 0.3973085272375704 |
| Adjusted mutual info score | 0.5506938573363206 |

# **n\_components = 5, metric = “euclidean”**



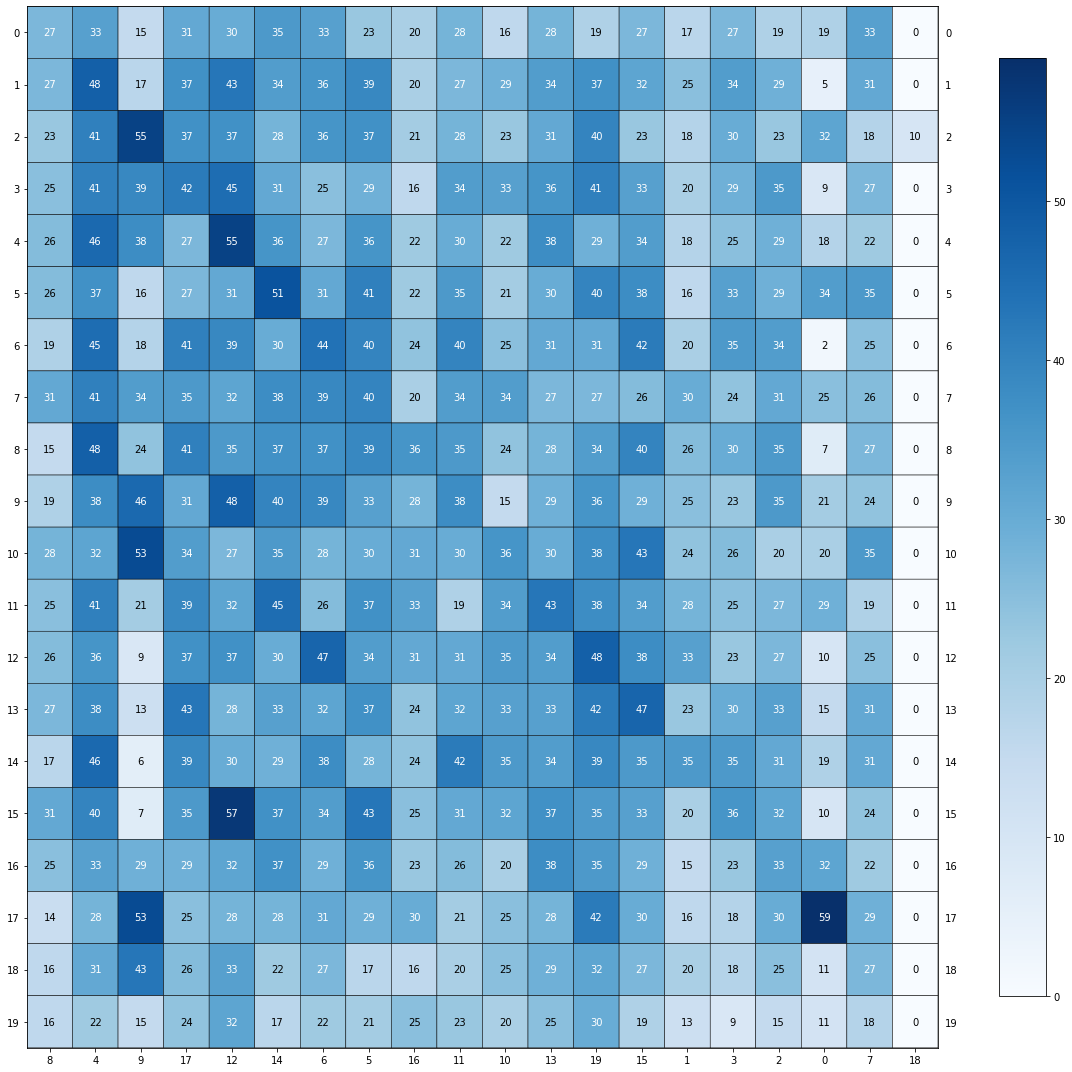
| Homogeneity score | 0.024083936687990457 |
| --- | --- |
| Completeness score | 0.027465296976092388 |
| V-measure score | 0.025663717051528107 |
| Adjusted rand score | 0.0012960653614713727 |
| Adjusted mutual info score | 0.019895657354386854 |

# **n\_components = 20, metric = “euclidean”**



| Homogeneity score | 0.011587997924333173 |
| --- | --- |
| Completeness score | 0.011922251393769694 |
| V-measure score | 0.011752748559582674 |
| Adjusted rand score | 0.0009652225345590279 |
| Adjusted mutual info score | 0.006328490929249487 |

# **n\_components = 200, metric = “euclidean”**



| Homogeneity score | 0.010906855017180063 |
| --- | --- |
| Completeness score | 0.011117021016700026 |
| V-measure score | 0.011010935247325689 |
| Adjusted rand score | 0.0013189015938329586 |
| Adjusted mutual info score | 0.005612514721267914 |

# **Question 12:**

# The setting with n\_components = 20, metric = “cosine” works the best in terms of the contingency matrices. Since the matrix we get from these parameters, the combination has the most matched labels on the diagonal, which means that the clusters we got and the true labels we have are mostly matched. The one with n\_components = 20 has 6104 labels matched on the diagonal while the 200 one has 6700 labels matched on the diagonal. Thus, n\_components = 20 yields the best performance.

# The setting with n\_components = 20, metric = “cosine” is also the best in terms of all the evaluation metrics since it has the highest metrics scores.

# 

# **Question 13:**

| Score | Sparse TF-IDF | SVD | NMF | UMAP |
| --- | --- | --- | --- | --- |
| Homogeneity score | 0.34810501531936705 | 0.34356185740291545 | 0.276001036719773 | 0.57526983707966 |
| Completeness score | 0.40067474927068214 | 0.3752679359730747 | 0.42097610555511633 | 0.5825063235461496 |
| V-measure score | 0.3725444952677556 | 0.3587156522858008 | 0.3334107663508976 | 0.5788654650880622 |
| Adjusted rand score | 0.12321219488700486 | 0.13936178731937282 | 0.05119369194370699 | 0.44236683204571786 |
| Adjusted mutual info score | 0.36889185403329405 | 0.3550675876242833 | 0.32892904653461336 | 0.5765737177023393 |

# UMAP-reduction is the best dimensionality reduction method which has the highest score for all 5 clustering metrics.

# 

# **Question 14:**

**linkage = “ward”**

| Homogeneity score | 0.5426632915800996 |
| --- | --- |
| Completeness score | 0.5739911022267402 |
| V-measure score | 0.5578877450348046 |
| Adjusted rand score | 0.40201854448225194 |
| Adjusted mutual info score | 0.5554071004883744 |

**linkage = “single”**

| Homogeneity score | 0.10724228088667982 |
| --- | --- |
| Completeness score | 0.6733207858585607 |
| V-measure score | 0.1850163296733359 |
| Adjusted rand score | 0.02033546495900337 |
| Adjusted mutual info score | 0.17921225565596471 |

# 

# **Question 15:**

**min\_cluster\_size = 20**

| Homogeneity score | 0.47674885802894584 |
| --- | --- |
| Completeness score | 0.49449796853857125 |
| V-measure score | 0.4854612346720681 |
| Adjusted rand score | 0.13843181451482206 |
| Adjusted mutual info score | 0.4750120846543949 |

# **min\_cluster\_size = 100**

| Homogeneity score | 0.4029950938532099 |
| --- | --- |
| Completeness score | 0.5960683509188109 |
| V-measure score | 0.4808756086081582 |
| Adjusted rand score | 0.16649491874635639 |
| Adjusted mutual info score | 0.47875290035666074 |

# **min\_cluster\_size = 200**

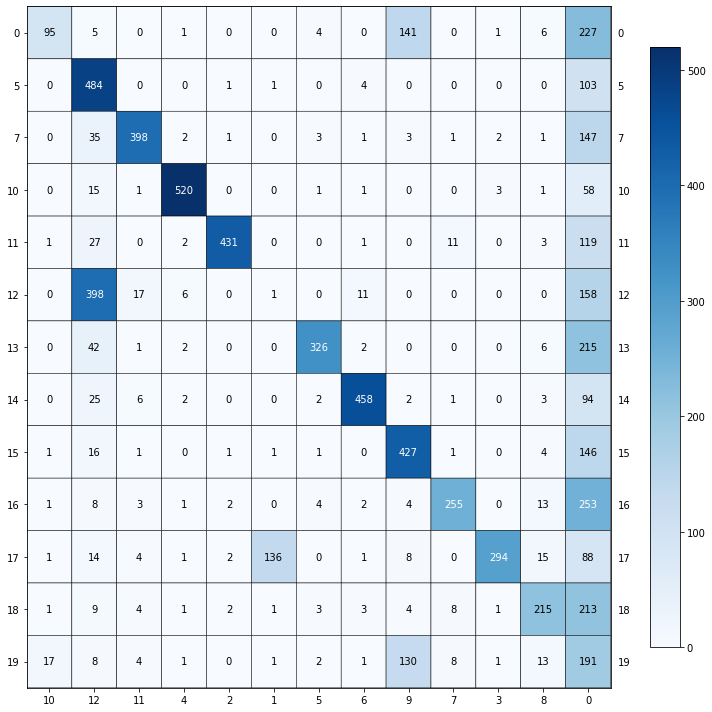
| Homogeneity score | 0.38149936498216575 |
| --- | --- |
| Completeness score | 0.5938503941689505 |
| V-measure score | 0.46455857736006734 |
| Adjusted rand score | 0.16624870152350765 |
| Adjusted mutual info score | 0.46289379036407907 |

# As the min\_cluster\_size gets larger, the homogeneity score will decrease but the completeness score will increase. The decreasing rate is quicker than the increasing rate, which results in a decreasing v-measure score. Both the adjusted rand index score and adjusted mutual information score will increase first and then decrease.

# 

# **Question 16:**

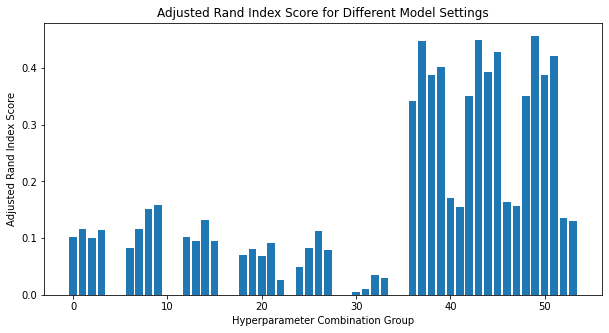
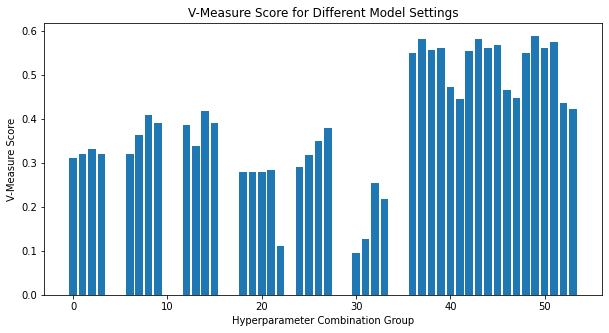
The best clustering model we use has min\_cluster\_size = 100. The major comparison is between min\_cluster\_size = 20 and 100. The min\_cluster\_size = 200 one has significantly worse scores in all aspects. While min\_cluster\_size = 20 has a slightly higher v-measure score, the other one has a higher adjusted rand index score and adjusted mutual information score.



13 clusters are given by the model. The label “-1” indicates that a data point, such as noise, has not been assigned to any cluster, which can happen when using certain clustering algorithms.

# **Question 17:**

There are 9 different dimensionality reduction methods and 6 different clustering methods. We plot the scores for these 54 hyperparameter settings. The v-measure score, adjusted rand index score, and adjusted mutual information score graphs are as follows:

****

# 

# Since the un-reduced data takes a long time to converge, the result is more likely to be worse. In our settings, the last three groups of bars use UMAP for dimensionality reduction. Each group of bars has its second bar being the highest score, which in our setting corresponds to the k-means with k = 20.

| n\_components | V-measure score | Adjusted rand index score | Adjusted mutual information score |
| --- | --- | --- | --- |
| 5 | 0.4450937468344214 | 0.15401066815164133 | 0.443515051443478 |
| 20 | 0.4477847487383584 | 0.15664372827401943 | 0.4462153591991176 |
| 200 | 0.4228258789977064 | 0.13008260013353123 | 0.4213319487848216 |

# The best model is UMAP with n\_components = 20 using K-Means clustering with k = 20.

# **Question 18:**

First cluster the data first using 2 classes the same as the first few parts in this project, then further cluster the data within these two classes using finer classes.

# **Question 19:**

Because features learned from the VGG network are generic representations of the image data, like the general image pattern and structure. As a result, even if the VGG network is trained with a dataset different from our custom dataset, it can still read the image information from our dataset and output the feature

# **Question 20:**

For the feature extraction part in the helper code, it first checks whether the training data is downloaded (the .tgz file). If not, it will download the data online. Then it creates a base class called FeatureExtractor for the neural network module from PyTorch. Inside the class, it defines the parameters (layers and forward) for VGG16 to vectorize the image from 224x224 images to vectors with length=4096. After that, it enables CUDA for GPU usage if GPU is available. Then it crops the images into 224x224 size and converts them into tensors. Then it uses DataLoader from PyTorch to enable iterations within the training data. Lastly, in the iteration, it outputs the VGG feature vectors and the labels and saves the results in a .npz file

# **Question 21:**

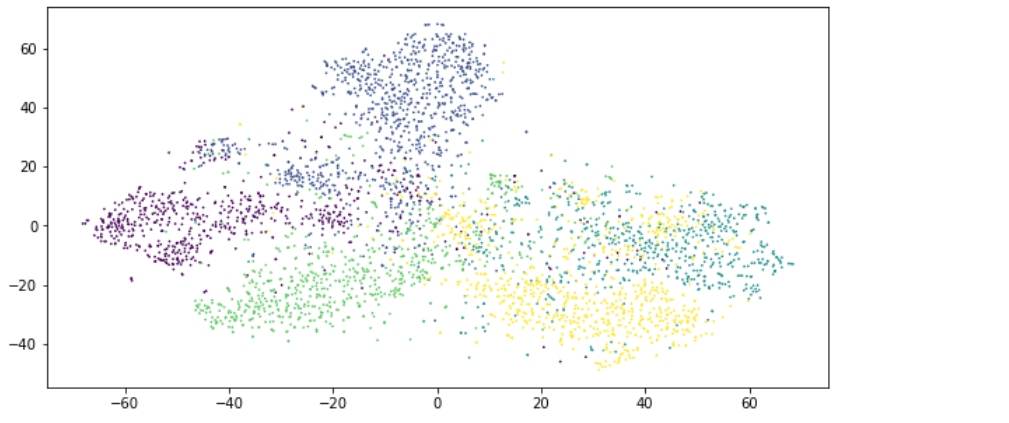
There are 224x224=50176 pixels in each original image. There are 4096 features extracted from the VGG network, so the size is 1x4096.

# **Question 22:**

The extracted features of the VGG network are dense. Unlike sparse TF-IDF features where most elements in the matrix are 0s, the VGG network outputs vectors with few 0 elements.

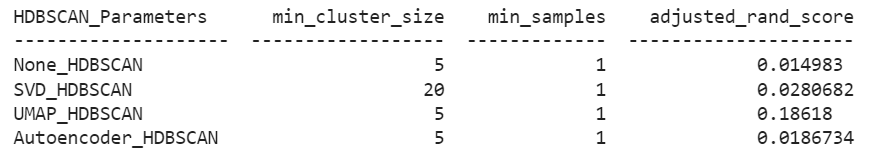
# **Question 23:**

As we can see from the graph, there are five classes which are represented by five different colors. The pattern is shown below, and t-SNE works properly.

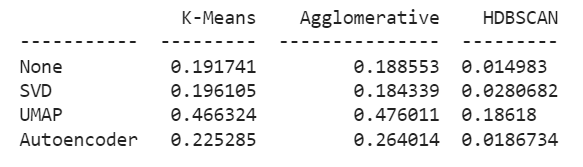


# **Question 24:**

Because HDBSCAN is not one of the sklearn built-in functions, I performed the gridsearch manually using parameters cluster\_size = [5, 20, 100] and samples = [1, 5, 10]. The result is shown below. According to the table, the best parameters are min\_cluster\_size=5 and min\_samples=1.



Then the predictions are performed for each pair of clustering and dimensionality reduction methods mentioned in the lab manual table. The result is shown in the below table. According to the result, the best adjusted\_rand\_score result is UMAP and K-Means, with a score of 0.476.



# **Question 25:**

In this part of the lab, I performed training and testing using the MLP classifier on VGG features. I performed the testing for both the original dataset and the dimensionally reduced dataset (using UMAP). The accuracy is calculated by num\_correct/num\_total.

As shown in the figures below, the accuracy for the original dataset is 90.6%, while the accuracy for the reduced dataset is 83.92%. According to the result, the performance does suffer with the reduced-dimension representations, and the suffering is significant (7.37%).

Compared to the results from Q24, this success makes sense. This classification tends to have better results since MLP classification is supervised and requires labels as input while clustering does not, and also we are calculating accuracy for classification but rand score for clustering.

