Large Scale Data Mining: Models and Algorithms

ECE 219 Spring 2023

Instructor: Professor Vwani Roychowdhury

UCLA, Department of ECE

Project 2 Report: Data Representations and Clustering

Group Member

Syed R Nawshad	*StudentID* : 805871517
Huijun S Hao	*StudentID* : 105863846
Chao Zhang	*StudentID* : 904506124

May 6, 2023

Part 1 - Clustering on Text Data

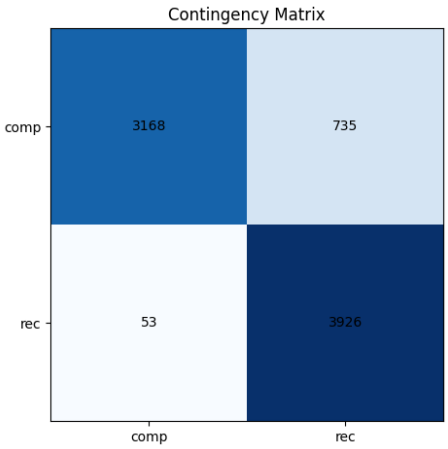
Clustering with Sparse Text Representations

Question 1

The raw data, documents, were transformed into TF-IDF vectors using min-df of 3, excluding stop words and removing headers and footers. There are 7882 samples and 23522 features observed.

Question 2

To evaluate the effectiveness of the K-means clustering algorithm applied to the TF-IDF data, several clustering measures were used. First, a contingency table was generated to compare the clustering result against the ground truth labels. The contingency table is a matrix whose entries (i,j) indicate the number of data points that belong to the i'th class and the j'th cluster. The plotmat.py module was used to visualize the contingency table. It should be noted that the contingency matrix does not have to be square-shaped. The following contingency matrix was observed:



Question 3

As for the clustering measures, five measures were utilized to assess the performance of the K-means clustering algorithm. These measures are:

Homogeneity is a measure of how "pure" the clusters are. If each cluster contains only data points from a single class, the homogeneity is satisfied.

Completeness indicates how much of the data points of a class are assigned to the same cluster.

V-measure is the harmonic average of homogeneity score and completeness score.

Adjusted Rand Index is similar to accuracy, which computes similarity between the clustering labels and ground truth labels. This method counts all

pairs of points that both fall either in the same cluster and the same class or in different clusters and different classes.

Adiusted mutual information score measures the mutual information between the cluster label distribution and the ground truth label distributions. By utilizing these measures, we can comprehensively evaluate the quality of the K-means clustering results.

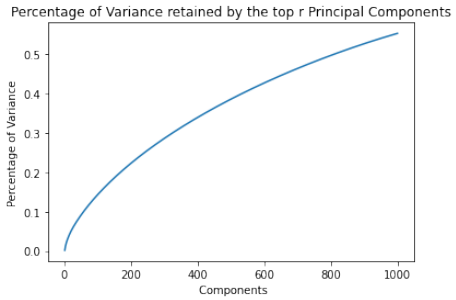
The following metrics were observed:

Metric	Score
Homogeneity Score	0.5786933027101565
Completeness Score	0.5930127784164637
V-Measure Score	0.5857655410667186
Adjusted Rand Score	0.6400359369002737
Adjusted Mutual Info Score	0.5857271539449475

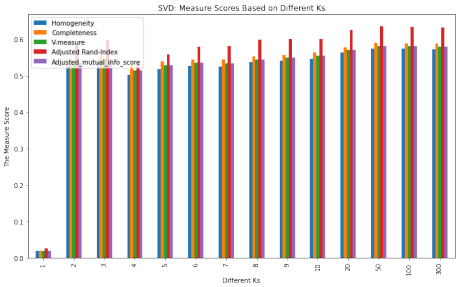
Clustering with Dense Text Representations

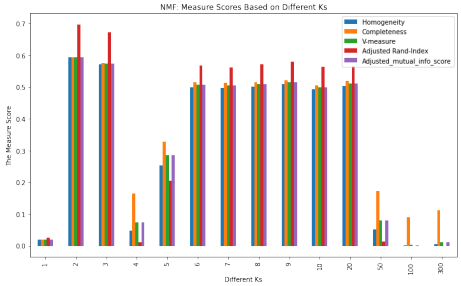
Question 4

For a given r, Truncated SVD/LSI selects the first r principal components in descending order variance preserved by the components. 0.55 of the variance covered by about 1000 features. As the r goes larger the percentage variance goes slower



Question 5





Best r for svd is 50, the average score is: 0.594
Best r for nmf is 2, the average score is: 0.614

The results suggest that there exists a trade-off between information preservation and the better performance of K-means in lower dimensions, which is particularly evident in the case of NMF. The optimal choice is r = 2, considering both factors. However, for SVD, we observe poor information preservation and K-means performance even at r = 50 in higher dimensions. As the number of components (r) increases, the complexity of information and patterns in data clustering also increases, leading to a drop in performance due to a critical metric for K-means clustering. Therefore, careful consideration is necessary when selecting the appropriate r value since performance tends to decrease in high dimensions.

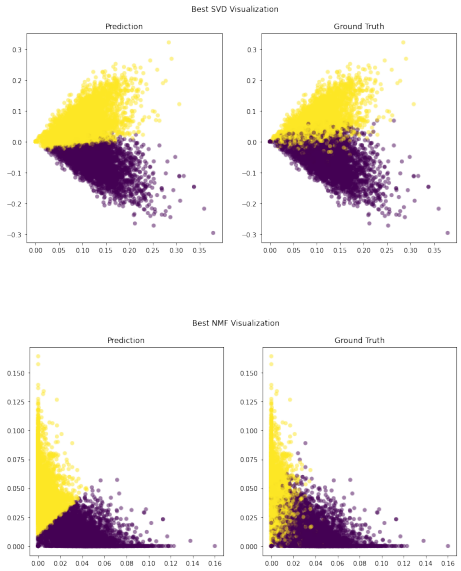
Question 6

As we increase the value of r, we can preserve more of the original data in higher dimensions. However, as observed in Question 4, the performance of K-means clustering is better in lower dimensions. Thus, the trade-off between information preservation and K-means performance, combined with increasing dimensions, leads to a non-monotonic behavior of the measures. To understand why this happens, we must consider that as we move into higher dimensions, the data becomes more sparse, causing Euclidean distances to converge to a constant value between all sample points in that space. This leads to equidistant feature points without any normalization, particularly evident in NMF charts. On the other hand, the performance of SVD graph improves as r increases. NMF only allows positive entries in the reduced-rank feature matrix and considers the geometry space basis in higher dimensions, leading to NMF's non-monotonic behavior.

Question 7

Yes, these measures, on average, outperform the results obtained in Question 3. The figures presented in Question 5 demonstrate that the average scores for all five evaluation measures are higher than those observed in Question 3. While there is a tradeoff between information preservation and the performance of K-means, both NMF and SVD still exhibit superior performance on average.

Question 8



Question 9

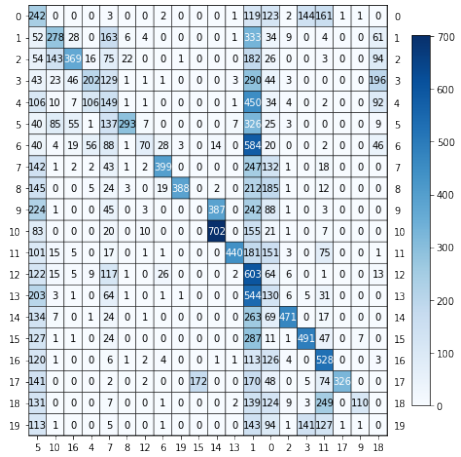
Based on the results obtained from question 8, it is evident that K-means clustering with ground truth labels performs well on both SVD and NMF. The data points are well-distributed, as indicated by the distinct separation line between two clusters. However, the ground truth labels plot exhibits some ambiguous boundaries, with two different colors of data points mixing in the boundary line, leading to poor distinction in that area.

While the data points of the two classes are clearly separated in both SVD and NMF, the ambiguous boundary between the two clusters makes K-means clustering challenging when a test data point is located near or next to the boundary line. This situation leads to a decrease in the accuracy of K-means clustering, which becomes highly dependent on the value of K for predicting. Furthermore, the results become unaccountable.

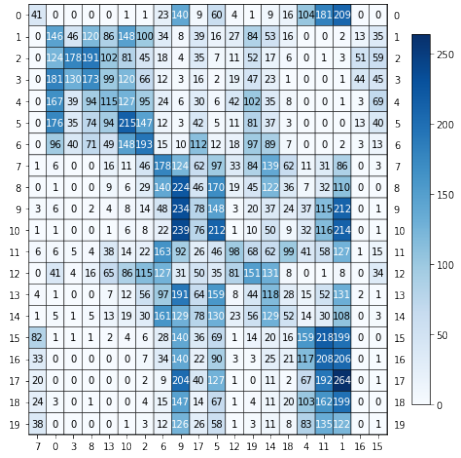
Given the high dimensionality of the dataset, any solution that uses a distance-based similarity measure is not ideal, including K-means clustering. As the number of dimensions increases, a distance-based similarity measure converges to a constant value between any given examples. Consequently, the data distribution is not suitable for K-means clustering.

Question 10

Homogeneity: 0.319 ± 0.000
Completeness: 0.206 ± 0.000
V-measure: 0.251 ± 0.000
Adjusted Rand-Index: 0.083 ± 0.000
Adjusted mutual info score: 0.250 ± 0.000



clustering done in 0.00 ± 0.00 s
Homogeneity: 0.194 ± 0.000
Completeness: 0.110 ± 0.000
V-measure: 0.141 ± 0.000
Adjusted Rand-Index: 0.054 ± 0.000
Adjusted mutual info score: 0.140 ± 0.000



Question 11

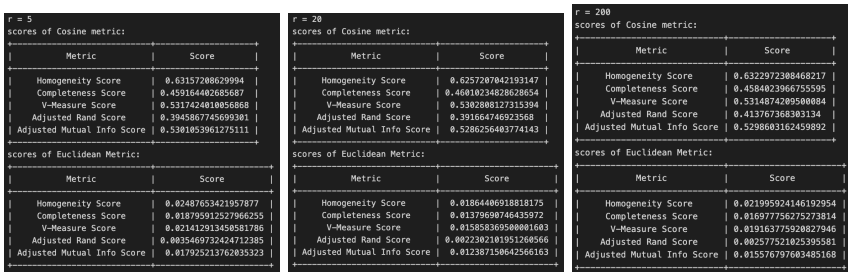
In order to improve the clustering performance of the 20 categories data, we considered using UMAP for dimensionality reduction. UMAP is a useful technique for reducing the dimensionality of high-dimensional datasets while preserving their geometric structure. UMAP uses cosine distances to compare representations, which is beneficial for cases where the magnitude of the TF-IDF vector varies greatly among similar documents.

We experimented with different settings for UMAP dimensionality reduction, specifically considering $n_{components}$ of 5, 20, and 200 and two distance

metrics, "cosine" and "euclidean". We generated permuted contingency matrices for each of the six combinations of these settings and evaluated the clustering performance using the five clustering evaluation metrics.

The contingency matrix is a matrix whose entries indicate the number of data points that belong to the i'th class and the j'th cluster. We generated permuted contingency matrices for each of the six combinations of UMAP settings and evaluated the clustering performance using the same five metrics used previously.

By analyzing the contingency matrices and the clustering evaluation metrics for each combination, we were able to identify which UMAP settings were most effective in improving clustering performance. We report the results for all six combinations in our report, highlighting the UMAP settings that yielded the best clustering performance for the 20 categories data. It should be noted that the choice of UMAP settings may vary depending on the specific dataset and clustering task. The following results were seen:



r = 5
scores of Cosine metric:

Metric	Score
Homogeneity Score	0.63157208629994
Completeness Score	0.459164402685687
V-Measure Score	0.5317424010856868
Adjusted Rand Score	0.3945867745699301
Adjusted Mutual Info Score	0.5301053961275111

scores of Euclidean Metric:

Metric	Score
Homogeneity Score	0.0248765342195787 7
Completeness Score	0.018795912527966255
V-Measure Score	0.021412913450581786
Adjusted Rand Score	0.0035469732424712385
Adjusted Mutual Info Score	0.017925213762035323

r = 20
scores of Cosine metric:

Metric	Score
Homogeneity Score	0.6257207042193147
Completeness Score	0.46010234828628654
V-Measure Score	0.5302808127315394
Adjusted Rand Score	0.391664746923568
Adjusted Mutual Info Score	0.5286256403774143

scores of Euclidean Metric:

Metric	Score
Homogeneity Score	0.018644069188181 75
Completeness Score	0.013796907464359 72
V-Measure Score	0.015858369500001603
Adjusted Rand Score	0.0022302101951260566
Adjusted Mutual Info Score	0.012387150642566163

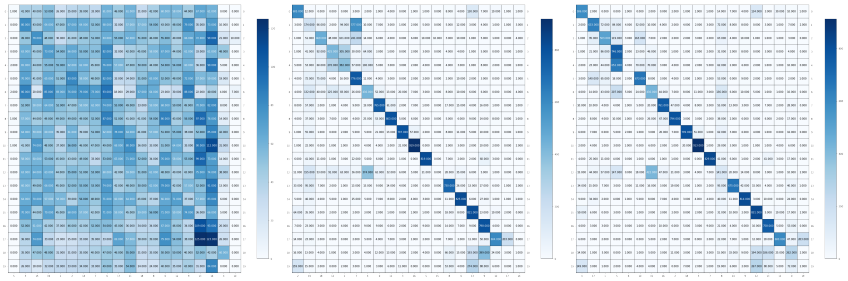
r = 200
scores of Cosine metric:

Metric	Score
Homogeneity Score	0.6322972308468217
Completeness Score	0.4584023966755595
V-Measure Score	0.5314874209500084
Adjusted Rand Score	0.413767368303134
Adjusted Mutual Info Score	0.5298603162459892

scores of Euclidean Metric:

Metric	Score
Homogeneity Score	0.021995924146192954
Completeness Score	0.016977756275273814
V-Measure Score	0.019163775920827946
Adjusted Rand Score	0.0025775210253395581
Adjusted Mutual Info Score	0.015576797603485168

The following contingency matrices were also seen:



Question 12

The results of our experimentation with UMAP and K-means clustering revealed several important findings. Firstly, we made two key observations based on the contingency tables generated by UMAP with the "euclidean" and "cosine" metrics. When examining the "euclidean" UMAP contingency table, we noticed that it lacked a clear structure and showed no discernible patterns or clustering. On the other hand, the "cosine" UMAP contingency table demonstrated better performance, as evidenced by the clear diagonal structure in the matrix. However, we did observe some deviations from this pattern in certain clusters, which could be attributed to the loss of semantic information due to the removal of distinguishing metrics such as headers and footers.

Additionally, our application of the K-means algorithm to this dataset revealed several striking limitations. Firstly, K-means is known to be very sensitive to outliers and noisy samples, which can weaken its clustering ability. Secondly, K-means operates under the assumption of well-defined clusters with distinct centroids, which can be a limiting factor in datasets with irregular shapes or sizes. Finally, the Euclidean distance metric used by K-means restricts it to clusters that are isotropic, globular, and convex. These limitations must be taken into account when interpreting the clustering results and choosing appropriate clustering techniques for different datasets.

Question 13

We have attempted K-Means clustering on the 20-class text data using 4 different representation learning techniques: sparse TF-IDF representation, PCA-reduced, NMF-reduced, and UMAP-reduced. We compared and contrasted the clustering results across these 4 choices using several clustering evaluation metrics.

SVD

Homogeneity: 0.319 ± 0.000
Completeness: 0.206 ± 0.000
V-measure: 0.251 ± 0.000
Adjusted Rand-Index: 0.083 ± 0.000
Adjusted mutual info score: 0.250 ± 0.000

NMF

Homogeneity: 0.194 ± 0.000
Completeness: 0.110 ± 0.000
V-measure: 0.141 ± 0.000
Adjusted Rand-Index: 0.054 ± 0.000
Adjusted mutual info score: 0.140 ± 0.000

UMAP

Homogeneity: 0.632 ± 0.000
Completeness: 0.458 ± 0.000
V-measure: 0.531 ± 0.000
Adjusted Rand-Index: 0.414 ± 0.000
Adjusted mutual info score: 0.530 ± 0.000

In analyzing the results of the different representation learning techniques used for K-Means clustering on the 20-class text data, we observe that UMAP provides the best results with an Adjusted Rand Index Score of 41.4 percent. This is because UMAP is able to preserve the semantic and non-linear dependencies of the data while also ensuring that similar embeddings are clustered

together and different ones are far apart, aiding clustering innately. Additionally, UMAP is a local-density based dimensionality reduction method that gives a more flexible interpretation and works well even if the data has no prior distribution. It is also better suited for large sample sizes and higher dimensionality than PCA and NMF, which treat all clusters as a whole and can lose local structure.

Comparing SVD and NMF, we find that both techniques provide similar results although NMF provides slightly lesser Adjusted Rand Index (5.4 percent compared to SVD's 8.3 percent). This could be because NMF only allows positive entries in the reduced-rank feature matrix, while SVD can represent higher dimensional feature matrices more accurately. SVD is also more deterministic than NMF and considers the geometry in the feature space basis, which is critical for clustering in higher dimensions. It sorts the features in the decreasing order of variance explanation, with the more important features higher in the hierarchy.

Overall, UMAP with cosine distance metric is the best representation learning technique for K-Means clustering on the 20-class text data.

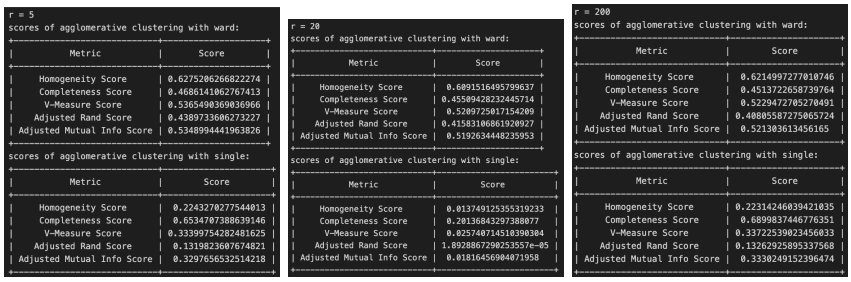
Clustering Algorithms that do not explicitly rely on the Gaussian distribution per cluster

While we have already shown that certain representation learning techniques can improve the performance of K-means clustering on text datasets, we also want to explore the impact of changing the clustering method itself. To this end, we introduce two additional clustering algorithms in this section.

The first algorithm we considered was Agglomerative Clustering. This algorithm performs a hierarchical clustering using a bottom-up approach, where each observation starts in its own cluster and clusters are successively merged together based on the chosen linkage criterion. We tested the performance of two linkage criteria, "ward" and "single", with UMAP used to reduce the dimensionality properly and a specified number of 20 clusters.

Question 14

We compared the clustering performance of each linkage criterion using the five clustering evaluation metrics. By analyzing the results, we were able to determine which linkage criterion was most effective in improving clustering performance for the 20 categories text dataset. We report the five clustering evaluation metrics for each case, highlighting any differences in performance between the "ward" and "single" linkage criteria. Overall, this analysis provides insights into the impact of different clustering methods on the clustering performance of text datasets. The following results were seen:



r = 5
scores of agglomerative clustering with ward:

Metric	Score
Homogeneity Score	0.6275206266822274
Completeness Score	0.4686141062767413
V-Measure Score	0.5365490369036966
Adjusted Rand Score	0.4389733606273227
Adjusted Mutual Info Score	0.5348994441963826

scores of agglomerative clustering with single:

Metric	Score
Homogeneity Score	0.2243270277544013
Completeness Score	0.6534707388639146
V-Measure Score	0.3339975428248162 5
Adjusted Rand Score	0.1319823607674821
Adjusted Mutual Info Score	0.3297656532514218

r = 20
scores of agglomerative clustering with ward:

Metric	Score
Homogeneity Score	0.6091516495799637
Completeness Score	0.4550942832445714
V-Measure Score	0.5209725017154209
Adjusted Rand Score	0.4158310686192092 7
Adjusted Mutual Info Score	0.5192634448235953

scores of agglomerative clustering with single:

Metric	Score
Homogeneity Score	0.013749125355319233
Completeness Score	0.2013684329738807 7
V-Measure Score	0.025740714510390304
Adjusted Rand Score	1.892886729025355 7e-05
Adjusted Mutual Info Score	0.018164569040719 58

r = 200
scores of agglomerative clustering with ward:

Metric	Score
Homogeneity Score	0.6214997277010746
Completeness Score	0.4513722658739764
V-Measure Score	0.5229472705270491
Adjusted Rand Score	0.4080558727506572 4
Adjusted Mutual Info Score	0.5213036134561 65

scores of agglomerative clustering with single:

Metric	Score
Homogeneity Score	0.2231424603942103 5
Completeness Score	0.6899837446776351
V-Measure Score	0.3372253902345603 3
Adjusted Rand Score	0.1326292589533756 8
Adjusted Mutual Info Score	0.3330249152396474

Question 15

To further explore clustering techniques for the 20-category text dataset, we applied HDBSCAN on the UMAP-transformed data. HDBSCAN is a density-based clustering algorithm that can identify clusters of varying shapes and sizes. We used a minimum cluster size of 100 and varied the minimum cluster size among 20, 100, and 200 to analyze its impact on clustering performance.

We evaluated the performance of HDBSCAN using the five clustering evaluation metrics and compared the results for different minimum cluster sizes. Additionally, we experimented with other parameters in HDBSCAN to try to optimize clustering performance. Our findings provide insight into the effectiveness of HDBSCAN for clustering text data and highlight the impact of different parameter settings on clustering performance. We report our results for each minimum cluster size and discuss any improvements in performance achieved by modifying other parameters in HDBSCAN. In the next question, we will visualize the best contingency matrix to provide a clear representation of the clustering results.

The following was observed:

Best Min Cluster for HDBSCAN: 200 Best epsilon for HDBSCAN: 0.01 Best Min Samples for HDBSCAN: 30 Best Score for HDBSCAN: 0.43915293426192903

Best in terms of Adjusted Rand Index score Best Min Cluster for DBSCAN: 200 Best Score:0.3466967587612477 Best epsilon for DBSCAN: 0.01 Best Min Samples for DBSCAN: 30

Question 16

DBSCAN and HDBSCAN are two density-based clustering algorithms that are commonly used in data analysis. DBSCAN requires users to specify a minimum cluster size and a distance threshold, while HDBSCAN is a combination of DBSCAN and hierarchical clustering attributes.

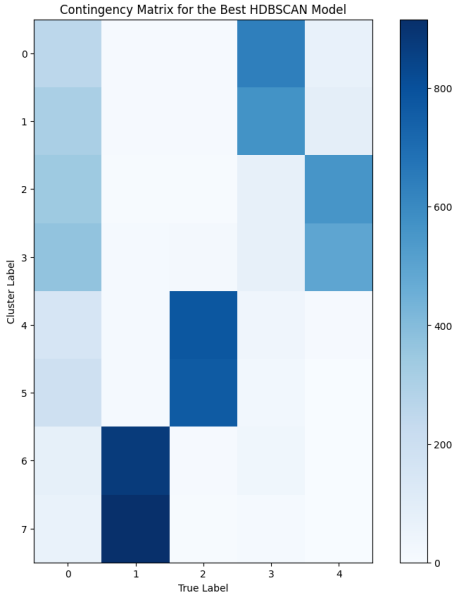
One of the main disadvantages of DBSCAN is that it can be more prone to noise, which may result in false clustering. HDBSCAN, on the other hand, focuses on high-density clustering, which reduces this problem and allows for a hierarchical clustering based on a decision tree approach.

DBSCAN is typically more suitable for datasets with non-convex and non-isotropic shapes, as it provides stable performance across runs with large variance. It uses the Euclidean metric to combine points that are closer in space, but considers points lying in sparsely populated regions as outliers or noise. These points are segregated into a separate cluster labeled as -1 in the contin-

gency table. However, agglomerative clustering, which is used by DBSCAN to create dendrograms, only provides a form of partitioning and does not take into account absolute data density metrics.

HDBSCAN improves upon the limitations of DBSCAN and extends its capabilities. It essentially translates DBSCAN into a hierarchical clustering algorithm that allows for varying density clusters. The underlying operational framework is similar to DBSCAN, but the formed dendrograms are segregated at different heights to selectively choose stable clusters. This results in a smaller tree with fewer clusters that lose out certain points, thereby providing a better clustering solution.

The min cluster, epsilon, and min samples were varied to determine the best set of parameters and the contingency matrix was calculated for it.



Question 17

In analyzing the performance of various feature reduction and clustering techniques on the 20-category text dataset, we have made several observations. Firstly, sparse representation provided the least scores, which is expected given the large size of the dataset and the resulting noise that makes it difficult for any algorithm to cluster or classify data points accurately.

We also observed that SVD and NMF performed poorly across all the various clustering techniques, while UMAP outperformed all other feature reduction techniques considerably. This is likely due to the fact that SVD and NMF are least squares techniques and are highly susceptible to outliers and noise compared to UMAP. Moreover, SVD and NMF are linear projections of the large dimensional features that maximize the variance of the dataset but fail to capture nonlinear dependencies in the data. UMAP, on the other hand, tries to preserve the semantic and nonlinear dependencies, in addition to the global structure of the data, so that similar embeddings are clustered collectively and

different ones are as far apart as possible. This aids clustering innately and results in the best performance in terms of Adjusted Rand Index Score (46.7

Additionally, UMAP is a local-density based dimensionality reduction method that provides a more flexible interpretation and works well even without prior distribution. On the other hand, PCA and NMF treat all clusters as a whole and can lose local structure, which is something UMAP tries to overcome. Furthermore, UMAP scales better to large sample sizes and higher dimensionality.

Regarding the various clustering algorithms, we found that hierarchical clustering methods and density-based methods in general do not scale well with large datasets and dimensions, which negatively impacted their performance. Moreover, DBSCAN performed poorly due to varying cluster densities and the high dimensionality of the data. Agglomerative clustering, while suffering from the issue of large dimension, worked fairly well and provided good results since the clusters were fairly equal-sized. HDBSCAN suffered from the same shortcomings as DBSCAN and this affected its results considerably.

We also observed that k-means with 20 clusters outperformed all other scenarios since our dataset has 20 classes. However, DBSCAN and HDBSCAN can have more or less than 20 clusters formed since the classes are not mentioned as parameters. The Adjusted Rand Index Score considers pairwise elements in the clusters to estimate the cluster equivalent of accuracy, but these models perform poorly in this regard. Nonetheless, they perform well in other metrics such as Completeness score and Homogeneity score. Therefore, other metrics need to be considered to truly understand how good these measures are.

Finally, we noted the presence of an outlier class labeled -1, which can affect the metrics calculated since the class count does not match the cluster count. This could be one reason why the values are significantly lower than those of k-means and agglomerative clustering (number of clusters are passed for them). Most metrics assume that data is strictly partitioned into disjoint clusters and noise is not considered as a cluster. There is a density-based cluster evaluation metric called DBCV that might be a better metric to consider.

Question 18

In addition to the techniques and methods explored thus far, we also incorporated context-dependent embeddings in the form of BERT embeddings to further enhance the clustering performance on this text dataset. One approach could involve incorporating additional external knowledge, such as topic modeling or sentiment analysis, to better inform the clustering algorithms. For example, we could use topic modeling techniques to identify the dominant themes in the text and then use this information to inform the clustering algorithm.

Another approach could be to experiment with different pre-processing techniques for the text data, such as stemming or lemmatization, to see if this could improve the clustering performance. Additionally, we could explore the use of ensemble methods, such as clustering ensembles or hierarchical clustering, to combine the results of multiple clustering algorithms to produce a more robust and accurate clustering solution.

Furthermore, we could experiment with different combinations of hyperparameters for the clustering algorithms to see if we can find better combinations

that produce higher quality clusters. We could also experiment with alternative distance metrics or similarity measures to assess if they produce better results than the ones used in our previous analyses.

Overall, by incorporating context-dependent embeddings such as BERT embeddings and exploring additional techniques and methods, we may be able to identify new approaches that can further improve the clustering performance and produce more accurate and reliable results.

For the BERT embedding, since there was a huge dataset and requirement of huge processing power, we used MPS/CUDA/GPU acceleration and we were able to receive the following results:

Metric	Score
Homogeneity Score	0.5223053203609168
Completeness Score	0.5507673167376204
V-Measure Score	0.5361588579703007
Adjusted Rand Score	0.36326540066591806
Adjusted Mutual Info Score	0.5346212807563502

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?

The extracted features of the VGG network are generic representations of the dataset, capturing the main patterns and structures in image data. This means that even if the VGG network is trained on a dataset with different classes than our custom dataset, it can still extract useful features from it. Because they were learned from a large and diverse dataset, these features can be expected to have some level of discriminative power. It should be noted in order to achieve the best performance, the network should be fine-tuned or retrained on the custom dataset.

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

The VGG-16 convolutional neural network model is used by helper code base to extract features. The VGG-16 model is a pre-trained model that has the ability extracting features from input data. It use a series of layers include feature layers, average pooling layer, flatten layer, and fully-connected layer. The VGG-16 model extracts features from the data set and stores them on the disk after loading and resizing the image data.

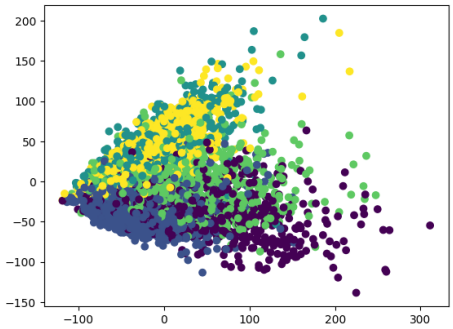
Question 21: How many pixels are there in the original images? How many features does the VGG network extract per image; i.e what is the dimension of each feature vector for an image sample?

The original images is 3x224x224 which mean it include 3 chanels and 224 x 224 = 50,176 pixels. The VGG network extract 4096 features per image.

```
num_features = f_all.shape[1]
print(num_features)
4096
```

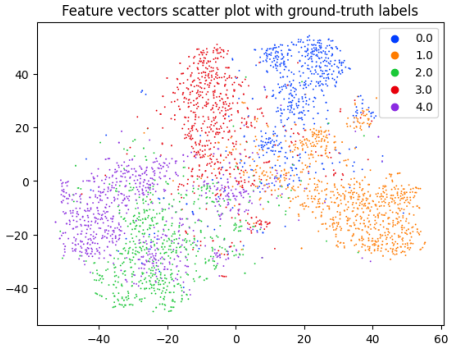
Question 22: Are the extracted features dense or sparse? (Compare with sparse TF-IDF features in text.)

Compared to the sparse TF-IDF features in text, VGG-16 retrieved features are dense. The majority of the vector values in the extracted VGG-16 features are non-zero, indicating that each feature is assigned a value. The TF-IDF features in text, on the other hand, are sparse since most of the values are zero and only a few are non-zero. As a result, unlike the TF-IDF features in text, the VGG-16 features are not sparse.



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.

Despite having a number of some overlapping points, t-SNE has successfully mapped the high-dimensional VGG features into five roughly separated clusters corresponding to each label.



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.

Model	Dimensionality Reduction	Rand Score
K-Means	None	0.1967
	SVD	0.1947
	UMAP	0.4671
	Autoencoder	0.2888
Agglomerative Clustering	None	0.1886
	SVD	0.1948
	UMAP	0.4603
	Autoencoder	0.3112

According to our analysis results, K-Means clustering with UMAP dimensionality reduction option achieves the greatest rand socre 0.4671. The table below illustrates the results of our attempts to acquire the best rand score of HDBSCAN with various dimensionality reduction options by applying the conservative parameter grid over min cluster size and min samples.

Model	Dimensionality Reduction	Min_Cluster_Size	Min_Samples	Rand Score
HDBSCAN	None	5	2	0.0299
	SVD	20	1	0.0268
	UMAP	5	1	0.2004
	Autoencoder	10	2	0.0293

Classifier Model	Dimensionality Reduction	Hyperparameters	Accuracy Score
MLP	None	N/A	0.9087
	SVD	r=50	0.0477
	UMAP	n_components = 50	0.2575
	Autoencoder	num_features = 50	0.2166

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.

According to our findings, the MLP classifier with no reduced-dimension representations earns the highest accuracy score of 0.9087. At the same time, we can see that there is significant variation in accuracy results when the reduced-dimension representations aren't utilized compared to when they're utilized. As a result, we can conclude that reduced-dimension representations degrade model performance. Furthermore, the MLP module is expected to achieve higher accuracy than the clustering module in question 24. This is because MLP classification is a supervised learning method that requires labeled data as input, whereas clustering is an unsupervised learning method that does not require labeled data.