Deniz Orkun Eren – UID: 905624625
Matthew Waliman – UID: 605848839

ECE 219 Project 2 Report

In this project, we are tasked with utilizing different unsupervised clustering methods to cluster the data points in the 20 Newsgroups dataset.

The first step is to get an easily separable portion of the 20 Newsgroups dataset. To do so, we used the following subgroups in the dataset:

Class 1	comp.graphics	comp.os.ms-windows.misc	comp.sys.ibm.pc.hardware	comp.sys.mac.hardware
Class 2	rec.autos	rec.motorcycles	rec.sport.baseball	rec.sport.hockey

After that, we went through the steps necessary to generate the TF-IDF matrix as was done for the previous project. However, this time we did not use any stemming or lemmatization to generate our vocabulary.

QUESTION 1

The dimension of the generated TF-IDF set was **7882 to 19198**. So, it was a dataset with **7882 rows** and **19198 columns**.

After generating the set, we applied K-means clustering with k=2 on this data. The goal of K-means clustering is to minimize the sum of the squares of the distances between each data point and the center of the cluster it belongs to to assign each data point to exactly 1 cluster. As we are setting the k value to 2, each data point will be assigned to one of 2 clusters.

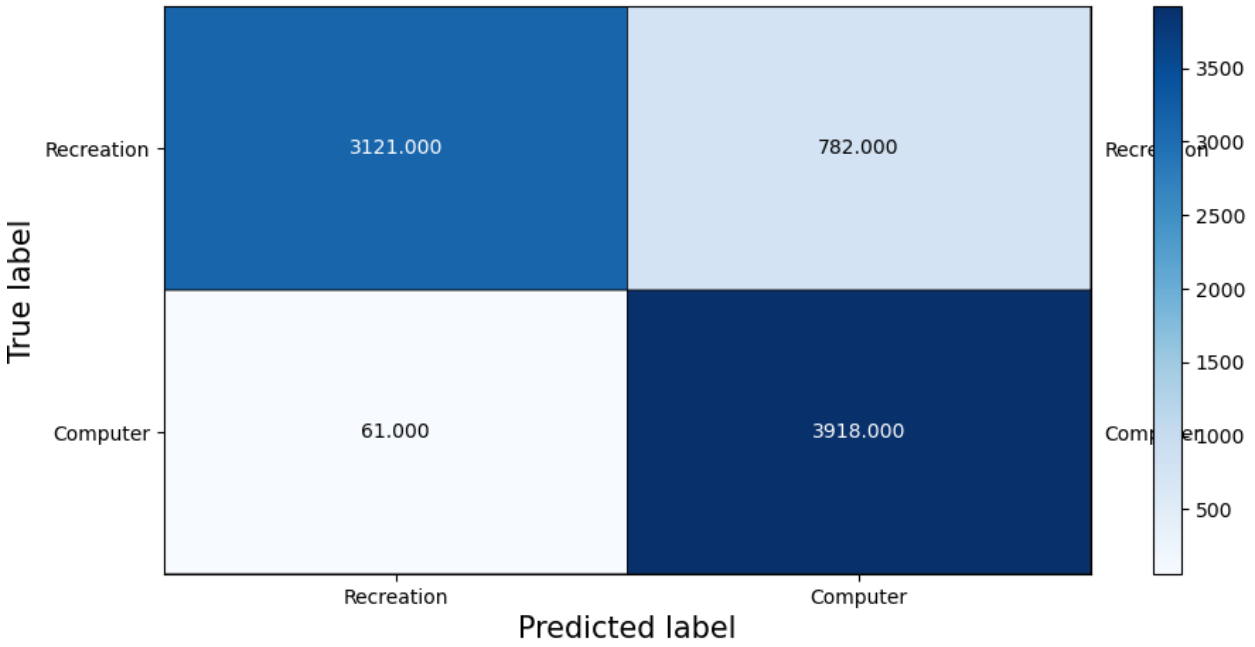
In our implementation of the K-means clustering, we set the ndom_state to 0, the max_iter parameter to 1500 and the n_init parameter to 35.

There are several metrics to evaluate the performance of a clustering algorithm. The contingency matrix is used to visualize the intersection cardinality for every true/predicted cluster pair. Homogeneity is used to determine how pure each formed cluster is. If the points in a cluster come from different classes then that cluster is not homogenous. Completeness measures if all the data points that are members of a given class are elements of the same cluster. The adjusted rand index is used to compare the similarity between clustering and true labels. Finally, the adjusted mutual information measures the mutual information between cluster label distribution and ground truth label distribution. We are gonna make use of these scores to evaluate our clustering techniques.

Deniz Orkun Eren – UID: 905624625
Matthew Waliman – UID: 605848839

QUESTION 2

After training our K-means model, the resulting contingency matrix is as follows:



As can be seen, the performance of our model is pretty satisfactory.

Although this contingency matrix is square shaped, this does not necessarily always have to be the case. For example, if there were three types of labels in our ground truth and we run K-means clustering with k=2 on this set, the resulting contingency matrix would be rectangular shaped.

QUESTION 3

The clustering measure scores for the K-means model are as follows:

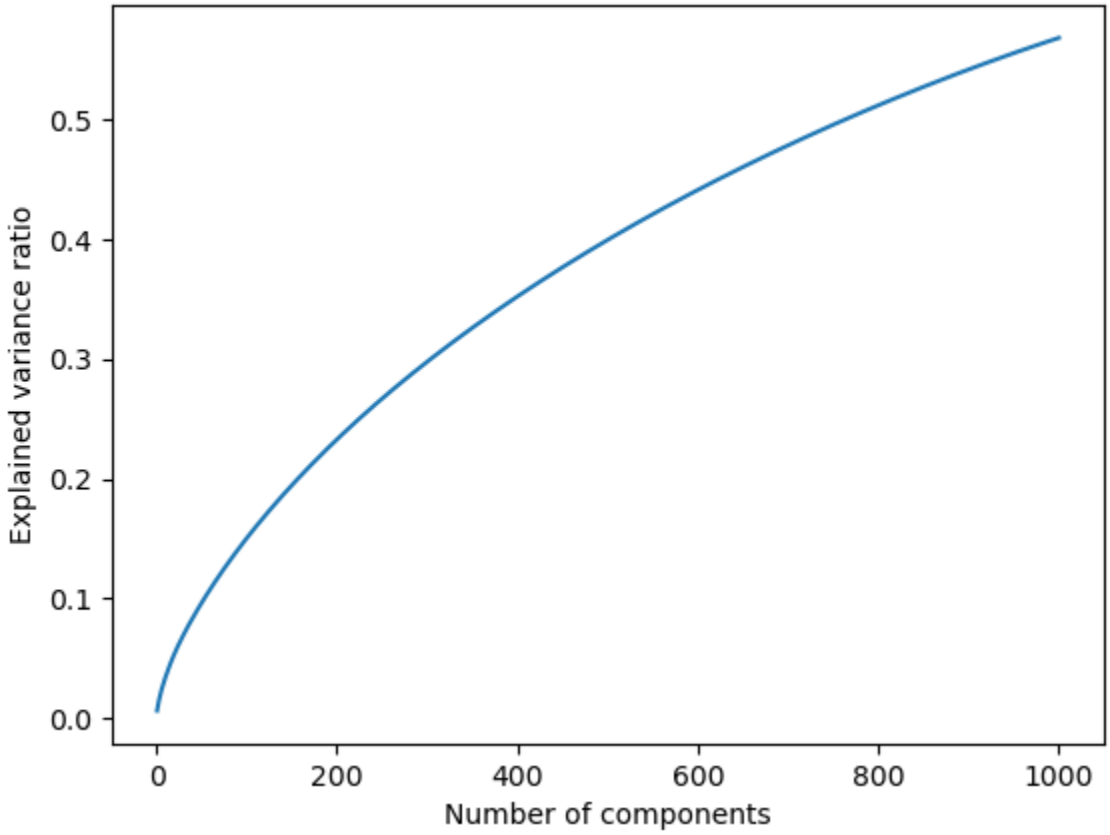
	Homogeneity	Completeness	V-measure	Adjusted rand	Adjusted Mutual Info
Score	0.557	0.572	0.565	0.617	0.565

Although these scores are good, there is room for improvement if we modify the TF-IDF dataset. To do so, we can use dimensionality reduction with truncated SVD.

Deniz Orkun Eren – UID: 905624625
Matthew Waliman – UID: 605848839

QUESTION 4

The plot of the change in the explained variance ratio with respect to the amount of principal components is as follows:



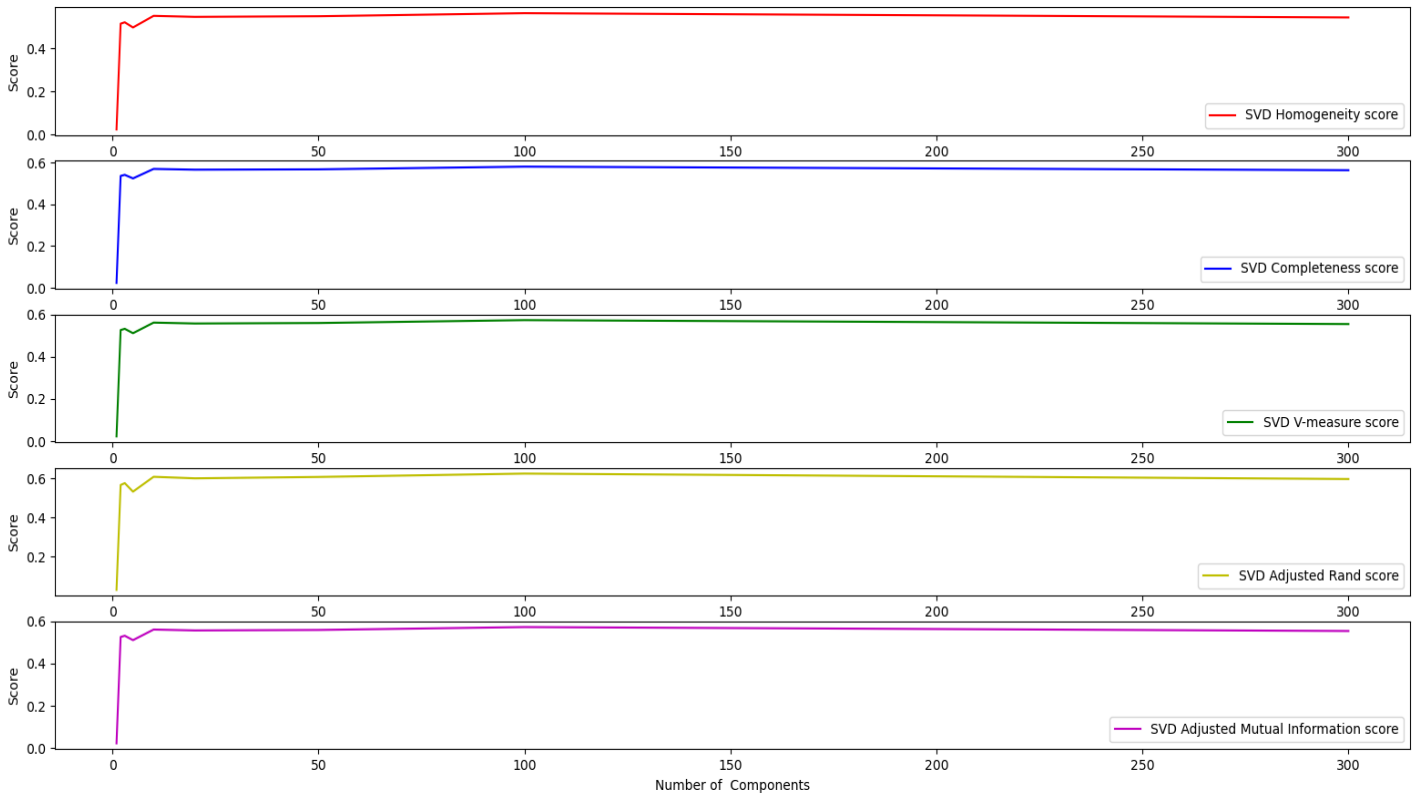
As observed in the previous project, the explained variance ratio and the number of components have a concave relationship. After a while, the effect of adding another component does not seem to improve the explained variance that much.

QUESTION 5

Let us now see the change in the clustering metrics as the dimension of the projected data changes for both truncated SVD and NMF.

For SVD, the change in the measures is as follows:

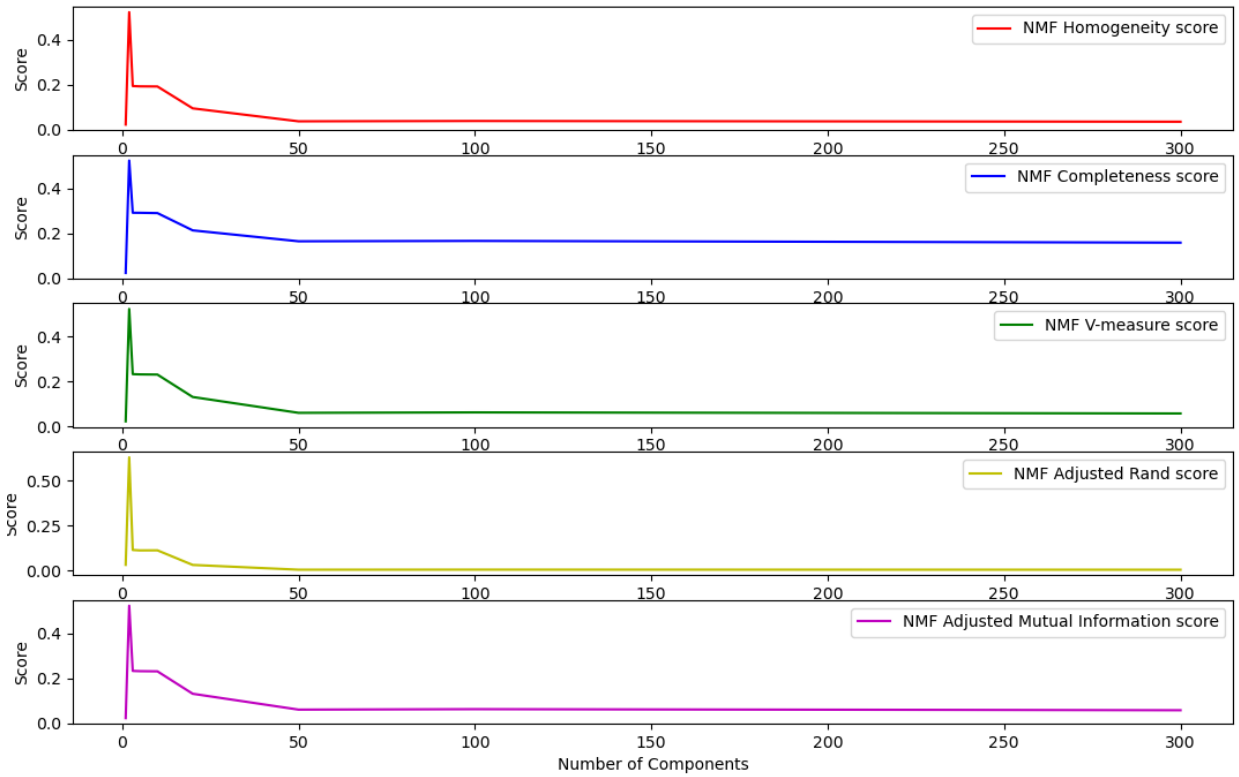
Deniz Orkun Eren – UID: 905624625
Matthew Waliman – UID: 605848839



By examining the change in V-measure, we can determine the best r value for SVD as 100.

The plot for the change in the measures related to the change in the number of dimensions for NMF is as follows:

Deniz Orkun Eren – UID: 905624625
Matthew Waliman – UID: 605848839



By examining the change in V-measure, the best number of components for NMF is found to be 2.

QUESTION 6

If we observe the behavior of the scores for both NMF and SVD, we can see that the change in the scores is not monotonic. When r is too low, there is not much information related to the dataset present in the dimensionality reduced set, which leads to poor clustering performance. As r increases, more information about the original dataset is retained, which results in an improved performance. However, we can see that when r is too high, the performance is not optimal. At too high dimensions, the euclidean distance metric used by the K-means algorithm will not be able to accurately distinguish between data points, which will lead to diminished performance. The combined effect of these points results in the non-monotonic behavior that we observe.

Although the above is a reasonable explanation for the observed behavior of the SVD approach, it does not fully explain the observed trend in NMF. For NMF, we need to consider the fact that dimensionality reduction with this technique is stochastic in nature and it is possible for the process to stop at a local minima. By observing the graph, it is possible to interpret that as the dimensionality increases, it becomes more likely for the NMF process to get stuck in a local

minima. So, there is a trade off. As the amount of dimensions increases, the information loss from the original dataset decreases. However, the NMF process can more easily get stuck in a local minima, which leads to an inaccurate projection of the dataset to lower dimensions. This can explain the non-monotonic behavior we observe.

QUESTION 7

The average scores for SVD and NMF is as follows:

	Homogeneity	Completeness	V-measure	Adjusted rand	Adjusted Mutual Info
SVD	0.48	0.49	0.48	0.527	0.488
NMF	0.14	0.23	0.17	0.11	0.17

So, on average, the performance of using K-means with SVD or NMF seems to be worse than using the model without dimensionality reduction techniques. However, let us now examine the scores for the ideal case: when r=100 for SVD and r=2 for NMF.

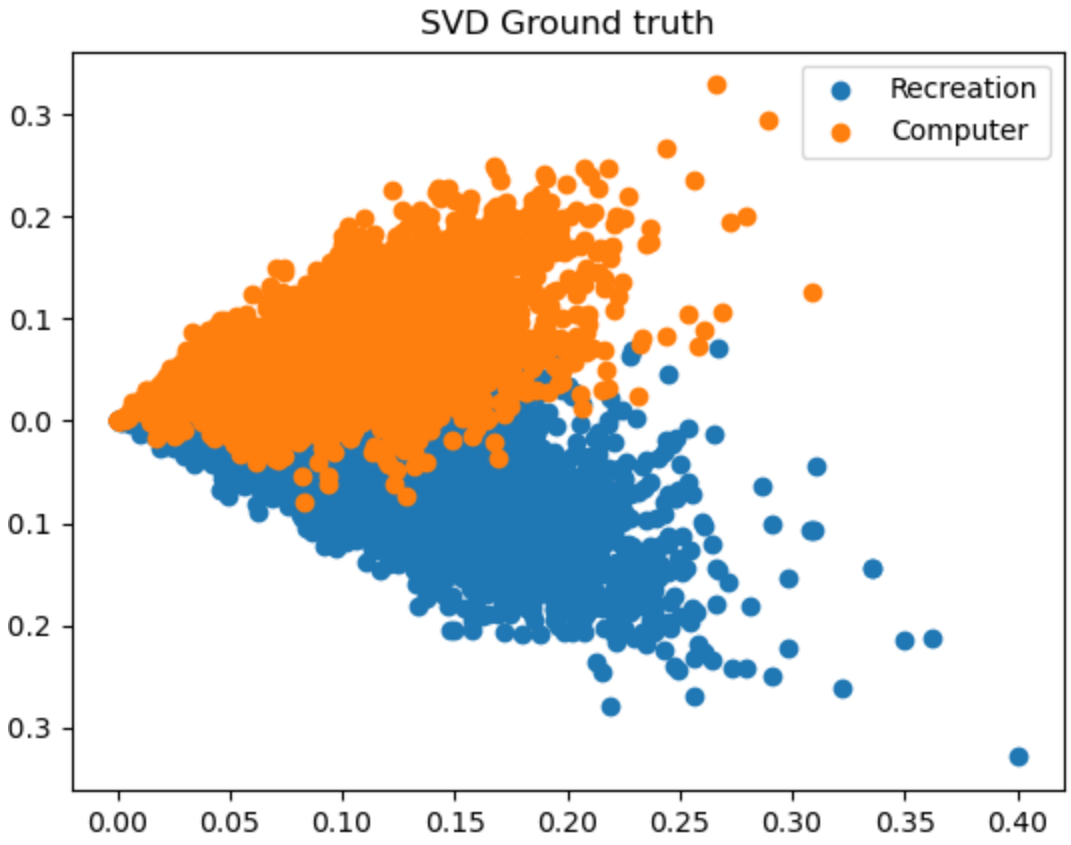
	Homogeneity	Completeness	V-measure	Adjusted rand	Adjusted Mutual Info
Ideal SVD	0.565	0.580	0.572	0.624	0.572
Ideal NMF	0.522	0.522	0.522	0.6311	0.522

As can be seen, choosing the ideal r values offers a significant boost in performance when compared to not using any dimensionality reduction.

QUESTION 8

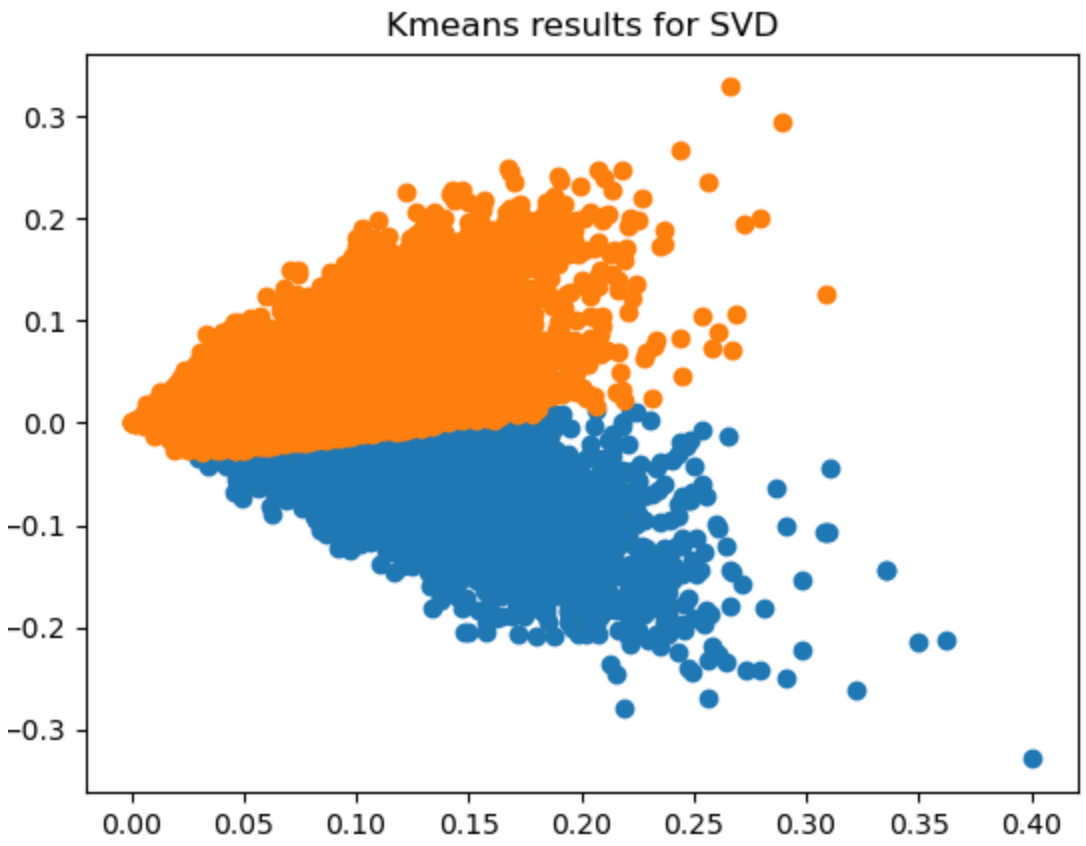
To visualize the ground truth for SVD, we project the data to two dimensions and color the projected data points based on the ground truth labels. The results are as follows:

Deniz Orkun Eren – UID: 905624625
Matthew Waliman – UID: 605848839



We then fit a truncated SVD with 10 components to the data to get our reduced set. Then, we fit a K-means model to this set to get the predictions of the system. Let us plot the data points by using the cluster labels provided by the K-means model.

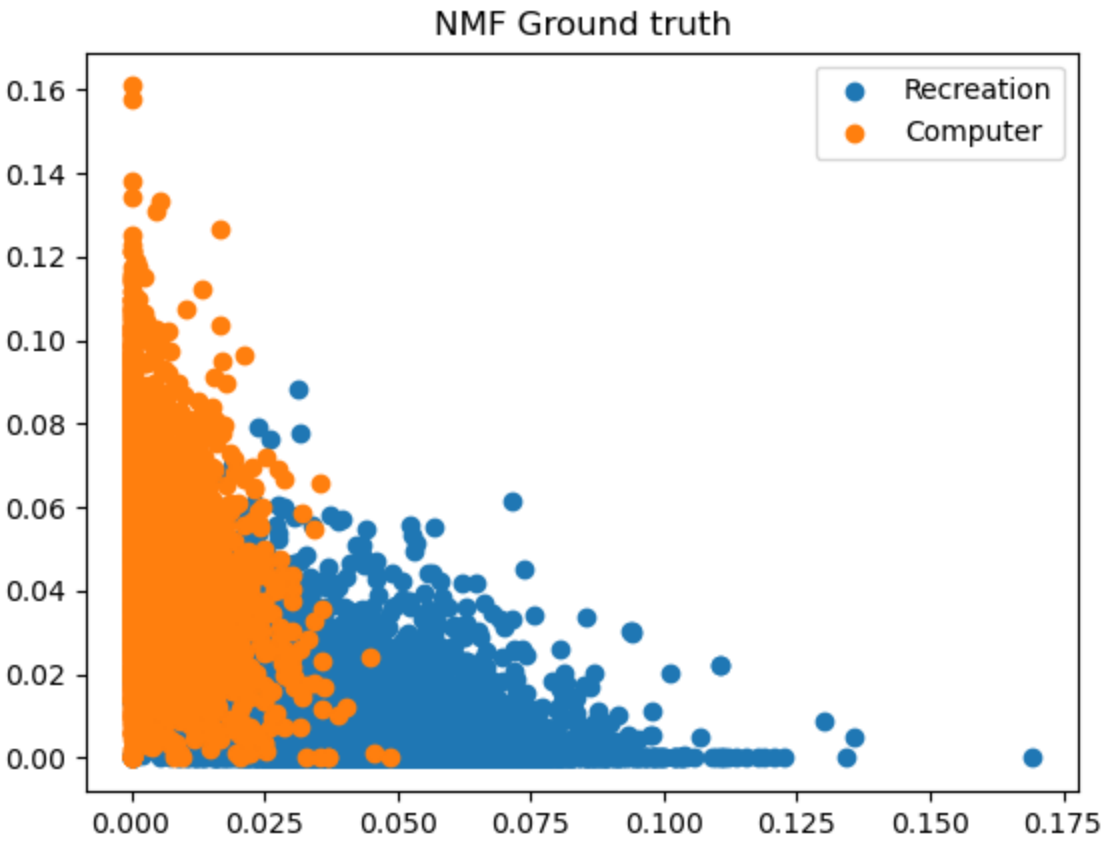
Deniz Orkun Eren – UID: 905624625
Matthew Waliman – UID: 605848839



Comparison of the two figures shows that the K-means algorithm has managed to get accurate clusters. Although some points are assigned wrongly, most of the points with different labels are assigned to different clusters.

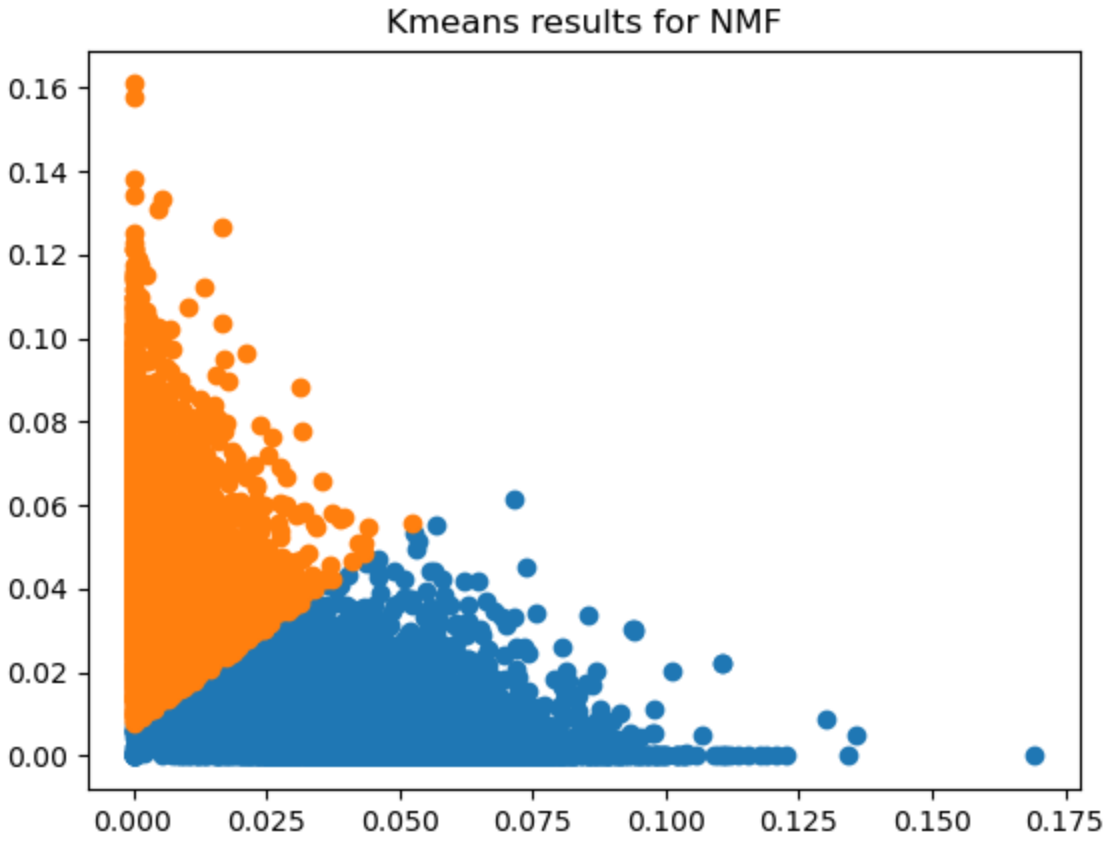
Let us now do the same process using NMF. To visualize the NMF ground truth, let us project the TF-IDF matrix to a set with 2 columns using NMF with 2 components and plot the resulting points by examining their ground truth labels.

Deniz Orkun Eren – UID: 905624625
Matthew Waliman – UID: 605848839



Since the ideal dimension for NMF is 2, we can fit the K-means model directly on the created dataset.

Deniz Orkun Eren – UID: 905624625
Matthew Waliman – UID: 605848839



QUESTION 9

Observing the ground truth figures for both SVD and NMF shows that the two clusters of the dataset are easily separable in 2 dimensions. The data points of different classes are mostly not within each other and they seem to be easily separable. This is ideal for our clustering algorithm as it makes finding good cluster centers that accurately represent the dataset easier. This observation turns out to be true as the results for K-means for both SVD and NMF show that the model is capable of accurately clustering most of the data points.

Clustering of the Entire 20 Classes

We saw that our K-means algorithm is capable of accurately clustering the data samples when specific categories are combined. Let us now examine the performance of the model when it is asked to cluster all 20 categories in the dataset.

QUESTION 10

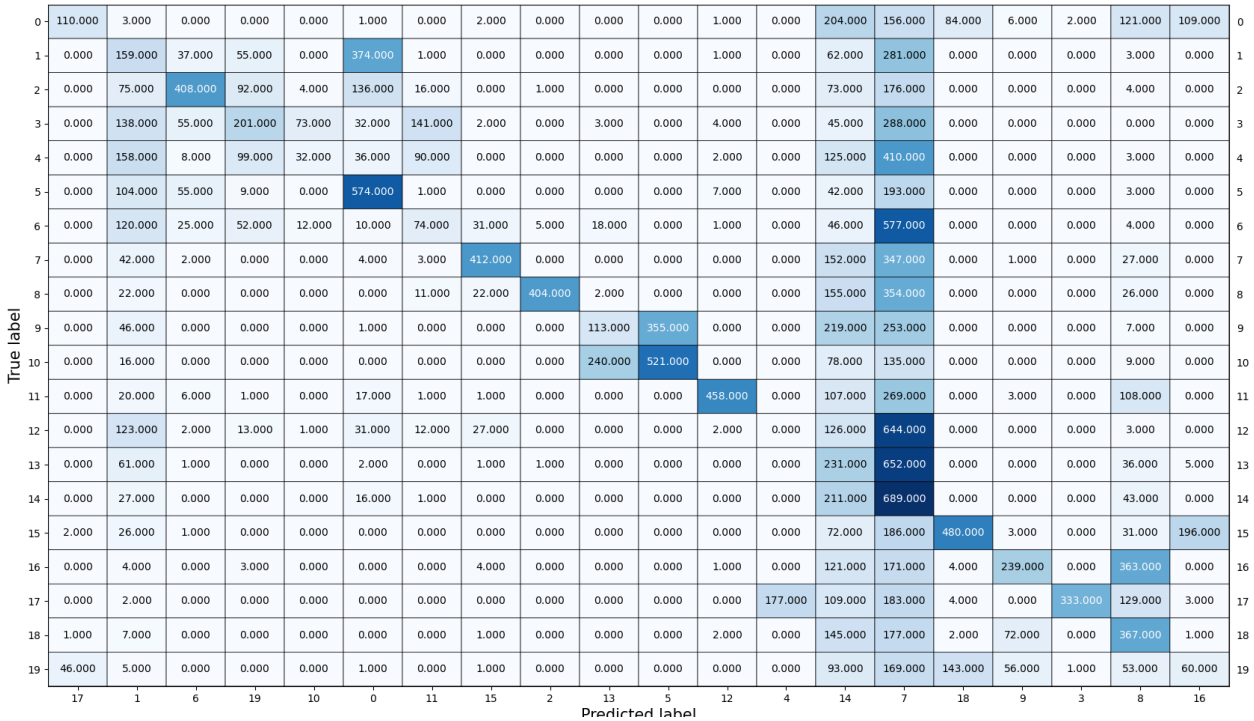
After loading the dataset and generating the TF-IDF representation as was done for the first part of this project, we had a final dataset of 18846 rows and 38627 columns.

Deniz Orkun Eren – UID: 905624625
Matthew Waliman – UID: 605848839

After generating the dataset, we experimented with different numbers of components for both truncated SVD and NMF for dimensionality reduction. In the end, it was observed that the best results were obtained using truncated SVD with 200 components.

After generating the reduced dataset, we fit the K-means model with 20 clusters to see the results. We can now evaluate the performance of our system.

Normally, we can generate the contingency matrix as done before. However in our case, some of the cluster labels are misaligned from each other. To get the right alignment, we used the provided code that uses scipy.optimize.linear_sum_assignment to identify the best-matching cluster-class pairs. The resulting contingency matrix after the alignment is as follows:



The 5 metrics for this model are as follows:

	Homogeneity	Completeness	V-measure	Adjusted rand	Adjusted Mutual Info
K-means with 20 clusters	0.329	0.408	0.364	0.09	0.362

Deniz Orkun Eren – UID: 905624625
Matthew Waliman – UID: 605848839

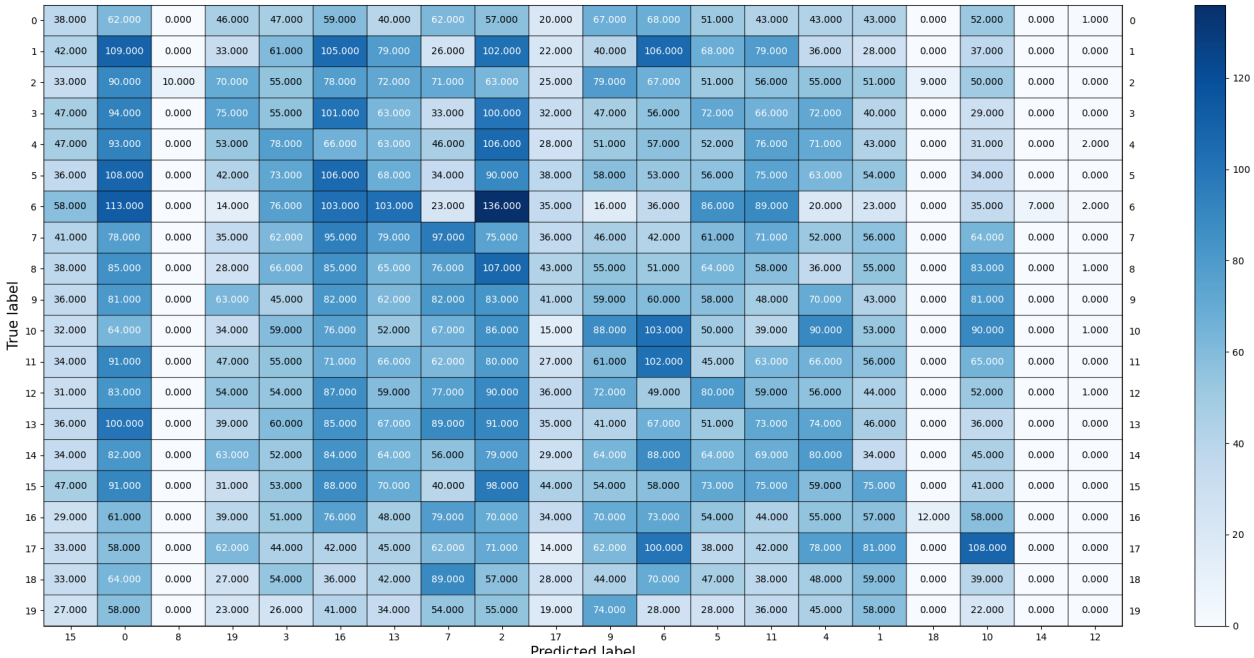
From the results, we can see that clustering with 20 labels led to less than ideal results.

UMAP

Instead of using truncated SVD or NMF, let us use another dimensionality reduction technique: UMAP and see if the results improve.

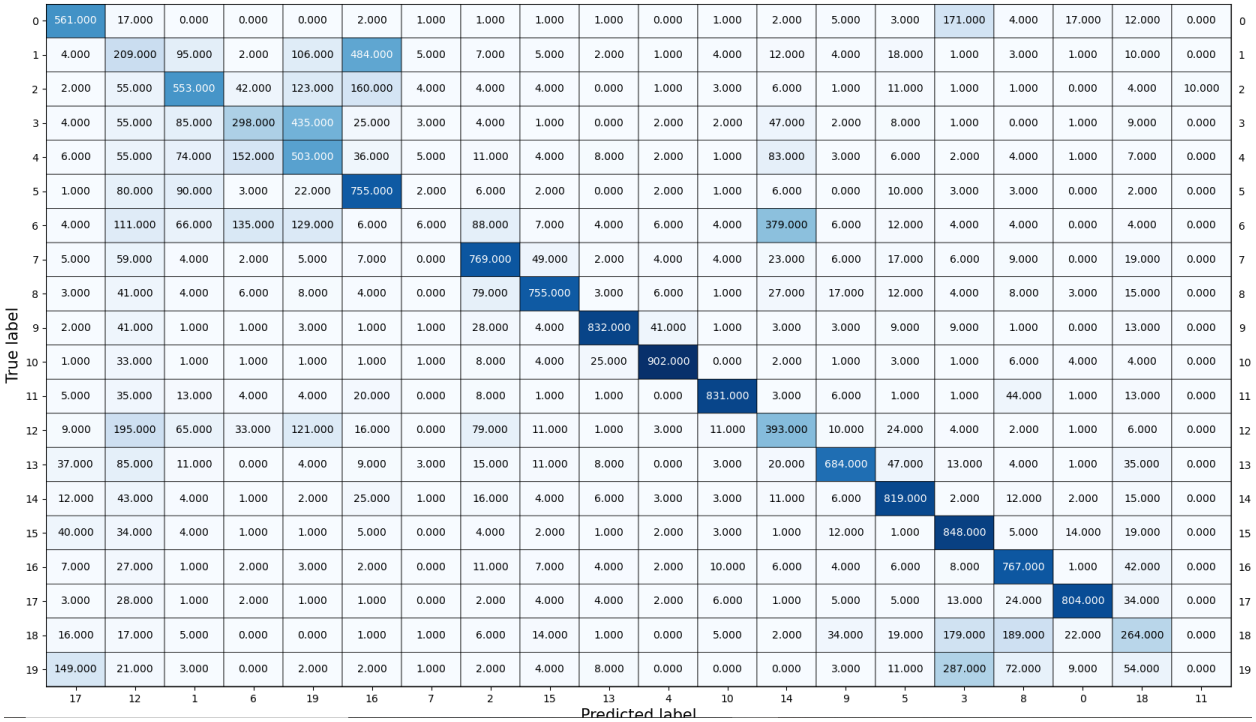
QUESTION 11

We tested out several number_of_components for UMAP and found that the best results were obtained when 200 components were utilized. After that, we reduced the dimensionality of our set using two ways: one where the euclidean distance metric is utilized and the other where the cosine distance metric is utilized. We fit a K-means model with 20 clusters to both sets to see the change in performance. The results are as follows:



Contingency Matrix for UMAP with 200 components and Euclidean Metric

Deniz Orkun Eren – UID: 905624625
Matthew Waliman – UID: 605848839



Contingency Matrix for UMAP with 200 components and Cosine Metric

As can be seen, the contingency matrix when euclidean distance is used appears to be haphazard, where the model predicts the same cluster for data points of different labels. However, the contingency matrix when cosine distance is used to generate the UMAP dataset is almost perfect, meaning that utilizing the cosine metric allows the data points of different classes to be better separated.

The results for the cluster metrics for both euclidean and cosine configurations are as follows:

	Homogeneity	Completeness	V-measure	Adjusted rand	Adjusted Mutual Info
Euclidean UMAP	0.013	0.014	0.014	0.003	0.010
Cosine UMAP	0.563	0.587	0.575	0.450	0.573

From the results, we can clearly see that using the cosine metric leads to better results.

QUESTION 12

Deniz Orkun Eren – UID: 905624625
Matthew Waliman – UID: 605848839

Examining the contingency matrix when euclidean distance is used to generate the UMAP embeddings, we can see that the K-means classifier has failed to accurately determine distinct clusters in the dataset. The clustering results almost appear to be random, as a predicted cluster appears to contain samples from distinct classes. This is also apparent from the low homogeneity score. This implies that using euclidean distance makes it difficult to separate the data points.

On the other hand, the contingency matrix when cosine distance is used to generate the embeddings is almost perfect. We can clearly see that the predicted clusters contain data points of almost one class and we can also see that elements of one class are assigned to a distinct cluster, implying high homogeneity and high completeness. This makes using this distance metric a much better choice for separating the data points.

The intuition towards why the cosine distance performs better is given in the project prompt. When we have two similar documents with different lengths, the euclidean distance will not be able to correctly identify the similarity of the data points because of the length difference. However, using cosine distance allows us to accurately capture the relationship between the data points.

QUESTION 13

Let us now examine the results for 20 news group clustering with K-means clustering using different approaches: No dimensionality reduction, SVD with 200 components, NMF with 20 components and UMAP with 200 components when cosine distance is utilized. The results of the 5 metrics for these configurations are as follows:

	Homogeneity	Completeness	V-measure	Adjusted rand	Adjusted Mutual Info
Cosine UMAP	0.563	0.587	0.575	0.450	0.573
200 SVD	0.329	0.408	0.364	0.09	0.362
20 NMF	0.316	0.367	0.340	0.101	0.337
No Dim Reduction	0.37	0.44	0.41	0.133	0.408

By comparing the results, we can see that the best clustering is obtained when UMAP with the cosine distance is utilized.

Deniz Orkun Eren – UID: 905624625
Matthew Waliman – UID: 605848839

QUESTION 14

So far, we only looked at how different data representations affect the resulting clustering. Now, let us look at different algorithms and see how they compare.

The first algorithm that we will look at is called **Agglomerative Clustering**. The algorithm starts by treating each object as a singleton cluster. Next, pairs of clusters are successively merged until all clusters have been merged into one big cluster containing all objects.

There are different criteria to consider when merging the clusters These are called linkage criteria. The ward criteria minimizes the variance of the merged clusters while the single criteria uses the minimum of the distances between all observations of the two sets.

After reducing the dimension of the dataset using UMAP with 200 components, let us now see how the different linkage criteria affect the clustering results.

	Homogeneity	Completeness	V-measure	Adjusted rand	Adjusted Mutual Info
Single Linkage	0.017	0.404	0.033	0.000	0.028
Ward linkage	0.533	0.568	0.550	0.404	0.549

By examining the metrics, we can state that using ward as the linkage criteria leads to more desirable results.

QUESTION 15

The next algorithms we are going to examine is DBSCAN and HDBSCAN.

The **DBSCAN** algorithm groups together points that are close to each other based on a distance measurement and a minimum number of points. It also marks points that are not close to the clusters as noise samples.

HDBSCAN is very similar to the DBSCAN algorithm. The main difference is that while DBSCAN relies on a maximum distance parameter when generating the clusters, HDBSCAN generates the clusters based on the minimum amount of points that should belong in a cluster.

Deniz Orkun Eren – UID: 905624625
Matthew Waliman – UID: 605848839

In our implementation, we set the maximum distance between neighbors to be 0.5 for DBSCAN and we set the minimum number of samples to form a cluster to be 100 for HDBSCAN.

The clustering evaluation scores for these two techniques are as follows:

	Homogeneity	Completeness	V-measure	Adjusted rand	Adjusted Mutual Info
DBSCAN	0.116	0.524	0.190	0.020	0.181
HDBSCAN	0.398	0.591	0.476	0.190	0.475

Based on these results, we can conclude that the HDBSCAN algorithm works better for our dataset.

QUESTION 16

Based on the previously obtained results, it is clear that the HDBSCAN algorithm offers a better performance. Let us now plot the contingency matrix of this method:

Deniz Orkun Eren – UID: 905624625
Matthew Waliman – UID: 605848839

In total, HDBSCAN generated a total of 12 clusters. It can be seen that one of the clusters is labelled as -1. These are the datapoints that have been identified to not belong to any cluster and thus are treated as noise.

The contingency matrix of HDBSCAN shows that it offers superior performance when compared to regular DBSCAN. However, it should also be noted that one of the clusters in the dataset has been identified by the algorithm as noise, which is not desirable. Furthermore, we expected the model to come to the conclusion that there are 20 clusters as there are 20 different classes in the dataset. However, this turned out to be untrue.

QUESTION 17

Now that we have a general idea about how to incorporate dimensionality reduction and clustering model selection into our pipeline, let us search through some possible scenarios to see which combination gives the best performance. The options that we searched through were as follows:

Module	Alternatives	Parameters
Dimensionality Reduction	None	None
	SVD	N_components = [5,20,200]
	NMF	N_components = [5,20,200]
	UMAP	N_components = [5,20,200] Metrics = cosine
Custering	K-means	k=[10,20,50]
	Agglomerative Clustering	n_clusters=20, linkage=ward
	DBSCAN	eps=[0.5,5]
	HDBSCAN	Min_cluster_size = [100,200]

It is important to note that some of the combinations in this table were not carried out. More specifically, combining the HDBSCAN and Agglomerative Clustering models with data when no dimensionality reduction technique is used proved to be challenging due to memory constraints. So, we omitted those from our search.

Deniz Orkun Eren – UID: 905624625
Matthew Waliman – UID: 605848839

After running the search process, the best five combinations in terms of adjusted rand index turned out to be:

Dim Reduction	Model	Adj Rand Score
UMAP 200 components	K-means 20 clusters	0.455
UMAP 20 components	K-means 20 clusters	0.436
UMAP 5 components	K-means 20 clusters	0.435
UMAP 200 components	Agglomerative Clustering	0.417
UMAP 5 components	Agglomerative Clustering	0.410

From the results, we can see that the best results were obtained with using UMAP with 200 components and K-means with 20 clusters.

The fact that this dimensionality reduction technique offered the best results is not that surprising. From question Question 11, we have already examined the amount of components that gives the best performance for UMAP and that was found to be 200. The fact that UMAP offers better performance when compared to SVD or NMF can be attributed to the fact that UMAP uses the cosine distance to generate the embeddings. This allows us to identify similar data points with differing lengths, which lead to a better representation of the data in the lower dimensions.

The fact that the best model for this dataset is K-means with 20 components is also not surprising. From the dataset, we know a priori that there will be 20 clusters. This information allows the model to form more accurate clusters and therefore get better results. In fact, observing the best combinations shows that the best models are the ones that set the number_of_components parameter to 20. Since HDBSCAN and DBSCAN do not make use of such information, we expect them to reach this conclusion on their own. However, results obtained from previous sections also show that this is not the case. Hence, the best results are obtained with K-means with k=20.

The values for the other performance metrics for the best combination are as follows:

Deniz Orkun Eren – UID: 905624625
Matthew Waliman – UID: 605848839

Model	Homogeneity	Completeness	V-measure	Adjusted rand	Adjusted Mutual Info
Best Combination	0.567	0.596	0.581	0.455	0.580

Part 2 - Deep Learning and Clustering of Image Data

In this part of the project, we were tasked with clustering the images available in the tf_flowers dataset. To generate meaningful features that can be used to perform clustering, we rely on a VGG network trained on the ImageNet dataset.

QUESTION 19

VGG16 was trained on Imagenet, which is a much larger dataset with 1000s of classes. However, we can still use part of the VGG architecture to extract features. VGG16 consists of a convolutional architecture which produces features that are then fed into three fully connected layers that are used for classification. As VGG16 was trained on a much more complicated task, we expect the extracted features to be general features of the images that are good for performing classification instead of features that are specific to a task. As such, we can use the features generated by VGG16 to perform clustering of an entirely new dataset because the features learned by VGG16 are generally applicable and not domain or task specific.

QUESTION 20

The helper codebase makes use of the FeatureExtractor class which uses the pretrained VGG16 model to extract a set of features from an image. In order to do this, we load all the images into a data loader which applies a resize function that resizes the images to 224x224 and normalizes them. These are then passed into the VGG16 architecture which consists of several convolutional and max pooling layers in order to extract features from the images.

QUESTION 21

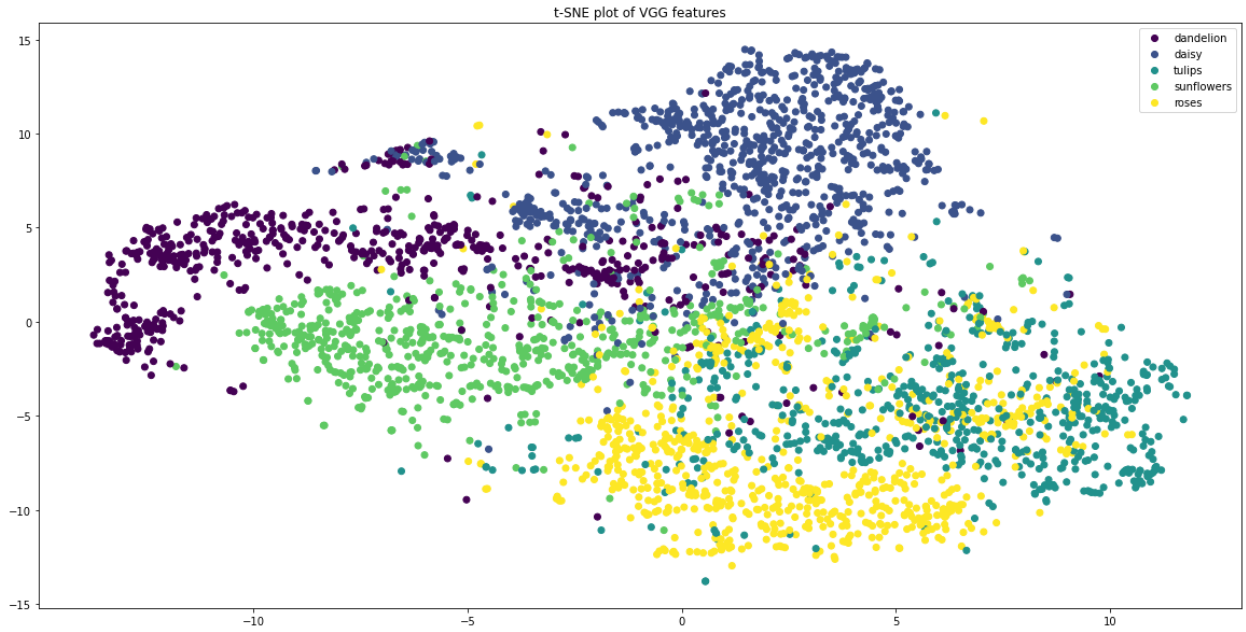
All the original images have varying sizes, however after preprocessing, they are all of size **224x224x3**. After passing through the VGG network, they have a feature vector of size **4096**.

QUESTION 22

The extracted features are dense as they contain no 0 values. On the other hand, the TF-IDF features generated for the previous part of this project are extremely sparse, with a high number of 0 values.

Deniz Orkun Eren – UID: 905624625
Matthew Waliman – UID: 605848839

QUESTION 23



By comparing the two figures on the helper code, we can see that t-SNE offers better separation between classes than a simple PCA dimensionality reduction.

It can be observed that most of the points that belong to the same class are clustered together in his 2D space. However, there is still some overlap present between clusters for different classes since some points of different classes are observed to be close to each other. This implies that clustering these data points will be difficult as they can easily be assigned to multiple clusters.

QUESTION 24

Now that we have our features, we can make use of the clustering algorithms that we utilized earlier to separate the generated features into clusters. On top of our earlier methods, we also test using an autoencoder to perform dimensionality reduction of the features generated by VGG16. The parameters that have been searched in order to get the best clustering performance are as follows:

Module	Alternatives	Parameters
Dimensionality Reduction	None	None
	SVD	N_components = [50]

Deniz Orkun Eren – UID: 905624625
Matthew Waliman – UID: 605848839

	UMAP	N_components = [50]
	Autoencoder	Num_features = [50]
Custering	K-means	k=[5]
	Agglomerative Clustering	n_clusters=5, linkage=ward
	HDBSCAN	For dimension reduction with None, SVD and Autoencoder: Min_cluster_size = [5,10,15,20,30,50] Min_samples = [1,5,10,15,20,50] For dimension reduction with UMAP: Min_cluster_size = [80,120, 200] Min_samples = [10,15,20]

After searching through the parameters, the adjusted rand scores obtained for each combination were as follows:

Adjusted Rand Scores	K-Means	Agglomerative Clustering	HDBSCAN
None	0.190	0.189	0.015 when Min_cluster_size = 10 Min_samples = 1
SVD	0.194	0.133	0.027 when Min_cluster_size = 15 Min_samples = 1
UMAP	0.396	0.375	0.35 when Min_cluster_size = 80 Min_samples = 20
Autoencoder	0.211	0.148	0.045 when Min_cluster_size = 20 Min_samples = 1

Deniz Orkun Eren – UID: 905624625
Matthew Waliman – UID: 605848839

From the results, we can see that the best performance on the clustering task is obtained when UMAP with 50 components is used to perform dimensionality reduction and the K-means algorithm with k=5 is used to perform clustering.

QUESTION 25

In the last part of the project, we are tasked with performing classification instead of clustering using a multilayer perceptron (MLP). The testing accuracy of the model when combined with different data dimensionality reduction techniques are as follows:

	Test Accuracy
Full VGG16 (No dimensionality reduction)	0.914
SVD with 50 components	0.907
UMAP with 50 components	0.853
Autoencoder with 50 components	0.887

As can be seen, the performance of the MLP classifier is best when no dimensionality reduction technique is utilized. Using any dimensionality reduction technique decreases performance by at least 0.007 percent and at most 0.06 percent. This can be attributed to the fact that the dimensionality reduction process leads to some loss of information that is present in the original dataset. When the model has access to all the original information, the performance is naturally better. However, as can be seen from the results, the difference is not that significant.

By comparing the results between clustering and classification, we can see that the combination that gives the best clustering performance does not necessarily give the best classification results. In clustering, the best performance was obtained when UMAP was used to perform dimensionality reduction. However, for classification, the best results were obtained when no dimensionality reduction was utilized. In fact, when compared to other dimensionality reduction methods, using UMAP resulted in the worst test accuracy, while using SVD which was the worst dimensionality reduction for clustering gave the best test results. These all show that having data in a form that is easily clusterable does not necessarily mean that it can be utilized to perform classification with good results.