Clustering and Dimensionality Reduction Summer Report

Veronica Alfaro, Daphney Cajuste, Jenna King, Morgan Ruffner, and Jacqui Schafer

November 3, 2020

1 Introduction

This summer, 5 current and recent undergraduate students at the University of Michigan worked on an 8-Week investigation into applications of Clustering and Dimensionality Reduction algorithms and how they can be used in succession to help with data visualization and analysis. Along with applying these techniques to unique topics of interest, each studied many different clustering and dimensionality reduction algorithms to better understand situations in which they are most useful. All projects were self guided in choice of data sets and project design.

2 Definitions and Terminology

Dimensionality Reduction (DR) Methods Dimensionality Reduction (DR) is the transformation of data from a high-dimensional space with a large number of variables into a low-dimensional space whilst retaining key features and properties of the original data. DR allows for faster data analysis by removing potentially extraneous features and reducing the size of input data. It is often used to facilitate visualization and interpretation of high-dimensional data. Generally, DR algorithms can be divided into feature selection (which involves choosing a subset of features) and feature extraction (which involves transforming high-dimensional data into fewer dimensions). Many common techniques for DR are described below.

Principal Component Analysis (PCA) Principal Component Analysis (PCA) aims to retain trends and patterns in a data set by projecting the data onto lower dimensions called principal components (PCs) with the goal of summarizing the data using a limited number of PCs [24]. The first PC is chosen to maximize the variance of the projected points onto the PC, subsequent PCs are selected similarly, and are to be uncorrelated from all previous PCs, so that each is geometrically orthogonal.

t-distributed Stochastic Neighbor Embedding (tSNE) t-distributed Stochastic Neighbor Embedding (tSNE) is a nonlinear DR technique appropriate for embedding high-dimensional data by finding patterns in the original data and identifying observed clusters based on similarity of multiple features. It generates a low-dimensional representation of the data so that similar objects are modeled by nearby points and dissimilar objects are modeled by distant points.

Uniform Manifold Approximation and Projection (UMAP) Uniform Manifold Approximation and Projection (UMAP) is useful when data is uniformly distributed on the Riemannian manifold, when the Riemannian metric is locally constant, and when the manifold is

locally connected. If these assumptions hold true, an embedding is found by searching for a low dimensional projection of the data that has the closest possible equivalent fuzzy topological structure [28].

Clustering Methods Clustering is the result of unsupervised machine learning techniques in which similar objects are grouped together. While different approaches use different metrics to measure similarity, each will construct potentially distinct but valid clusters. Many common clustering techniques are described below.

K-Means Clustering K-means Clustering is the most commonly used clustering algorithm that uses squared error. The algorithm initially selects a random k elements from the data set as centers, and assigns each object to the closest cluster center. The mean of each cluster is then calculated and taken as the new center. Each object is once again reassigned to the closest cluster center, and this process is repeated until convergence. Although K-means clustering is sensitive to the selection of the initial k cluster centers, it is widely popular as it is simple to implement and has a relatively low time complexity [16].

Hierarchical Clustering Hierarchical Clustering describes clustering methods that group data into a nested series of partitions. This can be done by starting with all data objects in one single cluster, which is then repeatedly split into distinct clusters until a stopping criterion is met (divisive clustering), or by starting with all data objects in singleton clusters and merging these clusters until a stopping criterion is met (agglomerative clustering). In either case, the goal is to create partitions with high similarity within clusters and high dissimilarity between different clusters [16].

Agglomerative Hierarchical Clustering Agglomerative Hierarchical Clustering is a form of hierarchical clustering in which each data object is initially put into its own distinct cluster, and these clusters are merged together based on their dissimilarity values until a prespecified stopping criterion is met. Agglomerative clustering can be done using either single-link or complete-link algorithms. In single-link clustering, the distance between clusters is defined as the minimum distance between all pairs of data objects in the two clusters, whereas in complete-link clustering, this distance is defined as the maximum distance between all pairs of data objects [16].

Random Forest Clustering A random forest is made up of multiple random decision trees, where each decision tree takes in an input vector and moves it down the tree based on a series of "decisions" that result in assigning the input a classification. The random forest chooses the classification for each input that the most trees agree on. As the data is run through each tree, proximities are computed between each pair of examples such that if two examples reach the same terminal node, or class, their proximity is increased by one. At the end of the process, the proximities are normalized by dividing by the number of trees. The resulting proximity table can be used as a similarity measure in some clustering methods, for instance in the multi-layer clustering method.

Gaussian Mixture Model Clustering Gaussian Mixture Model is a probabilistic model that assumes data is generated from a finite number of Gaussian distributions with unknown parameters. It can be used to create clusters of data by randomly initializing these distributions

and optimizing their parameters to better fit the dataset. Data objects can then be assigned to the Gaussian distribution to which they are most likely to belong [31].

3 How clustering plays a vital role in Smart Cities data

Daphney Cajuste

3.1 Introduction

Have you ever been riding a bus and then out of nowhere you begin to wonder, "how do they even come up with the bus routes? Who thought it was ideal to have a bus stop right here? And how do they choose how often the buses should run?" Well the answer is, data analysts use clustering to find the optimal solutions to their problems.

Now, what exactly is a cluster? Clustering is the act of grouping objects based on their similarity. For instance, we cluster food all the time. When we go to the grocery store, all the foods are placed based on clusters, whether it is fruits, vegetables, dairy, meat, bread, etc.

So what exactly is a smart city? A smart city is a city that infuses technology in order to enhance and effectively optimize the quality of life or production of the city. This includes but is not limited to solid waste management, power supply, public transportation, and IT connectivity. Smart cities are important because they improve the quality of life while also providing a way for us to engage and improve our environment. A few examples of smart city productions are smart street lamps, smart air quality sensors, and garbage sensors. Smart street lamps are public street lights with sensors to detect and adjust the brightness based on activity. This way the city can conserve energy and money. Smart air quality sensors collect air quality data using IoT sensors. The collected data allows the city to observe air pollution and take the necessary steps to ensure a healthy city. Lastly, garbage sensors are trash drop off centers that collect trash below ground in order to reduce noise pollution, carbon footprint, and odor in the air. Through examples like these, cities are able to improve the environment while also protecting its citizens.

3.2 Dublin Bikes

One particular Smart Cities initiative is the Smart Dublin project, which is an initiative to make Dublin more environment friendly and enhance the quality and production of living. Within this initiative, they have various projects like Smart Benches and DublinBikes. Smart Benches are benches, charged by solar panels, with charging ports, digital advertising spaces, and WI-FI. These benches collect non-personal data on temperature, humidity, and how much energy the benches produce and consume. Then there is the DublinBikes project which was my focus for this summer. The DublinBikes project was Dublin's way of encouraging its citizens to bike more often, by placing bike stations throughout Dublin in order to reduce air pollution, noise pollution, and congestion. This way people can drive into Dublin and bike within the inner city.

With the help of Dublinked, Dublin's open data platform, I was able to find data on how many bike stands were available at a given time. I was even able to find an article called Usage Patterns of Dublin Bikes stations by James Larlow to replicate [23]. With Larlow's data, he took the number of bikes available in a bike stand, in February 2017, and he clustered the data using k means, where k=3. This way he was able to cluster the data based on the average weekday usage.

From there he was able to plot a graph of the average weekday usages of the bikes, where x is the time throughout the day and the y-axis is the % full. Based on the graph, Larlow concluded that the green line was used for commuting into the city in the morning. So the green bike stands would be found in the residential area. The blue clusters, on the other hand, would be found in the

city center as they are mostly empty from midnight to about 8 am and also around 5 and till 11:59. Larlow also noticed that in the evenings the blue bike stations would be nearly empty while the green bike stations were gradually getting more full and that is because people would head back home from work. As for the red bike stations, they had a more steady supply throughout the day.

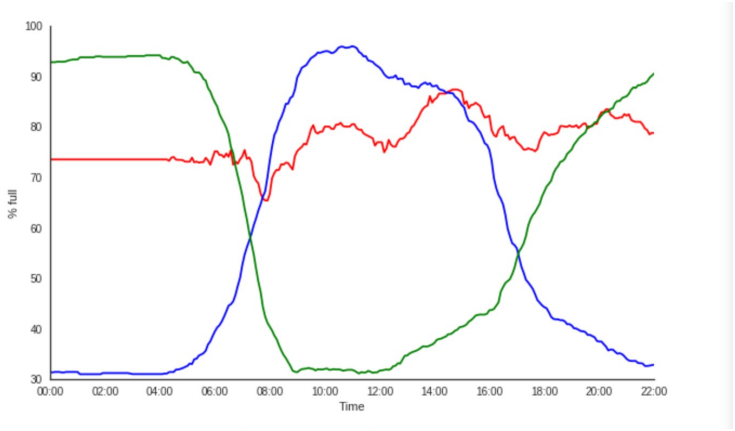


Figure 1: James Larlow's graph of the Average weekday usage for the three types of stations

Finally, after analyzing the graph, he plotted the bike stand cluster onto a map to visualize which bike stands were used and how often. After plotting the cluster onto a map, Larlow noticed that the blue stations were focused by business offices and the green stations were outside the city center in the residential areas.

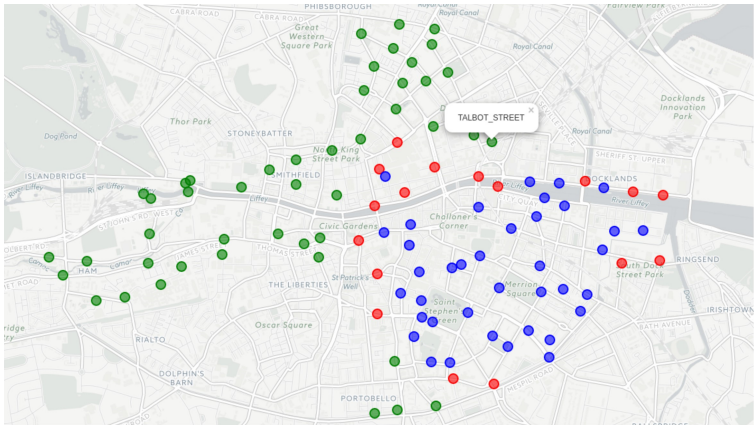


Figure 2: James Larlow's clusters on a map of Dublin

So, I began to follow in his footsteps, using Matlab instead of Python and HTML. I first read in the data file into a matrix and then clustered the data using k means clustering, where k is 3. I then plotted the results, which are displayed in the figure below. At the time, I was surprised to see that the data clustered into a straight line. I was expecting for the clusters to be more distinctive, so I began to analyze why my data turned out that way.

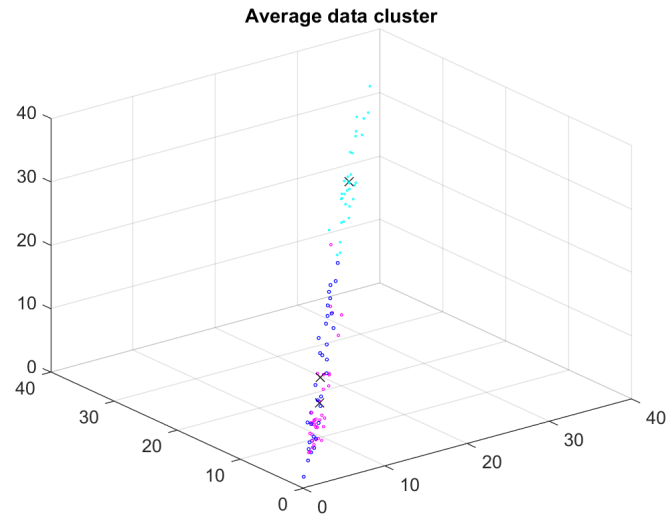


Figure 3: The results when Kmeans didn't cluster well.

After my analysis, I realized that the x,y, and z coordinates of the graph was the timestamps at 00:00:08, 00:02:09, and 00:04:08. Therefore since the number of bikes available was mostly the same from 12 am to 12:04 am, the bike stands clustered into a straight line. However, once I had changed the dimensions of the graph to 00:00:08, 11:38:09. and 23;18:09, the clusters became more distinctive. As you can see by the example below the graph shows the 3 clusters. Each cluster was nicely formed.

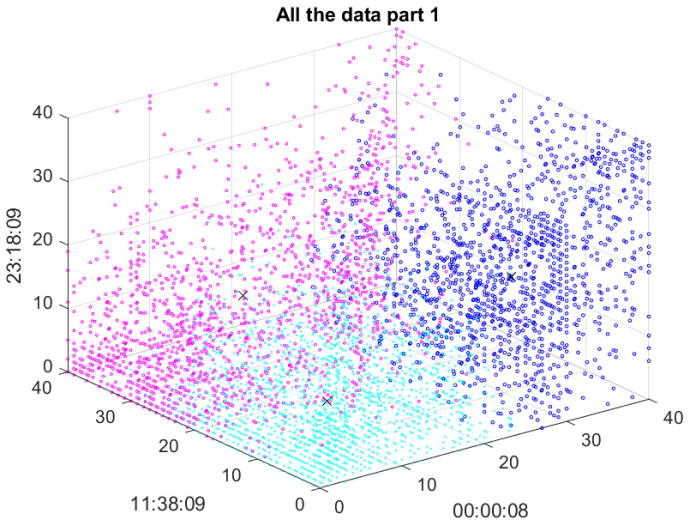


Figure 4: Kmeans clustering of the Dublin Bikes

From there, I separated the cluster. So if the bike stands clustered in the first cluster, then that bike stand would be in a matrix of sums of all the bike stands in cluster one. Afterward, I got the average of all the bike stands in the first clusters and calculated for the percentage of how many bikes were available. Then I proceeded to plot a graph of each cluster and the average amount of

bikes available over a course of time.

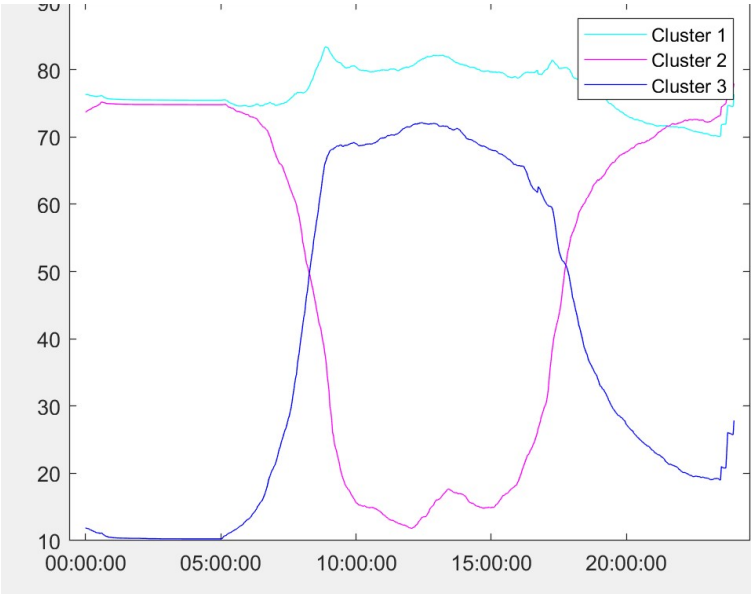


Figure 5: My results after calculating the percentage of how full the stations were

My results almost clustered the same as Larlow's. Just like Larlow, my cluster 1, cyan, resembled mostly like his red cluster. While the blue cluster resembles his blue cluster and the magenta cluster was most like the green cluster. As a result, I was able to predict that the blue cluster was most likely in the inner city, the magenta cluster in the outskirts of the city, and the cyan was probably in the middle.

My next step was to plot the clusters onto a map of Dublin. In order to do that, I needed to read in the station location file. Once I did that I made sure to match the station cluster numbers with there latitude and longitude location. Once I completed that I used webmap to plot my clusters. My results are below.

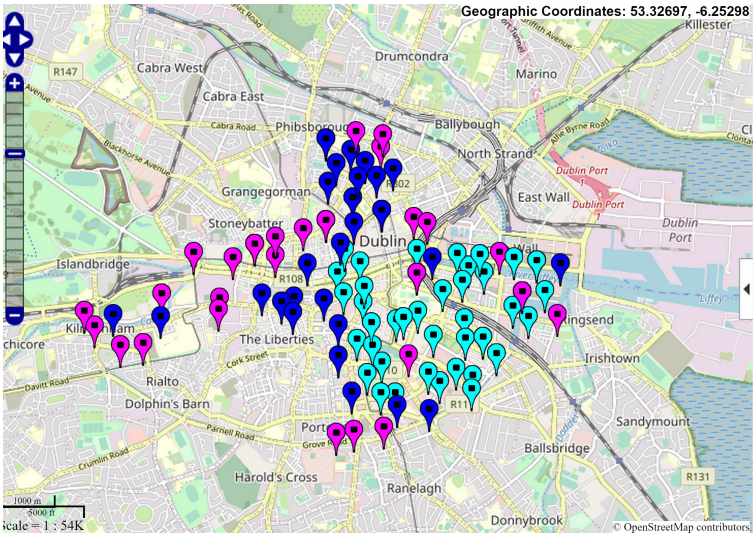


Figure 6: My results after plotting the clusters onto a map of Dublin

Based on my results, I could see that my prediction wasn't 100% correct. The magenta is still in the outskirts like I predicted however, the cyan is more in the city center rather than the blue and the blue is more in the middle.

After realizing this I did more analyzing to see why my clusters would be so different. I ended up doing some research on the way in which kmeans clusters in matlab verses python and found out that they cluster differently. I found out that while python default is to run kmeans 10 times, matlab default is to run kmeans once unless specified. That means whatever random location matlab chooses to start with, will not be updated in order to find the best solution. Therefore, Matlab doesn't search for the optimal solution, while python is closer to the optimal solution.

3.3 Conclusion

Although my data didn't cluster the same, a few things can be taken away from this. Clustering the Dublinbikes data allowed for me to see the trends of the bike stands. I was able to see which bike stand was used more often than the others. Clustering this data allows the city of Dublin to make further decisions on the usefulness of the Dublinbikes project, if they want to add more bikes to a particular station, or even if they want to add more bike stands in general. Other cities can even use these clusters to see if they want to replicate this project within their city. This shows the importance of clustering in smart cities because with this information cities are able to evaluate the effectiveness. One could even go further and cluster the congestion of the roads before and after the bike stands were placed and see if congestion decreased.

Clustering is also important with smart city data when it comes to smart meter data. Smart meter data measures the consumption of electricity in a given area. As a result, you can use these clusters to find out which homes consume the most electricity. This allows for a better understanding of household behaviors and can be used for further analysis, like whether or not a city wants to go solar-powered.

Consequently, clustering plays a vital role in smart city data in order to further evaluate. When implementing smart city projects, the city must constantly analyze data to see how it affecting the people and the environment around them. This allows for the city to make decisions that will optimize and enhance the quality of life and production of the city.

4 Current Applications of Clustering for the Diagnosis, Progression, and Treatment of Alzheimer's Disease

Morgan Ruffner

Clustering and dimensionality reduction are becoming increasingly more relevant in biomedical fields as the amount of medical data widely available continues to increase. These data sets are often unlabeled, and thus nothing is known about the relationship between observations or what patterns may exist in the data. One beneficial aspect of clustering for biomedical purposes is the ability to take large amounts of data about a patient, which could include risk factors, test scores, physical measurements and characteristics, etc., and use clustering to visualize underlying patterns and subgroups of patients within this data. This can be especially helpful for creating more personalized treatment for patients - instead of treating all patients with a given disease as a uniform block, groups can be identified with particular characteristics or needs, which can inform decisions about their plan of treatment. In addition, by keeping track of the progression of the disease in similar patients, one may be able to gain better insights into how the disease will progress in another, for example with Alzheimer's disease, it may help to predict which patients will

transition from mild cognitive impairment to Alzheimer's disease. These applications of clustering are not limited to Alzheimer's disease - applications of unsupervised learning and dimensionality reduction in Alzheimer's Disease studies could also act as a model for wider biomedical applications such as studies involving the diagnosis and treatment of other diseases. The goal of my work this summer was to investigate some of the current clustering methods associated with Alzheimer's Disease, produce similar results of my own, and investigate the comparative viability of these methods for future research.

4.1 Comparison of Clustering and Dimensionality Methods on OASIS-3 Dataset

The Open Access Series of Imaging Studies (OASIS) project consists of three neuroimaging datasets that are available for use by the scientific community [3]. I used the OASIS-3 dataset, which is entitled "Longitudinal Neuroimaging, Clinical, and Cognitive Dataset for Normal Aging and Alzheimer's Disease", for the first portion of my research. This dataset consists of data from 609 cognitively normal adults and 489 adults in various stages of cognitive decline, collected through MR sessions, PET imaging, and clinical assessments. The direction of my research with the OASIS-3 dataset was guided by the insights from "The Application of Unsupervised Clustering Methods to Alzheimer's Disease," in which the authors describe the clustering algorithms that give the most effective insights into the nature, diagnosis, and progression of Alzheimer's disease (AD), based on a collection of prior studies. The most common methods mentioned in this paper are K-Means, K-Means-Mode, multi-layer clustering, and hierarchical agglomerative clustering. I implemented K-Means and hierarchical agglomerative clustering with the OASIS-3 dataset, and will focus specifically on multi-layer clustering in Section 4.2.

The three clustering algorithms were performed on the FreeSurfer table, which was downloaded from the OASIS-3 dataset and converted into an Excel spreadsheet. FreeSurfer is a brain imaging software package which conducts volumetric segmentation of brain features. This table initially consisted of 2048 examples and 13 features, a portion of which is shown in Figure 7 for reference. However, I removed the first three columns ('Label', 'Label Num', and 'ID') since they did not contain information relevant to the desired clustering, leaving 10 features of various brain region volumes.

Label	Label_Num	ID	IntraCranialVol	lhCortexVol	rhCortexVol	CortexVol
OAS30783_Freesurfer53_d0056	30783530056	30783	865980.0594	175425.9982	178487.3783	353913.3765
OAS31026_Freesurfer53_d0043	31026530043	31026	1561633.158	201692.1527	206110.4191	407802.5718
OAS30210_Freesurfer53_d0047	30210530047	30210	1317624.322	174800.271	174519.6573	349319.9283
OAS30869_Freesurfer53_d1691	30869531691	30869	1504320.145	198121.0169	217500.878	415621.895
OAS30869_Freesurfer53_d2290	30869532290	30869	1506510.159	178641.9608	207175.3125	385817.2733
OAS30271_Freesurfer53_d0004	30271530004	30271	1564811.793	160035.8274	166736.7792	326772.6066

Figure 7: Example portion of the OASIS-3 dataset

I used Principal Component Analysis to perform dimensionality reduction on the data in Matlab before clustering it. Because the features of the data had varying ranges of values, I normalized the data first by computing $\frac{1}{variance}$ and including this variable as the 'VariableWeights' input to the PCA function. I then plotted the explained variance for the three most significant components, as shown in Figure 8. Principal component 1 had a value of 76.56%, principal component 2 had a

value of 11.47%, and principal component 3 had a value of 7.58%

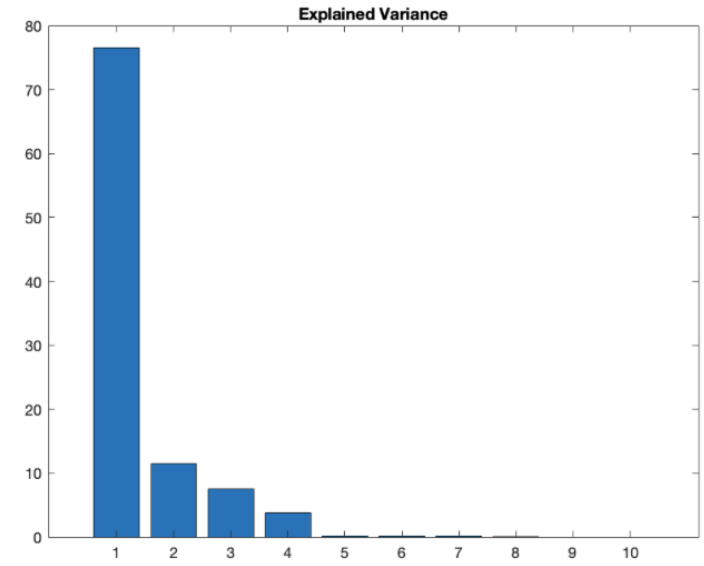


Figure 8: Plot of explained variance for the OASIS data after running PCA

I ran K-Means clustering on the data with dimensions reduced in Matlab, initially with the number of clusters k=5. The resulting clusters are shown in Figure 9.

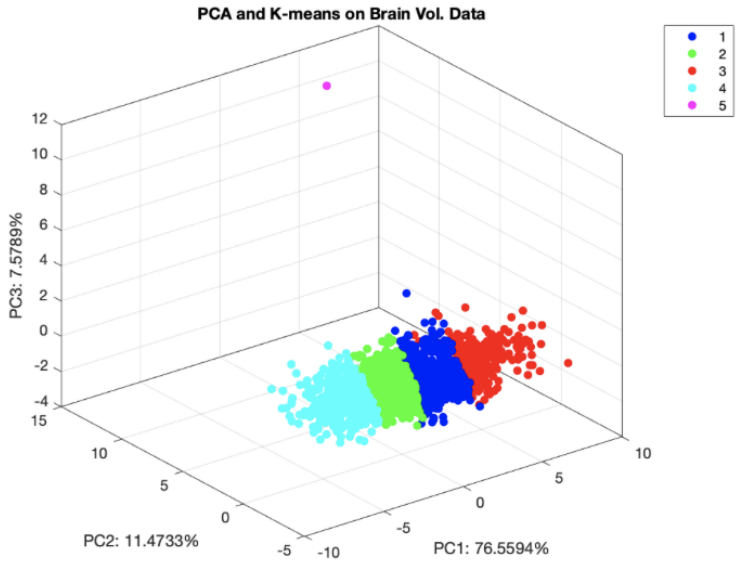


Figure 9: K-Means clustering results of OASIS data when k=5

Cluster averages for several brain volume measurements are given in Figure 10. The ranking from highest to lowest volume is consistent across all four measures (3, 2, 1, 4, 5).

Cluster	Avg. Intracranial Vol. (mm^3)	Avg. Cortex Vol. (mm^3)	Avg. Total Gray Vol. (mm^3)	Avg. Cortical White Matter Vol. (mm^3)
1	1459718.238	400757.2555	563910.5821	453807.1376
2	1608292.159	438042.5522	587653.1389	458830.7158
3	1743989.86	479506.2611	641172.8123	517438.8507
4	1442595.117	396590.6167	534729.3097	408218.4086
5	1314724.66	353870.8448	483550.5274	360084.875

Figure 10: K-Means cluster averages for significant brain volume measurements

Since the FreeSurfer data table did not contain diagnosis information, I obtained diagnosis data from the clinical assessment table and matched examples with diagnoses by ID number. The examples in the FreeSurfer table consisted of fourteen distinct diagnoses, which I sorted into three broader categories: definite dementia (AD/dem, DLBD, frontotemporal dem, AD dem language dysf, dem/pd, non AD dem, vascular dem, DAT), uncertain/unknown (uncertain dem, unc. Ques. impairment, 0.5 in memory only, unknown), and no dementia (CN, no dementia). The percentages of examples in each cluster for each of the three diagnosis categories are displayed in Figure 11.

Cluster #	**Definite dementia**	**No dementia**	**Uncertain/Unknown**
1	24.63%	66.72%	8.65%
2	18.94%	72.98%	8.08%
3	24.42%	67.28%	8.29%
4	15.82%	77.91%	6.27%
5	22.90%	71.37%	5.73%

Figure 11: K-Means cluster percentages for each of the three major diagnosis categories

The diagnosis data indicates that the pattern seen in the brain volume measurements is somewhat mirrored by the diagnoses, and therefore it seems reasonable that there may be a correlation between measures of brain volume and the likelihood of being diagnosed with dementia/AD. This would be consistent with current understandings that brain atrophy is linked to cognitive decline. For instance in Gamberger et al. it was observed that smaller brain volumes was correlated with the transition from mild cognitive impairment to dementia [9].

I ran K-Means again with k=3 in order to get a better understanding of the relationship between clusters and diagnosis, since I found there to be three major diagnosis categories. Two plots showing different perspectives of the resulting clusters are shown in Figure 12. I also colored these same scatterplots by diagnosis category, as shown in Figure 13.

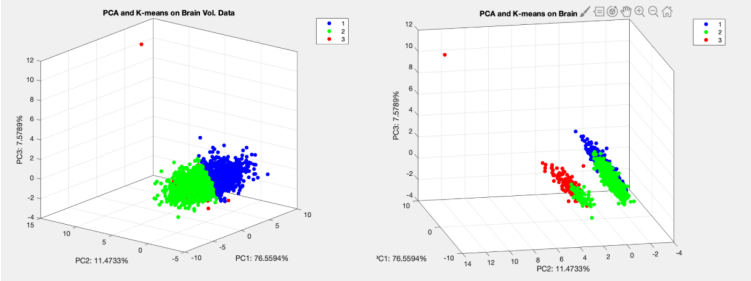


Figure 12: K-Means clustering results of OASIS data when k=3, colored by cluster numbers

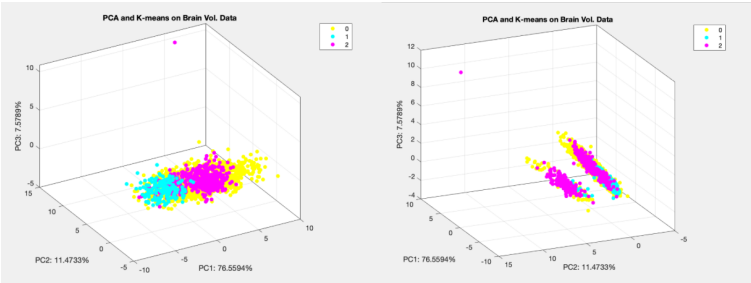


Figure 13: K-Means clustering results of OASIS data when k=3, colored by diagnosis category

In comparing the coloring of the first plot in Figure 12 and the first plot in Figure 13, both seem to have two distinct clusters side by side (green and blue in Figure 12, light blue and pink in Figure 13) with another cluster behind them (red in Figure 12, yellow in Figure 13). This seems to indicate some kind of correlation between K-Means cluster number and diagnosis category. In the second two plots, the green cluster in Figure 12 seems similar to the pink cluster in Figure 13. However, in Figure 12 the red and blue clusters are separated into distinct regions, while in Figure 13 there are some yellow and blue points in both of the two regions. This indicates that there is some degree of overlap in the K-Means clusters in regards to diagnosis category.

I next ran an agglomerative hierarchical clustering algorithm on the data in Python. Like I did for K-Means, I first performed dimensionality reduction on the data using principal component analysis. I then created a dendrogram to visualize the hierarchical clustering process, as shown in Figure 14. I utilized Euclidean distance and the ward linkage measure, which computes the distance between clusters as the sum of squared differences between all clusters.

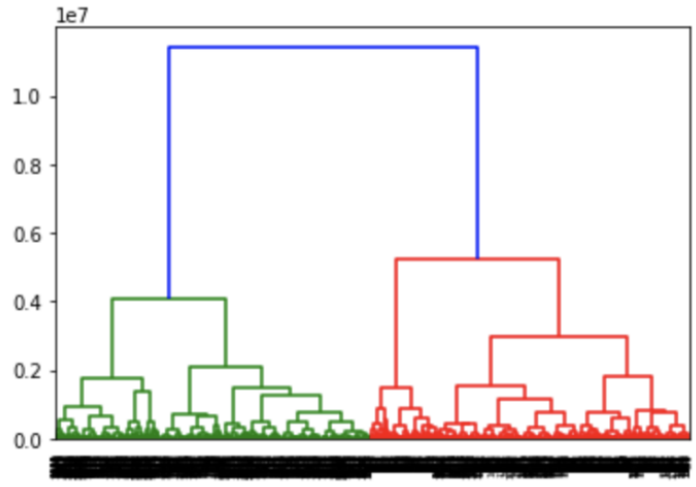


Figure 14: Dendrogram of the hierarchical agglomerative clustering process

After running the clustering algorithm with the number of clusters = 5, I obtained the clusters shown in Figure 15.

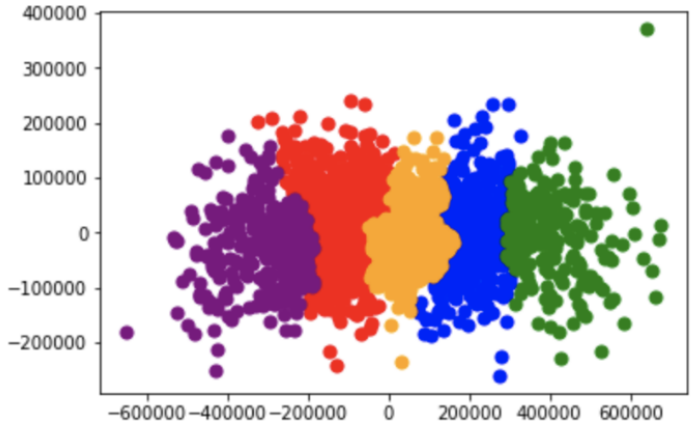


Figure 15: Scatterplot of the hierarchical agglomerative clustering results

The hierarchical clustering method seemed to produce clusters with a similar range of percentages of those with definite dementia versus no dementia. It seems that the clusters formed by hierarchical clustering were more similar to one another in ratios of diagnosis, whereas with k-means there was a more distinct range. Another interesting observation is that the rankings of the clusters with definite dementia was exactly opposite of that with no dementia for the hierarchical clustering, whereas the two were slightly different for k-means. This seems to indicate that the uncertain/unknown subjects were distributed more evenly between all five clusters.

The last clustering and dimensionality reduction method I used on the OASIS-3 dataset was UMAP. I used UMAP for dimensionality reduction, then visualized the results using the previously obtained K-Means labels, as shown in Figure 16.

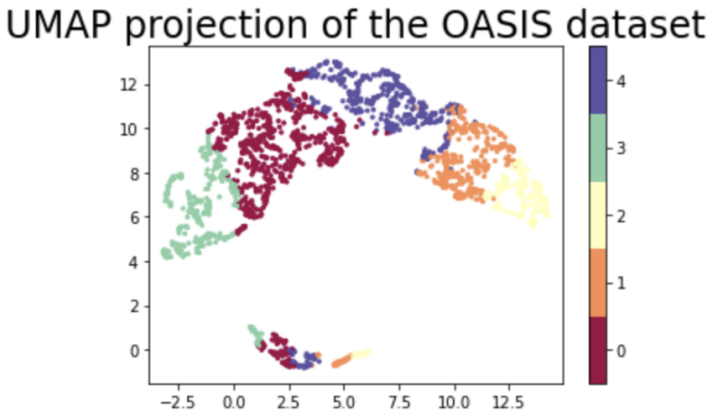


Figure 16: Scatterplot of the UMAP results with K-Means labels applied

The arrangement of the resulting clusters seems quite similar to those obtained by the hierarchical clustering algorithm in that the clusters are arranged side by side. One advantage of UMAP is that it generally preserves more of the original local structure of the data than other dimensionality reduction methods such as PCA. Therefore it may be reasonable to infer that the semicircle shape of the clustering achieved with UMAP is more representative of the original local structure than the more flat shape obtained with PCA and K-Means.

4.2 Multi-layer Clustering on ADNI Dataset

The second major portion of my work this summer was focused on replicating the results of a novel clustering algorithm described in *Homogeneous clusters of Alzheimer's disease patient population* [14]. The objective described in this paper was to identify connections between biological and clinical characteristics of Alzheimer's disease patients in order to achieve better understanding of the AD pathophysiology, improve clinical trial design, and assist in predicting outcomes of mild cognitive impairment. Biological descriptors often encompass a very noisy domain in which useful information may be hidden, and clinical descriptors are often defined by scoring systems that may be imprecise or biased. The multi-layer clustering approach aims to address these challenges by identifying homogeneous subpopulations of Alzheimer's disease patients in which relationships between clinical and biological descriptors may become more clear. This method uses clinical data as one layer and biological data as another, and constructs clusters that are simultaneously homogeneous in both layers.

The data for this section was obtained from the Alzheimer's Disease Neuroimaging Initiative (ADNI) database [1]. I used data from the ADNIMERGE table, the same table used in the paper, and extracted data from the same columns, which included ten biological descriptors (genetic variations of APOE4 related gene, PET imaging results FDG-PET and AV45, and MRI volumetric data of: ventricles, hippocampus, wholeBrain, entorhinal, fusiform gyrus, middle temporal gyrus (midtemp), and intracerebral volume (ICV)) and twenty three clinical descriptors (clinical dementia rating sum of boxes (CDRSB), Alzheimer's disease assessment scale (ADAS13), mini mental state examination (MMSE), rey auditory verbal learning test (RAVLT immediate, learning, forgetting, percentage of forgetting), functional assessment questionnaire (FAQ), montreal cognitive assessment (MOCA) and everyday cognition which are cognitive functions questionnaire filled by the patient (ECogPt) and the patient study partner (ECogSP) (memory, language, visuospatial abilities, planning, organization, divided attention, and total score)). I extracted the same number

of rows from the table as mentioned in the paper (916), but as the authors did not specify which specific rows they used, I chose these at random. The resulting data obtained consisted of one table with biological data consisting of 916 rows and 10 columns and one table with clinical data consisting of 916 rows and 23 columns, where each of these tables represented an attribute layer for the clustering process.

I implemented the multi-layer clustering approach by following the three main steps outlined in the paper.

Step 1: Compute example similarity tables (EST) through supervised learning

The first step of the multi-layer clustering method is to compute example similarity tables for each layer. These are *NxN* symmetric matrices, where N is the number of examples, in which each similarity value $v_{(i,j)}$ is a value between 0 and 1 that represents the similarity between example i and example j. The EST is created by constructing an artificial classification problem in which the original dataset are treated as positive examples and negative examples are generated by shuffling the values of the original dataset. I did this by generating columns of random numbers in Excel and sorting the original data based on these columns. Then a supervised learning model is used to discriminate between the positive and negative examples. The goal of doing so is not the classification model itself, but the information that is generated about the similarity between examples. The random forest classifier satisfies this goal by returning a proximity matrix, which gives the similarity between all pairs of examples. I ran the random forest algorithm in R on the dataset which included the original data combined with the shuffled data, using na.roughfix to impute values for missing data based on the column medians. The resulting proximity matrix was *2Nx2N*, and I extracted the first N rows and N columns, which represented the similarity between the original examples, as the EST. I did this for both attribute layers, so I ended up with two *NxN* ESTs.

Step 2: Compute clustering related variability (CRV) scores

Once the ESTs had been constructed, I implemented the rest of the clustering algorithm as a C++ program. The multi-layer clustering algorithm is similar to an agglomerative hierarchical clustering algorithm in that each example starts in its own cluster, and a measure of proximity is used to merge similar clusters together. This measure is defined by the clustering related variability (CRV) score. The CRV score for each example is obtained using the values of the EST. For each example i,

$$CRV_i = CRV_{i,wc} + CRV_{i,oc} \tag{1}$$

where

$$CRV_{i,wc} = \sum_{j \in C} (v_{i,j} - v_{mean,wc})^2 \tag{2}$$

is the within cluster value and

$$CRV_{i,oc} = \sum_{j \notin C} (v_{i,j} - v_{mean,oc})^2 \tag{3}$$

is the outside cluster value. The within cluster value is computed as a sum over columns j of row i of examples included in the same cluster C as example ex_i. In this expression $v_{mean,wc}$ is the mean value of all $v_{i,j}$ in the cluster. If example ex_i is the only example in its cluster C then $CRV_{i,wc} = 0$

because the sum will include only value $v_{i,i}$ and $v_{mean,wc} = v_{i,i}$, the difference of which will equal 0. The outside cluster value is computed in the same way, except that the sum will be computed only over examples not in the same cluster as ex_i, and $v_{mean,oc}$ is the mean value of all $v_{i,j}$ not in the cluster. The final CRV score for each cluster is the sum of the CRV scores for all the examples in that cluster,

$$CRV_c = \sum_{i \in C} CRV_i \tag{4}$$

Step 3: Apply CRV score based multi-layer clustering algorithm

Using the CRV scores, the clustering algorithm iteratively merges the most similar clusters together. Unlike other clustering algorithms such as K-Means in which a predefined number of clusters must be chosen, the multi-layer clustering algorithm has a defined stopping criterion. The process stops when further merging does not result in the reduction of example variability measured by the CRV score, therefore automatically determining the optimal number of clusters. The multi-layer algorithm is defined in the paper as follows in Figure 17.

CRV score based multi-layer clustering algorithm

1) Each example is in its own cluster
2) Iteratively repeat steps 3–8
3) For each pair of clusters x,y do
4) For each attribute layer l compute
CRV$^l x$ (CRV for examples in cluster x in layer l)
CRV$^l y$ (CRV for examples in cluster y in layer l)
CRV$^l xy$ (CRV score in union of clusters x and y in layer l)
DIFF^l = CRV$^l x$ + CRV$^l y$ − CRV$^l xy$
5) For the given cluster pair x,y: DIFF = $\min_l$ DIFF^l
6) Select pair of clusters x,y with maximal DIFF value
7) If maximal DIFF is positive then merge clusters x and y
8) Else stop

Figure 17: Steps of the multi-layer clustering algorithm as described in *Homogeneous clusters of Alzheimer's disease patient population*

One point to note is that the CRV score in the union of clusters x and y must be computed for each pair of clusters, which means that the CRV score for each element must be computed not only for the cluster that it is actually in, but also for the union of that cluster and all other clusters. In this case, any element that is in either of the two clusters should be treated as part of the within cluster score, and any element that is in neither should be treated as part of the outside cluster score. In my implementation of this step, I began by constructing a vector of length N to contain the cluster number of each example, a matrix of size N *x n_clusters* to contain the CRV scores for each element, and a matrix of size *n_clusters x n_clusters* to contain the CRV scores for each cluster. Each element v_i of the first vector represents the cluster number of element i, each element v_{ij} of the first matrix represents the CRV score of element i assuming the union of the cluster element i is in and cluster j, and each element v_{ij} of the second matrix represents the CRV score of the union of clusters i and j.

Once the CRV scores for all cluster pairs have been computed for each layer, the DIFF score for each pair is calculated by taking the minimum of the DIFF scores between the two layers. DIFF is defined as $CRV_x + CRV_y - CRV_{xy}$, which I calculated from my matrix of cluster CRV scores

as $v_{xx} + v_{yy} - v_{xy}$. Finally, the maximal DIFF score is taken from all the pairs of clusters. If this score is positive, then the two clusters are merged, and the algorithm continues by recomputing all CRV scores. In my implementation, I merged the clusters by reassigning all elements in the cluster with a higher number to the cluster with the lower number and decrementing the cluster number of all clusters with a higher number than the reassigned cluster. For example, if clusters 1 and 7 were being merged out of 10, I would reassign all elements in cluster 7 to cluster 1 and decrement clusters 8, 9, and 10 to 7, 8, and 9 respectively. Otherwise if the maximum DIFF score is not positive, the algorithm halts.

Some advantages of the multi-layer clustering approach identified by the authors are that no explicit distance measure is used, there is a well-defined stopping criterion, that the data can include many attributes, and that examples can include both numerical and categorical values and may have missing values. However, some disadvantages are that the algorithm does not work with a very small number of attributes and is sensitive to attribute copies. The result of the algorithm may be a large set of clusters in which some clusters contain only a small number of examples. Generally a small number of large clusters should be chosen and evaluated by the user.

After running the multi-layer clustering algorithm on the ADNI data, I obtained 823 clusters, out of which the three largest clusters contained 44, 31, and 12 patients. In comparison, the three largest clusters obtained in the paper contained 42, 35, and 21 patients. I ran PCA on the data in order to visualize the results of the multi-layer clustering algorithm in three dimensions, as seen in Figure 18.

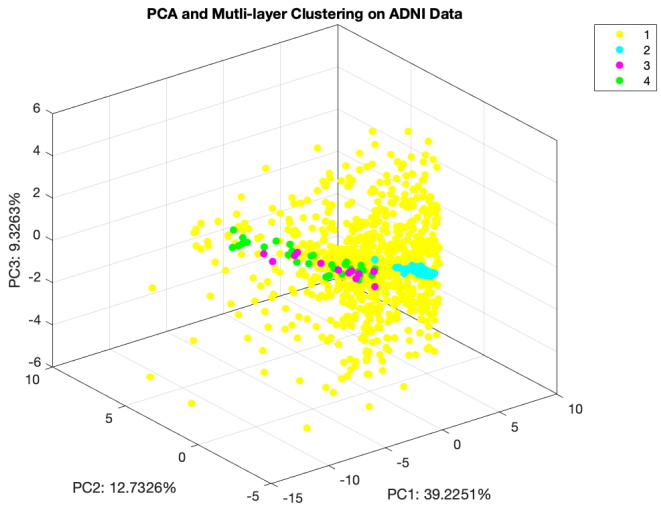


Figure 18: Results of the multi-layer clustering algorithm

The green, pink, and blue data points represent the three largest clusters obtained by the algorithm and the yellow data points represent all other clusters. In comparing the three clusters I obtained with the clusters obtained in the paper, I found some similarities in that two of the clusters showed characteristics of Alzheimer's Disease, while the third consisted of patients with characteristics of being cognitively normal. However, one setback in my results was that the three largest clusters I obtained were missing significant amounts of biological data, which made it difficult to compare the results. I hypothesize that since I used na.roughfix to impute values for missing data based on the column medians, these values all ended up clustering together. The authors did

not explicitly mention how they dealt with missing data, so it is possible that they used a different strategy that did not impact the results as much. Another possibility is that this error occurred from the rows that I chose from the original data table, and that the authors specifically selected rows that were not missing as much data. However, I was able to compare the clinical descriptors of the clusters I obtained with their clusters. Two tables containing the main diagnosis characteristics for each cluster are shown in Figures 19 and 20, in which Figure 19 represents the results obtained by the paper and Figure 20 represents my results.

Cluster	# Patients	# AD	# LMCI	# EMCI	CDRSB
A	35	30	4	1	4.81
B	21	10	9	2	3.55
C	42	30	10	2	4.15

Figure 19: Table of diagnosis statistics for clusters obtained by *Homogeneous clusters of Alzheimer's disease patient population*

Cluster	# Patients	# AD	# LMCI	# EMCI	# CN	# SMC	CDRSB
A	31	6	15	10	0	0	6.2258
B	12	2	4	6	0	0	7.125
C	44	1	1	13	19	10	0.1625

Figure 20: Table of diagnosis statistics for clusters I obtained

One observation is that the clusters I obtained consisted of a higher percentage of late mild cognitive impairment (LMCI) and early mild cognitive impairment (EMCI) individuals than the paper, in which most patients were diagnosed with Alzheimer's Disease. Two more tables containing clinical test scores are shown in Figures 21 and 22.

Cluster	ADAS13	MMSE	MOCA	FAQ
A	35	22	15	15
B	24	25	20	10
C	30	24	18	12
AD	31	23	17	13
CN	9	29	26	0

Figure 21: Table of clinical assessment statistics for clusters obtained by *Homogeneous clusters of Alzheimer's disease patient population*

Cluster	ADAS13	MMSE	MOCA	FAQ
A	33.333	22.0968	16.3226	18.51613
B	33.2418	22.0833	17.9091	19.5
C	9.2652	28.9545	26.5581	0
AD	33.43835616	22.19594595	16.42253521	15.38095238
CN	8.6471123	29.0106952	25.9193548	0.125

Figure 22: Table of clinical assessment statistics for clusters I obtained

The Alzheimer's disease assessment scale (ADAS13) and functional assessment questionnaire (FAQ) are assessments in which individuals with Alzheimer's Disease generally score high, and the mini mental state examination (MMSE) and montreal cognitive assessment (MOCA) are cognitive tests in which individuals with Alzheimer's Disease generally score low. The rows labeled AD and CN represent the average scores for Alzheimer's Disease patients and cognitively normal individuals respectively.

Conclusions Although my results differed some from those that I was aiming to replicate, I believe that working through this algorithm was a worthwhile task in that it helped me to gain a better understanding of how clustering algorithms work at the base level of the code. I also think that the clusters I obtained were still meaningful, but that the missing data in the biological layer caused them to be more heavily influenced by the clinical layer. If I were to continue this project in the future, I would try the algorithm again using a different subset of the original dataset or using a different method to replace missing values, and I would be interested in analyzing the resulting clusters further from a medical perspective in order to better characterize the subgroups. In general, I found that the multi-layer clustering approach may be advantageous when the data contains many attributes or a mix of numerical and categorical values, or when the objective is to identify small subgroups of the dataset. However, in situations in which there is a small number of features or the objective is to group the data into a smaller number of larger clusters, one of the previously mentioned clustering algorithms such as K-Means or hierarchical clustering may be more advantageous.

Overall, it seems that clustering and dimensionality reduction are very advantageous tools for learning about biomedical data, and when combined with the proper analysis by experts in the field, may be able to provide new insights into the understanding of diseases such as Alzheimer's. In addition, in the course of this research, I was able to gain a much better understanding of machine learning, in particular unsupervised learning, which I previously had very little knowledge of. This opportunity also allowed me to practice working with real data and pushed me to take initiative in designing and implementing code in various programming languages, which I think was really valuable experience. These 8 weeks of research showed me just how much there is to learn and explore in the fields of data science and machine learning, and I hope it will not be my last venture into these topics.

5 How dimensionality reduction algorithms change the clustering of US counties by confirmed COVID-19 cases

Jacqui Schafer

Does reducing the feature space of high-dimensional data affect the grouping of original feature-vectors? Over the past two months, I have been using COVID-19 US county-level data to investigate how various dimensionality reduction (DR) techniques can change the formation of clusters. I have conducted clustering before and after lossy DR to see if there is meaningful movement of data points between clusters.

Introduction to Data The data in use were publicly available US county-level COVID-19 data from the NY-Times github [35]. This data-set is regularly updated from state and local governments and health departments to track cumulative cases and deaths by US county. Figure 23 shows a snapshot of the original csv-formatted data provided by NY-Times.

These data were then transformed into a time-series format, with each row as a US county and each feature representing one day (starting from 01/21). The data were adjusted to count individual cases rather than a cumulative count. Lastly, the individual case counts were normalized by county according to the maximum daily case count. For example, if CountyA recorded 3 cases on February 1, and its maximum recorded daily case count was 100, the data point for February 1 would then be 3/100, or 0.03. Figure 24 shows this transformed and normalized data.

date	county	state	fips	cases	deaths
1/21/20	Snohomish	Washington	53061	1	0
1/22/20	Snohomish	Washington	53061	1	0
1/23/20	Snohomish	Washington	53061	1	0
1/24/20	Cook	Illinois	17031	1	0
1/24/20	Snohomish	Washington	53061	1	0

Figure 23: Original NY-Times Data

	1/21/20	1/22/20	1/23/20	1/24/20	1/25/20	1/26/20
Snohomish, Washington	0.00408163	0	0	0	0	0
Cook, Illinois	0	0	0	0.00046404	0	0
Orange, California	0	0	0	0	0.00056721	0
Maricopa, Arizona	0	0	0	0	0	0.00028121
Los Angeles, California	0	0	0	0	0	0.00022427

Figure 24: Transformed and Normalized County-Level Data

Before reducing the dimensionality of the input data, k-means clustering was performed on high-dimensional input data to group counties into four distinct clusters.

Following this clustering, the data were reduced to 2 dimensions using three different methods and then clustered again. This allowed for comparison between multiple sets of clusters of the same data, to see which DR technique was most appropriate for and faithful to the given input.

Clustering Techniques There are a range of commonly practiced clustering techniques, applicable in varying degrees depending on the nature of data. I experimented with several of these techniques, including Agglomerative clustering, Spectral clustering, Random Forest Clustering, and Birch Clustering. There were not significant changes in clustering error using any of these techniques. For consistency and straightforward comparison of clusters in both high-dimensional space and low-dimensional space, I used k-means clustering for all subsequent analysis.

Choosing Number of Clusters To choose the appropriate number of clusters to separate the data into, the k-means algorithm was run with k ranging from 1 to 10. For each k-clustering, the inertia was measured, which sums the squared errors (SSE) of all samples to their closest center. SSE will always decrease as the number of clusters increases, therefore k was chosen at a value where the improvements in inertia began to slow. Figure 25 shows the inertia for each k measured. For this particular dataset, k was chosen to be 4.

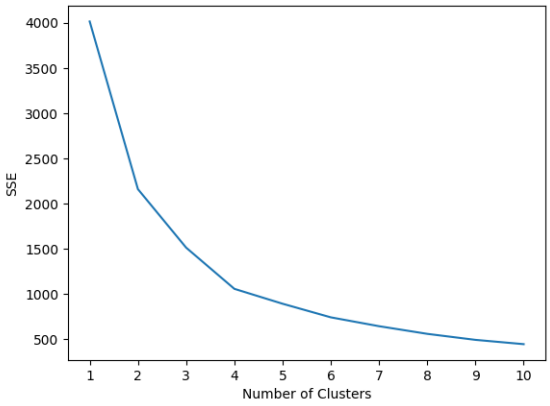


Figure 25: KMeans Clustering Performance with Changing K

DR Techniques The input feature space was over 180 dimensions representing each day since the first recorded case in the US (January 21). To allow for clear data visualization, the feature space had to be reduced to 2 dimensions. This was achieved through DR algorithms of PCA, UMAP and TSNE. These algorithms were chosen to sample a range of techniques that are suited to different applications. PCA is a very well documented and commonly used algorithm aiming to preserve the original data by generating linear combinations of input features. In contrast, t-SNE (2008) and UMAP (2018) are both far newer methods that aim to retain local and global structure using non-linear dimensionality reduction. UMAP specifically is in its early stages of adoption, and is aimed particularly at efficiency for very large datasets. By using each of these three techniques, I was able to sample a range of algorithms to see which is best suited to US-County COVID-19 time series data.

For each algorithm, two 2D scatter plots are shown, with markers colored according to cluster membership both before and after DR. These graphs help to visualize the faithfulness of each DR technique to the original high-dimension data. The same segmentation of colors would show that the data clustered in exactly the same way before and after DR, and a completely different and inconsistent coloring would show that completely new clusters were formed after DR.

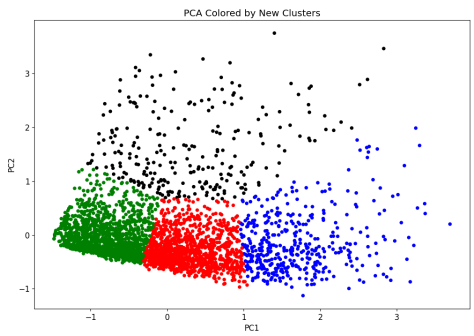


Figure 26: PCA Colored by New Clusters

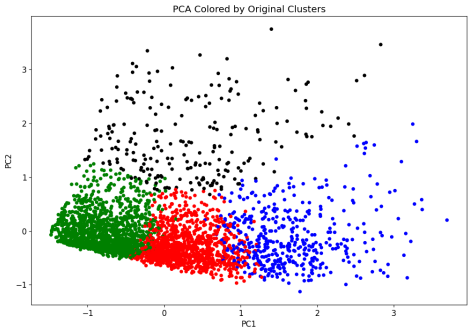


Figure 27: PCA Colored by Original Clusters

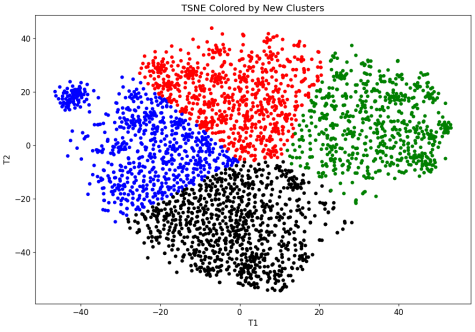


Figure 28: TSNE Colored by New Clusters

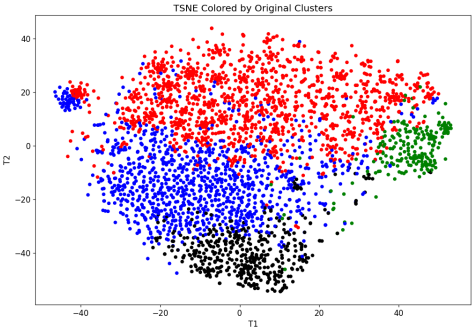


Figure 29: TSNE Colored by Original Clusters

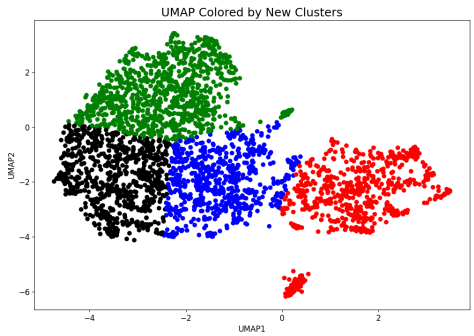


Figure 30: UMAP Colored by New Clusters

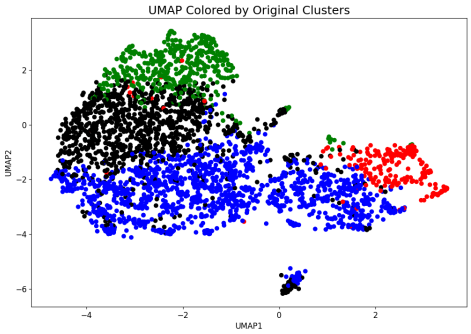


Figure 31: UMAP Colored by Original Clusters

Analysis and Comparisons of DR Techniques The DR method that affected clustering the least was PCA, with only 197/3201 counties moving from their original clusters after DR. Using PCA to generate 2D data provided a wide spread of data points, with no extreme outliers. In contrast, UMAP changed cluster formation the most, suggesting that UMAP was not appropriate for time-series COVID-19 data. There was not very clear cluster separation in any DR method. This suggests that clustering may not be the best tool for analysis of these data. Figure 32 shows a detailed analysis of key properties of each cluster before DR and after PCA, tSNE and UMAP. It shows that the original data and the PCA-reduced data share many common cluster properties, whereas tSNE and UMAP have significantly different measures of case counts and mean dates of the first recorded case.

Cluster		Original	PCA	TSNE	UMAP
0	# Counties	1079	1112	717	787
	Median Cases	265	269	158	93
	Mean Cases	796	862	407	312.73
	Min Cases	9	9	1	1
	Max Cases	37072	37072	14925	17816
	Mean Date of First Case	29-Mar	28-Mar	7-Apr	11-Apr
1	# Counties	1443	1407	820	734
	Median Cases	72	68	47	73
	Mean Cases	387	375	271	384
	Min Cases	1	1	1	1-Jan
	Max Cases	44005	44005	44005	44005
	Mean Date of First Case	13-Apr	13-Apr	19-Apr	14-Apr
2	# Counties	455	426	947	973
	Median Cases	1720	1681	766	758
	Mean Cases	5596	5780	3256	3194
	Min Cases	96	96	24	26
	Max Cases	220762	220762	220762	220762
	Mean Date of First Case	17-Mar	17-Mar	21-Mar	21-Mar
3	# Counties	224	256	717	707
	Median Cases	1507	1369	387	367
	Mean Cases	6204	5494	2450	2431
	Min Cases	61	61	1	1
	Max Cases	234609	234609	234609	234609
	Mean Date of First Case	16-Mar	17-Mar	24-Mar	28-Mar
# Counties Changed Clusters (of 3201)		-	197	1338	1587

Figure 32: Features of Each Cluster in High Dimension, and after PCA, TSNE and UMAP

Conclusion This was a valuable exploration on how US County COVID-19 data can be clustered using a range of dimensionality reduction techniques. My understanding of dimensionality reduction and clustering grew significantly, as well as my ability to interpret papers discussing new theories and the applications of these topics. I was able to develop my skills in working with real data, and in making design decisions on concepts such as normalization, standardization and the treatment of outliers. I look forward to continuing work in data science, armed with a greater understanding of how to analyze and segment high dimensional data for visualization and comprehension.

6 Clustering and Dimensionality Reduction Error Analysis and Real World Application

Jenna King

6.1 Clustering and Dimensionality Reduction Error Analysis

Clustering and Dimensionality Reduction techniques have infinite applications in most all fields of study. Due to the countless number of algorithms, the choice of techniques used is often very ad hoc. This approach of using these methods as needed has provided results in the past, but often lead to missing fundamental information. The goal of this study is to find a way to streamline the clustering and dimensionality reduction process in such a way that as little data is lost as possible. We approach this problem by learning about how much information is lost when we solve the two problems of dimensionality reduction and clustering separately in succession, as opposed to solving them together. Currently, this study has focused solely on K-Means Clustering and PCA, with the plan to expand to other clustering and dimensionality reduction algorithms in time.

Data Introduction and Method In order to accurately study the error produced in running K-Means Clustering and PCA in succession, the algorithms were run on randomly generated synthetic data. By using synthetic data, rather than real world collected data, it is easy to separate the error due to running a clustering algorithm from the error caused by the dimensionality reduction process. The methods used to generate the data are detailed below, with each experiment following the same underlying structure. Each cluster contained 500 samples, with 100 features ($d = 100$) for each sample. The mean of the first cluster was randomly chosen, and the successive cluster centers were chosen at a fixed distance from the first cluster. This distance was varied, ranging from 0 to 10, to analyze the influence of the similarity of means between clusters. The standard deviation of each cluster was randomly chosen, giving strong variation in the data.

The experiment consisted of five sequential steps: Data Generation, Pre-Clustering, Data Reduction, Post-Clustering, and Error Analysis. The process was repeated 100 times for each distance between cluster means, and the average error over those 100 iterations was recorded. This process was then repeated to evaluate error at reduced dimensions of $d' = 100, 90, 80, 70, 60, 50, 40, 30, 20, 10$.

Data Generation This step varied based on the method of construction of the data. The first method involved generating multiple different simple Gaussian Mixtures, ranging from $k = 2$ to 10 clusters. The second method involved randomly constructing the k-cluster centers in some subspace of dimension $r < d$. Gaussian noise was then generated around these cluster centers in the full $d = 100$ dimension. By doing this, a Gaussian Mixture is created, but there is a notion of lower dimensionality that exists in the data.

Pre-Clustering K-Means clustering was then applied to the synthetic data to evaluate the accuracy of the clustering before any reduction steps were taken. This step was essential, as there is the possibility that the k-means algorithm does not properly cluster the elements, especially if the means are incredibly close together. By clustering before running dimensionality reduction, the the error that came from running the K-Means algorithm could be differentiated from the error generated by applying PCA. The K-means algorithm converged to a solution after at most 10 replicates for each data set.

Dimensionality Reduction Next, a dimensionality reduction technique was applied to the synthetic data. For the time being, PCA was used during this step it is one of the most widely used methods. For this, the number of principal components chosen spanned ten intervals from $d = d/10$ to d. Each mean distance was tested at all 10 different possible reduced dimensions.

Post-Clustering The reduced data was then re-clustered with the same pre-determined number of clusters (k). This was done using K-Means Clustering. Once again, the algorithm was run with 10 replicates to ensure construction of the best possible clustering.

Error Analysis Once the data was reduced and clustered, the clusters were evaluated for improperly clustered elements. This was done using clustering error, which is computed by matching the true labels and the labels output by a given clustering algorithm [26]. Cluster labels are purely arbitrary, and for this reason, all possible cluster labelings needed to be analyzed for the best fit. The algorithm used to evaluate this error was the missRate function written by John Lipor [25]. To calculate error, the function compares the ground-truth labelings of the clusters to all possible permutations of the output cluster labels using the Hungarian Algorithm. The fraction output of mislabeled data points was then multiplied by 100 to get the percent of points mislabeled. Each iteration, the newfound error was added to the running total and once all 100 trials ran, the average was taken.

Findings

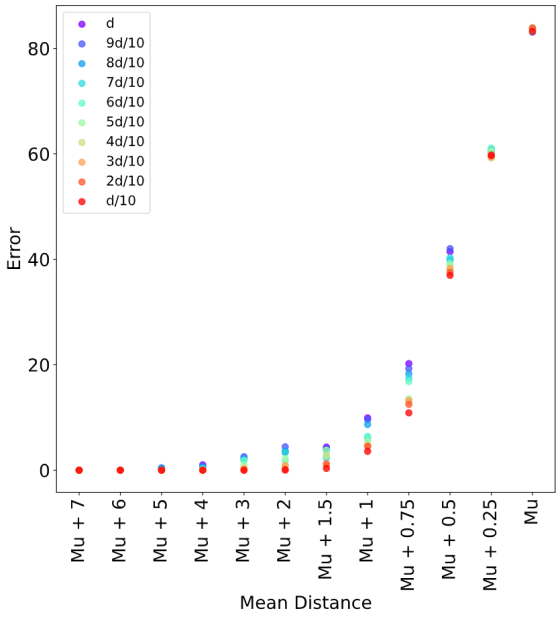


Figure 33: Clustering Error (%) of a Simple Mixture of 10 Gaussian Distributions

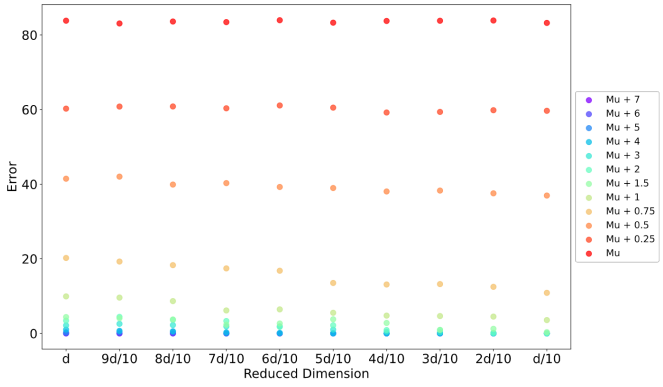


Figure 34: Clustering Error (%) of a Simple Mixture of 10 Gaussian Distributions

Simple Gaussian Mixture The results for the simple Gaussian mixtures aligned well with the expected output. The initial distance between the cluster centers had a much larger effect than the

dimensions to which the data was reduced. This finding makes sense given the nature of Gaussian mixtures and how they reduce. An interesting result found is that the error was lower for the smaller reduced dimensions than it was for the data in its original d dimensions. This can be seen in Figure 33, as all points have similarly negligent error, and as the mean increases, the error for largest dimensions increases at a faster rate. There is a point at which the means are so close together that the size of the reduced dimension no longer is relevant. At this value, error among dimension sizes is approximately equal. In Figure 34 this is evident in the fact that the center errors seem to decrease with reduced dimension, but the lowest and highest mean differences remain relatively constant in error.

Gaussian Mixture with Lower Dimensionality

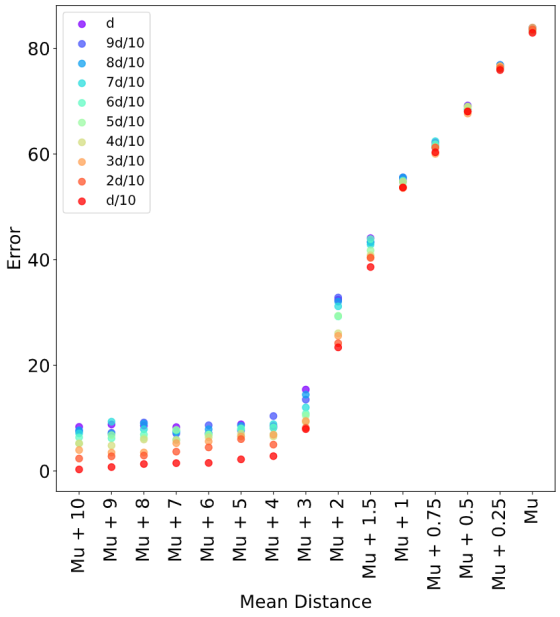


Figure 35: Clustering Error (%) of a Gaussian Mixture with $r = 10$

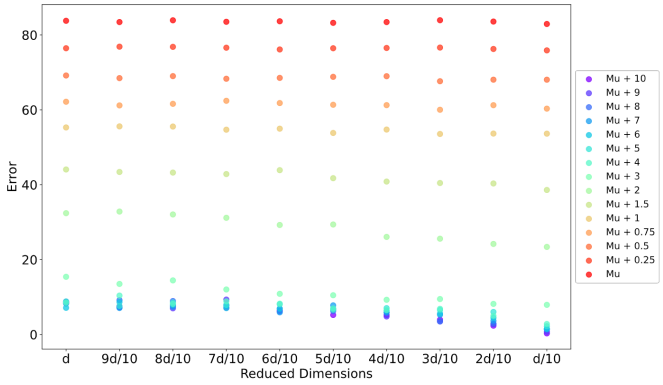


Figure 36: Clustering Error (%) of a Gaussian Mixture with $r = 10$

r = 10 Through adding a notion of lower dimensionality to the data, some interesting results were produced. In this test, the cluster centers were generated in a subspace of $r = \frac{d}{10} = 10$ and $d = 100$. Gaussian noise was then generated in all 100 dimensions. Similarly to the standard Gaussian mixture, the reduced dimensions provided more accurate results. This makes sense as the centers were created in the reduced dimension. An interesting difference between the standard Gaussian mixture and the mixture with a notion of lower dimensionality is that even with very distant cluster centers, there is still a sizeable error when analyzed in the larger dimensions. As evident in Figure 35, the point at which the distance between cluster centers overpowers any effect of dimension reduction is at a much higher distance between clusters when the data is generated in this fashion.

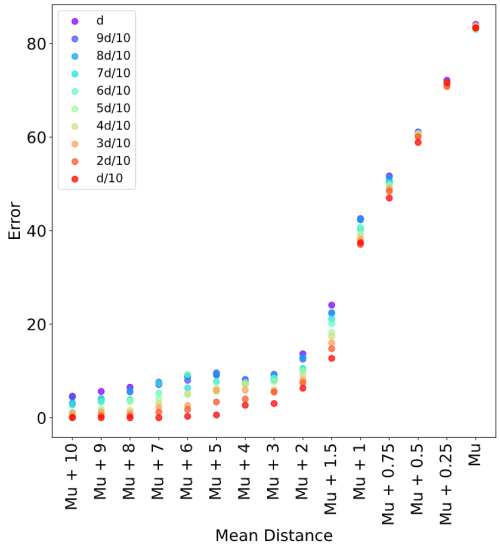


Figure 37: Clustering Error (%) of a Gaussian Mixture with $r = 25$

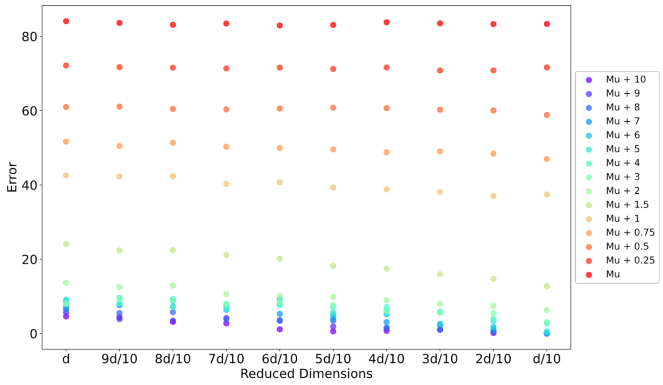


Figure 38: Clustering Error (%) of a Gaussian Mixture with $r = 25$

r = 25 Next, the same experiment was run, but this rime with $r = \frac{d}{4} = 25$ and $d = 100$. The results were very similar to those found when $r = 10$. The largest difference between $r = 10$ and $r = 25$ is that the point in which the closeness of cluster centers overpowers the reduction of dimension is much closer. It is still true that as dimension decreases, the error is much smaller for the further reduced dimensions. Similar to when $r = 10$, there is still a sizeable error for distant cluster means when analyzed in the higher dimensions.

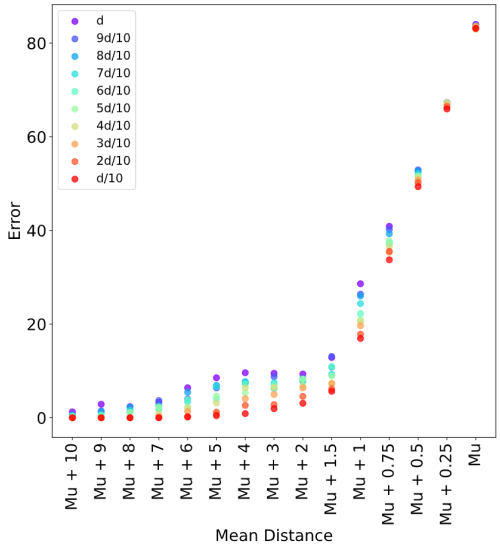


Figure 39: Clustering Error (%) of a Gaussian Mixture with $r = 50$

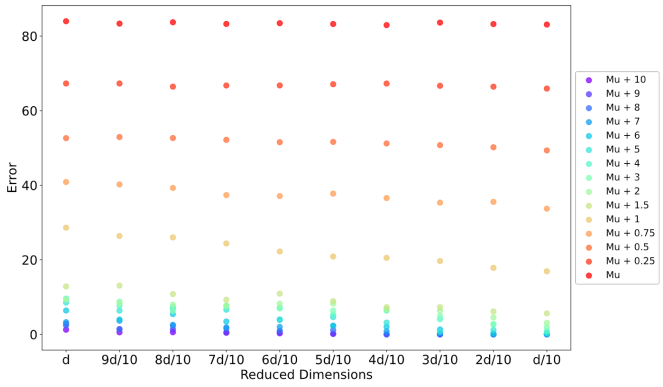


Figure 40: Clustering Error (%) of a Gaussian Mixture with $r = 50$

r = 50 As r is increased to 50, the results begin to even more closely resemble those of the simple Gaussian mixture. As shown in Figure 39, the further apart the cluster center distances become, the smaller the error. This was less evident with smaller values for r, but as r increases, this becomes more and more true. As is true in the simple Gaussian mixture, the distance in which the closeness of the cluster centers completely overpowers the importance of the reduced dimension is much smaller than for the smaller r values.

Conclusions Clustering and Dimensionality Reduction techniques have numerous multidisciplinary applications, making this work useful in many different fields. This study is ever-evolving and there are still numerous paths to explore. Next steps include continuing with generating synthetic data using many different techniques, as well as exploring the effects of running dimensionality reduction on the individual clusters of data. The work outlined in this report begins to scratch the surface of the future steps for this study.

6.2 Delta Fleet Clustering and Dimensionality Reduction

Clustering and Dimensionality Reduction techniques have numerous real-world applications. In an attempt to explore less conventional applications of these algorithms, I applied these techniques to aircraft data to see how different airplanes clustered together. All airplanes analyzed are used by Delta Airlines and the information is available on their main web-page [2]. Although the data is publicly available, many formatting and cleaning steps must be taken in order for it to be usable. Due to this, I found a similar study done in 2014 and was able to download their cleaned data set [15].

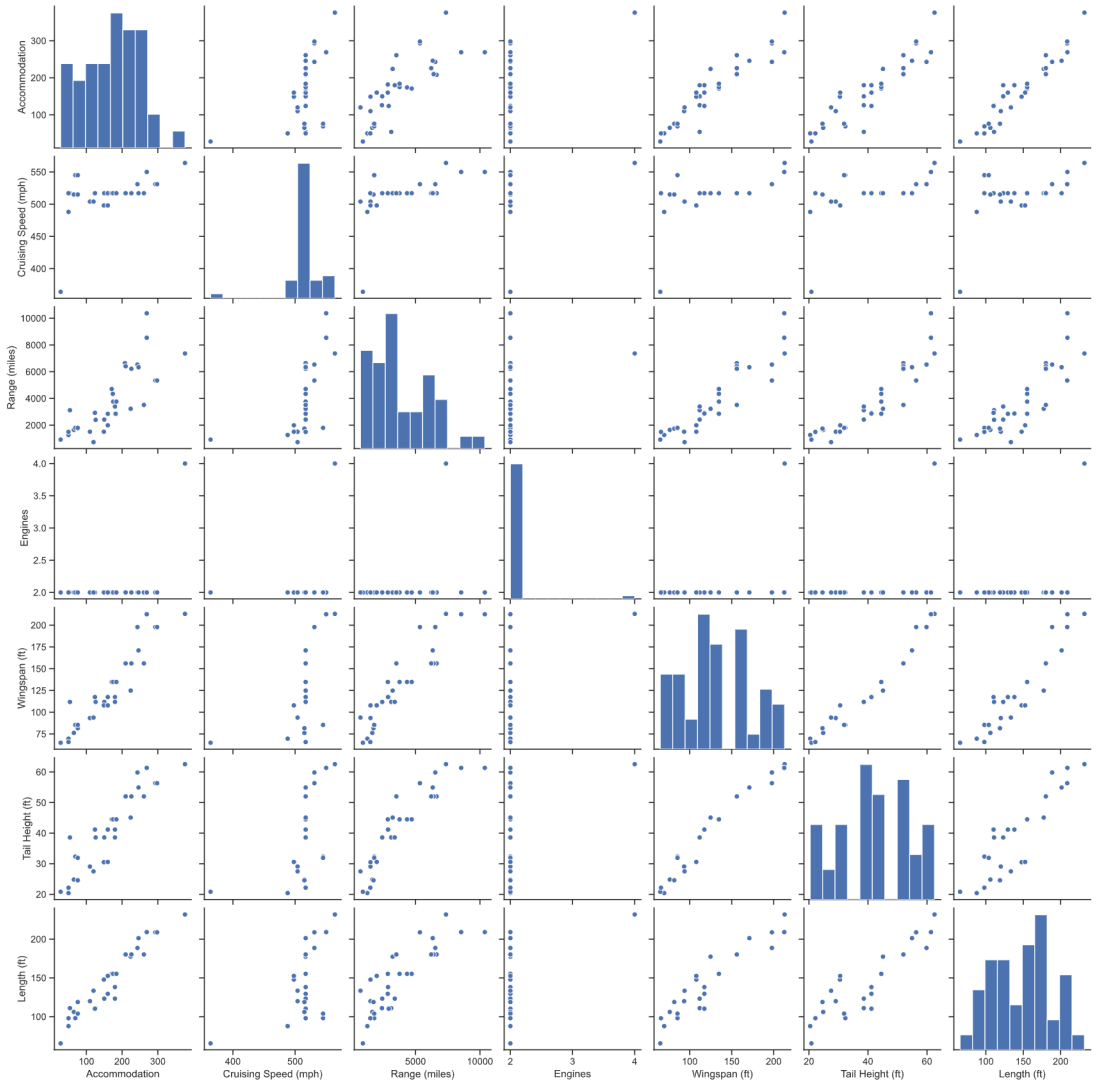
Introduction to the Data The data used in this study is available via the Delta Airlines Website. The airline makes public all information on each aircraft in use in their current fleet. When this data set was formulated in 2014, there were 48 different aircraft models in use. Currently, there are 33 different types of aircraft used by Delta Airlines. Each aircraft has 33 different features defining both qualitative and quantitative measures. Categorical variables were made into quantitative using a binary style format, assigning 1 for yes/true and 0 for no/false. Figure 41 shows a snapshot of the original csv-formatted data file.

Figure 41: Delta Fleet Data

Aircraft	Seat Width (Club)	Seat Pitch (Club)	Seat (Club)	Seat Width (First Class)	Seat Pitch (First Class)	Seats (First Class)	Seat Width (Business)	Seat Pitch (Business)	Seats (Business)	...	Video	Power	Satellite	Flat-bed	Sleeper	Club	First Class	Business	Eco Comfort	Economy
Airbus A319	0.0	0	0	21.0	36.0	12	0.0	0.0	0	...	0	0	0	0	0	0	1	0	1	1
Airbus A319 VIP	19.4	44	12	19.4	40.0	28	21.0	59.0	14	...	1	0	0	0	0	1	1	1	0	0
Airbus A320	0.0	0	0	21.0	36.0	12	0.0	0.0	0	...	0	0	0	0	0	0	1	0	1	1
Airbus A320 32-R	0.0	0	0	21.0	36.0	12	0.0	0.0	0	...	0	0	0	0	0	0	1	0	1	1
Airbus A330-200	0.0	0	0	0.0	0.0	0	21.0	60.0	32	...	1	1	0	1	0	0	0	1	1	1

Upon initial analysis of the data, it is clear to see that there are strong positive linear correlations among many different features (Figure 42). This is indicative that the data set is a strong candidate for dimensionality reduction. The data also needed to be scaled so that features, such as Range, did not overpower features with smaller units, such as Number of Engines.

Figure 42: Delta Fleet Feature Comparison Pre-PCA



Dimensionality Reduction To make the data more manageable, I applied PCA to the normalized data set. Based on the scree plot produced by the method, the first four principal components were used. By choosing these four dimensions, it is ensured that over 85% of the variance in the data is explained.

Upon plotting and visualizing the four-dimensional data, clear clusters begin to emerge. There are no longer any strong linear correlations between components, and therefore has reduced nicely. Although there are clear clusters appearing, there is a single data point acting as an outlier.

Figure 43: Principal Component Scree Plot

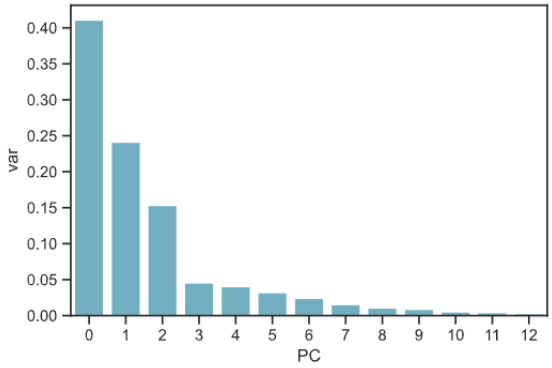
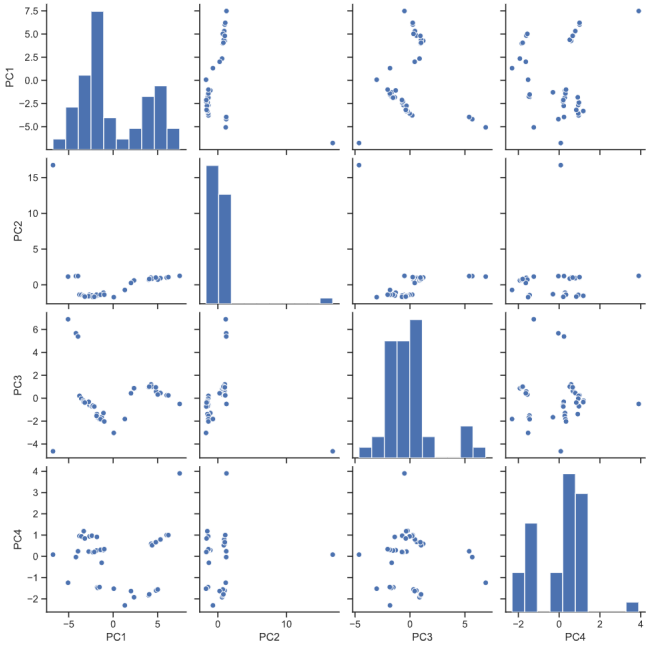


Figure 44: Visualization of Reduced Data Set



Clustering As clear clusters appeared post-dimensionality reduction, it was opportune to fit a clustering technique to the data. Due to this, I chose to apply k-means clustering to the reduced data set. At first glance, the elbow method did not provide a clear choice for the optimal number of k clusters, but k = 4 proved to be the strongest candidate. The clustering algorithm provided clusters nearly identical to the assumed clustering by eye.

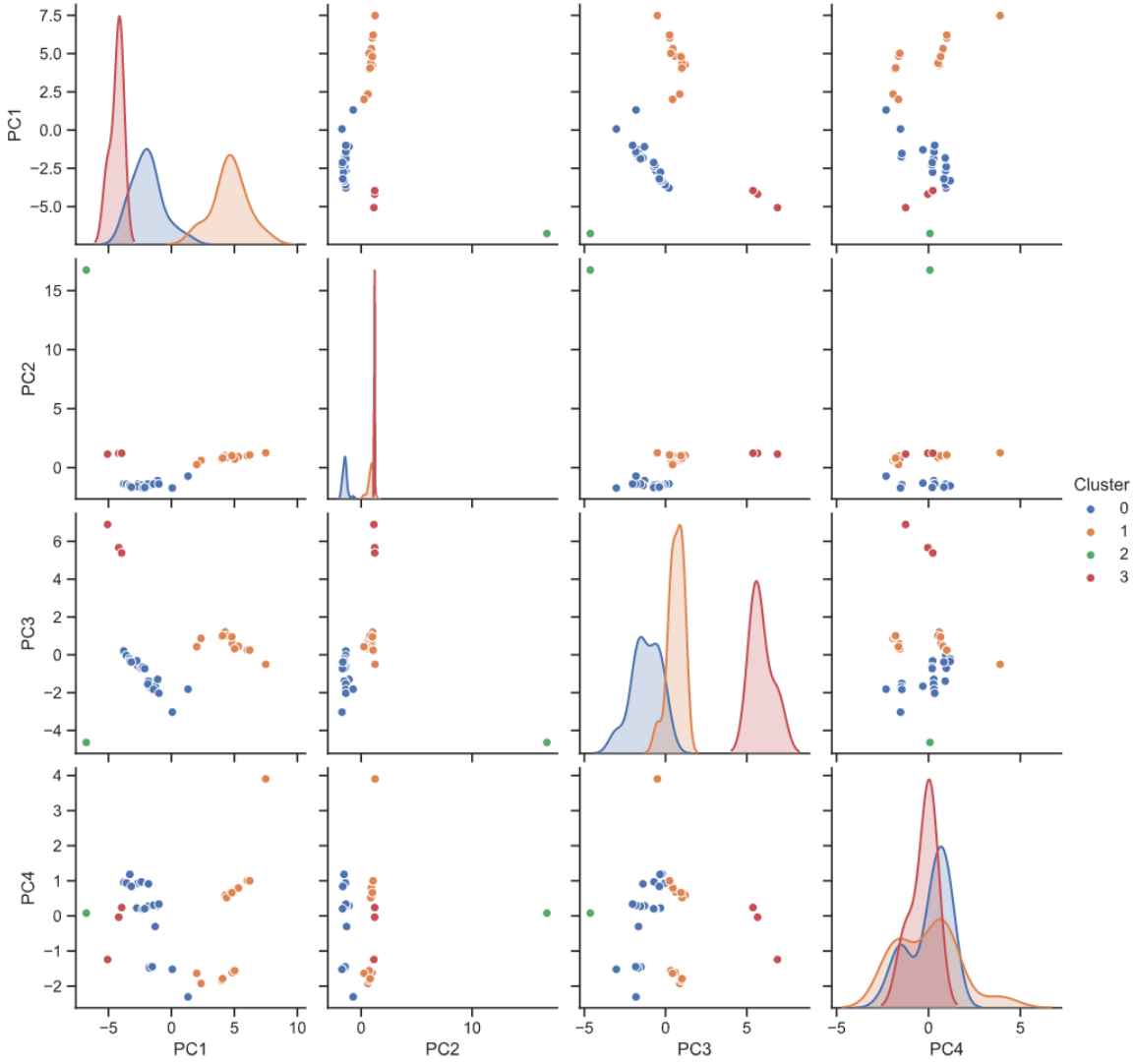
Findings: The K-means clustering algorithm divided the data into four distinct clusters, each with distinct sets of characteristics.

The most obvious cluster is cluster 2, as it contains only one aircraft. Upon further inspection, it is clear as to why this is the case. The Airbus A319 VIP is the sole member of its cluster and is also the sole charter plane listed in the Delta fleet. This aircraft is not a part of the standard Delta flight class but rather is rented privately. Unlike all other aircraft listed, the Airbus A319 VIP includes club seats surrounding tables, rather than the standard row after row of accommodation. This aircraft is exponentially different from all others utilized by Delta Airlines, and thus makes sense that it appeared in a cluster by itself.

The next smallest cluster is cluster number 3. This cluster contains the smaller aircraft employed by Delta Airlines such as the CRJ 100/200 Pinnacle/SkyWest, CRJ 100/200 ExpressJet, E120, and ERJ-145. One of the most distinguishing features of these aircraft is that they only contain economy seating, distinguishing them from the rest of the fleet. These airplanes are the smallest passenger planes and offer the most minimal accommodations.

The remaining two clusters contain the majority of the aircraft being analyzed. These planes are all much larger than the previously discussed aircraft and are also divided by size between the two clusters. Nearly identical in most features, the one distinctive difference between these clusters is the existence of First Class (Cluster 0) vs. Business Class (Cluster 1). There is a single point of crossover in which the Boeing 767-300 (76U), which has a Business Class cabin rather than

Figure 45: K-Means Clustering on Reduced Fleet Data



First-Class, is clustered in Cluster 0 with the First-Class containing aircraft. It is also clear that the overall size, including accommodation size and wingspan, of the aircraft in Cluster 1 is slightly larger than those in Cluster 0.

Conclusions Although analysis of the Delta Fleet does not bring about any monumental new information, it was interesting to explore less conventional applications of dimensionality reduction and clustering techniques. Overall, the methods very successfully classified similar aircraft into generally well-defined clusters. Within the year, Delta will be retiring at least three more aircraft models, two of which are quite small. As a next step, it would be interesting to see what new information and clustering comes out of running the algorithms on the smaller current fleet.

Figure 46: Delta Fleet Clusters

Cluster	Aircraft
0	Airbus A319
0	MD-88
0	E175
0	E170
0	CRJ 900
0	CRJ 700
0	Boeing 767-300 (76U)
0	Boeing 767-300 (76Q)
0	Boeing 767-300 (76P)
0	MD-90
0	Boeing 757-200 (75V)
0	Boeing 757-200 (757)
0	Boeing 757-200 (75N)
0	Boeing 757-200 (75M)
0	Boeing 757-300
0	MD-DC9-50
0	Airbus A320
0	Boeing 737-900ER (739)
0	Boeing 737-800 (73H)
0	Boeing 737-800 (738)
0	Airbus A320 32-R
0	Boeing 737-700 (73W)
0	Boeing 717
0	Boeing 757-200 (75A)
1	Airbus A330-200
1	Airbus A330-200 (3L2)
1	Boeing 777-200LR
1	Boeing 777-200ER
1	Boeing 767-400 (76D)
1	Boeing 757-200 (75E)
1	Airbus A330-200 (3L3)
1	Boeing 767-300 (76T)
1	Airbus A330-300
1	Boeing 767-300 (76L)
1	Boeing 767-300 (76G)
1	Boeing 757-200 (75X)
1	Boeing 747-400 (74S)
1	Boeing 767-300 (76Z V.1)
1	Boeing 767-300 (76Z V.2)
2	Airbus A319 VIP
3	CRJ 100/200 Pinnacle/SkyWest
3	CRJ 100/200 ExpressJet
3	E120
3	ERJ-145

7 Using Clustering and Dimensionality Reduction to Classify Countries and U.S. Counties by COVID-19 Impact

Veronica Alfaro

7.1 Introduction to Machine Learning Methods for COVID-19 Data

As the Coronavirus pandemic continues to evolve, novel approaches to face the challenges it presents appear every day in hopes of gaining insight into how to reduce the spread and impact of the virus. Machine learning methods have been used on both a local and global scale in order to assess which factors correlate most closely with the scope of the virus, as well as to classify locations by their response to COVID-19. These studies are done in hopes to provide a framework for predicting how COVID-19 and similar epidemics will affect specific areas, which can guide policy makers in preparing for outbreaks of these diseases to mitigate their devastation in these communities.

Over the past eight weeks, I have been investigating existing machine learning approaches to understanding COVID-19. Machine learning algorithms and analysis has been used to assess

COVID-19 impact on a variety of areas, including countries [32], as well as ZIP codes in New York City [21]. Clustering has also been used to determine which factors do, and do not, seem to to correlate with COVID-19, including temperature [33] and number of tests conducted [20], neither of which were found to be correlated with COVID-19 cases.

Prompted by this research, I decided to use clustering and dimensionality reduction techniques on COVID-19 data on both a global and national scale. In this chapter, I will replicate an existing study on global COVID-19 data, and adapt its methods for use on county-level data with different features. I plan to answer the following questions: How can dimensionality reduction and clustering be used to create clusters of U.S. counties based on factors believed to influence how COVID-19 affects an area? Can these clusters successfully classify counties by COVID-19 cases, deaths, and fatality rate?

7.2 Replication of PCA and K-means on Country-level Variables to Classify by COVID-19 Cases and Deaths

Overview of Original Research To better understand how clustering and dimensionality reduction have been used on COVID-19 data, I replicated the methods used in the peer reviewed research article "Using country-level variables to classify countries according to the number of confirmed COVID-19 cases: An unsupervised machine learning approach," [32] in which the authors clustered countries on 8 features which measured disease prevalence, air quality, socio-economic status, health system, and male population. They used Principal Component Analysis (PCA) and K-means clustering to form a model using 3 principal components which clustered 155 countries into 5 and 6 clusters based on these features. After comparing COVID-19 cases, deaths, death rate, and order of first reported case between these clusters with a one-way ANOVA test and t-tests, they found that they were able to classify countries based on confirmed COVID-19 cases, but not on COVID-19 deaths or case fatality rate.

Data Since the original researchers' dataset and Python code were publicly available, I was able to perform my replication using the same data and using their code as a reference. I downloaded their data and used it to perform PCA, clustering, and statistical tests in Python. The data included unnecessary variables; for example, the population of men, population of women, and proportion of men in a country were all included. In following the original research, I dropped the first two and left the third variable. I dropped all other features not included in the researchers' final dataset used for PCA and clustering. Figure 47 shows the first five rows of the final dataset, with eight features and 155 countries. Figure 48 shows the meaning of each feature in the dataset.

Figure 47: Country-level Variables for PCA and Clustering

Country	p.db	p.copd	p.hiv	p.tbc	GDP_2017	prop_men	pm2.5	UHC_index_2017
Afghanistan	0.105599	0.050140	0.000186	0.261146	556.302138	0.509613	53.2	37
Albania	0.044023	0.042848	0.000012	0.132001	4532.890162	0.502360	17.9	59
Algeria	0.085691	0.038137	0.000285	0.117467	4044.298372	0.506184	35.2	78
Angola	0.097936	0.030040	0.013423	0.222921	4095.812942	0.485228	27.9	40
Antigua and Barbuda	0.085484	0.020195	0.000960	0.189396	15383.415190	0.486567	17.9	73

Figure 48: Feature Key

Feature Name	Variable
p.db	Age-standardized prevalence of diabetes as of 2017
p.copd	Age-standardized prevalence of chronic obstructive disease [COPD] as of 2017
p.hiv	Age-standardized prevalence of HIV/AIDS as of 2017
p.tbc	Age-standardized prevalence of tuberculosis as of 2017
GDP_2017	Gross domestic product per capita as of 2017
prop_men	Proportion of males in country
pm2.5	Concentration of 2.5 particulate matter by country
UHC_index_2017	Universal health coverage index of service coverage as of 2017

Principal Component Analysis I first standardized the dataset, scaling the data so each feature vector had a mean of zero and unit variance. This accounts for discrepancies between the ranges of features. For example, the proportion of men in each country ranges from 0.459 (Latvia) to 0.75 (Qatar), with most values being around 0.50. GDP values, on the other hand, range from 309 to 107361. The significantly larger range of values for the latter would give it a disproportionately greater effect on the principal components when PCA is done. Standardizing the data before PCA eliminates this source of error. This was done using the preprocessing function from Scikit-Learn [31].

I performed PCA on the standardized data using the PCA algorithm in Scikit-Learn [31]. While the original research had prespecified 3 components, I chose to use 4. This choice was made by examining the cumulative explained variance for each number of principal components [49]. Using the Python package kneed [11], I determined that the knee point of the cumulative explained variance plot happened at 4 components. This number provides the largest dimensionality reduction (least number of components) with a sufficiently large variance. The percentage of explained variance for each component is shown in Figure 50. Together, these four components retained roughly 77.7% of the explained variance of the original set of features.

Figure 49: Cumulative Explained Variance of Principal Components

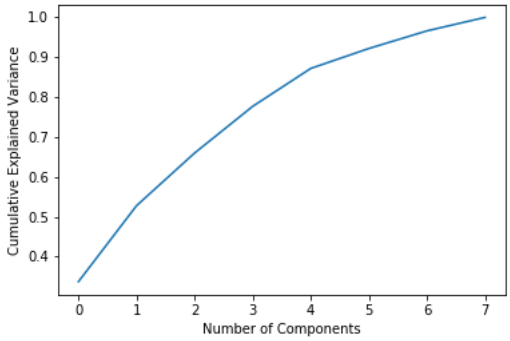


Figure 50: Percent of Explained Variance of Principal Components

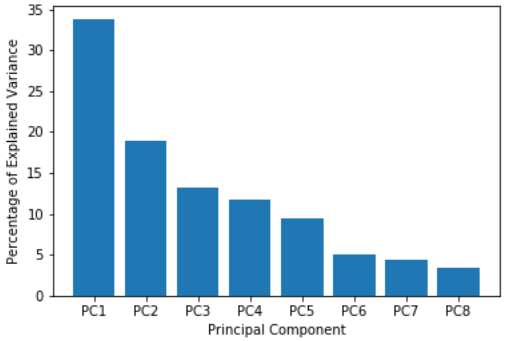


Figure 51 shows the data points plotted in two dimensions, against the first two principal components after PCA, and Figure 52 shows the same data points plotted in three dimensions, against the first three principal components, after PCA.

Figure 51: Data Plotted in 2 Dimensions

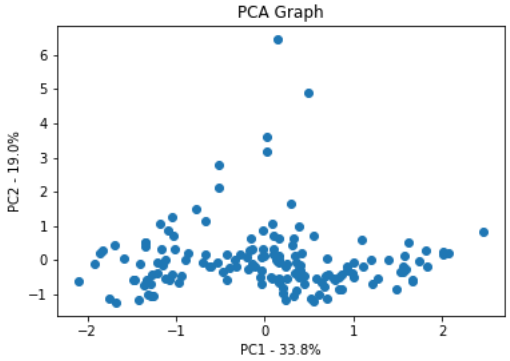
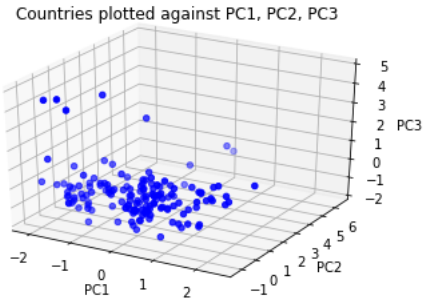
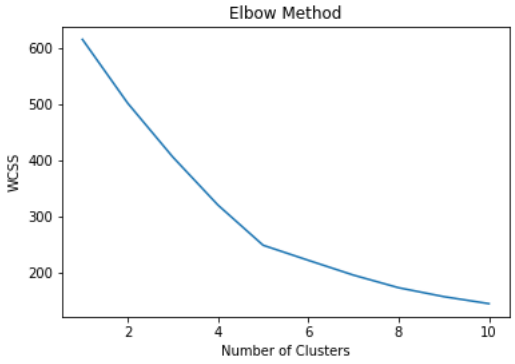


Figure 52: Data Plotted in 3 Dimensions



Choosing Number of Clusters I determined the number of clusters I would use for K-means clustering by plotting the within cluster sum of squares (WCSS), the sum of the squared distance from each observation to its cluster centroid [5], for each number of clusters used. A smaller WCSS indicates more compact clusters, that is, less variability of data points in each cluster. Thus, smaller WCSS values are desired. To choose our number of clusters, I plotted the WCSS for a range of numbers and choose the knee/elbow point of the plot. This is the point of maximum curvature, and in this case, the point at which the decrease in WCSS becomes sufficiently small. To find this value, I used the same kneed Python package [11] I used for choosing PCA components. The plot of WCSS values is shown in Figure 53. The knee/elbow point is 5, meaning 5 clusters gives us a sufficiently small WCSS. This agrees with the original research, in which both 5 and 6 clusters were chosen.

Figure 53: Within Cluster Sum of Squares for Different Values of K



K-means Clustering After the dimensionality of the original data was reduced to 4 principal components, I clustered with the K-means algorithm available from Scikit-Learn [31] using 500 iterations. This was done with both 5 clusters [54, 56] and 6 clusters [55, 57], and plotted against the first 2 and 3 principal components.

Figure 54: K-Means with 5 Clusters, 2D

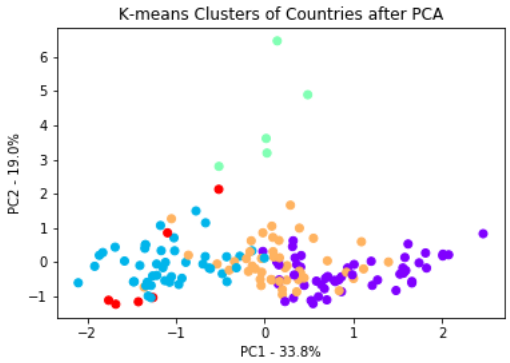


Figure 55: K-Means with 6 Clusters, 2D

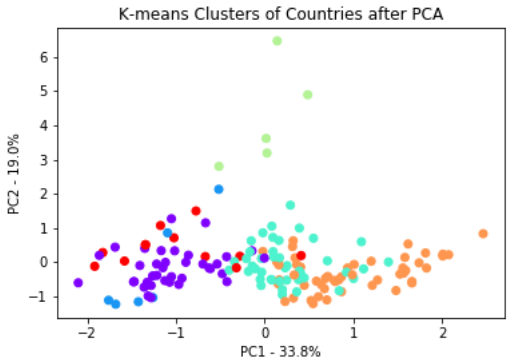


Figure 56: K-Means with 5 Clusters, 3D

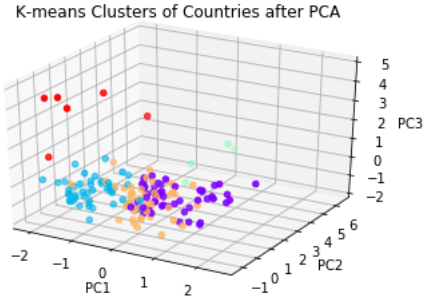
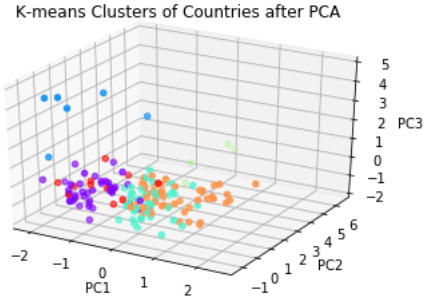


Figure 57: K-Means with 6 Clusters, 3D



Cluster Evaluation with Statistical Analysis To evaluate the quality of the clusters obtained from K-means in context, I compared COVID-19 features - number of cases, number of deaths, cases per 1 million population, and death rate in each country - across clusters. The original research, published in June, used COVID-19 data up to March 23rd. My replication used more recently updated COVID-19 data [8], up to August 26th.

I first compared COVID-19 features using a one-way ANOVA test, which determines whether there is a statistically significant difference in a feature between at least two clusters [4]. If a sufficiently small (< 0.05) p-value is obtained from the ANOVA test, this indicates that there is a statistically significant difference in the mean of the variable of interest between at least two clusters. The p-values (rounded to the nearest thousandth for clarity) from running the ANOVA test [10] on four COVID-19 features for both 5 and 6 cluster models is shown in Figure 58.

Figure 58: p-values for ANOVA tests on COVID-19 Variables between Clusters

	Total Cases	Total Deaths	Cases / 1 Million	Death Rate
5 Clusters	0.660	0.424	3.052e-13	0.052
6 Clusters	0.418	0.417	8.946e-13	0.043

From the results of the ANOVA test, it appears that there is no statistically significant difference in total cases or deaths between any two clusters. There is strong evidence of a statistically

significant difference in cases per 1 million population, and weaker, yet sufficient, evidence of a statistically significant difference in death rate between at least two clusters in the 6 cluster model.

Given these results, I decided to use Tukey's multiple comparison method [10] only on cases per 1 million population for both models, and on death rate for the 6 cluster model. This method computes pairwise comparisons, testing whether there is a statistically significant difference in a variable between all possible cluster pairs using $p < 0.05$ [13]. It also controls for errors which would normally arise with multiple comparisons. The p-value results for pairwise comparisons are shown in Figure 59 and 60 for cases per 1 million, with the 5 and 6 cluster models, respectively. For pairwise comparisons on death rate in the 6 cluster model, all p values were greater than 0.05.

Figure 59: P-value < 0.05 for Tukey Multiple Comparison Method on COVID-19 Cases / 1 Million Population on 5 Cluster Model Pairs

	cluster 0	cluster 1	cluster 2	cluster 3	cluster 4
cluster 0	-				
cluster 1	True	-			
cluster 2	True	True	-		
cluster 3	False	True	True	-	
cluster 4	False	False	True	False	-

Figure 60: P-value < 0.05 for Tukey Multiple Comparison Method on COVID-19 Cases / 1 Million Population on 6 Cluster Model Pairs

	cluster 0	cluster 1	cluster 2	cluster 3	cluster 4	cluster 5
cluster 0	-					
cluster 1	False	-				
cluster 2	True	False	-			
cluster 3	True	True	True	-		
cluster 4	True	False	False	True	-	
cluster 5	False	False	False	True	False	-

Results Our results from the ANOVA test indicate that there is strong evidence that COVID-19 cases per 1 Million population is significantly different between at least two clusters for both the 5 and 6 cluster models, since for both we have $p << 0.05$. There is no evidence that there are any cluster pairs with statistically significant differences in COVID-19 cases or deaths. Our results from the Tukey Multiple Comparison Method indicate that 6/10 cluster pairs in the 5 cluster model and 7/15 cluster pairs in the 6 cluster model have a statistically significant difference in COVID-19 cases per 1 million people. There is no evidence of any cluster pairs with a statistically significant difference in death rate for either model.

Strengths and Limitations The key strength of this research is that it developed a simple model using publicly accessible data, so it is can be replicated and adjusted fairly easily (as I did). However, it only used 8 predictors, so there was likely a substantial amount of valuable information about each country that was missing, which could have improved the clustering model. Additionally, COVID-19 cases and deaths are likely underrepresented, due to lack of testing and diagnosis. Although this issue is not unique to this study, it can still harm the accuracy of the results. COVID-19 has spread significantly since the original research was published, so as the virus continues to develop, clusters and classification results will likely keep changing.

Conclusions My replication produced different results from the original research. I was able to classify countries only in terms of COVID-19 cases per 1 million, while the original research was able to classify only by confirmed COVID-19 cases. This was likely due to a number of reasons, including using updated COVID-19 data for the statistical analysis, as well as using a different number of components for PCA.

From this work, I was able to develop a clustering model using readily available data that was able to stratify countries by number of COVID-19 cases per 1 million residents. These results, as with the original research, provide a preliminary model which could help identify countries that may be especially vulnerable or resistant to the effects of COVID-19.

7.3 Clustering and Dimensionality Reduction on U.S. County-level Variables to Classify Counties by COVID-19 Cases and Deaths

Overview Based on the potential applications that I saw in the research I replicated in 7.2, I wanted to continue to investigate this method of using clustering and PCA together to create clusters and determine whether they could be used to make more accurate predictions of how COVID-19, or future pandemics, will affect a specific area.

Because of the broad availability of county-level data, both COVID-19 related and otherwise, I chose to cluster U.S. counties based on both existing conditions and local and state efforts to limit the spread of COVID-19. In this section I will outline my process of data collection, dimensionality reduction, clustering, and cluster evaluation. I aim to create clusters which can classify U.S. counties by COVID-19 cases, deaths, and/or fatality rate.

Data I utilized existing COVID-19 literature to decide which county-level variables to include in my database. Motivated by a New York Times article about COVID-19 in areas of Michigan exposed to high levels or air pollution [34], and a study confirming the correlation between long-term exposure to PM2.5 and increased COVID-19 death rate [36], I added measures of air quality and PM 2.5 concentration to my dataset. Due to COVID-19's disproportionate effect on the elderly [7], males [17], and Black and Latino populations [18], I added factors for age, gender, and race and ethnicity to my dataset. I also added measures that directly impact the spread of the virus, such as mask usage [27], and restrictive policies put into place to curb the spread of the disease. Healthcare infrastructure was considered by using number of active physicians, ICU beds, and hospitals in each county. Since large U.S. cities such as New York City have become epicenters of the virus [29], I added features for population and housing density. In total, I gathered 25 features to use for PCA and clustering on U.S. counties.

Because my data was pulled from a number of sources, there were some modifications and cleaning that had to be done before it could be used. I combined and consolidated data using the Python pandas merge and groupby functions [30]. Mask usage data was originally separated into 5 categories, each category containing the estimated proportion of people in a county who wore a mask in a given frequency level. I consolidated this into one feature vector by transforming the 5 categorical features - "never," "rarely," "sometimes," "frequently," and "always" wears a mask - into numerical values, 0-4. Then, using the proportions for each category, I created a weighted average that measures mask usage for each county. Some features, such as age and race, were given as populations. I divided these values by county population to obtain proportions, which I was more interested in as a feature. Additionally, the data on restrictions on public gatherings and travel was originally given by date the restriction was put into place. I changed these features to days between the county's first reported COVID-19 case and restrictions put into place. This measures how quickly officials acted in response to cases in their own counties.

The first 5 rows and 13 features of the dataset can be seen in Figure 61. Below that are descriptions of the 25 features included, as well as their source.

Figure 61: U.S. County-level Data

state	county	AQI Median	# Days PM2.5	avg mask use	2018 pop	median hh income 2018	p.male	pop density per sq.mi	housing density per sq.mi	p.0-17	p.18-64	...	public schools	restaurant dine-in	entertainment/gym
Alabama	Baldwin	37.0	53	2.968	218022	57588.0	0.484616	114.6	65.5	0.216079	0.579487	...	2.0	5.0	14.0
	Clay	30.0	107	2.371	13275	39201.0	0.489040	23.1	11.2	0.202486	0.590282	...	-9.0	-6.0	3.0
	Colbert	37.0	36	2.917	54762	49055.0	0.479511	91.8	43.5	0.207626	0.593075	...	-9.0	-6.0	3.0
	Elmore	39.0	-999	3.045	81887	60367.0	0.483410	128.2	52.8	0.222392	0.626180	...	3.0	6.0	15.0
	Etowah	43.0	64	2.829	102501	44903.0	0.484415	195.2	88.7	0.214603	0.595282	...	-9.0	-6.0	3.0

Feature	**Source**
Air Quality Index Median for 2019	[6]
Number of Days in 2019 with PM 2.5 as Main Pollutant	[6]
Average Mask Usage	[19]
Total Population in 2018	[22]
Median Household Income in 2018	[22]
Proportion of Males in Total Population	[22]
Population Density per Square Mile	[22]
Housing Density per Square Mile	[22]
Proportion of Population 0-17 years old	[22]
Proportion of Population 18-64 years old	[22]
Proportion of Population 65+ years old	[22]
Proportion of Population 85+ years old	[22]
Days between first COVID-19 Case and Stay at Home Order	[22] [35]
Days between first COVID-19 Case and >50 Person Gatherings Restricted	[22] [35]
Days between first COVID-19 Case and >500 Person Gatherings Restricted	[22] [35]
Days between first COVID-19 Case and Public Schools Closed	[22] [35]
Days between first COVID-19 Case and Dine-in Restaurants Closed	[22] [35]
Days between first COVID-19 Case and Entertainment and Gyms Closed	[22] [35]
Days between first COVID-19 Case and Federal Guidelines Enacted	[22] [35]
Days between first COVID-19 Case and Foreign Travel Ban	[22] [35]
Proportion of Population that is Black	[22]
Proportion of Population that is Hispanic/Latino	[22]
Number of Active Physicians per 100000 People	[22]
Total Hospitals	[22]
ICU Beds per County	[22]

Principal Component Analysis Using the same PCA methods used in section 7.2, I started by standardizing my data. Next, I chose my number of components by finding the knee of the cumulative explained variance vs. number of components plot [Figure 62]. In doing so, I chose to use 9 principal components in PCA. I then performed PCA using the algorithm from Scikit-learn [31]. The percentage of explained variance of all 25 possible principal components is shown in Figure 63. I then plotted the data against the first two principal components [Figure 64] for visualization.

Figure 62: Choosing Number of Components for PCA

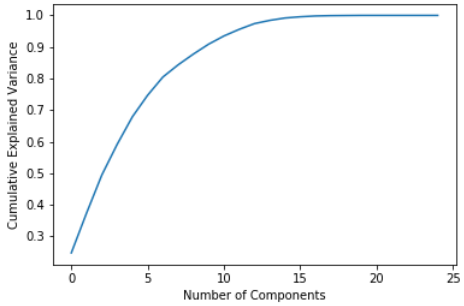


Figure 63: Percentage of Explained Variance for Each Principal Component

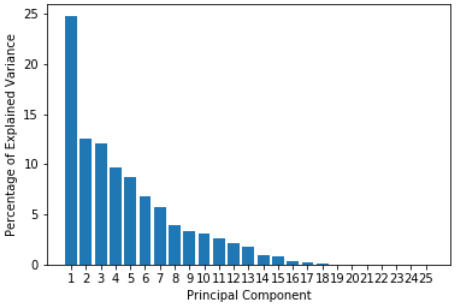
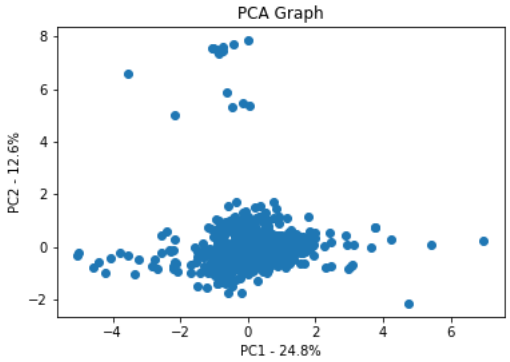
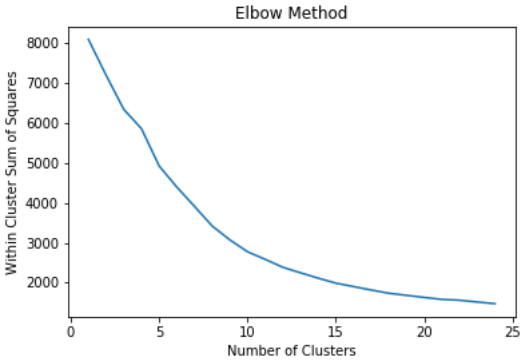


Figure 64: Plotting Data against PC1 and PC2



Choosing Number of Clusters Using the same methods as in section 7.2, I chose my number of clusters based on the knee point of the plot showing the within cluster sum of squares (WCSS) for each number of clusters [Figure 65]. That is, I chose the number of clusters at which adding any more would not significantly decrease the WCSS. This resulted in using 10 clusters.

Figure 65: Choosing Number of Clusters with WCSS



Clustering Using $k = 10$ as chosen above, I clustered my data in 6 different ways. I used three different clustering algorithms: K-means, Agglomerative, and Gaussian Mixture Model [12], all

using Scikit-Learn [31]. I ran each of these algorithms twice - once on the original (standardized) dataset, and once on the data post-PCA, that is, on 9 components obtained by reducing the dimensionality of my original data. The resulting clusters are visualized in two dimensions, against the first two principal components, in Figures 66, 67, 68, 69, 70, and 71.

Figure 66: K-means before PCA

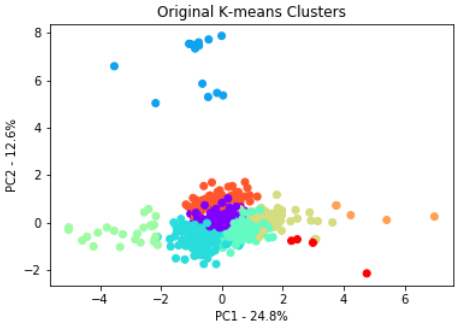


Figure 67: K-means after PCA

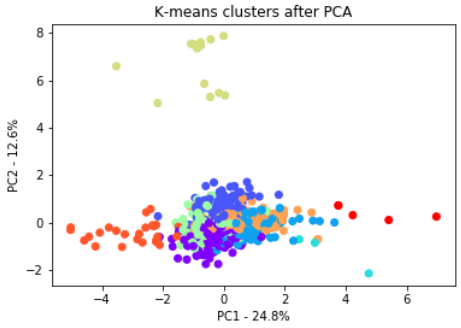


Figure 68: Agglomerative before PCA

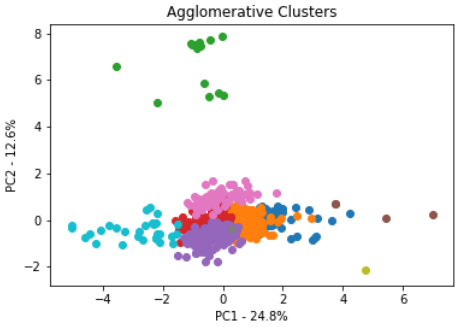


Figure 69: Agglomerative after PCA

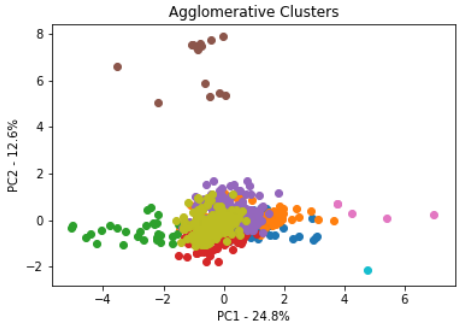


Figure 70: Gaussian Mixture Model before PCA

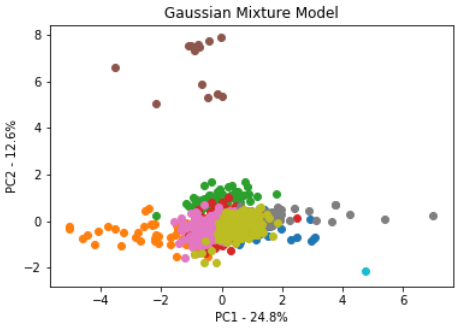
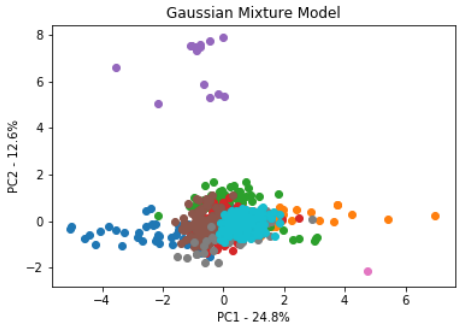


Figure 71: Gaussian Mixture Model after PCA



Clustering Error Next, to determine which clustering algorithm was most resistant to change after dimensionality reduction with PCA, I determined the clustering error between each clustering algorithm's clusters before and after PCA was performed. Clustering error calculates the proportion of data points which are clustered differently when comparing two clustering algorithms [26]. Using the Python module munkres to permute clusters [37], I found the clustering error for each clustering

method; my results are shown in Figure 72. The clustering errors for all three methods, before and after PCA, are similar. While they are larger than I had expected, they still indicate that the majority of data points remain in the same cluster, even after PCA is applied.

Figure 72: Clustering Error

Clustering Method	**Clustering Error from Before & After PCA**
K-Means	47.17%
Agglomerative	47.61%
Gaussian Mixed Model	45.39%

One-way ANOVA Test I used ANOVA statistical tests to determine if there was a statistically significant difference in three variables - number of COVID-19 cases, number of COVID-19 deaths, and COVID-19 death rate - between any pair of clusters in each of my six clustering models. I obtained a very small p-value, less than 0.001, for each variable in each model. This indicates that for each clustering model, there is a statistically significant difference in each variable between at least two clusters. This motivated me to investigate this further with pairwise comparisons.

Tukey's Multiple Comparison Method for Pairwise Comparisons I used Tukey's Multiple Comparison Method on each COVID-19 variable for each clustering model. To determine how well each clustering method was able to classify U.S. counties by COVID-19 variables, I looked at how many pairs of clusters had a statistically significant difference in each variable. I then divided that by the number of total possible cluster pairs to obtain the percentage of cluster pairs with a statistically significant difference in a given variable. Results are shown in the table in Figure 73.

Figure 73: Percentage of Cluster Pairs with Significant Differences in COVID-19 Variables

	COVID-19 Cases	COVID-19 Deaths	COVID-19 Fatality Rate
K-means before PCA	55.55%	40.00%	31.11%
K-means after PCA	51.11%	44.44%	33.33%
Agglomerative before PCA	51.11%	46.67%	15.56%
Agglomerative after PCA	51.11%	44.44%	26.67%
Gaussian Mixed Model before PCA	48.89%	33.33%	40%
Gaussian Mixed Model after PCA	62.22%	44.44%	22.22%

From the table we can see that the Gaussian Mixed Model after PCA has the largest percent of cluster pairs with significant differences in COVID-19 cases. This is also the largest percent of cluster pairs with differences in any of the three variables. For COVID-19 deaths, Agglomerative Clustering before PCA had the largest percent of cluster pairs with significant differences. For fatality rate, Gaussian Mixed Model before PCA had the largest percent of cluster pairs with significant differences. All 6 clustering methods had more cluster pairs with significant differences in COVID-19 cases than with deaths or fatality rate.

Results and Discussion Overall, this study has shown the application of clustering and dimensionality reduction to COVID-19 data on a county level. By using three different clustering algorithms before and after PCA, I have successfully shown that it is possible to cluster U.S. counties on the basis of widely available county-level data.

The usefulness of these clusters was examined using clustering error, ANOVA tests, and pairwise comparisons with Tukey's Multiple Comparison Method.

Clustering error gives us information about how resistant a specific clustering algorithm is to dimensionality reduction of the data. There was not a substantial difference in clustering errors for the three methods we applied it to, so we cannot confidently say that one method retained its clusters after PCA better or worse than either of the others. The size of the clustering errors, just under 50%, implies that, while there was a notable change in clusters due to PCA, the error was much smaller than we would have expected for 10 random clusters. These clustering errors could likely be decreased by using more components in PCA to retain more variance. However, using too many components defeats the original purpose of dimensionality reduction - so this is always a trade off.

The results of the ANOVA test demonstrate some level of success in classifying U.S. counties by COVID-19 variables. Since all ANOVA tests that we ran gave extremely small p-values, we know clustering U.S. counties using the given methods results in at least two clusters with statistically significant differences for each variable we tested. Why is this important? Suppose county x does not yet have many cases of COVID-19, and we want to have a better idea of how COVID-19 is expected to affect it. Based on county x's existing demographics and general information, along with their initial social distancing restrictions, we can cluster county x with similar counties. Then, assuming COVID-19 has already spread through these other counties, we can make predictions of county x's expected cases, deaths, and fatality rate using information from its cluster. These predictions will likely be more accurate than if we used nationwide data, because, as we have seen with the ANOVA results, there are statistically significant differences in COVID-19 cases, deaths, and fatality rate between clusters of counties in the U.S.

The pairwise comparisons made with Tukey's Multiple Comparison Method confirm the success shown by the ANOVA test results. With these results, we can see that for all clustering models, there is a statistically significant difference in COVID-19 cases between the majority of cluster pairs. This strengthens our evidence that using clustering and dimensionality reduction can create clusters that (imperfectly) classify counties by COVID-19 cases.

Strengths, Limitations, and Sources of Error The key strength of this study is the potential it has to be applied in ways that will help us better understand and predict how COVID-19 affects specific areas. Using clustering and dimensionality reduction on publicly available data can create clusters which can give specific information about how COVID-19 will affect a county. This can give experts the information they need to guide policy makers in deciding what is best for the public health of their community.

Despite these exciting possibilities, there is still much work to be done before fully trusting and utilizing the results of this model. First, since new information about how COVID-19 spreads and affects the body is constantly arising, the features used in our dataset could be improved. Adding more features, such as proportion of population with related diseases, could likely improve the model. Additionally, missing data likely had a negative effect on our clustering. Namely, some missing data was filled with a -999 value. This was done so that it would seem like an "outlier" to the rest of the feature vector, and not create false clusters. However, this has an adverse effect when there is a large amount of missing data. If this happens, the -999s may cause unrelated

data points to cluster together based on this shared value. Alternatives include making predictions for missing values, or using random values to prevent this issue. Another issue that may have negatively affected the clustering was outliers. Cities such as New York City and Miami have had extremely high numbers of COVID-19 cases. Outlying data such as this makes it harder to make accurate predictions of COVID-19 on the basis of existing factors. Finally, like the previous study, underreported numbers of COVID-19 cases and deaths also hurts the accuracy of the model.

Conclusions This research has shown the relevant applications of clustering and dimensionality reduction to the COVID-19 pandemic. Although there are still improvements to be made to the model proposed by this research, its initial results suggest that it is possible to classify U.S. counties by COVID-19 features by clustering them on existing factors. In our model specifically, we were best able to classify by COVID-19 cases. There is a wide range of applicability to this model, both in the current pandemic and in the future.

References

[1] Alzheimer's disease neuroimaging initiative.

[2] Delta aircraft seat maps, specs & amenities. *Delta Aircraft Seat Maps, Specs & Amenities : Delta Air Lines*.

[3] Oasis brains.

[4] One-way anova. *Laerd Statistics*, 2018.

[5] Interpret all statistics and graphs for cluster k-means. *Minitab*, 2019.

[6] Air quality index report, 2020.

[7] Older adults. *Centers for Disease Control and Prevention*, 2020.

[8] Reported cases and deaths by country, territory, or conveyance, 2020.

[9] Hany Alashwal, Mohamed El Halaby, Jacob J. Crouse, Areeg Abdalla, and Ahmed A. Moustafa. The application of unsupervised clustering methods to alzheimer's disease. *Frontiers in Computational Neuroscience*, 13.

[10] Michael Allen. Statistics: Multi-comparison with tukey's test and the holm-bonferroni method. *Python for healthcare modelling and data science*, 2018.

[11] Kevin Arvai. Knee/elbow point detection. *kaggle*, 2019.

[12] Jason Brownlee. 10 clustering algorithms with python. *Machine Learning Mastery*, 2020.

[13] Nicola Crichton. Tukey multiple comparison test. *Journal of Clinical Nursing*, 1999.

[14] Dragan Gamberger, Bernard Ženko, Alexis Mitelpunkt, and Nada Lavrač. Homogeneous clusters of alzheimer's disease patient population. *BioMedical Engineering OnLine*.

[15] Myles Harrison. mylesmharrison/delta_pca_kmeans. *GitHub*.

[16] Anil K Jain, M Narasimha Murty, and Patrick J Flynn. Data clustering: a review. *ACM computing surveys (CSUR)*, 31(3):264–323, 1999.

[17] Jian-Min Jin, Peng Bai, Wei He, Fei Wu, Xiao-Fang Liu, De-Min Han, Shi Liu, and Jin-Kui Yang. Gender differences in patients with covid-19: Focus on severity and mortality. *Frontiers in Public Health*, 2020.

[18] Richard A. Oppel Jr., Robert Gebeloff, K.K. Rebecca Lai, Will Wright, and Mitch Smith. The fullest look yet at the racial inequity of coronavirus. *The New York Times*, 2020.

[19] Josh Katz, Margot Sanger-Katz, and Kevin Quealy. Estimates from the new york times, based on roughly 250,000 interviews conducted by dynata from july 2 to july 14., 2020.

[20] Hasinur Rahaman Khan and Ahmed Hossain. Countries are clustered but number of tests is not vital to predict global covid-19 confirmed cases: A machine learning approach. *medRxiv*.

[21] Fadoua Khmaissia, Pegah Sagheb Haghighi, Aarthe Jayaprakash, Zhenwei Wu, Sokratis Papadopoulos, Yuan Lai, and Freddy T. Nguyen. An unsupervised machine learning approach to assess the zip code level impact of covid-19 in nyc. *Cornell University*.

[22] Benjamin D. Killeen, Jie Ying Wu, Kinjal Shah, Anna Zapaishchykova, Philipp Nikutta, Aniruddha Tamhane, Shreya Chakraborty, Jinchi Wei, Tiger Gao, Mareike Thies, and Mathias Unberath. A County-level Dataset for Informing the United States' Response to COVID-19. April 2020.

[23] James Larlow. Usage patterns of dublin bikes stations. *towards data science*, 2017.

[24] Jake Lever, Martin Krzywinski, and Naomi Altman. Principal component analysis. *Nature Methods*, 14(7):641–642, 2017.

[25] John Lipor. personal communication.

[26] John Lipor and Laura Balzano. Clustering quality metrics for subspace clustering. *Pattern Recognition*, page 107328, 2020.

[27] Wei Lyu and George L. Wehby. Community use of face masks and covid-19: Evidence from a natural experiment of state mandates in the us. *Health Affairs*.

[28] Leland McInnes, John Healy, and James Melville. Umap: Uniform manifold approximation and projection for dimension reduction. *arXiv preprint arXiv:1802.03426*, 2018.

[29] Jesse McKinley. New york city region is now an epicenter of the coronavirus pandemic. *The New York Times*.

[30] Wes McKinney and the Pandas Development Team. Api reference. *pandas: powerful Python data analysis toolkit*.

[31] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[32] Carrillo-Larco RM and Castillo-Cara M. Using country-level variables to classify countries according to the number of confirmed covid-19 cases: An unsupervised machine learning approach [version 3; peer review: 2 approved]. *Wellcome Open Research*, 2020.

[33] Mohammad Khubeb Siddiqui, Ruben Morales-Menendez, Pradeep Gupta, Hafiz Iqbal, Fida Hussain, Khudeja Khatoon, and Sultan Ahmad. Correlation between temperature and covid-19 (suspected, confirmed and death) cases based on machine learning analysis. *Journal of Pure and Applied Microbiology*, 14, 04 2020.

[34] Hiroko Tabuchi. In the shadow of america's smokestacks, virus is one more deadly risk. *The New York Times*, 2020.

[35] The New York Times. covid-19-data, 2020.

[36] Xiao Wu, Rachel C Nethery, M Benjamin Sabath, Danielle Braun, and Francesca Dominici. Exposure to air pollution and covid-19 mortality in the united states: A nationwide cross-sectional study. *medRxiv*, 2020.

[37] Pengfei Zhu, Binyuan Hui, Changqing Zhang, Dawei Du, Longyin Wen, and Qinghua Hu. Multi-view deep subspace clustering networks. *ArXiv: 1908.01978*.