Kathryn Van Etten-	Data Sanitization
Marc Angelo Acebedo-	Programming
Serena Bonci-	Report Write-Up

Dataset information

For our dataset, Kathryn chose to look into Movehub.com, a website that aims to make the process of moving less stressful. A Kaggle user named Blitzer uploaded this data with a table of city names along with their countries corresponding to Wikipedia's list of cities with populations over 100,000. Blitzer narrowed down the data to the areas that had populations over 100,000 and also had data on Movehub.com. This dataset focused on purchasing power, healthcare and corresponding cost of living represented by the cost of various goods such as a movie ticket, cappuccino, wine, and more. The data was relatively straightforward and clean, but Kathryn decided to narrow it down to join the cost of living and quality of life tables.

Kathrynn's idea is that one would be able to locate clusters in certain types of cities, such as ones that have cheap food and low crime rate, but high rent. Ideally, users would be able to filter characteristics of cities that they like or dislike in order to find trends. This would ultimately lead the user to find cities that would fit their criteria more easily. For the purposes of testing the efficacy of this approach with our algorithms, we decided to plot the cost of a cappuccino in various cities to the average purchase power of a city.

Algorithm Implementation-

Python was used to code K-Means, K-Medoids, and H-Clustering. For K-Means and K-Medoids, we chose a k of 2, because by looking at the initial visualization of data points as a scatterplot, it appeared that there were two large clusters with a few data points in between them. We tested a k of 2, and it served as evidence towards our hypothesis. When we tried k>2, we came to the conclusion that the visualization looked too cluttered and hard to understand. In terms of H-Clustering, we decided to use a dendrogram for our visualization by using matplotlib.

Results-

We found that K-Means is slower but more accurate in terms of finding and defining clusters, whereas K-Medoids is faster but less accurate. K-Medoids is less accurate because it splits the second cluster (in blue) down its left half, causing for those data points to be included in the first cluster (in green). Visually, this makes sense considering that the points clustered around the medoid point for each point, but this type of clustering visualization did not reflect our aim in depicting the two dense collections of points from x=20 to x=40 and x=50 to x=90 as separate clusters. On the contrary, K-Means did well to cluster the points from x=0 to x=40 and x=50 to x=90 separately, as K-Means is more mathematically suitable for cluster visualization based on density. H-Clustering was the slowest of the three because it does not work well with large datasets, and our dataset included 216 points--the H-Clustering dendrogram was largely ambiguous

Takeaways-

For our data, in particular, K-Means might be the best approach because the data visualizations provided with it are comparatively much more legible, and while it does work more slowly compared to K-Medoids, it is far more accurate and more understandable due to the simplicity of using a scatterplot. While the dendrograms used in our H-Clustering algorithm are extremely detailed and make a point of where each cluster joins one another, they do not work well with our dataset in particular because the visualizations created with them are extremely cluttered and illegible. Despite this, for smaller datasets, H-Clustering could be a viable alternative.