Performance Assessment D212 – Data Mining II  
Task I

Doug Haunsperger

College of IT, Western Governors University

March 15, 2024

# Part I. Rese**arch Questi**on

## A1. Question Proposal

I propose to research the question, “Can our patients be clustered effectively into a reasonable number of groups based on demographic and service data?”

## A2. Goals

The goal of this analysis would be to cluster our patients to identify non-obvious similarities between them. Since clustering is an unsupervised learning method, the goal is not in this phase to correlate those clusters with outcomes such as readmission, but that would be a logical next step once the clusters are identified.

# Part II. Technique Justification

## B1. Explanation of Clustering Method

Clustering methods “seek to segment the entire data set into relatively homogeneous subgroups… where the similarity of the records within the cluster is maximized and the similarity to records outside this cluster is minimized.” (Larose & Larose, 2019, p. 141). The *k-means* clustering algorithm “divides the data into *k* clusters by minimizing the sum of the squared distances of each record to the *mean* of its assigned cluster.” (Bruce, Bruce & Gedeck, 2020, p. 295).

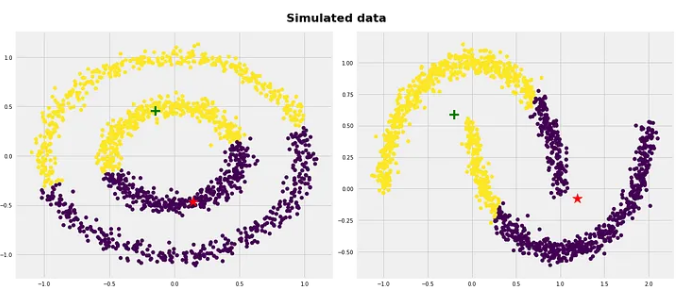
The value of *k* is not predetermined, so it is necessary to run *k-means* several times with different numbers of clusters and determine the “best” value of *k.* Bruce, Bruce & Gedeck state that “[i]n the absence of a cluster number dictated by practical or managerial considerations, a … common approach, called the elbow method, is to identify when the set of clusters explains ‘most’ of the variance in the data.” (2020, p.302).

Another method, given by the scikit-learn authors, is silhouette analysis. A silhouette plot “displays a measure of how close each point in one cluster is to points in the neighboring clusters and thus provides a way to assess parameters like number of clusters visually.” (scikit-learn.org, n.d.)

## B2. Method Assumptions

The *k*-means algorithm assumes, among other things, that the data forms relatively compact clusters. Data that has “complicated geometric shapes” or clusters that interpenetrate each other will not be clustered correctly by *k-*means. (Dabbura, 2018).

Figure 1  
*Incorrect clustering by k-means on oddly shaped data (Dabbura, 2018)*

**

## B3. Package / Library List

Figure 2  
*Package and library import statements* A screen shot of a computer code

Description automatically generated

Figure 2 shows my import code. Each package or library imported is useful in the full scope of the project. Pandas is used to work with DataFrames. Numpy is used for various mathematical functions (finding means, covariance matrices, eigenvalues, etc.). Pyplot and cm (colormap) are used for graphical display.

The remaining lines import certain functions or classes from the scikit-learn package. Following the API reference (2024), StandardScaler is used to scale the quantitative variables to unit variance with zero mean. I used the PCA class for principal component analysis to reduce the dimensionality of the dataset because like many algorithms, *k-*means suffers from the curse of dimensionality (Needham, 2016). The KMeans class is used to actually run the *k-*means algorithm. Finally, the silhouette\_samples and silhouette\_score classes are used to evaluate the *k-*means cluster fit.

# Part III. Data Preparation

## C1. Data Preprocessing

As mentioned in the previous section, I performed PCA to reduce the dimensionality of the dataset in order to have the best chance of a well-performing *k-*means clustering model. After examining the eigenvalues and explained variance given by each principal component, I chose to retain attempt *k-*means with both 2 and 3 PCs. Two principal components only explain ~53% of the variance in the data set but is easier to visualize. Increasing to three PCs explains about 68% of the variance in the data set but decreases the performance of *k-*means and is harder to visualize the cluster grouping.

## C2. Data Set Variables

I have chosen to do the clustering analysis on a set of 7 variables, each quantitative: Population, Age, Income, VitD\_levels, Initial\_days, TotalCharge, and Additional\_charges. Population and Income underwent a log transformation to make their distributions closer to normal. The rubric asks us to categorize these variables as continuous or categorical, which leaves out the possibility of discrete data. Technically, since both Population and Age take only integer values, they are properly categorized as discrete, but as they have a wide range of possible values, they are functionally continuous. The other five variables all take floating point values and are properly continuous.

## C3. Preparation Steps for Analysis

See attached Jupyter notebook, ‘D212\_PA1.ipynb’ under the heading ‘Data Preparation’. I have provided headers to each code block indicating the preparation steps.

## C4. Cleaned Data Set

The cleaned data set (prior to dimension reduction) is attached as ‘clean\_medical\_data.csv’.

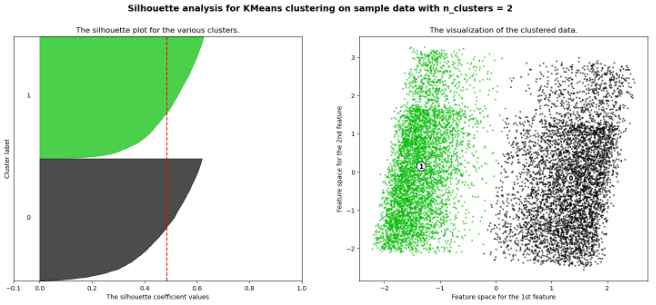
# Part IV. Analysis

## D1. Output and Intermediate Calculations

See attached Jupyter notebook, under the heading ‘Data Analysis’ for code and output. I largely followed the methodology given by scikit-learn in the silhouette analysis article (n.d.), combined with those given by Bruce, Bruce & Gedeck (2020) and Kaloyanova (2021).

Using the reduced dimension dataset with 2 principal components, I run a loop running the *k-*means algorithm with between 2 and 7 clusters, keeping track of the inertia for each and outputting the silhouette score for each. I generate 2 plots for each n\_clusters value, one plotting the silhouette scores for each sample, and one plotting the samples colored by the cluster number. At the end I plot the inertia value for each value of n\_clusters. With this data set, the data visually shows 2 distinct clusters (left/right), with an argument to be made for separating each larger cluster into 2 subclusters top to bottom.

Figure 3



The average silhouette score is slightly higher for n\_clusters = 4, and the inertia value is much lower. The elbow method would suggest choosing 4 for n\_clusters.

Figure 4

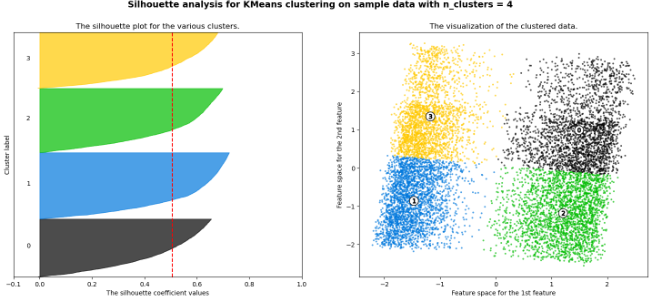
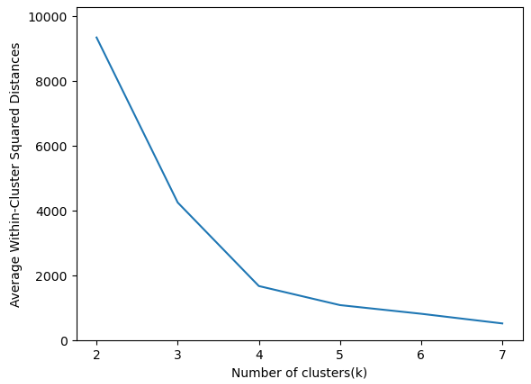


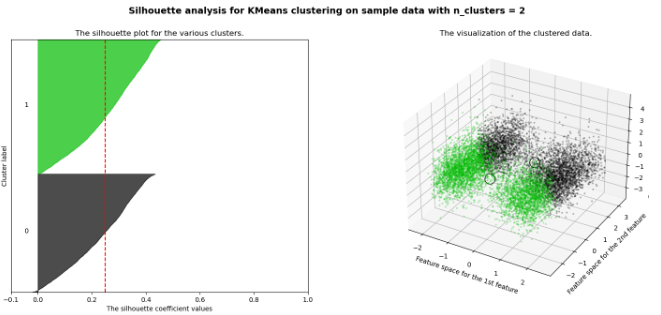
Figure 5  
*Elbow plot of inertia data*



The top-bottom cluster split with n\_clusters = 4 does not align with the slight gap about 2/3 of the way up the larger clusters. This shows the *k-*means algorithm’s proclivity for near-equal-sized clusters over more visually apparent groupings.

I then re-ran the analysis on the reduced data set with 3 principal components to see if it would behave similarly. Interestingly, in this case, when n\_clusters = 2, the cluster split was horizontal rather than vertical with respect to the first two PCs. The silhouette score was much lower, indicating a poor fit.

Figure 6



The best silhouette score was on n\_clusters = 4 again, though 3 and 5 performed nearly as well. The elbow on the inertia graph was not as pronounced with this data set, so arguments could be made for n\_clusters = 3, 4, or 5.

In the end, I would choose to use the 2-PC dataset with n\_clusters = 4. This choice provides the overall best silhouette score tested, and the only one to rise above the 0.5 mark, which is considered a good threshold (Dabbura, 2018).

## D2. Code

See attached Jupyter notebook.

# Part V. Data Summary & Implications

## E1. Quality

According to Dabbura, silhouette analysis is one method to evaluate the quality of the clusters determined by the *k-*means algorithm and to choose the proper *k* (2018)*.* The analysis works by calculating for each data point: the average distance from all data points in the same cluster (ai), the average distance from all data points in the closest cluster (bi), and the silhouette score (Dabbura, 2018). This score will fall in the range [-1, 1]. Positive values close to 1 indicate that the sample is “far away from the neighboring clusters”; close to zero indicates the sample is “close to neighboring clusters”; negative values indicate the sample is likely “assigned to the wrong cluster”. (Dabbura, 2018).

Working with the 2-PC reduced data set and choosing a *k* of 4, I achieved an average silhouette score of 0.506 with each cluster having a large number of points above that value.

## E2. Results & Implications

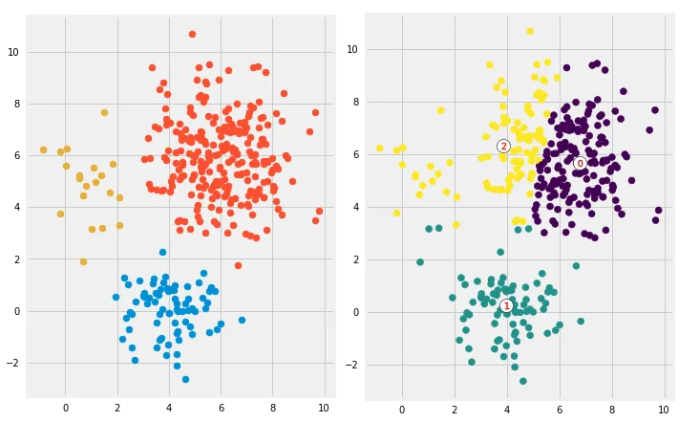
While there are multiple reasonable ways to cluster our patients based on the seven chosen variables, choosing to group them into four clusters seems the most reasonable, edging out other choices in silhouette score and having a low comparative inertia.

The clusters were determined and plotted in a two-dimensional principal component space. The first PC is strongly correlated with Vitamin D levels (loading = -0.976), while the second PC is roughly equally determined by patient age and total charge (loadings = 0.701, 0.707 respectively). This implies that there appears to be some relationship between these variables in our patient data.

## E3. Limitations

As discussed in section D1, the *k-*means algorithm prefers determining nearly equal sized clusters, even when the data is truly divided into large and small clusters. Dabbura gives an example of this issue using three randomly generated clusters with sizes (20, 75, 250).

Figure 7  
*K-means incorrect cluster assignment with unequal cluster sizes (Dabbura, 2018)*



Given the data points from the 2-PC data set and 4 clusters, I would draw the cluster boundary differently than *k-*means did in Figure 4. (See Figure 8)

Figure 8  
*Data from Figure 4 with proposed redrawn cluster boundary*



## E4. Course of Action

I would recommend that the hospital further investigate its data using these determined patient clusters to see if any patient outcome (for example, readmission) is correlated with any cluster.

# Part VI. Demonstration & Supporting Documentation

## F. Demonstration Video

## A video describing my methods and code can be found at: <https://wgu.hosted.panopto.com/Panopto/Pages/Viewer.aspx?id=dc86e076-3358-4627-898f-b136013e7bcd>

## G. Third-party Code Sources

Bruce, P., Bruce, A., & Gedeck, P. (2020). *Practical Statistics for Data Scientists : 50+ Essential Concepts Using R and Python*. O'Reilly Media, Inc.

Scikit-learn.org. (n.d.). *Selecting the number of clusters with silhouette analysis on kmeans clustering*. <https://scikit-learn.org/stable/auto_examples/cluster/plot_kmeans_silhouette_analysis.html>

## H. References

Bruce, P., Bruce, A., & Gedeck, P. (2020). *Practical Statistics for Data Scientists : 50+ Essential Concepts Using R and Python*. O'Reilly Media, Inc.

Dabbura, I. (Sept. 17, 2018). *K-means Clustering: Algorithm, Applications, Evaluation Methods, and Drawbacks*. Towards Data Science. <https://towardsdatascience.com/k-means-clustering-algorithm-applications-evaluation-methods-and-drawbacks-aa03e644b48a>

Kaloyanova, E. (July 29, 2021). 365 Data Science. *How to Combine PCA and K-means Clustering in Python?* <https://365datascience.com/tutorials/python-tutorials/pca-k-means/>

Larose, C., & Larose, D. (2019). *Data Science Using Python and R.* Wiley.

Middleton, K. (n.d.). *Getting Started with D206 | Principal Component Analysis.* Western Governors University. <https://wgu.hosted.panopto.com/Panopto/Pages/Viewer.aspx?id=3bcc452f-fa35-43be-b69f-b05901356f95>

Needham, Mark (Aug. 27, 2016). *scikit-learn: Clustering and the curse of dimensionality*. <https://www.markhneedham.com/blog/2016/08/27/scikit-learn-clustering-and-the-curse-of-dimensionality/>

Scikit-learn.org. (n.d.). *Selecting the number of clusters with silhouette analysis on kmeans clustering*. <https://scikit-learn.org/stable/auto_examples/cluster/plot_kmeans_silhouette_analysis.html>

Scikit-learn.org. (February 2024). *scikit-learn 1.4.1 API reference documentation*. [https://scikit-learn.org/stable/modules/classes.html#](https://scikit-learn.org/stable/modules/classes.html)