D603 Task 2
By Eric Williams

B1:PROPOSAL OF QUESTION
Is it possible to group patients by demographic and medical history in a meaningful way that allows us to make predictions about a patient’s future and provide them with better care? If so, what do those classification clusters look like and what are their primary attributes?

B2:DEFINED GOAL
The goal of this analysis is to use k-cluster analysis to group patients by demographic and medical history in order to provide them with better care. By grouping patients with similar characteristics, we can better identify trends in patient health and help create tailor-made health plans that will be easier to do by grouping them into clusters.

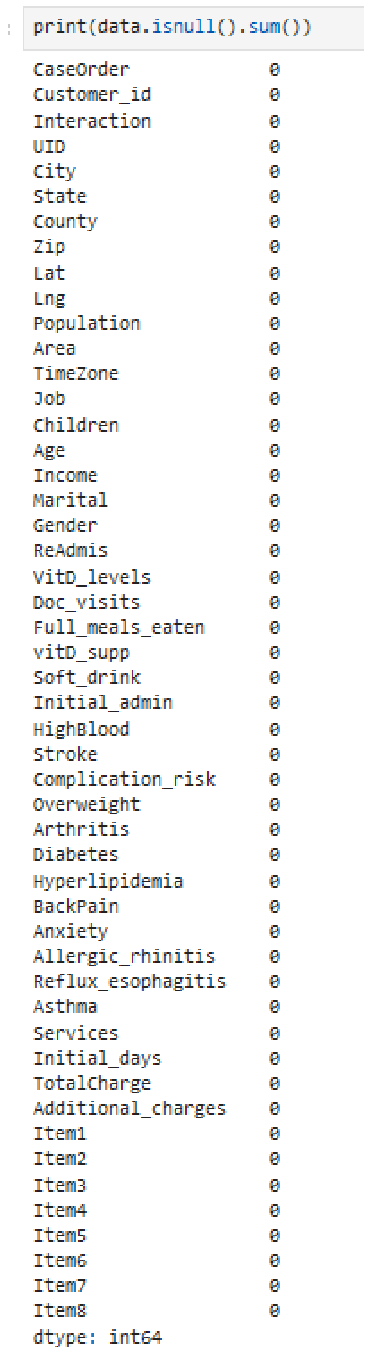
C1:EXPLANATION OF CLUSTERING TECHNIQUE
K-clustering is a machine learning algorithm that can split our data into any number of clusters we would like. The point of clustering is to identify similar datapoints to find trends in the data and create groups of similar patients. In this case, finding patients with similar characteristics can help us find similarities and trends between real life people with medical needs in order to offer them better treatment.

C2:SUMMARY OF TECHNIQUE ASSUMPTION
One assumption of the k-clustering technique is that the clusterings have similar sizes. If there is a small group of clustered data that is much smaller than the other clusters, that could significantly alter our clustering analysis and introduce inaccuracies. It must also be true that the data is not clustered in an unusual shape, with spherical or elliptical shapes being much more friendly to k-clustering.

C3:PACKAGES OR LIBRARIES LIST
Here are the packages I imported and why I needed them

1. Pandas and Numpy to work with the data in a dataframe and perform calculations needed for analysis
2. Sklearn for its various tools for machine learning. Those specific tools are:
    a. StandardScaler to scale the data correctly to prevent one datapoint from dominating the analysis
    b. KMeans to run the k-cluster analysis
    c. Sklearn.decomposition import PCA to help with dimensions reduction for the clustering
3. Matplotlib.pyplot to create plots for the k-cluster and elbow method analysis

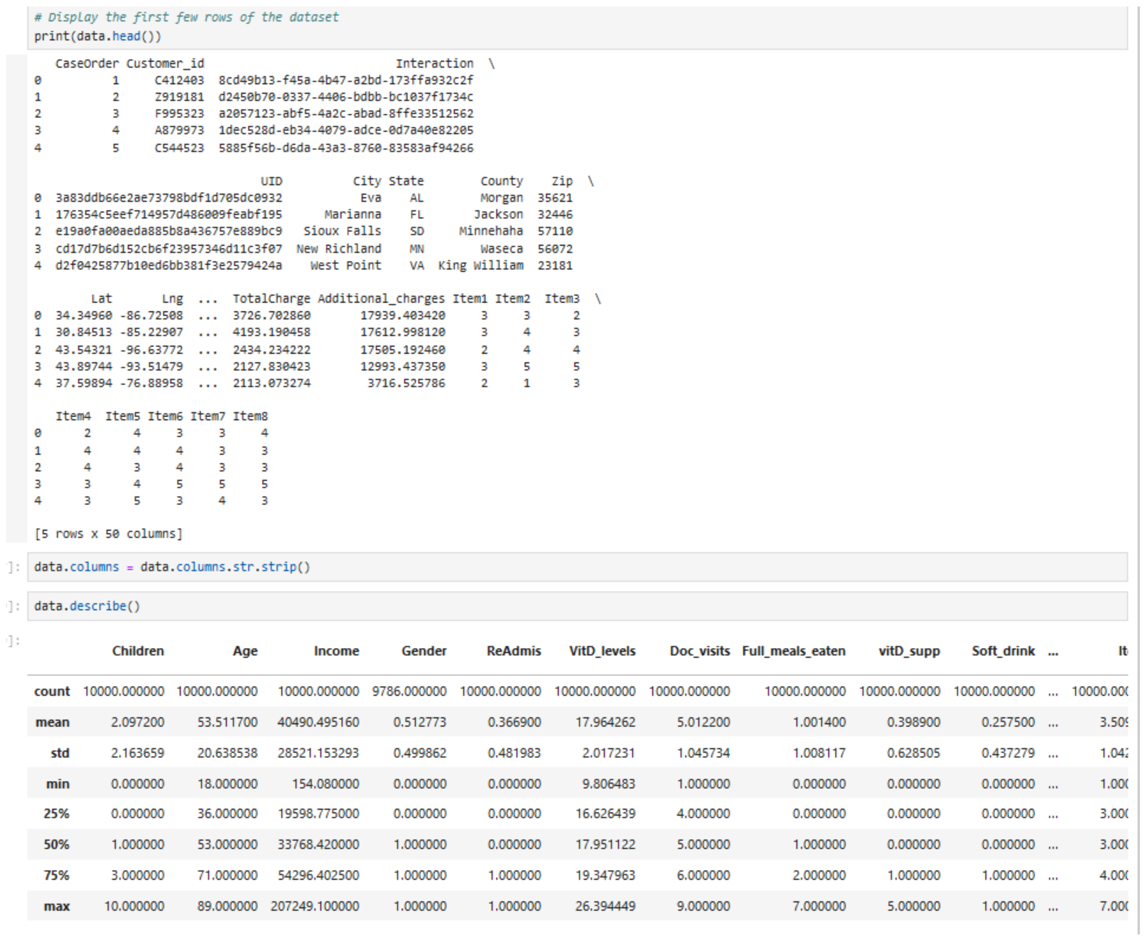
D1:DATA PREPROCESSING
To prepare the data for processing, I needed to ensure the data was clean with no missing values.



```
print(data.isnull().sum())
```

```
CaseOrder             0
Customer_id           0
Interaction           0
UID                   0
City                  0
State                 0
County                0
Zip                   0
Lat                   0
Lng                   0
Population            0
Area                  0
TimeZone              0
Job                   0
Children              0
Age                   0
Income                0
Marital               0
Gender                0
ReAdmis               0
VitD_levels           0
Doc_visits            0
Full_meals_eaten      0
vitD_supp             0
Soft_drink            0
Initial_admin         0
HighBlood             0
Stroke                0
Complication_risk     0
Overweight            0
Arthritis             0
Diabetes              0
Hyperlipidemia        0
BackPain              0
Anxiety               0
Allergic_rhinitis     0
Reflux_esophagitis    0
Asthma                0
Services              0
Initial_days          0
TotalCharge           0
Additional_charges    0
Item1                 0
Item2                 0
Item3                 0
Item4                 0
Item5                 0
Item6                 0
Item7                 0
Item8                 0
dtype: int64
```

I also stripped any spaces before or after the column titles to reduce the likelihood of problems with column names later on. I then inspected the data to ensure it looked properly prepared for analysis:



```
# Display the first few rows of the dataset
print(data.head())
```

```
   CaseOrder Customer_id                           Interaction  \
0          1     C412403  8cd49b13-f45a-4b47-a2bd-173ffa932c2f
1          2     Z919181  d2450b70-0337-4406-bdbb-bc1037f1734c
2          3     F995323  a2057123-abf5-4a2c-abad-8ffe33512562
3          4     A879973  1dec528d-eb34-4079-adce-0d7a40e82205
4          5     C544523  5885f56b-d6da-43a3-8760-83583af94266

                                UID         City State        County    Zip  \
0  3a83ddb66e2ae73798bdf1d705dc0932          Eva    AL        Morgan  35621
1  176354c5eef714957d486009feabf195     Marianna    FL       Jackson  32446
2  e19a0fa00aeda885b8a436757e889bc9  Sioux Falls    SD     Minnehaha  57110
3  cd17d7b6d152cb6f23957346d11c3f07  New Richland    MN        Waseca  56072
4  d2f0425877b10ed6bb381f3e2579424a   West Point    VA  King William  23181

        Lat       Lng  ...  TotalCharge Additional_charges Item1 Item2  Item3  \
0  34.34960 -86.72508  ...  3726.702860       17939.403420     3     3      2
1  30.84513 -85.22907  ...  4193.190458       17612.998120     3     4      3
2  43.54321 -96.63772  ...  2434.234222       17505.192460     2     4      4
3  43.89744 -93.51479  ...  2127.830423       12993.437350     3     5      5
4  37.59894 -76.88958  ...  2113.073274        3716.525786     2     1      3

   Item4  Item5 Item6 Item7 Item8
0      2      4     3     3     4
1      4      4     4     3     3
2      4      3     4     3     3
3      3      4     5     5     5
4      3      5     3     4     3

[5 rows x 50 columns]
```

```
]: data.columns = data.columns.str.strip()
```

```
]: data.describe()
```

]:

	Children	Age	Income	Gender	ReAdmis	VitD_levels	Doc_visits	Full_meals_eaten	vitD_supp	Soft_drink	...	It
count	10000.000000	10000.000000	10000.000000	9786.000000	10000.000000	10000.000000	10000.000000	10000.000000	10000.000000	10000.000000	...	10000.00
mean	2.097200	53.511700	40490.495160	0.512773	0.366900	17.964262	5.012200	1.001400	0.398900	0.257500	...	3.50
std	2.163659	20.638538	28521.153293	0.499862	0.481983	2.017231	1.045734	1.008117	0.628505	0.437279	...	1.04
min	0.000000	18.000000	154.080000	0.000000	0.000000	9.806483	1.000000	0.000000	0.000000	0.000000	...	1.00
25%	0.000000	36.000000	19598.775000	0.000000	0.000000	16.626439	4.000000	0.000000	0.000000	0.000000	...	3.00
50%	1.000000	53.000000	33768.420000	1.000000	0.000000	17.951122	5.000000	1.000000	0.000000	0.000000	...	3.00
75%	3.000000	71.000000	54296.402500	1.000000	1.000000	19.347963	6.000000	2.000000	1.000000	1.000000	...	4.00
max	10.000000	89.000000	207249.100000	1.000000	1.000000	26.394449	9.000000	7.000000	5.000000	1.000000	...	7.00

In addition, I dropped columns not related to demographic and medical history, and then encoded the variables because K-cluster analysis requires numerical input. The code for this is provided below.

D2:DATASET VARIABLES

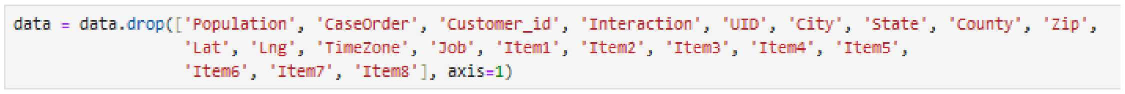
I dropped any columns not related to medical history or patient demographic. Below I have sorted the remaining variables by continuous and categorical, although some could technically be listed as either. This is because some columns are numerical but fit nicely into a category. However, I have chosen to list all numerical data as continuous because they do not need to be encoded for analysis:

<u>Categorical (non-numerical)</u>
Marital
Gender
ReAdmis
Soft_drink
Initial_admin
HighBlood
Stroke

Overweight
Arthritis
Diabetes
Hyperlipidemia
BackPain
Anxiety
Allergic_rhinitis
Reflux_esophagitis
Asthma
Services
Complication_risk

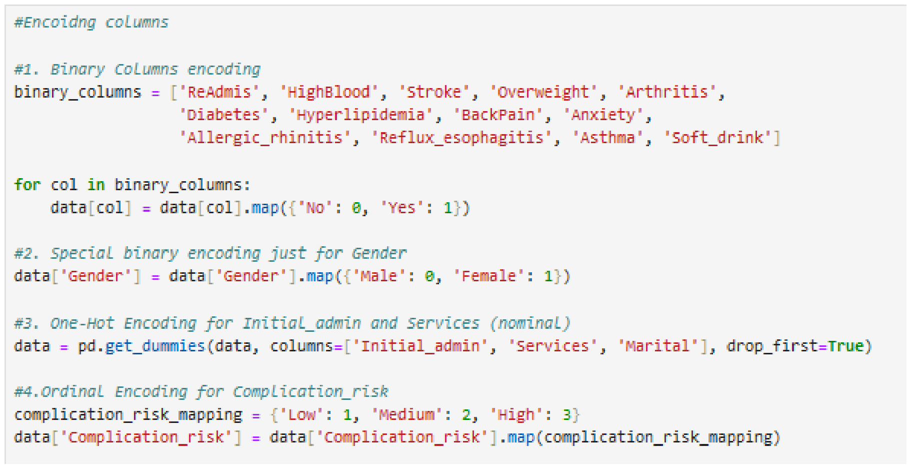
Continuous (numerical)
Income
VitD_levels
Doc_visits
Initial_days
TotalCharge
Additional_charges
Full_meals_eaten
vitD_supp
Children
Age

D3:STEPS FOR ANALYSIS
As mentioned above, I dropped the columns not needed for the analysis:

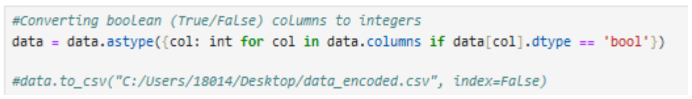


```python
data = data.drop(['Population', 'CaseOrder', 'Customer_id', 'Interaction', 'UID', 'City', 'State', 'County', 'Zip',
                  'Lat', 'Lng', 'TimeZone', 'Job', 'Item1', 'Item2', 'Item3', 'Item4', 'Item5',
                  'Item6', 'Item7', 'Item8'], axis=1)
```

Then I encoded the variables:



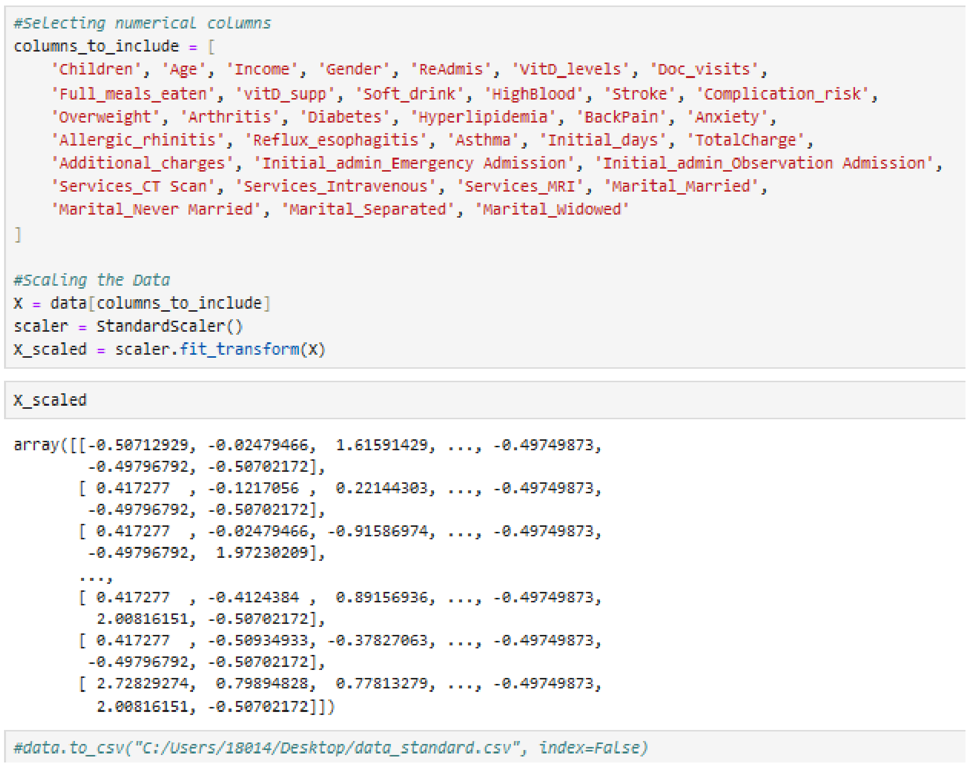
```python
#Encoidng columns

#1. Binary Columns encoding
binary_columns = ['ReAdmis', 'HighBlood', 'Stroke', 'Overweight', 'Arthritis',
                  'Diabetes', 'Hyperlipidemia', 'BackPain', 'Anxiety',
                  'Allergic_rhinitis', 'Reflux_esophagitis', 'Asthma', 'Soft_drink']

for col in binary_columns:
    data[col] = data[col].map({'No': 0, 'Yes': 1})

#2. Special binary encoding just for Gender
data['Gender'] = data['Gender'].map({'Male': 0, 'Female': 1})

#3. One-Hot Encoding for Initial_admin and Services (nominal)
data = pd.get_dummies(data, columns=['Initial_admin', 'Services', 'Marital'], drop_first=True)

#4.Ordinal Encoding for Complication_risk
complication_risk_mapping = {'Low': 1, 'Medium': 2, 'High': 3}
data['Complication_risk'] = data['Complication_risk'].map(complication_risk_mapping)
```



```
#Converting boolean (True/False) columns to integers
data = data.astype({col: int for col in data.columns if data[col].dtype == 'bool'})

#data.to_csv("C:/Users/18014/Desktop/data_encoded.csv", index=False)
```

Essentially, the columns that were already boolean just needed to be converted to 0 and 1 values. All of the encoding steps work toward converting the data to be numerical. Although Male and Female was binary with only two entries, I did a special encoding case because the values were not “yes” and “no” like the others and there was also a ‘nonbinary’ option. I then encoded the nominal variables that had more than two entries. Lastly, I did ordinal encoding as a special case for complication risk because it is more accurate to assign low complication a lower value than medium, and assign ‘High’ the highest value (rather than assigning random distinct values to each).

My next step was to select the columns I wanted to include, then scale the data. This is necessary because there are some numerical values that are much higher than others (such as income) that we don’t want dominating our values. Scaling the data puts all the columns on the same footing, from a weight perspective. I then exported the data at this point after inspecting it.



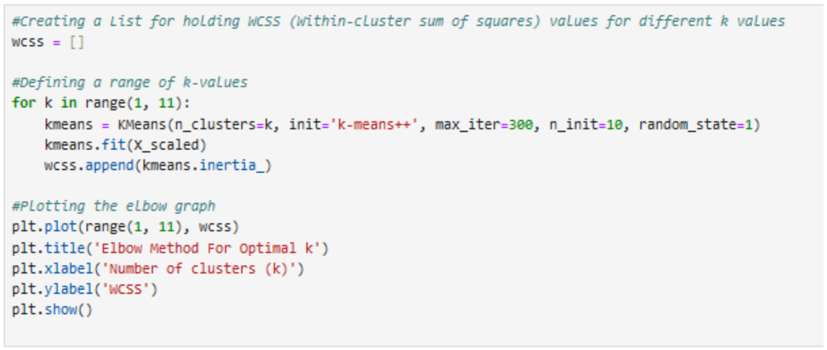
```
#Selecting numerical columns
columns_to_include = [
    'Children', 'Age', 'Income', 'Gender', 'ReAdmis', 'VitD_levels', 'Doc_visits',
    'Full_meals_eaten', 'vitD_supp', 'Soft_drink', 'HighBlood', 'Stroke', 'Complication_risk',
    'Overweight', 'Arthritis', 'Diabetes', 'Hyperlipidemia', 'BackPain', 'Anxiety',
    'Allergic_rhinitis', 'Reflux_esophagitis', 'Asthma', 'Initial_days', 'TotalCharge',
    'Additional_charges', 'Initial_admin_Emergency Admission', 'Initial_admin_Observation Admission',
    'Services_CT Scan', 'Services_Intravenous', 'Services_MRI', 'Marital_Married',
    'Marital_Never Married', 'Marital_Separated', 'Marital_Widowed'
]

#Scaling the Data
X = data[columns_to_include]
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)
```

```
X_scaled
```

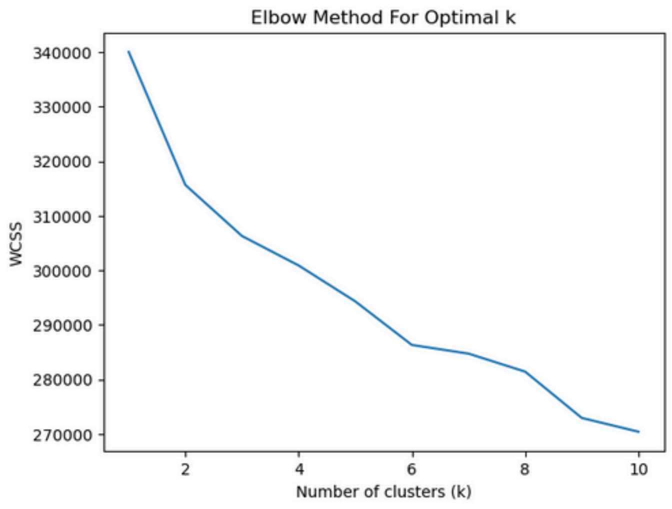
```
array([[-0.50712929, -0.02479466,  1.61591429, ..., -0.49749873,
        -0.49796792, -0.50702172],
       [ 0.417277  , -0.1217056 ,  0.22144303, ..., -0.49749873,
        -0.49796792, -0.50702172],
       [ 0.417277  , -0.02479466, -0.91586974, ..., -0.49749873,
        -0.49796792,  1.97230209],
       ...,
       [ 0.417277  , -0.4124384 ,  0.89156936, ..., -0.49749873,
         2.00816151, -0.50702172],
       [ 0.417277  , -0.50934933, -0.37827063, ..., -0.49749873,
        -0.49796792, -0.50702172],
       [ 2.72829274,  0.79894828,  0.77813279, ..., -0.49749873,
         2.00816151, -0.50702172]])
```

```
#data.to_csv("C:/Users/18014/Desktop/data_standard.csv", index=False)
```

Next I used the elbow method to see what the optimal value for k would be. I created a plot of the WCSS values against the number of clusters to look for an elbow shape in the graph. Although I had a few options to choose from, I decided the k=3 value to be most appropriate based on the graph shape.

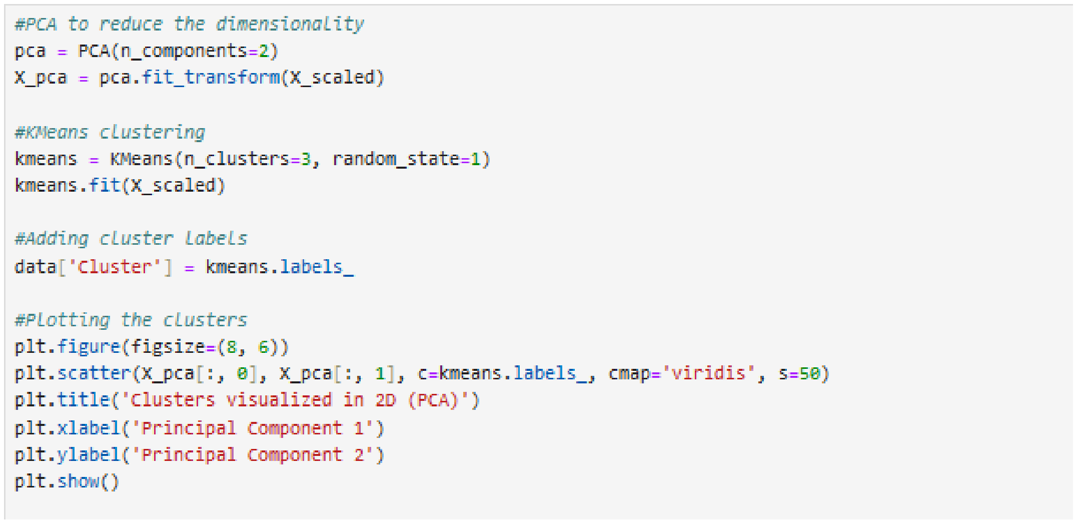


```python
#Creating a List for holding WCSS (Within-cluster sum of squares) values for different k values
wcss = []

#Defining a range of k-values
for k in range(1, 11):
    kmeans = KMeans(n_clusters=k, init='k-means++', max_iter=300, n_init=10, random_state=1)
    kmeans.fit(X_scaled)
    wcss.append(kmeans.inertia_)

#Plotting the elbow graph
plt.plot(range(1, 11), wcss)
plt.title('Elbow Method For Optimal k')
plt.xlabel('Number of clusters (k)')
plt.ylabel('WCSS')
plt.show()
```

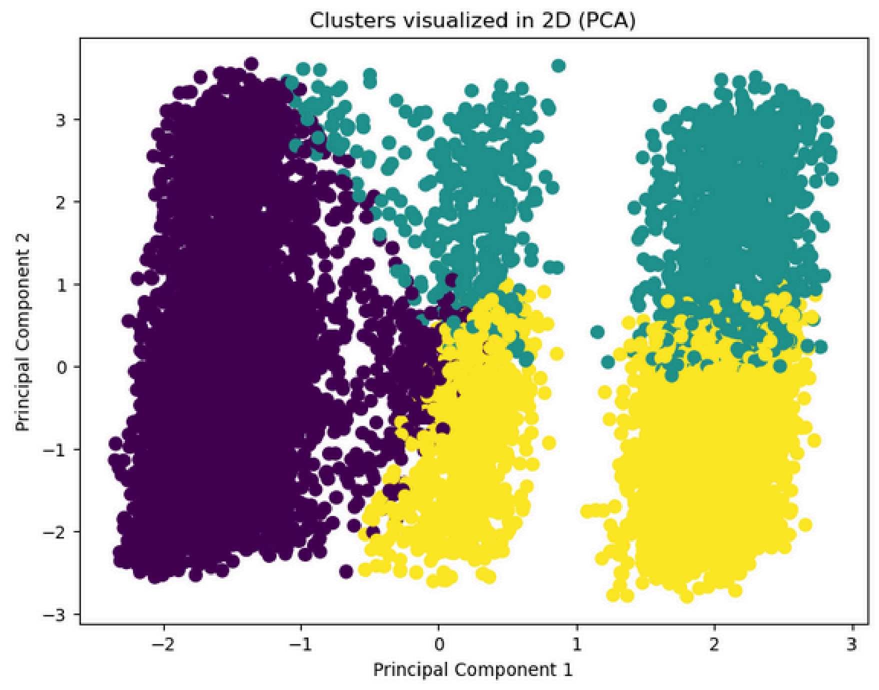


E1:OUTPUT AND INTERMEDIATE CALCULATIONS

My next step was to finally run the k-clustering with a k-value of 3. I an a quick PCA to fix the dimensionality issue and then created a plot in order to visualize how well the clustering went:



```python
#PCA to reduce the dimensionality
pca = PCA(n_components=2)
X_pca = pca.fit_transform(X_scaled)

#KMeans clustering
kmeans = KMeans(n_clusters=3, random_state=1)
kmeans.fit(X_scaled)

#Adding cluster labels
data['Cluster'] = kmeans.labels_

#Plotting the clusters
plt.figure(figsize=(8, 6))
plt.scatter(X_pca[:, 0], X_pca[:, 1], c=kmeans.labels_, cmap='viridis', s=50)
plt.title('Clusters visualized in 2D (PCA)')
plt.xlabel('Principal Component 1')
plt.ylabel('Principal Component 2')
plt.show()
```



F1:QUALITY OF CLUSTERING TECHNIQUE
Upon visual inspection of the scatterplot, the clustering technique seems to work just okay. The machine was effective in identifying the purple cluster, however, it appears the clustering algorithm got confused and split two clusters in half instead of considering them their own clusters. On the positive side, the dots do not have a lot of overlap, so the model was somewhat successful. I think the ideal clustering would show a left, right, and middle cluster.

F2:RESULTS AND IMPLICATIONS
Although I have built a model that can classify all patients into three clusters, the clustering leaves some to be desired based on the plot above. Given the scatterplot above, it appears there are three general clusters of patients, but the model did not successfully define two of the clusters as well as it could have. Still, there may be some implications we are not seeing in the above model because we have reduced the dimensions after starting with so many variables. In short, the result is that the model can identify 3 clear-cut classes of patients, but further work needs to be done to either improve the model or understand why the clustering functions this way (splitting the middle cluster and the right cluster into halves). If further examination shows this clustering is effective, we should use this classification system for helping patients receive better medical care by analyzing groups of patients and addressing their needs. The specifics of this are outlined below.

F3:LIMITATION
One limitation we must consider is the number we chose for our k clusters. If we had chosen more clusters, we could have possibly created a more in depth model of clustering. After evaluating the elbow graph, we could have also reasonably created a model with 2 clusters or 6

clusters which is clear from the above graph. However, the choice to include 3 clusters proved reasonable, as justified in the above plot.

It is also worth noting that k-cluster models are sensitive to outliers. Although we don't see any major outliers in our PCA plot, the existence of any outliers before the PCA analysis could have affected our clustering.

Lastly, if we were to reduce the number of variables used, we may get more accurate clustering results. With a very complicated and high dimensional dataset, there is always a chance the k-clustering doesn't work perfectly.

F4:COURSE OF ACTION

The first thing we should do with this clustering is look into whether or not a better model can be created. I would start by reducing the number of variables used to see if we can get better clustering results. It would also be worth looking further into the clusters to see if they have more in common than the simple scatterplot above shows--the scatterplot may be limited because it is attempting to turn 34 variables into a two dimensional graph.

However, if we find the model to be effective despite the problem shown in the scatterplot, the most useful thing we can do with this analysis is look for trends in customer care to provide them with better healthcare options. If we find clusters with more medical problems and higher expenses, we could offer them different pricing or more treatment options that could benefit them. This analysis could also help predict and prioritize how the hospital allocates resources to help different classes of patients by grouping them together.

Sources

Because of the similarity in process, the formatting of this paper and code for this assignment were partially taken from my assignment for Task 1 in this class.

Besides my own work, no sources were used except for WGU official course materials.