Western Governors University

D212 – Data Mining II – Task 2

By Krista Moik

Table of Contents

Part I: Research Question

A1. Proposal of Question

Using the WGU provided medical_clean data set, my research question is: **can we use PCA to reduce the dimensionality of the data set to find the most important components?**

A2. Defined Goal

My goal is to use PCA to reduce the dimensionality of the dataset to find which components have the most and least significance.

Part II: Method Justification

B1. Explanation of PCA

Principal Component Analysis will use the continuous variables present in the data set to transform them into principal components to reduce the dimensionality of the data set while maintaining as much of the important information as possible. The principal components are linear combinations of the original variables that are then weighted by their variances in an orthogonal dimension. PCA works by normalizing and scaling the data, creating a covariance matrix to see how the features vary amongst each other, use eigenvalues and Kaiser Criterion (you can also use the Elbow Method) to find the most important Principal Components, select the top PCs to reduce the dimensionality, and then finally it transforms the dataset into the new space as determined by the selected PCs (Towards Data Science, 2020). The outcome of performing PCA are reduced variables in the form of principal components, where the most meaningful components can be found to include in the model, and which also makes it easier and quicker to find trends in the data.

B2. PCA Assumption

One assumption of PCA is that it assumes the features are correlated and have a linear relationship (Keboola 2022).

Part III: Data Preparation

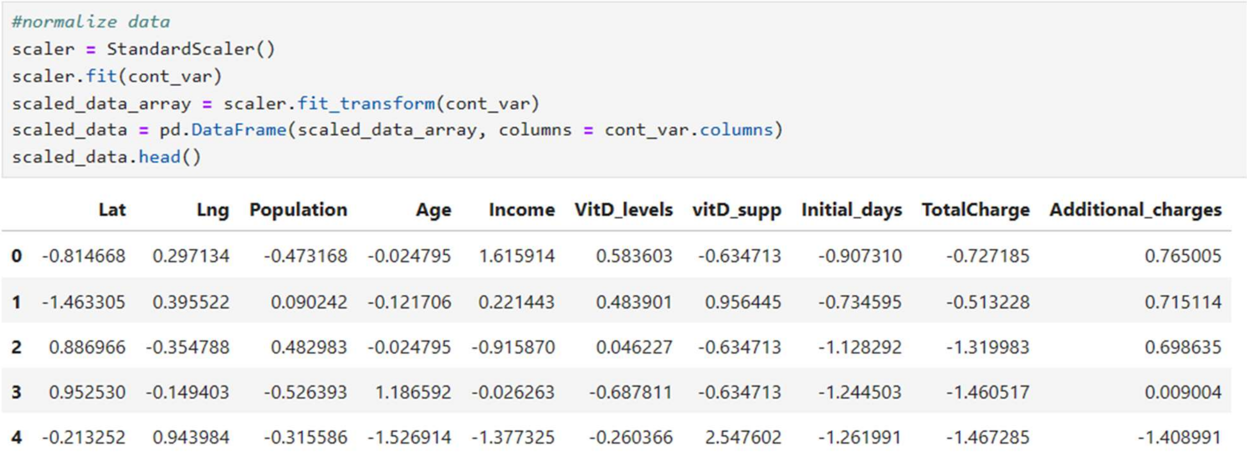
C1. Continuous Data Set Variables

The continuous variables I will use in my PCA are the 10 continuous variables in the data set:

Lat, Lng, Population, Age, Income, VitD_levels, vitD_supp, Initial_days, TotalCharge, Additional_charges

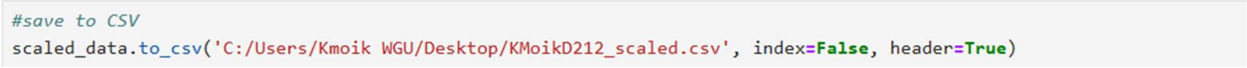
C2. Standardization of Data Set Variables

I originally used RobustScaler to scale the data, however, this resulted in just 1 principal component with an eigenvalue greater than 1, but that only explained approximately 22% of the variance in the dataset. I removed the RobustScaler and used StandardScaler instead to scale data using the mean, resulting in more principal components with eigenvalues greater than 1 that explained more of the variance in the dataset (Towards Data Science, 2020). It was important to me to maintain as much of the important information in the dataset as possible (Medium, 2021). This dataset is available in the attached CSV file titled KMoikD212_scaled:



```
#normalize data
scaler = StandardScaler()
scaler.fit(cont_var)
scaled_data_array = scaler.fit_transform(cont_var)
scaled_data = pd.DataFrame(scaled_data_array, columns = cont_var.columns)
scaled_data.head()
```

	Lat	Lng	Population	Age	Income	VitD_levels	vitD_supp	Initial_days	TotalCharge	Additional_charges
0	-0.814668	0.297134	-0.473168	-0.024795	1.615914	0.583603	-0.634713	-0.907310	-0.727185	0.765005
1	-1.463305	0.395522	0.090242	-0.121706	0.221443	0.483901	0.956445	-0.734595	-0.513228	0.715114
2	0.886966	-0.354788	0.482983	-0.024795	-0.915870	0.046227	-0.634713	-1.128292	-1.319983	0.698635
3	0.952530	-0.149403	-0.526393	1.186592	-0.026263	-0.687811	-0.634713	-1.244503	-1.460517	0.009004
4	-0.213252	0.943984	-0.315586	-1.526914	-1.377325	-0.260366	2.547602	-1.261991	-1.467285	-1.408991

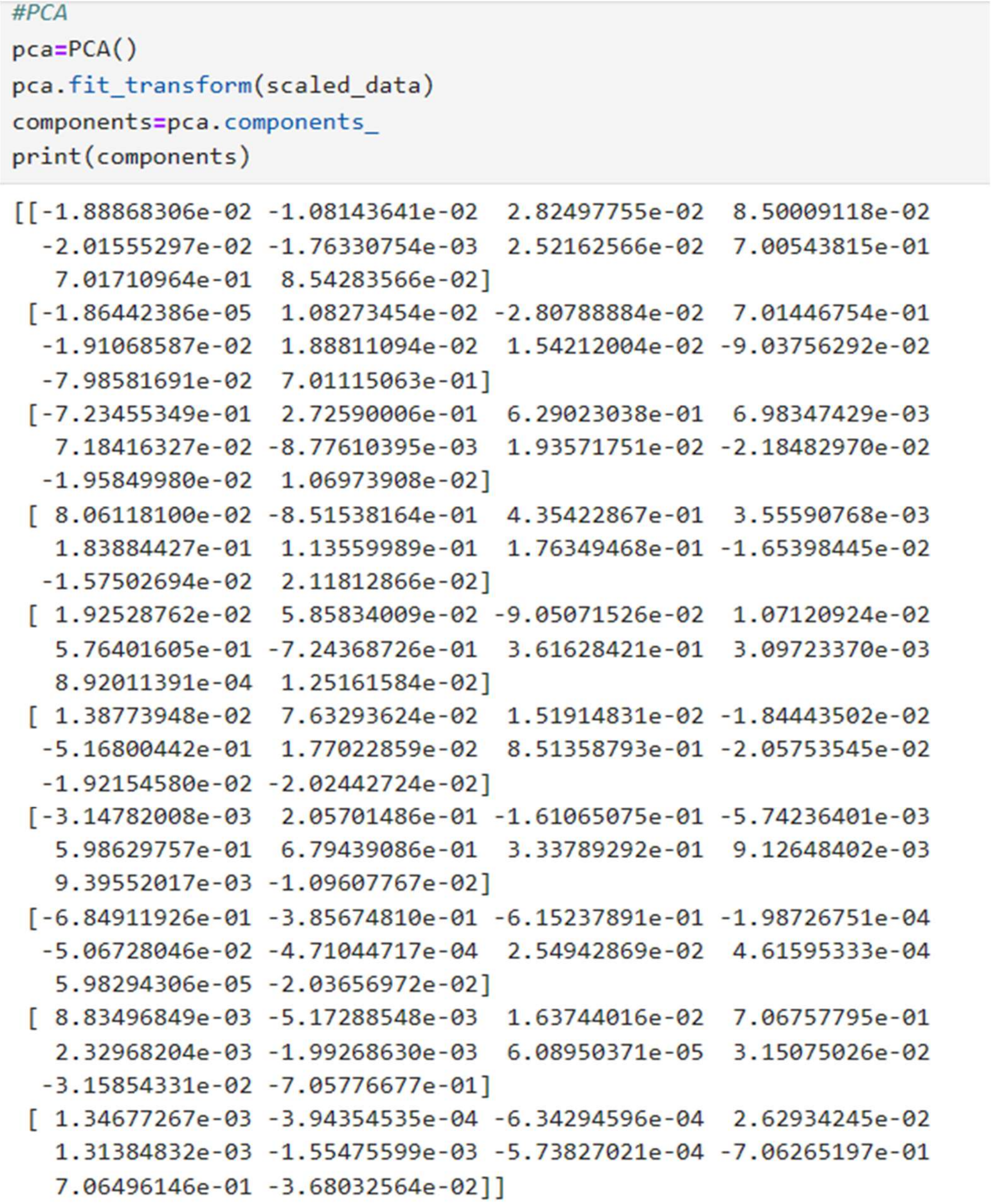


```
#save to CSV
scaled_data.to_csv('C:/Users/Kmoik WGU/Desktop/KMoikD212_scaled.csv', index=False, header=True)
```

Part IV: Analysis

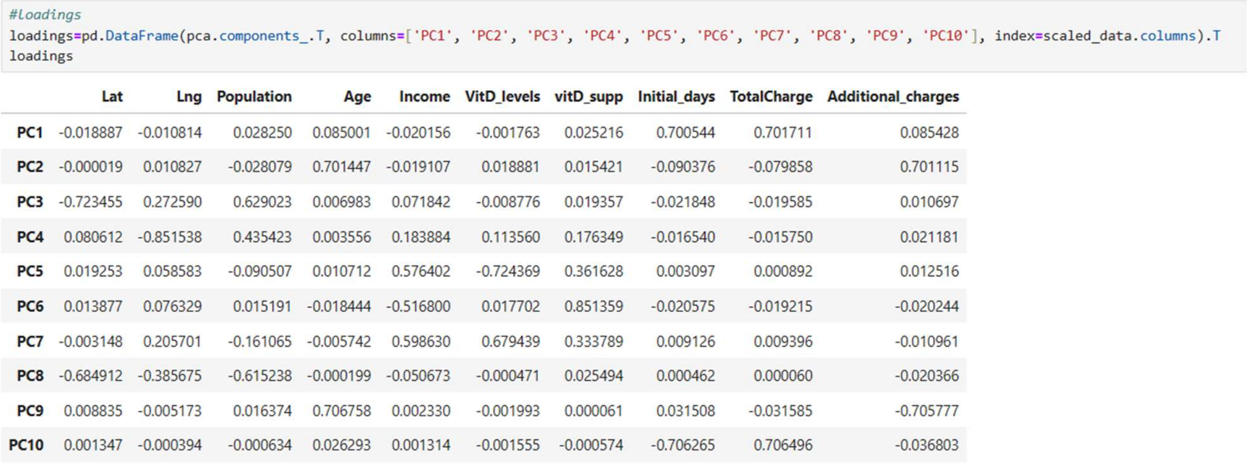
D1. Principal Components

Once PCA was instantiated, I was able to create a loadings matrix of the principal components (Western Governors University, n.d.):



```
#PCA
pca=PCA()
pca.fit_transform(scaled_data)
components=pca.components_
print(components)
```

```
[[-1.88868306e-02 -1.08143641e-02  2.82497755e-02  8.50009118e-02
  -2.01555297e-02 -1.76330754e-03  2.52162566e-02  7.00543815e-01
   7.01710964e-01  8.54283566e-02]
 [-1.86442386e-05  1.08273454e-02 -2.80788884e-02  7.01446754e-01
  -1.91068587e-02  1.88811094e-02  1.54212004e-02 -9.03756292e-02
  -7.98581691e-02  7.01115063e-01]
 [-7.23455349e-01  2.72590006e-01  6.29023038e-01  6.98347429e-03
   7.18416327e-02 -8.77610395e-03  1.93571751e-02 -2.18482970e-02
  -1.95849980e-02  1.06973908e-02]
 [ 8.06118100e-02 -8.51538164e-01  4.35422867e-01  3.55590768e-03
   1.83884427e-01  1.13559989e-01  1.76349468e-01 -1.65398445e-02
  -1.57502694e-02  2.11812866e-02]
 [ 1.92528762e-02  5.85834009e-02 -9.05071526e-02  1.07120924e-02
   5.76401605e-01 -7.24368726e-01  3.61628421e-01  3.09723370e-03
   8.92011391e-04  1.25161584e-02]
 [ 1.38773948e-02  7.63293624e-02  1.51914831e-02 -1.84443502e-02
  -5.16800442e-01  1.77022859e-02  8.51358793e-01 -2.05753545e-02
  -1.92154580e-02 -2.02442724e-02]
 [-3.14782008e-03  2.05701486e-01 -1.61065075e-01 -5.74236401e-03
   5.98629757e-01  6.79439086e-01  3.33789292e-01  9.12648402e-03
   9.39552017e-03 -1.09607767e-02]
 [-6.84911926e-01 -3.85674810e-01 -6.15237891e-01 -1.98726751e-04
  -5.06728046e-02 -4.71044717e-04  2.54942869e-02  4.61595333e-04
   5.98294306e-05 -2.03656972e-02]
 [ 8.83496849e-03 -5.17288548e-03  1.63744016e-02  7.06757795e-01
   2.32968204e-03 -1.99268630e-03  6.08950371e-05  3.15075026e-02
  -3.15854331e-02 -7.05776677e-01]
 [ 1.34677267e-03 -3.94354535e-04 -6.34294596e-04  2.62934245e-02
   1.31384832e-03 -1.55475599e-03 -5.73827021e-04 -7.06265197e-01
   7.06496146e-01 -3.68032564e-02]]
```



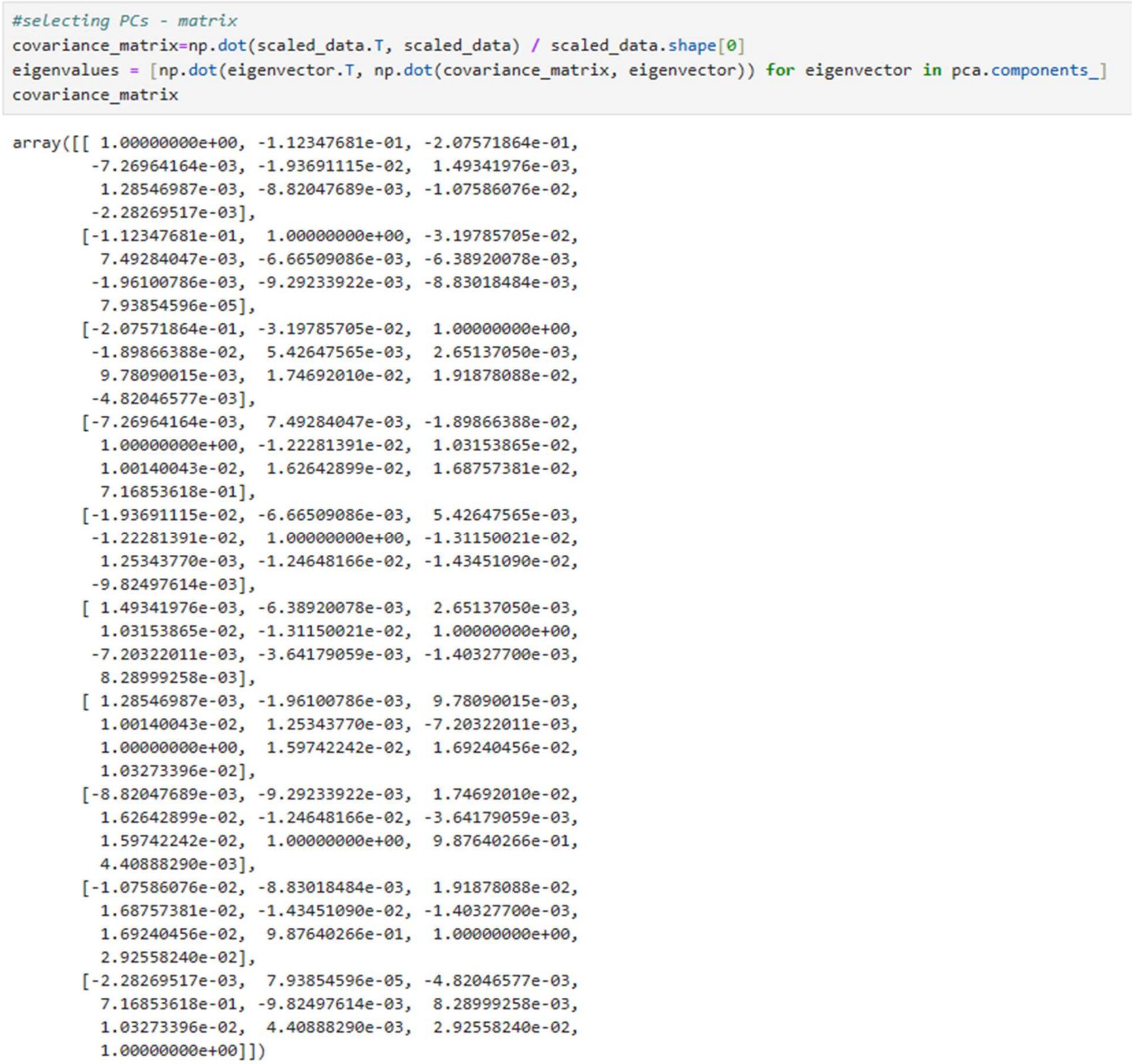
```
#loadings
loadings=pd.DataFrame(pca.components_.T, columns=['PC1', 'PC2', 'PC3', 'PC4', 'PC5', 'PC6', 'PC7', 'PC8', 'PC9', 'PC10'], index=scaled_data.columns).T
loadings
```

	Lat	Lng	Population	Age	Income	VitD_levels	vitD_supp	Initial_days	TotalCharge	Additional_charges
PC1	-0.018887	-0.010814	0.028250	0.085001	-0.020156	-0.001763	0.025216	0.700544	0.701711	0.085428
PC2	-0.000019	0.010827	-0.028079	0.701447	-0.019107	0.018881	0.015421	-0.090376	-0.079858	0.701115
PC3	-0.723455	0.272590	0.629023	0.006983	0.071842	-0.008776	0.019357	-0.021848	-0.019585	0.010697
PC4	0.080612	-0.851538	0.435423	0.003556	0.183884	0.113560	0.176349	-0.016540	-0.015750	0.021181
PC5	0.019253	0.058583	-0.090507	0.010712	0.576402	-0.724369	0.361628	0.003097	0.000892	0.012516
PC6	0.013877	0.076329	0.015191	-0.018444	-0.516800	0.017702	0.851359	-0.020575	-0.019215	-0.020244
PC7	-0.003148	0.205701	-0.161065	-0.005742	0.598630	0.679439	0.333789	0.009126	0.009396	-0.010961
PC8	-0.684912	-0.385675	-0.615238	-0.000199	-0.050673	-0.000471	0.025494	0.000462	0.000060	-0.020366
PC9	0.008835	-0.005173	0.016374	0.706758	0.002330	-0.001993	0.000061	0.031508	-0.031585	-0.705777
PC10	0.001347	-0.000394	-0.000634	0.026293	0.001314	-0.001555	-0.000574	-0.706265	0.706496	-0.036803

Please see the attached ipynb and pdf titled KMoikD212Code2 for the full code.

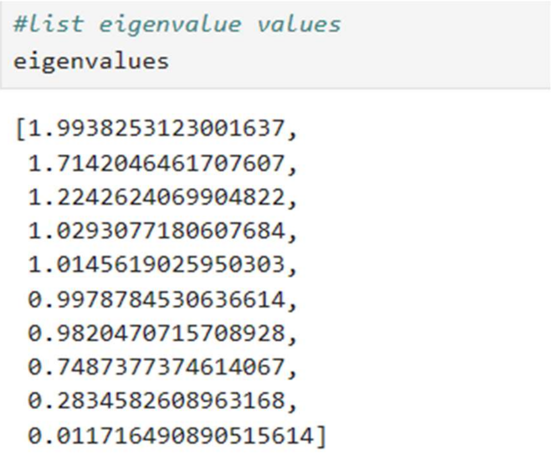
D2. Identification of the Total Number of Components

Using the Kaiser criterion in obtaining eigenvalues, I was able to create a covariance matrix, list, and plot of the eigenvalues of my 10 initial PCs:



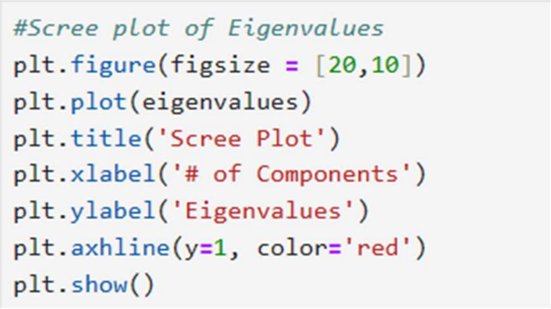
```
#selecting PCs - matrix
covariance_matrix=np.dot(scaled_data.T, scaled_data) / scaled_data.shape[0]
eigenvalues = [np.dot(eigenvector.T, np.dot(covariance_matrix, eigenvector)) for eigenvector in pca.components_]
covariance_matrix
```

```
array([[ 1.00000000e+00, -1.12347681e-01, -2.07571864e-01,
        -7.26964164e-03, -1.93691115e-02,  1.49341976e-03,
         1.28546987e-03, -8.82047689e-03, -1.07586076e-02,
        -2.28269517e-03],
       [-1.12347681e-01,  1.00000000e+00, -3.19785705e-02,
         7.49284047e-03, -6.66509086e-03, -6.38920078e-03,
        -1.96100786e-03, -9.29233922e-03, -8.83018484e-03,
         7.93854596e-05],
       [-2.07571864e-01, -3.19785705e-02,  1.00000000e+00,
        -1.89866388e-02,  5.42647565e-03,  2.65137050e-03,
         9.78090015e-03,  1.74692010e-02,  1.91878088e-02,
        -4.82046577e-03],
       [-7.26964164e-03,  7.49284047e-03, -1.89866388e-02,
         1.00000000e+00, -1.22281391e-02,  1.03153865e-02,
         1.00140043e-02,  1.62642899e-02,  1.68757381e-02,
         7.16853618e-01],
       [-1.93691115e-02, -6.66509086e-03,  5.42647565e-03,
        -1.22281391e-02,  1.00000000e+00, -1.31150021e-02,
         1.25343770e-03, -1.24648166e-02, -1.43451090e-02,
        -9.82497614e-03],
       [ 1.49341976e-03, -6.38920078e-03,  2.65137050e-03,
         1.03153865e-02, -1.31150021e-02,  1.00000000e+00,
        -7.20322011e-03, -3.64179059e-03, -1.40327700e-03,
         8.28999258e-03],
       [ 1.28546987e-03, -1.96100786e-03,  9.78090015e-03,
         1.00140043e-02,  1.25343770e-03, -7.20322011e-03,
         1.00000000e+00,  1.59742242e-02,  1.69240456e-02,
         1.03273396e-02],
       [-8.82047689e-03, -9.29233922e-03,  1.74692010e-02,
         1.62642899e-02, -1.24648166e-02, -3.64179059e-03,
         1.59742242e-02,  1.00000000e+00,  9.87640266e-01,
         4.40888290e-03],
       [-1.07586076e-02, -8.83018484e-03,  1.91878088e-02,
         1.68757381e-02, -1.43451090e-02, -1.40327700e-03,
         1.69240456e-02,  9.87640266e-01,  1.00000000e+00,
         2.92558240e-02],
       [-2.28269517e-03,  7.93854596e-05, -4.82046577e-03,
         7.16853618e-01, -9.82497614e-03,  8.28999258e-03,
         1.03273396e-02,  4.40888290e-03,  2.92558240e-02,
         1.00000000e+00]])
```

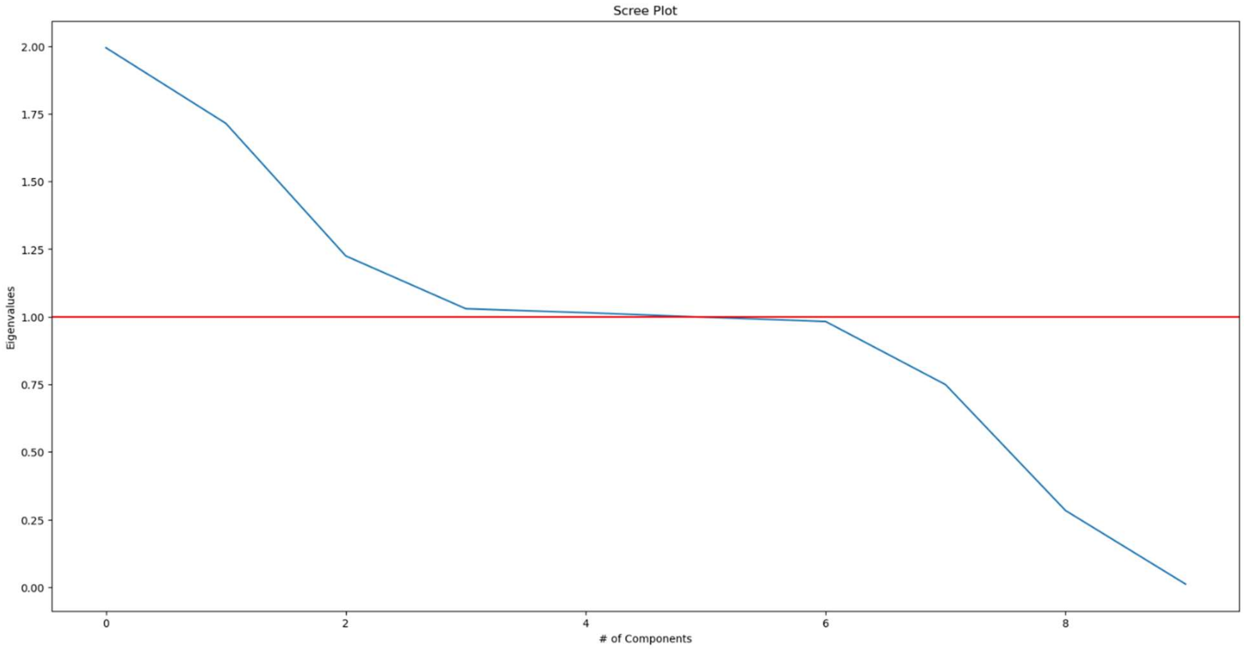


```
#list eigenvalue values
eigenvalues
```

```
[1.9938253123001637,
 1.7142046461707607,
 1.2242624069904822,
 1.0293077180607684,
 1.0145619025950303,
 0.9978784530636614,
 0.9820470715708928,
 0.7487377374614067,
 0.2834582608963168,
 0.011716490890515614]
```



```
#Scree plot of Eigenvalues
plt.figure(figsize = [20,10])
plt.plot(eigenvalues)
plt.title('Scree Plot')
plt.xlabel('# of Components')
plt.ylabel('Eigenvalues')
plt.axhline(y=1, color='red')
plt.show()
```

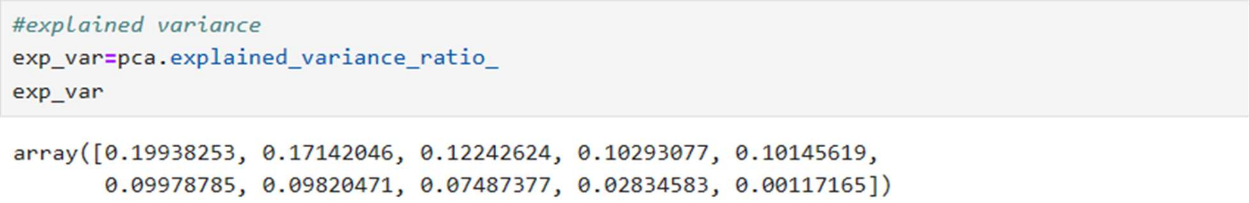


Based on the plot and listed eigenvalues, PC1, PC2, PC3, PC4, and PC5 all have eigenvalues greater than 1, indicating they are the most important and thus most relevant in explaining the variance in the dataset.

Please see the attached ipynb and pdf titled KMoikD212Code2 for the full code.

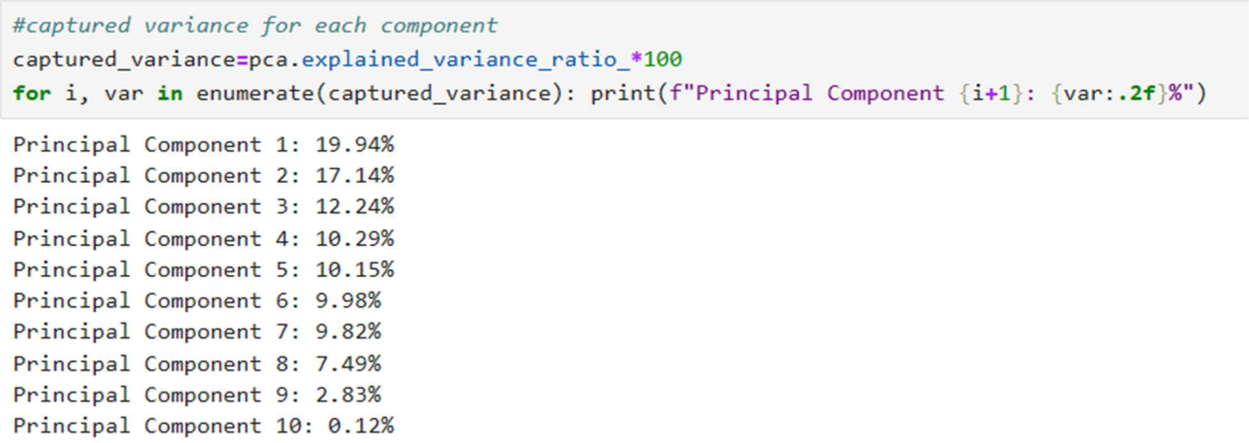
D3. Variance of Each Component

PC1 has an explained variance of 19.94%, meaning it explains 19.94% of the variances in the dataset. PC2 accounts for 17.14%, PC3 accounts for 12.24%, PC4 accounts for 10.29%, and PC5 accounts for 10.15%. The explained variance of all of the PCs is also included in the screenshot of my code below (Kumar, 2023):



```
#explained variance
exp_var=pca.explained_variance_ratio_
exp_var
```

```
array([0.19938253, 0.17142046, 0.12242624, 0.10293077, 0.10145619,
       0.09978785, 0.09820471, 0.07487377, 0.02834583, 0.00117165])
```



```
#captured variance for each component
captured_variance=pca.explained_variance_ratio_*100
for i, var in enumerate(captured_variance): print(f"Principal Component {i+1}: {var:.2f}%")
```

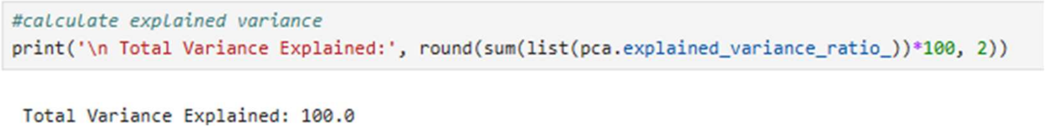
```
Principal Component 1: 19.94%
Principal Component 2: 17.14%
Principal Component 3: 12.24%
Principal Component 4: 10.29%
Principal Component 5: 10.15%
Principal Component 6: 9.98%
Principal Component 7: 9.82%
Principal Component 8: 7.49%
Principal Component 9: 2.83%
Principal Component 10: 0.12%
```

Please see the attached ipynb and pdf titled KMoikD212Code2 for the full code.

D4. Total Variance Captured by Components

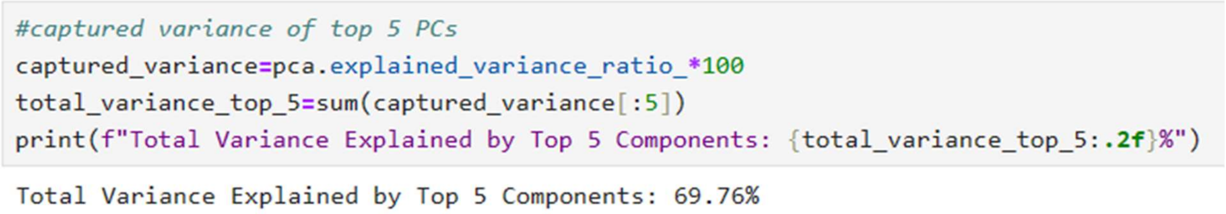
Identify the total variance captured by the principal components identified in part D2.

All of the PCs account for 100% of the variance. The first 5 PCs that were identified via eigenvalues as most significant account for 69.76% (Western Governors University, n.d.).



```
#calculate explained variance
print('\n Total Variance Explained:', round(sum(list(pca.explained_variance_ratio_))*100, 2))
```

```
 Total Variance Explained: 100.0
```



```
#captured variance of top 5 PCs
captured_variance=pca.explained_variance_ratio_*100
total_variance_top_5=sum(captured_variance[:5])
print(f"Total Variance Explained by Top 5 Components: {total_variance_top_5:.2f}%")
```

```
Total Variance Explained by Top 5 Components: 69.76%
```

Please see the attached ipynb and pdf titled KMoikD212Code2 for the full code.

D5. Summary of Data Analysis

My analysis determined that the first 5 Principal Components explained 69.76% of the variances within the dataset, meaning my data has been substantially reduced while still explaining a high percentage of the total variance.

Part V: Attachments

E. Sources for Third-Party Code

Towards Data Science. (2020). *Dealing with Highly Dimensional Data using Principal Component Analysis (PCA)*. Towards Data Science. https://towardsdatascience.com/dealing-with-highly-dimensional-data-using-principal-component-analysis-pca-fea1ca817fe6

Western Governors University. (n.d.). *PCA With Data Mining II - D212*. WGU. [Video]. https://wgu.hosted.panopto.com/Panopto/Pages/Viewer.aspx?id=a7592d75-cc72-47fd-8a9a-b07a00efff97

Kumar, Ajitesh. (2023). *PCA Explained Variance Concepts with Python Example*. Analytics Yogi. https://vitalflux.com/pca-explained-variance-concept-python-example/

F. Sources

Towards Data Science. (2020). *Dealing with Highly Dimensional Data using Principal Component Analysis (PCA)*. Towards Data Science. https://towardsdatascience.com/dealing-with-highly-dimensional-data-using-principal-component-analysis-pca-fea1ca817fe6

Keboola. (2022). *A Guide to Principal Component Analysis (PCA) for Machine Learning*. Keboola. https://www.keboola.com/blog/pca-machine-learning

Medium. (2021). *Feature Scaling with Scikit-Learn for Data Science*. Medium. https://hersanyagci.medium.com/feature-scaling-with-scikit-learn-for-data-science-8c4cbcf2daff

Kumar, Ajitesh. (2023). *PCA Explained Variance Concepts with Python Example*. Analytics Yogi. https://vitalflux.com/pca-explained-variance-concept-python-example/

G. Professional Communication

Demonstrate professional communication in the content and presentation of your submission.