D212 Data Mining II Performance Task 2

Sean Simmons

WDU Data Analytics

MSDA D212

February 2023

“Scenario 1

One of the most critical factors in customer relationship management that directly affects a company’s long-term profitability is understanding its customers. When a company can better understand its customer characteristics, it is better able to target products and marketing campaigns for customers, resulting in better profits for the company in the long term.

You are an analyst for a telecommunications company that wants to better understand the characteristics of its customers. You have been asked to use principal component analysis (PCA) to analyze customer data to identify the principal variables of your customers, ultimately allowing better business and strategic decision-making.”

Part I: Research Question

A. Describe the purpose of this data mining report by doing the following:

   1. Propose **one** question relevant to a real-world organizational situation that you will answer by using principal component analysis (PCA).

   2. Define **one** goal of the data analysis. Ensure that your goal is reasonable within the scope of the scenario and is represented in the available data.

1. One question relevant to their telecommunication organization that I will answer using PCA is, “Using PCA, can we reduce the variables given in the dataset and find the principal variables that can be used for predictions?” This question informs customer retention and demographic efforts which are key metrics this organization wants to know.

2. One goal of this analysis is to identify the principal components of this dataset to gain a better understanding of the customers. This goal is reasonable in scenario one because identifying the characteristics of the customers and how to predict them inform the organization of one of their needs. The scenario describes wanting to learn more about the demographics of the customers and gaining an ability to understand characteristics, therefore, finding the principal components of this dataset will be very helpful. Also, it will inform the retention efforts of the company since identifying and grouping the dataset into the most important variables will aid in prediction.

Part II: Method Justification

B. Explain the reasons for using PCA by doing the following:

1. Explain how PCA analyzes the selected data set. Include expected outcomes.

2. Summarize **one** assumption of PCA.

1. Principal Component Analysis (PCA) is a statistical method used to transform a very large dataset into a smaller number of variables that may have some linear correlation. In this way, PCA reduces large datasets to the variables that are most important. The PCA

analyzes the churn_clean dataset by taking the entire dataset and analyzing them for the strongest linear relationships and using two key metrics. Variance is used to identify the strongest correlation (closest to 1) and loadings are used to identify how much a variable correlates with a component (*Principal Component Analysis with Python - GeeksforGeeks* 2018). The expected outcome is a list of the most important principal components (PC's) and the total variance for each of these components, giving us metrics to inform the organization of that need for more resources and support. We also expect to see the eigenvalues of the components and the explained variance for each of the components.

2. One assumption of PCA is there is linearity in the dataset. That is to say there is a linear relationship between all variables.

Part III: Data Preparation

C. Perform data preparation for the chosen dataset by doing the following:

1. Identify the continuous dataset variables that you will need in order to answer the PCA question proposed in part A1.

2. Standardize the continuous dataset variables identified in part C1. Include a copy of the cleaned dataset.

1. The continuous variables I will need in order to answer the question from part A1 are in the list:

Variable Name	Data Type

Outage_Sec_perweek	Continuous
Tenure	Continuous
MonthlyCharge	Continuous
Bandwidth_GB_year	Continuous
Email	Continuous
Yearly_equip_failure	Continuous
Contacts	Continuous
Children	Continuous
Age	Continuous
Income	Continuous

2. The continuous variables were standardized using sklearn’s standardization package. The cleaned dataset has been provided in “D212_Task2_Clean.csv” and now includes the standardized values.
    a. Standardization code output:

```
#Use the standardscaler package to standardize our values
num_col = churn_df.columns[churn_df.dtypes.apply(lambda c: np.issubdtype(c, np.number))]
scaler = StandardScaler()
churn_df[num_col] = scaler.fit_transform(churn_df[num_col])
#Check for scaling
print(churn_df)
```

```
      Children       Age    Income  Outage_sec_perweek     Email  Contacts  \
0    -0.972338  0.720925 -0.398778           -0.679978 -0.666282 -1.005852
1    -0.506592 -1.259957 -0.641954            0.570331 -0.005288 -1.005852
2     0.890646 -0.148730 -1.070885            0.252347 -0.996779 -1.005852
3    -0.506592 -0.245359 -0.740525            1.650506  0.986203  1.017588
4    -0.972338  1.445638  0.009478           -0.623156  1.316700  1.017588
...        ...       ...       ...                 ...       ...       ...
9995  0.424900 -1.453214  0.564456           -0.196888 -0.005288  1.017588
9996  0.890646 -0.245359 -0.201344           -1.095915  0.986203  1.017588
9997 -0.506592 -0.245359  0.219037           -1.146198 -0.666282 -1.005852
9998 -0.506592 -0.680187 -0.820588            0.695616  0.655706  0.005868
9999 -0.506592 -1.211643 -1.091760            0.589028  1.647197  0.005868

      Yearly_equip_failure    Tenure  MonthlyCharge  Bandwidth_GB_Year
0                 0.946658 -1.048746      -0.003943          -1.138487
1                 0.946658 -1.262001       1.630326          -1.185876
2                 0.946658 -0.709940      -0.295225          -0.612138
3                -0.625864 -0.659524      -1.226521          -0.561857
4                 0.946658 -1.242551      -0.528086          -1.428184
...                    ...       ...            ...                ...
9995             -0.625864  1.273401      -0.294484           1.427298
9996             -0.625864  1.002740       0.811726           1.054194
9997             -0.625864  0.487513      -0.061729           0.350984
9998             -0.625864  1.383018       1.863005           1.407713
9999             -0.625864  1.090120       1.044672           1.128163

[10000 rows x 10 columns]
```

Initial analysis from the code:

```
Floats
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10000 entries, 0 to 9999
Data columns (total 7 columns):
 #   Column              Non-Null Count  Dtype  
---  ------              --------------  -----  
 0   Lat                 10000 non-null  float64
 1   Lng                 10000 non-null  float64
 2   Income              10000 non-null  float64
 3   Outage_sec_perweek  10000 non-null  float64
 4   Tenure              10000 non-null  float64
 5   MonthlyCharge       10000 non-null  float64
 6   Bandwidth_GB_Year   10000 non-null  float64
dtypes: float64(7)
memory usage: 547.0 KB
None
Integers
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10000 entries, 0 to 9999
Data columns (total 16 columns):
 #   Column                Non-Null Count  Dtype
---  ------                --------------  -----
 0   CaseOrder             10000 non-null  int64
 1   Zip                   10000 non-null  int64
 2   Population            10000 non-null  int64
 3   Children              10000 non-null  int64
 4   Age                   10000 non-null  int64
 5   Email                 10000 non-null  int64
 6   Contacts              10000 non-null  int64
 7   Yearly_equip_failure  10000 non-null  int64
 8   Item1                 10000 non-null  int64
 9   Item2                 10000 non-null  int64
 10  Item3                 10000 non-null  int64
 11  Item4                 10000 non-null  int64
 12  Item5                 10000 non-null  int64
 13  Item6                 10000 non-null  int64
 14  Item7                 10000 non-null  int64
 15  Item8                 10000 non-null  int64
dtypes: int64(16)
memory usage: 1.2 MB
None
```

```
Objects
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10000 entries, 0 to 9999
Data columns (total 27 columns):
 #   Column            Non-Null Count  Dtype
---  ------            --------------  -----
 0   Customer_id       10000 non-null  object
 1   Interaction       10000 non-null  object
 2   UID               10000 non-null  object
 3   City              10000 non-null  object
 4   State             10000 non-null  object
 5   County            10000 non-null  object
 6   Area              10000 non-null  object
 7   TimeZone          10000 non-null  object
 8   Job               10000 non-null  object
 9   Marital           10000 non-null  object
 10  Gender            10000 non-null  object
 11  Churn             10000 non-null  object
 12  Techie            10000 non-null  object
```

PA_D209_C

```
 13  Contract          10000 non-null  object
 14  Port_modem        10000 non-null  object
 15  Tablet            10000 non-null  object
 16  InternetService   10000 non-null  object
 17  Phone             10000 non-null  object
 18  Multiple          10000 non-null  object
 19  OnlineSecurity    10000 non-null  object
 20  OnlineBackup      10000 non-null  object
 21  DeviceProtection  10000 non-null  object
 22  TechSupport       10000 non-null  object
 23  StreamingTV       10000 non-null  object
 24  StreamingMovies   10000 non-null  object
 25  PaperlessBilling  10000 non-null  object
 26  PaymentMethod     10000 non-null  object
dtypes: object(27)
memory usage: 2.1+ MB
None
```

```
Dataset Information
<bound method DataFrame.info of       CaseOrder Customer_id
Interaction  \
0             1     K409198  aa90260b-4141-4a24-8e36-b04ce1f4f77b
1             2     S120509  fb76459f-c047-4a9d-8af9-e0f7d4ac2524
2             3     K191035  344d114c-3736-4be5-98f7-c72c281e2d35
3             4      D90850  abfa2b40-2d43-4994-b15a-989b8c79e311
4             5     K662701  68a861fd-0d20-4e51-a587-8a90407ee574
...         ...         ...                                   ...
9995       9996     M324793  45deb5a2-ae04-4518-bf0b-c82db8dbe4a4
9996       9997     D861732  6e96b921-0c09-4993-bbda-a1ac6411061a
9997       9998     I243405  e8307ddf-9a01-4fff-bc59-4742e03fd24f
9998       9999     I641617  3775ccfc-0052-4107-81ae-9657f81ecdf3
9999      10000      T38070  9de5fb6e-bd33-4995-aec8-f01d0172a499

                                   UID         City State  \
0     e885b299883d4f9fb18e39c75155d990   Point Baker    AK
1     f2de8bef964785f41a2959829830fb8a  West Branch    MI
2     f1784cfa9f6d92ae816197eb175d3c71      Yamhill    OR
3     dc8a365077241bb5cd5ccd305136b05e      Del Mar    CA
4     aabb64a116e83fdc4befc1fbab1663f9   Needville    TX
...                                ...          ...   ...
9995  9499fb4de537af195d16d046b79fd20a  Mount Holly    VT
9996  c09a841117fa81b5c8e19afec2760104  Clarksville    TN
9997  9c41f212d1e04dca84445019bbc9b41c     Mobeetie    TX
9998  3e1f269b40c235a1038863ecf6b7a0df   Carrollton    GA
9999  0ea683a03a3cd544aefe8388aab16176 Clarkesville    GA

                    County    Zip       Lat        Lng  ...  MonthlyCharge  \
0     Prince of Wales-Hyder  99927  56.25100 -133.37571  ...     172.455519
1                   Ogemaw  48661  44.32893  -84.24080  ...     242.632554
2                  Yamhill  97148  45.35589 -123.24657  ...     159.947583
3                San Diego  92014  32.96687 -117.24798  ...     119.956840
4                Fort Bend  77461  29.38012  -95.80673  ...     149.948316
...                    ...    ...       ...        ...  ...            ...
9995               Rutland   5758  43.43391  -72.78734  ...     159.979400
9996            Montgomery  37042  36.56907  -87.41694  ...     207.481100
9997               Wheeler  79061  35.52039 -100.44180  ...     169.974100
9998               Carroll  30117  33.58016  -85.13241  ...     252.624000
9999             Habersham  30523  34.70783  -83.53648  ...     217.484000

     Bandwidth_GB_Year Item1 Item2  Item3  Item4  Item5 Item6 Item7 Item8
0           904.536110     5     5      5      3      4     4     3     4
```

1	800.982766	3	4	3	3	4	3	4	4
2	2054.706961	4	4	2	4	4	3	3	3
3	2164.579412	4	4	4	2	5	4	3	3
4	271.493436	4	4	4	3	4	4	4	5
...	...	...	...	...	...	...	...	...	...
9995	6511.252601	3	2	3	3	4	3	2	3
9996	5695.951810	4	5	5	4	4	5	2	5
9997	4159.305799	4	4	4	4	4	4	4	5
9998	6468.456752	4	4	6	4	3	3	5	4
9999	5857.586167	2	2	3	3	3	3	4	1

Missing Values:

```
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
Churn                     0
Outage_sec_perweek        0
Email                     0
Contacts                  0
Yearly_equip_failure      0
Techie                    0
Contract                  0
Port_modem                0
Tablet                    0
InternetService           0
Phone                     0
Multiple                  0
OnlineSecurity            0
OnlineBackup              0
DeviceProtection          0
TechSupport               0
StreamingTV               0
StreamingMovies           0
PaperlessBilling          0
PaymentMethod             0
Tenure                    0
MonthlyCharge             0
Bandwidth_GB_Year         0
item1_responses           0
item2_fixes               0
item3_replacements        0
item4_reliability         0
item5_options             0
item6_respectfulness      0
item7_courteous           0
item8_listening           0
dtype: int64
```

Part IV: Analysis

D. Perform PCA by doing the following:

1. Determine the matrix of *all* the principal components.

2. Identify the *total* number of principal components using the elbow rule or the Kaiser criterion. Include a screenshot of the scree plot.

3. Identify the variance of *each* of the principal components identified in part D2.

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

5. Summarize the results of your data analysis.

1. The matrix of all principal components and initial calculations:

```
#Creating PCA dataframe
pca = PCA(n_components=10)
Principal_components=pca.fit_transform(churn_df)
pca_df = pd.DataFrame(data = Principal_components, columns = ['PC1', 'PC2', 'PC3', 'PC4', 'PC5', 'PC6','PC7','PC8','PC9','PC10'])
print(pca_df)
```

```
           PC1       PC2       PC3       PC4       PC5       PC6       PC7  \
0    -1.536762  0.171914  1.454843  0.095288 -1.256114  0.702028 -0.110407
1    -1.658873 -0.084919 -0.961470  1.269270 -1.102248  1.130166 -0.208518
2    -0.903180 -1.078642  0.158621  0.897587 -1.617589 -0.048204 -0.389739
3    -0.940344  0.924346 -1.249369 -0.261404  0.060980 -1.970635 -0.238152
4    -1.928860  1.402067  1.159692 -0.719183 -0.136843 -0.918535  1.516884
...        ...       ...       ...       ...       ...       ...       ...
9995  1.893553 -0.657609 -0.633016 -0.221280  1.390348 -0.707759 -0.654197
9996  1.463002  0.132182 -0.794348 -0.981426  0.660400  0.110216  0.554934
9997  0.574807 -0.592765  0.701985 -0.856073 -0.011257  1.062299 -0.867484
9998  2.013656  1.085126 -1.786779  0.163483 -0.285542  0.906422 -0.271276
9999  1.553021  0.693810 -2.222608 -0.599757 -0.440660  0.089376  0.020432

           PC8       PC9      PC10
0    -0.440597  0.199732 -0.026854
1    -0.682203  1.359864 -0.038338
2     0.438785 -0.340794  0.060687
3    -0.772428 -0.564671  0.130091
4    -0.125387  0.469995 -0.056519
...        ...       ...       ...
9995  0.260375  0.646857  0.081025
9996  1.426217  0.852965 -0.024736
9997 -0.292028  0.095906 -0.087166
9998 -0.159951  0.777656 -0.068782
9999 -0.568523  1.095979 -0.033770

[10000 rows x 10 columns]
```

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

	PC1	PC2	PC3	PC4	PC5	PC6	PC7	PC8	PC9	PC10
Children	0.014135	-0.559467	-0.285319	0.141418	0.031679	-0.057721	0.287326	0.646749	-0.282399	-0.021585
Age	0.001708	0.479836	0.421944	-0.089805	-0.159621	0.125006	0.405096	0.207965	-0.578529	0.022366
Income	0.004360	-0.223932	0.267257	0.166468	0.787136	0.210454	0.294875	-0.302723	-0.090721	-0.000935
Outage_sec_perweek	0.005884	0.212260	-0.479537	0.578438	-0.025686	-0.243383	-0.001698	-0.367329	-0.442194	0.000281
Email	-0.020779	0.107067	-0.438465	-0.454312	-0.004960	-0.153997	0.686128	-0.229615	0.205475	0.000246
Contacts	0.004175	0.458770	0.013844	0.104530	0.465026	-0.550932	-0.043184	0.438267	0.254313	-0.000943
Yearly_equip_failure	0.017565	-0.143555	0.395131	0.530963	-0.368864	-0.227787	0.424544	-0.078997	0.408176	-0.000095
Tenure	0.705422	0.001851	0.021078	-0.041735	-0.004963	-0.037044	-0.004471	-0.029719	-0.022244	-0.705262
MonthlyCharge	0.040423	0.344887	-0.299619	0.329364	0.029915	0.704988	0.116154	0.244887	0.328190	-0.045755
Bandwidth_GB_Year	0.706917	-0.007922	-0.019661	-0.012803	0.004627	0.002619	-0.000835	-0.000232	0.009110	0.706784

```
#Total explained variance for all 10 principal components
print('Variance explained by all 10 principal components =',
      sum(pca.explained_variance_ratio_*100).round(3))
```

```
Variance explained by all 10 principal components = 100.0
```

```
#Explained variance for each PC in order
pca.explained_variance_ratio_ * 100
```

```
array([19.94133677, 10.53229292, 10.27451155, 10.12457321,  9.99695569,
        9.9368061 ,  9.88959678,  9.64670994,  9.60254553,  0.0546715 ])
```

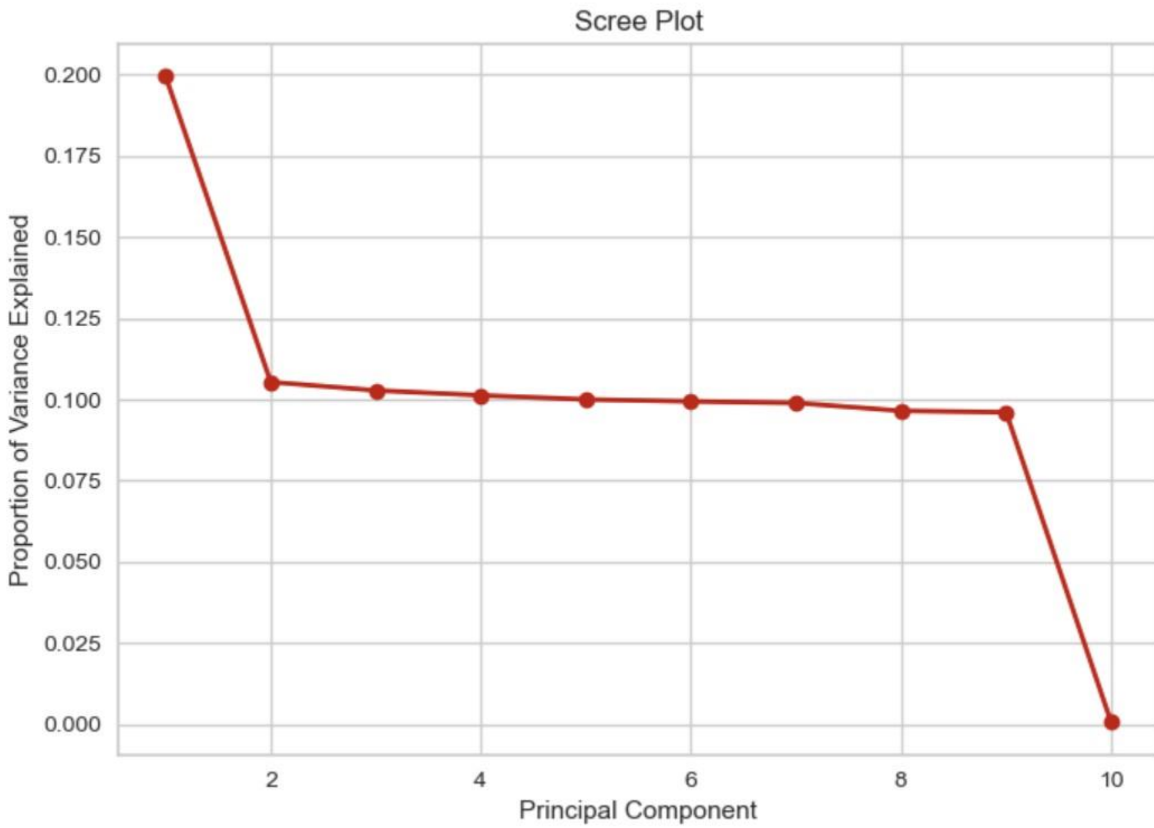
```
#Eigenvalues per PC
eigenvalues = pca.explained_variance_
eigenvalues
```

```
array([1.99433311, 1.05333463, 1.02755391, 1.01255858, 0.99979555,
       0.99377999, 0.98905858, 0.96476747, 0.96035059, 0.0054677 ])
```

```
#Cumulative sum
np.cumsum(pca.explained_variance_ratio_*100)
```

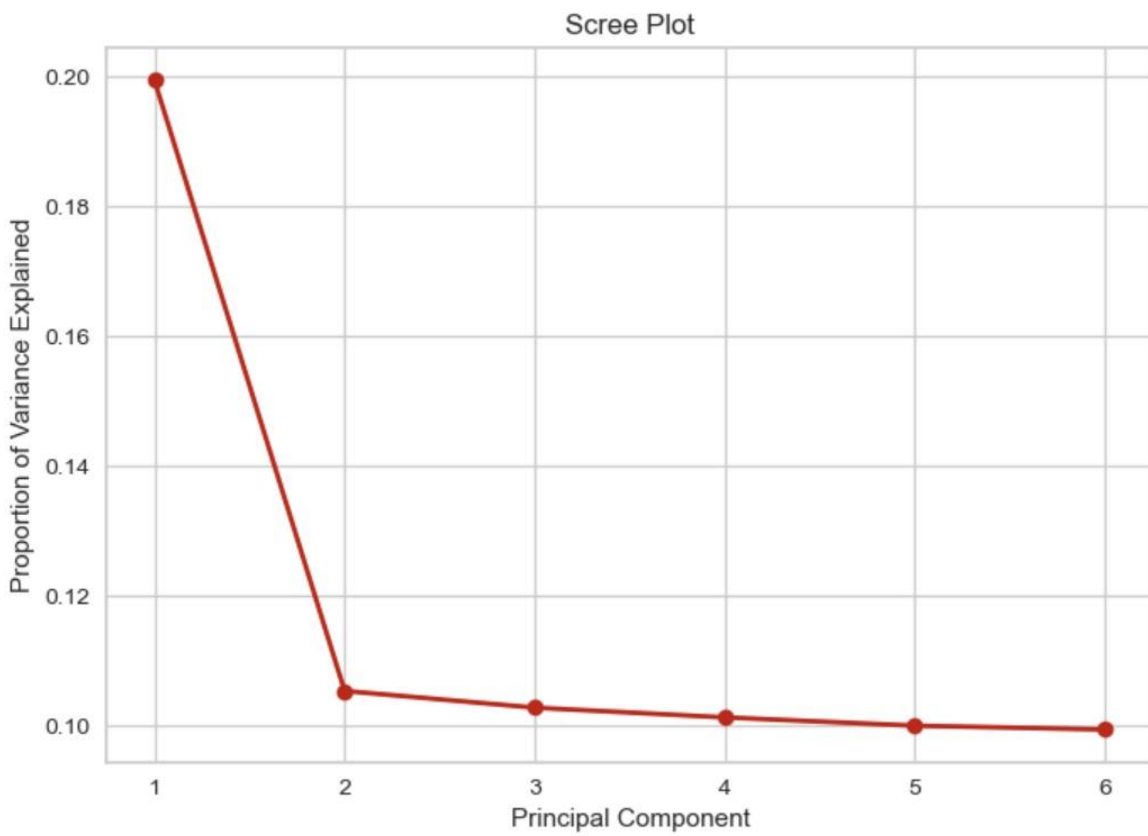
```
array([ 19.94133677,  30.47362969,  40.74814124,  50.87271445,
        60.86967014,  70.80647624,  80.69607303,  90.34278297,
        99.9453285 , 100.        ])
```

```
#Creating Scree Plot
PC_values = np.arange(pca.n_components_) + 1
mpl.plot(PC_values, pca.explained_variance_ratio_, 'ro-', linewidth=2)
mpl.title('Scree Plot')
mpl.xlabel('Principal Component')
mpl.ylabel('Proportion of Variance Explained')
mpl.show()
```



2. The total number of components were reduced from the initial 10 and found to be 6 using the Kaiser rule, as those are the PC's whose eigenvalues were equal to or above 1 (with rounding .99 to 1.0)
   a. The scree plot is shown below and the eigenvalues are included in the screenshots for the initial calculations in step 1:

```
#Final Scree Plot
PC_values = np.arange(pca.n_components_) + 1
mpl.plot(PC_values, pca.explained_variance_ratio_, 'ro-', linewidth=2)
mpl.title('Scree Plot')
mpl.xlabel('Principal Component')
mpl.ylabel('Proportion of Variance Explained')
mpl.show()
```



3. The variance of the principal components identified in part D2 are as follows, please identify the array goes from PC1 to PC6, using commas to separate them:

```
#Explained variance for each PC, 1-6
pca.explained_variance_ratio_ * 100
```

```
array([19.94133677, 10.53229292, 10.27451155, 10.12457321,  9.99695569,
        9.9368061 ])
```

4. The total variance captured by the PC's identified is:

```
#Total explained variance for the 6 PC's
print('Variance explained by all 10 principal components =',
      sum(pca.explained_variance_ratio_*100).round(3))
```

```
Variance explained by all 10 principal components = 70.806
```

5. The results of our analysis show that the best number of principal components to explain the variance is 6 PC’s. This was determined using the Kaiser rule and only keeping PC’s whose eigenvalues were equal to or above 1. Due to machine rounding, the eigenvalues at a greater than .99 values were also kept, bringing the final Principal Component list down to 6. The explained variance was reduced, but we were able to capture about half of the starting number of principal components, so I would suggest the organization weigh if the missing explained variance is acceptable. Also, I would recommend the organization to put more resources into investigating the 6 principal components found here to further understand their customers and how they can alter their business model to increase key metrics they may look for.

Part V: Attachments

E. Record the web sources used to acquire data or segments of third-party code to support the analysis. Ensure the web sources are reliable.

References

Principal Component Analysis with Python - GeeksforGeeks. (2018, October 3). GeeksforGeeks. https://www.geeksforgeeks.org/principal-component-analysis-with-python/

F. Acknowledge sources, using in-text citations and references, for content that is quoted, paraphrased, or summarized.

References

Principal Component Analysis with Python - GeeksforGeeks. (2018, October 3). GeeksforGeeks. https://www.geeksforgeeks.org/principal-component-analysis-with-python/

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

This aspect of the rubric is evaluated through the entirety of this report and I hope professionalism has shown continuously.