Dimension Reduction of Telcom Customer Churn Data

Mike Mattinson

Western Governors University

D212: Data Mining II

Task 2: Dimension Reduction

Dr. Kesselly Kamara

May 18, 2022

Revision 7

Abstract

Telecom customer data will be analyzed for feature selection using principal component analysis (PCA). The dataset consists of 50 features associated with 10,000 customer records. The analysis looked at the 50 features and select 13 continuous features to be used by the PCA. PCA results indicate that 11 principal components are needed to explained approx. 95% of the total variance. 10 principal components are needed to explain approx. 85% of total variance. Eigenvalue analysis suggests using only the first six (6) PCs. The first principal component (PC1) has high correlation to the original features of Tenure and Bandwidth_GB_Year. The second principal component (PC2) has high correlation to population and location.

Keywords: Telecom. Churn. Data Mining. Dimension Reduction. PCA. Principal Components. Scree Plot. Load Factors.

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.

List of Tables

List of Figures

Part I: Research Question

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

A1. Propose one question relevant to a real-world organizational situation that you will answer using principal component analysis (PCA)

What are the best metrics when conducting dimension reduction using principal component analysis (PCA)?

A2. 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.

Reduce high dimension dataset (over 10) to a simpler, lower dimension dataset (less than 10) using principal component analysis (PCA).

Part II: Method Justification

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

B1. Explain how the PCA analyzes the selected the selected dataset. Include expected outcomes.

According to Vadapalli (2020), “Principal component analysis (PCA) is a statistical method used to transform a large number of possibly correlated variables into a much smaller number of uncorrelated variables referred to as principal components. PCA can be used as a data reduction technique as it allows us to find the most important variables that are needed to describe a dataset. PCA can also be used to reduce the dimensionality of the data space in order to get insight on the inner structure of the data. This is helpful when dealing with large datasets.”

The expected outcome of the reduction analysis is the appropriate number of principal components and total variance for each component.

The cost or limitation of the PCA is model accuracy, you are giving up accuracy for simplicity. The PCA will determine a lower dimension dataset and the order of significance of the identified principal components so that the results can be visualized and understood by less technical team members. Often, it can be very easy to visualize a 2-dimension plot using just the first two principal components.

The PCA key metrisc are “explained variance” and “factor loading”. According to Schmalen, (Schmalen, 2020) “explained variance measures how much a model can reflect the variance of the whole data. Principle components try to capture as much of the variance as possible and this measure shows to what extent they can do that. It helps to see Components are sorted by explained variance, with the first one scoring highest and with a total sum of up to 1 across all components.” He continues, “factor loading indicates how much a variable correlates with a component." Using these two metrics and some scree plots and bar plots, the results can be verified as to how effectively the PCA accomplished the dimension reduction.

B2. Summarize one assumption of PCA.

PCA is dependent on having numeric, scaled data. If any one feature is not scaled and has large values, the PCA will give more weight to those higher values. In order to place equal weight of all features, all of the data should be scaled such that the mean of the feature is 0 and the standard deviation is 1.

Part III: Data Preparation

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

C1. Identify the continuous dataset variables that you will need in order to answer the PCA question proposed in part A1.

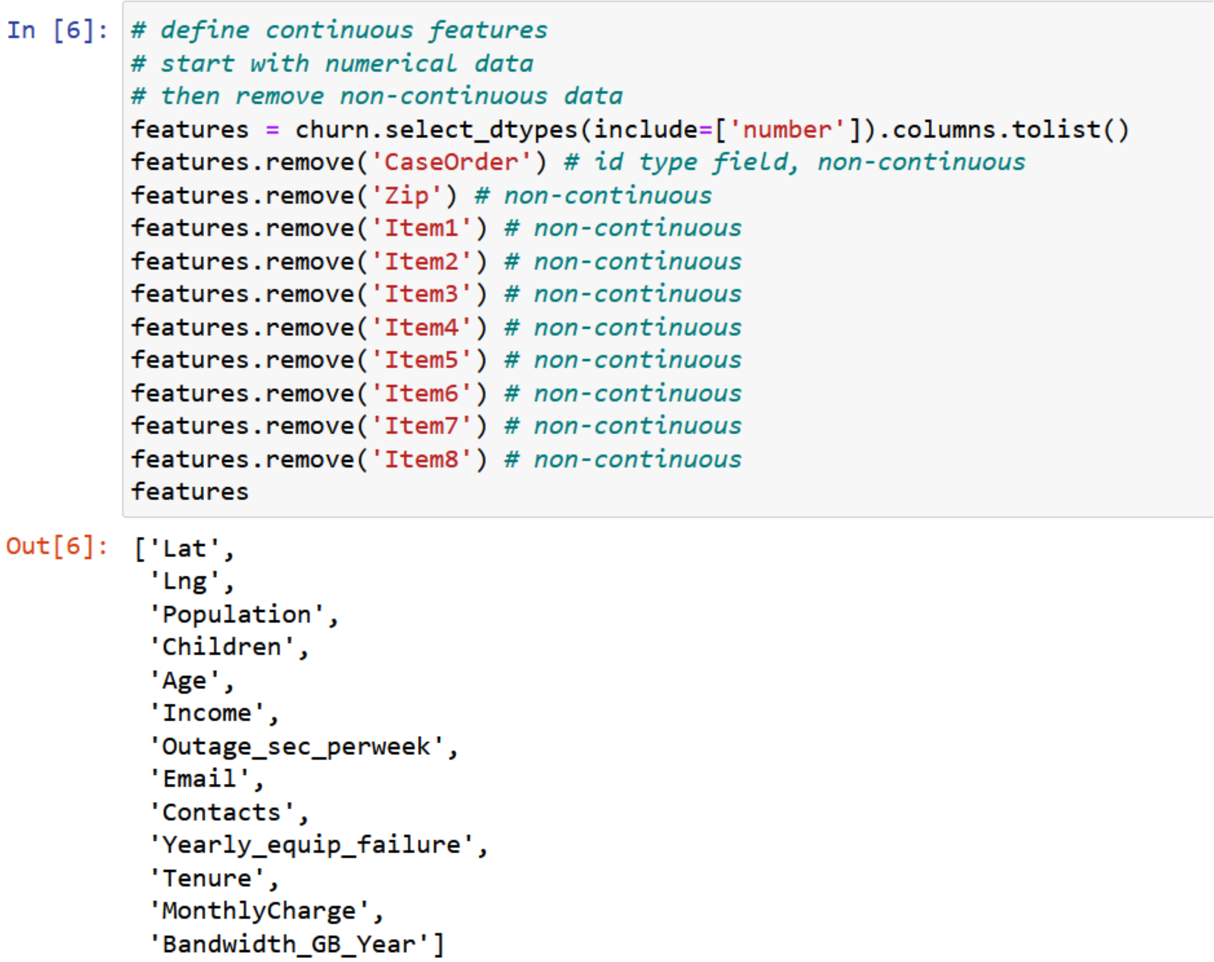
The list of continuous features that will be used for the PCA is shown in Table 1 and Table 2. They are:

- Children
- Population
- Age
- Income
- Tenure
- Email
- Contacts
- Outage_sec_perweek
- MonthlyCharge
- Bandwidth_GB_Year
- Lat
- Lng

All remaining data is either categorical or discreet and will not be used for the PCA.

Table 1

Define continuous features

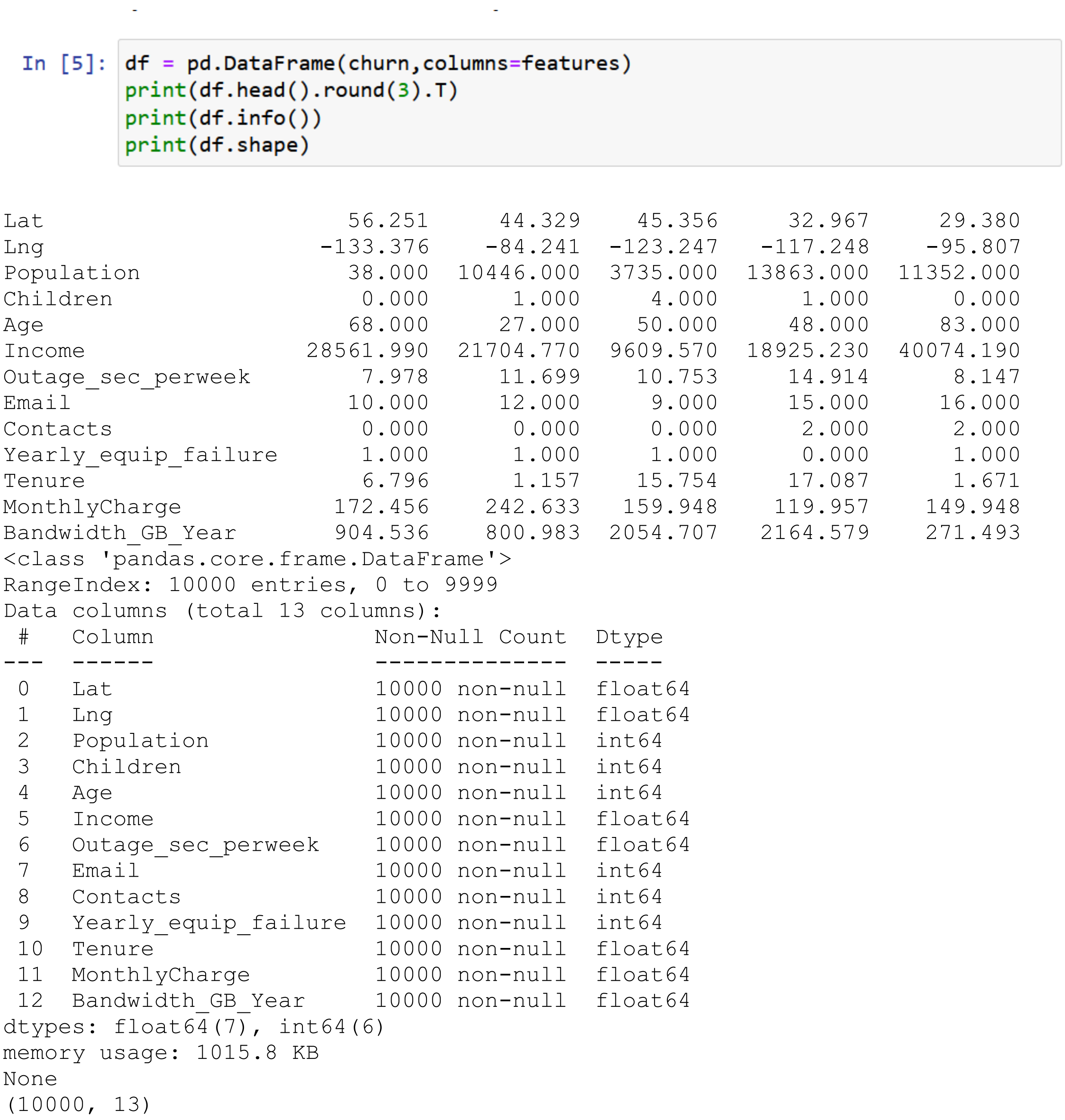


```
In [6]: # define continuous features
        # start with numerical data
        # then remove non-continuous data
        features = churn.select_dtypes(include=['number']).columns.tolist()
        features.remove('CaseOrder') # id type field, non-continuous
        features.remove('Zip') # non-continuous
        features.remove('Item1') # non-continuous
        features.remove('Item2') # non-continuous
        features.remove('Item3') # non-continuous
        features.remove('Item4') # non-continuous
        features.remove('Item5') # non-continuous
        features.remove('Item6') # non-continuous
        features.remove('Item7') # non-continuous
        features.remove('Item8') # non-continuous
        features

Out[6]: ['Lat',
         'Lng',
         'Population',
         'Children',
         'Age',
         'Income',
         'Outage_sec_perweek',
         'Email',
         'Contacts',
         'Yearly_equip_failure',
         'Tenure',
         'MonthlyCharge',
         'Bandwidth_GB_Year']
```

Notes.

Table 2

First 5 rows of the original dataset



```
In [5]: df = pd.DataFrame(churn,columns=features)
        print(df.head().round(3).T)
        print(df.info())
        print(df.shape)
```

```
Lat                      56.251     44.329     45.356     32.967     29.380
Lng                    -133.376    -84.241   -123.247   -117.248    -95.807
Population               38.000  10446.000   3735.000  13863.000  11352.000
Children                  0.000      1.000      4.000      1.000      0.000
Age                      68.000     27.000     50.000     48.000     83.000
Income                28561.990  21704.770   9609.570  18925.230  40074.190
Outage_sec_perweek        7.978     11.699     10.753     14.914      8.147
Email                    10.000     12.000      9.000     15.000     16.000
Contacts                  0.000      0.000      0.000      2.000      2.000
Yearly_equip_failure      1.000      1.000      1.000      0.000      1.000
Tenure                    6.796      1.157     15.754     17.087      1.671
MonthlyCharge           172.456    242.633    159.948    119.957    149.948
Bandwidth_GB_Year       904.536    800.983   2054.707   2164.579    271.493
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10000 entries, 0 to 9999
Data columns (total 13 columns):
 #   Column                Non-Null Count  Dtype  
---  ------                --------------  -----  
 0   Lat                   10000 non-null  float64
 1   Lng                   10000 non-null  float64
 2   Population            10000 non-null  int64  
 3   Children              10000 non-null  int64  
 4   Age                   10000 non-null  int64  
 5   Income                10000 non-null  float64
 6   Outage_sec_perweek    10000 non-null  float64
 7   Email                 10000 non-null  int64  
 8   Contacts              10000 non-null  int64  
 9   Yearly_equip_failure  10000 non-null  int64  
 10  Tenure                10000 non-null  float64
 11  MonthlyCharge         10000 non-null  float64
 12  Bandwidth_GB_Year     10000 non-null  float64
dtypes: float64(7), int64(6)
memory usage: 1015.8 KB
None
(10000, 13)
```

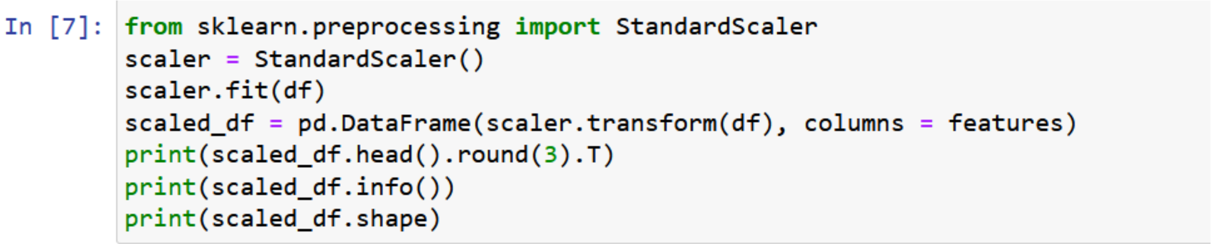
Notes. Using the original dataset, it can be seen that there are 13 numerical features that comprise the dataset for the company’s 10,000 customer records. This will be the basis of the PCA.

C2. Standardize the continuous dataset variables identified in part C1. Include a copy of the cleaned dataset.

Scaled data is critical to the PCA. If not scaled properly, larger values will tend to dominate the PCA. Table 2 shows the first five rows of the scaled data. The StandardScaler was used to fit and transform the raw data.

Table 3

First 5 rows of the scaled data



```
In [7]: from sklearn.preprocessing import StandardScaler
        scaler = StandardScaler()
        scaler.fit(df)
        scaled_df = pd.DataFrame(scaler.transform(df), columns = features)
        print(scaled_df.head().round(3).T)
        print(scaled_df.info())
        print(scaled_df.shape)
```

```
                          0      1      2      3      4
Lat                   3.217  1.025  1.214 -1.065 -1.725
Lng                  -2.810  0.432 -2.142 -1.746 -0.332
Population           -0.673  0.048 -0.417  0.285  0.111
Children             -0.972 -0.507  0.891 -0.507 -0.972
Age                   0.721 -1.260 -0.149 -0.245  1.446
Income               -0.399 -0.642 -1.071 -0.741  0.009
Outage_sec_perweek   -0.680  0.570  0.252  1.651 -0.623
Email                -0.666 -0.005 -0.997  0.986  1.317
Contacts             -1.006 -1.006 -1.006  1.018  1.018
Yearly_equip_failure  0.947  0.947  0.947 -0.626  0.947
Tenure               -1.049 -1.262 -0.710 -0.660 -1.243
MonthlyCharge        -0.004  1.630 -0.295 -1.227 -0.528
Bandwidth_GB_Year    -1.138 -1.186 -0.612 -0.562 -1.428
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10000 entries, 0 to 9999
Data columns (total 13 columns):
 #   Column                Non-Null Count  Dtype
---  ------                --------------  -----
 0   Lat                   10000 non-null  float64
 1   Lng                   10000 non-null  float64
 2   Population            10000 non-null  float64
 3   Children              10000 non-null  float64
 4   Age                   10000 non-null  float64
 5   Income                10000 non-null  float64
 6   Outage_sec_perweek    10000 non-null  float64
 7   Email                 10000 non-null  float64
 8   Contacts              10000 non-null  float64
 9   Yearly_equip_failure  10000 non-null  float64
 10  Tenure                10000 non-null  float64
 11  MonthlyCharge         10000 non-null  float64
 12  Bandwidth_GB_Year     10000 non-null  float64
dtypes: float64(13)
memory usage: 1015.8 KB
None
(10000, 13)
```

Notes. The data is available as an attached file located at tables\scaled_df.csv. This is the final scaled dataset, it can be seen that we have the same 13 features from the original data, but now scaled and ready to be used in PCA.

Part IV: Analysis

D. Perform the data analysis and report on the results by doing the following:

D1. Determine the matrix of all the principal components.

Figure 2 shows the complete set of principal components and the factor loadings to the original features. It is called a correlation matrix with load factors and it shows how much each feature contributes towards each principal component.

D2. Identify the total number of principal components using the elbow rule or the Kaiser criterion. Include a screenshot of the scree plot.

Figure 3 shows the scree plot for all principal components showing the cumulative sum of total explained variance for each of the principal components. The plot also has lines to indicate 85%, 90% and 95% of total explained variance. The scree plot can be effectively used to select the desired number of principal components to use in the final PCA model.

From the scree plot in Figure 3, and from the eigenvalue data, the appropriate number of components is somewhere between 6 and 10.

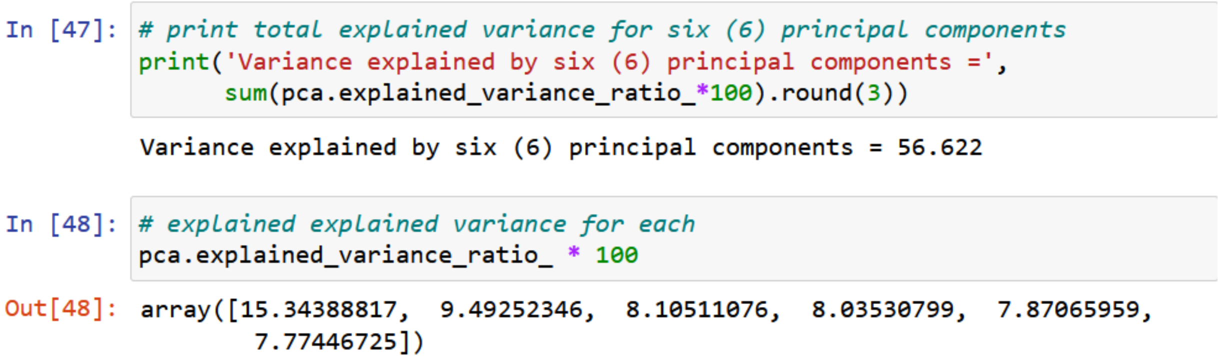
From the Eigenvalue calculations, you should select PCs with eigenvalues greater than 1, so PC6 would be the cutoff.

Table 4 and Figure 3 shows the PCA for six (6) principal components with the loadings to original features.

D3. Identify the variance of each of the principal components identified in part D2.

Table 5 shows the total variance for each of the principal components.

D4. Identify the total variance captured by the principal components identified in part D2.

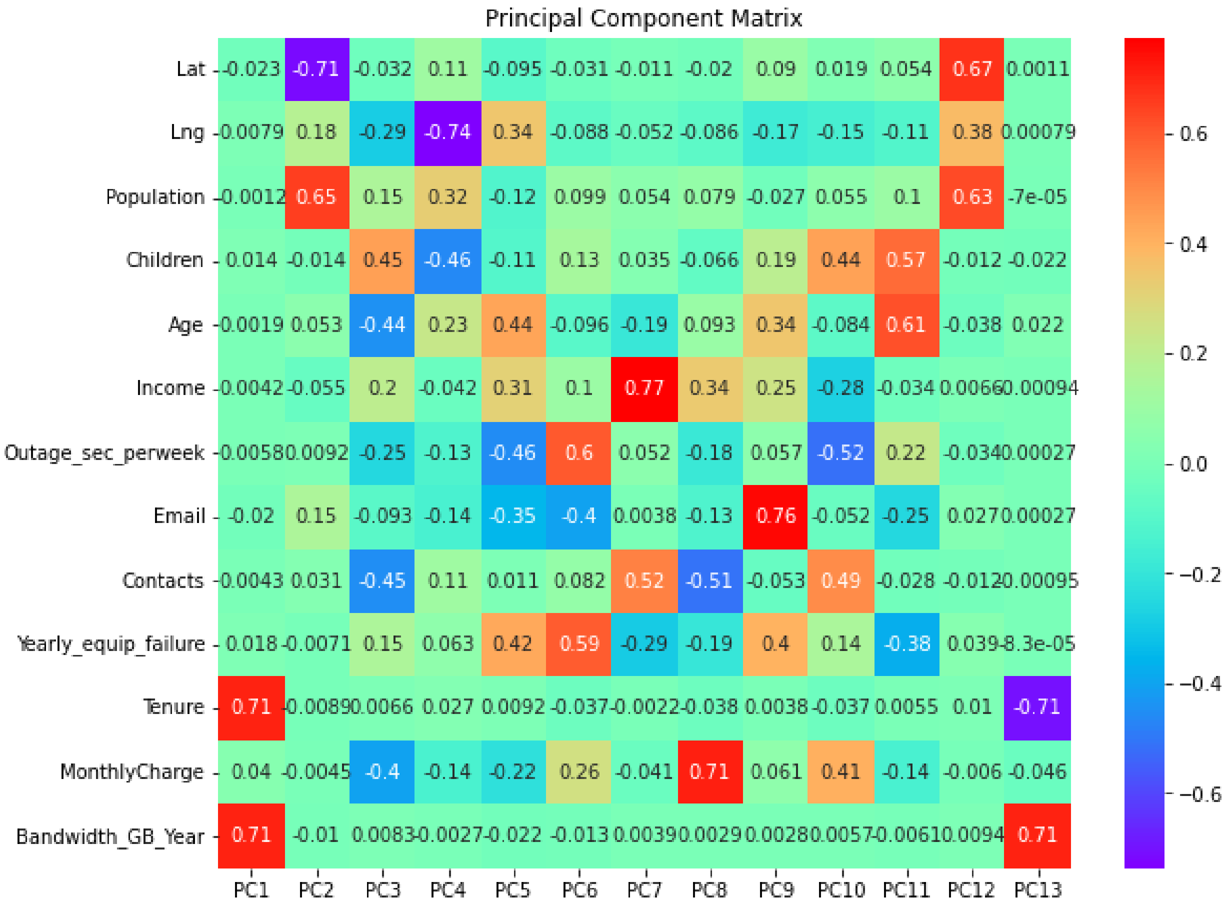


```
In [47]: # print total explained variance for six (6) principal components
         print('Variance explained by six (6) principal components =',
               sum(pca.explained_variance_ratio_*100).round(3))

         Variance explained by six (6) principal components = 56.622

In [48]: # explained explained variance for each
         pca.explained_variance_ratio_ * 100

Out[48]: array([15.34388817,  9.49252346,  8.10511076,  8.03530799,  7.87065959,
                 7.77446725])
```

The total explained variance for six (6) principal components is 56.6. Figure 4 shows a bar plot of total variance for each principal component as well as a line plot showing the cumulative total variance for all of the PCs up to that point.

Figure 1

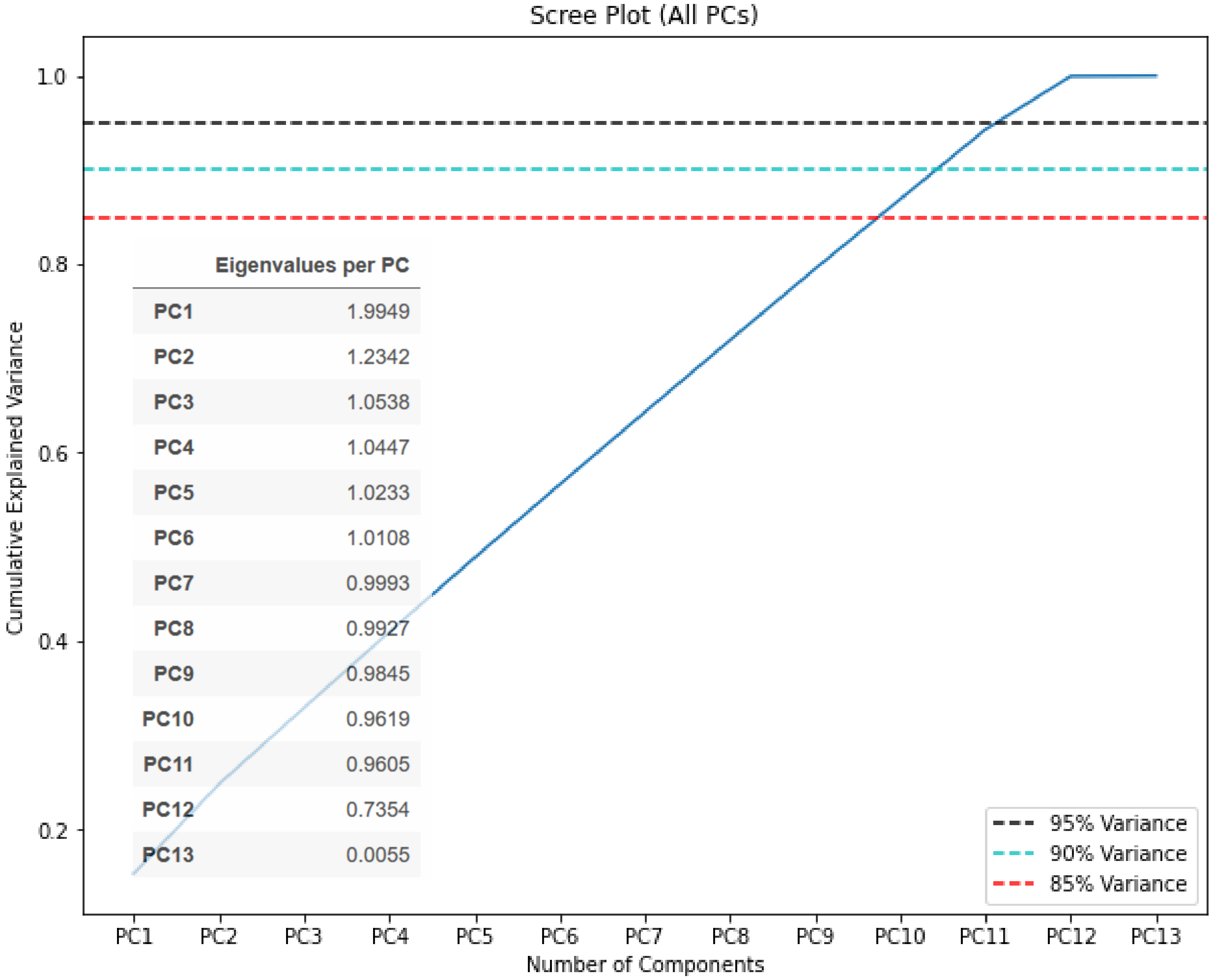
Correlation matrix of all 13 principal components with loadings to original features



Notes. The correlation matrix for the initial PCA using all 13 PCs. Load factors or +/- .7 indicate a high load factor for that component.

Figure 2

Scree plot to identify explained variance for each principal component



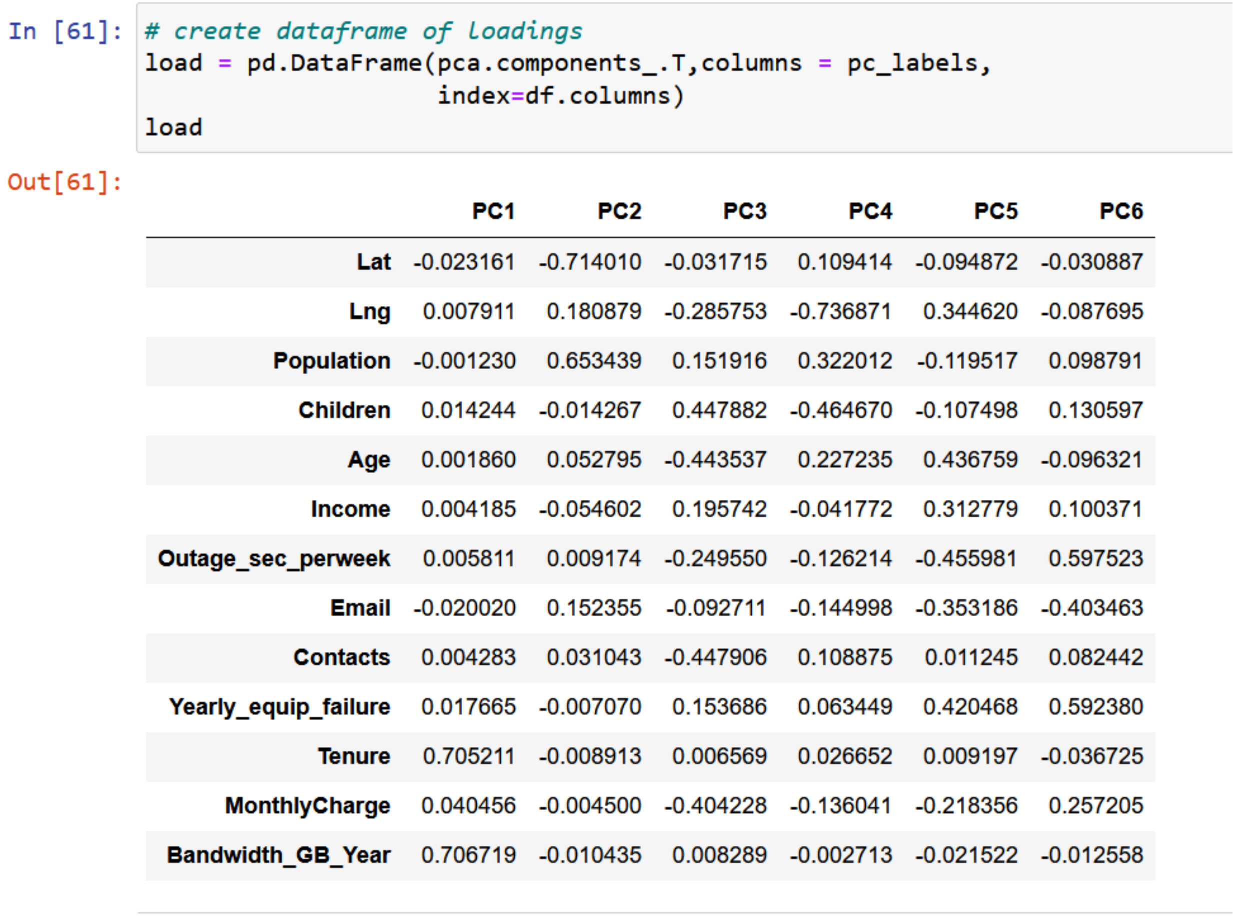
	Eigenvalues per PC
PC1	1.9949
PC2	1.2342
PC3	1.0538
PC4	1.0447
PC5	1.0233
PC6	1.0108
PC7	0.9993
PC8	0.9927
PC9	0.9845
PC10	0.9619
PC11	0.9605
PC12	0.7354
PC13	0.0055

Notes. The scaled dataset used for the initial PCA is using all of the original 13 features, so the scree plot should have 13 principal components, PC1 through PC13. It looks like it will take about 10 or 11 PCs to account for a 90% total variance. But, from the Eigenvalue data, select PCs with eigenvalues greater than 1, so PC1 through PC6.

```
Each PC: [15.344   9.493   8.105   8.035   7.871   7.774   7.686   7.635   7.573
7.398
  7.388   5.656   0.042]

Cumulative Sum: [ 15.344   24.836   32.942   40.977   48.847   56.622   64.308
71.943   79.516
  86.914   94.302   99.958 100.    ]
```

Table 4

Loadings for each PC to original features



In [61]:

```
# create dataframe of loadings
load = pd.DataFrame(pca.components_.T,columns = pc_labels,
                    index=df.columns)
load
```

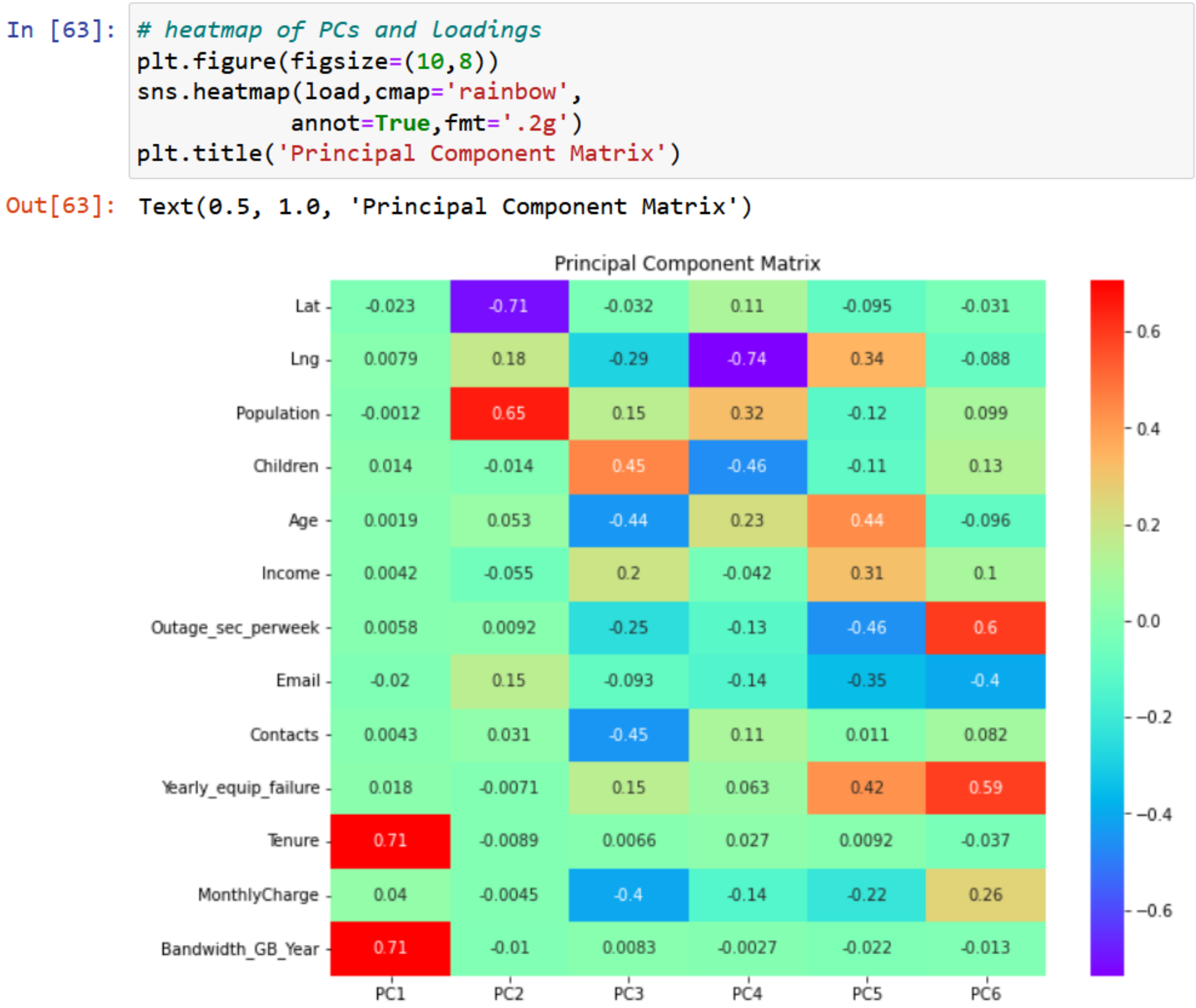
Out[61]:

	PC1	PC2	PC3	PC4	PC5	PC6
Lat	-0.023161	-0.714010	-0.031715	0.109414	-0.094872	-0.030887
Lng	0.007911	0.180879	-0.285753	-0.736871	0.344620	-0.087695
Population	-0.001230	0.653439	0.151916	0.322012	-0.119517	0.098791
Children	0.014244	-0.014267	0.447882	-0.464670	-0.107498	0.130597
Age	0.001860	0.052795	-0.443537	0.227235	0.436759	-0.096321
Income	0.004185	-0.054602	0.195742	-0.041772	0.312779	0.100371
Outage_sec_perweek	0.005811	0.009174	-0.249550	-0.126214	-0.455981	0.597523
Email	-0.020020	0.152355	-0.092711	-0.144998	-0.353186	-0.403463
Contacts	0.004283	0.031043	-0.447906	0.108875	0.011245	0.082442
Yearly_equip_failure	0.017665	-0.007070	0.153686	0.063449	0.420468	0.592380
Tenure	0.705211	-0.008913	0.006569	0.026652	0.009197	-0.036725
MonthlyCharge	0.040456	-0.004500	-0.404228	-0.136041	-0.218356	0.257205
Bandwidth_GB_Year	0.706719	-0.010435	0.008289	-0.002713	-0.021522	-0.012558

Notes. Compare table to Figure 3 (next page) of heatmaps of same data.

Figure 3

Heatmap of loadings for each PC to original feature



```
In [63]: # heatmap of PCs and Loadings
         plt.figure(figsize=(10,8))
         sns.heatmap(load,cmap='rainbow',
                     annot=True,fmt='.2g')
         plt.title('Principal Component Matrix')

Out[63]: Text(0.5, 1.0, 'Principal Component Matrix')
```

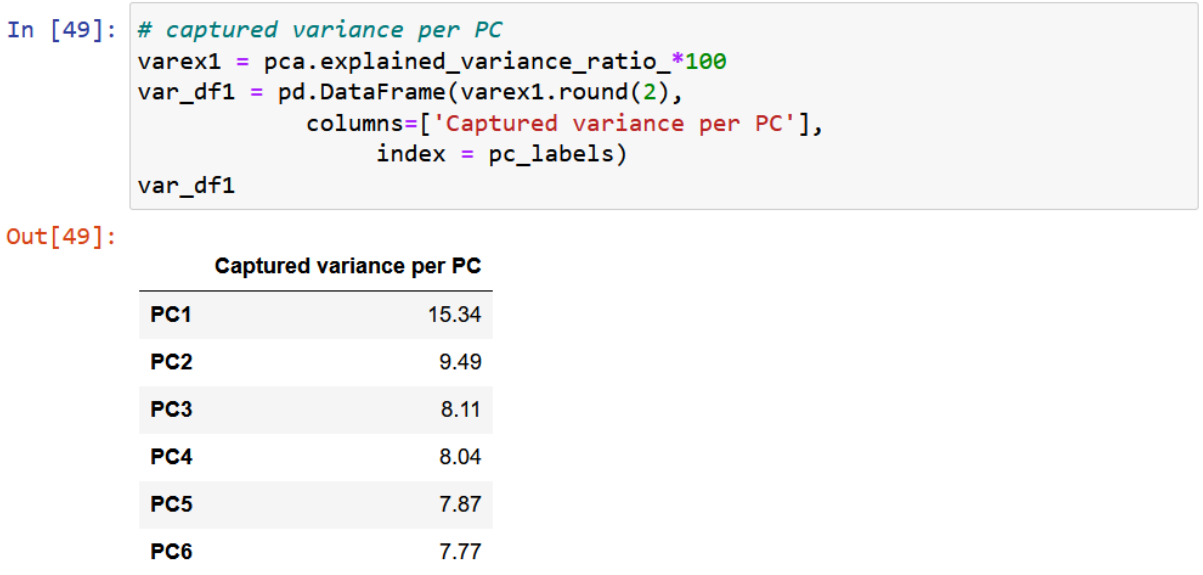
Principal Component Matrix

	PC1	PC2	PC3	PC4	PC5	PC6
Lat	-0.023	-0.71	-0.032	0.11	-0.095	-0.031
Lng	0.0079	0.18	-0.29	-0.74	0.34	-0.088
Population	-0.0012	0.65	0.15	0.32	-0.12	0.099
Children	0.014	-0.014	0.45	-0.46	-0.11	0.13
Age	0.0019	0.053	-0.44	0.23	0.44	-0.096
Income	0.0042	-0.055	0.2	-0.042	0.31	0.1
Outage_sec_perweek	0.0058	0.0092	-0.25	-0.13	-0.46	0.6
Email	-0.02	0.15	-0.093	-0.14	-0.35	-0.4
Contacts	0.0043	0.031	-0.45	0.11	0.011	0.082
Yearly_equip_failure	0.018	-0.0071	0.15	0.063	0.42	0.59
Tenure	0.71	-0.0089	0.0066	0.027	0.0092	-0.037
MonthlyCharge	0.04	-0.0045	-0.4	-0.14	-0.22	0.26
Bandwidth_GB_Year	0.71	-0.01	0.0083	-0.0027	-0.022	-0.013

Notes.

Table 5

Total variance by principal component



```
In [49]: # captured variance per PC
         varex1 = pca.explained_variance_ratio_*100
         var_df1 = pd.DataFrame(varex1.round(2),
                     columns=['Captured variance per PC'],
                          index = pc_labels)
         var_df1
```

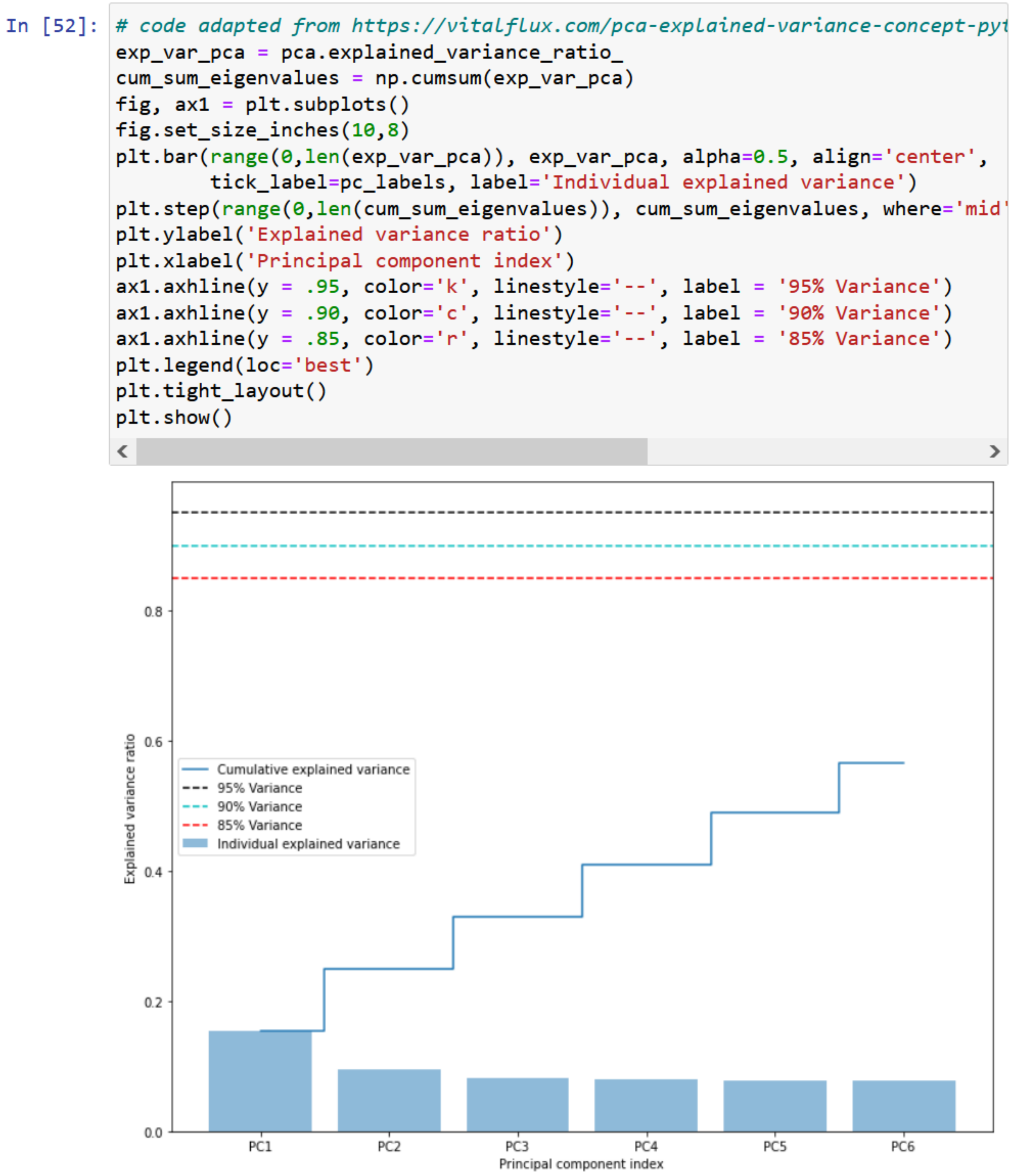
Out[49]:

	Captured variance per PC
PC1	15.34
PC2	9.49
PC3	8.11
PC4	8.04
PC5	7.87
PC6	7.77

Notes.

Figure 4

Bar plot/line plot of variance and cumulative variance for six (6) PCs



```
In [52]: # code adapted from https://vitalflux.com/pca-explained-variance-concept-pyt
         exp_var_pca = pca.explained_variance_ratio_
         cum_sum_eigenvalues = np.cumsum(exp_var_pca)
         fig, ax1 = plt.subplots()
         fig.set_size_inches(10,8)
         plt.bar(range(0,len(exp_var_pca)), exp_var_pca, alpha=0.5, align='center',
                 tick_label=pc_labels, label='Individual explained variance')
         plt.step(range(0,len(cum_sum_eigenvalues)), cum_sum_eigenvalues, where='mid'
         plt.ylabel('Explained variance ratio')
         plt.xlabel('Principal component index')
         ax1.axhline(y = .95, color='k', linestyle='--', label = '95% Variance')
         ax1.axhline(y = .90, color='c', linestyle='--', label = '90% Variance')
         ax1.axhline(y = .85, color='r', linestyle='--', label = '85% Variance')
         plt.legend(loc='best')
         plt.tight_layout()
         plt.show()
```

Notes. Code adapted from Kumar (Kumar, 2020)

D5. Summarize the results of your data analysis.

The first observation is that 12 PCs are required to achieve a total explained variance of 99.9%. That would be a dimension reduction but not quite what was hoped for.

The first 2 principal components have a total explained variance of 24.8%. A much simpler dimensional dataset, but at the extreme expense of all accuracy.

The first 3 principal components have a total explained variance of 32.9%. Not much better.

The first principal component (PC1) is highly correlated to tenure and bandwidth. Either experienced customers who prefer more bandwidth or possibly new customers with only limited bandwidth. But, the correlation for tenure and bandwidth are both above a impact threshold of 0.70.

The second principal component (PC2) is correlated to location and population count in the region nearest to each customer. There is a positive correlation to population and a negative correlation to the location.

Lastly, selecting six (6) principal components as determined by eigenvalue criteria, the total cumulative explained variance is only 56.6, not good at all. Future work should focus on getting more data or defining better continuous data to use in the PCA.

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. (see References below)

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

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

References

Albon, C. (2017). Drop Highly Correlated Features. Retrieved from https://chrisalbon.com/code/machine_learning/feature_selection/drop_highly_correlated_features/

Alkhayrat, M., Aljnidi, M., & Aljoumaa, K. (2020). A comparative dimensionality reduction study in telecom customer segmentation using deep learning and PCA. Retrieved from https://journalofbigdata.springeropen.com/articles/10.1186/s40537-020-0286-0

Arvai, K. (2022). K-Means Clustering in Python: A Practical Guide. Retrieved from https://realpython.com/k-means-clustering-python/#: :text=The SSE is defined as,try to minimize this value.&text=The purpose of this figure,centroids is an important step.

Bex, T. (2021, April 13). *How to Use Pairwise Correlation For Robust Feature Selection*. Retrieved May 8, 2022, from How to Use Pairwise Correlation For Robust Feature Selection: https://towardsdatascience.com/how-to-use-pairwise-correlation-for-robust-feature-selection-20a60ef7d10

Boyle, T. (2019, March 25). *Feature Selection and Dimensionality Reduction*. Retrieved May 8, 2022, from Feature Selection and Dimensionality Reduction: https://towardsdatascience.com/feature-selection-and-dimensionality-reduction-f488d1a035de

Brownlee, J. (2019). How to Calculate Principal Component Analysis (PCA) from Scratch in Python. Retrieved from https://machinelearningmastery.com/calculate-principal-component-analysis-scratch-python/

Bruce, P. C., Gedeck, P., Shmueli, G., & Patel, N. R. (2019). *Data Mining for Business Analytics Concepts, Techniques and Applications in Python.* Wiley & Sons, Incorporated, John.

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

Fenner, M. (2018). *Machine Learning with Python for Everyone.* Addison Wesley.

Galarnyk, M. (2017). PCA using Python (scikit-learn). Retrieved from https://towardsdatascience.com/pca-using-python-scikit-learn-e653f8989e60

Géron, A. (2019). *Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow Concepts, Tools, and Techniques to Build Intelligent Systems.* O'Reilly Media.

h1ros. (2019). Drop Highly Correlated Features. Retrieved from https://h1ros.github.io/posts/drop-highly-correlated-features/

Kaloyanova, E. (2020, March 10). *How to Combine PCA and K-means Clustering in Python?* Retrieved May 8, 2022, from How to Combine PCA and K-means Clustering in Python?: https://365datascience.com/tutorials/python-tutorials/pca-k-means/

Kumar, A. (2020, August). PCA Explained Variance Concepts with Python Example. *PCA Explained Variance Concepts with Python Example*. Retrieved May 16, 2022, from https://vitalflux.com/pca-explained-variance-concept-python-example/

Larose, C. D., & Larose, D. T. (2019). *Data Science.* Wiley.

Parveez, S., & Iriondo, R. (2018). Principal Component Analysis (PCA) with Python Examples — Tutorial. Retrieved from https://pub.towardsai.net/principal-component-analysis-pca-with-python-examples-tutorial-67a917bae9aa

ProjectPro. (2022). How to drop out highly correlated features in Python? Retrieved from https://www.projectpro.io/recipes/drop-out-highly-correlated-features-in-python

Sawlani, D. (2017). Customer Churn Prediction Model. Retrieved from https://rstudio-pubs-static.s3.amazonaws.com/344958_a3c32d38a83043c8ae8fc13cd6abbda4.html

Schmalen, P. (2020, August 16). *Understand your data with principal component analysis (PCA) and discover underlying patterns*. Retrieved May 15, 2022, from Understand your data with principal component analysis (PCA) and discover underlying patterns: https://towardsdatascience.com/understand-your-data-with-principle-component-analysis-pca-and-discover-underlying-patterns-d6cadb020939

Sharma, A. (2021). PCA or Principal Component Analysis on Customer Churn Data. Retrieved from https://medium.com/data-science-on-customer-churn-data/pca-or-principal-component-analysis-on-customer-churn-data-d18ca60397ed

Srinivas. (2022). Reducing Telecom Churn with PCA and Modeling. Retrieved from https://www.kaggle.com/code/manoharsrinivas/reducing-telecom-churn-with-pca-and-modeling/notebook

VanderPlas, J. (2016). *In Depth: Principal Component Analysis.* O'Reilly. Retrieved from https://jakevdp.github.io/PythonDataScienceHandbook/05.09-principal-component-analysis.html