Andreas Sebastian Dunn

Western Governor's University, D212: Data Mining II

February 2, 2022

Task 2: Principal Component Analysis

Part I

Learning the why behind customer decisions can provide great value to an organization. It can help them attract new customers, broaden their service catalog, and retain customers for longer periods of time. Principal Component Analysis (PCA) can aid in this understanding of customers by looking at the components that are most relevant to understanding customer needs and wants. In this paper I will use PCA in order to better understand why customers in the telecom industry decide to churn or not. Therefore our business question is as follows: which principal components are the most relevant in understanding why customers churn in the telecom industry.

The goal in this analysis to reach a better understanding of our customer base and explore the relationships between our features in answering the above research question. Our overall goal is to reach some conclusions about our customers' needs that can hopefully help us retain customers and prevent churning.

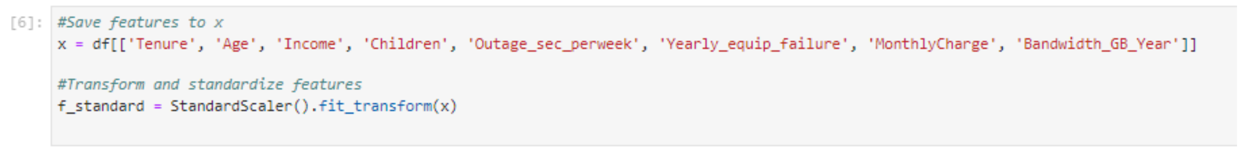
Part II

PCA will be used to extract the features that are more "principal" to understanding our customers. Due to a higher number of observations and features, it is pertinent to reduce the amount of features to those which are the most statistically significant in answering our research question. PCA is basically a way to filter out noise in the data so that we can focus our attention on those features which present the most meaningful information. The reason that PCA is important is because, as features increase in number, the manageability of data will decrease. Specifically, as more "dimensions are added to a data set, the average and minimum distance between points increase."[1]

PCA works by transforming the original feature variables into a new set of dimensions using the eigenvalues (and eigenvectors), which are calculated using a covariance matrix. Eigenvectors are essential the "special vectors" in the vector space that do not fluctuate when linear transformations occur. Eigenvalues are the factor by which the Eigenvectors are stretched during the transformation. Eigenvectors simply expand in their natural span starting from the basis vector and because eigenvectors do not change direction during linear transformation, they can be used in PCA to reduce 'noise' in the vector space. Therefore a critical assumption of PCA is that there are linear relationships between the underlying features; that the variables do have some real relationship in which can be analyzed using their variance.

Part III

For the purposes of PCA, the only relevant variables in the dataset are the continuous numeric variables which include the following: Tenure, Age, Income, Children, Outage_sec_perweek, Yearly_equip_failure, MonthlyCharge, and Bandwidth_GB_Year. Churn is a categorical variable which we will include in our analysis at the end after our PCA has been concluded using continuous variables. Below is a screenshot showing the code that standardizes the dataset to process the data for PCA.

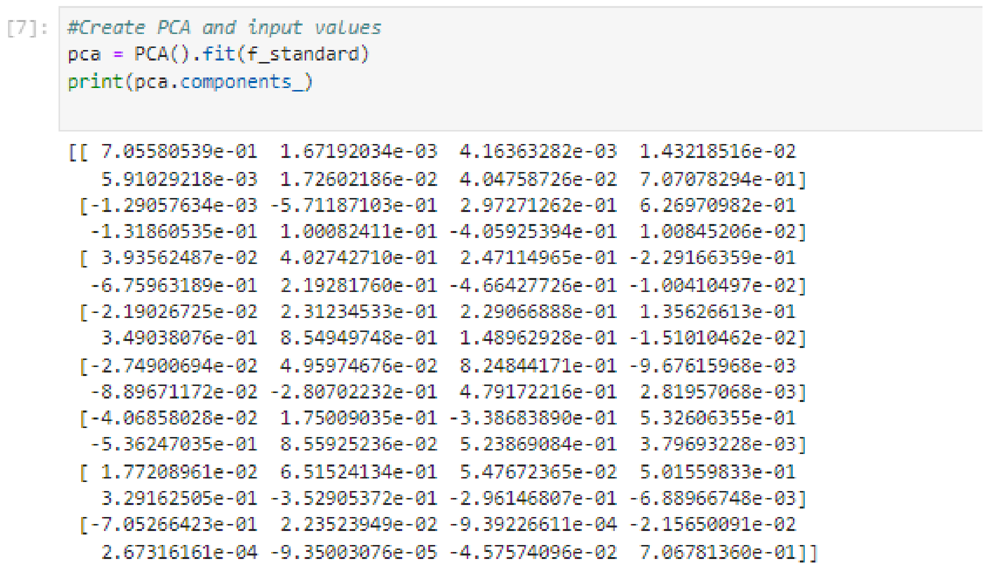


```
[6]: #Save features to x
     x = df[['Tenure', 'Age', 'Income', 'Children', 'Outage_sec_perweek', 'Yearly_equip_failure', 'MonthlyCharge', 'Bandwidth_GB_Year']]

     #Transform and standardize features
     f_standard = StandardScaler().fit_transform(x)
```

Part IV: Analysis

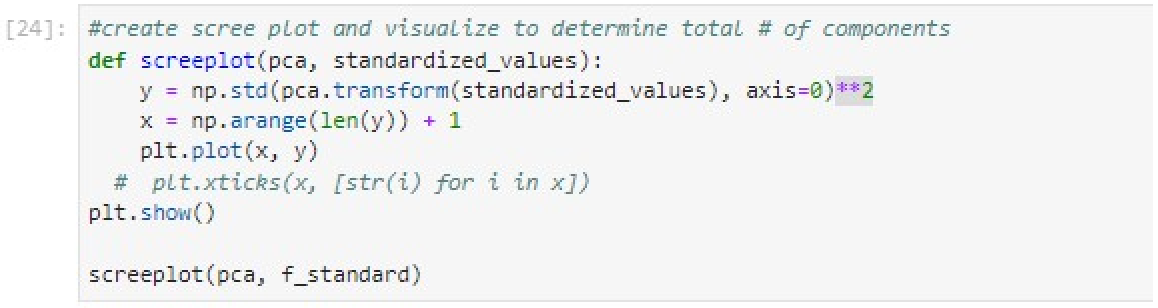
A matrix of the explained variance is provided below. The following screenshots show the matrix of all principal components by calling the .components_ parameter on the pca variable.



```
[7]: #Create PCA and input values
     pca = PCA().fit(f_standard)
     print(pca.components_)
```

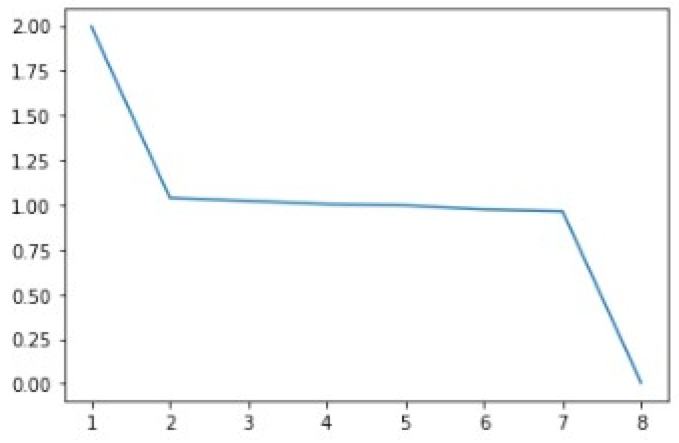
```
[[ 7.05580539e-01  1.67192034e-03  4.16363282e-03  1.43218516e-02
   5.91029218e-03  1.72602186e-02  4.04758726e-02  7.07078294e-01]
 [-1.29057634e-03 -5.71187103e-01  2.97271262e-01  6.26970982e-01
  -1.31860535e-01  1.00082411e-01 -4.05925394e-01  1.00845206e-02]
 [ 3.93562487e-02  4.02742710e-01  2.47114965e-01 -2.29166359e-01
  -6.75963189e-01  2.19281760e-01 -4.66427726e-01 -1.00410497e-02]
 [-2.19026725e-02  2.31234533e-01  2.29066888e-01  1.35626613e-01
   3.49038076e-01  8.54949748e-01  1.48962928e-01 -1.51010462e-02]
 [-2.74900694e-02  4.95974676e-02  8.24844171e-01 -9.67615968e-03
  -8.89671172e-02 -2.80702232e-01  4.79172216e-01  2.81957068e-03]
 [-4.06858028e-02  1.75009035e-01 -3.38683890e-01  5.32606355e-01
  -5.36247035e-01  8.55925236e-02  5.23869084e-01  3.79693228e-03]
 [ 1.77208961e-02  6.51524134e-01  5.47672365e-02  5.01559833e-01
   3.29162505e-01 -3.52905372e-01 -2.96146807e-01 -6.88966748e-03]
 [-7.05266423e-01  2.23523949e-02 -9.39226611e-04 -2.15650091e-02
   2.67316161e-04 -9.35003076e-05 -4.57574096e-02  7.06781360e-01]]
```

The total number of principal components are shown below in the scree plot; 8 total components are shown.

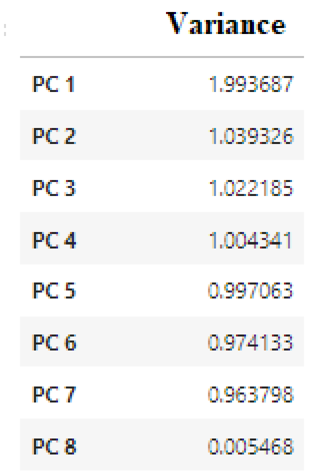


```
[24]: #create scree plot and visualize to determine total # of components
      def screeplot(pca, standardized_values):
          y = np.std(pca.transform(standardized_values), axis=0)**2
          x = np.arange(len(y)) + 1
          plt.plot(x, y)
        #  plt.xticks(x, [str(i) for i in x])
      plt.show()

      screeplot(pca, f_standard)
```

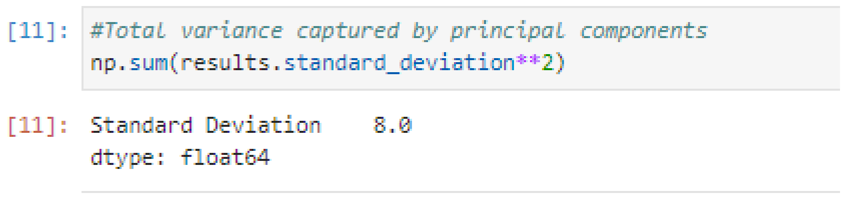


The variance of each principal component is listed below.



	Variance
PC 1	1.993687
PC 2	1.039326
PC 3	1.022185
PC 4	1.004341
PC 5	0.997063
PC 6	0.974133
PC 7	0.963798
PC 8	0.005468

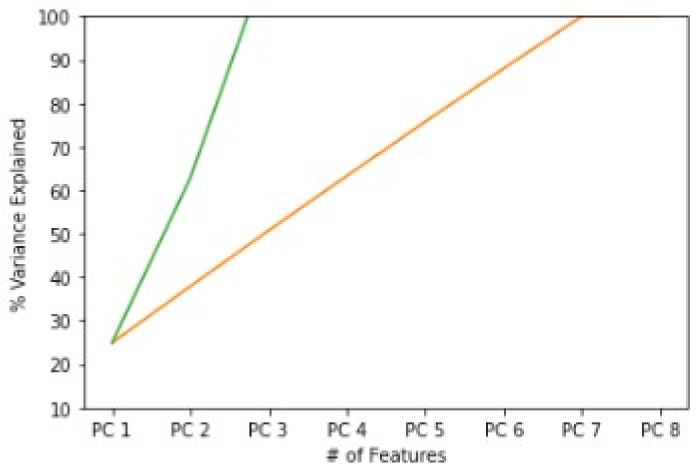
The total variance captured by the principal components is shown below (note: code displays standard deviation as default title, but as can be seen in the first argument for np.sum, I have squared the standard_deviation key in the summary object to get the variance. Therefore the value 8.0 actual means the variance, not standard deviation.



```
[11]: #Total variance captured by principal components
      np.sum(results.standard_deviation**2)

[11]: Standard Deviation    8.0
      dtype: float64
```

In summary, as can be seen below by the graph, 50% of the variance is explained by the first 3 components. This is verified above when I printed the explained_variance_ratio from the pca object. Notice the first 3 values, when added together, equals very close to 50% (0.249 + 0.129 + 0.127 = ~.50)



PCA is an important tool when looking at big data sets since it reduces the dimensionality of our data down to its most important information. After we perform a PCA, we can now use that data to increase the speed of a machine learning model or even use PCA in a data visualization project. PCA is a powerful tool in understanding how data can be stripped down to its most important elements. For our task, it can help us see which components carry the most weight in discovering the reasons why customers may churn.

Sources:

1. *https://towardsdatascience.com/tidying-up-with-pca-an-introduction-to-principal-components-analysis-f876599af383*

Web Sources:

https://www.datacamp.com/community/tutorials/principal-component-analysis-in-python

https://towardsdatascience.com/eigenvectors-and-eigenvalues-all-you-need-to-know-df92780c591f

https://www.3blue1brown.com/

All other sources from course materials.