Data Cleaning Performance Assessment

Briana Churchill
Student ID: 011009463
Western Governors University, Data Analytics M.S.
D206: Data Cleaning
Dr. Eric Straw
July 8, 2023

Table of Contents

Part I: Research Question and Variables

A. Identifying the research question

The research question I would like to ask is: "What causes churn?" Analyzing specific pertinent data within the dataset would justify an answer. There are many variables within the dataset, but the ones most relevant to the research question are: age, children, income, tenure, bandwidth, marital status, monthly charge, yearly equipment failure and customer survey responses. Answering this question would allow the business to grow strengths in positive areas, and address necessary changes in other areas to have better chances of maintaining loyal customers.

B. Identifying the variables

This dataset consists of 50 columns, which all contain separate variables relating to 10,000 individual customers. I've created tables below describing the different variables within. For the first table seen below, the following columns are unique values to maintain order and uniqueness.

Variable	Data Type	Description	Example
CaseOrder	Quantitative	This variable serves as a placeholder within the data, to ensure that all the rows in the dataset are maintained in the order as described by the original datafile.	Row 1: 1
Customer_id	Qualitative	The Customer ID column contains a unique ID to identify each individual customer.	Row 1: K409198
Interaction (UID)	Qualitative	This column is associated with a unique ID created for each individual transaction a customer has had with the business	Row 1: aa90260b-4141-4a24-8e36-b04ce1f4f77b

The following table describes variables pertaining to customer demographics. The table explains the variable and its corresponding data type, a description of the variable and an

example from the dataset. According to the data dictionary, there are different sources to the variables, and as such they will be explained individually within the description.

Variable	Data Type	Description	Example
City	Qualitative	The customer's residential city, based on the billing statement.	Row 1: Point Baker
State	Qualitative	The customer's residential state, based on the billing statement.	Row 1: AK
County	Qualitative	The county in which the customer's address is located based on the billing statement.	Row 1: Prince of Wales-Hyder
Zip	Qualitative	The customer's zip code, based on the billing statement.	Row 1: 99927
Lat	Quantitative	The latitudinal coordinate of the customer's residence, based on the billing statement.	Row 1: 56.251
Lng	Quantitative	The longitudinal coordinate of the customer's residence, based on the billing statement.	Row 1: -133.37571
Population	Quantitative	Based on census data for the customer's address, this column provides the population within a mile.	Row 1: 38
Area	Qualitative	Using census data, this column provides the customer's area type (ie: urban, suburban, rural).	Row 1 : Urban
Timezone	Qualitative	According to the customer's information retrieved upon signing up for services, this is the customer's timezone.	Row 1: America/Sitka

Job	Qualitative	According to the customer's information retrieved upon signing up for services, this is the number of children in the customer's job title.	Row 1: Environmental health practitioner
Children	Quantitative	According to the customer's information retrieved upon signing up for services, this is the number of children in the customer's household.	Row 1: NA
Age	Quantitative	According to the customer's information retrieved upon signing up for services, this is the age of the customer.	Row 1: 68
Education	Qualitative	According to the customer's information retrieved upon signing up for services, this is the highest level of schooling that the customer has completed.	Row 1: Master's Degree
Employment	Qualitative	According to the customer's information retrieved upon signing up for services, this is the customer's current employment status.	Row 1: Part Time
Income	Quantitative	According to the customer's information retrieved upon signing up for services, this is the customer's annual income.	Row 1: 28561.99
Marital	Qualitative	According to the customer's information retrieved upon signing up for services, this column contains their marital status.	Row 1: Widowed
Gender	Qualitative	According to the customer's information retrieved upon signing up for services, this column includes their gender identity. Choices	Row 1: Male

		available are: male, female or nonbinary.	

This next table contains variables relating to the account of each customer in regards to service, equipment, and more. This table includes information that may be used to assess the research question discussed earlier. Similar to the table above, there are different sources to the variables, and as such they will be explained individually within the description.

Variable	**Data Type**	**Description**	**Example**
Churn	Qualitative	A yes or no value as to whether the customer has discontinued services with the business in the last month	Row 1: No
Outage_sec_perweek	Quantitative	Average time as described in seconds per week, that a customer's neighborhood experienced outages	Row 1: 6.972566093
Email	Quantitative	A numerical value of how many emails have been sent to the customer	Row 1: 10
Contacts	Qualitative	The amount of times in which a customer has contacted technical support	Row 1: 0
Yearly_equip_failure	Quantitative	The amount of times within the past year in which a customer had to reset and/or replace their equipment due to failure	Row 1: 1
Techie	Qualitative	According to the customer's information retrieved upon signing up for services, this is a yes or no response regarding the customer's self-reported consideration of being technically inclined.	Row 1: No
Contract	Qualitative	The contract term for the customer, contract terms available are: month-to-month, one year, or two year	Row 1: One year

Port_modem	Qualitative	A yes or no value indicating whether or not the customer has a portable modem	Row 1: Yes
Tablet	Qualitative	A yes or no value indicating whether or not the customer owns a tablet device (such as ipad, etc)	Row 1: Yes
InternetService	Qualitative	The internet service provider for each customer (DSL, fiber optic, none)	Row 1: Fiber Optic
Phone	Qualitative	A yes or no value indicating whether or not the customer has a phone service	Row 1: Yes
Multiple	Qualitative	A yes or no value indicating whether or not the customer has multiple lines	Row 1: No
OnlineSecurity	Qualitative	A yes or no value indicating whether or not the customer has an online security add-on to their services	Row 1: Yes
OnlineBackup	Qualitative	A yes or no value indicating whether or not the customer has an online backup add-on to their services	Row 1: Yes
DeviceProtection	Qualitative	A yes or no value indicating whether or not the customer has a device protection add-on to their services	Row 1: No
TechSupport	Qualitative	A yes or no value indicating whether or not the customer has a tech support add-on to their services	Row 1: No
StreamingTV	Qualitative	A yes or no value indicating whether or not the customer has streaming TV	Row 1: No
StreamingMovies	Qualitative	A yes or no value indicating whether or not the customer has streaming movies	Row 1: Yes

PaperlessBilling	Qualitative	A yes or no value indicating whether or not the customer has paperless billing	Row 1: Yes
PaymentMethod	Qualitative	This column describes the type of payment method a customer utilizes to fulfill their billing statements	Row 1: Credit Card (automatic)
Tenure	Quantitative	The amount of months in which a customer has remained with the service provider	Row 1: 6.795512947
MonthlyCharge	Quantitative	An average monthly amount charged to each customer for their services	Row 1: 171.4497621
Bandwidth_GB_Year	Quantitative	A yearly average amount of data used, measured in GB	Row 1: 904.5361102

This final table documents customer's responses to survey questions regarding their customer experience during each individual interaction. The survey uses 1 to indicate that the applicable topic is "most important" and an 8 to indicate "least important".

Variable	**Data Type**	**Description**	**Example**
Item1	Qualitative	This column documents the timely responses from the company.	Row 1: 5
Item2	Qualitative	This column documents the customer's response to a survey question regarding the importance of timely fixes.	Row 1: 5
Item3	Qualitative	This column documents the customer's response to a survey question regarding the importance of timely replacements.	Row 1: 3
Item4	Qualitative	This column documents the customer's response to a survey	Row 1: 3

		question regarding the importance of reliability.	
Item5	Qualitative	This column documents the customer's response to a survey question regarding the importance of options.	Row 1: 4
Item6	Qualitative	This column documents the customer's response to a survey question regarding the importance of respectful response.	Row 1: 4
Item7	Qualitative	This column documents the customer's response to a survey question regarding the importance of courteous exchange.	Row 1: 3
Item8	Qualitative	This column documents the customer's response to a survey question regarding the importance of evidence of active listening.	Row 1: 4

Part II: Data Cleaning

C1: Plan To Detect Data Anomalies

In order to identify data anomalies within the provided dataset, I will import the data from the CSV file, as well as Python packages numpy, pandas, matplotlib.pyplot, and PCA from SKlearn. I will import the packages after importing the data, to ensure that I can utilize the packages with ease once I am ready to do so.

Starting with an initial view of the data types and column information by using the info() and describe() functions, I will look for any values that initially stand out. The describe() function will give me the summary statistics of the variables with data type int64. During this step I will look for any values that are drastically different from the mean value. The info() step will provide information regarding the dataframe dimensions, such as how many columns and entries are available, I will compare this with the data dictionary provided.

For values that were not assessed in the previous step using the describe() function, I will evaluate the remaining columns using value_counts() and describe() to seek any obvious issues that stand out. I will also assess all of the data types for each column to ensure that they are consistent, and if any necessary changes are needed for proper formatting.

Next, I will check for missing values by utilizing the (isnull().sum()) function. After identifying the null values, I will check for duplicates using the following functions:

```
duplicates = df.duplicated(keep=False)

duplicate_rows = df[duplicates]

print(df.duplicated().value_counts())
```

Once I have reviewed all of the outputs for the inputs described, I will be able to identify any data anomalies and will be able to move forward with the cleaning process.

C2: Justification of Methods Used

Executing the plan I've described above will render me a complete view of the data. This view will show the specific data types for all columns, descriptions of the data within each column, statistical values for applicable columns, evidence of null values and any duplicates. The dataframe dimensions will be available as well, and all column names and data can be compared to the data dictionary to confirm accuracy. Comparing the output values to the data dictionary will allow me to confirm the data will be used as intended, and the functions I will use will allow me to assess the quality of the data further for additional cleaning.

C3: Justification of Programming Language/Packages Used

As described above, I will be utilizing Python for this assessment. Python is flexible, simple and easy to interpret. This language allowed me to use packages and customized scripts to complete all the necessary steps within the process of cleaning the data. The packages I used were numpy, pandas, matplotlib.pyplot, and PCA from SKlearn.decomposition. Each package has its own functions that allow different processes throughout the cleaning stages. The Numpy package allows mathematical equations needed to transform the data. Pandas provides a logical structure, otherwise known as the dataframe, which was an absolute necessity to transform the data.

The other two packages used, matplotlib and PCA, allowed me to perform a Principal Component Analysis once the data was cleaned. The PCA from SKlearn package was used to perform the analysis. Matplotlib plotted the components for a visual representation of the

eigenvalues. All of these packages used within this programming language allowed me to complete the assessment in a clean, understandable way.

C4: Code Used To Identify Anomalies

The annotated scripts used to detect the anomalies can be assessed via the attached executable script file.

Part III: Data Cleaning

D1. Describe Anomalies Found

After implementing my plan as described in C, I found quite a few anomalies within the dataset. Here is a list of those anomalies:

- Improper use of capitalized letters in all columns other than items 1-8
- Improper name for variable CaseOrder
- Improper name for variable Outage_sec_perweek
- Improper name for variable InternetService
- Improper name for variable OnlineSecurity
- Improper name for variable OnlineBackup
- Improper name for variable DeviceProtection
- Improper name for variable TechSupport
- Improper name for variable StreamingTV
- Improper name for variable StreamingMovies
- Improper name for variable PaperlessBilling
- Improper name for variable PaymentMethod
- Improper name for variable MonthlyCharge
- Outage_sec_perweek has a negative value (ie: -1.348571), which is not possible. There is the potential to have service outages at a greater than or equal to 0, but not less than.
- Gender column has “Female” , “Male” and “prefer not to answer” values. The “prefer not to answer” should be “nonbinary” as described in the data dictionary
- The “Timezone” variable has city-specific data, and often numerous cities within the same state. A more centralized timezone by region would be more appropriate
- “Zip” has a minimum of 3 values (ie: 601), zip codes contain 5 numbers typically
- There are 32 cases in which the minimum value found within the “Population” is 0
- There are 2495 null values within the Children variable
- There are 2475 null values within the Age variable
- There are 2490 null values within the Income variable
- There are 2477 null values within the Techie variable

- There are 1026 null values within the Phone variable
- There are 991 null values within the TechSupport variable
- There are 931 null values within the Tenure variable
- There are 1021 null values within the Bandwidth_GB_Year variable
- Education column has object data type
- Churn column has object data type
- Techie column has object data type
- Contract column has object data type
- Port_modem column has object data type
- Tablet column column has object data type
- Phone column column has object data type
- Multiple column column has object data type
- OnlineSecurity column has object data type
- InternetService column has object data type
- OnlineBackup column has object data type
- DeviceProtection column has object data type
- TechSupport column has object data type
- StreamingTV column has object data type
- StreamingMovies column has object data type
- PaperlessBilling column has object data type
- item1 column has int64 data type
- item2 column has int64 data type
- item3 column has int64 data type
- item4 column has int64 data type
- item5 column has int64 data type
- item6 column has int64 data type
- item7 column has int64 data type
- item8 column has int64 data type

D2. Justification of Methods Used to Mitigate the Anomalies

Throughout this step I utilized many different methods, but also a lot of similar scripts. As I had mentioned before, Python and its packages are simple and convenient which makes replicating feasible. Methods used to mitigate the data quality issues described above, and why they were used are explained below.

- Changing all column names to be entirely lowercase
    - This was necessary for proper usage of the data, Python is case sensitive so it was important that all variables be lowercase

- Renaming incorrect column names was the next step. There was proper spacing for some columns, but not the majority
    - All columns with multiple words should have spaces between words, or in python a "_"
    - Having appropriate spacing is necessary for clean and correct transformation and utilization of the data in further data analysis processes

See the applicable columns below:

- caseorder
- outage_sec_perweek
- onlinesecurity
- internetservice
- onlinebackup
- deviceprotection
- techsupport
- streamingtv
- streamingmovies
- paperlessbilling
- paymentmethod
- monthlycharge

- Many columns were originally of a data type that did not suit their values well for transformation and analysis of the data within.
    - With the information obtained from the D206 course webinar #3, I learned that "[Ordinal encoding] transforms categorical value to numerical value based on rank or order."
    - Given their values (ie: yes/no, college/no college etc.) the data within the following columns are better utilized by implementing ordinal encoding:

- education
- churn
- techie
- contract
- port_modem
- tablet
- phone
- multiple
- onlinesecurity
- internetservice
- onlinebackup

- deviceprotection
- techsupport
- streamingtv
- streamingmovies
- paperlessbilling

- Drop negative value in outage_sec_per_week column
    - The lowest applicable value for this column in reality would be 0, not negative
    - Because I will not be using this variable for my research question, it would be appropriate to drop

- Drop 0 values in the “population” column
    - Each customer has their own address, and as such could not live within a city that has a zero population size
    - This column would not be necessary for the research question, and would better be dropped rather than filled

- Drop missing or null values from columns “techie”, “phone” , and “tech_support”
    - Because I will not be using these variables for my research question, it would be appropriate to drop
    - Especially considering these variables are “Yes/No” rather than a number in which I could have potentially replaced the values with a numerical value such as the mean, median, or mode

- Replacing null values within the tenure and bandwidth columns with -1 to reflect less than one year
    - With data type float64, pandas will automatically replace decimal values of less than 1 as null, which is inaccurate

- Fill in missing values for age, children, and income with the mean value for each respective column
    - The mean value is the most common value, which would be a more appropriate change than keeping the null values or dropping the null values altogether
    - By not dropping these null values, we can still use the variables for analysis

- Replace location based timezone values with time-zone specific values
    - Using location based timezone values rendered dirty data
    - Time-zone specific data is cleaner and easier to analyze

- Fix minimum zip codes by converting column to string from int64, then front-fill with zeros to fill
    - Puerto Rico's zip codes start with zeros, which were automatically removed in this data frame
    - Converting the column to string from int64 allows appropriate usage of all numbers (including zeros)
    - By front-filling the zip codes with zeros, we can achieve an actual zip code with 5 characters

- Convert columns "payment_method", "area", "timezone", and "marital" to category from object
    - These columns are more appropriately categorized rather than to be input as data type object

- Convert item1 - item8 column to category from int64
    - These columns are more appropriately categorized rather than to be input as data type int64

- Change "prefer not to answer" values within the gender column to "nonbinary"
    - The "prefer not to answer" value was updated to reflect "nonbinary" as expected given the data dictionary

D3. Outcome from implementing the data-cleaning steps

Writing code to transform the column names rendered the following variable name updates:

- Every column name now has all lowercase letters
- caseorder is now case_order
- outage_sec_perweek is now outage_sec_per_week
- onlinesecurity is now online_security
- internetservice is now internet_service
- onlinebackup is now online_backup
- deviceprotection is now device_protection
- techsupport is now tech_support
- streamingtv is now streaming_tv
- streamingmovies is now streaming_movies
- paperlessbilling is now paperless_billing
- paymentmethod is now payment_method
- monthlycharge is now monthly_charge

In the image below, these changes can be seen.

```
[46]: print(df.columns)

Index(['case_order', 'customer_id', 'interaction', 'city', 'state', 'county',
       'zip', 'lat', 'lng', 'population', 'area', 'timezone', 'job',
       'children', 'age', 'education', 'employment', 'income', 'marital',
       'gender', 'churn', 'outage_sec_per_week', 'email', 'contacts',
       'yearly_equip_failure', 'techie', 'contract', 'port_modem', 'tablet',
       'internet_service', 'phone', 'multiple', 'online_security',
       'online_backup', 'device_protection', 'tech_support', 'streaming_tv',
       'streaming_movies', 'paperless_billing', 'payment_method', 'tenure',
       'monthly_charge', 'bandwidth_gb_year', 'item1', 'item2', 'item3',
       'item4', 'item5', 'item6', 'item7', 'item8'],
      dtype='object')
```

To ensure ease of transformation and analysis of the data within particular columns, they needed to be categorized differently. Enforcing ordinal encoding and assigning a unique integer to each category for the following columns changed the variables to be categorized by a numerical order:

- education
- churn
- techie
- contract
- port_modem
- tablet
- phone
- multiple
- onlinesecurity
- internetservice
- onlinebackup
- deviceprotection
- techsupport
- streamingtv
- streamingmovies
- paperlessbilling

Dropping the negative value in the outage_sec_per_week column removed the incorrect data point that was negative. Now the minimum value is 0.113821.

```
[14]: #Drop negative value in outage_sec_perweek column
      df = df[df['outage_sec_per_week'] >= 0]
```

```
[15]: # Get summary statistics for outage_sec_per_week to confirm that negative min was dropped
      print(df.outage_sec_per_week.describe())
```

```
count    9989.000000
mean       11.466205
std         7.018413
min         0.113821
25%         8.064574
50%        10.208720
75%        12.491290
max        47.049280
Name: outage_sec_per_week, dtype: float64
```

Dropping the 0 value entries in the population column removed the incorrect data. Now the minimum value is 4.

Implementing scripts to remove null values cleaned out the missing data for the following columns:

- techie
- phone
- tech_support

Scripts used to fill in missing values with the mean value for each column corrected the null or missing values for the following columns:

- age
- income
- children

Using -1 in place of null values within the tenure and bandwidth columns was necessary to accurately reflect customer's that have been with the business for less than one year. This also mitigated the remaining null values within the dataset. This can be seen below by using the isnull().sum() again.

```
[89]: # Check for missing values
      print(df.isnull().sum())
```

```
case_order              0
customer_id             0
interaction             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
education               0
employment              0
income                  0
marital                 0
gender                  0
churn                   0
outage_sec_per_week     0
email                   0
contacts                0
yearly_equip_failure    0
techie                  0
contract                0
port_modem              0
tablet                  0
internet_service        0
phone                   0
multiple                0
online_security         0
online_backup           0
device_protection       0
tech_support            0
streaming_tv            0
streaming_movies        0
paperless_billing       0
payment_method          0
tenure                  0
monthly_charge          0
bandwidth_gb_year       0
item1                   0
item2                   0
item3                   0
item4                   0
item5                   0
item6                   0
item7                   0
item8                   0
dtype: int64
```

Replacing location/city based timezone values with time-zone specific values rendered proper categorization of the data within the “timezone” column to be placed in one of the following timezones:

- PDT - Pacific Daylight Time
- AT - Alaskan Time
- MST - Mountain Standard Time
- CDT - Central Daylight Time
- EDT - Eastern Daylight Time
- Atlantic Standard Time
- HST - Hawaii Aleutian Time

Converting the “zip” column from int64 to string and front-filling the variables with zeros until five characters are met allowed the minimum “zip” variable of 601 to be transformed to 00601, which is a Puerto Rican zip code. (Magaly Rivera, 2023) In addition to this particular minimum value being corrected, the script used also fixed remaining zip code values that were less than 5 characters in length.

```
[86]: # Fix minimum zip codes by converting column to string from int64, then front-fill with zeros to fil
      df['zip'] = df['zip'].astype("str").str.zfill(5)
```

```
[88]: # Get summary statistics of the dataset with datatype int64
      print(df.zip.min())
```

```
00662
```

For the columns with original data type object that would better be suited as a category, transforming the data to reflect this update simplified the values for the following columns:

- payment_method
- area
- timezone
- marital

For the columns pertaining to customer responses, “item1, item2…” their original data type was int64. Although the responses are numerical (ie: 1. Most important), int64 is not the correct data type for these variables. Changing the data type from int64 to category removed these columns from any statistical formulas, which was unnecessary.

Changing the "prefer not to answer" variable in the gender column to "nonbinary” allowed for the data to be represented correctly, as described in the data dictionary.

D4. Provide the annotated code used to mitigate the data quality issues

The annotated scripts used to mitigate the anomalies can be assessed via the attached executable script file.

D5. Provide a copy of the cleaned data set as a CSV file.

The code use to save the cleaned data as a CSV file within Python:

```
# Save dataframe to CSV, ignore the index to avoid additional unnecessary column
df.to_csv('clean_data.csv', index=False)
```

This CSV file was then downloaded to my desktop, and was uploaded in addition to all files for this assessment.

D6: Summarize Limitations of the Data Cleaning Process

Limitations I found while cleaning the data are described here:

- Finding the appropriate values to fill the null values found in the age, income, children, tenure, and bandwidth columns was difficult

- Dropping the 0 values within the population column, the negative values within the outages column, and all null values within the population, techie, phone and tech_support columns resulted in a loss of 3,974 data points
- Updating the gender column data from “prefer not to answer” to “nonbinary” to reflect the data dictionary was a bit problematic

D7: Discussion of Limitations

To rectify the null values found in the age, income and children columns the values needed to be filled rather than dropped. All null values within these columns were replaced with the mean for each respective column. It was initially difficult to decide what these values would be filled with. Filling the null values with the average for each column is most appropriate considering that we can not determine these missing values with the other data in the dataset. However, because the accuracy of these specific values may be a necessity to the research question, replacing them with anything other than the absolute accurate answer could potentially skew the results in later data analysis steps.

Correcting the 0, or null, values within the tenure and bandwidth columns was another challenge. Because their data type is initially float64, pandas representation of values less than 1, but greater than 0 would be considered null. Viewing all of the float characters and verifying that values were not null but rather a decimal point < 1 helped in my decision process for this step. I concluded that replacing values that pandas considers null with a -1, would be most appropriate. This allows customers with less than one year as a customer, and/or less than 1 gb of bandwidth to be accounted for in the analysis. Although the applicable variables were not dropped and are still usable, the value of -1 may still cause skewed results which could derail the analysis.

For the remaining null values within the techie, phone and tech_support columns, the 0 value in the population column and the negative value in the outages column, dropping these values was most appropriate. These variables are not going to be used when analyzing the research question. However, dropping these values resulted in a dataset of 6,018 entries rather than 10,000. With less data available, our analysis is inevitably not an entire view of all 10,000 values. As such, the analysis may not be as usable or beneficial to the business.

Updating the gender column data from “prefer not to answer” to “nonbinary” per the data dictionary could be problematic because no one can be sure that each individual who selected “prefer not to answer" identifies as “nonbinary.” As someone that identifies as nonbinary myself, I found it extremely difficult to update this data. There are many different reasons a person may select “prefer not to answer” and that may not always be because they identify as “nonbinary.” Fortunately, customers would not see this data. If they ever did however, it is concerning to think of how some people may react to being updated to nonbinary, given this country’s current status towards the LGBTQ+ community. It would be more appropriate to get updated demographics from customers rather than change this, even for analysis.

E1. Identify principal components and provide the output of the PCA

Because a Principal Component Analysis (PCA) requires quantifiable numeric data, the variables used for this dataset are latitude, longitude, population, children, age, income, outage in seconds per week, email, yearly equipment failure, tenure, monthly charge and bandwidth in gb per year. These variables were identified as quantitative earlier in the cleaning stages when reviewing the data dictionary. There is one more column which is quantitative, which is the "case order." Case order would not be applicable when completing a PCA, because this column is a unique value given to maintain order in the original dataset. By following the steps in the D206 course webinar #4 all other quantitative data was implemented and the following output shows the PCA loadings:

```
[101]: # Print out the component loadings to find the correlation coefficients of each Principal Component
       pca_loadings
```

[101]:

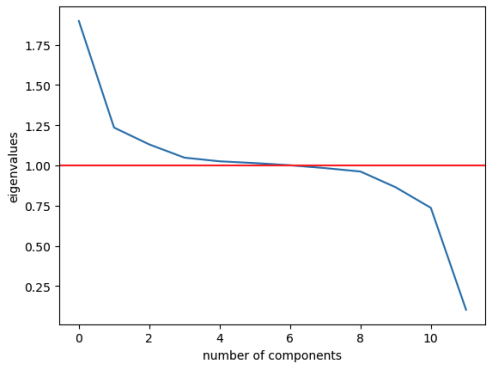
	PC1	PC2	PC3	PC4	PC5	PC6	PC7	PC8	PC9	PC10	PC11	PC12
lat	-0.024273	-0.710619	-0.084138	0.118615	-0.028258	-0.084494	-0.022869	0.079716	-0.016443	-0.014970	0.676736	0.000565
lng	0.008132	0.171682	0.067357	-0.783448	-0.278250	0.207583	0.064480	-0.211829	0.195836	0.066471	0.373922	0.001081
population	0.005489	0.648009	0.068912	0.331100	0.213297	-0.053515	0.050566	0.004115	-0.125888	0.000345	0.631583	-0.000490
children	0.000235	-0.021609	-0.006613	-0.484327	0.444534	-0.211548	-0.111556	0.358485	-0.617236	-0.011685	-0.007701	-0.020519
age	-0.012728	0.021700	-0.037302	0.135395	-0.417794	0.659304	0.126607	0.391657	-0.431936	0.117782	0.009068	0.021483
income	0.005591	-0.060681	-0.023541	-0.066103	0.387049	0.147562	0.794132	0.295478	0.311947	-0.063468	-0.022069	0.001215
outage_sec_per_week	0.021303	-0.108844	0.695378	0.048995	0.102361	-0.016042	0.002428	0.021517	0.036197	0.699181	-0.019498	0.000288
email	-0.020552	0.149566	0.099776	-0.044328	-0.392580	-0.417134	-0.118977	0.715470	0.326401	-0.062087	0.026272	0.005615
yearly_equip_failure	0.015581	-0.011743	0.045107	-0.019905	0.427524	0.517695	-0.560893	0.247324	0.392935	-0.121410	0.038589	-0.002355
tenure	0.704688	-0.005196	-0.058134	0.012758	-0.026907	-0.001413	0.001561	0.020509	0.006412	0.038104	0.008280	-0.705077
monthly_charge	0.045620	-0.080480	0.694704	0.017548	-0.078808	0.055277	0.069767	-0.074264	-0.135227	-0.684294	-0.003243	-0.048131
bandwidth_gb_year	0.706637	-0.012273	-0.010616	-0.002542	-0.002172	-0.019320	-0.003113	0.008791	-0.009810	-0.012682	0.006830	0.706842

E2. Justify the reduced number of the principal components and include a screenshot of a scree plot.

Following the loadings of the PCA matrix I was able to identify the amount of principal components of the new dataset by using a scree plot, which included the eigenvalues, as a visualization tool. Again following the guidelines from the D206 course webinar #4, I reviewed the PCA data with the Kaiser Rule in mind. As such, retaining the principal components with an absolute value of greater than 1 is necessary for this step. There are 6 principal components following the Kaiser Rule. Justifying this reduced number is easy and best summed up by the textbook explanation, "You will never be selecting redundant information by using any given subset of principal components in your analysis. You will always lose some information when reducing a data set; however, you will gain precision." (Larose & Larose, 2019)

```
[110]: covariance_matrix = np.dot(df_pca_normalized.T, df_pca_normalized) / df_pca.shape[0]
       eigenvalues = [np.dot(eigenvector.T, np.dot(covariance_matrix, eigenvector)) for eigenvector in pca.components_]

[111]: plt.plot(eigenvalues)
       plt.xlabel('number of components')
       plt.ylabel('eigenvalues')
       plt.axhline(y=1, color= "red")
       plt.show()
```



E3. Describe how the organization would benefit from the use of PCA.

The data within this dataset initially was quite "dirty." Even after the data was cleaned, there were variables that were unnecessary or redundant. By using PCA, the business will spend less time acquiring and utilizing unnecessary redundant data. Instead, the business can stick to the important, or primary, components and not waste time with overlapping data. This will also be beneficial to the data analyst who later implements machine learning with the data.

Part IV. Supporting Documents

F. Provide a Panopto video recording

The Panopto video recorded for this assessment can be found in the corresponding folder for this course.

G. Third-Party Code References

Middleton, K. (2022) Webinar 3: Getting Started with Re-expression of Categorical Variables. Western Governors University. <https://wgu.hosted.panopto.com/Panopto/Pages/Viewer.aspx?id=bfc98490-7757-4dbb-8794-af56013265ab>

Middleton, K. (2022) Webinar 4: Getting Started with PCA. Western Governors University.<https://wgu.hosted.panopto.com/Panopto/Pages/Viewer.aspx?id=7b31791b-24e8-4077-ba1a-af5d0005144c>

H. References

Larose, C. D., & Larose, D. T. (2019). Data science using Python and R. ISBN-13: 978-1-119-52684-1.

Rivera, M. (2023). Puerto Rico ZIP/Postal Codes. Welcome to Puerto Rico. <https://welcome.topuertorico.org/reference/zipcodes.shtml>