Jeffery Bills

A:Research Question

The Research question I will be investigating is “Are there variables that contribute to readmission more than others?” There is relevance to this question because Hospitals can focus on addressing the specific variable in order to reduce hospital readmissions, freeing up resources for new injuries and illnesses.B.Description of Variables

In the medical services dataset, there were 64 columns each representing variables related to a patient and their hospital admission.

1. CaseOrder (quantitative), row 1 example: 1  
   This column enforces the ordering of all rows in the dataset by acting as an index included in the data.
2. Customer\_id (qualitative), row 1 example: C412403  
   Each patient is uniquely identified by the ID in this column.
3. Interaction (qualitative), row 1 example: 8cd49b13-f45a-4b47-a2bd-173ffa932c2f

Unique IDs associated with transactions, procedures, and admissions involving patients

1. UID (qualitative), row 1 example: 3a83ddb66e2ae73798bdf1d705dc0932  
   The IDs in these two columns serve to uniquely identify data pertaining to transactions, procedures, and admissions for the patient record. Though their design suggests they are not meant to be human-readable, it is unknown how these manage to do so.
2. City (qualitative), row 1 example: Eva. The billing statement's indicated patient's city of residence
3. State (qualitative), row 1 example: AL.State of residency of the patient as shown on the billing statement
4. County (qualitative), row 1 example: Morgan. The county of domicile of the patient as indicated on the bill
5. Zip (qualitative), row 1 example: 35621. The patient's residential zip code as shown on the billing statement
6. Lat (quantitative), row 1 example: 34.3496GPS. Coordinates for the patient's home as shown on the invoice
7. Lng (quantitative), row 1 example: -86.7251  
   GPS coordinates for the patient's home as shown on the invoice
8. Population (quantitative), row 1 example: 295.Population residing one mile around the patient, as determined by census data
9. Area (qualitative), row 1 example: Suburban  
   This gives the population within a mile of the patient's residence based on census data, along with information about whether the area is rural, suburban, or urban.
10. Timezone (qualitative), row 1 example: America/Chicago  
    Informs us of the patient's timezone.
11. Job (qualitative), row 1 example: Psychologist, sport and exercise  
    This gives the population within a mile of the patient's residence based on census data, along with information about whether the area is rural, suburban, or urban.
12. Children (quantitative), row 1 example: 1  
    this represents the number of kids living in the patient's home and is derived from admissions data. This appears to be deceptive since, in the absence of the data dictionary, one would typically interpret "children" to mean the number of children the patient possesses as opposed to the number of children they live with.
13. Age (quantitative), row 1 example: 53  
    This is the patient's age, taken from the admissions data.
14. Education (qualitative), row 1 example: Some College, Less than 1 Year  
    This represents the patient's greatest degree of education, as taken from the admissions records.
15. Employment (qualitative), row 1 example: Full Time  
    Taken from admissions information, this is the patient's current employment status.
16. Income (quantitative), row 1 example: 86575.93  
    The data dictionary indicates that this is the patient's or primary insurance holder's annual income, derived from admissions data. Once more, it's uncertain whether it's possible to determine which applies to a particular patient. Once more, it appears that this would be deceptive because the term "income" associated with a patient would suggest that it comes from their own source rather than from their parent, spouse, or other caregiver. There are also some very serious problems about whether or not this (and some related) data doesn't encourage explicitly capitalistic activities at the expense of patient care or other ethical considerations, and why a hospital should even care about it.
17. Marital (qualitative), row 1 example: Divorced
18. The data dictionary indicates that this is the patient's or primary insurance holder's marital status, derived from admissions information. Once more, it's uncertain whether it's possible to determine which applies to a particular patient. Once more, it appears that this would be deceptive because the term "marital status" when applied to a patient would suggest that it refers to the patient's marital status rather than the marital status of their parent, spouse, or other caregiver.
19. Gender (qualitative), row 1 example: Male
20. The patient self-identifies as this gender, which is presumably derived from admissions data. This is limited to male, female, or non-binary, according to the data dictionary.
21. ReAdmis (qualitative), row 1 example: No
22. This column, which is a binary variable, shows if the patient was readmitted within a month of being released.
23. VitD\_levels (quantitative), row 1 example: 17.802330  
    This column shows the patient's vitamin D level in nanograms per milliliter.
24. Doc\_visits (quantitative), row 1 example: 6
25. The number of visits the patient received from the primary care physician is shown in this column.
26. Full\_meals\_eaten (quantitative), row 1 example: 0  
    This column records how many complete meals the patient ate while they were in the hospital.
27. VitD\_supp (quantitative), row 1 example: 0  
    The number of vitamin D supplements given to the patient is shown in this column.
28. Soft\_drink (qualitative), row 1 example: NA  
    This column, which is a binary variable, shows whether or not the patient regularly consumes three or more sodas each day.
29. Initial\_admin (qualitative), row 1 example: Emergency Admission
30. HighBlood (qualitative), row 1 example: Yes
31. This column, which is a binary variable, shows whether the patient has high blood pressure or not.
32. Stroke (qualitative), row 1 example: No  
    This column, which is a binary variable, shows whether the patient has experienced a stroke or not.
33. Complication\_risk (qualitative), row 1 example: Medium
34. Overweight (qualitative), row 1 example: 0
35. This column, which is a binary variable, shows whether or not the patient is overweight for their specific age, gender, and height.
36. Arthritis (qualitative), row 1 example: Yes
37. This column, which is a binary variable, shows whether the patient has arthritis or not.
38. Diabetes (qualitative), row 1 example: Yes  
    This column, which is a binary variable, shows whether the patient has diabetes or not.
39. Hyperlipidemia (qualitative), row 1 example: No
40. This column, which is a binary variable, shows whether the patient has Hyperlipidemia or not.
41. BackPain (qualitative), row 1 example: Yes  
    This column, which is a binary variable, shows whether the patient has chronic back pain or not.
42. Anxiety (qualitative), row 1 example: 1  
    This column, which is a binary variable, shows whether the patient has anxiety or not.
43. Allergic\_rhinitis (qualitative), row 1 example: Yes  
    This column, which is a binary variable, shows whether the patient has Allergic\_rhinitis or not.
44. Reflux\_esophagitis (qualitative), row 1 example: No  
    A binary variable, this column indicates whether or not the patient has reflux esophagitis.
45. Asthma (qualitative), row 1 example: Yes  
    A binary variable, this column indicates whether or not the patient has asthma.
46. Services (qualitative), row 1 example: Blood Work  
    This column contains a list of the services provided to the patient while they were in the hospital. The data dictionary restricts this to blood testing, intravenous, CT, and MRI scans.
47. Initial\_days (quantitative), row 1 example: 10.585770
48. This column contains the total number of days the patient was admitted to the hospital during their first visit. This appears to be determined using hours and minutes, most likely by timestamps for admission and release, based on the accuracy of the floating point number.
49. TotalCharge (quantitative), row 1 example: 3191.048774  
    TThis column shows the daily cost of care in USD that the patient is charged. According to the data dictionary, this is an average that was determined by dividing the total charge by the number of hospital days (but only "typical" charges—a hospital might not give specific services). The fact that this column is a calculated daily average rather than a total and that it contains exclusions based on the type of care/service provided based on what is (or is not) "typical"—despite the fact that a hospital purposefully provides a broad range of medical services—make it appear blatantly misleading. The column does not reflect a "total charge" by any intuitive meaning of the term.
50. Additional\_charges (quantitative), row 1 example:17939.403420
51. This column lists the "average amount... for miscellaneous procedures, treatments, medicines, anesthesiology, etc." that was charged to the patient in USD. Although the data dictionary does not specify the methodology, one could guess that it follows the same steps as TotalCharge and divides by the number of hospital days to arrive at this average. Once more, this column appears to be blatantly misleading due to its failure to reflect a total (the intuitive interpretation of "additional charges") and its arbitrary classification of several common medical care options as "non-typical" charges, which are shown separately from the "typical" charges in the TotalCharge column.
52. Item1 (qualitative), row 1 example: 3
53. The patient's response to a survey question about the significance of prompt admission is recorded in this column. According to the survey, a 1 means this is "most important," and an 8 means this is "least important".
54. Item2 (qualitative), row 1 example: 3
55. The patient's response to a survey question about the value of prompt treatment is recorded in this column. According to the survey, a 1 means this is "most important," and an 8 means this is "least important".
56. Item3 (qualitative), row 1 example: 2  
    The patient's response to a survey question about the value of on-time appointments is recorded in this column. According to the survey, a 1 means this is "most important," and an 8 means this is "least important".
57. Item4 (qualitative), row 1 example: 2  
    The patient's response to a survey question about the value of reliability is recorded in this column. According to the survey, a 1 means this is "most important," and an 8 means this is "least important".
58. Item5 (qualitative), row 1 example: 4  
    The patient's response to a survey question about the value of options is recorded in this column. According to the survey, a 1 means this is "most important," and an 8 means this is "least important".
59. Item6 (qualitative), row 1 example: 3  
    The patient's response to a survey question about the significance of treatment hours is recorded in this column. According to the survey, a 1 means this is "most important," and an 8 means this is "least important".
60. Item7 (qualitative), row 1 example: 3  
    The patient's reaction to a survey question about how important polite personnel is is documented in this column. According to the survey, a 1 means this is "most important," and an 8 means this is "least important".
61. Item8 (qualitative), row 1 example: 4  
    This column presents the patient's reaction to a survey question about how important it is to see proof that the doctor is actively listening to them. According to the survey, a 1 means this is "most important," and an 8 means this is "least important".

C:Plan to Identify Data Anomalies

To Identify data anomalies in the dataset, ill use the data in the csv file, the data dictionary, and visual exploration to examine problems present. After that, I will use.info() to cross reference the column names with the data dictionary. From there, I will look through the table to see if I can find any Null values to determine how they affect the dataset and what actions I should do with them. To detect duplicates, I will use the duplicate() function unique columns to make sure there is no redundant information. on each variable. I can review each column of the table individually after taking a look at it as a whole. I can then check for data that is obviously inaccurate or should not be there, data that needs to be standardized, or data that is formatted correctly for the purpose for which it is intended. Depending on the column, either the value\_counts() or description() functions will be used to do this. Altogether, this will examine missing values, outliers, and if we have to do re-expression of categorical variables and inspect any other data quality issues.

C2: Justification of Plan

This plan lets me look at the data as a whole by looking at the raw csv file, a guidebook of sorts with the data dictionary to supplement information, and the visual exploration to examine the variable entries to get a summary of the data. Additionally, by using the describe() function, I will be able to summarize quantitative data by its appearance and learn about the data’s mean,standard deviation, and min/max.

C3: Justify Programming Language/Packages

I will be using Python for this assignment as I don’t need the full statistical scope of r and Python has easy to access scripts to clean data. I will be using numpy,pandas,SKlearn,and seaborn as packages to assist me in my analysis. Numpy will let me use mathematical functions when transforming data, Pandas will be used so I can create a dataframe for the CSV file when importing, SKlearn is required for Principal Component Analysis, and Seaborn will be used for graphing the PCA.

C4: Code used to identify Anomalies

See Code Attached D206CCode

**#importing pandas**

**import pandas as pd**

**from pandas.api.types import CategoricalDtype**

**#impoting numpy sklearn seaborn matoplot scipy**

**import numpy as np**

**import matplotlib.pyplot as plt**

**from sklearn.decomposition import PCA**

**import seaborn as sns**

**from scipy import stats**

**#Read DataSet off of computer**

**df = pd.read\_csv('C:\\Users\jefft\\Desktop\\medical\_raw\_data.csv', index\_col=0)**

**#basic information about dataset**

**df.info()**

**#Visually Looking at data frame**

**pd.set\_option("display.max\_columns", None)**

**df**

**#Detection of Missing Data**

**df.isnull().sum()**

**# Check that the placeholder variable has 10,000 outcomes and is unique.**

**df.CaseOrder.value\_counts().count()**

**# Check if Customer\_id is distinct (there ought to be 10,000 results).**

**df.Customer\_id.value\_counts().count()**

**# Check that UID is distinct (there should be 10,000 results).**

**df.UID.value\_counts().count()**

**# Verify the entire value range and decide if the datatype is appropriate for category**

**df.Timezone.value\_counts()**

**# Check that the data exists and that it is appropriate for the datatype Int64.**

**df.Children.describe()**

**# Check that the data exists and that it is appropriate for the datatype Int64.**

**df.Age.describe()**

**# Verify if the values are appropriate for the datatype categorical.**

**df.Education.value\_counts()**

**# Verify if the values are appropriate for the datatype categorical.**

**df.Employment.value\_counts()**

**# Confirm the existence of data within reasonable limitations**

**df.Income.describe()**

**# Verify if the values are appropriate for the datatype categorical.**

**df.Marital.value\_counts()**

**# Verify if the values are appropriate for the datatype categorical.**

**df.Gender.value\_counts()**

**# Verify that the values are appropriate for the datatype boolean.**

**df.ReAdmis.value\_counts()**

**# Verify data exists within reasonable bounds**

**df.VitD\_levels.describe()**

**df.VitD\_levels.nlargest(n=20)**

**# Verify data exists within reasonable bounds**

**df.Doc\_visits.describe()**

**# Verify data exists within reasonable bounds**

**df.Full\_meals\_eaten.describe()**

**# Verify data exists within reasonable bounds**

**df.VitD\_supp.describe()**

**# Verify that the values are appropriate for the datatype boolean.**

**df.Soft\_drink.value\_counts()**

**# Verify if the values are appropriate for the datatype categorical.**

**df.Initial\_admin.value\_counts()**

**# Verify that the values are appropriate for the datatype boolean.**

**df.HighBlood.value\_counts()**

**# Verify that the values are appropriate for the datatype boolean.**

**df.Stroke.value\_counts()**

**# Verify if the values are appropriate for the datatype categorical.**

**df.Complication\_risk.value\_counts()**

**# Verify data exists to be sensibly translated into datatype boolean**

**df.Overweight.value\_counts()**

**# Verify that the values are appropriate for the datatype boolean.**

**df.Arthritis.value\_counts()**

**# Verify that the values are appropriate for the datatype boolean.**

**df.Diabetes.value\_counts()**

**# Verify that the values are appropriate for the datatype boolean.**

**df.Hyperlipidemia.value\_counts()**

**# Verify that the values are appropriate for the datatype boolean.**

**df.BackPain.value\_counts()**

**# Verify data exists to be sensibly translated into datatype boolean**

**df.Anxiety.value\_counts()**

**# Verify that the values are appropriate for the datatype boolean.**

**df.Allergic\_rhinitis.value\_counts()**

**# Verify that the values are appropriate for the datatype boolean.**

**df.Asthma.value\_counts()**

**# Verify if the values are appropriate for the datatype categorical.**

**df.Services.value\_counts()**

**# Examine data format**

**df.Initial\_days.value\_counts()**

**# Verify data exists within reasonable bounds, is reasonable to be rounded**

**df.TotalCharge.describe()**

**# Verify data exists within reasonable bounds, is reasonable to be rounded**

**df.Additional\_charges.describe()**

**# Confirm that the data fits inside the 1–8 limit and is appropriate for storage as an ordered categorical datatype.**

**df.Item1.value\_counts()**

**# Confirm that the data fits inside the 1–8 limit and is appropriate for storage as an ordered categorical datatype.**

**df.Item2.value\_counts()**

**# Confirm that the data fits inside the 1–8 limit and is appropriate for storage as an ordered categorical datatype.**

**df.Item3.value\_counts()**

**# Confirm that the data fits inside the 1–8 limit and is appropriate for storage as an ordered categorical datatype.**

**df.Item4.value\_counts()**

**# Confirm that the data fits inside the 1–8 limit and is appropriate for storage as an ordered categorical datatype.**

**df.Item5.value\_counts()**

**# Confirm that the data fits inside the 1–8 limit and is appropriate for storage as an ordered categorical datatype.**

**df.Item6.value\_counts()**

**# Confirm that the data fits inside the 1–8 limit and is appropriate for storage as an ordered categorical datatype.**

**df.Item7.value\_counts()**

**# Confirm that the data fits inside the 1–8 limit and is appropriate for storage as an ordered categorical datatype.**

**df.Item8.value\_counts()**

D1:Describe Anomalous findings.

While area is saved as a string object, it would be more appropriate & economical to store it as a category. Zip codes are stored as integers rather than strings and have lost their leading zeros as a result.

Children should be integers because they are kept as a floating point number.

Age should be an integer since it is stored as a floating point value.

Education is currently saved as a string object, but categorizing it would be more appropriate and effective.

Although married is recorded as a string object, storing it as a category would be more appropriate and effective.

The gender object is saved as a string, however a category would be a more appropriate & efficient way to store it.

According to the data dictionary, gender should contain "Male", "Female", and "Non-Binary", however it now contains "Male", "Female", and "Prefer not to answer". While there is opportunity for further consideration to make this more precise and instructive, the data dictionary is the authoritative source that the data must follow in this regard.

ReAdmis is kept as a string object, although a boolean would be maintained more effectively. VitD levels are kept to six decimal places, which is an excessive amount of accuracy; they will round down to three decimal places.

Although soft drink is kept as a text object, a boolean object would be a more appropriate and efficient storage format.

Although initial\_admin is saved as a string object, storing it as a category would be more appropriate & efficient.

HighBlood is kept as a string object, but a boolean would be kept more effectively & suitably.

Although stroke is kept as a string object, a boolean object would be a more appropriate and efficient storage format.

Though it would be more appropriate & economical to record Complication\_risk as a category, it is now saved as a text object.

Floating point numbers are used to store overweight, yet boolean data would be more appropriate & economical.

Although arthritis is currently recorded as a string object, a boolean object would be a more appropriate & efficient storage format.

Although hyperlipidemia is now kept as a text object, a boolean object would be a more appropriate and efficient storage format.

Although BackPain is kept as a string object, a boolean object would be a more appropriate and efficient storage format.

Although anxiety is currently kept as a floating point number, boolean storage would be more appropriate & economical.

Though a boolean would be more appropriate & efficient to store, allergic\_rhinitis is now stored as a string object.

Though a boolean would be more appropriate & efficient to store, reflux\_esophagitis is now stored as a text object.

Although asthma is recorded as a string object, a boolean object would be a more appropriate and efficient storage format.

Although services are kept as string objects, categories would be more appropriate & efficient to store.

While Items 1 through 9 are now kept as integers, it would be more appropriate and efficient to store them as ordered categories

D2:Plans for Mitigating Anomalies

To add zeros to the zip codes missing integers, I converted them into a string and did a front fill to add 0 to the entries that did not have 5 integers. To fix the gender wording, I decided to find all instances in the column where a patient answered “Prefer not to say” and instead I replaced the answer with “nonbinary” in order to align with the data dictionary. All NaN values in the Initial\_days column will be changed to 0s. Given that the column contains no values lower than 1 (or even 1.00), it is plausible to assume that anyone with a stay of less than 1 was simply registered as null.For the columns Childr Soft Drinkome,Soft\_drink, overweight, and anxiety that contained null values, I opted to just remove the rows that were null altogether as it doesn’t make sense to guesstimate information regarding a patient as that could lead to wildly skewed results.

D3:Summarize Outcomes of Cleaning Operations

In addition to providing limitations that help maintain the integrity of certain data points when altered, the aforementioned actions will make it easier to analyze this data in the future. By changing the zipcodes, it creates a more uniform data set in a standard entry. Eliminating null values allows easier analysis into the correlation of the variables, and changing answers to align with the data dictionary will help create a stable, uniform information supplement that is accurate.

## D4: Code used to Mitigate Anomalies

# Convert column to string from int, then front-fill string with 0's to reach 5 chars

df['Zip'] = df['Zip'].astype("str").str.zfill(5)

# Convert column to category from string

df["Area"] = df["Area"].astype("category")

# Convert column to category from string

df["Education"] = df["Education"].astype("category")

# Convert column to category from string

df["Employment"] = df["Employment"].astype("category")

# Convert column to category from string

df["Marital"] = df["Marital"].astype("category")

# Convert column to category from string

df["Timezone"] = df["Timezone"].astype("category")

# Convert column from float to int, use type Int64 to handle NaN values (Int64 supports NaN, int64 does not)

df["Children"] = df["Children"].astype("Int64")

# Convert column from float to int, use type Int64 to handle NaN values (Int64 supports NaN, int64 does not)

df["Age"] = df["Age"].astype("Int64")

df.Gender.replace({

"Female" : "Female",

"Male" : "Male",

"Prefer not to answer" : "NonBinary"

}, inplace=True)

# Convert column to category from string

df["Gender"] = df["Gender"].astype("category")

# Convert column to boolean from string

df["ReAdmis"] = df["ReAdmis"].astype("bool")

#Reformat the vitamin D level column from six to three decimal points.

df["VitD\_levels"] = df.VitD\_levels.round(3)

# Convert column to boolean from string

df["Soft\_drink"] = df["Soft\_drink"].astype("bool")

# Convert column to category from string

df["Initial\_admin"] = df["Initial\_admin"].astype("category")

# Convert column to boolean from string

df["HighBlood"] = df["HighBlood"].astype("bool")

# Convert column to boolean from string

df["Stroke"] = df["Stroke"].astype("bool")

# Convert column to category from string

df["Complication\_risk"] = df["Complication\_risk"].astype("category")

# Convert column to boolean from float

df["Overweight"] = df["Overweight"].astype("bool")

# Convert column to boolean from string

df["Arthritis"] = df["Arthritis"].astype("bool")

# Convert column to boolean from string

df["Diabetes"] = df["Diabetes"].astype("bool")

# Convert column to boolean from string

df["Hyperlipidemia"] = df["Hyperlipidemia"].astype("bool")

# Convert column to boolean from string

df["BackPain"] = df["BackPain"].astype("bool")

# Convert column to boolean from float

df["Anxiety"] = df["Anxiety"].astype("bool")

# Convert column to boolean from string

df["Allergic\_rhinitis"] = df["Allergic\_rhinitis"].astype("bool")

# Convert column to boolean from string

df["Reflux\_esophagitis"] = df["Reflux\_esophagitis"].astype("bool")

# Convert column to boolean from string

df["Asthma"] = df["Asthma"].astype("bool")

# Convert column to category from string

df["Services"] = df["Services"].astype("category")

#Since Initial\_days only includes hospital stays longer than one day, it is reasonable to believe that NaNs indicate 0 days.

df.Initial\_days.fillna(0, inplace=True)

# Convert days in hospital from float to integer

df["Initial\_days"] = df["Initial\_days"].astype("int64")

# Reformat currency column from six to two decimal places.

df["TotalCharge"] = df.TotalCharge.round(2)

# Reformat column representing currency in USD to 2 decimal places from 6

df["Additional\_charges"] = df.Additional\_charges.round(2)

#For the survey response columns, created an ordered categorical datatype structure ("1" > "2" >... > "7" > "8").

survey\_scores = CategoricalDtype(categories=["8", "7", "6", "5", "4", "3", "2", "1"], ordered=True)

# Instead, map integers to strings (without this, the conversion from int to ordered categorical will behave strangely).

df["Item1"] = df["Item1"].map(str)

# Convert the strings datatype to the newly generated survey\_scores datatype.

df["Item1"] = df["Item1"].astype(survey\_scores)

# Instead, map integers to strings (without this, the conversion from int to ordered categorical will behave strangely).

df["Item2"] = df["Item2"].map(str)

# Convert the strings datatype to the newly generated survey\_scores datatype.

df["Item2"] = df["Item2"].astype(survey\_scores)

# Instead, map integers to strings (without this, the conversion from int to ordered categorical will behave strangely).

df["Item3"] = df["Item3"].map(str)

Convert the strings datatype to the newly generated survey\_scores datatype.

df["Item3"] = df["Item3"].astype(survey\_scores)

# Instead, map integers to strings (without this, the conversion from int to ordered categorical will behave strangely).

df["Item4"] = df["Item4"].map(str)

Convert the strings datatype to the newly generated survey\_scores datatype.

df["Item4"] = df["Item4"].astype(survey\_scores)

# Instead, map integers to strings (without this, the conversion from int to ordered categorical will behave strangely).

df["Item5"] = df["Item5"].map(str)

# Convert the strings datatype to the newly generated survey\_scores datatype.

df["Item5"] = df["Item5"].astype(survey\_scores)

# Instead, map integers to strings (without this, the conversion from int to ordered categorical will behave strangely).

df["Item6"] = df["Item6"].map(str)

# Convert the strings datatype to the newly generated survey\_scores datatype.

df["Item6"] = df["Item6"].astype(survey\_scores)

# Instead, map integers to strings (without this, the conversion from int to ordered categorical will behave strangely).

df["Item7"] = df["Item7"].map(str)

# Convert the strings datatype to the newly generated survey\_scores datatype.

df["Item7"] = df["Item7"].astype(survey\_scores)

# Instead, map integers to strings (without this, the conversion from int to ordered categorical will behave strangely).

df["Item8"] = df["Item8"].map(str)

# Convert the strings datatype to the newly generated survey\_scores datatype.

df["Item8"] = df["Item8"].astype(survey\_scores)

# erase all rows containing null values (these exist only in Children, Age, and Income)

df.dropna(inplace=True)

df.info()

D5.Copy of Cleaned Data Set

Cleaned data set is submitted as output206.csv attached to the report.

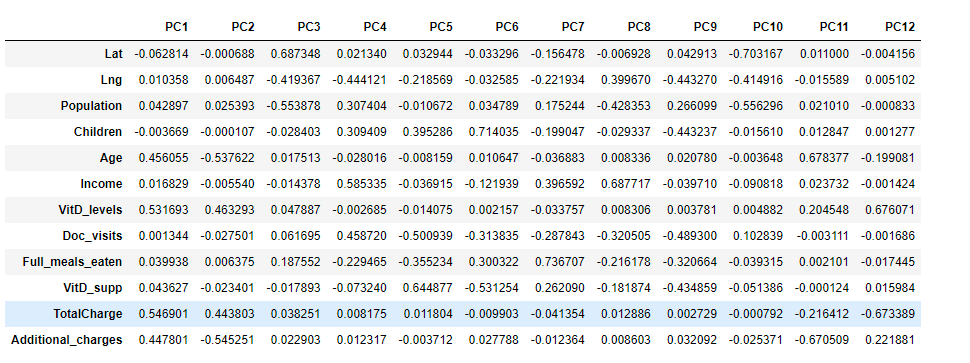
D6.Summarize Limitations of Data Cleaning Process

I struggled with Subject Matter Expert knowledge because there are indeed answers for the missing variables, but I do not know the patients in order to fill them in. Additionally, the missing information is quite vague in referring to the original variable, so an NA in Overweight might have a different meaning than an NA in Anxiety. I noticed Non-binary as an answer doesn't actually indicate that a person is non binary if they originally wrote prefer not to say. Finally a large chunk of data had to be reclassified/recategorized in order to get rid of the nulls and in order to move on with the PCA component.

D7. Discussion of Limitations

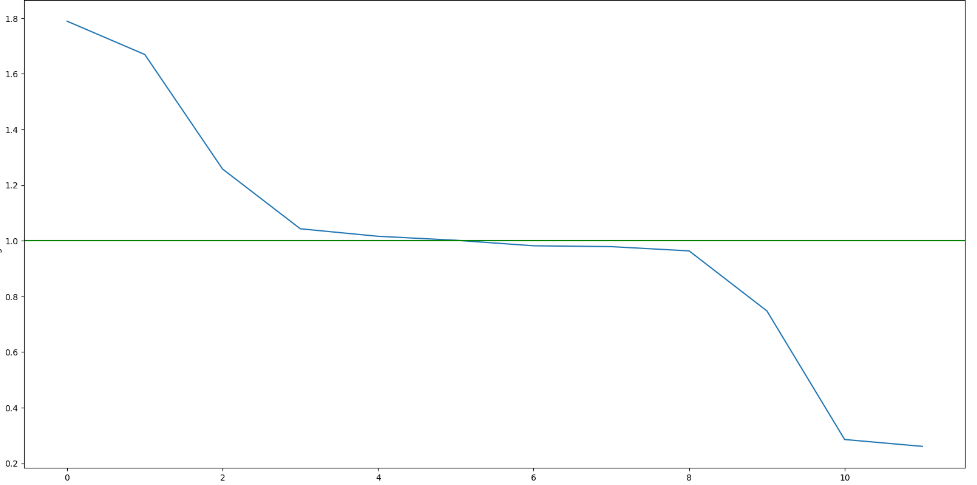
For the lack of subject matter expert knowledge in this dataset, I felt it was impossible to make a true analysis based off of this data set due to the missing information. Since the missing information for each entry is unique to a patient, it wouldn't be accurate to fill in the information myself. For example, in the children column with missing entries, I could do exploratory data analysis to find a number to replace the nulls, but that would be forging information that could equally be wrong, so I went against it. Even with this cleaned data set, I find the information to be almost random instead of actually acquired. I believe its a great dataset to learn how to do techniques and learn how to do analysis on, however for practical purposes with deep insight I imagine it would give any analysts issues.

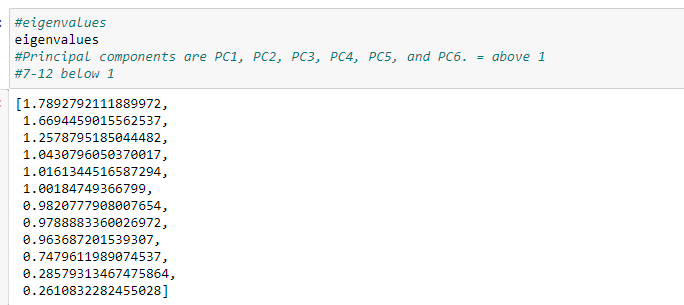
E1:List Principal Components

For The Principal Component Analysis U used latitude, longitude, population, children, age, income, Vitamin D level, doctor visits, full meals, Vitamin D supplements, initial stay, daily charge, and additional charges as the categorical variables. These also were the quantitative variables in the dataset so I used them for the PCA.

E2:Identification of Principal Components

To find the principal components I used a graph to plot the eigenvalues of the principal components, then determined which ones had a value greater than one as instructed by the Kaiser Rule.(wgu courseware)





E3.Benefits of Principal COmponent Analysis

A dataset with many variables can be made simpler by using Principal Component Analysis (PCA), which divides the dataset into fewer dimensions, each of which represents a collection of related variables (Towards Data Science, 2017). The variables in this dataset are not entirely quantitative and associated, which is a requirement of PCA. The "severity of the initial complaint" is reflected in all the factors, including days spent in the hospital, total expenses, and additional charges; a more serious problem usually entails longer hospital stays and greater costs. PCA combines these variables into a single component that may be assessed for its impact on readmission, rather than evaluating each one separately versus readmission. More factors may be included, depending on the details of the study and the variables that are accessible.

F:Panopto Recording

Video is included with submission of report.

G:Code References

[WGU Courseware resources](https://wgu.hosted.panopto.com/Panopto/Pages/Viewer.aspx?id=b627935e-73f8-41ad-b97b-af1e000d3015) was used to help write the code for the PCA portion (Part E) of this project.

H: Source References

[WGU Courseware resources](https://wgu.hosted.panopto.com/Panopto/Pages/Viewer.aspx?id=b627935e-73f8-41ad-b97b-af1e000d3015) was used to help write the code and knowledge in this report.

[Matt Brems, Towards Data Science](https://towardsdatascience.com/a-one-stop-shop-for-principal-component-analysis-5582fb7e0a9c) was used With PCA concepts and understanding.