D206_medical_data

October 26, 2021

1 D206 Data Cleaning - Medical Data Performance Assessment

Andrew Butler
Student Id: 001053557
MSDA
Mentor: Linda Howell

1.1 Part I: Research Question

1.1.1 A. Question

Can we determine the likelihood that a patient will be readmitted, if so can we isolate the factors that contribute the most to readmission and develop strategies to mitigate them.

1.1.2 B. Description of Variables

The dataset contains 10,000 semi-anonymized records containing various information about the patient and their treatment including demographic information, readmission status, medical history, and treatment details. For each record there are 50 variables, which are described here:

- CaseOrder(categorical): A placeholder variable to preserve the original order of the raw data file.
- Customer_id(categorical): Unique patient ID.
- Interaction, UID(categorical): Internal identifying variable.
- City(categorical): Patient city of residence.
- State(categorical): Patient state of residence.
- County(categorical): Patient county of residence.
- Zip(categorical): Patient zip code of residence.
- Lat(categorical), Lng(categorical): GPS coordinates of patient residence.
- Population(numeric): Population within a mile radius of patient, based on census data.
- Area(categorical): Area type (rural, urban, suburban).
- TimeZone(categorical): Time zone of patient residence.
- Job(categorical): Occupation of the patient (or primary insurance holder).
- Children(numeric): Number of children in the patient's household.
- Age(numeric): Age of the patient.
- Education(categorical): Highest earned degree of patient.
- Employment(categorical): Employment status of patient.
- Income(numeric): Annual income of the patient (or primary insurance holder).
- Marital(categorical): Marital status of the patient (or primary insurance holder).
- Gender(categorical): Patient self-identification as male, female, or non-binary.

- ReAdmis(categorical): Whether or not the patient was readmitted within a month of release.
- VitD_levels(numeric): The patient's vitamin D levels as measured in ng/mL.
- Doc_visits(numeric): Number of times the primary physician visited the patient during the initial hospitalization.
- Full_meals_eaten(numeric): Number of full meals the patient ate while hospitalized (partial meals count as 0, and some patients had more than three meals in a day if requested).
- VitD_supp(numeric): The number of times that vitamin D supplements were administered to the patient.
- Soft_drink(categorical): Whether or not the patient habitually drinks three or more sodas in a day.
- Initial_admin(categorical): The means by which the patient was admitted into the hospital initially (emergency admission, elective admission, observation).
- HighBlood(categorical): Whether or not the patient has high blood pressure.
- Stroke(categorical): Whether or not the patient has had a stroke.
- Complication_risk(categorical): Level of complication risk for the patient(high, medium, low).
- Overweight(categorical): Whether or not the patient is considered overweight.
- Arthritis(categorical): Whether or not the patient has arthritis.
- Diabetes(categorical): Whether or not the patient has diabetes.
- Hyperlipidemia(categorical): Whether the patient has hyperlipidemia.
- BackPain(categorical): Whether or not the patient has chronic back pain.
- Anxiety(categorical): Whether or not the patient has an anxiety disorder.
- Allergic_rhinitis(categorical): Whether or not the patient has allergic rhinitis.
- Reflux_esophagitis(categorical): Whether or not the patient has reflux esophagitis.
- Asthma(categorical): Whether or not the patient has asthma.
- Services(categorical): Primary service the patient received while hospitalized (blood work, intravenous, CT scan, MRI).
- Initial_days(numeric): The number of days the patient stayed in the hospital during the initial visit.
- TotalCharge(numeric): The amount charged to the patient daily. This value reflects an average per patient based on the total charge divided by the number of days hospitalized. This amount reflects the typical charges billed to patients, not including specialized treatments.
- Additional_charges(numeric): The average amount charged to the patient for miscellaneous procedures, treatments, medicines, anesthesiology, etc.

  The following variables represent responses to an eight-question survey asking customers to rate the importance of various factors/surfaces on a scale of 1 to 8 (1 = most important, 8 = least important)

```
>-  Item1(categorical): Timely admission
>-  Item2(categorical): Timely treatment
>-  Item3(categorical): Timely visits
>-  Item4(categorical): Reliability
>-  Item5(categorical): Options
>-  Item6(categorical): Hours of treatment
>-  Item7(categorical): Courteous staff
>-  Item8(categorical): Evidence of active listening from doctor
```

1.2 Part II: Data-Cleaning Plan

1.2.1 C. Explanation of data cleaning plan

1. My plan for cleaning the data set will follow these steps:
    1. Import the raw data set and converting it to a dataframe using the read_csv function provided by the pandas library.
    2. Use the functions provided by Pandas to inspect the structure of the data and get detailed information about variables
    3. Remove redundant columns, columns that potentially contain PID, and columns that will not contribute meaningfully to analysis.
    4. Standardize column names, and update column names that are vague or ambiguous to be more descriptive.
    5. Check for duplicate rows, or rows that only contain null values and drop them.
    6. Determine which columns contain null values, and impute null values and add them to a separate dataframe.
        - impute categorical variables using mode.
        - use histograms to analyze numerical data columns that contain nulls and determine the best method to impute null values.
    7. Merge changes from last step into main data frame.
    8. Use pandas to view unique values of each column.
    9. Analyze the results of the previous step to determine if any columns contain incorrect data, need to be converted to a different data type, or have their precision reduced and take those actions as needed.
    10. Isolate numeric values for outlier detection and add them to a separate dataframe.
    11. Calculate Z-scores for numeric data, and use Z-scores and box plots to identify outliers.
    12. Add column identifying outliers for each numeric column to main dataframe.
    13. Re-express ordinal and binary categorical variables.
    14. Preform Principal Component Analysis.
    15. Export cleaned data as csv.

2. Approach
    - Because the data set contains an amount of missing data that cannot be simply dropped without substantially skewing the data, I will analyze each column that has missing values and determine the best method for imputation.
    - Several columns contain outliers, but many of them fall within acceptable ranges for their type, so I have opted to add an additional column to the dataset identifying when a variable is an outlier, rather than removing or impuning them, so they can easily be included or excluded in future analysis.
    - I will reduce the precision of values where it is necessary, or re-express the categories of columns where it would meaningfully reduce the amount of categories for that variable without causing a meaningful loss of information.
3. I have decided to use Python 3 in a Jupyter notebook environment to analyze and clean the dataset, because of my familiarity with its ecosystem, the ease of presenting my findings that Jupyter notebooks provides, and the availability of specialized tools and packages for data analysis. I will be using the flowing packages:
    - numpy - required for pandas
    - pandas - to organize and manipulate data into data frames.
    - matplotlib, seaborn - for creating charts to aid in analysis.

- scipy: libraries that provides statistical functions.
- sklearn: provides models to preform PCA.

4. The code I am using preform the above mentioned steps and and the results of its execution is shown here:

[11]:
```
#install necessary packages if not already installed
import sys
!conda install --yes --prefix {sys.prefix} pandas numpy matplotlib seaborn␣
↪scipy scikit-learn
```

```
Collecting package metadata (current_repodata.json): …working… done
Solving environment: …working… done

# All requested packages already installed.
```

[12]:
```
#import necessary packages

import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from scipy import stats
from sklearn.decomposition import PCA
%matplotlib inline
```

[13]:
```
#import raw data as dataframe
df = pd.read_csv('medical_raw_data.csv')
#inspect structure of data
print(df.shape)
```

```
(10000, 53)
```

[14]:
```
print(df.describe())
```

```
        Unnamed: 0    CaseOrder           Zip           Lat           Lng  \
count  10000.00000  10000.00000  10000.000000  10000.000000  10000.000000
mean    5000.50000   5000.50000  50159.323900     38.751099    -91.243080
std     2886.89568   2886.89568  27469.588208      5.403085     15.205998
min        1.00000      1.00000    610.000000     17.967190   -174.209690
25%     2500.75000   2500.75000  27592.000000     35.255120    -97.352982
50%     5000.50000   5000.50000  50207.000000     39.419355    -88.397230
75%     7500.25000   7500.25000  72411.750000     42.044175    -80.438050
max    10000.00000  10000.00000  99929.000000     70.560990    -65.290170

         Population     Children          Age        Income    VitD_levels  \
count  10000.000000  7412.000000  7586.000000   7536.000000  10000.000000
mean    9965.253800     2.098219    53.295676  40484.438268     19.412675
std    14824.758614     2.155427    20.659182  28664.861050      6.723277
```

```
min         0.000000     0.000000    18.000000     154.080000      9.519012
25%       694.750000     0.000000    35.000000   19450.792500     16.513171
50%      2769.000000     1.000000    53.000000   33942.280000     18.080560
75%     13945.000000     3.000000    71.000000   54075.235000     19.789740
max    122814.000000    10.000000    89.000000  207249.130000     53.019124

       …   TotalCharge  Additional_charges         Item1         Item2  \
count  …  10000.000000        10000.000000  10000.000000  10000.000000
mean   …   5891.538261        12934.528586      3.518800      3.506700
std    …   3377.558136         6542.601544      1.031966      1.034825
min    …   1256.751699         3125.702716      1.000000      1.000000
25%    …   3253.239465         7986.487642      3.000000      3.000000
50%    …   5852.250564        11573.979365      4.000000      3.000000
75%    …   7614.989701        15626.491033      4.000000      4.000000
max    …  21524.224210        30566.073130      8.000000      7.000000

              Item3         Item4         Item5         Item6         Item7  \
count  10000.000000  10000.000000  10000.000000  10000.000000  10000.000000
mean       3.511100      3.515100      3.496900      3.522500      3.494000
std        1.032755      1.036282      1.030192      1.032376      1.021405
min        1.000000      1.000000      1.000000      1.000000      1.000000
25%        3.000000      3.000000      3.000000      3.000000      3.000000
50%        4.000000      4.000000      3.000000      4.000000      3.000000
75%        4.000000      4.000000      4.000000      4.000000      4.000000
max        8.000000      7.000000      7.000000      7.000000      7.000000

              Item8
count  10000.000000
mean       3.509700
std        1.042312
min        1.000000
25%        3.000000
50%        3.000000
75%        4.000000
max        7.000000

[8 rows x 26 columns]
```

[15]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10000 entries, 0 to 9999
Data columns (total 53 columns):
 #   Column                Non-Null Count  Dtype
---  ------                --------------  -----
 0   Unnamed: 0            10000 non-null  int64
 1   CaseOrder             10000 non-null  int64
 2   Customer_id           10000 non-null  object
```

```
3   Interaction         10000 non-null  object
4   UID                 10000 non-null  object
5   City                10000 non-null  object
6   State               10000 non-null  object
7   County              10000 non-null  object
8   Zip                 10000 non-null  int64
9   Lat                 10000 non-null  float64
10  Lng                 10000 non-null  float64
11  Population          10000 non-null  int64
12  Area                10000 non-null  object
13  Timezone            10000 non-null  object
14  Job                 10000 non-null  object
15  Children            7412 non-null   float64
16  Age                 7586 non-null   float64
17  Education           10000 non-null  object
18  Employment          10000 non-null  object
19  Income              7536 non-null   float64
20  Marital             10000 non-null  object
21  Gender              10000 non-null  object
22  ReAdmis             10000 non-null  object
23  VitD_levels         10000 non-null  float64
24  Doc_visits          10000 non-null  int64
25  Full_meals_eaten    10000 non-null  int64
26  VitD_supp           10000 non-null  int64
27  Soft_drink          7533 non-null   object
28  Initial_admin       10000 non-null  object
29  HighBlood           10000 non-null  object
30  Stroke              10000 non-null  object
31  Complication_risk   10000 non-null  object
32  Overweight          9018 non-null   float64
33  Arthritis           10000 non-null  object
34  Diabetes            10000 non-null  object
35  Hyperlipidemia      10000 non-null  object
36  BackPain            10000 non-null  object
37  Anxiety             9016 non-null   float64
38  Allergic_rhinitis   10000 non-null  object
39  Reflux_esophagitis  10000 non-null  object
40  Asthma              10000 non-null  object
41  Services            10000 non-null  object
42  Initial_days        8944 non-null   float64
43  TotalCharge         10000 non-null  float64
44  Additional_charges  10000 non-null  float64
45  Item1               10000 non-null  int64
46  Item2               10000 non-null  int64
47  Item3               10000 non-null  int64
48  Item4               10000 non-null  int64
49  Item5               10000 non-null  int64
50  Item6               10000 non-null  int64
```

```
 51  Item7                10000 non-null  int64
 52  Item8                10000 non-null  int64
dtypes: float64(11), int64(15), object(27)
memory usage: 4.0+ MB
```

[16]:
```
df.head()
```

[16]:
```
   Unnamed: 0  CaseOrder Customer_id                           Interaction  \
0           1          1     C412403  8cd49b13-f45a-4b47-a2bd-173ffa932c2f
1           2          2     Z919181  d2450b70-0337-4406-bdbb-bc1037f1734c
2           3          3     F995323  a2057123-abf5-4a2c-abad-8ffe33512562
3           4          4     A879973  1dec528d-eb34-4079-adce-0d7a40e82205
4           5          5     C544523  5885f56b-d6da-43a3-8760-83583af94266

                                UID          City State        County    Zip  \
0  3a83ddb66e2ae73798bdf1d705dc0932           Eva    AL        Morgan  35621
1  176354c5eef714957d486009feabf195      Marianna    FL       Jackson  32446
2  e19a0fa00aeda885b8a436757e889bc9   Sioux Falls    SD     Minnehaha  57110
3  cd17d7b6d152cb6f23957346d11c3f07  New Richland    MN        Waseca  56072
4  d2f0425877b10ed6bb381f3e2579424a    West Point    VA  King William  23181

        Lat  …  TotalCharge  Additional_charges Item1 Item2 Item3  Item4  \
0  34.34960  …  3191.048774        17939.403420     3     3     2      2
1  30.84513  …  4214.905346        17612.998120     3     4     3      4
2  43.54321  …  2177.586768        17505.192460     2     4     4      4
3  43.89744  …  2465.118965        12993.437350     3     5     5      3
4  37.59894  …  1885.655137         3716.525786     2     1     3      3

  Item5 Item6 Item7  Item8
0     4     3     3      4
1     4     4     3      3
2     3     4     3      3
3     4     5     5      5
4     5     3     4      3

[5 rows x 53 columns]
```

[17]:
```
#remove redundant columns, and columns that will not contribute meaningfully to␣
↪analysis
#column Unnamed: 0 is removed because it is functionally identical to␣
↪CaseOrder, Lat and Lng are
#dropped due to being personally identifiable information about patient that do␣
↪not contribute
#more meaningfully to analysis than anonomized columns such as Area and␣
↪Timezone already do.
#other columns are dropped due to being internal system labels that are not␣
↪useful for analysis.
```

```
#set index to colum CaseOrder
df = df.drop(columns=["Unnamed: 0", "Customer_id", "Interaction", "UID", "Lat",␣
↪"Lng"])
```

[18]:
```
df.rename(columns={"CaseOrder" : "Case_order"}, inplace=True)
df = df.set_index("Case_order", drop = True)
df.head()
```

[18]:
```
                   City State        County    Zip  Population      Area  \
Case_order
1                   Eva    AL        Morgan  35621        2951  Suburban
2              Marianna    FL       Jackson  32446       11303     Urban
3           Sioux Falls    SD     Minnehaha  57110       17125  Suburban
4          New Richland    MN        Waseca  56072        2162  Suburban
5            West Point    VA  King William  23181        5287     Rural

                   Timezone                              Job  Children  \
Case_order
1           America/Chicago  Psychologist, sport and exercise       1.0
2           America/Chicago      Community development worker       3.0
3           America/Chicago           Chief Executive Officer       3.0
4           America/Chicago               Early years teacher       0.0
5          America/New_York       Health promotion specialist       NaN

             Age  …  TotalCharge Additional_charges  Item1 Item2 Item3  \
Case_order       …
1           53.0  …  3191.048774       17939.403420      3     3     2
2           51.0  …  4214.905346       17612.998120      3     4     3
3           53.0  …  2177.586768       17505.192460      2     4     4
4           78.0  …  2465.118965       12993.437350      3     5     5
5           22.0  …  1885.655137        3716.525786      2     1     3

           Item4  Item5  Item6  Item7  Item8
Case_order
1              2      4      3      3      4
2              4      4      4      3      3
3              4      3      4      3      3
4              3      4      5      5      5
5              3      5      3      4      3

[5 rows x 46 columns]
```

[19]:
```
df.columns
```

[19]:
```
Index(['City', 'State', 'County', 'Zip', 'Population', 'Area', 'Timezone',
       'Job', 'Children', 'Age', 'Education', 'Employment', 'Income',
       'Marital', 'Gender', 'ReAdmis', 'VitD_levels', 'Doc_visits',
```

```
        'Full_meals_eaten', 'VitD_supp', 'Soft_drink', 'Initial_admin',
        'HighBlood', 'Stroke', 'Complication_risk', 'Overweight', 'Arthritis',
        'Diabetes', 'Hyperlipidemia', 'BackPain', 'Anxiety',
        'Allergic_rhinitis', 'Reflux_esophagitis', 'Asthma', 'Services',
        'Initial_days', 'TotalCharge', 'Additional_charges', 'Item1', 'Item2',
        'Item3', 'Item4', 'Item5', 'Item6', 'Item7', 'Item8'],
      dtype='object')
```

[20]:
```
#standardise column names, and update column names to be more descriptive
df.rename(columns={"Marital": "Mariage_status", "ReAdmis": "Readmited",␣
 ↪"VitD_supp": "VitD_supplements",
                   "Soft_drink": "Habitual_soft_drink_use", "BackPain":␣
 ↪"Back_pain", "Services":
                   "Primary_service_recived", "HighBlood":␣
 ↪"High_blood_pressure", "TotalCharge": "Total_charge",
                   "Item1": "Survey_timely_addmission", "Item2":␣
 ↪"Survey_timely_treatment",
                   "Item3": "Survey_timely_visits", "Item4":␣
 ↪"Survey_reliability",
                   "Item5": "Survey_options", "Item6": "Survey_hours",
                   "Item7": "Survey_courtesy", "Item8":␣
 ↪"Survey_active_listening"}, inplace=True)
```

[21]:
```
#check for duplicated rows
df.duplicated().any()
```

[21]: False

[22]:
```
#check if any rows contain only null values
df.isnull().all(axis=1).any()
```

[22]: False

[23]:
```
#determine which columns contain null values
contains_missing = df.loc[:,df.isnull().any()].copy()
contains_missing
```

[23]:

```
            Children   Age    Income Habitual_soft_drink_use  Overweight  \
Case_order
1                1.0  53.0  86575.93                     NaN         0.0
2                3.0  51.0  46805.99                      No         1.0
3                3.0  53.0  14370.14                      No         1.0
4                0.0  78.0  39741.49                      No         0.0
5                NaN  22.0   1209.56                     Yes         0.0
…                  …     …         …                       …           …
9996             NaN  25.0  45967.61                      No         NaN
9997             4.0  87.0  14983.02                      No         1.0
```

```
9998               3.0   NaN  65917.81                     Yes         1.0
9999               3.0  43.0  29702.32                      No         1.0
10000              8.0   NaN  62682.63                      No         1.0

            Anxiety  Initial_days
Case_order
1               1.0     10.585770
2               NaN     15.129562
3               NaN      4.772177
4               NaN      1.714879
5               0.0      1.254807
…               …         …
9996            1.0     51.561217
9997            0.0     68.668237
9998            1.0           NaN
9999            0.0     63.356903
10000           0.0     70.850592

[10000 rows x 7 columns]
```

[24]:
```
#use mode to impute missing values in catagorical varibles(columns:␣
↪Habitual_soft_drink_use, Overweight, Anxiety),
#mode is used due to the varibles being catagorical
contains_missing['Habitual_soft_drink_use'].
↪fillna(contains_missing['Habitual_soft_drink_use'].mode()[0], inplace = True)
contains_missing['Overweight'].fillna(contains_missing['Overweight'].mode()[0],␣
↪inplace = True)
contains_missing['Anxiety'].fillna(contains_missing['Anxiety'].mode()[0],␣
↪inplace = True)
contains_missing
```

[24]:
```
            Children   Age    Income Habitual_soft_drink_use  Overweight  \
Case_order
1                1.0  53.0  86575.93                      No         0.0
2                3.0  51.0  46805.99                      No         1.0
3                3.0  53.0  14370.14                      No         1.0
4                0.0  78.0  39741.49                      No         0.0
5                NaN  22.0   1209.56                     Yes         0.0
…                …     …       …                     …         …
9996             NaN  25.0  45967.61                      No         1.0
9997             4.0  87.0  14983.02                      No         1.0
9998             3.0   NaN  65917.81                     Yes         1.0
9999             3.0  43.0  29702.32                      No         1.0
10000            8.0   NaN  62682.63                      No         1.0

            Anxiety  Initial_days
Case_order
```

```
1                1.0     10.585770
2                0.0     15.129562
3                0.0      4.772177
4                0.0      1.714879
5                0.0      1.254807
…              …          …
9996             1.0     51.561217
9997             0.0     68.668237
9998             1.0           NaN
9999             0.0     63.356903
10000            0.0     70.850592

[10000 rows x 7 columns]
```

[25]:
```
contains_missing['Habitual_soft_drink_use'].value_counts()
```

[25]:
```
No     8056
Yes    1944
Name: Habitual_soft_drink_use, dtype: int64
```

[26]:
```
contains_missing['Overweight'].value_counts()
```

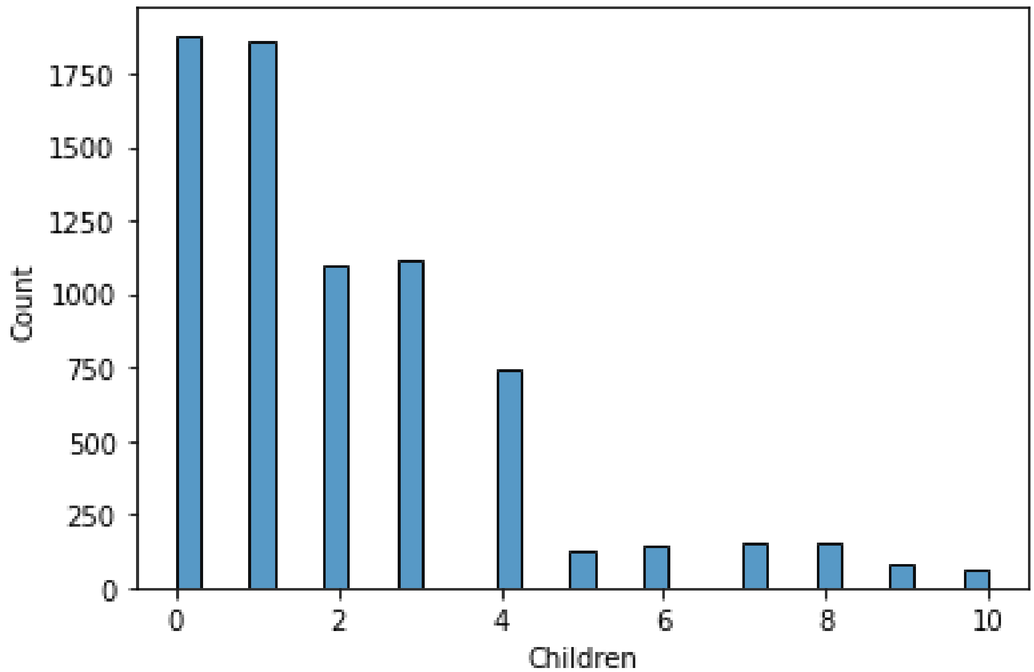
[26]:
```
1.0    7377
0.0    2623
Name: Overweight, dtype: int64
```

[27]:
```
contains_missing['Anxiety'].value_counts()
```

[27]:
```
0.0    7094
1.0    2906
Name: Anxiety, dtype: int64
```

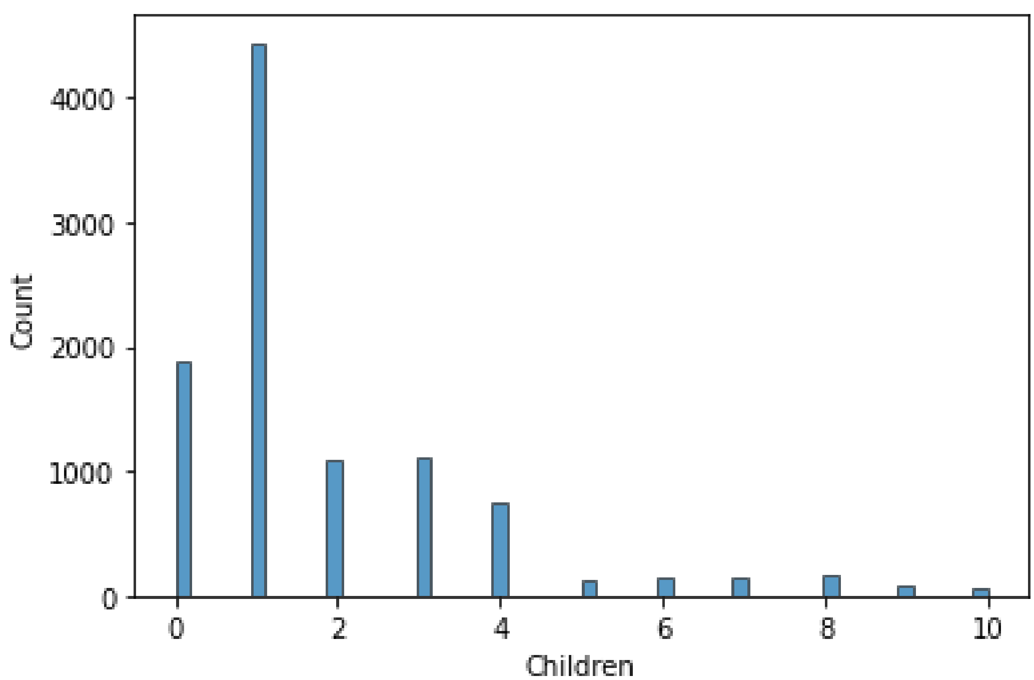
[28]:
```
sns.histplot(contains_missing['Children'])
```

[28]:
```
<AxesSubplot:xlabel='Children', ylabel='Count'>
```



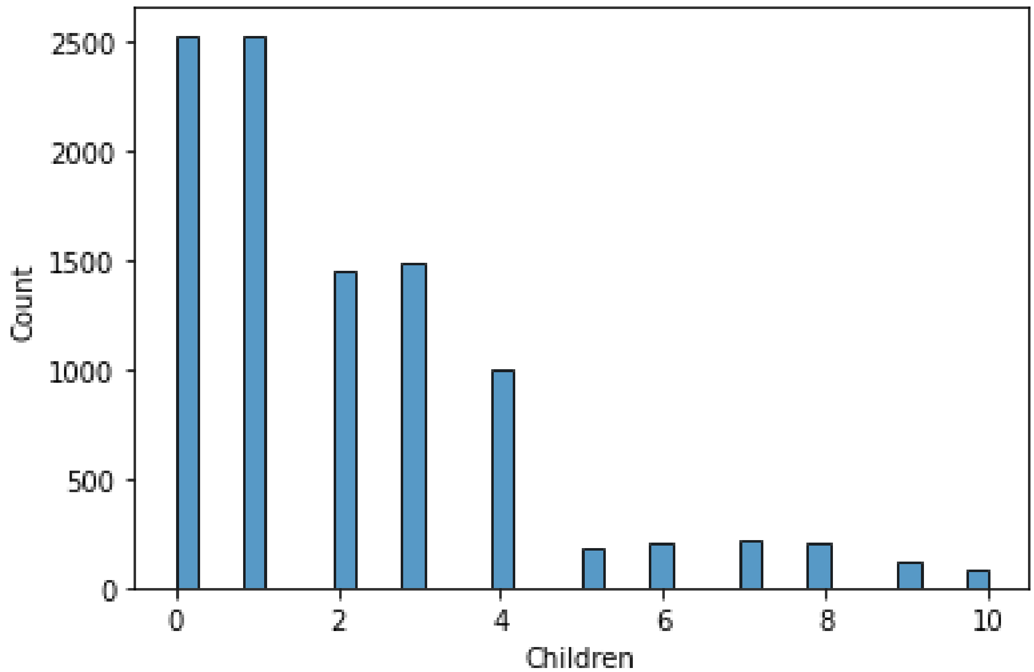
[29]:
```
sns.histplot(contains_missing['Children'].fillna(contains_missing['Children'].
    ↪median()))
```

[29]: <AxesSubplot:xlabel='Children', ylabel='Count'>



[30]:
```
sns.histplot(contains_missing['Children'].interpolate(method='pad'))
```

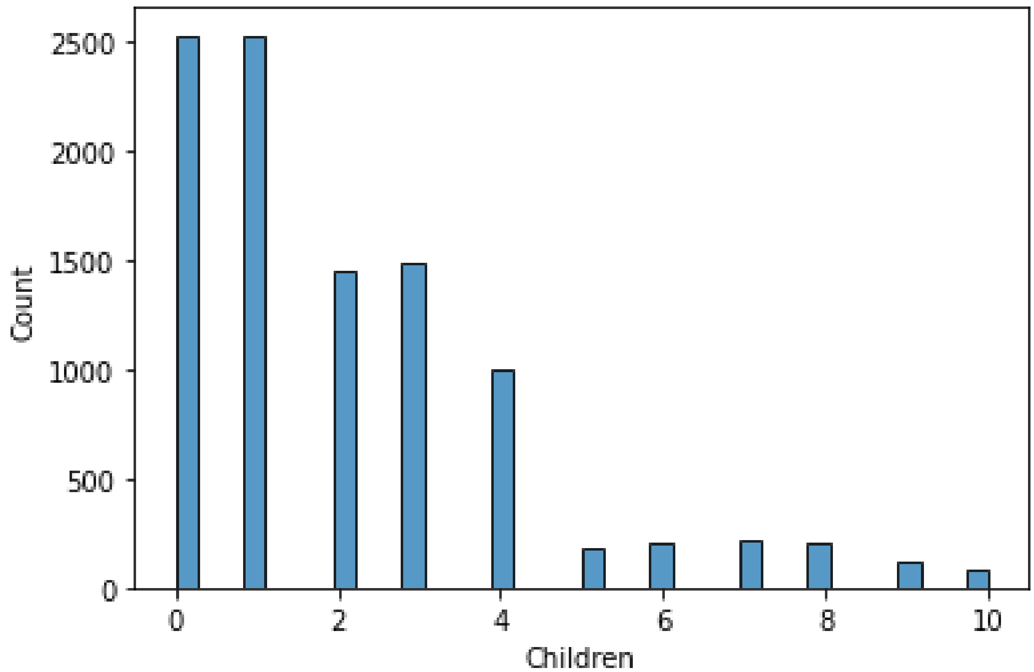
[30]: <AxesSubplot:xlabel='Children', ylabel='Count'>



[31]:
```
#use interpolation to impune missing data for number of children. pad method is␣
↪used to avoid adding values that are
#not whole numbers, interpolation is used because data skews to the right and␣
↪using median to impune values caused
# the amount of data points equaling 1 to more than double
contains_missing['Children'].interpolate(method='pad', inplace=True)
```

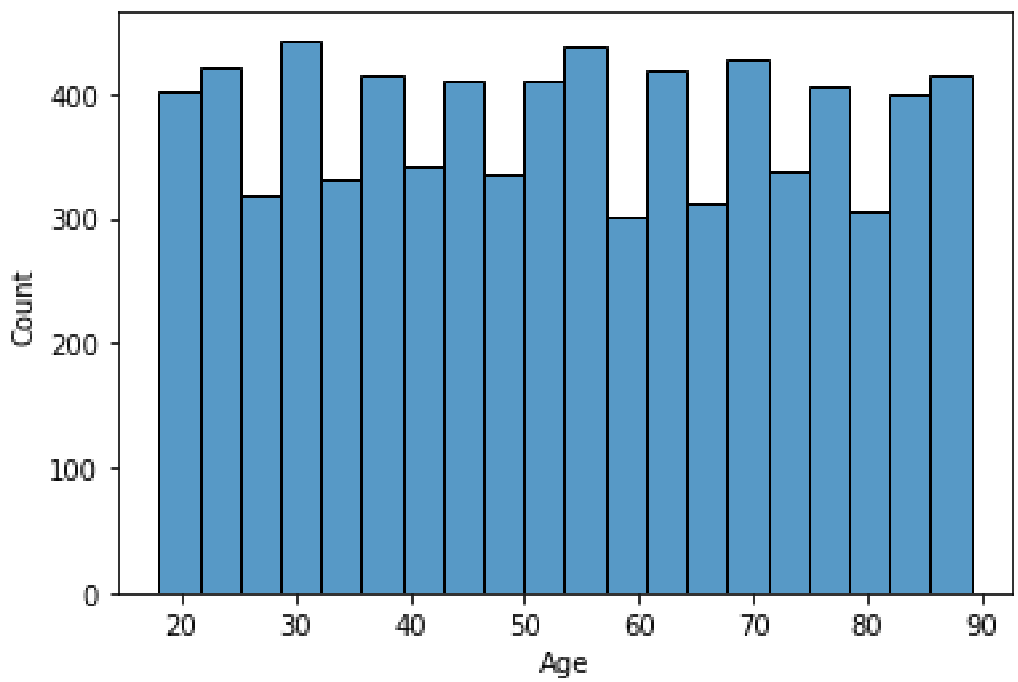
[32]:
```
sns.histplot(contains_missing['Children'])
```

[32]: <AxesSubplot:xlabel='Children', ylabel='Count'>



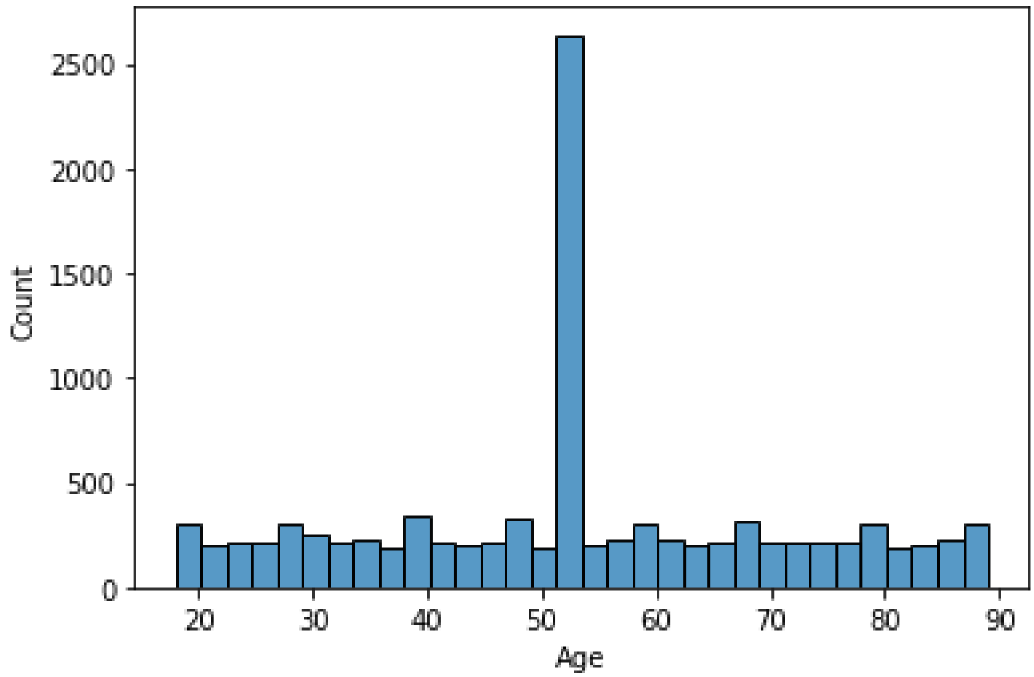
[33]:
```
sns.histplot(contains_missing['Age'])
```

[33]: <AxesSubplot:xlabel='Age', ylabel='Count'>



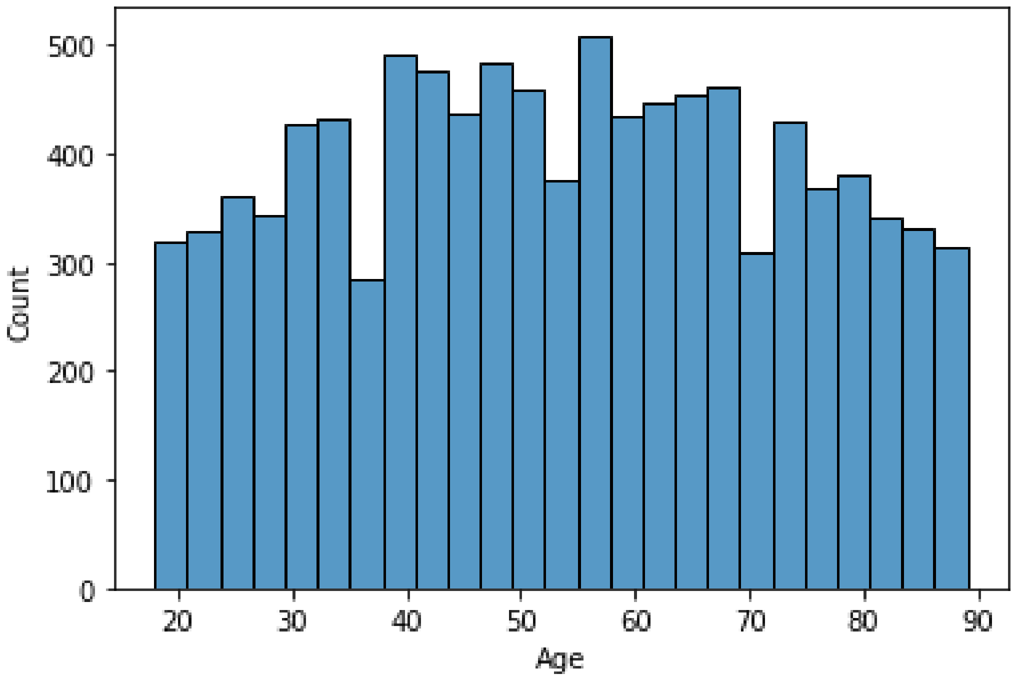
[34]:
```
sns.histplot(contains_missing['Age'].fillna(contains_missing['Age'].mean()))
```

[34]: <AxesSubplot:xlabel='Age', ylabel='Count'>



[35]:
```
sns.histplot(contains_missing['Age'].interpolate())
```

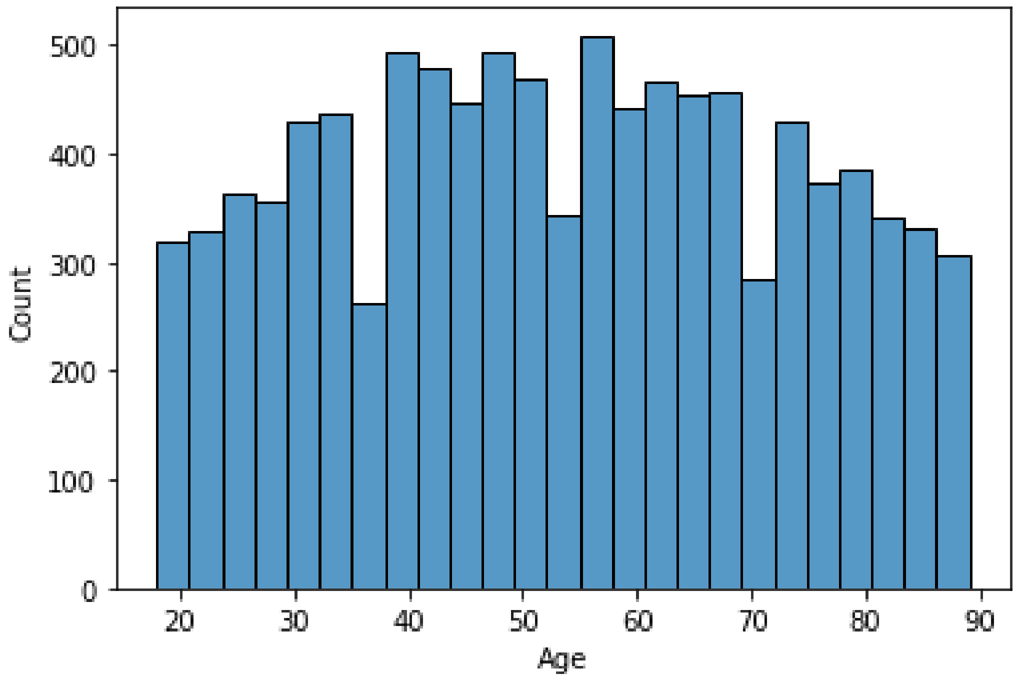
[35]: <AxesSubplot:xlabel='Age', ylabel='Count'>



[36]:
```
#use interpolation to impune missing data for number patient age. interpolation␣
↪is used because histogram revealed
#that data is evenly distributed, and using mean created a drastic change in␣
↪the distribution.
contains_missing.interpolate(inplace=True)
contains_missing['Age'] = contains_missing['Age'].astype(int)
```

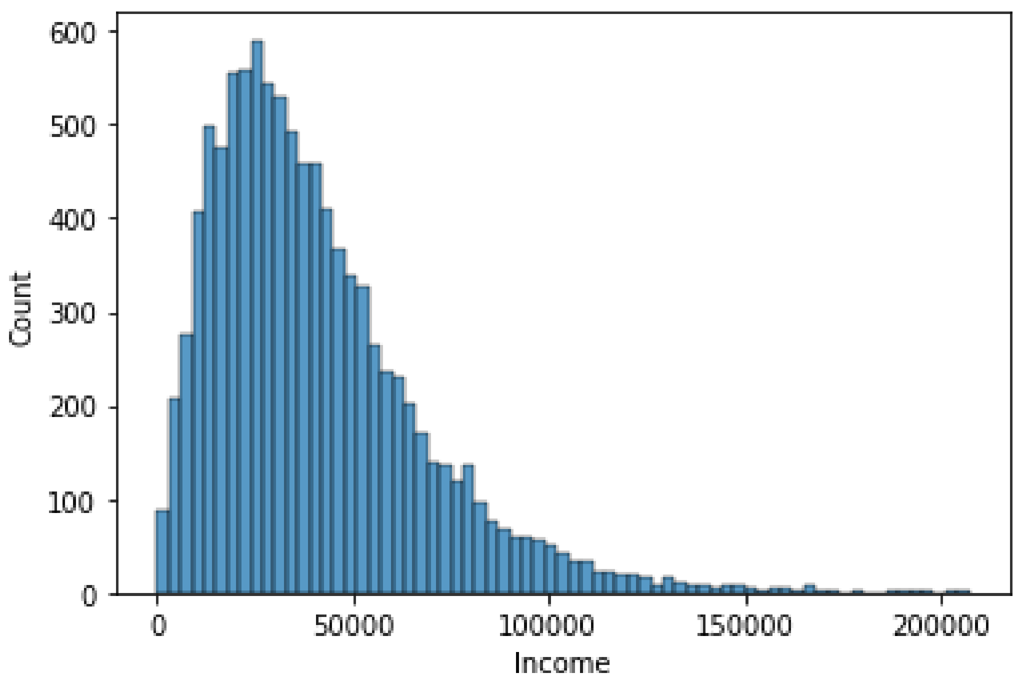
[37]:
```
sns.histplot(contains_missing['Age'])
```

[37]: <AxesSubplot:xlabel='Age', ylabel='Count'>



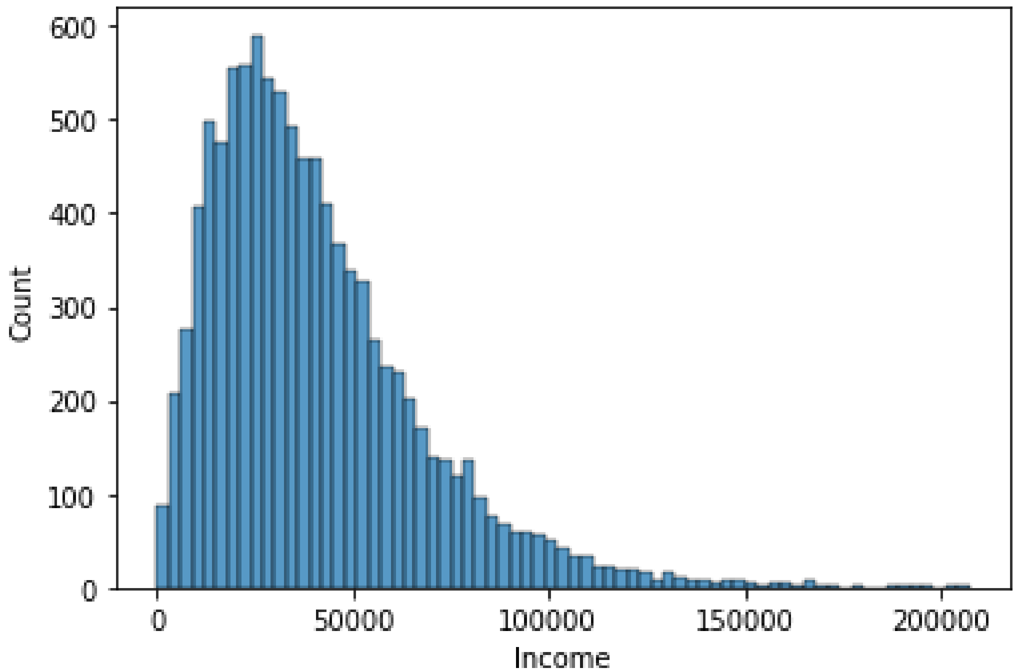
[38]:
```
sns.histplot(contains_missing['Income'])
```

[38]: <AxesSubplot:xlabel='Income', ylabel='Count'>



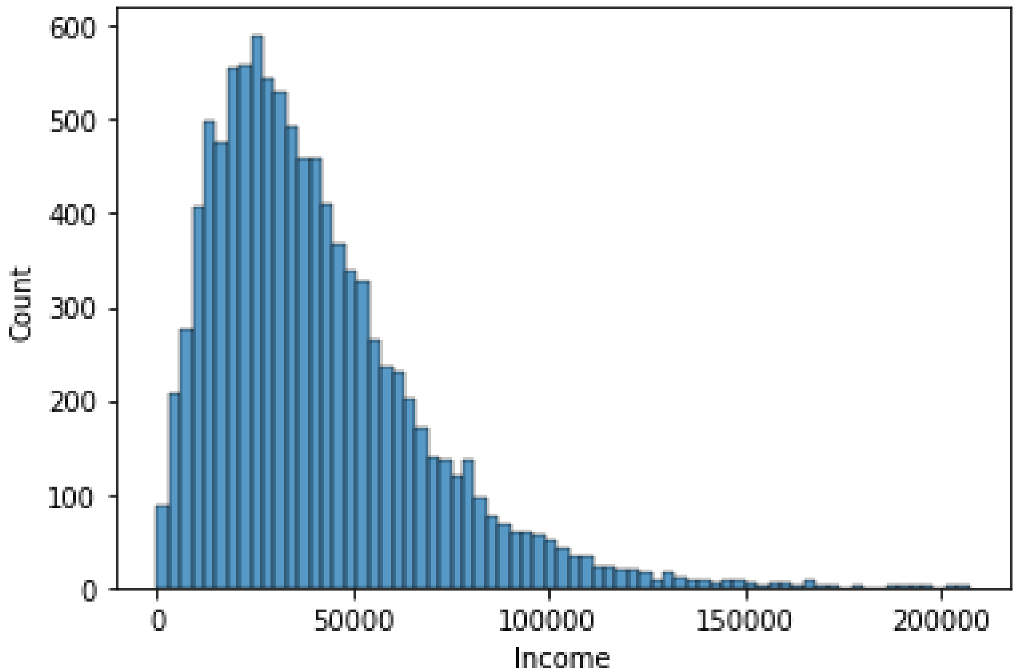
[39]:
```
sns.histplot(contains_missing['Income'].fillna(contains_missing['Income'].
 ↪median()))
```

[39]: <AxesSubplot:xlabel='Income', ylabel='Count'>



[40]:
```
sns.histplot(contains_missing['Income'].interpolate())
```

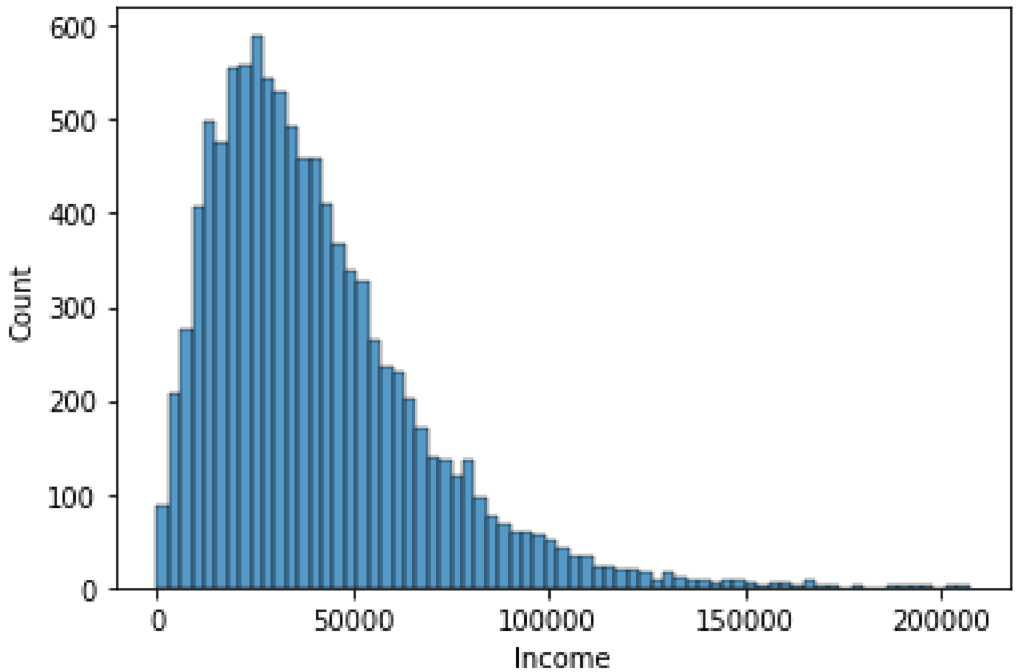
[40]: <AxesSubplot:xlabel='Income', ylabel='Count'>



[41]:
```
#use median value to fill missing values in income column. median was chosen as␣
↪imputation method because histogram
#reveled that data skews to the right, and there was no descernible difference␣
↪bettwen imputation and interpolation
#in maintaining distrobution
contains_missing['Income'].fillna(contains_missing['Income'].median(), inplace␣
↪= True)
```

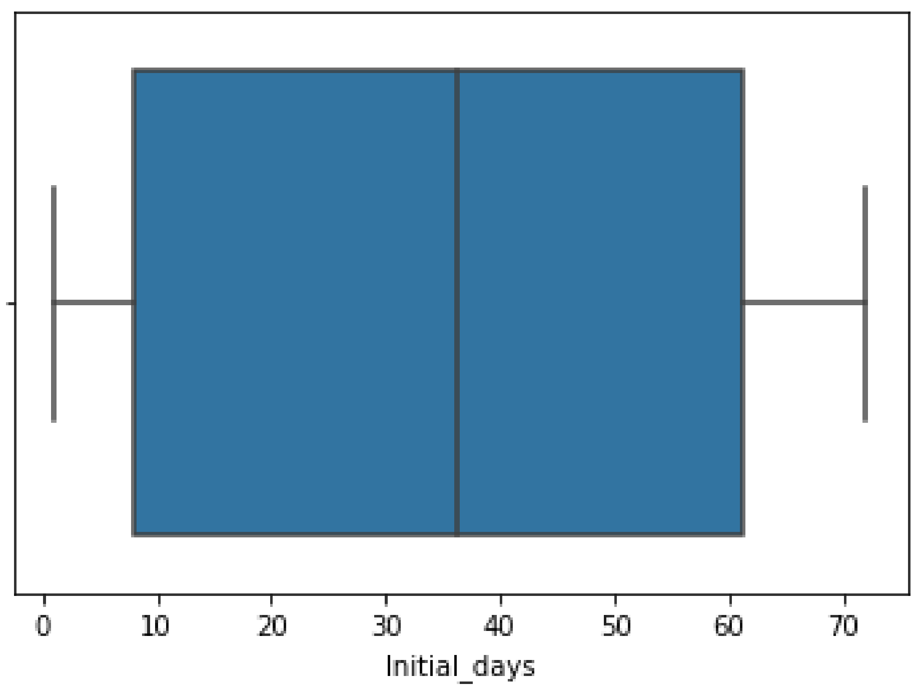
[42]:
```
sns.histplot(contains_missing['Income'])
```

[42]: <AxesSubplot:xlabel='Income', ylabel='Count'>



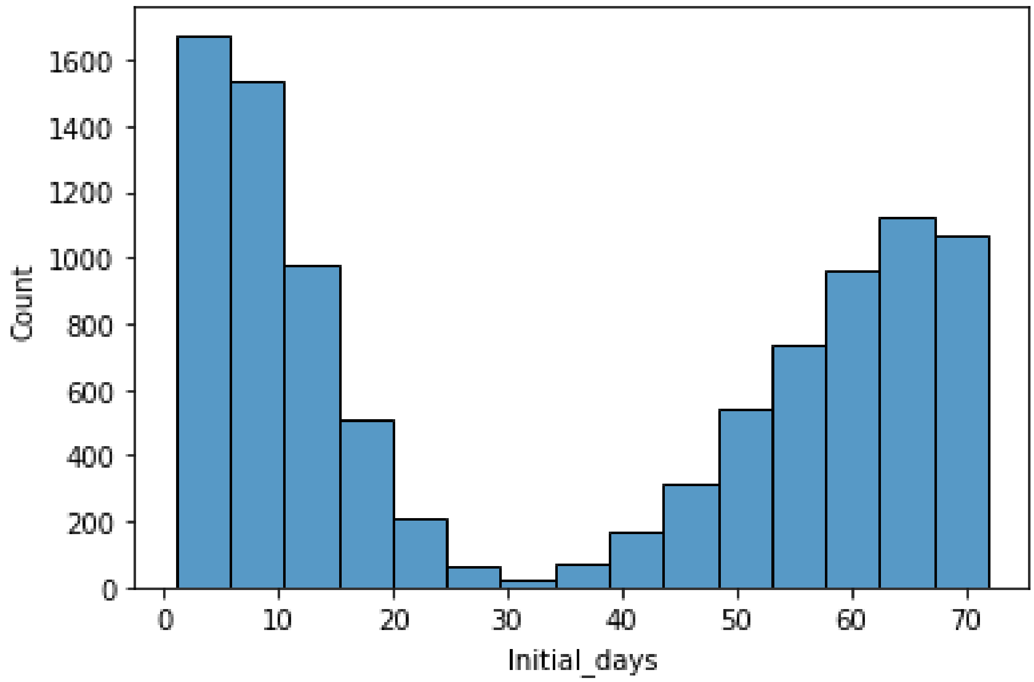
[43]:
```
sns.boxplot(x=contains_missing['Initial_days'])
```

[43]: <AxesSubplot:xlabel='Initial_days'>



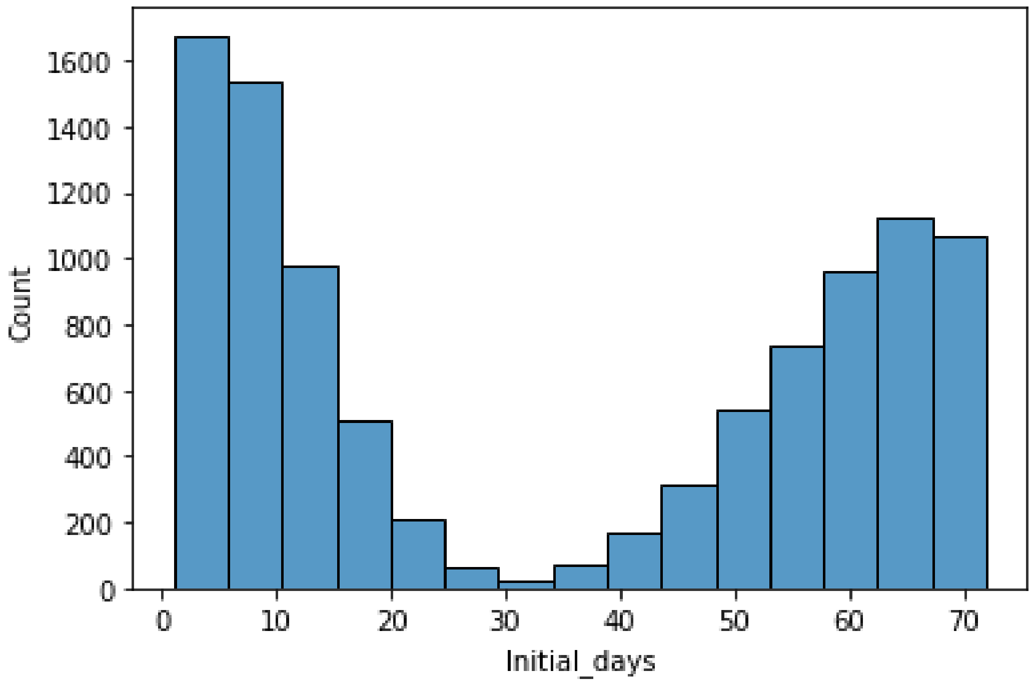
[44]:
```
sns.histplot(contains_missing['Initial_days'])
```

[44]: <AxesSubplot:xlabel='Initial_days', ylabel='Count'>



[45]:
```
sns.histplot(contains_missing['Initial_days'].
↪fillna(contains_missing['Initial_days'].mean()))
```

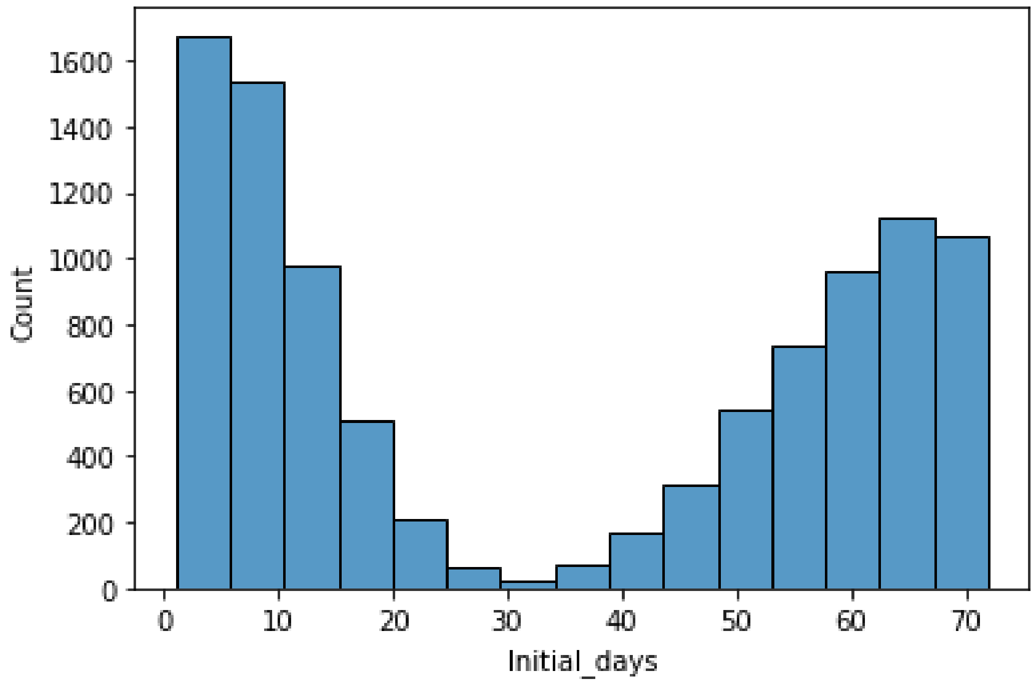
[45]: <AxesSubplot:xlabel='Initial_days', ylabel='Count'>



[46]:
```
#use mean to impune missing data for number patient age. mean is used because␣
↪histogram and box plot revealed
#that data has a bimodal distribution, and using mean maintains this␣
↪distribution.
contains_missing['Initial_days'].fillna(contains_missing['Initial_days'].
↪mean(), inplace=True)
```

[47]:
```
sns.histplot(contains_missing['Initial_days'])
```

[47]: <AxesSubplot:xlabel='Initial_days', ylabel='Count'>



[48]:
```
#replace columns in original dataframe with corrected values in␣
↪contains_missing dataframe
for x in contains_missing:
    df[x] = contains_missing[x]
```

[49]:
```
#display unique values of each column
for col in df:
    print(col + ', ', df[col].dtypes,  ': ')
    print(df[col].unique())
```

```
City,  object :
['Eva' 'Marianna' 'Sioux Falls' … 'Milmay' 'Quinn' 'Coraopolis']
State,  object :
['AL' 'FL' 'SD' 'MN' 'VA' 'OK' 'OH' 'MS' 'WI' 'IA' 'CA' 'IN' 'MO' 'MI'
 'NE' 'PA' 'AR' 'WV' 'KS' 'MA' 'KY' 'NY' 'VT' 'DC' 'IL' 'ND' 'SC' 'AK'
 'NM' 'NH' 'GA' 'NC' 'MD' 'TN' 'WA' 'TX' 'CO' 'NJ' 'LA' 'OR' 'AZ' 'ME'
 'ID' 'UT' 'RI' 'MT' 'PR' 'NV' 'CT' 'HI' 'WY' 'DE']
County,  object :
['Morgan' 'Jackson' 'Minnehaha' … 'Navarro' 'Los Alamos' 'Sterling']
Zip,  int64 :
[35621 32446 57110 …  8340 57775 15108]
Population,  int64 :
[ 2951 11303 17125 …  8368  7908 41524]
Area,  object :
['Suburban' 'Urban' 'Rural']
```

```
Timezone,  object :
['America/Chicago' 'America/New_York' 'America/Los_Angeles'
 'America/Indiana/Indianapolis' 'America/Detroit' 'America/Denver'
 'America/Nome' 'America/Anchorage' 'America/Phoenix' 'America/Boise'
 'America/Puerto_Rico' 'America/Yakutat' 'Pacific/Honolulu'
 'America/Menominee' 'America/Kentucky/Louisville'
 'America/Indiana/Vincennes' 'America/Toronto' 'America/Indiana/Marengo'
 'America/Indiana/Winamac' 'America/Indiana/Tell_City' 'America/Sitka'
 'America/Indiana/Knox' 'America/North_Dakota/New_Salem'
 'America/Indiana/Vevay' 'America/Adak' 'America/North_Dakota/Beulah']
Job,  object :
['Psychologist, sport and exercise' 'Community development worker'
 'Chief Executive Officer' 'Early years teacher'
 'Health promotion specialist' 'Corporate treasurer' 'Hydrologist'
 'Psychiatric nurse' 'Computer games developer'
 'Production assistant, radio' 'Contractor'
 'Surveyor, planning and development'
 'English as a second language teacher' 'Actuary' 'Media planner'
 'Fast food restaurant manager' 'Horticulturist, commercial'
 'Secretary, company' 'Designer, graphic' 'Personnel officer'
 'Telecommunications researcher' 'Restaurant manager, fast food'
 'Surveyor, minerals' 'Architectural technologist'
 'Therapist, speech and language' 'Accounting technician'
 'Glass blower/designer' 'Travel agency manager' 'Illustrator'
 'Police officer' 'Accountant, chartered public finance'
 'Sport and exercise psychologist' 'Pensions consultant'
 'Community education officer' 'Radio producer'
 'Designer, television/film set' 'Conference centre manager'
 'Advertising account executive' 'Civil Service fast streamer'
 'Training and development officer' 'Buyer, retail' 'Event organiser'
 'IT technical support officer'
 'Historic buildings inspector/conservation officer'
 'Research scientist (physical sciences)' 'Games developer'
 'Manufacturing engineer' 'Embryologist, clinical' 'Merchant navy officer'
 'Television floor manager' 'Web designer' 'Industrial buyer' 'Aid worker'
 'Systems developer' 'Probation officer'
 'Scientific laboratory technician' 'Environmental health practitioner'
 'Prison officer' 'Naval architect' 'Pilot, airline'
 'Medical sales representative' 'Learning disability nurse'
 'Agricultural engineer' 'Multimedia programmer' 'Cartographer'
 'Company secretary' 'Operations geologist' 'Conservation officer, nature'
 'Therapist, art' 'Therapist, sports' 'Oncologist'
 'Armed forces logistics/support/administrative officer' 'Podiatrist'
 'Translator' 'Geochemist' 'Engineer, technical sales'
 'Production designer, theatre/television/film' 'Site engineer'
 'Teacher, primary school' 'Clinical molecular geneticist'
 'Armed forces operational officer' 'Careers information officer'
 'Camera operator' 'Engineer, aeronautical' 'Learning mentor'
```

```
'Neurosurgeon' 'Clothing/textile technologist' 'Financial controller'
'Education officer, museum' 'Set designer'
'Accountant, chartered certified' 'Solicitor' 'Forensic psychologist'
'Outdoor activities/education manager' 'Heritage manager'
'Hospital doctor' 'Engineer, chemical' 'Musician'
'Engineer, control and instrumentation' 'Engineer, mining'
'Editor, commissioning' 'Sports development officer' 'Teacher, music'
"Nurse, children's" 'Editor, film/video' 'Acupuncturist' 'Data scientist'
'Tax inspector' 'Engineer, maintenance' 'Radiographer, therapeutic'
'Surveyor, commercial/residential' 'Engineer, civil (contracting)'
'Therapist, nutritional' 'Public affairs consultant' 'Warehouse manager'
'Consulting civil engineer' 'Museum/gallery exhibitions officer'
'Risk manager' 'Air traffic controller' 'Health service manager'
'Teacher, adult education' 'Theatre stage manager'
'Designer, fashion/clothing' 'Engineer, site' 'Psychologist, counselling'
'Product/process development scientist' 'Financial adviser'
'Quarry manager' 'Librarian, public' 'Presenter, broadcasting'
'Structural engineer' 'Trade mark attorney' 'Amenity horticulturist'
'Building services engineer' 'Primary school teacher' 'Network engineer'
'Psychotherapist, child' 'Archaeologist' 'Publishing rights manager'
'Economist' 'Herbalist' 'Legal secretary'
'Engineer, manufacturing systems' 'Psychologist, occupational'
'Journalist, broadcasting' 'Lexicographer' 'Clinical psychologist'
'Scientist, water quality'
'Chartered legal executive (England and Wales)' 'Statistician'
'Chartered accountant' 'Operational investment banker'
'Nutritional therapist' 'Actor' 'Ecologist' 'Conservator, furniture'
'Archivist' 'Industrial/product designer' 'Air broker' 'Sports coach'
'Chief Technology Officer' 'Arts administrator' 'Restaurant manager'
'Editorial assistant' 'Cytogeneticist' 'Scientist, marine'
'Surveyor, quantity' 'Designer, exhibition/display' 'Curator'
'Human resources officer' 'Osteopath' 'Therapist, music'
'Volunteer coordinator' 'Office manager' 'Research officer, government'
'Quality manager' 'Artist' 'Museum education officer'
'Exercise physiologist'
'Administrator, charities/voluntary organisations' 'Purchasing manager'
'Therapeutic radiographer' 'Farm manager' 'Tour manager' 'Writer'
'Designer, industrial/product' 'Science writer' 'Engineer, biomedical'
'Development worker, international aid' 'Journalist, newspaper'
'Multimedia specialist' 'Dealer' 'Water engineer'
'Scientist, clinical (histocompatibility and immunogenetics)'
'Special effects artist' 'Engineer, agricultural'
'Corporate investment banker' 'Best boy'
'Production assistant, television' 'Chiropractor' 'Jewellery designer'
'Energy engineer' 'Scientist, forensic' 'Biomedical engineer'
'Insurance account manager' 'Occupational psychologist'
'Diagnostic radiographer' 'Banker' 'Medical technical officer'
'Quantity surveyor' 'Biochemist, clinical' 'Broadcast engineer'
```

'Chartered management accountant' 'Theatre manager' 'Animal technologist'
'Animator' 'Producer, radio' 'Chiropodist' 'Exhibition designer'
'Occupational therapist' 'Database administrator'
'Arts development officer' 'Health and safety inspector'
'Press photographer' 'Recruitment consultant'
'Dance movement psychotherapist' 'Audiological scientist'
'Soil scientist' 'Equities trader' 'Orthoptist' 'Engineer, materials'
'Regulatory affairs officer' 'Trade union research officer'
'Research scientist (maths)' 'Television production assistant'
'Chief of Staff' 'Advertising copywriter'
'Programme researcher, broadcasting/film/video'
'Technical sales engineer' 'Music therapist' 'Electronics engineer'
'Waste management officer' 'Plant breeder/geneticist'
'Operational researcher' 'Further education lecturer'
'Electrical engineer' 'Television camera operator'
'Runner, broadcasting/film/video' 'Pharmacist, community'
'Ophthalmologist' 'Wellsite geologist' 'Psychologist, educational'
'Advertising account planner' 'Sports therapist'
'Surveyor, building control' 'Engineer, land' 'Clinical embryologist'
'Marine scientist' 'Teacher, secondary school' 'Chief Financial Officer'
'Landscape architect' 'Community pharmacist' 'Product manager'
'Financial risk analyst' 'Administrator' 'Civil engineer, contracting'
'Engineer, maintenance (IT)' 'Scientist, audiological'
'Management consultant' 'Dentist' 'Barrister' 'Surveyor, insurance'
'Customer service manager' 'Clinical cytogeneticist'
'Forest/woodland manager' 'Insurance underwriter'
'Speech and language therapist' 'Trading standards officer'
'Surveyor, building' 'Engineering geologist' 'Investment analyst'
'Research scientist (life sciences)' 'Firefighter'
'Higher education careers adviser' 'Theatre director'
'Passenger transport manager' 'English as a foreign language teacher'
'Research officer, trade union'
'Conservation officer, historic buildings'
'Scientist, product/process development' 'Air cabin crew'
'Colour technologist' 'Research officer, political party'
'Chemist, analytical' 'Hydrogeologist' 'Music tutor' 'Therapist, drama'
'Health physicist' 'Lecturer, higher education' 'Records manager'
'Scientist, research (medical)' 'Field trials officer'
'Adult guidance worker' 'Fine artist'
'Social research officer, government' 'Interior and spatial designer'
'Freight forwarder' 'Production engineer' 'Accommodation manager'
'Retail banker' 'Research scientist (medical)' 'Occupational hygienist'
'Diplomatic Services operational officer' "Barrister's clerk"
'Call centre manager' 'Tourism officer' 'Agricultural consultant'
'Armed forces technical officer' "Politician's assistant"
'Geographical information systems officer' 'Chief Operating Officer'
'Higher education lecturer' 'Therapist, occupational' 'Land'
'Print production planner' 'Tree surgeon' 'Physiological scientist'

```
'Producer, television/film/video' 'Facilities manager'
'Designer, blown glass/stained glass' 'Location manager'
'Maintenance engineer' 'Meteorologist' 'Local government officer'
'Energy manager' 'Estate agent' 'Counsellor' 'Dispensing optician'
'Geophysical data processor' 'Adult nurse' 'Educational psychologist'
'Mental health nurse' 'IT sales professional' 'Water quality scientist'
'Advice worker' 'Intelligence analyst' 'Community arts worker'
'Optometrist' 'Patent examiner' 'Psychotherapist, dance movement'
'Gaffer' 'Risk analyst' 'Financial trader'
'Sales promotion account executive' 'Equality and diversity officer'
'Administrator, education' 'Medical secretary'
'Claims inspector/assessor' 'Child psychotherapist' 'Immigration officer'
'Metallurgist' 'Education administrator' 'Fitness centre manager'
'Chief Strategy Officer' 'Public librarian'
'Furniture conservator/restorer' 'Photographer' 'Production manager'
'Nature conservation officer' 'Phytotherapist' 'Therapist, horticultural'
'Aeronautical engineer' 'Engineer, civil (consulting)'
'Television/film/video producer' 'Solicitor, Scotland'
'Psychologist, forensic' 'Development worker, community'
'Engineer, manufacturing' 'Garment/textile technologist'
'Charity officer' 'Insurance risk surveyor' 'Broadcast presenter'
'Secretary/administrator' 'Civil Service administrator'
'Surveyor, hydrographic' 'Loss adjuster, chartered'
'Secondary school teacher' 'Teacher, special educational needs'
'Engineer, petroleum' 'Surveyor, rural practice'
'Information systems manager' 'Designer, furniture' 'Engineer, energy'
'Conservator, museum/gallery' 'Environmental consultant'
'Doctor, general practice' 'Nurse, mental health' 'Graphic designer'
'Investment banker, corporate' 'Astronomer' 'Data processing manager'
'Stage manager' 'Textile designer' 'Drilling engineer'
'Scientist, research (life sciences)' 'Furniture designer'
'Ambulance person' 'Buyer, industrial' 'Copywriter, advertising'
'Academic librarian' 'Scientist, research (maths)'
'International aid/development worker' 'Engineer, structural'
'Lecturer, further education' 'Interpreter' 'Chief Marketing Officer'
'Transport planner' 'Pharmacist, hospital' 'Toxicologist' 'Proofreader'
'Contracting civil engineer' 'Psychologist, clinical' 'Retail manager'
'Manufacturing systems engineer' 'Art therapist'
'Chartered certified accountant' 'Sales professional, IT'
'Dramatherapist' 'Designer, interior/spatial'
'Administrator, Civil Service' 'Printmaker' 'Engineer, electrical'
'Planning and development surveyor' 'Paediatric nurse'
'Designer, multimedia' 'Herpetologist' 'Mudlogger' 'Engineer, water'
'Arboriculturist' 'Sub' 'Sports administrator' 'Mechanical engineer'
'Physicist, medical' 'Armed forces training and education officer'
'Marketing executive' 'Magazine features editor' 'Ergonomist'
'Mining engineer' 'Dancer' 'Optician, dispensing' 'Designer, textile'
'Ranger/warden' 'Psychiatrist' 'Bonds trader' 'Technical brewer'
```

'Engineer, building services' 'Field seismologist'
'Engineer, electronics' 'Medical illustrator' 'Architect'
'Engineer, production' 'Licensed conveyancer' 'Surveyor, mining'
'Applications developer' 'Museum/gallery curator' 'Market researcher'
'Radiation protection practitioner'
'Control and instrumentation engineer' 'Programmer, applications'
'Advertising art director'
'Clinical scientist, histocompatibility and immunogenetics'
'Professor Emeritus' 'Horticulturist, amenity' 'Physiotherapist'
'Race relations officer' 'Surveyor, land/geomatics' 'Youth worker'
'Horticultural therapist' 'IT consultant' 'Make'
'Public relations account executive' 'Private music teacher'
'Fashion designer' 'Hospital pharmacist' 'Tax adviser'
'Engineer, broadcasting (operations)' 'Commercial art gallery manager'
'Legal executive' 'Visual merchandiser' 'Commercial/residential surveyor'
'Personal assistant' 'Insurance claims handler' 'Financial manager'
'Tourist information centre manager' 'Scientist, physiological'
'Designer, ceramics/pottery' 'Accountant, chartered management'
'Psychotherapist' 'Health visitor' 'Pharmacologist'
'Special educational needs teacher' 'Public relations officer'
'Town planner' 'Animal nutritionist' 'Building control surveyor'
'Engineer, automotive' 'Information officer'
'Senior tax professional/tax inspector' 'Film/video editor' 'Cabin crew'
'Radiographer, diagnostic' 'Warden/ranger' 'Video editor' 'Airline pilot'
'Newspaper journalist' 'Education officer, community'
'Geologist, engineering' 'Librarian, academic' 'Paramedic'
'Recycling officer' 'Merchandiser, retail' 'Retail merchandiser'
'Administrator, local government' 'Counselling psychologist'
'Estate manager/land agent' 'Oceanographer' 'Haematologist'
'Scientist, research (physical sciences)' 'Medical physicist'
'Communications engineer' 'Surgeon' 'Homeopath' 'Charity fundraiser'
'Theme park manager' 'Barista' 'Chartered public finance accountant'
'Teaching laboratory technician' 'Microbiologist'
'Programmer, multimedia' 'Automotive engineer' 'Holiday representative'
'Systems analyst' 'Product designer' 'Forensic scientist'
'Museum/gallery conservator' 'Patent attorney' 'Ship broker'
'Technical author' 'Pension scheme manager' 'Ceramics designer'
'Careers adviser' 'Building surveyor' 'Public house manager'
'Environmental education officer' 'Journalist, magazine'
'Magazine journalist' 'Analytical chemist'
'Teacher, English as a foreign language'
'Lighting technician, broadcasting/film/video' 'Teacher, early years/pre'
'Commercial horticulturist' 'Publishing copy' 'Clinical biochemist'
'IT trainer' 'Programmer, systems' 'Logistics and distribution manager'
'Horticultural consultant' 'Hotel manager' 'Associate Professor'
'Nurse, learning disability' 'Hydrographic surveyor' 'Nurse, adult'
'Fisheries officer' 'Administrator, sports' 'Insurance broker'
'Veterinary surgeon' 'Designer, jewellery' 'Lobbyist' 'Chemical engineer'

```
 'Chartered loss adjuster' 'Social researcher' 'Petroleum engineer'
 'Social worker' 'Education officer, environmental' 'Futures trader'
 'Fish farm manager' 'Lawyer' 'Seismic interpreter' 'TEFL teacher'
 'Immunologist' 'Engineer, drilling'
 'Emergency planning/management officer' 'Pathologist'
 'Broadcast journalist' 'Geologist, wellsite'
 'Investment banker, operational' 'Biomedical scientist' 'Bookseller'
 'Copy' 'Midwife' 'Media buyer' 'Geneticist, molecular'
 'Housing manager/officer' 'Geophysicist/field seismologist'
 'Art gallery manager' 'Food technologist' 'Land/geomatics surveyor'
 'Radio broadcast assistant' 'Psychologist, prison and probation services'
 'Dietitian' 'Civil engineer, consulting' 'Sales executive'
 'Leisure centre manager' 'Scientist, biomedical'
 'Exhibitions officer, museum/gallery' 'Engineer, communications'
 'Catering manager' 'Administrator, arts' 'Software engineer'
 'Medical laboratory scientific officer' 'Commissioning editor'
 'Geoscientist' 'Materials engineer' 'Financial planner'
 'Brewing technologist' 'Minerals surveyor' 'Editor, magazine features'
 'General practice doctor' 'Health and safety adviser' 'Doctor, hospital'
 'Environmental manager' 'Clinical research associate'
 'Sound technician, broadcasting/film/video' 'Press sub' 'Retail buyer'
 'Comptroller' 'Government social research officer'
 'Rural practice surveyor' 'Accountant, chartered']
Children,  float64 :
[ 1.  3.  0.  7.  2.  4. 10.  5.  6.  9.  8.]
Age,  int32 :
[53 51 78 22 76 50 40 48 55 64 41 45 85 44 54 72 84 68 52 31 60 75 70 63
 56 32 86 65 66 67 79 25 58 59 33 83 73 43 57 36 49 39 20 69 26 47 18 38
 82 34 74 37 77 27 89 30 87 23 29 80 19 24 88 62 46 71 21 61 81 42 35 28]
Education,  object :
['Some College, Less than 1 Year'
 'Some College, 1 or More Years, No Degree'
 'GED or Alternative Credential' 'Regular High School Diploma'
 "Bachelor's Degree" "Master's Degree" 'Nursery School to 8th Grade'
 '9th Grade to 12th Grade, No Diploma' 'Doctorate Degree'
 "Associate's Degree" 'Professional School Degree'
 'No Schooling Completed']
Employment,  object :
['Full Time' 'Retired' 'Unemployed' 'Student' 'Part Time']
Income,  float64 :
[86575.93 46805.99 14370.14 … 65917.81 29702.32 62682.63]
Mariage_status,  object :
['Divorced' 'Married' 'Widowed' 'Never Married' 'Separated']
Gender,  object :
['Male' 'Female' 'Prefer not to answer']
Readmited,  object :
['No' 'Yes']
VitD_levels,  float64 :
```

```
[17.80233049 18.99463952 17.4158887  … 15.75275136 21.95630508
 20.42188348]
Doc_visits,  int64 :
[6 4 5 7 3 2 8 9 1]
Full_meals_eaten,  int64 :
[0 2 1 3 4 5 7 6]
VitD_supplements,  int64 :
[0 1 2 3 4 5]
Habitual_soft_drink_use,  object :
['No' 'Yes']
Initial_admin,  object :
['Emergency Admission' 'Elective Admission' 'Observation Admission']
High_blood_pressure,  object :
['Yes' 'No']
Stroke,  object :
['No' 'Yes']
Complication_risk,  object :
['Medium' 'High' 'Low']
Overweight,  float64 :
[0. 1.]
Arthritis,  object :
['Yes' 'No']
Diabetes,  object :
['Yes' 'No']
Hyperlipidemia,  object :
['No' 'Yes']
Back_pain,  object :
['Yes' 'No']
Anxiety,  float64 :
[1. 0.]
Allergic_rhinitis,  object :
['Yes' 'No']
Reflux_esophagitis,  object :
['No' 'Yes']
Asthma,  object :
['Yes' 'No']
Primary_service_recived,  object :
['Blood Work' 'Intravenous' 'CT Scan' 'MRI']
Initial_days,  float64 :
[10.58576971 15.12956221  4.77217721 … 66.01257016 63.35690285
 70.85059182]
Total_charge,  float64 :
[3191.048774 4214.905346 2177.586768 … 7725.953391 8462.831883
 8700.856021]
Additional_charges,  float64 :
[17939.40342  17612.99812  17505.19246  … 15281.21466   7781.678412
 11643.18993 ]
Survey_timely_addmission,  int64 :
```

```
[3 2 4 1 5 7 6 8]
Survey_timely_treatment,  int64 :
[3 4 5 1 2 6 7]
Survey_timely_visits,  int64 :
[2 3 4 5 1 6 7 8]
Survey_reliability,  int64 :
[2 4 3 5 6 1 7]
Survey_options,  int64 :
[4 3 5 2 6 1 7]
Survey_hours,  int64 :
[3 4 5 2 6 1 7]
Survey_courtesy,  int64 :
[3 5 4 2 6 1 7]
Survey_active_listening,  int64 :
[4 3 5 6 2 1 7]
```

[50]:
```
#standardize time zones to utc, if a time zone does not observe daylight␣
↪savings time it is appended with (ND)
timezone_dict = {'America/Chicago': 'UTC-6:00', 'America/New_York': 'UTC-5:00',␣
↪'America/Los_Angeles': 'UTC-8:00',
 'America/Indiana/Indianapolis': 'UTC-5:00', 'America/Detroit': 'UTC-5:00',␣
↪'America/Denver': 'UTC-7:00',
 'America/Nome': 'UTC-9:00',  'America/Anchorage': 'UTC-9:00',  'America/
↪Phoenix': 'UTC-8:00(ND)',
 'America/Boise': 'UTC-8:00', 'America/Puerto_Rico': 'UTC-4:00(ND)',  'America/
↪Yakutat': 'UTC-9:00',
 'Pacific/Honolulu': 'UTC-10:00(ND)', 'America/Menominee': 'UTC-6:00', 'America/
↪Kentucky/Louisville': 'UTC-5:00',
 'America/Indiana/Vincennes': 'UTC-5:00', 'America/Toronto': 'UTC-5:00',␣
↪'America/Indiana/Marengo': 'UTC-5:00',
 'America/Indiana/Winamac': 'UTC-5:00', 'America/Indiana/Tell_City': 'UTC-6:
↪00', 'America/Sitka': 'UTC-9:00',
 'America/Indiana/Knox': 'UTC-6:00', 'America/North_Dakota/New_Salem': 'UTC-6:
↪00', 'America/Indiana/Vevay': 'UTC-5:00',
 'America/Adak': 'UTC-10:00', 'America/North_Dakota/Beulah': 'UTC-6:00'}
df['Timezone'].replace(timezone_dict, inplace = True)
df['Timezone']
```

[50]:
```
Case_order
1        UTC-6:00
2        UTC-6:00
3        UTC-6:00
4        UTC-6:00
5        UTC-5:00
            …
9996     UTC-5:00
9997     UTC-5:00
```

```
9998      UTC-6:00
9999      UTC-7:00
10000     UTC-5:00
Name: Timezone, Length: 10000, dtype: object
```

[51]:
```
#Convert columns that that express whole number values that are currently float␣
↪to int
df.loc[:,['Children', 'Overweight', 'Anxiety']] = df[['Children', 'Overweight',␣
↪'Anxiety']].astype(int)
```

[52]:
```
#convert zip colum to string type, identify records with invalid zip codes
df['Zip'] = df['Zip'].astype(str)
```

[53]:
```
invalid_zips = df['Zip'].apply(len) != 5
invalid_list = df.loc[invalid_zips, ['Zip', 'City', 'State']]
invalid_list
```

[53]:
```
             Zip                 City State
Case_order
32          2584            Nantucket    MA
36          5043        East Thetford    VT
37          2468                Waban    MA
38          2138            Cambridge    MA
68          3464             Stoddard    NH
…            …                  …    …
9976        4415  Brownville Junction    ME
9977        6084              Tolland    CT
9983        8401        Atlantic City    NJ
9994        7647           Northvale    NJ
9997        8340              Milmay    NJ

[723 rows x 3 columns]
```

[54]:
```
#list all invalid zip codes, cities, and states in list
for i in range(0, invalid_list.shape[0]):
    print(invalid_list.iloc[i])
#while manualy cross referencing this data against a
#United states zip code database(https://www.zipdatamaps.com/index.php),
#it became apparent that the invalid zip codes where caused by leading 0's␣
↪being ommited
```

```
Zip           2584
City     Nantucket
State           MA
Name: 32, dtype: object
Zip               5043
City     East Thetford
State               VT
```

```
Name: 36, dtype: object
Zip       2468
City     Waban
State       MA
Name: 37, dtype: object
Zip           2138
City     Cambridge
State           MA
Name: 38, dtype: object
Zip          3464
City     Stoddard
State          NH
Name: 68, dtype: object
Zip           8332
City     Millville
State           NJ
Name: 109, dtype: object
Zip               7935
City     Green Village
State               NJ
Name: 114, dtype: object
Zip            7882
City     Washington
State            NJ
Name: 120, dtype: object
Zip          3462
City     Spofford
State          NH
Name: 145, dtype: object
Zip        4408
City     Aurora
State        ME
Name: 149, dtype: object
Zip                  4940
City     Farmington Falls
State                  ME
Name: 172, dtype: object
Zip         2889
City     Warwick
State         RI
Name: 174, dtype: object
Zip          3885
City     Stratham
State          NH
Name: 190, dtype: object
Zip           2835
City     Jamestown
State           RI
```

```
Name: 195, dtype: object
Zip         3220
City     Belmont
State         NH
Name: 198, dtype: object
Zip             4344
City     Farmingdale
State             ME
Name: 203, dtype: object
Zip        669
City     Lares
State       PR
Name: 226, dtype: object
Zip         7030
City     Hoboken
State         NJ
Name: 248, dtype: object
Zip        4449
City     Hudson
State        ME
Name: 309, dtype: object
Zip         7630
City     Emerson
State         NJ
Name: 310, dtype: object
Zip               6119
City     West Hartford
State               CT
Name: 311, dtype: object
Zip         3446
City     Swanzey
State         NH
Name: 313, dtype: object
Zip               4926
City     China Village
State               ME
Name: 338, dtype: object
Zip             4344
City     Farmingdale
State             ME
Name: 341, dtype: object
Zip          2364
City     Kingston
State          MA
Name: 345, dtype: object
Zip         6401
City     Ansonia
State         CT
```

```
Name: 368, dtype: object
Zip            8876
City     Somerville
State            NJ
Name: 388, dtype: object
Zip        6263
City     Rogers
State        CT
Name: 393, dtype: object
Zip        4626
City     Cutler
State        ME
Name: 395, dtype: object
Zip            7028
City     Glen Ridge
State            NJ
Name: 416, dtype: object
Zip           6498
City     Westbrook
State           CT
Name: 417, dtype: object
Zip          6264
City     Scotland
State          CT
Name: 457, dtype: object
Zip      8004
City     Atco
State      NJ
Name: 474, dtype: object
Zip          2838
City     Manville
State          RI
Name: 486, dtype: object
Zip      4530
City     Bath
State      ME
Name: 512, dtype: object
Zip           1832
City     Haverhill
State           MA
Name: 513, dtype: object
Zip                   7716
City     Atlantic Highlands
State                   NJ
Name: 522, dtype: object
Zip           7460
City     Stockholm
State           NJ
```

```
Name: 524, dtype: object
Zip          1940
City     Lynnfield
State           MA
Name: 530, dtype: object
Zip        7055
City     Passaic
State         NJ
Name: 532, dtype: object
Zip       4988
City     Unity
State       ME
Name: 533, dtype: object
Zip        1562
City     Spencer
State         MA
Name: 534, dtype: object
Zip            6375
City     Quaker Hill
State             CT
Name: 542, dtype: object
Zip                 3227
City     Center Sandwich
State                 NH
Name: 546, dtype: object
Zip            7311
City     Jersey City
State             NJ
Name: 552, dtype: object
Zip            2745
City     New Bedford
State             MA
Name: 561, dtype: object
Zip        7857
City     Netcong
State         NJ
Name: 565, dtype: object
Zip          8098
City     Woodstown
State           NJ
Name: 574, dtype: object
Zip          2072
City     Stoughton
State           MA
Name: 588, dtype: object
Zip                3830
City     East Wakefield
State                NH
```

```
Name: 591, dtype: object
Zip         3864
City     Ossipee
State         NH
Name: 593, dtype: object
Zip         8858
City     Oldwick
State         NJ
Name: 609, dtype: object
Zip               7935
City     Green Village
State               NJ
Name: 615, dtype: object
Zip      4847
City     Hope
State      ME
Name: 641, dtype: object
Zip             6033
City     Glastonbury
State             CT
Name: 655, dtype: object
Zip        2341
City     Hanson
State        MA
Name: 665, dtype: object
Zip           7410
City     Fair Lawn
State           NJ
Name: 670, dtype: object
Zip          5660
City     Moretown
State          VT
Name: 707, dtype: object
Zip           6281
City     Woodstock
State           CT
Name: 708, dtype: object
Zip           8023
City     Deepwater
State           NJ
Name: 720, dtype: object
Zip           6498
City     Westbrook
State           CT
Name: 745, dtype: object
Zip           4347
City     Hallowell
State           ME
```

```
Name: 755, dtype: object
Zip      3870
City      Rye
State      NH
Name: 785, dtype: object
Zip         3745
City     Cornish
State         NH
Name: 802, dtype: object
Zip         8201
City     Absecon
State         NJ
Name: 824, dtype: object
Zip                 5252
City     East Arlington
State                 VT
Name: 850, dtype: object
Zip        4765
City     Patten
State        ME
Name: 878, dtype: object
Zip                  4669
City     Prospect Harbor
State                  ME
Name: 881, dtype: object
Zip       3079
City     Salem
State       NH
Name: 907, dtype: object
Zip         7014
City     Clifton
State         NJ
Name: 910, dtype: object
Zip            694
City     Vega Baja
State           PR
Name: 911, dtype: object
Zip        5149
City     Ludlow
State        VT
Name: 912, dtype: object
Zip          8270
City     Woodbine
State          NJ
Name: 929, dtype: object
Zip        5821
City     Barnet
State        VT
```

```
Name: 937, dtype: object
Zip          751
City     Salinas
State         PR
Name: 945, dtype: object
Zip         4928
City     Corinna
State         ME
Name: 948, dtype: object
Zip         8317
City     Dorothy
State         NJ
Name: 984, dtype: object
Zip         2452
City     Waltham
State         MA
Name: 994, dtype: object
Zip            1050
City     Huntington
State            MA
Name: 1021, dtype: object
Zip            8741
City     Pine Beach
State            NJ
Name: 1032, dtype: object
Zip        1908
City     Nahant
State        MA
Name: 1045, dtype: object
Zip            2762
City     Plainville
State            MA
Name: 1055, dtype: object
Zip          4964
City     Oquossoc
State          ME
Name: 1104, dtype: object
Zip                  3225
City     Center Barnstead
State                  NH
Name: 1108, dtype: object
Zip             7740
City     Long Branch
State             NJ
Name: 1115, dtype: object
Zip         1036
City     Hampden
State         MA
```

```
Name: 1126, dtype: object
Zip          4970
City     Rangeley
State          ME
Name: 1129, dtype: object
Zip           1085
City     Westfield
State           MA
Name: 1171, dtype: object
Zip           5866
City     Sheffield
State           VT
Name: 1172, dtype: object
Zip            3257
City     New London
State            NH
Name: 1189, dtype: object
Zip           4974
City     Searsport
State           ME
Name: 1194, dtype: object
Zip         4573
City     Walpole
State         ME
Name: 1218, dtype: object
Zip        8217
City     Elwood
State        NJ
Name: 1225, dtype: object
Zip         5089
City     Windsor
State         VT
Name: 1254, dtype: object
Zip                4265
City     North Monmouth
State                ME
Name: 1264, dtype: object
Zip               678
City     Quebradillas
State              PR
Name: 1265, dtype: object
Zip            6320
City     New London
State            CT
Name: 1326, dtype: object
Zip              5679
City     Williamstown
State              VT
```

```
Name: 1328, dtype: object
Zip              617
City     Barceloneta
State             PR
Name: 1340, dtype: object
Zip               8232
City     Pleasantville
State               NJ
Name: 1349, dtype: object
Zip              4091
City     West Baldwin
State              ME
Name: 1375, dtype: object
Zip           631
City     Castaner
State          PR
Name: 1379, dtype: object
Zip         5062
City     Reading
State         VT
Name: 1399, dtype: object
Zip           5866
City     Sheffield
State           VT
Name: 1414, dtype: object
Zip           8511
City     Cookstown
State           NJ
Name: 1421, dtype: object
Zip           3054
City     Merrimack
State           NH
Name: 1439, dtype: object
Zip           5472
City     New Haven
State           VT
Name: 1450, dtype: object
Zip          957
City     Bayamon
State         PR
Name: 1487, dtype: object
Zip             2129
City     Charlestown
State             MA
Name: 1517, dtype: object
Zip            8755
City     Toms River
State            NJ
```

```
Name: 1519, dtype: object
Zip             6419
City     Killingworth
State              CT
Name: 1524, dtype: object
Zip         3846
City     Jackson
State         NH
Name: 1537, dtype: object
Zip                3844
City     Hampton Falls
State               NH
Name: 1551, dtype: object
Zip           2584
City     Nantucket
State           MA
Name: 1561, dtype: object
Zip             1118
City     Springfield
State             MA
Name: 1599, dtype: object
Zip           8323
City     Greenwich
State           NJ
Name: 1653, dtype: object
Zip            6785
City     South Kent
State            CT
Name: 1684, dtype: object
Zip          6880
City     Westport
State          CT
Name: 1685, dtype: object
Zip                  1373
City     South Deerfield
State                 MA
Name: 1693, dtype: object
Zip            1506
City     Brookfield
State            MA
Name: 1695, dtype: object
Zip      3861
City      Lee
State      NH
Name: 1699, dtype: object
Zip                   4650
City     Little Deer Isle
State                   ME
```

```
Name: 1702, dtype: object
Zip                1370
City     Shelburne Falls
State                MA
Name: 1733, dtype: object
Zip           3269
City     Sanbornton
State           NH
Name: 1735, dtype: object
Zip          7642
City     Hillsdale
State          NJ
Name: 1736, dtype: object
Zip            4219
City     Bryant Pond
State            ME
Name: 1771, dtype: object
Zip            7756
City     Ocean Grove
State            NJ
Name: 1781, dtype: object
Zip           4489
City     Stillwater
State           ME
Name: 1792, dtype: object
Zip          2445
City     Brookline
State          MA
Name: 1801, dtype: object
Zip           7661
City     River Edge
State           NJ
Name: 1827, dtype: object
Zip      3588
City     Milan
State      NH
Name: 1829, dtype: object
Zip          7757
City     Oceanport
State          NJ
Name: 1876, dtype: object
Zip          2140
City     Cambridge
State          MA
Name: 1890, dtype: object
Zip         6902
City     Stamford
State         CT
```

```
Name: 1898, dtype: object
Zip            1945
City     Marblehead
State            MA
Name: 1908, dtype: object
Zip                976
City     Trujillo Alto
State               PR
Name: 1914, dtype: object
Zip             8829
City     High Bridge
State             NJ
Name: 1952, dtype: object
Zip                   2852
City     North Kingstown
State                   RI
Name: 1971, dtype: object
Zip          7755
City     Oakhurst
State          NJ
Name: 1974, dtype: object
Zip           983
City     Carolina
State          PR
Name: 1975, dtype: object
Zip        5820
City     Albany
State        VT
Name: 1976, dtype: object
Zip         4950
City     Madison
State         ME
Name: 1985, dtype: object
Zip              3057
City     Mont Vernon
State             NH
Name: 2003, dtype: object
Zip             656
City     Guayanilla
State            PR
Name: 2014, dtype: object
Zip                6092
City     West Simsbury
State               CT
Name: 2028, dtype: object
Zip               3872
City     Sanbornville
State              NH
```

```
Name: 2041, dtype: object
Zip        3752
City     Goshen
State        NH
Name: 2043, dtype: object
Zip            5663
City     Northfield
State            VT
Name: 2046, dtype: object
Zip             656
City     Guayanilla
State            PR
Name: 2064, dtype: object
Zip         6854
City     Norwalk
State         CT
Name: 2081, dtype: object
Zip         5748
City     Hancock
State         VT
Name: 2089, dtype: object
Zip        7036
City     Linden
State        NJ
Name: 2099, dtype: object
Zip             8060
City     Mount Holly
State             NJ
Name: 2112, dtype: object
Zip           3868
City     Rochester
State           NH
Name: 2144, dtype: object
Zip       7066
City     Clark
State       NJ
Name: 2147, dtype: object
Zip        5046
City     Groton
State        VT
Name: 2163, dtype: object
Zip         1378
City     Warwick
State         MA
Name: 2171, dtype: object
Zip                4030
City     East Waterboro
State                ME
```

```
Name: 2176, dtype: object
Zip                 2673
City     West Yarmouth
State               MA
Name: 2180, dtype: object
Zip       1256
City     Savoy
State       MA
Name: 2188, dtype: object
Zip         3223
City     Campton
State         NH
Name: 2197, dtype: object
Zip           4750
City     Limestone
State           ME
Name: 2229, dtype: object
Zip        730
City     Ponce
State       PR
Name: 2250, dtype: object
Zip                7480
City     West Milford
State              NJ
Name: 2270, dtype: object
Zip        3440
City     Antrim
State        NH
Name: 2273, dtype: object
Zip           3576
City     Colebrook
State           NH
Name: 2277, dtype: object
Zip           1606
City     Worcester
State           MA
Name: 2282, dtype: object
Zip                  3826
City     East Hampstead
State                NH
Name: 2285, dtype: object
Zip        2053
City     Medway
State        MA
Name: 2301, dtype: object
Zip        1033
City     Granby
State        MA
```

```
Name: 2315, dtype: object
Zip            3258
City     Chichester
State            NH
Name: 2368, dtype: object
Zip       1343
City     Drury
State       MA
Name: 2382, dtype: object
Zip            1824
City     Chelmsford
State            MA
Name: 2393, dtype: object
Zip              2538
City     East Wareham
State              MA
Name: 2396, dtype: object
Zip               4042
City     Hollis Center
State               ME
Name: 2399, dtype: object
Zip        2203
City     Boston
State        MA
Name: 2402, dtype: object
Zip         2571
City     Wareham
State         MA
Name: 2404, dtype: object
Zip           1876
City     Tewksbury
State           MA
Name: 2428, dtype: object
Zip               4003
City     Bailey Island
State               ME
Name: 2449, dtype: object
Zip           1086
City     Westfield
State           MA
Name: 2489, dtype: object
Zip        1740
City     Bolton
State        MA
Name: 2490, dtype: object
Zip        6469
City     Moodus
State        CT
```

```
Name: 2502, dtype: object
Zip              4673
City     Sargentville
State              ME
Name: 2519, dtype: object
Zip           5866
City     Sheffield
State           VT
Name: 2553, dtype: object
Zip        1612
City     Paxton
State        MA
Name: 2557, dtype: object
Zip            2122
City     Dorchester
State            MA
Name: 2582, dtype: object
Zip         1510
City     Clinton
State         MA
Name: 2590, dtype: object
Zip            2912
City     Providence
State            RI
Name: 2591, dtype: object
Zip            5361
City     Whitingham
State            VT
Name: 2619, dtype: object
Zip          3106
City     Hooksett
State          NH
Name: 2628, dtype: object
Zip           7450
City     Ridgewood
State           NJ
Name: 2632, dtype: object
Zip            5658
City     Marshfield
State            VT
Name: 2661, dtype: object
Zip           7450
City     Ridgewood
State           NJ
Name: 2667, dtype: object
Zip             1118
City     Springfield
State             MA
```

```
Name: 2671, dtype: object
Zip           4614
City     Blue Hill
State           ME
Name: 2686, dtype: object
Zip               1088
City     West Hatfield
State               MA
Name: 2692, dtype: object
Zip       1970
City     Salem
State       MA
Name: 2693, dtype: object
Zip           7843
City     Hopatcong
State           NJ
Name: 2698, dtype: object
Zip         2859
City     Pascoag
State         RI
Name: 2719, dtype: object
Zip              3746
City     Cornish Flat
State              NH
Name: 2736, dtype: object
Zip          2364
City     Kingston
State          MA
Name: 2771, dtype: object
Zip         1754
City     Maynard
State         MA
Name: 2776, dtype: object
Zip        8217
City     Elwood
State        NJ
Name: 2794, dtype: object
Zip        2151
City     Revere
State        MA
Name: 2821, dtype: object
Zip                 6332
City     Central Village
State                 CT
Name: 2832, dtype: object
Zip               3751
City     Georges Mills
State               NH
```

```
Name: 2842, dtype: object
Zip          2631
City     Brewster
State          MA
Name: 2846, dtype: object
Zip               8904
City     Highland Park
State               NJ
Name: 2849, dtype: object
Zip           1355
City     New Salem
State           MA
Name: 2851, dtype: object
Zip                 7640
City     Harrington Park
State                 NJ
Name: 2852, dtype: object
Zip        1984
City     Wenham
State        MA
Name: 2866, dtype: object
Zip                   4637
City     Grand Lake Stream
State                   ME
Name: 2870, dtype: object
Zip         1867
City     Reading
State         MA
Name: 2910, dtype: object
Zip         7928
City     Chatham
State         NJ
Name: 2922, dtype: object
Zip         5765
City     Proctor
State         VT
Name: 2948, dtype: object
Zip        4002
City     Alfred
State        ME
Name: 2952, dtype: object
Zip         4539
City     Bristol
State         ME
Name: 2967, dtype: object
Zip             1550
City     Southbridge
State             MA
```

```
Name: 2971, dtype: object
Zip          1960
City      Peabody
State          MA
Name: 2987, dtype: object
Zip                 3826
City      East Hampstead
State                 NH
Name: 3000, dtype: object
Zip            4092
City      Westbrook
State            ME
Name: 3023, dtype: object
Zip            4750
City      Limestone
State            ME
Name: 3024, dtype: object
Zip           2370
City      Rockland
State           MA
Name: 3027, dtype: object
Zip             2564
City      Siasconset
State             MA
Name: 3035, dtype: object
Zip             3854
City      New Castle
State             NH
Name: 3042, dtype: object
Zip          6855
City      Norwalk
State          CT
Name: 3056, dtype: object
Zip        8318
City      Elmer
State        NJ
Name: 3074, dtype: object
Zip               4108
City      Peaks Island
State               ME
Name: 3081, dtype: object
Zip               6424
City      East Hampton
State               CT
Name: 3088, dtype: object
Zip             3442
City      Bennington
State             NH
```

```
Name: 3101, dtype: object
Zip              3827
City     East Kingston
State               NH
Name: 3103, dtype: object
Zip           924
City     San Juan
State          PR
Name: 3116, dtype: object
Zip       2666
City     Truro
State       MA
Name: 3138, dtype: object
Zip          5845
City     Irasburg
State          VT
Name: 3154, dtype: object
Zip             6016
City     Broad Brook
State             CT
Name: 3155, dtype: object
Zip           2814
City     Chepachet
State           RI
Name: 3194, dtype: object
Zip                  2461
City     Newton Highlands
State                  MA
Name: 3215, dtype: object
Zip        769
City     Coamo
State       PR
Name: 3237, dtype: object
Zip           4646
City     Islesford
State           ME
Name: 3245, dtype: object
Zip           4674
City     Seal Cove
State           ME
Name: 3254, dtype: object
Zip        1083
City     Warren
State        MA
Name: 3263, dtype: object
Zip           7208
City     Elizabeth
State           NJ
```

```
Name: 3270, dtype: object
Zip          4943
City     Hartland
State          ME
Name: 3286, dtype: object
Zip              3260
City     North Sutton
State              NH
Name: 3288, dtype: object
Zip        5867
City     Sutton
State        VT
Name: 3307, dtype: object
Zip        6883
City     Weston
State        CT
Name: 3331, dtype: object
Zip                   4637
City     Grand Lake Stream
State                   ME
Name: 3333, dtype: object
Zip          8559
City     Stockton
State          NJ
Name: 3336, dtype: object
Zip           5455
City     Fairfield
State           VT
Name: 3346, dtype: object
Zip             7304
City     Jersey City
State             NJ
Name: 3349, dtype: object
Zip                    8889
City     Whitehouse Station
State                    NJ
Name: 3356, dtype: object
Zip       5672
City     Stowe
State       VT
Name: 3362, dtype: object
Zip          3885
City     Stratham
State          NH
Name: 3371, dtype: object
Zip          7675
City     Westwood
State          NJ
```

```
Name: 3406, dtype: object
Zip          1543
City     Rutland
State         MA
Name: 3421, dtype: object
Zip        2030
City     Dover
State      MA
Name: 3422, dtype: object
Zip         7880
City     Vienna
State        NJ
Name: 3451, dtype: object
Zip           1330
City     Ashfield
State          MA
Name: 3456, dtype: object
Zip         6019
City     Canton
State        CT
Name: 3472, dtype: object
Zip         5065
City     Sharon
State        VT
Name: 3473, dtype: object
Zip           1860
City     Merrimac
State          MA
Name: 3491, dtype: object
Zip           3779
City     Piermont
State          NH
Name: 3494, dtype: object
Zip                3226
City     Center Harbor
State               NH
Name: 3497, dtype: object
Zip                5043
City     East Thetford
State               VT
Name: 3525, dtype: object
Zip            6830
City     Greenwich
State           CT
Name: 3549, dtype: object
Zip                 7444
City     Pompton Plains
State                NJ
```

```
Name: 3566, dtype: object
Zip          2364
City     Kingston
State          MA
Name: 3600, dtype: object
Zip               8887
City     Three Bridges
State               NJ
Name: 3627, dtype: object
Zip               7758
City     Port Monmouth
State               NJ
Name: 3628, dtype: object
Zip           8078
City     Runnemede
State           NJ
Name: 3631, dtype: object
Zip           4756
City     Madawaska
State           ME
Name: 3649, dtype: object
Zip        6514
City     Hamden
State        CT
Name: 3657, dtype: object
Zip          3865
City     Plaistow
State          NH
Name: 3676, dtype: object
Zip         4444
City     Hampden
State         ME
Name: 3698, dtype: object
Zip           7201
City     Elizabeth
State           NJ
Name: 3716, dtype: object
Zip                   8880
City     South Bound Brook
State                   NJ
Name: 3735, dtype: object
Zip            3245
City     Holderness
State            NH
Name: 3746, dtype: object
Zip              3467
City     Westmoreland
State              NH
```

```
Name: 3754, dtype: object
Zip          8721
City     Bayville
State          NJ
Name: 3765, dtype: object
Zip        1590
City     Sutton
State        MA
Name: 3769, dtype: object
Zip           1420
City     Fitchburg
State           MA
Name: 3796, dtype: object
Zip        4051
City     Lovell
State        ME
Name: 3798, dtype: object
Zip              4849
City     Lincolnville
State              ME
Name: 3802, dtype: object
Zip               6812
City     New Fairfield
State               CT
Name: 3805, dtype: object
Zip          961
City     Bayamon
State         PR
Name: 3821, dtype: object
Zip           773
City     Luquillo
State          PR
Name: 3893, dtype: object
Zip           2184
City     Braintree
State           MA
Name: 3950, dtype: object
Zip        4843
City     Camden
State        ME
Name: 3960, dtype: object
Zip          7501
City     Paterson
State          NJ
Name: 3980, dtype: object
Zip           927
City     San Juan
State          PR
```

```
Name: 3981, dtype: object
Zip         777
City     Juncos
State        PR
Name: 3997, dtype: object
Zip          6234
City     Brooklyn
State          CT
Name: 4021, dtype: object
Zip      1432
City     Ayer
State      MA
Name: 4024, dtype: object
Zip        7106
City     Newark
State        NJ
Name: 4036, dtype: object
Zip         727
City     Caguas
State        PR
Name: 4075, dtype: object
Zip                      8852
City     Monmouth Junction
State                      NJ
Name: 4076, dtype: object
Zip                      1084
City     West Chesterfield
State                      MA
Name: 4089, dtype: object
Zip           1834
City     Groveland
State           MA
Name: 4094, dtype: object
Zip           6353
City     Montville
State           CT
Name: 4119, dtype: object
Zip            8872
City     Sayreville
State            NJ
Name: 4128, dtype: object
Zip            4063
City     Ocean Park
State            ME
Name: 4131, dtype: object
Zip          782
City     Comerio
State         PR
```

```
Name: 4145, dtype: object
Zip           6702
City     Waterbury
State           CT
Name: 4155, dtype: object
Zip          2561
City     Sagamore
State          MA
Name: 4162, dtype: object
Zip            8092
City     West Creek
State            NJ
Name: 4165, dtype: object
Zip           3045
City     Goffstown
State           NH
Name: 4177, dtype: object
Zip             3603
City     Charlestown
State             NH
Name: 4214, dtype: object
Zip            3269
City     Sanbornton
State            NH
Name: 4228, dtype: object
Zip        698
City     Yauco
State       PR
Name: 4229, dtype: object
Zip           1008
City     Blandford
State           MA
Name: 4239, dtype: object
Zip          2052
City     Medfield
State          MA
Name: 4249, dtype: object
Zip                  7080
City     South Plainfield
State                  NJ
Name: 4250, dtype: object
Zip                8640
City     Joint Base Mdl
State              NJ
Name: 4285, dtype: object
Zip            6279
City     Willington
State            CT
```

```
Name: 4337, dtype: object
Zip                4544
City     East Boothbay
State               ME
Name: 4350, dtype: object
Zip         1341
City     Conway
State        MA
Name: 4362, dtype: object
Zip          7011
City     Clifton
State         NJ
Name: 4366, dtype: object
Zip           950
City     Toa Baja
State          PR
Name: 4395, dtype: object
Zip            3245
City     Holderness
State            NH
Name: 4400, dtype: object
Zip            8042
City     Juliustown
State            NJ
Name: 4403, dtype: object
Zip          5454
City     Fairfax
State         VT
Name: 4406, dtype: object
Zip           7803
City     Mine Hill
State           NJ
Name: 4408, dtype: object
Zip              7936
City     East Hanover
State              NJ
Name: 4410, dtype: object
Zip           7041
City     Millburn
State          NJ
Name: 4436, dtype: object
Zip            4635
City     Frenchboro
State            ME
Name: 4442, dtype: object
Zip                          7676
City     Township Of Washington
State                          NJ
```

```
Name: 4446, dtype: object
Zip                   1535
City     North Brookfield
State                  MA
Name: 4461, dtype: object
Zip         3055
City     Milford
State         NH
Name: 4477, dtype: object
Zip          6160
City     Hartford
State          CT
Name: 4517, dtype: object
Zip              7932
City     Florham Park
State              NJ
Name: 4520, dtype: object
Zip             1550
City     Southbridge
State             MA
Name: 4522, dtype: object
Zip         7012
City     Clifton
State         NJ
Name: 4527, dtype: object
Zip                   3225
City     Center Barnstead
State                  NH
Name: 4544, dtype: object
Zip       4979
City     Solon
State       ME
Name: 4549, dtype: object
Zip           8012
City     Blackwood
State           NJ
Name: 4580, dtype: object
Zip             6447
City     Marlborough
State             CT
Name: 4588, dtype: object
Zip        1854
City     Lowell
State        MA
Name: 4591, dtype: object
Zip        3034
City     Candia
State        NH
```

```
Name: 4594, dtype: object
Zip          1841
City     Lawrence
State          MA
Name: 4639, dtype: object
Zip            5753
City     Middlebury
State            VT
Name: 4652, dtype: object
Zip           3261
City     Northwood
State           NH
Name: 4656, dtype: object
Zip                            6777
City     New Preston Marble Dale
State                           CT
Name: 4677, dtype: object
Zip            4781
City     Wallagrass
State            ME
Name: 4702, dtype: object
Zip            4554
City     New Harbor
State            ME
Name: 4703, dtype: object
Zip               703
City     Aguas Buenas
State              PR
Name: 4737, dtype: object
Zip           987
City     Carolina
State          PR
Name: 4739, dtype: object
Zip          7524
City     Paterson
State          NJ
Name: 4785, dtype: object
Zip           8027
City     Gibbstown
State           NJ
Name: 4789, dtype: object
Zip        6517
City     Hamden
State        CT
Name: 4801, dtype: object
Zip            7095
City     Woodbridge
State            NJ
```

```
Name: 4804, dtype: object
Zip           1029
City     East Otis
State           MA
Name: 4812, dtype: object
Zip           8501
City     Allentown
State           NJ
Name: 4838, dtype: object
Zip                4623
City     Columbia Falls
State                ME
Name: 4840, dtype: object
Zip          7981
City     Whippany
State          NJ
Name: 4843, dtype: object
Zip         7940
City     Madison
State         NJ
Name: 4857, dtype: object
Zip          704
City     Aguirre
State         PR
Name: 4874, dtype: object
Zip               7970
City     Mount Freedom
State               NJ
Name: 4891, dtype: object
Zip           1368
City     Royalston
State           MA
Name: 4919, dtype: object
Zip              5654
City     Graniteville
State              VT
Name: 4923, dtype: object
Zip           5250
City     Arlington
State           VT
Name: 4932, dtype: object
Zip         4563
City     Cushing
State         ME
Name: 4938, dtype: object
Zip         2767
City     Raynham
State         MA
```

```
Name: 4953, dtype: object
Zip        7605
City     Leonia
State        NJ
Name: 4963, dtype: object
Zip        5907
City     Norton
State        VT
Name: 4964, dtype: object
Zip           1606
City     Worcester
State           MA
Name: 5011, dtype: object
Zip            4739
City     Eagle Lake
State            ME
Name: 5043, dtype: object
Zip              1532
City     Northborough
State              MA
Name: 5054, dtype: object
Zip           2915
City     Riverside
State           RI
Name: 5071, dtype: object
Zip         7005
City     Boonton
State         NJ
Name: 5079, dtype: object
Zip            3442
City     Bennington
State            NH
Name: 5094, dtype: object
Zip            2145
City     Somerville
State            MA
Name: 5119, dtype: object
Zip            7003
City     Bloomfield
State            NJ
Name: 5131, dtype: object
Zip           1085
City     Westfield
State           MA
Name: 5137, dtype: object
Zip           8332
City     Millville
State           NJ
```

```
Name: 5138, dtype: object
Zip            8553
City     Rocky Hill
State            NJ
Name: 5176, dtype: object
Zip              7079
City     South Orange
State              NJ
Name: 5183, dtype: object
Zip       4473
City     Orono
State       ME
Name: 5185, dtype: object
Zip                  4662
City     Northeast Harbor
State                  ME
Name: 5190, dtype: object
Zip          2725
City     Somerset
State          MA
Name: 5192, dtype: object
Zip         4071
City     Raymond
State         ME
Name: 5204, dtype: object
Zip      6001
City     Avon
State      CT
Name: 5210, dtype: object
Zip                7444
City     Pompton Plains
State                NJ
Name: 5256, dtype: object
Zip                 5759
City     North Clarendon
State                 VT
Name: 5258, dtype: object
Zip          5033
City     Bradford
State          VT
Name: 5261, dtype: object
Zip         5062
City     Reading
State         VT
Name: 5264, dtype: object
Zip          718
City     Naguabo
State         PR
```

```
Name: 5275, dtype: object
Zip         1040
City     Holyoke
State         MA
Name: 5309, dtype: object
Zip           8733
City     Lakehurst
State           NJ
Name: 5312, dtype: object
Zip              6088
City     East Windsor
State              CT
Name: 5314, dtype: object
Zip          6277
City     Thompson
State          CT
Name: 5327, dtype: object
Zip        4460
City     Medway
State        ME
Name: 5380, dtype: object
Zip            8014
City     Bridgeport
State            NJ
Name: 5381, dtype: object
Zip        1501
City     Auburn
State        MA
Name: 5388, dtype: object
Zip         4942
City     Harmony
State         ME
Name: 5404, dtype: object
Zip            6525
City     Woodbridge
State            CT
Name: 5412, dtype: object
Zip                 5759
City     North Clarendon
State                 VT
Name: 5432, dtype: object
Zip            8553
City     Rocky Hill
State            NJ
Name: 5452, dtype: object
Zip        3233
City     Elkins
State        NH
```

```
Name: 5454, dtype: object
Zip              4747
City     Island Falls
State              ME
Name: 5466, dtype: object
Zip           918
City     San Juan
State          PR
Name: 5468, dtype: object
Zip            6320
City     New London
State            CT
Name: 5473, dtype: object
Zip             8829
City     High Bridge
State             NJ
Name: 5489, dtype: object
Zip             4958
City     North Anson
State             ME
Name: 5491, dtype: object
Zip            7440
City     Pequannock
State            NJ
Name: 5527, dtype: object
Zip                   5471
City     Montgomery Center
State                   VT
Name: 5537, dtype: object
Zip           7111
City     Irvington
State           NJ
Name: 5562, dtype: object
Zip          5033
City     Bradford
State          VT
Name: 5598, dtype: object
Zip          4841
City     Rockland
State          ME
Name: 5608, dtype: object
Zip                     4110
City     Cumberland Foreside
State                     ME
Name: 5617, dtype: object
Zip          7016
City     Cranford
State          NJ
```

```
Name: 5622, dtype: object
Zip           7040
City     Maplewood
State           NJ
Name: 5680, dtype: object
Zip             6052
City     New Britain
State             CT
Name: 5685, dtype: object
Zip           3867
City     Rochester
State           NH
Name: 5699, dtype: object
Zip                 1585
City     West Brookfield
State                 MA
Name: 5739, dtype: object
Zip               3607
City     South Acworth
State               NH
Name: 5756, dtype: object
Zip            7711
City     Allenhurst
State            NJ
Name: 5764, dtype: object
Zip               1864
City     North Reading
State               MA
Name: 5790, dtype: object
Zip         3452
City     Jaffrey
State         NH
Name: 5794, dtype: object
Zip         2149
City     Everett
State         MA
Name: 5804, dtype: object
Zip           622
City     Boqueron
State          PR
Name: 5814, dtype: object
Zip            8804
City     Bloomsbury
State            NJ
Name: 5834, dtype: object
Zip            3269
City     Sanbornton
State            NH
```

```
Name: 5859, dtype: object
Zip                   4570
City     Squirrel Island
State                  ME
Name: 5867, dtype: object
Zip         7641
City     Haworth
State         NJ
Name: 5872, dtype: object
Zip           5763
City     Pittsford
State           VT
Name: 5898, dtype: object
Zip                   5071
City     South Woodstock
State                  VT
Name: 5899, dtype: object
Zip                   4570
City     Squirrel Island
State                  ME
Name: 5912, dtype: object
Zip           1420
City     Fitchburg
State           MA
Name: 5915, dtype: object
Zip         6783
City     Roxbury
State         CT
Name: 5920, dtype: object
Zip             7307
City     Jersey City
State             NJ
Name: 5923, dtype: object
Zip         3782
City     Sunapee
State         NH
Name: 5944, dtype: object
Zip             8052
City     Maple Shade
State             NJ
Name: 5947, dtype: object
Zip         7753
City     Neptune
State         NJ
Name: 5952, dtype: object
Zip                 5448
City     East Fairfield
State                 VT
```

```
Name: 5957, dtype: object
Zip          2886
City      Warwick
State          RI
Name: 5958, dtype: object
Zip            6021
City      Colebrook
State            CT
Name: 5971, dtype: object
Zip             1504
City      Blackstone
State             MA
Name: 5974, dtype: object
Zip         2109
City      Boston
State         MA
Name: 5985, dtype: object
Zip                   1230
City      Great Barrington
State                   MA
Name: 5992, dtype: object
Zip                5084
City      West Hartford
State                VT
Name: 5998, dtype: object
Zip          4949
City      Liberty
State          ME
Name: 6042, dtype: object
Zip              3875
City      Silver Lake
State              NH
Name: 6056, dtype: object
Zip            2770
City      Rochester
State            MA
Name: 6057, dtype: object
Zip          2035
City      Foxboro
State          MA
Name: 6077, dtype: object
Zip          4427
City      Corinth
State          ME
Name: 6083, dtype: object
Zip               3852
City      Milton Mills
State               NH
```

```
Name: 6086, dtype: object
Zip          1606
City     Worcester
State           MA
Name: 6092, dtype: object
Zip                   4538
City     Boothbay Harbor
State                   ME
Name: 6108, dtype: object
Zip        3771
City     Monroe
State        NH
Name: 6133, dtype: object
Zip            7711
City     Allenhurst
State            NJ
Name: 6136, dtype: object
Zip                3575
City     Bretton Woods
State               NH
Name: 6144, dtype: object
Zip          1368
City     Royalston
State           MA
Name: 6154, dtype: object
Zip        4951
City     Monroe
State        ME
Name: 6155, dtype: object
Zip            5201
City     Bennington
State            VT
Name: 6160, dtype: object
Zip         6120
City     Hartford
State          CT
Name: 6165, dtype: object
Zip         4927
City     Clinton
State         ME
Name: 6189, dtype: object
Zip         2061
City     Norwell
State         MA
Name: 6198, dtype: object
Zip                   2494
City     Needham Heights
State                   MA
```

```
Name: 6202, dtype: object
Zip            1008
City      Blandford
State            MA
Name: 6209, dtype: object
Zip            1602
City      Worcester
State            MA
Name: 6223, dtype: object
Zip              5158
City      Westminster
State              VT
Name: 6239, dtype: object
Zip                 5452
City      Essex Junction
State                 VT
Name: 6257, dtype: object
Zip           7701
City      Red Bank
State           NJ
Name: 6263, dtype: object
Zip           5828
City      Danville
State           VT
Name: 6282, dtype: object
Zip                 3259
City      North Sandwich
State                 NH
Name: 6288, dtype: object
Zip          3836
City      Freedom
State          NH
Name: 6313, dtype: object
Zip            7004
City      Fairfield
State            NJ
Name: 6349, dtype: object
Zip           6798
City      Woodbury
State           CT
Name: 6364, dtype: object
Zip               7092
City      Mountainside
State               NJ
Name: 6375, dtype: object
Zip              3911
City      York Harbor
State              ME
```

```
Name: 6379, dtype: object
Zip              7480
City     West Milford
State              NJ
Name: 6396, dtype: object
Zip                7417
City     Franklin Lakes
State                NJ
Name: 6412, dtype: object
Zip      4463
City     Milo
State      ME
Name: 6420, dtype: object
Zip                  1230
City     Great Barrington
State                  MA
Name: 6431, dtype: object
Zip            4760
City     Monticello
State            ME
Name: 6432, dtype: object
Zip           4220
City     Buckfield
State           ME
Name: 6458, dtype: object
Zip            2122
City     Dorchester
State            MA
Name: 6459, dtype: object
Zip        4551
City     Bremen
State        ME
Name: 6467, dtype: object
Zip         3036
City     Chester
State         NH
Name: 6482, dtype: object
Zip            3801
City     Portsmouth
State            NH
Name: 6491, dtype: object
Zip             8805
City     Bound Brook
State             NJ
Name: 6495, dtype: object
Zip          6387
City     Wauregan
State          CT
```

```
Name: 6519, dtype: object
Zip           3604
City     Drewsville
State            NH
Name: 6526, dtype: object
Zip        4464
City     Monson
State        ME
Name: 6533, dtype: object
Zip          8096
City     Woodbury
State          NJ
Name: 6553, dtype: object
Zip          7068
City     Roseland
State          NJ
Name: 6574, dtype: object
Zip          7833
City     Delaware
State          NJ
Name: 6627, dtype: object
Zip        7104
City     Newark
State        NJ
Name: 6636, dtype: object
Zip          7068
City     Roseland
State          NJ
Name: 6647, dtype: object
Zip       8863
City     Fords
State       NJ
Name: 6650, dtype: object
Zip          2790
City     Westport
State          MA
Name: 6651, dtype: object
Zip               2650
City     North Chatham
State               MA
Name: 6673, dtype: object
Zip        5867
City     Sutton
State        VT
Name: 6682, dtype: object
Zip      7723
City     Deal
State      NJ
```

```
Name: 6696, dtype: object
Zip         6856
City     Norwalk
State         CT
Name: 6706, dtype: object
Zip           1834
City     Groveland
State           MA
Name: 6707, dtype: object
Zip          8344
City     Newfield
State          NJ
Name: 6727, dtype: object
Zip          8341
City     Minotola
State          NJ
Name: 6779, dtype: object
Zip         4975
City     Shawmut
State         ME
Name: 6798, dtype: object
Zip            1833
City     Georgetown
State            MA
Name: 6809, dtype: object
Zip           926
City     San Juan
State          PR
Name: 6814, dtype: object
Zip               6256
City     North Windham
State               CT
Name: 6860, dtype: object
Zip                7417
City     Franklin Lakes
State                NJ
Name: 6895, dtype: object
Zip            4901
City     Waterville
State            ME
Name: 6900, dtype: object
Zip            6604
City     Bridgeport
State            CT
Name: 6916, dtype: object
Zip            2904
City     Providence
State            RI
```

```
Name: 6926, dtype: object
Zip          6498
City     Westbrook
State           CT
Name: 6949, dtype: object
Zip        8312
City     Clayton
State         NJ
Name: 6955, dtype: object
Zip                7920
City     Basking Ridge
State              NJ
Name: 6959, dtype: object
Zip          6519
City     New Haven
State           CT
Name: 6964, dtype: object
Zip           6790
City     Torrington
State            CT
Name: 6980, dtype: object
Zip            4066
City     Orrs Island
State             ME
Name: 6981, dtype: object
Zip         6480
City     Portland
State          CT
Name: 6989, dtype: object
Zip       6260
City     Putnam
State        CT
Name: 7041, dtype: object
Zip        7002
City     Bayonne
State         NJ
Name: 7042, dtype: object
Zip             1267
City     Williamstown
State              MA
Name: 7056, dtype: object
Zip               4359
City     South Gardiner
State                ME
Name: 7060, dtype: object
Zip             4024
City     East Baldwin
State              ME
```

```
Name: 7076, dtype: object
Zip         4105
City    Falmouth
State         ME
Name: 7084, dtype: object
Zip        7670
City    Tenafly
State        NJ
Name: 7107, dtype: object
Zip           913
City     San Juan
State          PR
Name: 7133, dtype: object
Zip               8401
City     Atlantic City
State               NJ
Name: 7135, dtype: object
Zip              7432
City     Midland Park
State              NJ
Name: 7140, dtype: object
Zip        3745
City    Cornish
State        NH
Name: 7151, dtype: object
Zip         8041
City    Jobstown
State         NJ
Name: 7158, dtype: object
Zip               6074
City     South Windsor
State               CT
Name: 7163, dtype: object
Zip                1151
City     Indian Orchard
State                MA
Name: 7184, dtype: object
Zip             2746
City     New Bedford
State             MA
Name: 7196, dtype: object
Zip       4611
City     Beals
State       ME
Name: 7229, dtype: object
Zip        2802
City     Albion
State        RI
```

```
Name: 7239, dtype: object
Zip          6106
City     Hartford
State          CT
Name: 7262, dtype: object
Zip         1040
City     Holyoke
State         MA
Name: 7275, dtype: object
Zip           4043
City     Kennebunk
State           ME
Name: 7296, dtype: object
Zip             5758
City     Mount Holly
State             VT
Name: 7344, dtype: object
Zip         5343
City     Jamaica
State         VT
Name: 7349, dtype: object
Zip          4478
City     Rockwood
State          ME
Name: 7365, dtype: object
Zip            6042
City     Manchester
State            CT
Name: 7372, dtype: object
Zip         3246
City     Laconia
State         NH
Name: 7392, dtype: object
Zip      8004
City     Atco
State      NJ
Name: 7398, dtype: object
Zip           7803
City     Mine Hill
State           NJ
Name: 7401, dtype: object
Zip            8854
City     Piscataway
State            NJ
Name: 7403, dtype: object
Zip         7030
City     Hoboken
State         NJ
```

```
Name: 7407, dtype: object
Zip       4261
City     Newry
State       ME
Name: 7426, dtype: object
Zip          4410
City     Bradford
State          ME
Name: 7456, dtype: object
Zip          3464
City     Stoddard
State          NH
Name: 7471, dtype: object
Zip         3904
City     Kittery
State         ME
Name: 7485, dtype: object
Zip            7876
City     Succasunna
State            NJ
Name: 7556, dtype: object
Zip             5441
City     Bakersfield
State             VT
Name: 7573, dtype: object
Zip           8078
City     Runnemede
State           NJ
Name: 7588, dtype: object
Zip            2050
City     Marshfield
State            MA
Name: 7613, dtype: object
Zip        1033
City     Granby
State        MA
Name: 7618, dtype: object
Zip           2048
City     Mansfield
State           MA
Name: 7642, dtype: object
Zip         4287
City     Bowdoin
State         ME
Name: 7655, dtype: object
Zip         2056
City     Norfolk
State         MA
```

```
Name: 7677, dtype: object
Zip        1035
City     Hadley
State        MA
Name: 7690, dtype: object
Zip            7062
City     Plainfield
State            NJ
Name: 7696, dtype: object
Zip          1037
City     Hardwick
State          MA
Name: 7697, dtype: object
Zip           1940
City     Lynnfield
State           MA
Name: 7698, dtype: object
Zip         1464
City     Shirley
State         MA
Name: 7708, dtype: object
Zip            4955
City     New Sharon
State            ME
Name: 7717, dtype: object
Zip         5038
City     Chelsea
State         VT
Name: 7733, dtype: object
Zip            6085
City     Unionville
State            CT
Name: 7734, dtype: object
Zip              4108
City     Peaks Island
State              ME
Name: 7740, dtype: object
Zip           1468
City     Templeton
State           MA
Name: 7748, dtype: object
Zip          2631
City     Brewster
State          MA
Name: 7753, dtype: object
Zip        5065
City     Sharon
State        VT
```

```
Name: 7799, dtype: object
Zip              6026
City     East Granby
State              CT
Name: 7816, dtype: object
Zip              4431
City     East Orland
State              ME
Name: 7829, dtype: object
Zip                   2333
City     East Bridgewater
State                   MA
Name: 7844, dtype: object
Zip           4970
City      Rangeley
State           ME
Name: 7847, dtype: object
Zip              1074
City      South Barre
State              MA
Name: 7873, dtype: object
Zip                 7417
City     Franklin Lakes
State                 NJ
Name: 7877, dtype: object
Zip                1350
City     Monroe Bridge
State                MA
Name: 7884, dtype: object
Zip                   8853
City      Neshanic Station
State                   NJ
Name: 7893, dtype: object
Zip            773
City      Luquillo
State           PR
Name: 7898, dtype: object
Zip            926
City      San Juan
State           PR
Name: 7903, dtype: object
Zip                1583
City     West Boylston
State                MA
Name: 7928, dtype: object
Zip         610
City     Anasco
State        PR
```

```
Name: 7946, dtype: object
Zip                  1561
City     South Lancaster
State                  MA
Name: 7949, dtype: object
Zip                5873
City     West Danville
State                VT
Name: 7959, dtype: object
Zip           6905
City     Stamford
State          CT
Name: 7960, dtype: object
Zip         6069
City     Sharon
State        CT
Name: 7974, dtype: object
Zip               8844
City     Hillsborough
State              NJ
Name: 7977, dtype: object
Zip             7657
City     Ridgefield
State            NJ
Name: 8009, dtype: object
Zip            1880
City     Wakefield
State           MA
Name: 8015, dtype: object
Zip                     2464
City     Newton Upper Falls
State                     MA
Name: 8027, dtype: object
Zip       676
City     Moca
State      PR
Name: 8038, dtype: object
Zip            4355
City     Readfield
State           ME
Name: 8040, dtype: object
Zip          6451
City     Meriden
State         CT
Name: 8053, dtype: object
Zip            3574
City     Bethlehem
State           NH
```

```
Name: 8131, dtype: object
Zip         3303
City     Concord
State         NH
Name: 8165, dtype: object
Zip          4231
City     Stoneham
State          ME
Name: 8169, dtype: object
Zip               1347
City     Lake Pleasant
State               MA
Name: 8176, dtype: object
Zip           6615
City     Stratford
State           CT
Name: 8184, dtype: object
Zip             670
City     Las Marias
State            PR
Name: 8191, dtype: object
Zip              7064
City     Port Reading
State              NJ
Name: 8200, dtype: object
Zip            6606
City     Bridgeport
State            CT
Name: 8204, dtype: object
Zip             4622
City     Cherryfield
State             ME
Name: 8205, dtype: object
Zip         8106
City     Audubon
State         NJ
Name: 8206, dtype: object
Zip         7419
City     Hamburg
State         NJ
Name: 8210, dtype: object
Zip             4050
City     Long Island
State             ME
Name: 8214, dtype: object
Zip          4925
City     Caratunk
State          ME
```

```
Name: 8219, dtype: object
Zip        669
City     Lares
State       PR
Name: 8235, dtype: object
Zip            8016
City     Burlington
State            NJ
Name: 8281, dtype: object
Zip        4740
City     Easton
State        ME
Name: 8299, dtype: object
Zip               8901
City     New Brunswick
State               NJ
Name: 8315, dtype: object
Zip          5828
City     Danville
State          VT
Name: 8329, dtype: object
Zip         6807
City     Cos Cob
State         CT
Name: 8344, dtype: object
Zip             7081
City     Springfield
State             NJ
Name: 8383, dtype: object
Zip            2125
City     Dorchester
State            MA
Name: 8394, dtype: object
Zip        4683
City     Sunset
State        ME
Name: 8404, dtype: object
Zip            8036
City     Hainesport
State            NJ
Name: 8414, dtype: object
Zip                 3774
City     North Haverhill
State                 NH
Name: 8421, dtype: object
Zip             8807
City     Bridgewater
State             NJ
```

```
Name: 8426, dtype: object
Zip        4843
City     Camden
State        ME
Name: 8501, dtype: object
Zip            8050
City     Manahawkin
State            NJ
Name: 8514, dtype: object
Zip                 6471
City     North Branford
State                CT
Name: 8517, dtype: object
Zip            8520
City     Hightstown
State            NJ
Name: 8520, dtype: object
Zip             1060
City     Northampton
State             MA
Name: 8572, dtype: object
Zip             1098
City     Worthington
State             MA
Name: 8607, dtype: object
Zip          4105
City     Falmouth
State          ME
Name: 8611, dtype: object
Zip            8110
City     Pennsauken
State            NJ
Name: 8620, dtype: object
Zip                  4669
City     Prospect Harbor
State                  ME
Name: 8622, dtype: object
Zip         778
City     Gurabo
State        PR
Name: 8627, dtype: object
Zip           8352
City     Rosenhayn
State           NJ
Name: 8634, dtype: object
Zip             6053
City     New Britain
State             CT
```

```
Name: 8645, dtype: object
Zip      1902
City     Lynn
State      MA
Name: 8650, dtype: object
Zip               6074
City     South Windsor
State               CT
Name: 8652, dtype: object
Zip               7662
City     Rochelle Park
State               NJ
Name: 8670, dtype: object
Zip           1029
City     East Otis
State           MA
Name: 8681, dtype: object
Zip              8752
City     Seaside Park
State              NJ
Name: 8682, dtype: object
Zip               6108
City     East Hartford
State               CT
Name: 8683, dtype: object
Zip           6787
City     Thomaston
State           CT
Name: 8690, dtype: object
Zip              8108
City     Collingswood
State              NJ
Name: 8694, dtype: object
Zip              2647
City     Hyannis Port
State              MA
Name: 8699, dtype: object
Zip        677
City     Rincon
State        PR
Name: 8726, dtype: object
Zip            6804
City     Brookfield
State            CT
Name: 8733, dtype: object
Zip          8403
City     Longport
State          NJ
```

```
Name: 8757, dtype: object
Zip        4472
City     Orland
State        ME
Name: 8775, dtype: object
Zip           6385
City     Waterford
State           CT
Name: 8793, dtype: object
Zip           4861
City     Thomaston
State           ME
Name: 8822, dtype: object
Zip            6804
City     Brookfield
State            CT
Name: 8832, dtype: object
Zip         1844
City     Methuen
State         MA
Name: 8837, dtype: object
Zip            4414
City     Brownville
State            ME
Name: 8867, dtype: object
Zip        6763
City     Morris
State        CT
Name: 8879, dtype: object
Zip          6467
City     Milldale
State          CT
Name: 8890, dtype: object
Zip        1057
City     Monson
State        MA
Name: 8891, dtype: object
Zip             2744
City     New Bedford
State             MA
Name: 8897, dtype: object
Zip           624
City     Penuelas
State          PR
Name: 8933, dtype: object
Zip            8070
City     Pennsville
State            NJ
```

```
Name: 8940, dtype: object
Zip           3743
City     Claremont
State           NH
Name: 8951, dtype: object
Zip              5042
City     East Ryegate
State              VT
Name: 8956, dtype: object
Zip            4441
City     Greenville
State            ME
Name: 8962, dtype: object
Zip         6360
City     Norwich
State         CT
Name: 8965, dtype: object
Zip            3855
City     New Durham
State            NH
Name: 8967, dtype: object
Zip          7522
City     Paterson
State          NJ
Name: 8978, dtype: object
Zip            4289
City     West Paris
State            ME
Name: 8980, dtype: object
Zip         4963
City     Oakland
State         ME
Name: 8996, dtype: object
Zip                  6269
City     Storrs Mansfield
State                  CT
Name: 9001, dtype: object
Zip        4930
City     Dexter
State        ME
Name: 9020, dtype: object
Zip            1050
City     Huntington
State            MA
Name: 9023, dtype: object
Zip                 7881
City     Wallpack Center
State                 NJ
```

```
Name: 9028, dtype: object
Zip           1830
City     Haverhill
State           MA
Name: 9038, dtype: object
Zip              5086
City     West Topsham
State              VT
Name: 9070, dtype: object
Zip                637
City     Sabana Grande
State               PR
Name: 9078, dtype: object
Zip         4237
City     Hanover
State         ME
Name: 9082, dtype: object
Zip           7506
City     Hawthorne
State           NJ
Name: 9109, dtype: object
Zip               8734
City     Lanoka Harbor
State               NJ
Name: 9132, dtype: object
Zip          4224
City     Dixfield
State          ME
Name: 9145, dtype: object
Zip       2030
City     Dover
State       MA
Name: 9167, dtype: object
Zip          2302
City     Brockton
State          MA
Name: 9202, dtype: object
Zip         6084
City     Tolland
State         CT
Name: 9209, dtype: object
Zip                 5651
City     East Montpelier
State                 VT
Name: 9221, dtype: object
Zip          775
City     Culebra
State         PR
```

```
Name: 9231, dtype: object
Zip                4095
City     West Newfield
State               ME
Name: 9234, dtype: object
Zip           907
City     San Juan
State          PR
Name: 9243, dtype: object
Zip        4766
City     Perham
State        ME
Name: 9271, dtype: object
Zip           2445
City     Brookline
State           MA
Name: 9297, dtype: object
Zip         4457
City     Lincoln
State         ME
Name: 9312, dtype: object
Zip          7501
City     Paterson
State          NJ
Name: 9411, dtype: object
Zip               2876
City     Slatersville
State              RI
Name: 9413, dtype: object
Zip                6091
City     West Hartland
State               CT
Name: 9436, dtype: object
Zip        5261
City     Pownal
State        VT
Name: 9437, dtype: object
Zip                 8641
City     Joint Base Mdl
State                NJ
Name: 9438, dtype: object
Zip       1346
City     Heath
State       MA
Name: 9441, dtype: object
Zip             4762
City     New Sweden
State             ME
```

```
Name: 9454, dtype: object
Zip                8732
City     Island Heights
State                NJ
Name: 9461, dtype: object
Zip               5850
City     Lyndon Center
State               VT
Name: 9473, dtype: object
Zip             4958
City     North Anson
State             ME
Name: 9479, dtype: object
Zip          3253
City     Meredith
State          NH
Name: 9485, dtype: object
Zip        8820
City     Edison
State        NJ
Name: 9495, dtype: object
Zip           7423
City     Ho Ho Kus
State           NJ
Name: 9505, dtype: object
Zip         2645
City     Harwich
State         MA
Name: 9510, dtype: object
Zip            5356
City     West Dover
State            VT
Name: 9560, dtype: object
Zip            2895
City     Woonsocket
State            RI
Name: 9578, dtype: object
Zip              7663
City     Saddle Brook
State              NJ
Name: 9607, dtype: object
Zip         3446
City     Swanzey
State         NH
Name: 9643, dtype: object
Zip          2878
City     Tiverton
State          RI
```

```
Name: 9659, dtype: object
Zip      4463
City     Milo
State      ME
Name: 9661, dtype: object
Zip           7450
City     Ridgewood
State           NJ
Name: 9719, dtype: object
Zip           6353
City     Montville
State           CT
Name: 9726, dtype: object
Zip         2771
City     Seekonk
State         MA
Name: 9739, dtype: object
Zip            4353
City     Whitefield
State            ME
Name: 9749, dtype: object
Zip         8223
City     Marmora
State         NJ
Name: 9753, dtype: object
Zip           3054
City     Merrimack
State           NH
Name: 9821, dtype: object
Zip               3609
City     North Walpole
State               NH
Name: 9826, dtype: object
Zip         4461
City     Milford
State         ME
Name: 9837, dtype: object
Zip               3751
City     Georges Mills
State               NH
Name: 9842, dtype: object
Zip               5902
City     Beecher Falls
State               VT
Name: 9846, dtype: object
Zip            7439
City     Ogdensburg
State            NJ
```

```
Name: 9865, dtype: object
Zip          8360
City     Vineland
State          NJ
Name: 9866, dtype: object
Zip          8077
City     Riverton
State          NJ
Name: 9888, dtype: object
Zip           682
City     Mayaguez
State          PR
Name: 9895, dtype: object
Zip            5762
City     Pittsfield
State            VT
Name: 9909, dtype: object
Zip         1093
City     Whately
State         MA
Name: 9925, dtype: object
Zip           7843
City     Hopatcong
State           NJ
Name: 9930, dtype: object
Zip              4475
City     Passadumkeag
State              ME
Name: 9945, dtype: object
Zip        7731
City     Howell
State        NJ
Name: 9946, dtype: object
Zip                     4415
City     Brownville Junction
State                     ME
Name: 9976, dtype: object
Zip         6084
City     Tolland
State         CT
Name: 9977, dtype: object
Zip               8401
City     Atlantic City
State               NJ
Name: 9983, dtype: object
Zip           7647
City     Northvale
State           NJ
```

```
Name: 9994, dtype: object
Zip         8340
City      Milmay
State         NJ
Name: 9997, dtype: object
```

[55]:
```
#correct invalid zipcodes
invalid_zip_indexes = invalid_list.index.values
for x in invalid_zip_indexes:
    df.loc[x, 'Zip'] = df.loc[x, 'Zip'].zfill(5)
```

[56]:
```
df['Zip'][df['Zip'].apply(len) != 5]
```

[56]:
```
Series([], Name: Zip, dtype: object)
```

[57]:
```
df.loc[invalid_zip_indexes, 'Zip']
```

[57]:
```
Case_order
32      02584
36      05043
37      02468
38      02138
68      03464
        …
9976    04415
9977    06084
9983    08401
9994    07647
9997    08340
Name: Zip, Length: 723, dtype: object
```

[58]:
```
#Round total and aditional charges to 2 decimal places. These values were␣
↪generated based on averages
#and were not standardized for typical use of monatary values
df[['Total_charge', 'Additional_charges']] = np.around(df[['Total_charge',␣
↪'Additional_charges']], 2)
df[['Total_charge', 'Additional_charges']]
```

[58]:

	Total_charge	Additional_charges
Case_order		
1	3191.05	17939.40
2	4214.91	17613.00
3	2177.59	17505.19
4	2465.12	12993.44
5	1885.66	3716.53
…	…	…
9996	6651.24	8927.64
9997	7851.52	28507.15

```
9998             7725.95           15281.21
9999             8462.83            7781.68
10000            8700.86           11643.19

[10000 rows x 2 columns]
```

[59]:
```
#reduce precision of initial days varible to allow for more meaningful data␣
↪analysis
df[['Initial_days']] = np.around(df[['Initial_days']], 1)
df['Initial_days'].value_counts()
```

[59]:
```
3.3     56
1.3     48
7.8     47
2.8     44
1.6     43
        ..
22.7     1
32.1     1
28.9     1
25.0     1
31.8     1
Name: Initial_days, Length: 646, dtype: int64
```

[60]:
```
#isolate numeric values for outlier detection
numeric_data = df[['Population', 'Children', 'Age', 'Income', 'VitD_levels',␣
↪'Doc_visits', 'Full_meals_eaten',
       'VitD_supplements', 'Initial_days', 'Total_charge',␣
↪'Additional_charges']].copy()
```

[61]:
```
#Outliers are identified and isolated using a combination of box plots and z␣
↪scores. Where needed histograms are used
#for further analysis. In cases where z scores were not suitable for outlier␣
↪isolation iqr was used instead.
#outliers are stored in a seperate varible named <varible_name>_outliers, but␣
↪not removed from the original dataset.
#This is done so that analysis can be performed on dataset both including and␣
↪excluding outliers,
#because while outliers are present they are not abnormal values for the data␣
↪type.
#helper function to add boolean outlier column to main dataframe for a specific␣
↪column. this can be used during later
#data analysis to easily include or exclude outliers from analysis
def Add_outlier_column(data_frame, outliers, column):
    data_frame[column + '_outliers'] = False
    for x in outliers.index:
        data_frame.at[x-1, column + '_outliers'] = True
```

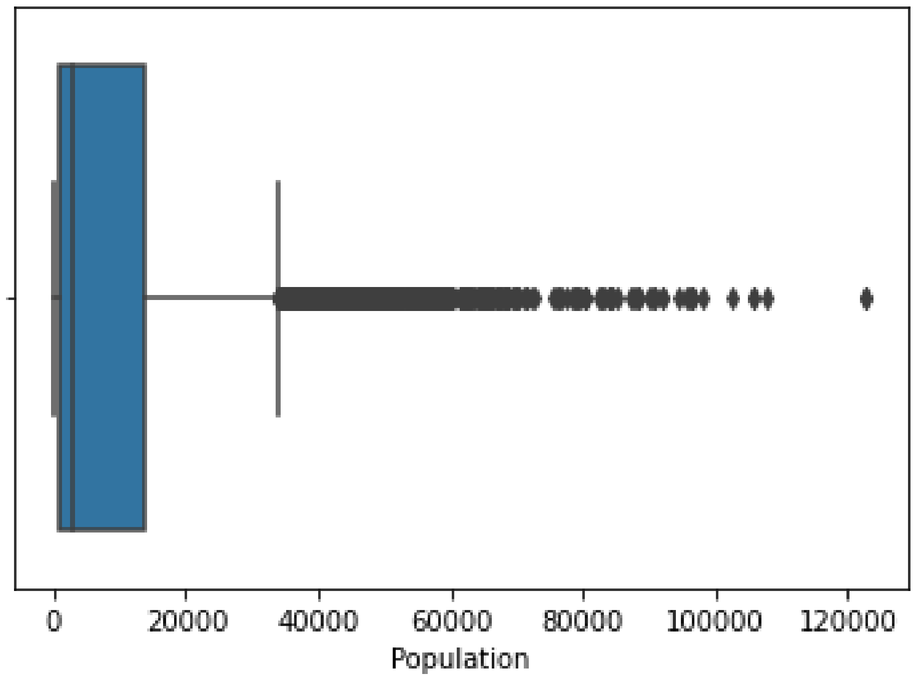
[62]:
```python
for x in numeric_data:
    numeric_data[x + '_z'] = stats.zscore(numeric_data[x])
numeric_data
```

[62]:
```
            Population  Children  Age    Income  VitD_levels  Doc_visits  \
Case_order
1                 2951         1   53  86575.93    17.802330           6
2                11303         3   51  46805.99    18.994640           4
3                17125         3   53  14370.14    17.415889           4
4                 2162         0   78  39741.49    17.420079           4
5                 5287         0   22   1209.56    16.870524           5
…                    …       …   …         …            …           …
9996              4762         6   25  45967.61    16.481612           4
9997              1251         4   87  14983.02    18.451601           5
9998               532         3   65  65917.81    15.752751           4
9999               271         3   43  29702.32    21.956305           5
10000            41524         8   43  62682.63    20.421883           5

            Full_meals_eaten  VitD_supplements  Initial_days  Total_charge  \
Case_order
1                          0                 0          10.6       3191.05
2                          2                 1          15.1       4214.91
3                          1                 0           4.8       2177.59
4                          1                 0           1.7       2465.12
5                          0                 2           1.3       1885.66
…                          …                 …             …           …
9996                       2                 1          51.6       6651.24
9997                       0                 0          68.7       7851.52
9998                       2                 0          66.0       7725.95
9999                       2                 1          63.4       8462.83
10000                      0                 1          70.9       8700.86

            …  Children_z     Age_z  Income_z  VitD_levels_z  Doc_visits_z  \
Case_order  …
1           …   -0.510287 -0.014419  1.709132      -0.239530      0.944647
2           …    0.412224 -0.117337  0.233169      -0.062181     -0.967981
3           …    0.412224 -0.014419 -0.970607      -0.297011     -0.967981
4           …   -0.971543  1.272056 -0.029012      -0.296388     -0.967981
5           …   -0.971543 -1.609647 -1.459030      -0.378131     -0.011667
…          …          …         …         …              …             …
9996        …    1.795992 -1.455270  0.202055      -0.435979     -0.967981
9997        …    0.873480  1.735187 -0.947862      -0.142954     -0.011667
9998        …    0.412224  0.603089  0.942457      -0.544393     -0.967981
9999        …    0.412224 -0.529009 -0.401591       0.378351     -0.011667
10000       …    2.718503 -0.529009  0.822391       0.150114     -0.011667

            Full_meals_eaten_z  VitD_supplements_z  Initial_days_z  \
```

```
Case_order
1                      -0.993387           -0.634713       -0.908650
2                       0.990609            0.956445       -0.737310
3                      -0.001389           -0.634713       -1.129488
4                      -0.001389           -0.634713       -1.247522
5                      -0.993387            2.547602       -1.262752
…                              …                   …               …
9996                    0.990609            0.956445        0.652447
9997                   -0.993387           -0.634713        1.303539
9998                    0.990609           -0.634713        1.200735
9999                    0.990609            0.956445        1.101738
10000                  -0.993387            0.956445        1.387305

            Total_charge_z  Additional_charges_z
Case_order
1                -0.799579              0.765005
2                -0.496427              0.715114
3                -1.099651              0.698635
4                -1.014517              0.009005
5                -1.186087             -1.408990
…                        …                     …
9996              0.224938             -0.612461
9997              0.580324              2.380307
9998              0.543145              0.358695
9999              0.761325             -0.787623
10000             0.831803             -0.197384

[10000 rows x 22 columns]
```

[63]:
```python
sns.boxplot(x=numeric_data['Population'])
```

[63]:
```
<AxesSubplot:xlabel='Population'>
```



[64]:
```
population_outliers = numeric_data.loc[(numeric_data['Population_z'] > 3) |␣
↪(numeric_data['Population_z'] < -3),
                    ['Population', 'Population_z']]
Add_outlier_column(df, population_outliers, 'Population')
population_outliers.sort_values('Population')
```

[64]:

	Population	Population_z
Case_order		
289	54453	3.001059
965	54460	3.001531
6797	54507	3.004701
3820	54647	3.014146
3186	54647	3.014146
…	…	…
768	105799	6.464762
7687	105799	6.464762
5966	107700	6.593000
9663	122814	7.612562
3025	122814	7.612562

[218 rows x 2 columns]

[65]:
```
population_outliers.sort_values('Population').value_counts()
```

[65]:

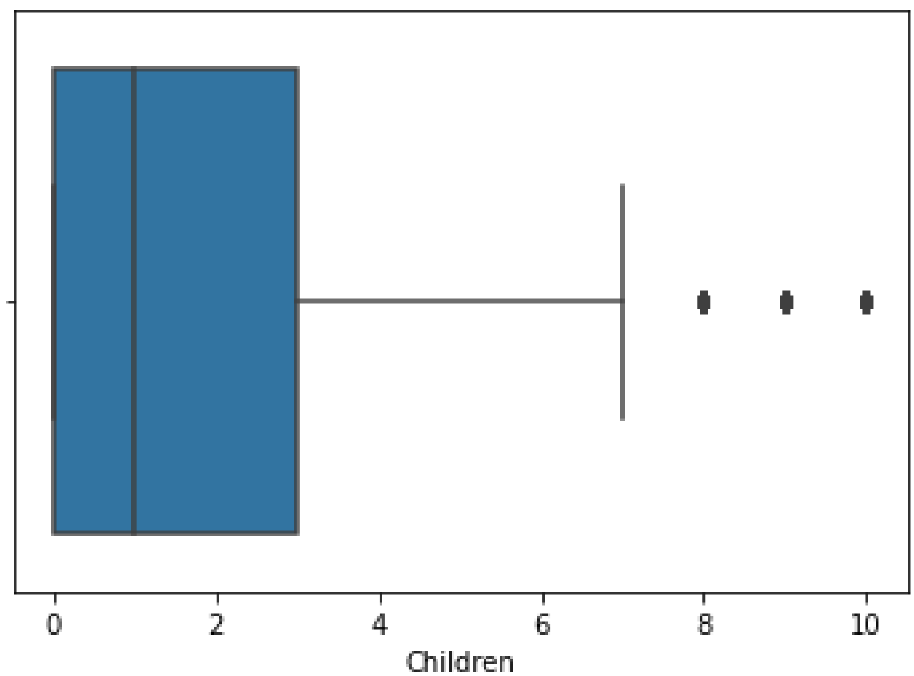
```
Population  Population_z
57775       3.225154        3
83960       4.991545        3
67597       3.887728        3
59129       3.316493        2
84418       5.022441        2
                           ..
59699       3.354944        1
60033       3.377475        1
60081       3.380713        1
60107       3.382467        1
63425       3.606293        1
Length: 186, dtype: int64
```

[66]:

```python
sns.boxplot(x=numeric_data['Children'])
```

[66]:

```
<AxesSubplot:xlabel='Children'>
```



[67]:

```python
children_outliers = numeric_data.loc[(numeric_data['Children_z'] > 3) |
↪(numeric_data['Children_z'] < -3),
                      ['Children', 'Children_z']]
Add_outlier_column(df, children_outliers, 'Children')
children_outliers.sort_values('Children')
```

[67]:

```
              Children  Children_z
Case_order
4459                 9    3.179759
4134                 9    3.179759
4110                 9    3.179759
4049                 9    3.179759
4048                 9    3.179759
…                  …          …
2282                10    3.641015
2196                10    3.641015
2125                10    3.641015
6831                10    3.641015
9846                10    3.641015

[210 rows x 2 columns]
```

[68]:

```
children_outliers.sort_values('Children').value_counts()
```

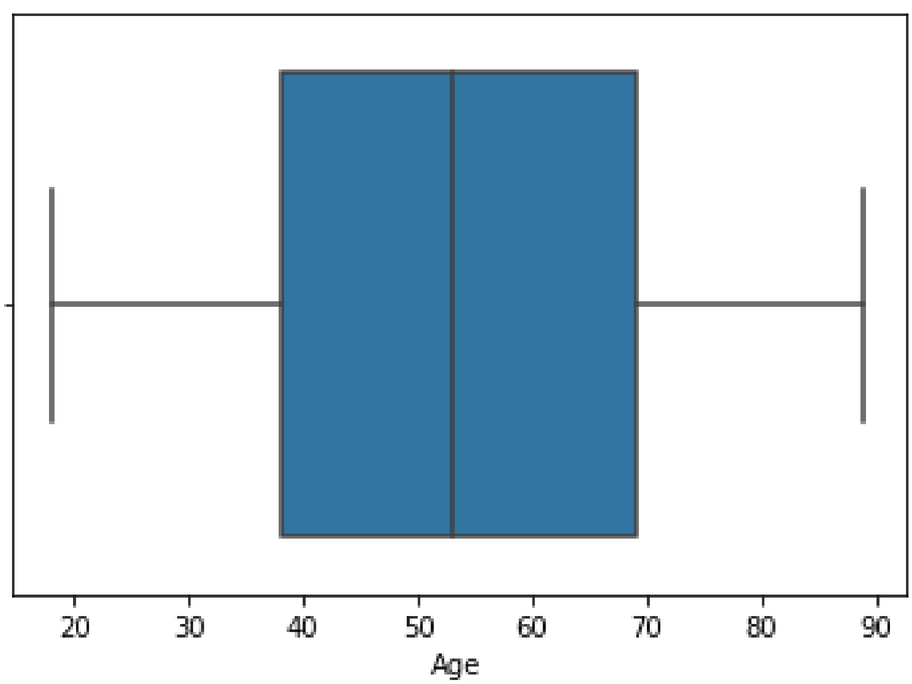
[68]:

```
Children  Children_z
9         3.179759      125
10        3.641015       85
dtype: int64
```

[69]:

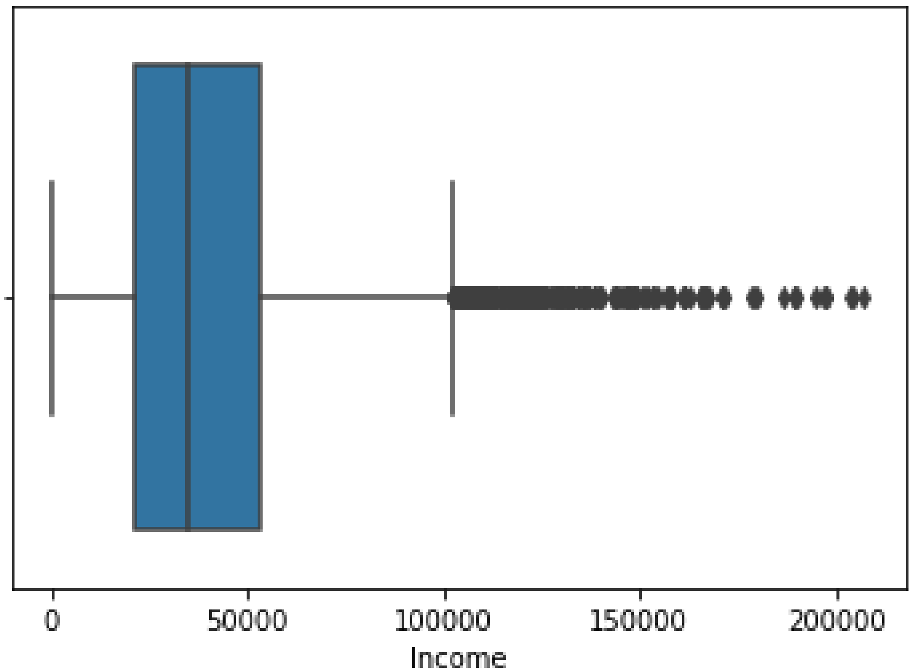
```
sns.boxplot(x=numeric_data['Age'])
```

[69]: <AxesSubplot:xlabel='Age'>



[70]:
```
sns.boxplot(x=numeric_data['Income'])
```

[70]: <AxesSubplot:xlabel='Income'>



[71]:
```
income_outliers = numeric_data.loc[(numeric_data['Income_z'] > 3) |␣
↪(numeric_data['Income_z'] < -3),
                 ['Income', 'Income_z']]
Add_outlier_column(df, income_outliers, 'Income')
income_outliers.sort_values('Income')
```

[71]:

	Income	Income_z
Case_order		
1515	121766.35	3.015137
9345	121931.19	3.021255
9956	122291.51	3.034627
9141	122361.47	3.037224
37	122615.82	3.046663
…	…	…
1779	197576.18	5.828632
6407	197675.05	5.832301
8599	203774.65	6.058672
842	204542.41	6.087166
8387	207249.13	6.187619

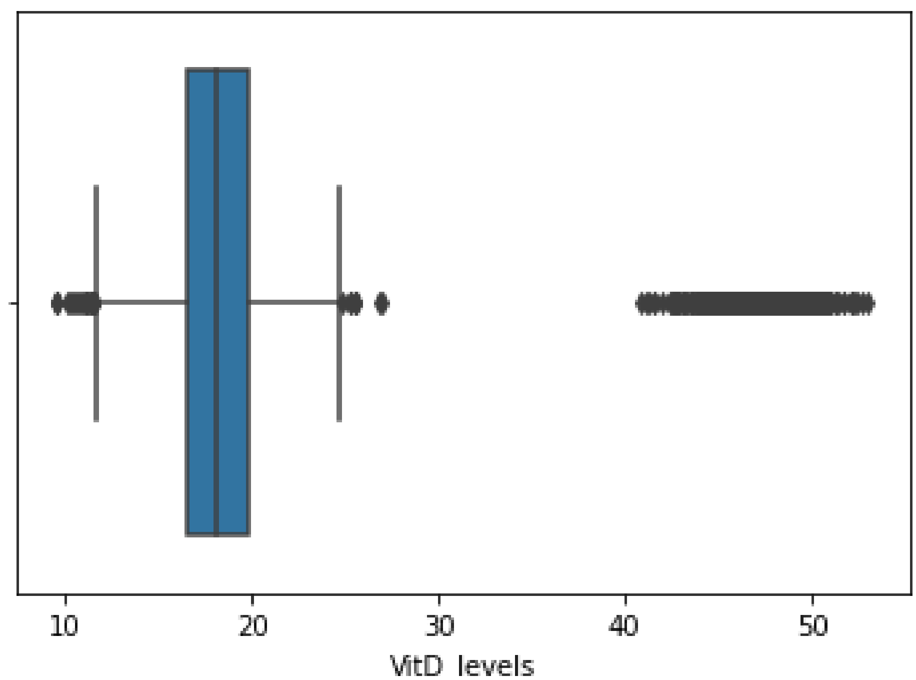
```
[140 rows x 2 columns]
```

[72]:
```
income_outliers.sort_values('Income').value_counts()
```

[72]:
```
Income     Income_z
121766.35  3.015137    1
148944.14  4.023774    1
147303.68  3.962892    1
147570.86  3.972808    1
148141.83  3.993998    1
                      ..
129987.32  3.320238    1
129945.51  3.318687    1
129586.68  3.305370    1
129349.07  3.296551    1
207249.13  6.187619    1
Length: 140, dtype: int64
```

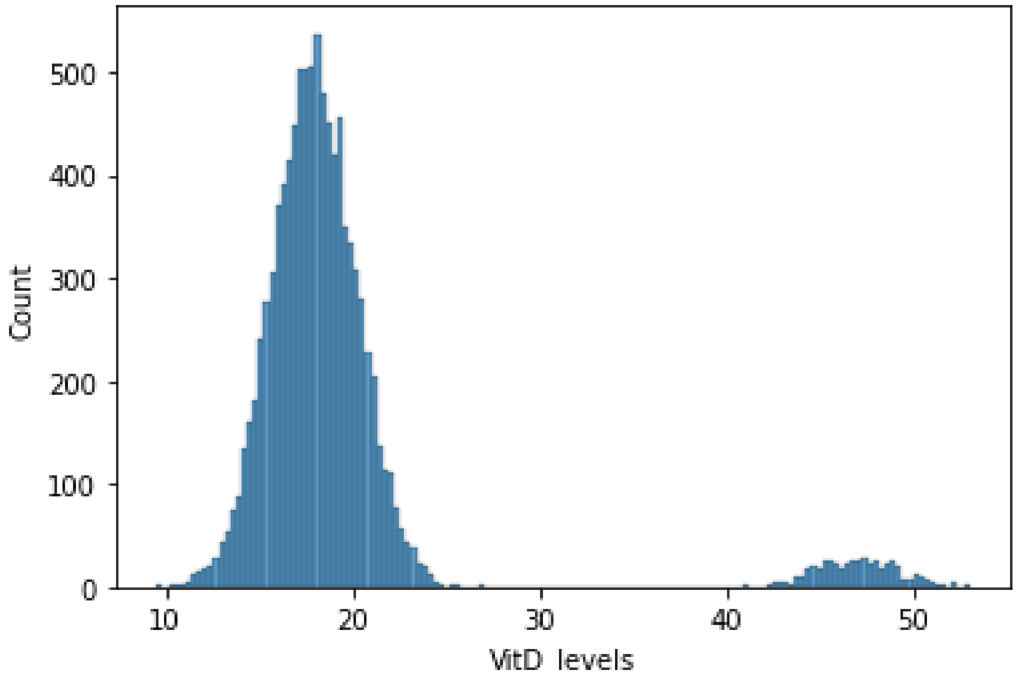
[73]:
```
sns.boxplot(x=numeric_data['VitD_levels'])
```

[73]:
```
<AxesSubplot:xlabel='VitD_levels'>
```



[74]:
```
sns.histplot(numeric_data['VitD_levels'])
```

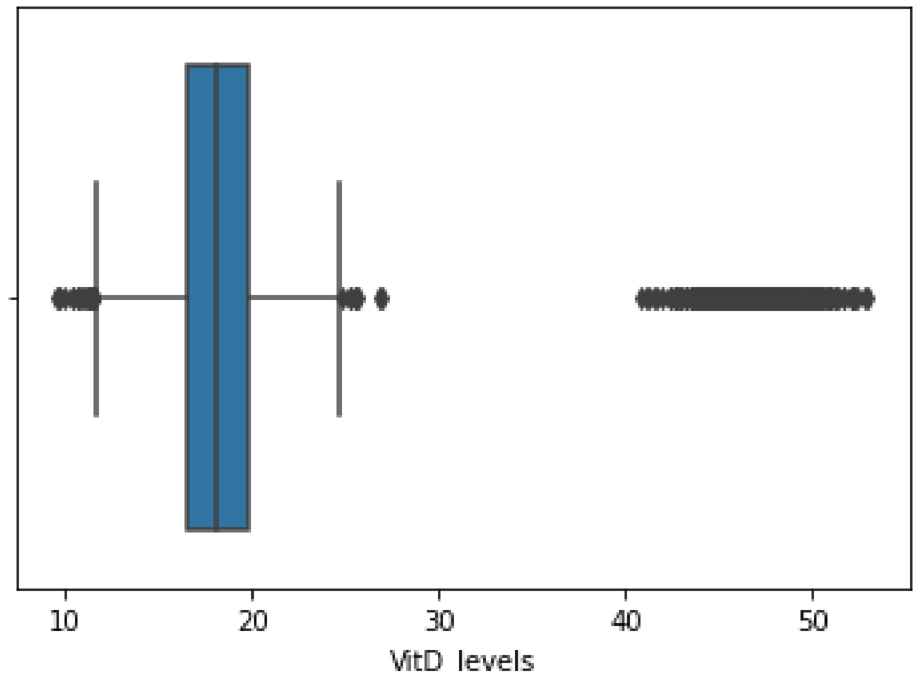
[74]: <AxesSubplot:xlabel='VitD_levels', ylabel='Count'>



[75]:
```
# VitD column in both numeric data and full data frame precision reduced to␣
↪tenths decimal place to
#conform to typical measurement of data of this type as shown listed in sources.

numeric_data[['VitD_levels']] = df[['VitD_levels']] = np.
↪around(numeric_data['VitD_levels'], 1)
numeric_data['VitD_levels_z'] = stats.zscore(numeric_data['VitD_levels'])
```

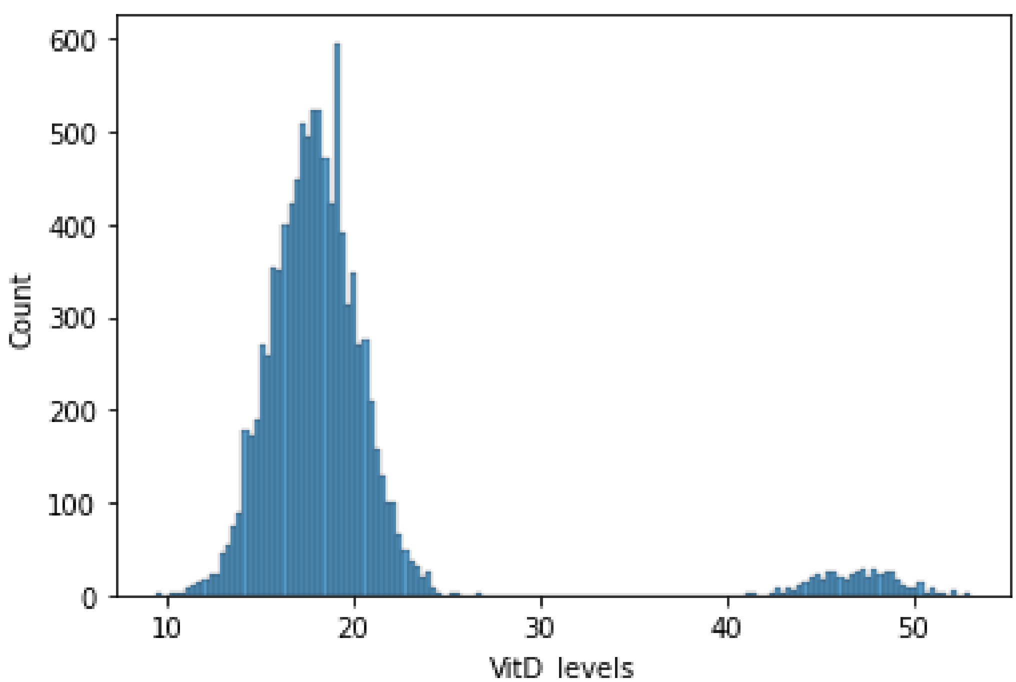
[76]:
```
sns.boxplot(x=numeric_data['VitD_levels'])
```

[76]: <AxesSubplot:xlabel='VitD_levels'>



[77]:
```
sns.histplot(numeric_data['VitD_levels'])
```

[77]: <AxesSubplot:xlabel='VitD_levels', ylabel='Count'>



[78]:
```
vitD_levels_outliers = numeric_data.loc[(numeric_data['VitD_levels_z'] > 3) |␣
↪(numeric_data['VitD_levels_z'] < -3),
                 ['VitD_levels', 'VitD_levels_z']]
Add_outlier_column(df, vitD_levels_outliers, 'VitD_levels')
vitD_levels_outliers.sort_values('VitD_levels')
```

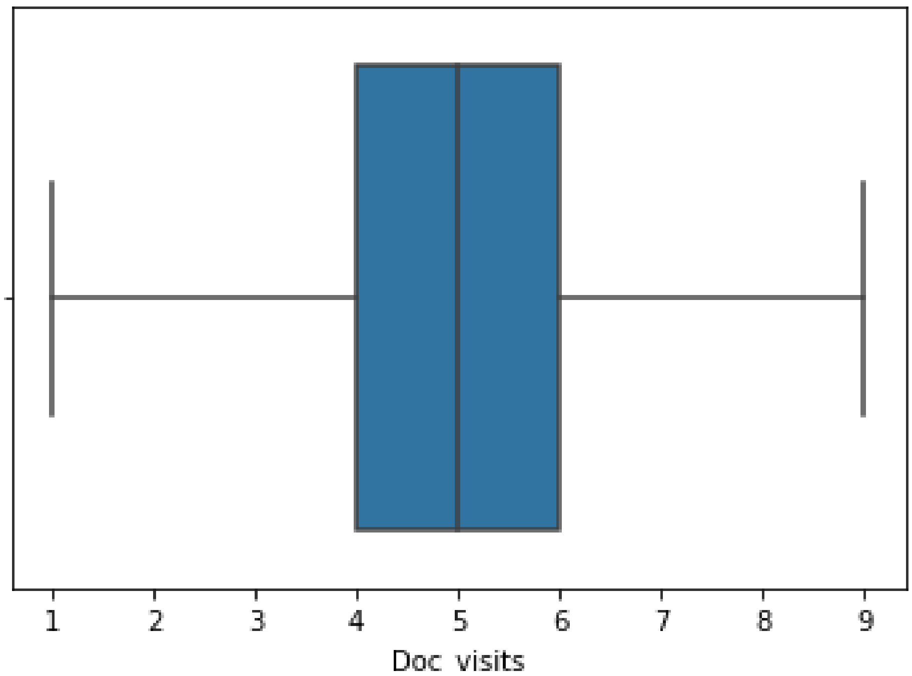
[78]:
```
            VitD_levels  VitD_levels_z
Case_order
8198               40.8       3.181200
787                41.1       3.225822
7271               41.2       3.240697
2947               41.5       3.285319
5689               41.6       3.300193
…                   …           …
2616               52.2       4.876861
7231               52.3       4.891736
7158               52.4       4.906610
1307               52.8       4.966107
1964               53.0       4.995855

[500 rows x 2 columns]
```

[79]:
```
vitD_levels_outliers.sort_values('VitD_levels').value_counts()
```

[79]:
```
VitD_levels  VitD_levels_z
47.8         4.222395         12
46.6         4.043905         11
48.5         4.326515         11
45.6         3.895162         11
45.9         3.939785         11
                              ..
42.0         3.359690          1
41.6         3.300193          1
41.5         3.285319          1
41.2         3.240697          1
53.0         4.995855          1
Length: 99, dtype: int64
```

[80]:
```
sns.boxplot(x=numeric_data['Doc_visits'])
```

[80]: <AxesSubplot:xlabel='Doc_visits'>



[81]:
```
doc_visits_outliers = numeric_data.loc[(numeric_data['Doc_visits_z'] > 3) |
↪(numeric_data['Doc_visits_z'] < -3),
                     ['Doc_visits', 'Doc_visits_z']]
Add_outlier_column(df, doc_visits_outliers, 'Doc_visits')
doc_visits_outliers.sort_values('Doc_visits')
```

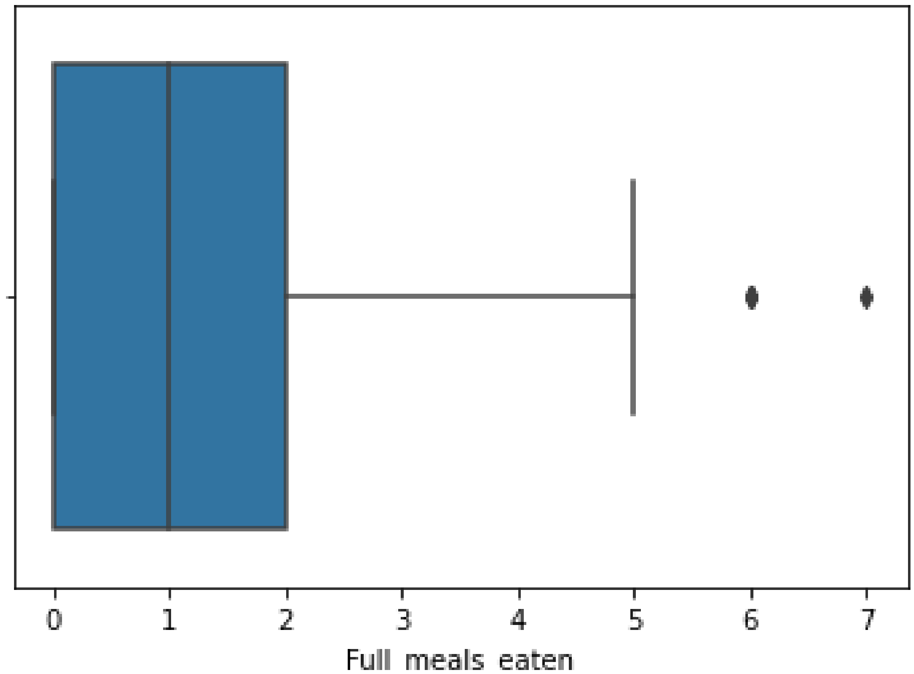
[81]:
```
            Doc_visits  Doc_visits_z
Case_order
5646                 1     -3.836921
5757                 1     -3.836921
6018                 1     -3.836921
6499                 1     -3.836921
6943                 1     -3.836921
7144                 1     -3.836921
963                  9      3.813587
2767                 9      3.813587
```

[82]:
```
doc_visits_outliers.sort_values('Doc_visits').value_counts()
```

[82]:
```
Doc_visits  Doc_visits_z
1           -3.836921       6
9            3.813587       2
dtype: int64
```

[83]:
```
sns.boxplot(x=numeric_data['Full_meals_eaten'])
```

[83]: <AxesSubplot:xlabel='Full_meals_eaten'>



[84]:
```
full_meals_eaten_outliers = numeric_data.loc[
    (numeric_data['Full_meals_eaten_z'] > 3) |␣
↪(numeric_data['Full_meals_eaten_z'] < -3),
                 ['Full_meals_eaten', 'Full_meals_eaten_z']]
Add_outlier_column(df, full_meals_eaten_outliers, 'Full_meals_eaten')
full_meals_eaten_outliers.sort_values('Full_meals_eaten')
```

[84]:

	Full_meals_eaten	Full_meals_eaten_z
Case_order		
551	5	3.966603
9068	5	3.966603
8995	5	3.966603
8903	5	3.966603
8327	5	3.966603
6803	5	3.966603
6695	5	3.966603
6084	5	3.966603
6027	5	3.966603
5860	5	3.966603
5712	5	3.966603
5598	5	3.966603
9221	5	3.966603

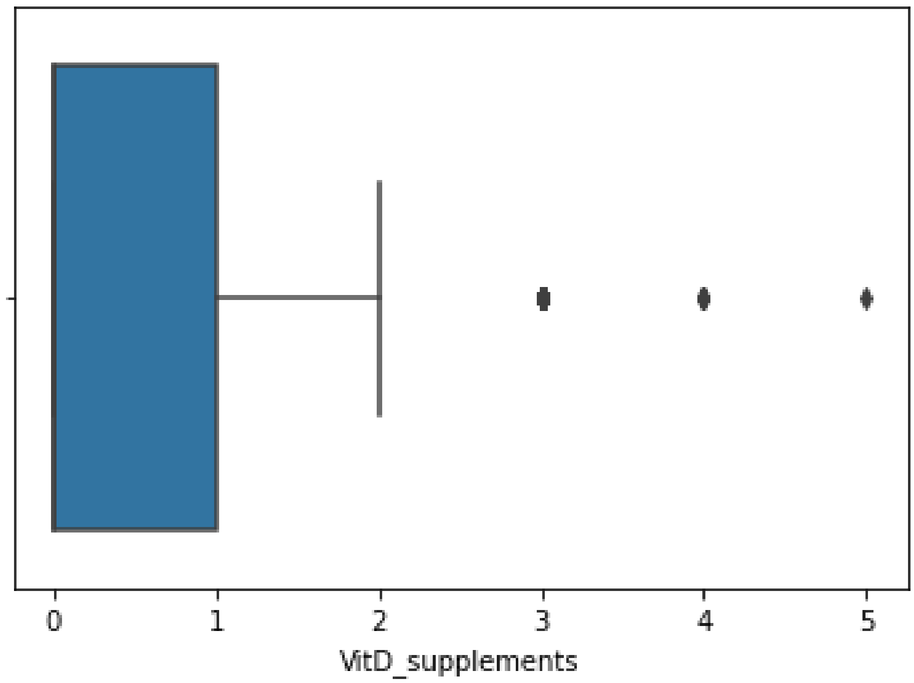
```
5368                    5           3.966603
5544                    5           3.966603
4346                    5           3.966603
2920                    5           3.966603
2878                    5           3.966603
2747                    5           3.966603
2653                    5           3.966603
698                     5           3.966603
2316                    5           3.966603
4903                    5           3.966603
1457                    5           3.966603
1149                    5           3.966603
2185                    6           4.958602
1232                    6           4.958602
9987                    6           4.958602
7218                    6           4.958602
6069                    6           4.958602
8145                    6           4.958602
959                     7           5.950600
4710                    7           5.950600
```

[85]:
```
full_meals_eaten_outliers.sort_values('Full_meals_eaten').value_counts()
```

[85]:
```
Full_meals_eaten  Full_meals_eaten_z
5                 3.966603              25
6                 4.958602               6
7                 5.950600               2
dtype: int64
```

[86]:
```
sns.boxplot(x=numeric_data['VitD_supplements'])
```

[86]:
```
<AxesSubplot:xlabel='VitD_supplements'>
```



[87]:
```
vitD_supplements_outliers = numeric_data.loc[
    (numeric_data['VitD_supplements_z'] > 3) |␣
↪(numeric_data['VitD_supplements_z'] < -3),
                 ['VitD_supplements', 'VitD_supplements_z']]
Add_outlier_column(df, vitD_supplements_outliers, 'VitD_Supplements')
vitD_supplements_outliers.sort_values('VitD_supplements')
```

[87]:

	VitD_supplements	VitD_supplements_z
Case_order		
63	3	4.138759
5000	3	4.138759
5045	3	4.138759
5217	3	4.138759
5352	3	4.138759
…	…	…
1343	4	5.729917
9092	4	5.729917
7181	4	5.729917
2534	4	5.729917
3132	5	7.321074

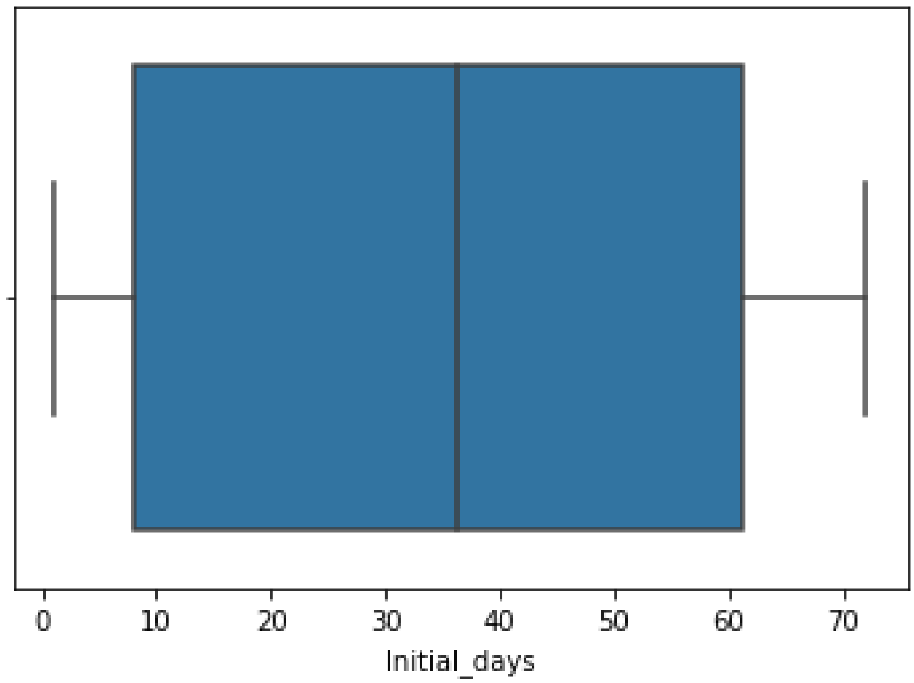
[70 rows x 2 columns]

[88]:
```
vitD_supplements_outliers.sort_values('VitD_supplements').value_counts()
```

[88]:
```
VitD_supplements  VitD_supplements_z
3                 4.138759              64
4                 5.729917               5
5                 7.321074               1
dtype: int64
```

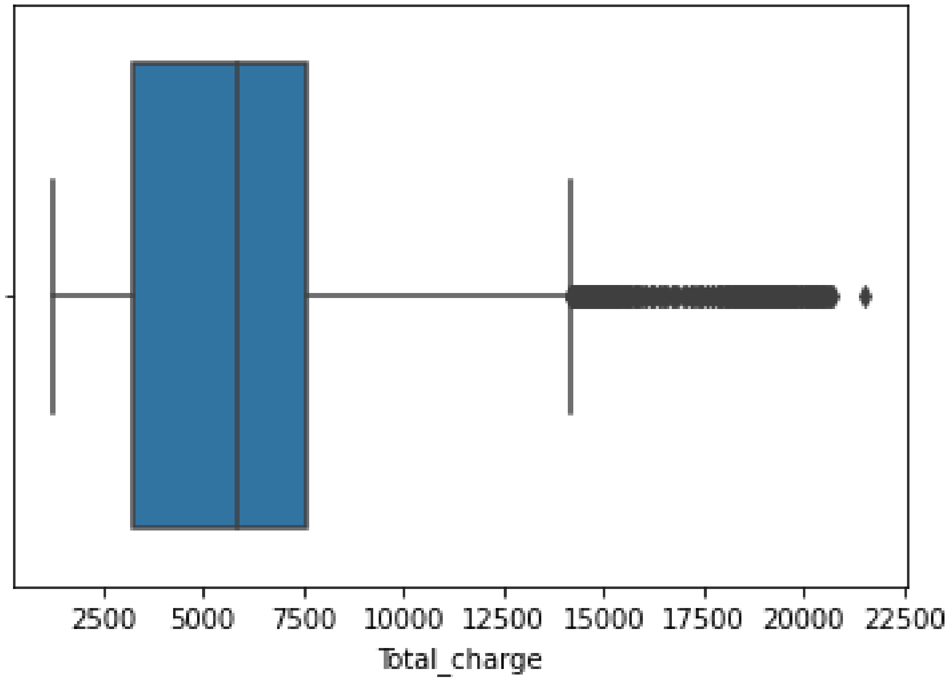
[89]:
```
sns.boxplot(x=numeric_data['Initial_days'])
```

[89]: <AxesSubplot:xlabel='Initial_days'>



[90]:
```
sns.boxplot(x=numeric_data['Total_charge'])
```

[90]: <AxesSubplot:xlabel='Total_charge'>



[91]:
```
total_charge_outliers = numeric_data.loc[(numeric_data['Total_charge_z'] > 3) |␣
↪(numeric_data['Total_charge_z'] < -3),
                    ['Total_charge', 'Total_charge_z']]
Add_outlier_column(df, total_charge_outliers, 'Total_Charge')
total_charge_outliers.sort_values('Total_charge')
```

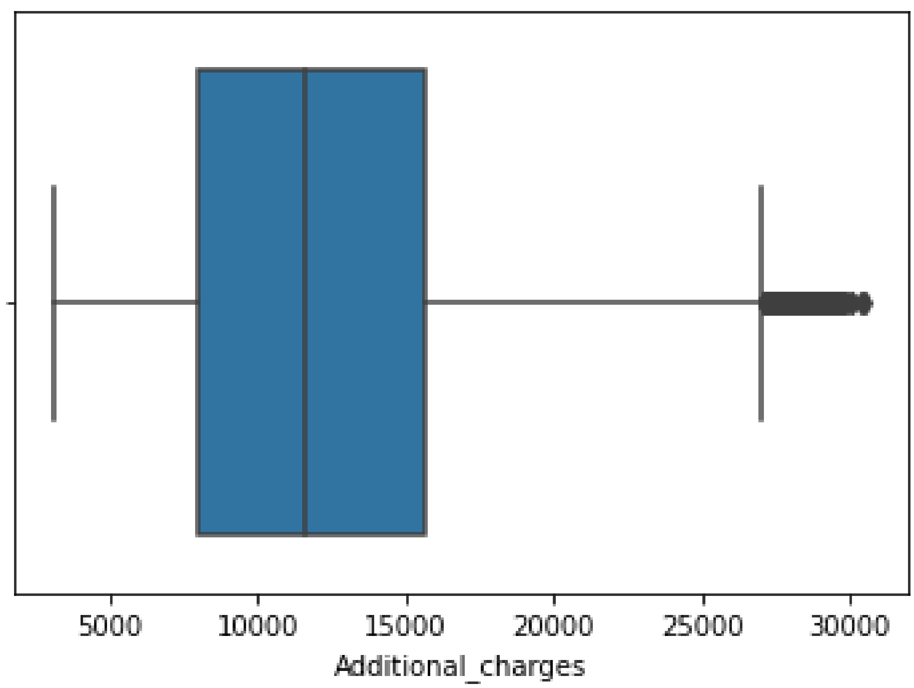
[91]:
```
            Total_charge  Total_charge_z
Case_order
528             16053.46        3.008810
3351            16057.31        3.009950
1848            16153.99        3.038575
3000            16173.62        3.044388
1964            16194.01        3.050425
…                  …               …
9160            20562.04        4.343740
5454            20632.44        4.364585
5245            20647.39        4.369011
9006            20673.97        4.376881
8801            21524.22        4.628629

[276 rows x 2 columns]
```

[92]:
```
total_charge_outliers.sort_values('Total_charge').value_counts()
```

[92]:
```
Total_charge  Total_charge_z
16053.46      3.008810          1
19367.21      3.989967          1
19409.18      4.002394          1
19404.99      4.001153          1
19403.19      4.000620          1
                               ..
18550.12      3.748038          1
18557.70      3.750282          1
18564.13      3.752186          1
18575.97      3.755691          1
21524.22      4.628629          1
Length: 276, dtype: int64
```

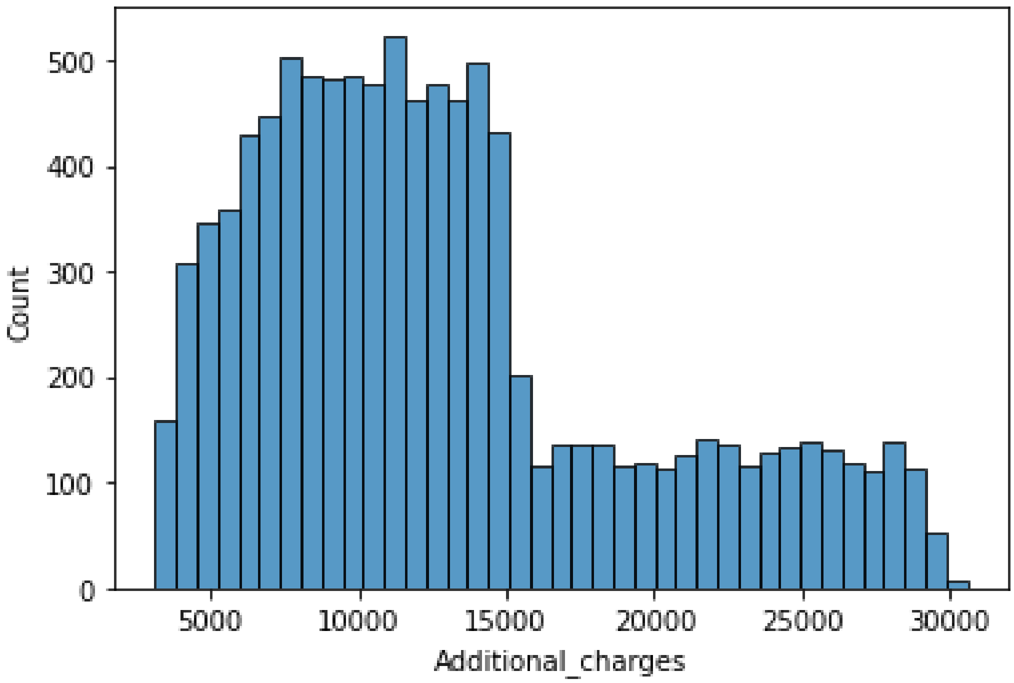
[93]:
```
sns.boxplot(x=numeric_data['Additional_charges'])
```

[93]: <AxesSubplot:xlabel='Additional_charges'>



[94]:
```
sns.histplot(x=numeric_data['Additional_charges'])
```

[94]: <AxesSubplot:xlabel='Additional_charges', ylabel='Count'>



[95]:
```
#iqr used because z score did not accurately capture outliers due to data␣
↪distribution
iqr_a_charges = stats.iqr(numeric_data['Additional_charges'])
q1_a_charges = numeric_data['Additional_charges'].quantile(0.25)
q3_a_charges = numeric_data['Additional_charges'].quantile(0.75)
additional_charges_outliers = numeric_data.loc[
    (numeric_data['Additional_charges'] > (q3_a_charges * 1.5 + iqr_a_charges))␣
↪| (numeric_data['Additional_charges'] < (q1_a_charges * 1.5 -␣
↪iqr_a_charges)),
                 ['Additional_charges']]
Add_outlier_column(df, additional_charges_outliers, 'Additional_charges')
additional_charges_outliers.sort_values('Additional_charges')
```

[95]:

	Additional_charges
Case_order	
6452	3125.70
2415	3132.26
1478	3132.26
4232	3139.05
3515	3139.05
…	…
6465	4327.02
5288	4327.02
7279	4332.13

```
4650                    4334.52
4539                    4337.80

[383 rows x 1 columns]
```

[96]:
```
additional_charges_outliers.sort_values('Additional_charges').value_counts()
```

[96]:
```
Additional_charges
3241.34               4
3585.74               3
3883.66               3
4228.07               3
4129.06               3
                     ..
3771.31               1
3767.15               1
3764.25               1
3760.09               1
4337.80               1
Length: 334, dtype: int64
```

[97]:
```
#Re-expression of catagorical varibles:
#categorical columns that can ony be yes or no will be converted to 1, or 0.
#categorical columns that can be expressed ordinally will have a numerical␣
↪column added with their ordinal value,
#in the format _numeric. --categorical columns that do not fit either of the␣
↪prior categories will not be altered and
#will be retained for use in data analysis as is.
df.columns
```

[97]:
```
Index(['City', 'State', 'County', 'Zip', 'Population', 'Area', 'Timezone',
       'Job', 'Children', 'Age', 'Education', 'Employment', 'Income',
       'Mariage_status', 'Gender', 'Readmited', 'VitD_levels', 'Doc_visits',
       'Full_meals_eaten', 'VitD_supplements', 'Habitual_soft_drink_use',
       'Initial_admin', 'High_blood_pressure', 'Stroke', 'Complication_risk',
       'Overweight', 'Arthritis', 'Diabetes', 'Hyperlipidemia', 'Back_pain',
       'Anxiety', 'Allergic_rhinitis', 'Reflux_esophagitis', 'Asthma',
       'Primary_service_recived', 'Initial_days', 'Total_charge',
       'Additional_charges', 'Survey_timely_addmission',
       'Survey_timely_treatment', 'Survey_timely_visits', 'Survey_reliability',
       'Survey_options', 'Survey_hours', 'Survey_courtesy',
       'Survey_active_listening', 'Population_outliers', 'Children_outliers',
       'Income_outliers', 'VitD_levels_outliers', 'Doc_visits_outliers',
       'Full_meals_eaten_outliers', 'VitD_Supplements_outliers',
       'Total_Charge_outliers', 'Additional_charges_outliers'],
      dtype='object')
```

[98]:
```
df.loc[:,['Readmited', 'Habitual_soft_drink_use', 'High_blood_pressure',␣
↪'Stroke', 'Arthritis','Diabetes',
        'Hyperlipidemia', 'Back_pain', 'Allergic_rhinitis',␣
↪'Reflux_esophagitis', 'Asthma']].replace({'Yes': 1, 'No': 0})
```

[98]:
```
            Readmited  Habitual_soft_drink_use  High_blood_pressure  Stroke  \
Case_order
1                   0                        0                    1       0
2                   0                        0                    1       0
3                   0                        0                    1       0
4                   0                        0                    0       1
5                   0                        1                    0       0
…                 …                      …                  …     …
9996                0                        0                    1       0
9997                1                        0                    1       0
9998                1                        1                    1       0
9999                1                        0                    0       0
10000               1                        0                    0       0

            Arthritis  Diabetes  Hyperlipidemia  Back_pain  Allergic_rhinitis  \
Case_order
1                   1         1               0          1                  1
2                   0         0               0          0                  0
3                   0         1               0          0                  0
4                   1         0               0          0                  0
5                   0         0               1          0                  1
…                 …       …            …        …                …
9996                0         0               0          0                  0
9997                1         1               0          0                  0
9998                0         0               0          0                  1
9999                0         0               0          1                  0
10000               1         0               1          0                  1

            Reflux_esophagitis  Asthma
Case_order
1                            0       1
2                            1       0
3                            0       0
4                            1       1
5                            0       0
…                          …     …
9996                         1       0
9997                         0       1
9998                         0       0
9999                         0       0
10000                        0       0
```

```
[10000 rows x 11 columns]
```

[99]:
```
education_dict = {'Some College, Less than 1 Year': 5,
        'Some College, 1 or More Years, No Degree': 6,
        'GED or Alternative Credential': 3, 'Regular High School Diploma': 4,
        "Bachelor's Degree": 9, "Master's Degree": 10,
        'Nursery School to 8th Grade': 1,
        '9th Grade to 12th Grade, No Diploma': 2, 'Doctorate Degree': 11,
        "Associate's Degree": 8, 'Professional School Degree': 7,
        'No Schooling Completed': 0}
df['Education_numeric'] = df['Education'].replace(education_dict)
df['Education_numeric'].value_counts()
```

[99]:
```
4     2444
9     1724
6     1484
2      832
8      797
10     701
5      642
1      552
3      389
7      208
0      133
11      94
Name: Education_numeric, dtype: int64
```

[100]:
```
complication_dict = {'Low': 0, 'Medium': 1, 'High': 2}
df['Complication_risk_numeric'] = df['Complication_risk'].
 ↪replace(complication_dict)
df['Complication_risk_numeric'].value_counts()
```

[100]:
```
1    4517
2    3358
0    2125
Name: Complication_risk_numeric, dtype: int64
```

Principal component Analysis

[101]:
```
df_pca = df[["Population", "Age", "Income", "VitD_levels", "Initial_days",␣
 ↪"Total_charge", "Additional_charges", "Survey_timely_addmission",
             "Survey_timely_treatment", "Survey_timely_visits",␣
 ↪"Survey_reliability", "Survey_options", "Survey_hours",
             "Survey_courtesy", "Survey_active_listening",␣
 ↪"Complication_risk_numeric"]]
```

[102]:
```
df_pca
```

[102]:
```
            Population  Age    Income  VitD_levels  Initial_days  \
Case_order
1                 2951   53  86575.93         17.8          10.6
2                11303   51  46805.99         19.0          15.1
3                17125   53  14370.14         17.4           4.8
4                 2162   78  39741.49         17.4           1.7
5                 5287   22   1209.56         16.9           1.3
…                  …   …       …           …            …
9996              4762   25  45967.61         16.5          51.6
9997              1251   87  14983.02         18.5          68.7
9998               532   65  65917.81         15.8          66.0
9999               271   43  29702.32         22.0          63.4
10000            41524   43  62682.63         20.4          70.9

            Total_charge  Additional_charges  Survey_timely_addmission  \
Case_order
1                3191.05            17939.40                         3
2                4214.91            17613.00                         3
3                2177.59            17505.19                         2
4                2465.12            12993.44                         3
5                1885.66             3716.53                         2
…                    …                 …                       …
9996             6651.24             8927.64                         3
9997             7851.52            28507.15                         3
9998             7725.95            15281.21                         3
9999             8462.83             7781.68                         5
10000            8700.86            11643.19                         4

            Survey_timely_treatment  Survey_timely_visits  Survey_reliability  \
Case_order
1                                 3                     2                   2
2                                 4                     3                   4
3                                 4                     4                   4
4                                 5                     5                   3
5                                 1                     3                   3
…                             …                   …                  …
9996                              2                     2                   3
9997                              3                     4                   2
9998                              3                     3                   4
9999                              5                     3                   4
10000                             3                     3                   2

            Survey_options  Survey_hours  Survey_courtesy  \
Case_order
1                        4             3                3
2                        4             4                3
3                        3             4                3
```

```
4                         4           5             5
5                         5           3             4
…                    …         …           …
9996                      4           3             4
9997                      5           3             4
9998                      4           2             3
9999                      4           3             4
10000                     3           6             4

            Survey_active_listening  Complication_risk_numeric
Case_order
1                                 4                          1
2                                 3                          2
3                                 3                          1
4                                 5                          1
5                                 3                          0
…                              …                          …
9996                              2                          1
9997                              4                          1
9998                              2                          2
9999                              3                          1
10000                             3                          0

[10000 rows x 16 columns]
```

[103]:
```
df_pca_normalized = (df_pca - df_pca.mean())/df_pca.std()
df_pca_normalized
```

[103]:
```
            Population       Age    Income  VitD_levels  Initial_days  \
Case_order
1            -0.473145 -0.014418  1.709047    -0.239860     -0.908604
2             0.090237 -0.117331  0.233157    -0.061378     -0.737273
3             0.482959 -0.014418 -0.970559    -0.299354     -1.129431
4            -0.526366  1.271992 -0.029011    -0.299354     -1.247460
5            -0.315570 -1.609567 -1.458957    -0.373722     -1.262689
…                 …       …       …          …           …
9996         -0.350984 -1.455198  0.202045    -0.433215      0.652414
9997         -0.587818  1.735100 -0.947814    -0.135746      1.303474
9998         -0.636318  0.603059  0.942410    -0.537330      1.200675
9999         -0.653923 -0.528982 -0.401571     0.384826      1.101683
10000         2.128787 -0.528982  0.822350     0.146850      1.387236

            Total_charge  Additional_charges  Survey_timely_addmission  \
Case_order
1              -0.799539            0.764967                 -0.502730
2              -0.496402            0.715078                 -0.502730
3              -1.099596            0.698600                 -1.471754
```

```
4            -1.014466           0.009004                   -0.502730
5            -1.186028          -1.408919                   -1.471754
…                  …                 …                       …
9996          0.224926          -0.612430                   -0.502730
9997          0.580295           2.380188                   -0.502730
9998          0.543118           0.358677                   -0.502730
9999          0.761287          -0.787584                    1.435319
10000         0.831761          -0.197374                    0.466295

            Survey_timely_treatment  Survey_timely_visits  Survey_reliability  \
Case_order
1                         -0.489648             -1.463173           -1.462054
2                          0.476699             -0.494890            0.467923
3                          0.476699              0.473394            0.467923
4                          1.443046              1.441677           -0.497066
5                         -2.422343             -0.494890           -0.497066
…                             …                     …                   …
9996                      -1.455995             -1.463173           -0.497066
9997                      -0.489648              0.473394           -1.462054
9998                      -0.489648             -0.494890            0.467923
9999                       1.443046             -0.494890            0.467923
10000                     -0.489648             -0.494890           -1.462054

            Survey_options  Survey_hours  Survey_courtesy  \
Case_order
1                 0.488355     -0.506114        -0.483647
2                 0.488355      0.462525        -0.483647
3                -0.482337      0.462525        -0.483647
4                 0.488355      1.431165         1.474440
5                 1.459048     -0.506114         0.495396
…                    …             …               …
9996              0.488355     -0.506114         0.495396
9997              1.459048     -0.506114         0.495396
9998              0.488355     -1.474753        -0.483647
9999              0.488355     -0.506114         0.495396
10000            -0.482337      2.399804         0.495396

            Survey_active_listening  Complication_risk_numeric
Case_order
1                          0.470397                  -0.168864
2                         -0.489009                   1.200677
3                         -0.489009                  -0.168864
4                          1.429802                  -0.168864
5                         -0.489009                  -1.538406
…                             …                          …
9996                      -1.448415                  -0.168864
9997                       0.470397                  -0.168864
```

```
9998                       -1.448415                  1.200677
9999                       -0.489009                 -0.168864
10000                      -0.489009                 -1.538406

[10000 rows x 16 columns]
```

[104]:
```python
component_number = df_pca.shape[1]
pca = PCA(n_components = component_number)
pca = pca.fit(df_pca_normalized)
```

[105]:
```python
pca_columns = []
for i in range(1, component_number + 1):
    pca_columns.append("PC" + str(i))
pca_columns
```

[105]:
```
['PC1',
 'PC2',
 'PC3',
 'PC4',
 'PC5',
 'PC6',
 'PC7',
 'PC8',
 'PC9',
 'PC10',
 'PC11',
 'PC12',
 'PC13',
 'PC14',
 'PC15',
 'PC16']
```

[106]:
```python
df_pca_components = pd.DataFrame(pca.transform(df_pca_normalized), columns =␣
↪pca_columns)
df_pca_components
```

[106]:
```
           PC1       PC2       PC3       PC4       PC5       PC6       PC7  \
0    -1.535728 -1.166059  0.247782  0.683519  0.842751 -1.329184  1.124139
1    -0.335370 -0.645760 -0.176366  0.554742  1.185733  0.038088 -0.526781
2    -0.202643 -1.323492 -0.761551  0.580741 -0.208920  0.496684 -0.945738
3     2.386521 -1.336477  0.317918  1.078790 -0.130114 -0.780522 -0.300955
4    -2.421516 -1.890243 -0.120102 -1.987082 -1.534425 -0.272241 -1.053780
…          …         …         …         …         …         …         …
9995 -2.103208  0.039659 -0.107729 -1.471427 -0.350066  0.073672  0.433191
9996 -0.666041  1.112658  1.413302  2.862242 -1.218419  0.399539 -0.068469
9997 -1.901673  0.661713  0.156946  0.694169  0.628890  0.075790  0.687979
9998  0.820920  1.068572  0.986507 -0.988237 -0.756588 -0.093300 -0.144634
```

```
9999  0.644805  1.241895  0.236817 -0.812903 -0.393779  1.335029  2.130521

           PC8       PC9      PC10      PC11      PC12      PC13      PC14  \
0     0.140740  1.309369  0.462301  0.591729  0.004999  0.734226  0.400176
1    -0.262993 -0.475043 -0.261596  0.497443  0.839671  0.335063  0.525564
2     0.772752 -0.622160 -0.477832  0.389151 -0.400082 -0.200802  0.482335
3     0.286811  1.030263  0.684113  0.076642  0.219009 -0.839837 -0.831675
4     1.074370  0.686573  1.348988 -0.281819  0.436187 -1.667385  0.511826
…          …         …         …         …         …         …         …
9995 -0.604823 -0.437996  1.562367  0.228524  0.359812  0.078645  0.601202
9996 -0.821966  1.278966  0.811684 -0.283100 -0.097498 -0.809422  0.570105
9997 -1.800319 -1.171954 -0.042065 -0.841923  0.280861  0.192810 -0.191969
9998 -0.592584 -0.739789  0.170496 -0.752657  0.908960  1.421993 -0.232237
9999  1.485442 -0.023174  1.269843  2.537787 -0.206764 -0.342090  0.169580

          PC15      PC16
0    -0.128484  0.021966
1     0.658885  0.026241
2     1.523072  0.104248
3     1.799686 -0.019168
4    -0.103280  0.000746
…          …         …
9995 -0.688118 -0.093963
9996  0.356480  0.144493
9997 -0.189085 -0.068527
9998 -0.506994  0.153526
9999 -0.492005  0.072410

[10000 rows x 16 columns]
```

[107]:
```python
loadings = pd.DataFrame(pca.components_.T, columns=pca_columns,
 index=df_pca_normalized.columns)
loadings
```

[107]:
```
                               PC1       PC2       PC3       PC4       PC5  \
Population                0.010362  0.016688  0.027855 -0.025958  0.325334
Age                       0.000054  0.073848 -0.030625  0.700266 -0.061654
Income                    0.000334 -0.020439 -0.024192  0.000398  0.494925
VitD_levels              -0.009115  0.527040  0.037250 -0.044069  0.223165
Initial_days             -0.018098  0.459401  0.070006 -0.066090 -0.296909
Total_charge             -0.017624  0.698430  0.070213 -0.067872  0.008938
Additional_charges        0.004131  0.077414 -0.039310  0.701221  0.011142
Survey_timely_addmission  0.454789 -0.019999  0.295228  0.017013  0.007306
Survey_timely_treatment   0.428522 -0.021550  0.291840  0.018835 -0.007487
Survey_timely_visits      0.395365 -0.020168  0.294343  0.015325  0.008897
Survey_reliability        0.152140  0.052287 -0.554488 -0.032091 -0.020364
Survey_options           -0.189950 -0.059193  0.579799  0.036944  0.006518
```

```
Survey_hours                0.410155  0.027291 -0.160879 -0.021724 -0.029129
Survey_courtesy             0.356499  0.034505 -0.170499 -0.002100 -0.021842
Survey_active_listening     0.312522  0.026007 -0.164878 -0.020334  0.015070
Complication_risk_numeric   0.012619  0.033334 -0.014466  0.046494  0.710666

                                 PC6       PC7       PC8       PC9      PC10  \
Population                  0.734726  0.201383  0.556621  0.007598 -0.024594
Age                         0.000039  0.030904  0.040969 -0.004317 -0.031211
Income                     -0.405712  0.762165 -0.025919 -0.084584  0.013183
VitD_levels                -0.347973 -0.302589  0.418793 -0.001421  0.010890
Initial_days                0.354795  0.377959 -0.452637 -0.011623 -0.023534
Total_charge               -0.017913  0.007562 -0.011043 -0.004001 -0.004839
Additional_charges          0.025581  0.019338 -0.000349  0.013020  0.014352
Survey_timely_addmission   -0.016526 -0.017296  0.005324 -0.096276 -0.074313
Survey_timely_treatment     0.004690  0.008752  0.007253 -0.148863 -0.133525
Survey_timely_visits       -0.023331 -0.028293 -0.040239 -0.207440 -0.209338
Survey_reliability          0.030478 -0.035181  0.006032 -0.367337 -0.365055
Survey_options             -0.018168 -0.002818  0.015054  0.124991  0.051941
Survey_hours                0.000024  0.004948  0.027602 -0.049685  0.063436
Survey_courtesy             0.033294  0.012943 -0.002692  0.044978  0.843262
Survey_active_listening    -0.039992  0.063749 -0.004489  0.873193 -0.281002
Complication_risk_numeric   0.206952 -0.368880 -0.551979  0.021665  0.011472

                                PC11      PC12      PC13      PC14      PC15  \
Population                 -0.007395 -0.022886 -0.027729 -0.009084 -0.005078
Age                        -0.027686  0.062475 -0.104608 -0.692641 -0.019669
Income                     -0.005484  0.017972 -0.009730 -0.008998 -0.003509
VitD_levels                -0.006858 -0.007904  0.007215 -0.006696  0.008693
Initial_days                0.009500  0.010495 -0.002204 -0.006945 -0.018363
Total_charge               -0.004081 -0.000876  0.000982  0.009461  0.004228
Additional_charges          0.041256 -0.065053  0.096759  0.695243  0.018654
Survey_timely_addmission   -0.010891  0.080368  0.188747  0.006992 -0.804722
Survey_timely_treatment    -0.062053  0.087067  0.621009 -0.086326  0.534459
Survey_timely_visits       -0.230675 -0.425830 -0.625098  0.066371  0.191819
Survey_reliability         -0.394853  0.471410 -0.092470  0.089542 -0.010276
Survey_options             -0.146266  0.694343 -0.281451  0.110732  0.094717
Survey_hours                0.789013  0.289144 -0.272023  0.022057  0.126143
Survey_courtesy            -0.341578  0.069325 -0.060711  0.000798  0.050877
Survey_active_listening    -0.151072  0.036770 -0.035601  0.011725  0.033537
Complication_risk_numeric   0.027748  0.037042  0.005778 -0.061065  0.008677

                                PC16
Population                  0.000447
Age                        -0.007604
Income                      0.002691
VitD_levels                 0.529695
Initial_days                0.465088
```

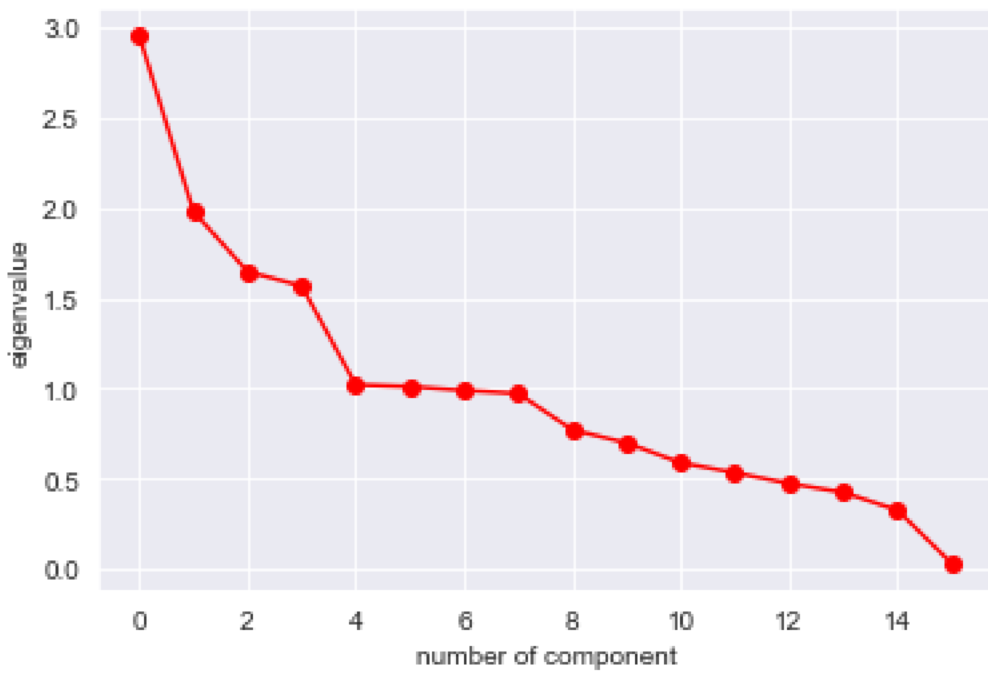
```
Total_charge                 -0.708239
Additional_charges            0.014052
Survey_timely_addmission     -0.006593
Survey_timely_treatment      -0.000119
Survey_timely_visits          0.004634
Survey_reliability            0.001653
Survey_options                0.001148
Survey_hours                 -0.002293
Survey_courtesy               0.003613
Survey_active_listening       0.003545
Complication_risk_numeric     0.033908
```

[108]:

```
cov_matrix = np.dot(df_pca_normalized.T, df_pca_normalized)/ df_pca.shape[0]
eigenvalues = [np.dot(eigenvector.T, np.dot(cov_matrix, eigenvector)) for␣
 ↪eigenvector in pca.components_]
```

[109]:

```
sns.set_style('darkgrid')
plt.plot(eigenvalues, 'ro-')
plt.xlabel("number of component")
plt.ylabel("eigenvalue")
plt.show()
```



[110]:

```
eigenvalues
```

[110]:
```
[2.9540878129785444,
 1.9815118980902646,
 1.647893356882892,
 1.567595662569451,
 1.0213461108725723,
 1.0112206936978132,
 0.9896952485605882,
 0.9727728788216048,
 0.7716036953231841,
 0.6989009760367672,
 0.5888498667484408,
 0.5339195750082404,
 0.4742183039682881,
 0.4248729760107605,
 0.327251753689872,
 0.03265919074060882]
```

[111]:
```python
eigen_count = 0
for x in eigenvalues:
    if x >= 1:
        eigen_count +=1
eigen_count
```

[111]: 6

[112]:
```python
df_reduced = df_pca_components.iloc[:,:eigen_count]
df_reduced
```

[112]:
```
           PC1       PC2       PC3       PC4       PC5       PC6
0    -1.535728 -1.166059  0.247782  0.683519  0.842751 -1.329184
1    -0.335370 -0.645760 -0.176366  0.554742  1.185733  0.038088
2    -0.202643 -1.323492 -0.761551  0.580741 -0.208920  0.496684
3     2.386521 -1.336477  0.317918  1.078790 -0.130114 -0.780522
4    -2.421516 -1.890243 -0.120102 -1.987082 -1.534425 -0.272241
…          …         …         …         …         …         …
9995 -2.103208  0.039659 -0.107729 -1.471427 -0.350066  0.073672
9996 -0.666041  1.112658  1.413302  2.862242 -1.218419  0.399539
9997 -1.901673  0.661713  0.156946  0.694169  0.628890  0.075790
9998  0.820920  1.068572  0.986507 -0.988237 -0.756588 -0.093300
9999  0.644805  1.241895  0.236817 -0.812903 -0.393779  1.335029

[10000 rows x 6 columns]
```

[113]:
```python
loadings.iloc[:, :eigen_count]
```

[113]:
```
                                  PC1       PC2       PC3       PC4       PC5  \
Population                   0.010362  0.016688  0.027855 -0.025958  0.325334
Age                          0.000054  0.073848 -0.030625  0.700266 -0.061654
```

```
Income                      0.000334 -0.020439 -0.024192  0.000398  0.494925
VitD_levels                -0.009115  0.527040  0.037250 -0.044069  0.223165
Initial_days               -0.018098  0.459401  0.070006 -0.066090 -0.296909
Total_charge               -0.017624  0.698430  0.070213 -0.067872  0.008938
Additional_charges          0.004131  0.077414 -0.039310  0.701221  0.011142
Survey_timely_addmission    0.454789 -0.019999  0.295228  0.017013  0.007306
Survey_timely_treatment     0.428522 -0.021550  0.291840  0.018835 -0.007487
Survey_timely_visits        0.395365 -0.020168  0.294343  0.015325  0.008897
Survey_reliability          0.152140  0.052287 -0.554488 -0.032091 -0.020364
Survey_options             -0.189950 -0.059193  0.579799  0.036944  0.006518
Survey_hours                0.410155  0.027291 -0.160879 -0.021724 -0.029129
Survey_courtesy             0.356499  0.034505 -0.170499 -0.002100 -0.021842
Survey_active_listening     0.312522  0.026007 -0.164878 -0.020334  0.015070
Complication_risk_numeric   0.012619  0.033334 -0.014466  0.046494  0.710666

                                 PC6
Population                  0.734726
Age                         0.000039
Income                     -0.405712
VitD_levels                -0.347973
Initial_days                0.354795
Total_charge               -0.017913
Additional_charges          0.025581
Survey_timely_addmission   -0.016526
Survey_timely_treatment     0.004690
Survey_timely_visits       -0.023331
Survey_reliability          0.030478
Survey_options             -0.018168
Survey_hours                0.000024
Survey_courtesy             0.033294
Survey_active_listening    -0.039992
Complication_risk_numeric   0.206952
```

[114]:
```
#df.to_csv('medical_data_cleaned.csv')
```

2 Part III: Data Cleaning

2.1 D. Summarize of data-cleaning process:

1. Findings:

   - During the analysis I determined that there were 6 variables that are were redundant or unnecessary for analysis. Several variables had names that were not sufficiently descriptive, or deviated in naming standards from the majority of data in the datasets.
   - 7 columns were found to contain missing values.
   - The time zone column contained 26 unique values, many of which are functionally identical for describing the time zone of the row, and any additional information these further granularity would provide were redundant. 3 columns that contained values

that could only logically be expressed as whole integers, had a variable type of floating point number. - The zip-code column's data type was set as a numerical integer, when it could be more properly be expressed as a string for categorization and analytic purposes. further analysis of the zip column revealed that there were several invalid fields caused by a data entry error.

- Total_charge and Additional_charge columns where based on averages and contained a much higher degree of precision than typical expression of monetary values. Initial days column contained a high degree of precision, that prevented meaningful grouping of its values.
- 11 numerical columns were selected for outlier detection, of these 9 were found to contain outlier. However, none of these outliers were outside of the logical range that their perspective value could contain.
- 11 columns were found to contain only yes or no values, and 2 columns were found to contain categories that could be expressed as a continuous variable.

2. Justification and Implementation summary:

   - Columns that were found to be redundant were dropped from the dataset, except for case_order which was set as the index of the data frame since it was functionally identical and more descriptive than index. Columns that had insufficiently descriptive names and non standard named were renamed and standardized.
   - Imputation of categorical variables that was handled by replacing null values with the mode for the column as suggested by Data Science: Using Python and R.(Larose, C. D. & Larose, D. T. (2019)). For numerical columns with missing data, nulls were replaced with the mean of the data set. except in cases where comparing before and after histograms revealed that imputation had skewed the dataset, where median or interpolation were used instead to maintain data distribution.
   - The time zone column contained 27 unique values several of which mapped to standard UTC time zones. I replaced these values because it created a smaller number of groupings for analysis, and any additional information that would be provided by more granular time zone definitions, is already provided by state, city, and zip code columns. Columns that contained numerical data that should only logically consist of whole numbers(number of children, etc;) that were incorrectly typed as precision floating point numbers where converted to integers.
   - The zip code column was initially typed as a numerical column, but was converted to a string to make it easier to group as a categorical variable. after converting to a string validation was preformed and I discovered several items in this column that were invalid zip codes. I isolated these values and the state and city they belonged to, and by comparing them with a database of US zip codes determined that they were all valid zip codes whose leading zeros had been removed when they were cast as a numeric data type.
   - The total_charge and additional_charge columns were based on calculations of averages, and had a much higher precision than the standard representation of a monetary value, so they were rounded to two decimal places. The initial days column also contained a high degree of precision, and was rounded to tenths of a day to form more consistent groupings of values.
   - I identified 11 numerical data columns that could be checked for outliers, of these the age and initial_days columns contained no outliers. For the others outliers were identified

and isolated using a combination of box plots and z scores, Where needed histograms were used for further analysis. In the case of the additional_charges column z scores were not suitable for outlier isolation due to data distribution, so iqr was used instead. Outliers are stored in a separate variable named _outliers, but not removed from the original dataset. This is done so that analysis can be performed on dataset both including and excluding outliers, because while outliers are present for each column, they are not abnormal values for the data type. I wrote a helper function to add boolean outlier column to main dataframe for each specific column, this function can be used during later data analysis to easily include or exclude outliers from analysis. also, during outlier detection I discovered that the vitD_levels column contained a much higher precision than is used in typical medical studies, which I have referenced in the sources section of this document, so I rounded the values in this column to 1 decimal point to create fewer groupings of values.
- Columns that contain a yes or no value were re-expressed as numeric columns with a value of 0 for no and 1 for yes, while categorical columns that could be expressed as an ordinal numeric value had their ordinal value added as an additional column with the name _numeric. This was done to allow statistical methods that only operate on numeric data such as PCA to be used on these columns.

3. Code used to clean Data is provided in the above sections of this document.
4. Cleaned Data is attached as file named medical_data_cleaned.csv
5. Limitations:
   - The dataset does not provide the reason for initial hospitalization.
   - Some data provided may be more meaningful to someone with a more through medical understanding then me.
   - Job, Insurance, or marital status columns can refer to the primary insurance holder rather than the patient themselves.
6. Effect of Limitations:
   - Without knowledge of initial reason for hospitalization, it cannot be determined if the reason for re-hospitalization is caused by factors provided, or factors resulting from a chronic condition that caused complications, or will require multiple hospitalizations due to being a chronic illness.
   - Since I am not a medical professional this introduces my own biases into the data, and I might have noticed correlations, that a person with more through domain knowledge would have.
   - Some varibles can refer to the primary insurance holder rather than the actual patient, so these factors may not accurately provide data that correlates to the patients likelihood of readmission.

2.2 E. Apply principal component analysis (PCA) to identify the significant features of the data set by

doing the following: 1. Principal components - VitD_levels - Initial_days - Total_charge - Survey_timely_addmission - Survey_timely_treatment - Survey_timely_visits 2. Process used to identify Principal Components: - Numerical data was isolated from the dataframe. - Data was standardized. - PCA was preformed using the scikit learn library. - Results were graphed in a scree plot to determine the cut off for component variance. - Because the results seamed to plateau, eigen values were used to cut off any component with an eigen value of less than 1. - Amount of Variance of each input value in each component was analyzed. 3. Benefits: PCA analysis identified the numerical columns that cause the most varience in the dataset, these items can be used in future data analysis to determine which categorical varibles correlate the most strongly with the Principal components. PCA analysis also allows for reduction of number of varibles, making future analysis more efficient.

3 Part IV. Supporting Documents

3.1 F. panopto link - https://wgu.hosted.panopto.com/Panopto/Pages/Viewer.aspx?id=34d 6fee-49a3-bac7-adcd003c30fe

3.2 Sources

3.2.1 Web

"Installing Python Packages from a Jupyter Notebook" Pythonic Perambulations, 05 December 2017 https://jakevdp.github.io/blog/2017/12/05/installing-python-packages-from-jupyter/

"Vitamin D numbers: what they really mean" Quest Diagnostics, accessed 01 September 2021 https://www.questdiagnostics.com/home/physicians/testing-services/condition/endocrinology/what-low-vitamin-d-numbers-mean/

"Vitamin D: Fact Sheet for Health Professionals" NIH, 17 August 2021 https://ods.od.nih.gov/factsheets/Vitamin%20D-HealthProfessional/

Monique Tello, MD, MPH "Vitamin D: What's the 'right' level?" Harvard Health Blog, 16 April 2020 https://www.health.harvard.edu/blog/vitamin-d-whats-right-level-2016121910893

"How to use Pandas filter with IQR?" Geeks for Geeks, 22 June 2021 https://www.geeksforgeeks.org/how-to-use-pandas-filter-with-iqr/

the pandas development team "pandas.DataFrame.replace" Pandas Documentation, accessed 05 September 2021 https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.replace.html

"Discrete vs Continuous variables: How to Tell the Difference" Statistics How To, accessed 15 August 2021 "https://www.statisticshowto.com/probability-and-statistics/statistics-definitions/discrete-vs-continuous-variables/

3.2.2 Text

Larose, C. D. & Larose, D. T. (2019). Data Science: Using Python and R. John Wiley & Sons, Inc.