Incremental_Capstone_3_EP

April 11, 2024

[3]:

```python
import numpy as np
import pandas as pd
import json

aura_df = pd.read_csv("NSMES1988.csv")
aura_df
```

[3]:

```
      Unnamed: 0  visits  nvisits  ovisits  novisits  emergency  hospital  \
0              1       5        0        0         0          0         1
1              2       1        0        2         0          2         0
2              3      13        0        0         0          3         3
3              4      16        0        5         0          1         1
4              5       3        0        0         0          0         0
…          …     …      …      …       …        …       …
4401        4402      11        0        0         0          0         0
4402        4403      12        0        0         0          0         0
4403        4404      10        0       20         0          1         1
4404        4405      16        1        0         0          0         0
4405        4406       0        0        0         0          0         0

         health  chronic      adl region  age afam  gender married  school  \
0       average        2   normal  other  6.9  yes    male     yes       6
1       average        2   normal  other  7.4   no  female     yes      10
2          poor        4  limited  other  6.6  yes  female      no      10
3          poor        2  limited  other  7.6   no    male     yes       3
4       average        2  limited  other  7.9   no  female     yes       6
…         …      …      …    …  …  …     …     …     …
4401    average        0   normal  other  8.4   no  female     yes       8
4402    average        2   normal  other  7.8   no  female      no      11
4403    average        5   normal  other  7.3   no    male     yes      12
4404    average        0   normal  other  6.6   no  female     yes      12
4405  excellent        0   normal  other  7.1   no    male     yes       0

        income employed insurance medicaid
0     2.881000      yes       yes       no
1     2.747800       no       yes       no
2     0.653200       no        no      yes
3     0.658800       no       yes       no
```

```
4     0.658800       no       yes       no
…        …       …        …       …
4401  2.249700       no       yes       no
4402  5.813200       no       yes       no
4403  3.877916       no       yes       no
4404  3.877916       no       yes       no
4405  6.596800      yes        no       no

[4406 rows x 20 columns]
```

Aura DataFrame Info

```
[4]: aura_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4406 entries, 0 to 4405
Data columns (total 20 columns):
 #   Column      Non-Null Count  Dtype  
---  ------      --------------  -----  
 0   Unnamed: 0  4406 non-null   int64  
 1   visits      4406 non-null   int64  
 2   nvisits     4406 non-null   int64  
 3   ovisits     4406 non-null   int64  
 4   novisits    4406 non-null   int64  
 5   emergency   4406 non-null   int64  
 6   hospital    4406 non-null   int64  
 7   health      4406 non-null   object 
 8   chronic     4406 non-null   int64  
 9   adl         4406 non-null   object 
 10  region      4406 non-null   object 
 11  age         4406 non-null   float64
 12  afam        4406 non-null   object 
 13  gender      4406 non-null   object 
 14  married     4406 non-null   object 
 15  school      4406 non-null   int64  
 16  income      4406 non-null   float64
 17  employed    4406 non-null   object 
 18  insurance   4406 non-null   object 
 19  medicaid    4406 non-null   object 
dtypes: float64(2), int64(9), object(9)
memory usage: 688.6+ KB
```

Aura DataFrame data types:

```
[5]: aura_df.dtypes
```

```
[5]: Unnamed: 0      int64
     visits          int64
     nvisits         int64
```

```
ovisits          int64
novisits         int64
emergency        int64
hospital         int64
health          object
chronic          int64
adl             object
region          object
age            float64
afam            object
gender          object
married         object
school           int64
income         float64
employed        object
insurance       object
medicaid        object
dtype: object
```

[6]:

```
# unnamed 0 column (corrupt data - should be dropped from dataframe)
# health, region, afam, gender, married data type is object (corrupt data - should be category)
# employed, insurance, medicaid data type is object (corrupt data - should be boolean)
```

Aura DataFrame Description

[7]:

```
aura_df.age.describe()
```

[7]:

```
count    4406.000000
mean        7.402406
std         0.633405
min         6.600000
25%         6.900000
50%         7.300000
75%         7.800000
max        10.900000
Name: age, dtype: float64
```

[8]:

```
# age range from 6.6 to 10.9 (corrupt data - should be 66 - 109)
# age data type is float64 (corrupt data - should be uint8)
```

[9]:

```
aura_df.income.describe()
```

[9]:

```
count    4406.000000
mean        2.527132
std         2.924648
min        -1.012500
```

```
25%          0.912150
50%          1.698150
75%          3.172850
max         54.835100
Name: income, dtype: float64
```

[10]:

```
# income range from -1.01 to 54.84 (corrupt data - should be positive value)
# income data type is float64 (corrupt data - should be float16)
```

[11]:

```
aura_df[aura_df.income < 0]
```

[11]:

```
       Unnamed: 0  visits  nvisits  ovisits  novisits  emergency  hospital  \
909           910      10        0        0         0          0         0
910           911       9        2        0         0          0         0
2592         2593       6        0        0         0          0         0

       health  chronic     adl region  age afam  gender married  school  \
909      poor        1  normal  other  7.8   no    male     yes      12
910   average        1  normal  other  7.5   no  female     yes      14
2592  average        4  normal   west  6.9   no    male     yes       6

      income employed insurance medicaid
909  -1.0125       no        no       no
910  -1.0125       no        no       no
2592 -0.8180       no       yes       no
```

[12]:

```
aura_df.isnull().sum()
```

[12]:

```
Unnamed: 0     0
visits         0
nvisits        0
ovisits        0
novisits       0
emergency      0
hospital       0
health         0
chronic        0
adl            0
region         0
age            0
afam           0
gender         0
married        0
school         0
income         0
employed       0
insurance      0
```

```
medicaid      0
dtype: int64
```

[13]:
```
# there is no missing data
```

[14]:
```
aura_df.drop('Unnamed: 0', axis=1, inplace=True)
aura_df
```

[14]:
```
      visits  nvisits  ovisits  novisits  emergency  hospital     health  \
0          5        0        0         0          0         1    average
1          1        0        2         0          2         0    average
2         13        0        0         0          3         3       poor
3         16        0        5         0          1         1       poor
4          3        0        0         0          0         0    average
…        …        …        …         …          …        …
4401      11        0        0         0          0         0    average
4402      12        0        0         0          0         0    average
4403      10        0       20         0          1         1    average
4404      16        1        0         0          0         0    average
4405       0        0        0         0          0         0  excellent

      chronic      adl region  age afam  gender married  school    income  \
0           2   normal  other  6.9  yes    male     yes       6  2.881000
1           2   normal  other  7.4   no  female     yes      10  2.747800
2           4  limited  other  6.6  yes  female      no      10  0.653200
3           2  limited  other  7.6   no    male     yes       3  0.658800
4           2  limited  other  7.9   no  female     yes       6  0.658800
…         …        …      …   …   …     …       …      …
4401        0   normal  other  8.4   no  female     yes       8  2.249700
4402        2   normal  other  7.8   no  female      no      11  5.813200
4403        5   normal  other  7.3   no    male     yes      12  3.877916
4404        0   normal  other  6.6   no  female     yes      12  3.877916
4405        0   normal  other  7.1   no    male     yes       0  6.596800

     employed insurance medicaid
0         yes       yes       no
1          no       yes       no
2          no        no      yes
3          no       yes       no
4          no       yes       no
…         …        …       …
4401       no       yes       no
4402       no       yes       no
4403       no       yes       no
4404       no       yes       no
4405      yes        no       no
```

```
[4406 rows x 19 columns]
```

Fix age:

```
[15]: aura_df['age'] = aura_df['age'] * 10
      aura_df['age'] = aura_df['age'].astype('uint8')
      aura_df
```

```
[15]:       visits  nvisits  ovisits  novisits  emergency  hospital     health  \
      0          5        0        0         0          0         1    average
      1          1        0        2         0          2         0    average
      2         13        0        0         0          3         3       poor
      3         16        0        5         0          1         1       poor
      4          3        0        0         0          0         0    average
      …      …      …      …       …        …       …        …
      4401      11        0        0         0          0         0    average
      4402      12        0        0         0          0         0    average
      4403      10        0       20         0          1         1    average
      4404      16        1        0         0          0         0    average
      4405       0        0        0         0          0         0  excellent

            chronic      adl  region age afam  gender married   school    income  \
      0           2   normal   other  69  yes    male     yes        6  2.881000
      1           2   normal   other  74   no  female     yes       10  2.747800
      2           4  limited   other  66  yes  female      no       10  0.653200
      3           2  limited   other  76   no    male     yes        3  0.658800
      4           2  limited   other  79   no  female     yes        6  0.658800
      …       …      …    …  …  …     …     …     …       …
      4401        0   normal   other  84   no  female     yes        8  2.249700
      4402        2   normal   other  78   no  female      no       11  5.813200
      4403        5   normal   other  73   no    male     yes       12  3.877916
      4404        0   normal   other  66   no  female     yes       12  3.877916
      4405        0   normal   other  71   no    male     yes        0  6.596800

           employed insurance medicaid
      0         yes       yes       no
      1          no       yes       no
      2          no        no      yes
      3          no       yes       no
      4          no       yes       no
      …       …       …      …
      4401       no       yes       no
      4402       no       yes       no
      4403       no       yes       no
      4404       no       yes       no
      4405      yes        no       no

      [4406 rows x 19 columns]
```

[16]:

```
# replace yes/no cells with 1/0 so the LLM can actually use the numerical data
```

[17]:

```
yes_no = {'yes':1, 'no':0}
yes_no_columns = ['afam', 'married', 'employed', 'insurance', 'medicaid']
aura_df[yes_no_columns] = aura_df[yes_no_columns].replace(yes_no)
aura_df[yes_no_columns] = aura_df[yes_no_columns].astype('uint8')
aura_df
```

```
/var/folders/_8/7k3l29n14j91jpxpd6v9rmtw0000gn/T/ipykernel_7185/3602872080.py:3:
FutureWarning: Downcasting behavior in `replace` is deprecated and will be
removed in a future version. To retain the old behavior, explicitly call
`result.infer_objects(copy=False)`. To opt-in to the future behavior, set
`pd.set_option('future.no_silent_downcasting', True)`
  aura_df[yes_no_columns] = aura_df[yes_no_columns].replace(yes_no)
```

[17]:

```
      visits  nvisits  ovisits  novisits  emergency  hospital     health  \
0          5        0        0         0          0         1    average
1          1        0        2         0          2         0    average
2         13        0        0         0          3         3       poor
3         16        0        5         0          1         1       poor
4          3        0        0         0          0         0    average
…        …       …       …        …         …         …        …
4401      11        0        0         0          0         0    average
4402      12        0        0         0          0         0    average
4403      10        0       20         0          1         1    average
4404      16        1        0         0          0         0    average
4405       0        0        0         0          0         0  excellent

      chronic      adl region  age  afam  gender  married  school    income  \
0           2   normal  other   69     1    male        1       6  2.881000
1           2   normal  other   74     0  female        1      10  2.747800
2           4  limited  other   66     1  female        0      10  0.653200
3           2  limited  other   76     0    male        1       3  0.658800
4           2  limited  other   79     0  female        1       6  0.658800
…         …       …     …    …    …      …       …       …        …
4401        0   normal  other   84     0  female        1       8  2.249700
4402        2   normal  other   78     0  female        0      11  5.813200
4403        5   normal  other   73     0    male        1      12  3.877916
4404        0   normal  other   66     0  female        1      12  3.877916
4405        0   normal  other   71     0    male        1       0  6.596800

      employed  insurance  medicaid
0            1          1         0
1            0          1         0
2            0          0         1
3            0          1         0
4            0          1         0
…         …        …        …
```

```
4401          0          1         0
4402          0          1         0
4403          0          1         0
4404          0          1         0
4405          1          0         0

[4406 rows x 19 columns]
```

[18]:

```
# set columns of category data type
```

[19]:

```
category_columns = ['health', 'adl', 'gender', 'region']
aura_df[category_columns] = aura_df[category_columns].astype('category')
aura_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4406 entries, 0 to 4405
Data columns (total 19 columns):
 #   Column     Non-Null Count  Dtype
---  ------     --------------  -----
 0   visits     4406 non-null   int64
 1   nvisits    4406 non-null   int64
 2   ovisits    4406 non-null   int64
 3   novisits   4406 non-null   int64
 4   emergency  4406 non-null   int64
 5   hospital   4406 non-null   int64
 6   health     4406 non-null   category
 7   chronic    4406 non-null   int64
 8   adl        4406 non-null   category
 9   region     4406 non-null   category
 10  age        4406 non-null   uint8
 11  afam       4406 non-null   uint8
 12  gender     4406 non-null   category
 13  married    4406 non-null   uint8
 14  school     4406 non-null   int64
 15  income     4406 non-null   float64
 16  employed   4406 non-null   uint8
 17  insurance  4406 non-null   uint8
 18  medicaid   4406 non-null   uint8
dtypes: category(4), float64(1), int64(8), uint8(6)
memory usage: 353.5 KB
```

Optimizing data by reducing memory size:

[20]:

```
int64_columns = aura_df.select_dtypes('int64').columns
aura_df[int64_columns].describe()
```

[20]:

```
            visits      nvisits      ovisits     novisits    emergency  \
count  4406.000000  4406.000000  4406.000000  4406.000000  4406.000000
```

```
mean      5.774399     1.618021     0.750794     0.536087     0.263504
std       6.759225     5.317056     3.652759     3.879506     0.703659
min       0.000000     0.000000     0.000000     0.000000     0.000000
25%       1.000000     0.000000     0.000000     0.000000     0.000000
50%       4.000000     0.000000     0.000000     0.000000     0.000000
75%       8.000000     1.000000     0.000000     0.000000     0.000000
max      89.000000   104.000000   141.000000   155.000000    12.000000

          hospital      chronic       school
count  4406.000000  4406.000000  4406.000000
mean      0.295960     1.541988    10.290286
std       0.746398     1.349632     3.738736
min       0.000000     0.000000     0.000000
25%       0.000000     1.000000     8.000000
50%       0.000000     1.000000    11.000000
75%       0.000000     2.000000    12.000000
max       8.000000     8.000000    18.000000
```

[21]:
```
aura_df[int64_columns] = aura_df[int64_columns].astype('int16')
aura_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4406 entries, 0 to 4405
Data columns (total 19 columns):
 #   Column     Non-Null Count  Dtype
---  ------     --------------  -----
 0   visits     4406 non-null   int16
 1   nvisits    4406 non-null   int16
 2   ovisits    4406 non-null   int16
 3   novisits   4406 non-null   int16
 4   emergency  4406 non-null   int16
 5   hospital   4406 non-null   int16
 6   health     4406 non-null   category
 7   chronic    4406 non-null   int16
 8   adl        4406 non-null   category
 9   region     4406 non-null   category
 10  age        4406 non-null   uint8
 11  afam       4406 non-null   uint8
 12  gender     4406 non-null   category
 13  married    4406 non-null   uint8
 14  school     4406 non-null   int16
 15  income     4406 non-null   float64
 16  employed   4406 non-null   uint8
 17  insurance  4406 non-null   uint8
 18  medicaid   4406 non-null   uint8
dtypes: category(4), float64(1), int16(8), uint8(6)
memory usage: 147.0 KB
```

[22]:
```
float64_columns = aura_df.select_dtypes('float64').columns
aura_df[float64_columns] = aura_df[float64_columns].astype('float16')
aura_df[float64_columns].info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4406 entries, 0 to 4405
Data columns (total 1 columns):
 #   Column  Non-Null Count  Dtype
---  ------  --------------  -----
 0   income  4406 non-null   float16
dtypes: float16(1)
memory usage: 8.7 KB
```

[23]:
```
# fix negative income
aura_df.loc[aura_df.income < 0, 'income'] = 0
aura_df.income.describe()
```

[23]:
```
count    4406.000000
mean        2.525391
std         2.921875
min         0.000000
25%         0.912231
50%         1.697754
75%         3.173340
max        54.843750
Name: income, dtype: float64
```

Exported optimized data to a CSV and PKL file:

[24]:
```
aura_df.to_csv("NSMES1988_optimized.csv", index=False)
optimized_aura_df = pd.read_csv("NSMES1988_optimized.csv")
optimized_aura_df.info()

# to preserve the data types, export dataframe to a pkl file

aura_df.to_pickle("NSMES1988_optimized.pkl")
optimized_aura_df = pd.read_pickle("NSMES1988_optimized.pkl")
optimized_aura_df.info()
```

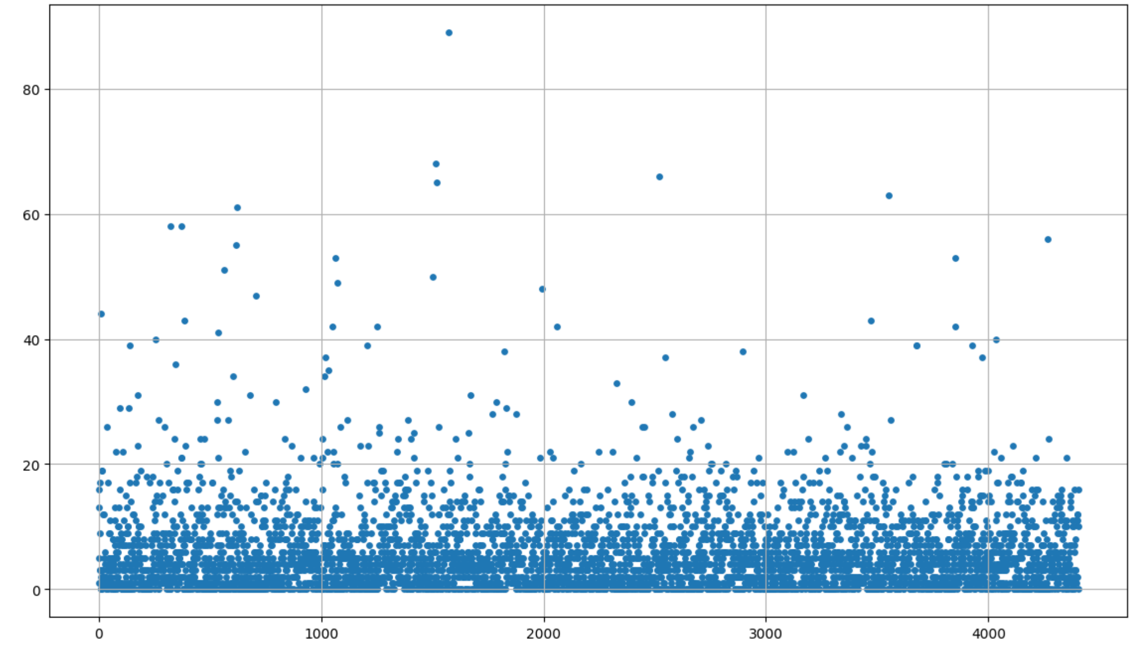
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4406 entries, 0 to 4405
Data columns (total 19 columns):
 #   Column     Non-Null Count  Dtype
---  ------     --------------  -----
 0   visits     4406 non-null   int64
 1   nvisits    4406 non-null   int64
 2   ovisits    4406 non-null   int64
 3   novisits   4406 non-null   int64
 4   emergency  4406 non-null   int64
 5   hospital   4406 non-null   int64
 6   health     4406 non-null   object
 7   chronic    4406 non-null   int64
 8   adl        4406 non-null   object
 9   region     4406 non-null   object
 10  age        4406 non-null   int64
 11  afam       4406 non-null   int64
 12  gender     4406 non-null   object
 13  married    4406 non-null   int64
 14  school     4406 non-null   int64
 15  income     4406 non-null   float64
 16  employed   4406 non-null   int64
 17  insurance  4406 non-null   int64
 18  medicaid   4406 non-null   int64
dtypes: float64(1), int64(14), object(4)
memory usage: 654.1+ KB
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4406 entries, 0 to 4405
Data columns (total 19 columns):
 #   Column     Non-Null Count  Dtype
---  ------     --------------  -----
 0   visits     4406 non-null   int16
 1   nvisits    4406 non-null   int16
 2   ovisits    4406 non-null   int16
 3   novisits   4406 non-null   int16
 4   emergency  4406 non-null   int16
 5   hospital   4406 non-null   int16
 6   health     4406 non-null   category
 7   chronic    4406 non-null   int16
 8   adl        4406 non-null   category
 9   region     4406 non-null   category
 10  age        4406 non-null   uint8
 11  afam       4406 non-null   uint8
 12  gender     4406 non-null   category
 13  married    4406 non-null   uint8
 14  school     4406 non-null   int16
 15  income     4406 non-null   float16
 16  employed   4406 non-null   uint8
 17  insurance  4406 non-null   uint8
 18  medicaid   4406 non-null   uint8
dtypes: category(4), float16(1), int16(8), uint8(6)
memory usage: 120.7 KB
```

[25]:

```
# Show significant correlations
#aura_df.corr()[abs(aura_df.corr()) > 0.25].fillna(0).style.
 ↪background_gradient(cmap='Spectral', axis=None)
```

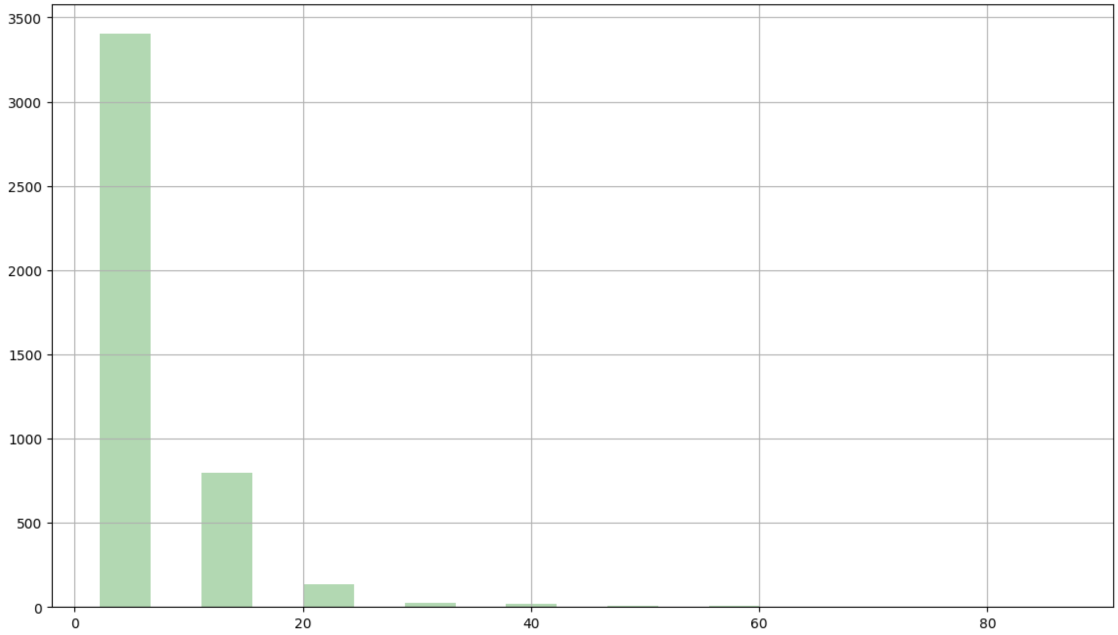
[26]:
```
# short report detailing visual observations such as number of visits
import matplotlib.pyplot as plt

x = aura_df.index
y = aura_df.visits

plt.figure(figsize=(14,8))
plt.grid()
plt.scatter(x, y, s=15)
```

[26]:
```
<matplotlib.collections.PathCollection at 0x16128ffa0>
```



[27]:
```
plt.figure(figsize=(14,8))
plt.grid()
plt.hist(aura_df.visits, bins=10, color='g', alpha=0.3, rwidth=0.5)
```

[27]:
```
(array([3.406e+03, 7.970e+02, 1.380e+02, 2.700e+01, 2.100e+01, 7.000e+00,
        5.000e+00, 4.000e+00, 0.000e+00, 1.000e+00]),
 array([ 0. ,  8.9, 17.8, 26.7, 35.6, 44.5, 53.4, 62.3, 71.2, 80.1, 89. ]),
 <BarContainer object of 10 artists>)
```



[28]:

```
optimized_aura_df
```

[28]:

```
      visits  nvisits  ovisits  novisits  emergency  hospital     health  \
0          5        0        0         0          0         1    average
1          1        0        2         0          2         0    average
2         13        0        0         0          3         3       poor
3         16        0        5         0          1         1       poor
4          3        0        0         0          0         0    average
…       …       …       …        …         …        …        …
4401      11        0        0         0          0         0    average
4402      12        0        0         0          0         0    average
4403      10        0       20         0          1         1    average
4404      16        1        0         0          0         0    average
4405       0        0        0         0          0         0  excellent

      chronic      adl  region  age  afam  gender  married  school    income  \
0           2   normal   other   69     1    male        1       6  2.880859
1           2   normal   other   74     0  female        1      10  2.748047
2           4  limited   other   66     1  female        0      10  0.653320
3           2  limited   other   76     0    male        1       3  0.658691
4           2  limited   other   79     0  female        1       6  0.658691
…        …       …    …  …   …      …      …      …        …
4401        0   normal   other   84     0  female        1       8  2.250000
4402        2   normal   other   78     0  female        0      11  5.812500
4403        5   normal   other   73     0    male        1      12  3.876953
4404        0   normal   other   66     0  female        1      12  3.876953
```

```
4405        0   normal  other   71     0    male        1       0  6.597656

      employed  insurance  medicaid
0            1          1         0
1            0          1         0
2            0          0         1
3            0          1         0
4            0          1         0
…        …        …       …
4401         0          1         0
4402         0          1         0
4403         0          1         0
4404         0          1         0
4405         1          0         0

[4406 rows x 19 columns]
```

[29]:

```
# multiply income by 10 to correct the corrupted data depicting income as X.xx

optimized_aura_df['income'] = optimized_aura_df['income'] * 10
optimized_aura_df['income']
```

[29]:

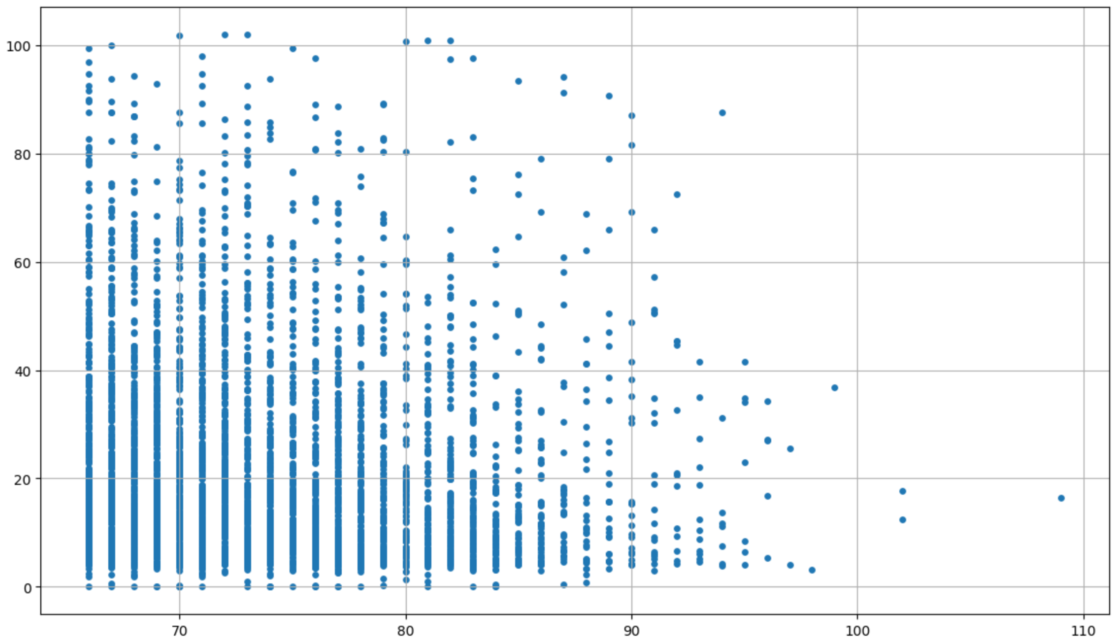
```
0       28.812500
1       27.484375
2        6.531250
3        6.585938
4        6.585938
          …
4401    22.500000
4402    58.125000
4403    38.781250
4404    38.781250
4405    66.000000
Name: income, Length: 4406, dtype: float16
```

[30]:

```
df_quant = optimized_aura_df.income.quantile(0.98)
filtered_data = optimized_aura_df[optimized_aura_df.income < df_quant]

plt.figure(figsize=(14,8))
plt.grid()
plt.scatter(filtered_data.age, filtered_data.income, s=15)
```

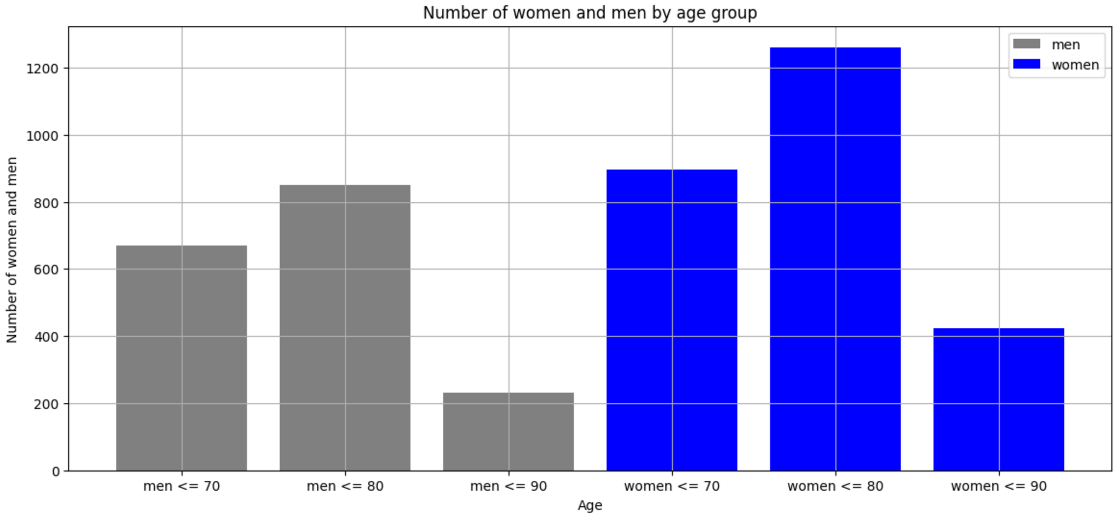
[30]:

```
<matplotlib.collections.PathCollection at 0x1614113a0>
```



The highest income band occurs under the age of 80.

```
[31]: # visualize how many women and men there are by age groups

grp_women = optimized_aura_df[optimized_aura_df['gender']=='female']
grp_men = optimized_aura_df[optimized_aura_df['gender']=='male']

grp_women_70 = grp_women[grp_women['age'] <= 70].shape[0]
grp_men_70 = grp_men[grp_men['age'] <= 70].shape[0]

grp_women_80 = grp_women[(grp_women['age'] > 70) & (grp_women['age'] <= 80)].
  ↪shape[0]
grp_men_80 = grp_men[(grp_men['age'] > 70) & (grp_men['age'] <= 80)].shape[0]

grp_women_90 = grp_women[(grp_women['age'] > 80) & (grp_women['age'] <= 90)].
  ↪shape[0]
grp_men_90 = grp_men[(grp_men['age'] > 80) & (grp_men['age'] <= 90)].shape[0]

plt.figure(figsize=(14,6))
plt.grid()
plt.bar(['men <= 70', 'men <= 80', 'men <= 90'],[grp_men_70, grp_men_80,␣
  ↪grp_men_90], color='gray', label='men')
plt.bar(['women <= 70', 'women <= 80', 'women <= 90'],[grp_women_70,␣
  ↪grp_women_80, grp_women_90], color='blue', label='women')
plt.xlabel('Age')
plt.ylabel('Number of women and men')
plt.title('Number of women and men by age group')
plt.legend()
plt.show()
```



Most men and women are between ages 71 and 80. Women between ages 71 and 80 consist of the largest age group by gender. Men between ages 81 and 90 consist of the smallest age group by gender.

```
[32]: optimized_aura_df.age.describe()
```

```
[32]: count    4406.000000
      mean       74.024058
      std         6.334050
      min        66.000000
      25%        69.000000
      50%        73.000000
      75%        78.000000
      max       109.000000
      Name: age, dtype: float64
```

When we describe our dataframe, we confirm most men and women are between ages 71 and 80.

```
[33]: optimized_aura_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4406 entries, 0 to 4405
Data columns (total 19 columns):
 #   Column     Non-Null Count  Dtype
---  ------     --------------  -----
 0   visits     4406 non-null   int16
 1   nvisits    4406 non-null   int16
 2   ovisits    4406 non-null   int16
 3   novisits   4406 non-null   int16
 4   emergency  4406 non-null   int16
 5   hospital   4406 non-null   int16
 6   health     4406 non-null   category
 7   chronic    4406 non-null   int16
 8   adl        4406 non-null   category
 9   region     4406 non-null   category
 10  age        4406 non-null   uint8
 11  afam       4406 non-null   uint8
 12  gender     4406 non-null   category
 13  married    4406 non-null   uint8
 14  school     4406 non-null   int16
 15  income     4406 non-null   float16
 16  employed   4406 non-null   uint8
 17  insurance  4406 non-null   uint8
 18  medicaid   4406 non-null   uint8
dtypes: category(4), float16(1), int16(8), uint8(6)
memory usage: 120.8 KB
```

All members in the ‘novisits’ column have a value of 0, and therefore this data is not usable for statistical analysis.

[34]:

```
optimized_aura_df.to_csv("NSMES1988_optimized_v2.csv", index=False)
optimized_aura_df_v2 = pd.read_csv("NSMES1988_optimized_v2.csv")

optimized_aura_df.to_pickle("NSMES1988_optimized_v2.pkl")
optimized_aura_df_v2 = pd.read_pickle("NSMES1988_optimized_v2.pkl")
optimized_aura_df_v2.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4406 entries, 0 to 4405
Data columns (total 19 columns):
 #   Column     Non-Null Count  Dtype
---  ------     --------------  -----
 0   visits     4406 non-null   int16
 1   nvisits    4406 non-null   int16
 2   ovisits    4406 non-null   int16
 3   novisits   4406 non-null   int16
 4   emergency  4406 non-null   int16
 5   hospital   4406 non-null   int16
 6   health     4406 non-null   category
 7   chronic    4406 non-null   int16
 8   adl        4406 non-null   category
 9   region     4406 non-null   category
 10  age        4406 non-null   uint8
 11  afam       4406 non-null   uint8
 12  gender     4406 non-null   category
 13  married    4406 non-null   uint8
 14  school     4406 non-null   int16
 15  income     4406 non-null   float16
 16  employed   4406 non-null   uint8
 17  insurance  4406 non-null   uint8
 18  medicaid   4406 non-null   uint8
dtypes: category(4), float16(1), int16(8), uint8(6)
memory usage: 120.7 KB
```

0.1 Conduct data pivoting, using cross tabulation for example, with pairs of categorical features.

0.2 In this case, let's examine the count of observations for combinations of health per region.

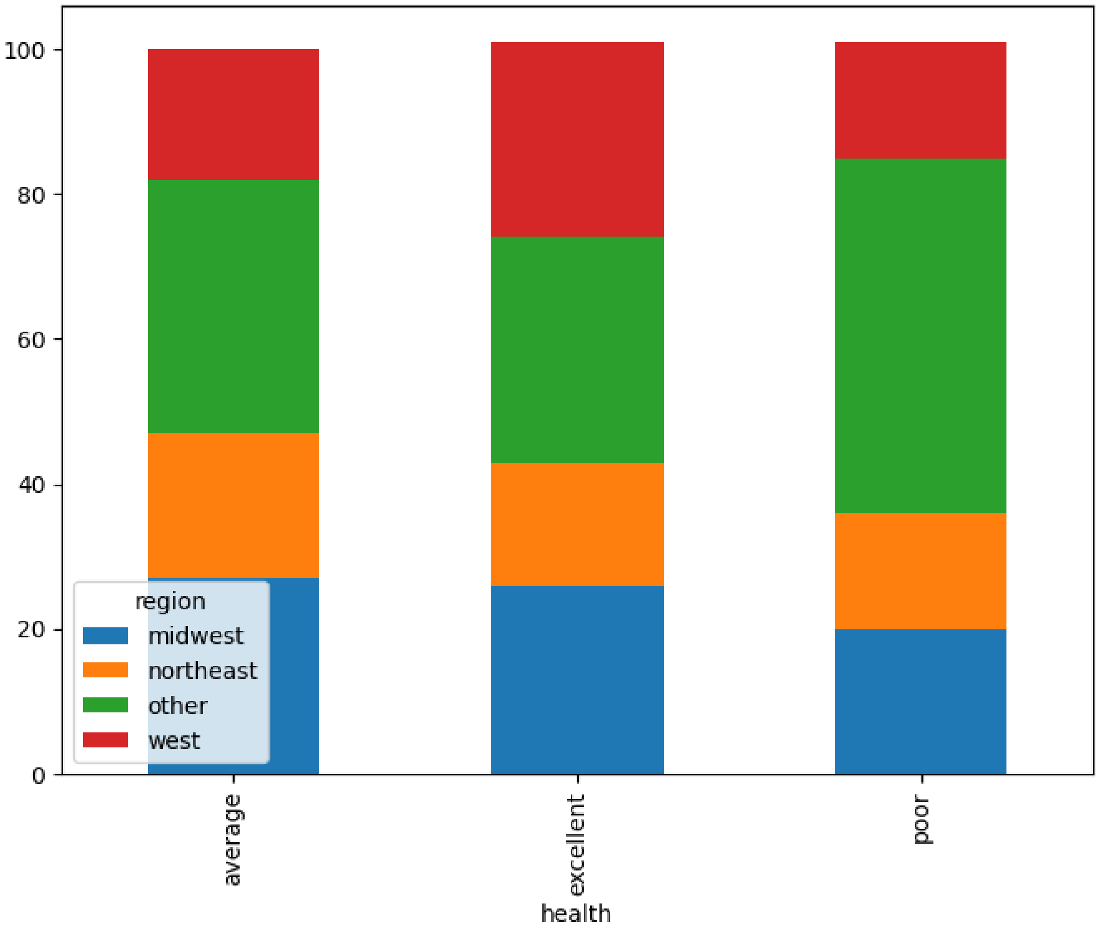
```
[87]: pd.crosstab(optimized_aura_df_v2['health'], optimized_aura_df_v2['region'], margins=True)
```

```
[87]: region     midwest  northeast  other  west   All
      health
      average        957        694   1237   621  3509
      excellent       90         57    105    91   343
      poor           110         86    272    86   554
      All           1157        837   1614   798  4406
```

```
[90]: health_region_crosstab = pd.crosstab(optimized_aura_df_v2['health'], optimized_aura_df_v2['region'], normalize='index').round(2)*100
      health_region_crosstab
```

```
[90]: region     midwest  northeast  other  west
      health
      average       27.0       20.0   35.0  18.0
      excellent     26.0       17.0   31.0  27.0
      poor          20.0       16.0   49.0  16.0
```

```
[92]: health_region_crosstab.plot(kind='bar', stacked=True, figsize=(8,6));
```



Is a patient's health dependent on the region they live in? Since 'health' and 'region' are categorical features, we can conduct the Chi-Squared Test to see if there is independence between the variables.

H_0 : 'health' is independent of 'region'

H_a : 'health' is highly related to 'region'

[48]:

```python
from scipy.stats import chi2_contingency

chi2, p_value, dof, expected = chi2_contingency(health_region_crosstab)
# Display the p-value in a legible manner
print(f"The p-value is: {p_value:.16f}")
```

```
The p-value is: 0.0000000000407694
```

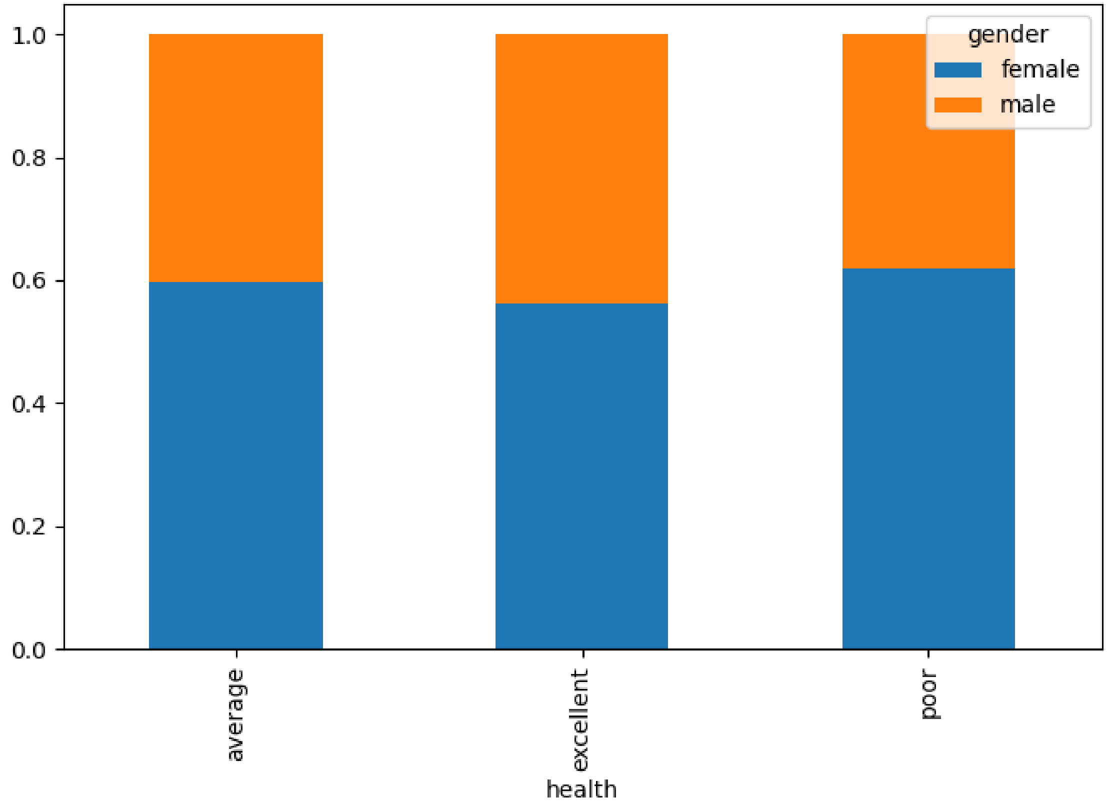
Since the 'p value' is < 0.05, we reject the null hypothesis. We conclude that a patient's health is highly correlated with the region they live in.

0.3 Create a distribution table that categorizes individuals by their health status, differentiated by gender.

```
[59]: contingency_table = pd.crosstab(optimized_aura_df_v2['health'],␣
       ↪optimized_aura_df_v2['gender'])
      contingency_table

      health_gender_distribution = contingency_table.div(contingency_table.
       ↪sum(axis=1), axis=0)
      health_gender_distribution.round(2)
```

```
[59]: gender     female  male
      health
      average      0.60  0.40
      excellent    0.56  0.44
      poor         0.62  0.38
```

```
[100]: health_gender_distribution.plot.bar(stacked='True', figsize=(8,5));
```



0.4 Determine if gender and health are independent categorical variables.

H_0 : ‘health’ and ‘gender’ are independent of each other.
H_a : ‘health’ and ‘gender’ are dependent of each other.

[62]:

```
chi2, p_value, dof, expected = chi2_contingency(health_gender_distribution)
p_value
```

[62]:

```
0.9968561005885185
```

The p value is 0.99, which is > 0.05.
We therefore accept the null hypothesis, H_0 :, and conclude that ‘health’ and ‘gender’ are independent of one another.

0.5 Compile a table to examine the income distribution across genders.

[117]:

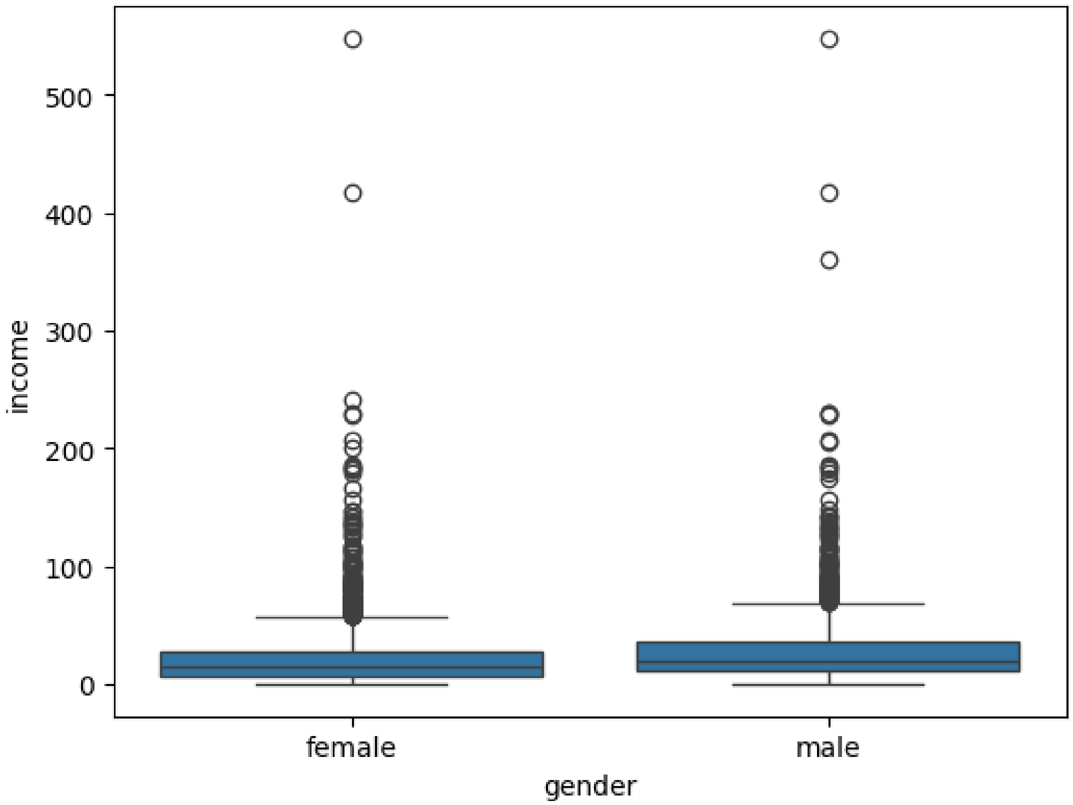
```
optimized_aura_df_v2['income'] = optimized_aura_df_v2['income'].
 ↪astype('float32') #crosstab does not accept float16
income_gender_contingency_table = pd.crosstab(optimized_aura_df_v2['income'],␣
 ↪optimized_aura_df_v2['gender'])
income_gender_contingency_table
```

[117]:

```
gender   female  male
income
0.0          20    10
1.0           6     1
2.0          21     2
3.0          60    17
4.0         193    33
…           …   …
230.0         1     2
242.0         1     0
360.0         0     1
417.0         1     1
548.0         1     1

[146 rows x 2 columns]
```

[109]:

```
import seaborn as sns
sns.boxplot(data=optimized_aura_df_v2, x="gender", y="income");
```



```
[111]: optimized_aura_df_v2.groupby('gender', observed=True)['income'].describe().
       ↪round(2).T
```

```
[111]: gender   female     male
       count   2628.00  1778.00
       mean      22.01    28.90
       std       27.21    31.57
       min        0.00     0.00
       25%        7.00    12.00
       50%       14.00    20.00
       75%       27.00    35.00
       max      548.00   548.00
```

```
[77]: chi2, p_value, dof, expected = chi2_contingency(income_gender_contingency_table)
      print(f"The p-value is: {p_value:.16f}")
```

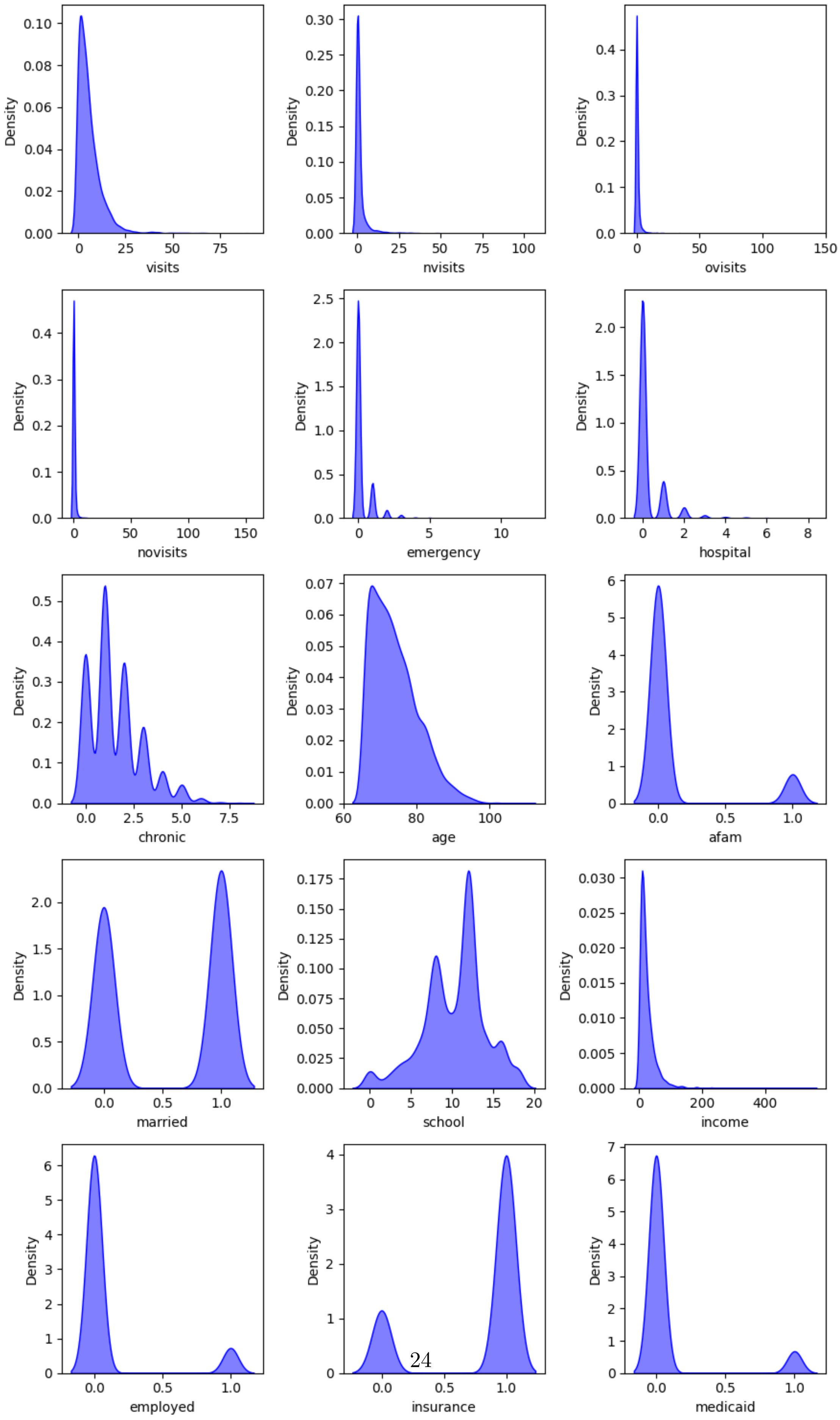
```
The p-value is: 0.0000000000003508
```

Income distribution and gender are highly dependent of one another.

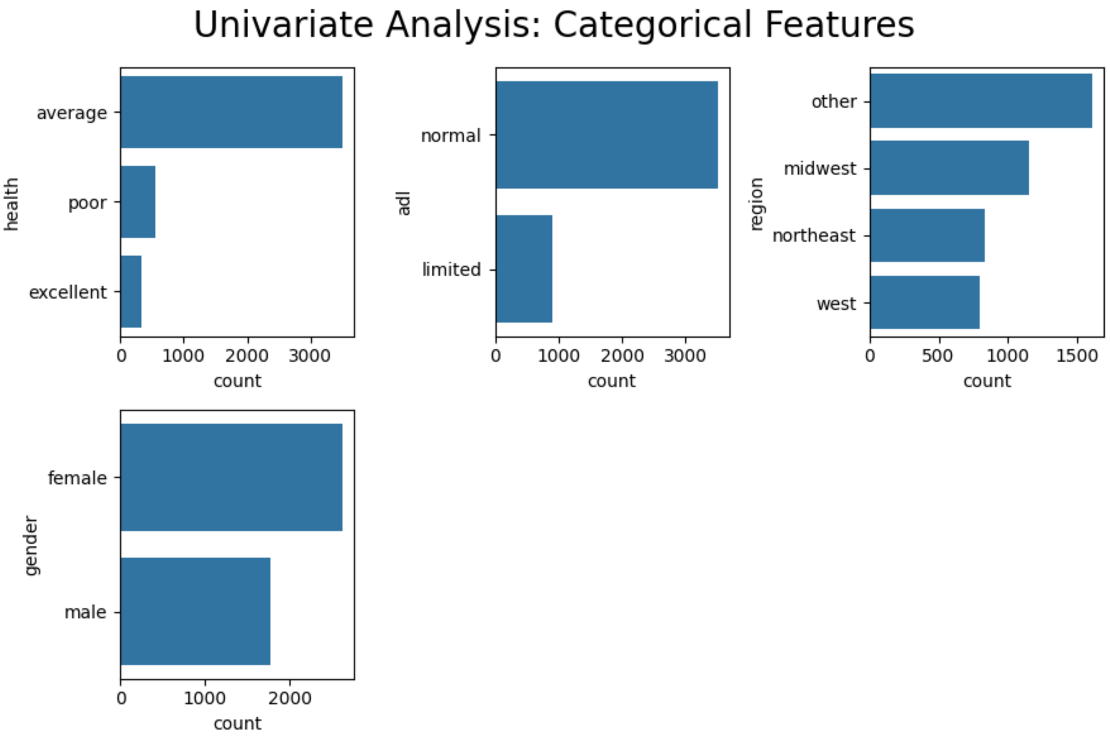
0.6 Univariate Analysis

[84]:

```
from vizad.univariate import plot_univariate_numeric,␣
 ↪plot_univariate_categorical

num_cols = optimized_aura_df_v2.select_dtypes(include=np.number).columns.
 ↪tolist()
cat_cols = [col for col in optimized_aura_df_v2.columns if col not in num_cols]

plot_univariate_numeric(optimized_aura_df_v2, num_cols, figsize=(8,8),␣
 ↪kind='density')
```

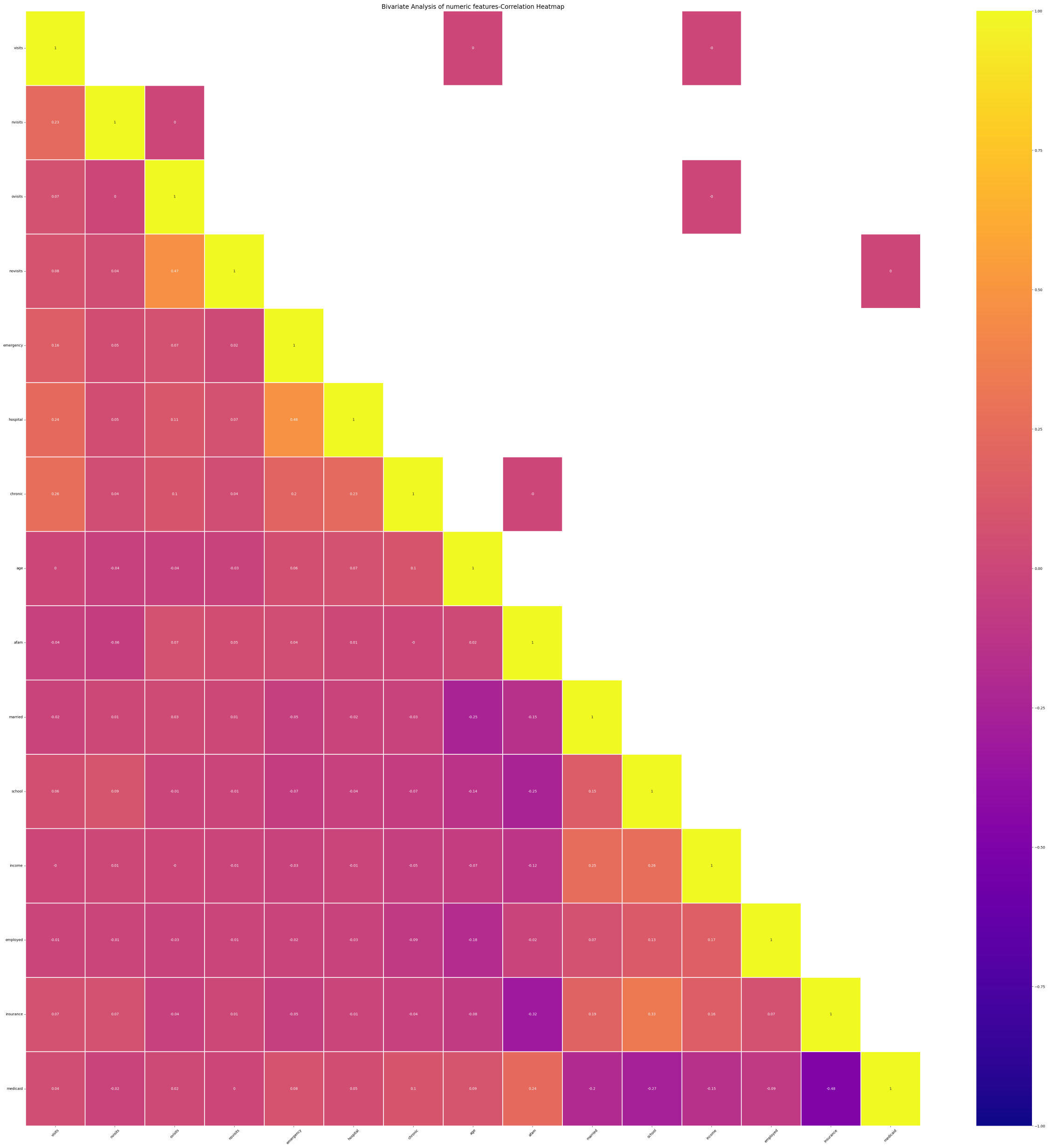

Density
0.00
0.02
0.04
0.06
0.08
0.10
0
25
50
75
visits
0.00
0.05
0.10
0.15
0.20
0.25
0.30
0
25
50
75
100
nvisits
0.0
0.1
0.2
0.3
0.4
0
50
100
150
ovisits
0.0
0.1
0.2
0.3
0.4
0
50
100
150
novisits
0.0
0.5
1.0
1.5
2.0
2.5
0
5
10
emergency
0.0
0.5
1.0
1.5
2.0
0
2
4
6
8
hospital
0.0
0.1
0.2
0.3
0.4
0.5
0.0
2.5
5.0
7.5
chronic
0.00
0.01
0.02
0.03
0.04
0.05
0.06
0.07
60
80
100
age
0
1
2
3
4
5
6
0.0
0.5
1.0
afam
0.0
0.5
1.0
1.5
2.0
0.0
0.5
1.0
married
0.000
0.025
0.050
0.075
0.100
0.125
0.150
0.175
0
5
10
15
20
school
0.000
0.005
0.010
0.015
0.020
0.025
0.030
0
200
400
income
0
1
2
3
4
5
6
0.0
0.5
1.0
employed
0
1
2
3
4
0.0
0.5
1.0
insurance
0
1
2
3
4
5
6
7
0.0
0.5
1.0
medicaid

```
[86]: plot_univariate_categorical(optimized_aura_df_v2, cat_cols, figsize=(8,4))
```

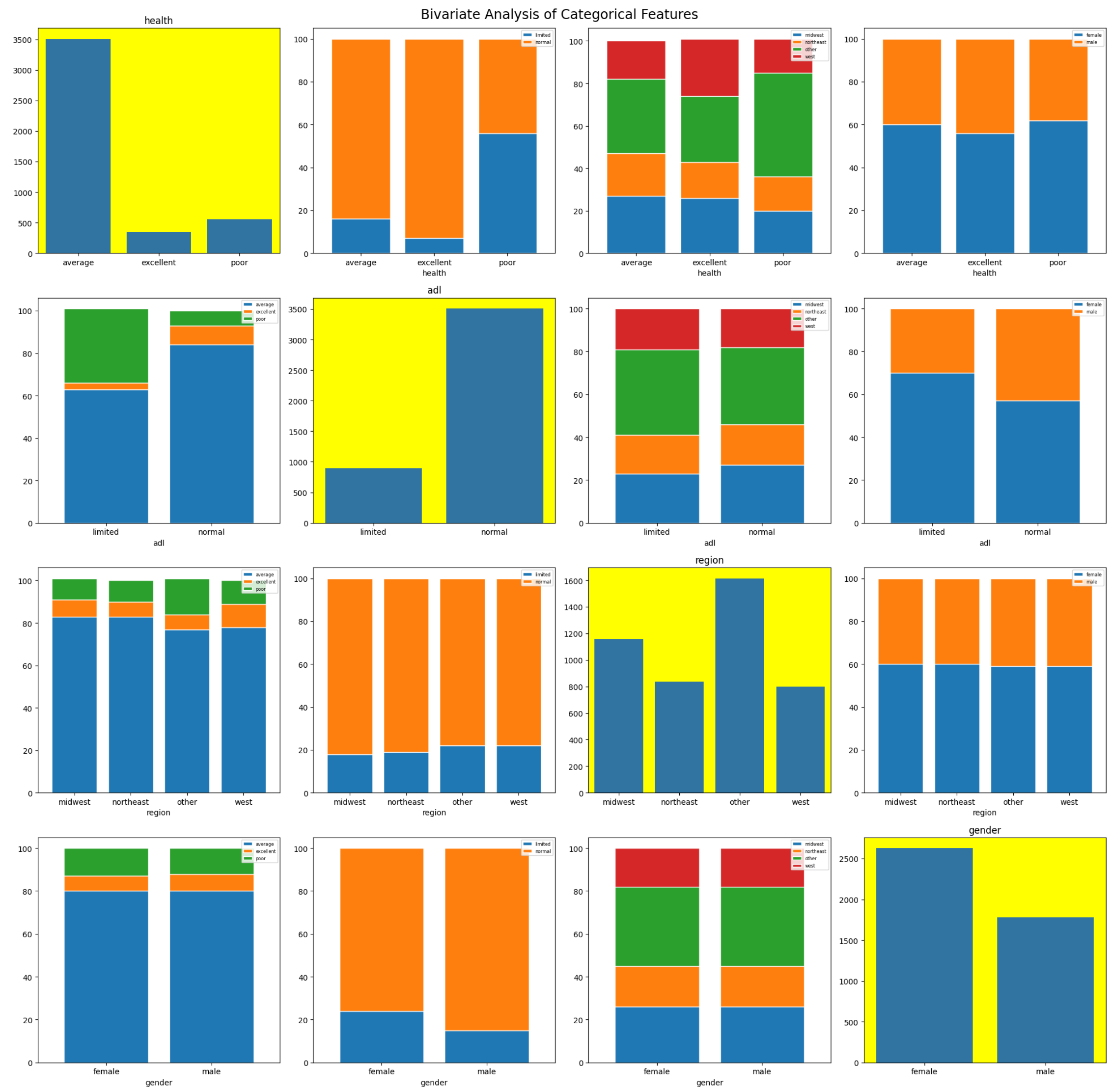


0.7 Bivariate Analysis of Categorical Features

```
[97]: plot_bivariate_numeric(optimized_aura_df_v2, num_cols, kind='heatmap')
```



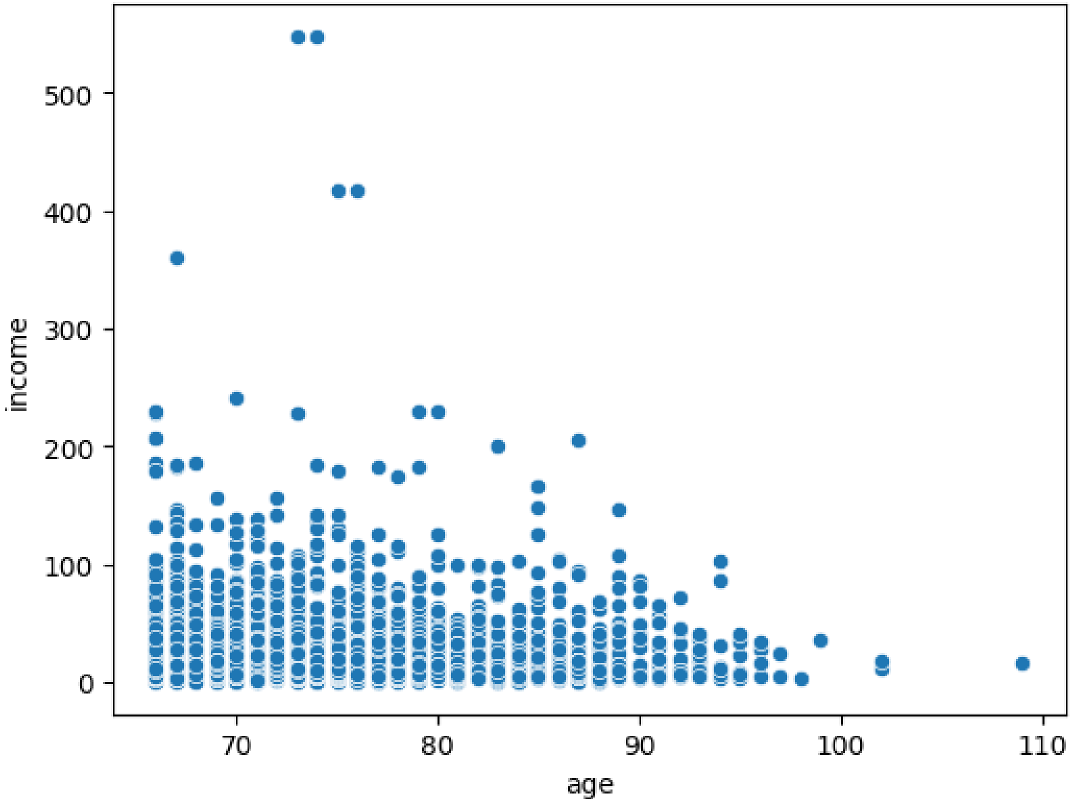
[98]:
```
from vizad.bivariate import plot_bivariate_numeric, plot_bivariate_categorical

num_cols = optimized_aura_df_v2.select_dtypes(include=np.number).columns.
  ↪tolist()
cat_cols = [col for col in optimized_aura_df_v2.columns if col not in num_cols]

plot_bivariate_categorical(optimized_aura_df_v2, cat_cols, figsize=(20,20))
```



0.8 Age vs Income Relationship

Develop a table to analyze the relationship between age and income.

```
[115]: sns.scatterplot(data=optimized_aura_df_v2 , x="age", y="income")
       plt.xlabel='age'
       plt.ylabel='income'
```


income
500
400
300
200
100
0
70
80
90
100
110
age

```
[ ]:
```