nhanes_univariate_practice

October 25, 2022

1 Univariate analysis using NHANES data

This notebook will give you the opportunity to perform some univariate analyses on your own using the NHANES.

```
In [1]: %matplotlib inline
        import matplotlib.pyplot as plt
        import seaborn as sns
        import pandas as pd
        import statsmodels.api as sm
        import numpy as np

        da = pd.read_csv("nhanes_2015_2016.csv")
```

1.1 Question 1

Relabel the marital status variable DMDMARTL to have brief but informative character labels. Then construct a frequency table of these values for all people, then for women only, and for men only. Then construct these three frequency tables using only people whose age is between 30 and 40.

```
In [33]: #Create an alternate dataframe so main data is unaltered
         da_clean = da

         #This is in order to relabel the numeric data to characters
         da_clean['DMDMarrital'] = da.DMDMARTL.replace({1: 'Married', 2: 'Single', 3: 'In a Rel
                                                        5: 'Complicated', 6: 'Not Interested', 77: 'I

         #This code was in order to see the count in each strata
         da_clean.DMDMarrital.value_counts()

Out[33]: Married              2780
         Complicated          1004
         In a Relationship     579
         Not Interested        527
         Single                396
         Divorced              186
         Prefer not to say       2
         Name: DMDMarrital, dtype: int64
```

```
In [54]: #This was to be able to relabel the numeric data according to sex
         da_clean['gender'] = da.RIAGENDR.replace({1: 'Male', 2: 'Female'})

         #This was to see the count in each strata by gender
         da_clean.groupby('gender')['DMDMarrital'].value_counts(normalize = True)
```

```
Out[54]: gender  DMDMarrital
         Female  Married                 0.457193
                 Complicated             0.182456
                 In a Relationship       0.122807
                 Single                  0.103860
                 Not Interested          0.091930
                 Divorced                0.041404
                 Prefer not to say       0.000351
         Male    Married                 0.562881
                 Complicated             0.184451
                 Not Interested          0.100991
                 In a Relationship       0.087271
                 Single                  0.038110
                 Divorced                0.025915
                 Prefer not to say       0.000381
         Name: DMDMarrital, dtype: float64
```

```
In [39]: # this is to get the agegroup 30 to 40
         da_clean["agegrp"] = pd.cut(da.RIDAGEYR, [30, 40])

         #This was to see the stratified age group's 30-40 and see reported marrital status
         da_clean.groupby("agegrp")["DMDMarrital"].value_counts()
```

```
Out[39]: agegrp    DMDMarrital
         (30, 40]  Married                 516
                   Complicated             186
                   Not Interested          129
                   In a Relationship        67
                   Divorced                 29
                   Single                    4
                   Prefer not to say         1
         Name: DMDMarrital, dtype: int64
```

```
In [38]: # Combining the two variables we created as filters to report marrital status
         da_clean.groupby(['gender', 'agegrp'])['DMDMarrital'].value_counts().unstack()
```

```
Out[38]: DMDMarrital      Complicated  Divorced  In a Relationship  Married  \
         gender agegrp
         Female (30, 40]         97.0      17.0               43.0    258.0
         Male   (30, 40]         89.0      12.0               24.0    258.0
         DMDMarrital      Not Interested  Prefer not to say  Single
         gender agegrp
         Female (30, 40]            57.0                NaN     2.0
         Male   (30, 40]            72.0                1.0     2.0
```

```
In [40]: #Replaced NaN variables in Table to 0
         x = da_clean.groupby(['gender', 'agegrp'])['DMDMarrital'].value_counts().unstack()
         x = x.fillna(0)
         x
```

```
Out[40]: DMDMarrital      Complicated  Divorced  In a Relationship  Married  \
         gender agegrp
         Female (30, 40]         97.0      17.0               43.0    258.0
         Male   (30, 40]         89.0      12.0               24.0    258.0

         DMDMarrital      Not Interested  Prefer not to say  Single
         gender agegrp
         Female (30, 40]            57.0                0.0     2.0
         Male   (30, 40]            72.0                1.0     2.0
```

Q1a. Briefly comment on some of the differences that you observe between the distribution of marital status between women and men, for people of all ages.

```
In [43]: #Created a new age strata covering ages with 10 years minimum difference
         da_clean['agestrata'] = pd.cut(da.RIDAGEYR, [10,20, 30, 40, 50, 60, 70, 80])

         #Replaced NaN variables in Table to 0
         y = da_clean.groupby(['gender', 'agestrata'])['DMDMarrital'].value_counts().unstack()
         y = y.fillna(0)
         y
```

```
Out[43]: DMDMarrital       Complicated  Divorced  In a Relationship  Married  \
         gender agestrata
         Female (10, 20]          30.0       0.0                0.0      1.0
                (20, 30]         229.0      11.0               11.0    157.0
                (30, 40]          97.0      17.0               43.0    258.0
                (40, 50]          63.0      33.0               69.0    288.0
                (50, 60]          42.0      27.0               83.0    257.0
                (60, 70]          38.0      22.0               85.0    212.0
                (70, 80]          21.0       8.0               59.0    130.0
         Male   (10, 20]          36.0       0.0                0.0      1.0
                (20, 30]         226.0       7.0                2.0    103.0
                (30, 40]          89.0      12.0               24.0    258.0
                (40, 50]          39.0      11.0               34.0    282.0
                (50, 60]          47.0      10.0               57.0    296.0
                (60, 70]          38.0      14.0               55.0    291.0
                (70, 80]           9.0      14.0               57.0    246.0
         DMDMarrital       Not Interested  Prefer not to say  Single
         gender agestrata
         Female (10, 20]              8.0                0.0     0.0
                (20, 30]            106.0                0.0     0.0
                (30, 40]             57.0                0.0     2.0
                (40, 50]             37.0                0.0    12.0
                (50, 60]             32.0                1.0    28.0
                (60, 70]             19.0                0.0    65.0
                (70, 80]              3.0                0.0   189.0
         Male   (10, 20]              3.0                0.0     0.0
                (20, 30]             92.0                0.0     2.0
                (30, 40]             72.0                1.0     2.0
                (40, 50]             33.0                0.0     2.0
                (50, 60]             34.0                0.0    10.0
                (60, 70]             22.0                0.0    17.0
                (70, 80]              9.0                0.0    67.0
```

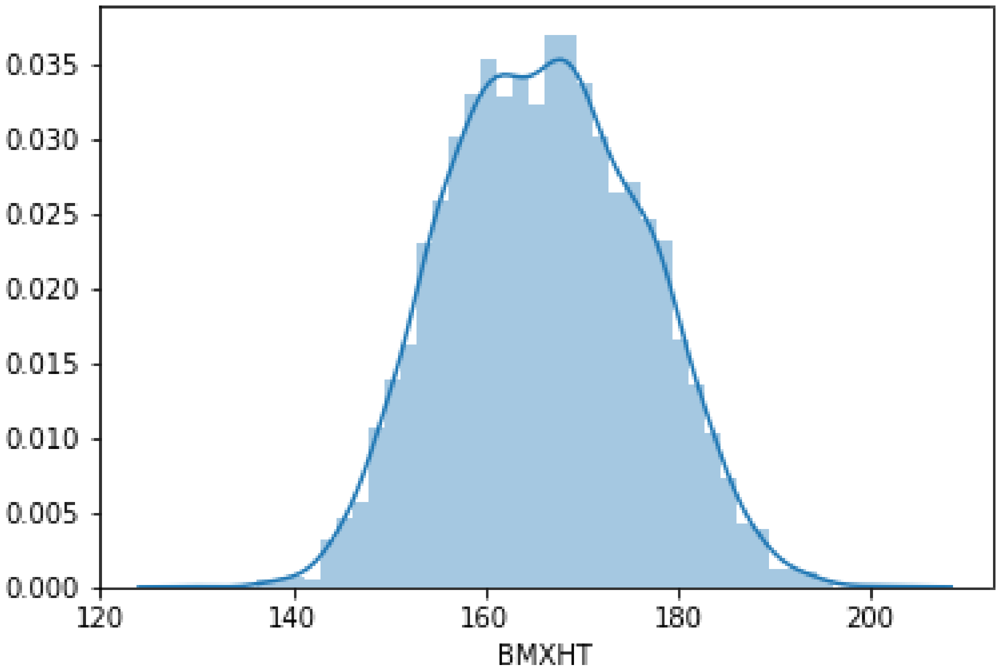
```
In [62]: x = da_clean[da.gender == "Female"]
         x["agegrp2"] = pd.cut(da_clean.RIDAGEYR, [10,20, 30, 40, 50, 60, 70, 80])
         dx = x.groupby(["agegrp2"])["DMDMarrital"].value_counts(normalize = True).unstack()
         dx = dx.fillna(0)
         print(dx)
```

```
DMDMarrital  Complicated  Divorced  In a Relationship   Married  \
agegrp2
(10, 20]        0.769231  0.000000           0.000000  0.025641
(20, 30]        0.445525  0.021401           0.021401  0.305447
(30, 40]        0.204641  0.035865           0.090717  0.544304
(40, 50]        0.125498  0.065737           0.137450  0.573705
(50, 60]        0.089362  0.057447           0.176596  0.546809
(60, 70]        0.086168  0.049887           0.192744  0.480726
(70, 80]        0.051220  0.019512           0.143902  0.317073

DMDMarrital  Not Interested  Prefer not to say    Single
agegrp2
(10, 20]           0.205128           0.000000  0.000000
(20, 30]           0.206226           0.000000  0.000000
(30, 40]           0.120253           0.000000  0.004219
(40, 50]           0.073705           0.000000  0.023904
(50, 60]           0.068085           0.002128  0.059574
(60, 70]           0.043084           0.000000  0.147392
(70, 80]           0.007317           0.000000  0.460976
```

1.2 Question 3

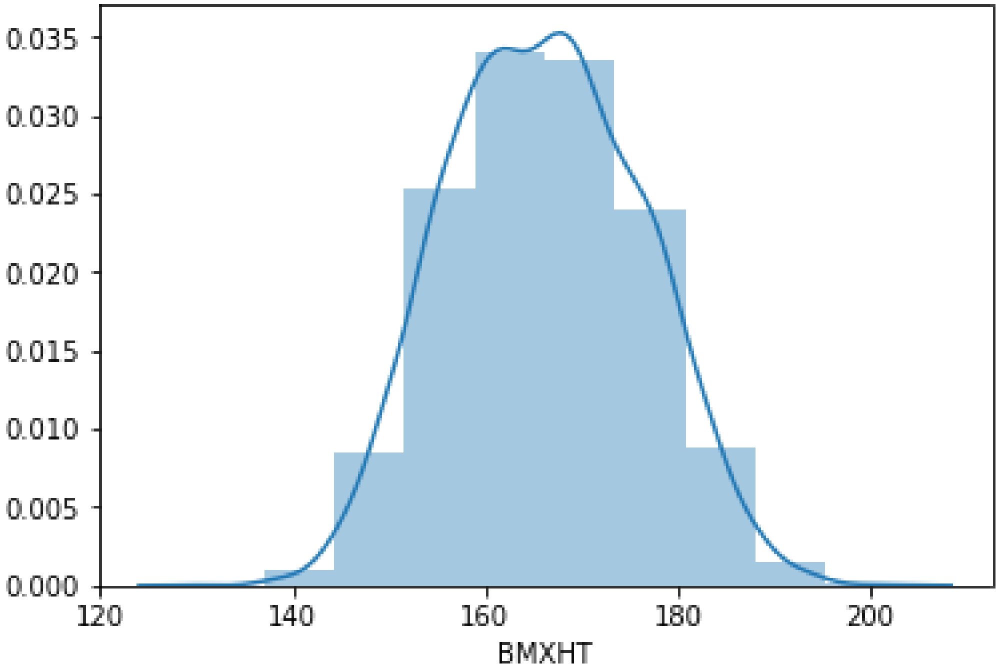
Construct a histogram of the distribution of heights using the BMXHT variable in the NHANES sample.

```
In [64]: # insert your code here

         sns.distplot(da.BMXHT.dropna())
         plt.show()
```



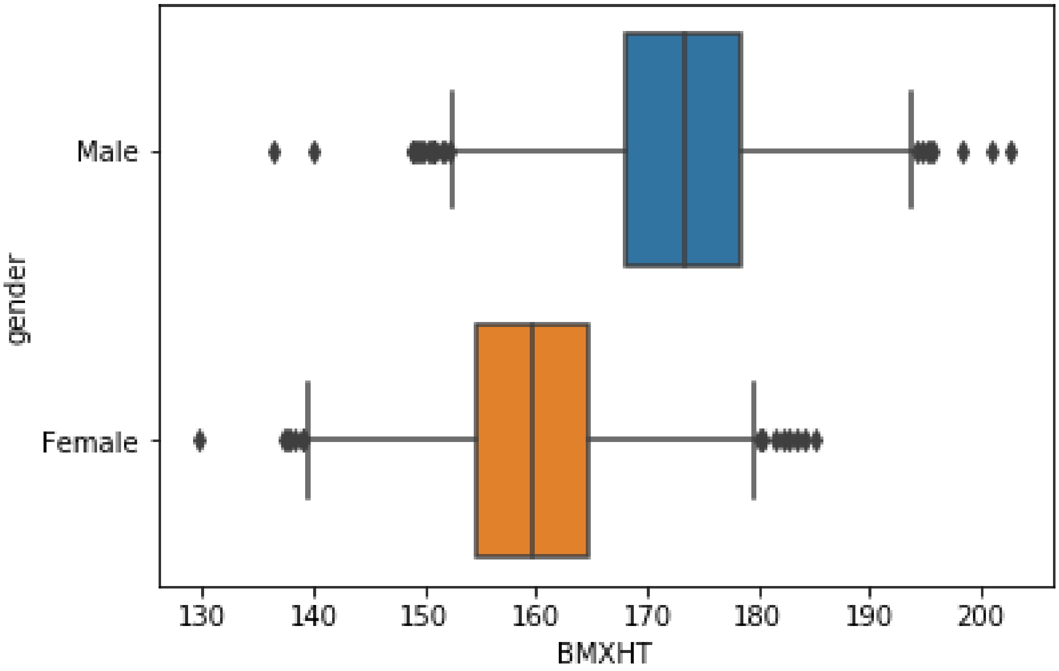
Q3a. Use the `bins` argument to distplot to produce histograms with different numbers of bins. Assess whether the default value for this argument gives a meaningful result, and comment on what happens as the number of bins grows excessively large or excessively small.

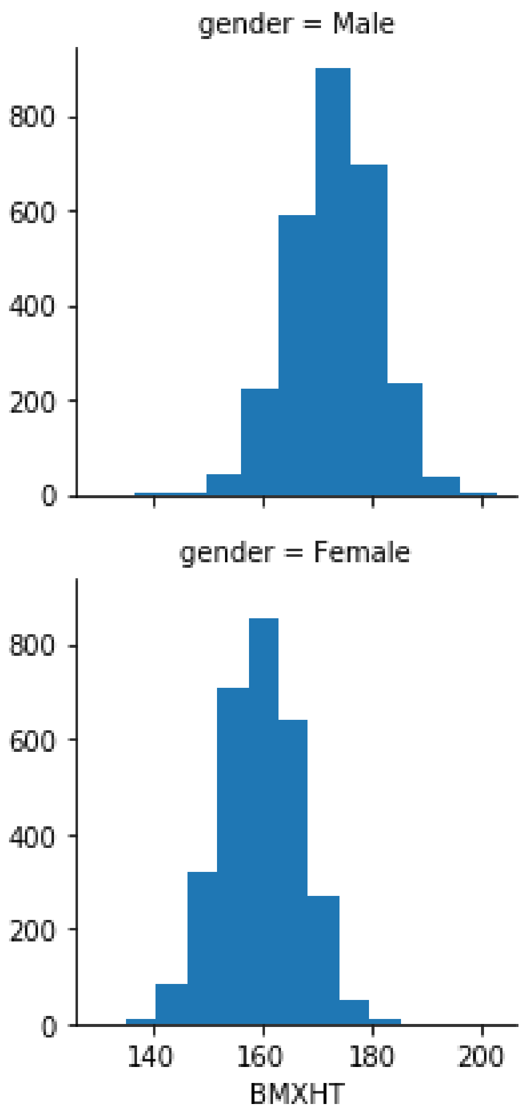
```
In [66]: sns.distplot(da.BMXHT.dropna(), bins = 10)
         plt.show()
```



Q3b. Make separate histograms for the heights of women and men, then make a side-by-side boxplot showing the heights of women and men.

```
In [68]: sns.boxplot(x = da_clean.BMXHT, y = da_clean.gender)
         hist = sns.FacetGrid(da_clean, row = 'gender')
         hist = hist.map(plt.hist, 'BMXHT')

         plt.show()
```


Male
Female
gender
130
140
150
160
170
180
190
200
BMXHT

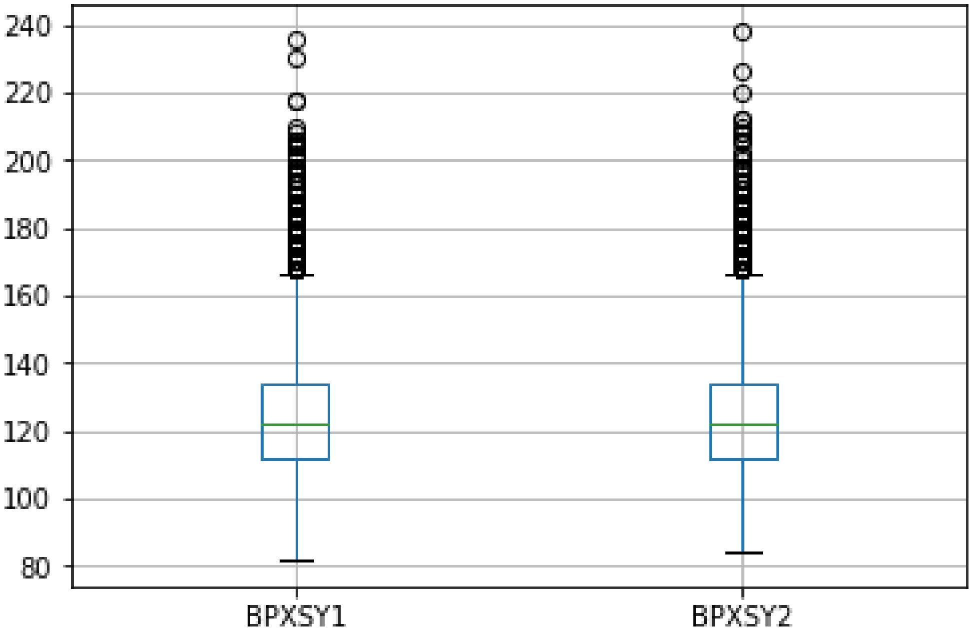


Q3c. Comment on what features, if any are not represented clearly in the boxplots, and what features, if any, are easier to see in the boxplots than in the histograms.

1.3 Question 4

Make a boxplot showing the distribution of within-subject differences between the first and second systolic blood pressure measurements (BPXSY1 and BPXSY2).

```
In [70]: x = da[['BPXSY1', 'BPXSY2']]
         x.boxplot()

Out[70]: <matplotlib.axes._subplots.AxesSubplot at 0x7f55563f9ac8>
```


240
220
200
180
160
140
120
100
80
BPXSY1
BPXSY2