Practical 4 - JosiahTeh

December 8, 2021

1 First Name : Josiah

2 Last Name : Teh

[2]:
```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
```

[3]:
```
nesarc = pd.read_csv('nesarc.csv', low_memory=False)
pd.set_option('display.float_format', lambda x:'%f'%x)
```

3 From Prac 1

4 Columns/Data used in Prac 1

[4]:
```
nesarc['S2AQ5B'] = pd.to_numeric(nesarc['S2AQ5B'], errors='coerce') #convert␣
 ↪variable to numeric
nesarc['S2AQ5D'] = pd.to_numeric(nesarc['S2AQ5D'], errors='coerce') #convert␣
 ↪variable to numeric
nesarc['S2AQ5A'] = pd.to_numeric(nesarc['S2AQ5A'], errors='coerce') #convert␣
 ↪variable to numeric
nesarc['S2BQ1B1'] = pd.to_numeric(nesarc['S2BQ1B1'], errors='coerce') #convert␣
 ↪variable to numeric
nesarc['AGE'] = pd.to_numeric(nesarc['AGE'], errors='coerce') #convert variable␣
 ↪to numeric
```

5 From Prac 2

6 A subset of nesarc data, with the following criteria

7 Age from 26 to 50

8 Beer drinking status - S2AQ5A = Y

[5]:

```
sub1=nesarc[(nesarc['AGE']>=26) & (nesarc['AGE']<=50) & (nesarc['S2AQ5A']==1)]
sub2=sub1.copy()
```

9 From Prac 2

10 SETTING MISSING DATA

[6]:

```
sub2['S2AQ5D']=sub2['S2AQ5D'].replace(99, np.nan)

sub2['S2AQ5B']=sub2['S2AQ5B'].replace(8, np.nan)
sub2['S2AQ5B']=sub2['S2AQ5B'].replace(9, np.nan)
sub2['S2AQ5B']=sub2['S2AQ5B'].replace(10, np.nan)
sub2['S2AQ5B']=sub2['S2AQ5B'].replace(99, np.nan)

sub2['S2BQ1B1']=sub2['S2BQ1B1'].replace(9, np.nan)
```

11 From Prac 2

12 Recode data

[7]:

```
recode2 = {1:30, 2:26, 3:14, 4:8, 5:4, 6:2.5, 7:1}
sub2['BEER_FEQMO']= sub2['S2AQ5B'].map(recode2)

recode3 = {2:0, 1:1}
sub2['S2BQ1B1']= sub2['S2BQ1B1'].map(recode3)
```

13 From Prac 2

14 Create secondary variables

[8]:

```
sub2['NUMBEERMO_EST']=sub2['BEER_FEQMO'] * sub2['S2AQ5D']
```

15 Draw a Line chart

16 Age vs Number of beer consumed per month (NUMBEERMO_EST)

17 a) mean number of beer consumed

18 var = mean number of beers consumed a month, grouped by age

[9]:

```
var = sub2.groupby(['AGE']).NUMBEERMO_EST.mean()
print(var)
```

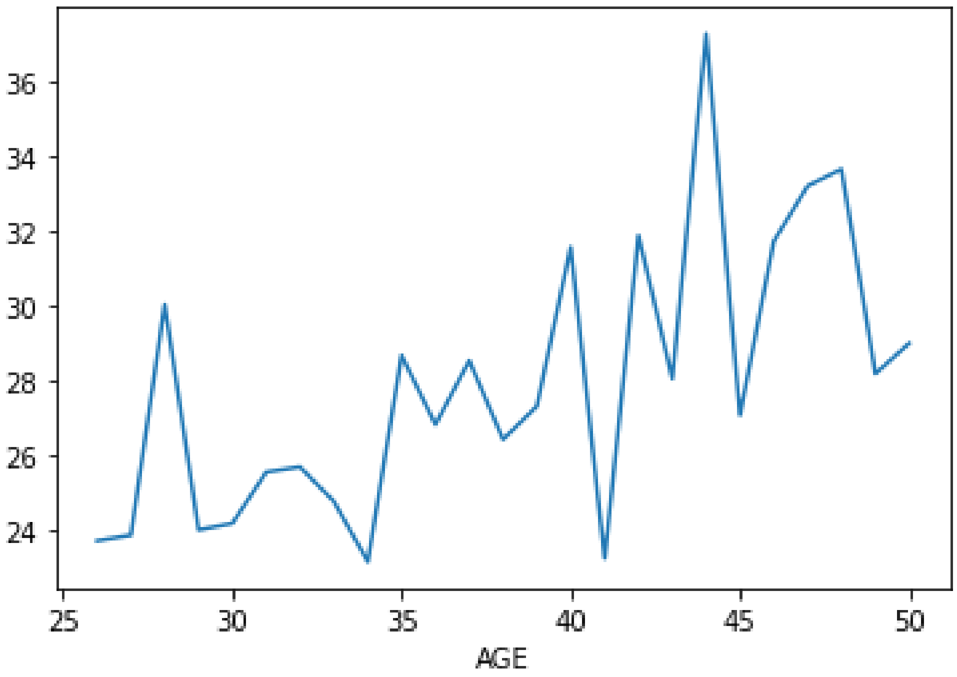
```
AGE
26    23.701357
27    23.854545
28    30.035270
29    23.994949
30    24.170530
31    25.541033
32    25.678994
33    24.761017
34    23.143713
35    28.668478
36    26.813272
37    28.530387
38    26.414773
39    27.307122
40    31.571023
41    23.233788
42    31.877676
43    28.045455
44    37.279762
45    27.067241
46    31.727799
47    33.204918
48    33.655303
49    28.177778
50    28.995614
Name: NUMBEERMO_EST, dtype: float64
```

[10]:

```
%matplotlib inline
var.plot(kind='line')
```

[10]:

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



19 b) total number of beer consumed

20 var2 = sum number of beers consumed a month, grouped by age

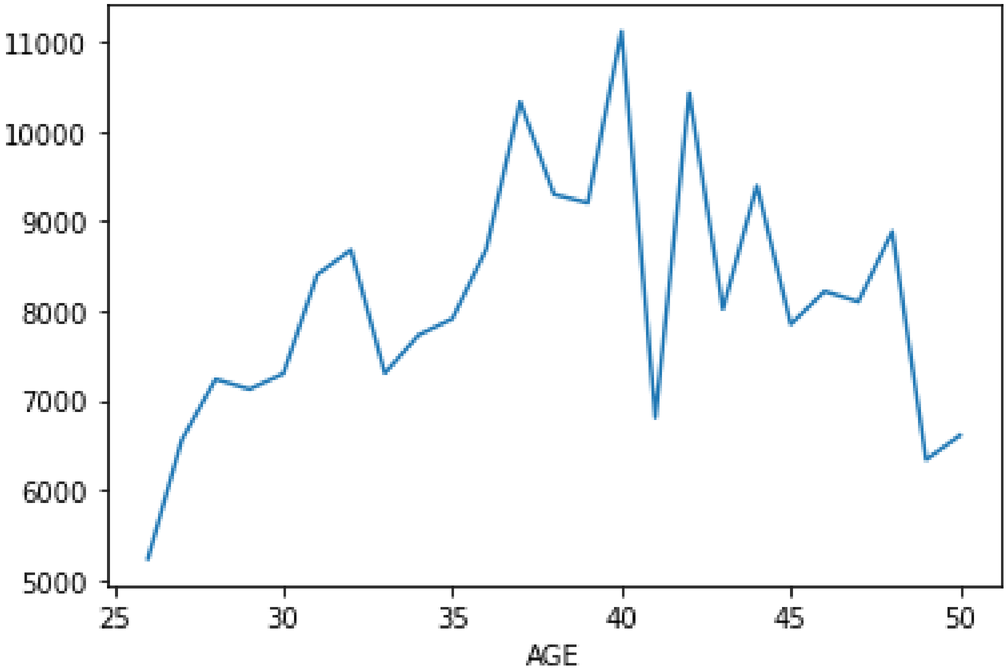
[11]:
```
var2 = sub2.groupby(['AGE']).NUMBEERMO_EST.sum()
print(var2)
```

```
AGE
26     5238.000000
27     6560.000000
28     7238.500000
29     7126.500000
30     7299.500000
31     8403.000000
32     8679.500000
33     7304.500000
34     7730.000000
35     7912.500000
36     8687.500000
37    10328.000000
38     9298.000000
39     9202.500000
40    11113.000000
```

```
41     6807.500000
42    10424.000000
43     8021.000000
44     9394.500000
45     7849.500000
46     8217.500000
47     8102.000000
48     8885.000000
49     6340.000000
50     6611.000000
Name: NUMBEERMO_EST, dtype: float64
```

```
[12]: fig = plt.figure()
      var2.plot(kind='line')
```

```
[12]: <AxesSubplot:xlabel='AGE'>
```



21 Draw a stacked Column Chart

22 x = age (AGE)

23 y = number of beers consumed per month (NUMBEERMO_EST)

24 stack is based on depedency on beer (S2BQ1B1)

25 var3 = mean number of beers consumed a month, grouped by age and beer depedency (S2BQ1B1)

[13]:

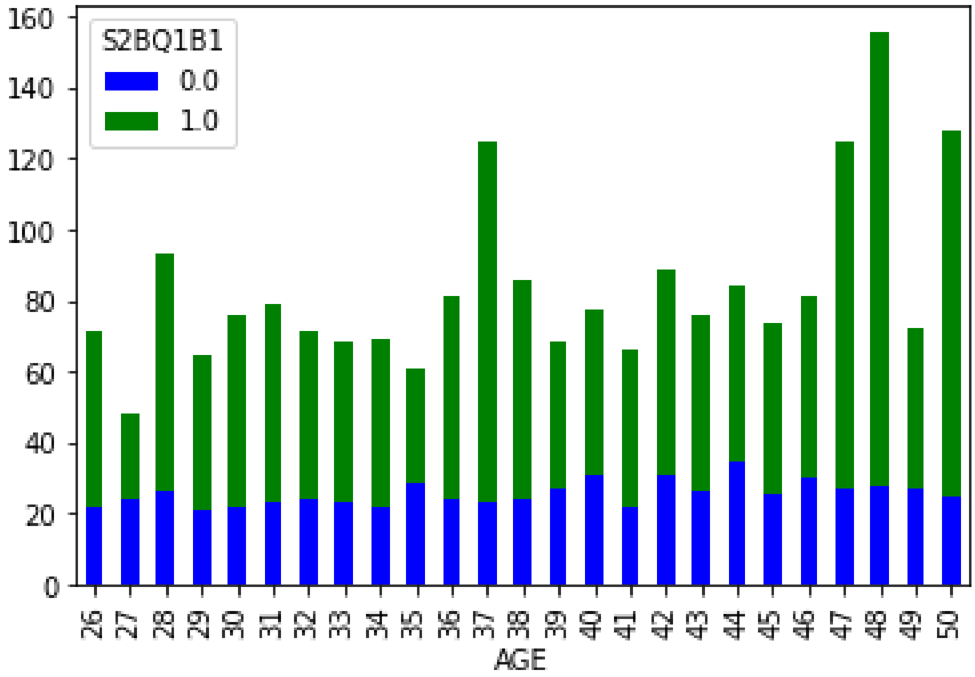
```
var3 = sub2.groupby(['AGE', 'S2BQ1B1']).NUMBEERMO_EST.mean()
print(var3)
```

```
AGE  S2BQ1B1
26   0.000000     21.449239
     1.000000     49.947368
27   0.000000     23.809524
     1.000000     24.347826
28   0.000000     26.021127
     1.000000     67.460000
29   0.000000     20.869650
     1.000000     44.078947
30   0.000000     21.530797
     1.000000     54.086957
31   0.000000     23.482026
     1.000000     55.113636
32   0.000000     23.871753
     1.000000     47.722222
33   0.000000     23.255556
     1.000000     45.075000
34   0.000000     21.732899
     1.000000     47.250000
35   0.000000     28.266537
     1.000000     32.375000
36   0.000000     24.372881
     1.000000     56.800000
37   0.000000     23.248503
     1.000000    101.240000
38   0.000000     24.274390
     1.000000     61.619048
39   0.000000     26.789308
     1.000000     41.718750
40   0.000000     30.580793
     1.000000     46.477273
41   0.000000     21.989091
```

```
          1.000000     44.441176
42        0.000000     30.563725
          1.000000     58.029412
43        0.000000     26.249071
          1.000000     49.642857
44        0.000000     34.893665
          1.000000     49.416667
45        0.000000     25.614232
          1.000000     48.083333
46        0.000000     30.041841
          1.000000     51.416667
47        0.000000     27.116438
          1.000000     97.450000
48        0.000000     27.997992
          1.000000    127.566667
49        0.000000     27.356132
          1.000000     44.636364
50        0.000000     25.077465
          1.000000    102.541667
Name: NUMBEERMO_EST, dtype: float64
```

```
[17]: var3.unstack().plot(kind='bar', stacked=True, color=['blue', 'green'],␣
       ↪grid=False)
```

```
[17]: <AxesSubplot:xlabel='AGE'>
```



26 Draw a horizontal stacked Column Chart

27 x = age (AGE)

28 y = number of beers consumed per month (NUMBEERMO_EST)

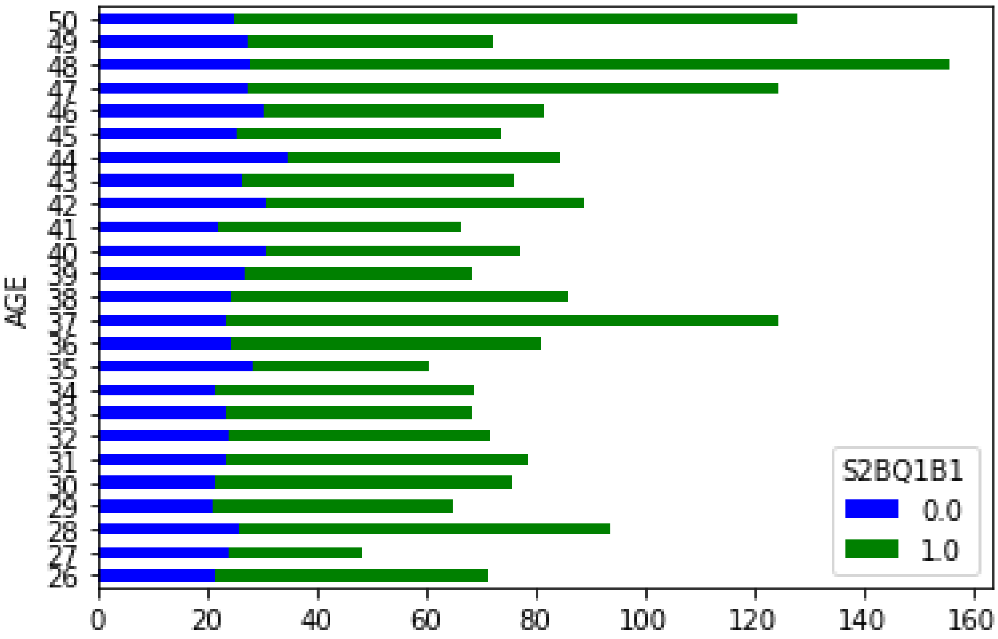
29 stack is based on depedency on beer (S2BQ1B1)

[18]:

```
var3.unstack().plot(kind='barh', stacked=True, color=['blue', 'green'],␣
 ↪grid=False)
```

[18]:

```
<AxesSubplot:ylabel='AGE'>
```



30 Draw a Pie Chart showing age (AGE) and total beer consumed a month (NUMBEERMO_EST)

31 hint use var2

[19]:

```
print(var2)
```

```
AGE
26      5238.000000
27      6560.000000
```

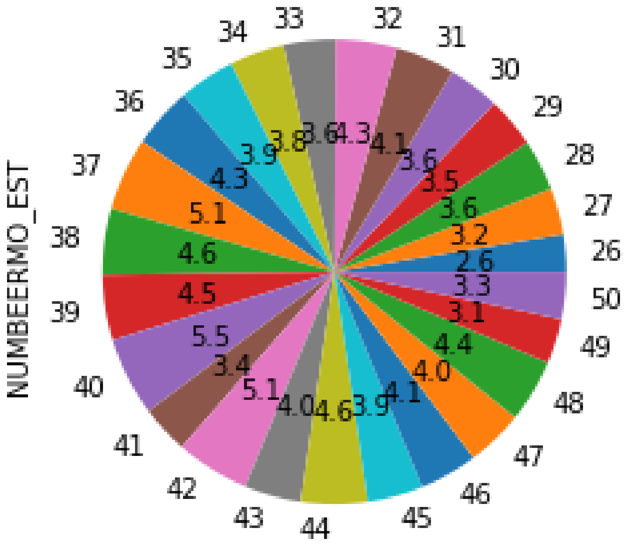
```
28     7238.500000
29     7126.500000
30     7299.500000
31     8403.000000
32     8679.500000
33     7304.500000
34     7730.000000
35     7912.500000
36     8687.500000
37    10328.000000
38     9298.000000
39     9202.500000
40    11113.000000
41     6807.500000
42    10424.000000
43     8021.000000
44     9394.500000
45     7849.500000
46     8217.500000
47     8102.000000
48     8885.000000
49     6340.000000
50     6611.000000
Name: NUMBEERMO_EST, dtype: float64
```

[20]:

```
fig = plt.figure()
var2.plot(kind='pie',autopct='%.1f')
# code for pie chart
```

[20]:

```
<AxesSubplot:ylabel='NUMBEERMO_EST'>
```



32 Draw a Violin Plot for age (AGE) and income (S1Q10A)

33 convert income (S1Q10A) to numeric

[21]:

```
sub2['S1Q10A'] = pd.to_numeric(nesarc['S1Q10A']) #convert variable to numeric
```

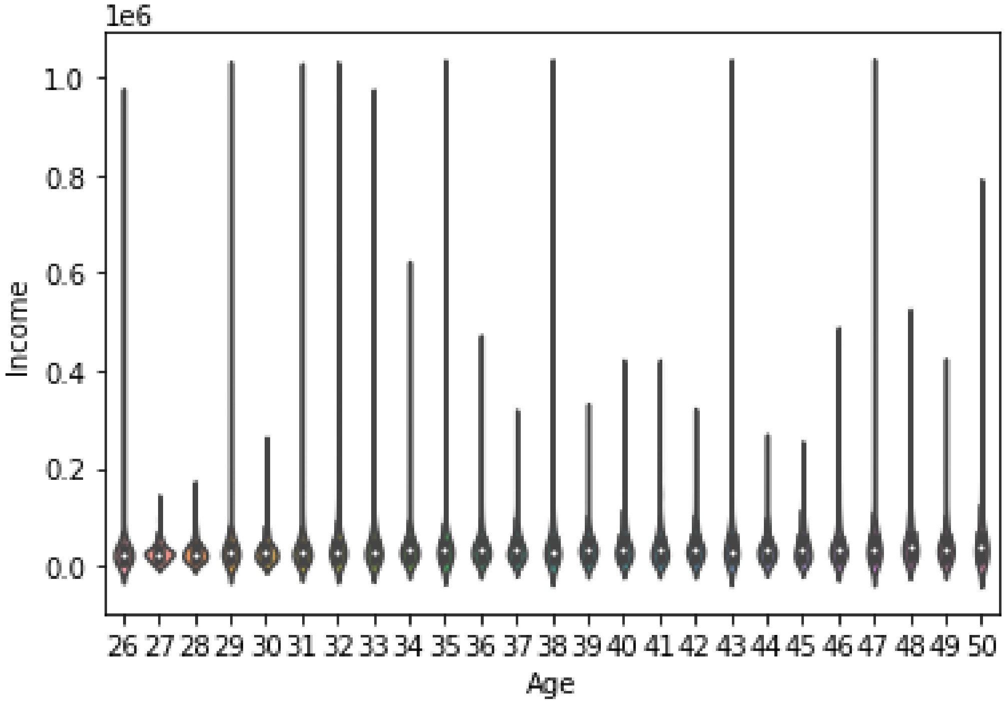
34 Plot violin plot

[22]:

```
fig = plt.figure()
sns.violinplot(x='AGE', y='S1Q10A', data=sub2)
plt.xlabel('Age')
plt.ylabel('Income')
```

[22]:

```
Text(0, 0.5, 'Income')
```



35 Draw a HeatMap for Ethnicity and Carton of Beer consumed per month, based on depedency on beer

36 Rename Race - From Module 4

[23]:
```
sub2['ETHRACE2A'] = sub2['ETHRACE2A'].astype('category')

sub2['ETHRACE2A']=sub2['ETHRACE2A'].cat.rename_categories(["White", "Black",␣
 ↪"NatAm", "Asian", "Hispanic"])
```

37 Create a new variable CARTON_ADAY using CARTON_ADAY function provided

[25]:
```
def CARTON_ADAY (row):
   if row['BEER_FEQMO'] >= 30 :
      return 1
   elif row['BEER_FEQMO'] < 30 :
      return 0

sub2['CARTON_ADAY'] = sub2.apply (lambda row:CARTON_ADAY (row),axis=1)
```

38 Print the size of CARTON_ADAY, grouped by category

[26]:
```
c4= sub2.groupby('CARTON_ADAY').size()
print(c4)
```

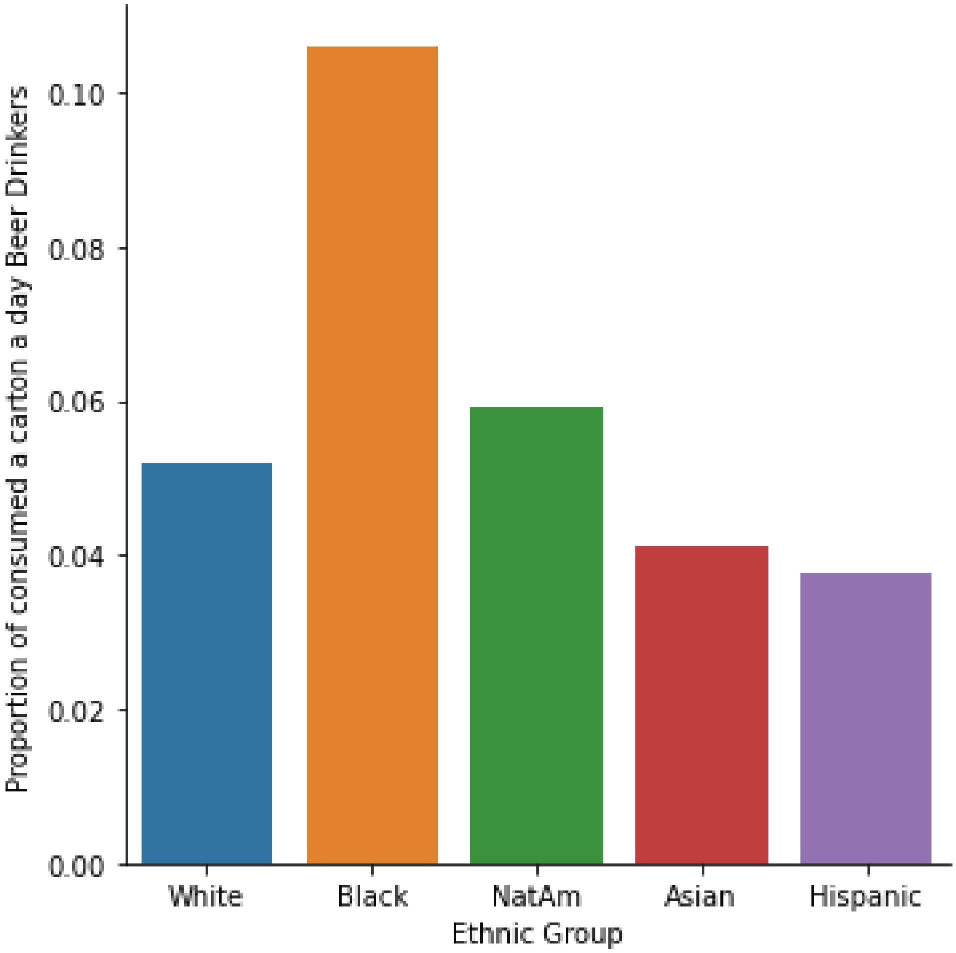
```
CARTON_ADAY
0.000000    6897
1.000000     417
dtype: int64
```

39 Draw bar chart to show relationship between race (ETHRACE2A) and CARTON_ADAY

[28]:
```
%matplotlib inline
sns.factorplot(x='ETHRACE2A', y='CARTON_ADAY', data=sub2, kind='bar', ci=None)
plt.xlabel('Ethnic Group')
plt.ylabel('Proportion of consumed a carton a day Beer Drinkers')
```

```
C:\Users\Admin\anaconda3\lib\site-packages\seaborn\categorical.py:3714:
UserWarning: The `factorplot` function has been renamed to `catplot`. The
original name will be removed in a future release. Please update your code. Note
that the default `kind` in `factorplot` (`'point'`) has changed `'strip'` in
`catplot`.
  warnings.warn(msg)
```

```
[28]: Text(0.42499999999999716, 0.5, 'Proportion of consumed a carton a day Beer
      Drinkers')
```



40 Make copy of just race (ETHRACE2A) and CARTON_ADAY

```
[29]: sub3 = sub2[['ETHRACE2A','CARTON_ADAY']].copy()
      sub3.head()
```

```
[29]:    ETHRACE2A  CARTON_ADAY
      1   Hispanic          NaN
      8      White          NaN
      12     Asian     0.000000
      16     White          NaN
      24  Hispanic          NaN
```

41 Create pivot table of race (ETHRACE2A) and CARTON_ADAY

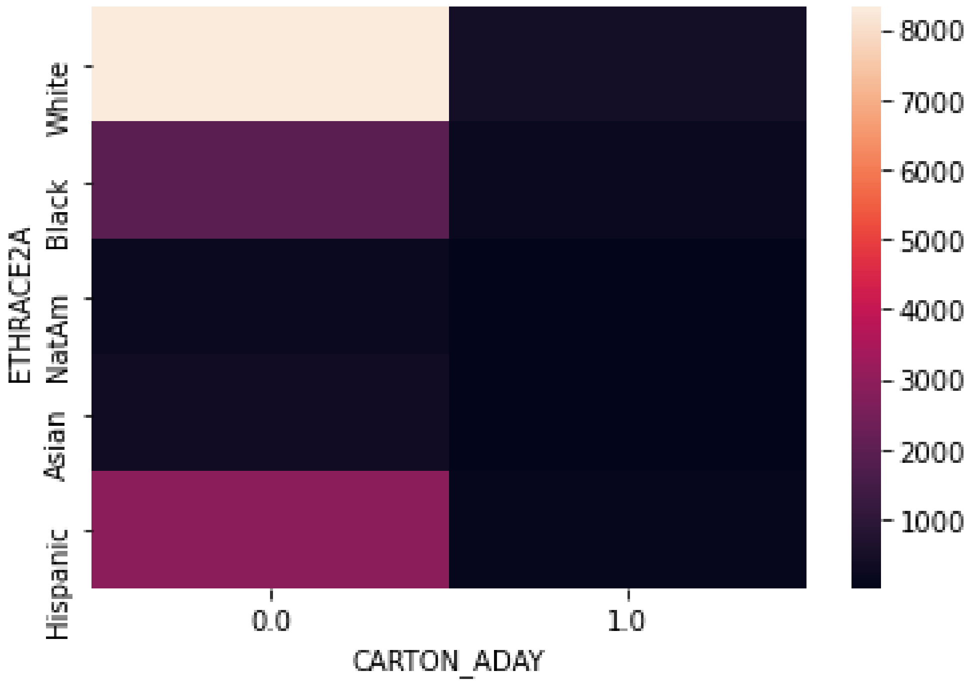
[30]:
```
table = pd.pivot_table(sub3, index=['ETHRACE2A'], columns=['CARTON_ADAY'],␣
 ↪aggfunc=np.size)
print(table)
```

```
CARTON_ADAY  0.000000  1.000000
ETHRACE2A
White            8312       456
Black            1972       234
NatAm             222        14
Asian             374        16
Hispanic         2914       114
```

42 Draw heat map

[31]:
```
fig = plt.figure()
sns.heatmap(table)
```

[31]:
```
<AxesSubplot:xlabel='CARTON_ADAY', ylabel='ETHRACE2A'>
```



43 Draw a bubble Chart

44 Read in gapminder.csv

[32]:
```
pd.set_option('display.float_format', lambda x:'%.2f'%x)

gapminder = pd.read_csv('gapminder.csv', low_memory=False)
gapminder.head()
```

[32]:
```
         country   incomeperperson alcconsumption armedforcesrate  \
0  Afghanistan                              .03        .5696534
1      Albania  1914.99655094922           7.29       1.0247361
2      Algeria  2231.99333515006            .69        2.306817
3      Andorra  21943.3398976022          10.17
4       Angola  1381.00426770244           5.57       1.4613288

  breastcancerper100th      co2emissions  femaleemployrate hivrate  \
0                 26.8          75944000  25.6000003814697
1                 57.4  223747333.333333  42.0999984741211
2                 23.5  2932108666.66667  31.7000007629394      .1
3
4                 23.1         248358000  69.4000015258789       2

    internetuserate lifeexpectancy     oilperperson polityscore  \
0  3.65412162280064         48.673                            0
1  44.9899469578783         76.918                            9
2  12.5000733055148         73.131  .42009452521537           2
3                81
4  9.99995388324075         51.093                           -2

  relectricperperson   suicideper100th         employrate urbanrate
0                     6.68438529968262  55.7000007629394     24.04
1   636.341383366604  7.69932985305786  51.4000015258789     46.72
2   590.509814347428   4.8487696647644              50.5     65.22
3                     5.36217880249023                       88.92
4   172.999227388199  14.5546770095825  75.6999969482422      56.7
```

45 Convert internetuserate, urbanrate and incomeperperson to numeric

[33]:
```
gapminder['internetuserate'] = pd.
 ↪to_numeric(gapminder['internetuserate'],errors='coerce')
gapminder['urbanrate'] = pd.to_numeric(gapminder['urbanrate'],errors='coerce')
gapminder['incomeperperson'] = pd.
 ↪to_numeric(gapminder['incomeperperson'],errors='coerce')
```

[34]:

```
gapminder_clean=gapminder.dropna()
```

46 Draw a bubble Chart

47 x = urbanrate

48 y = income per person

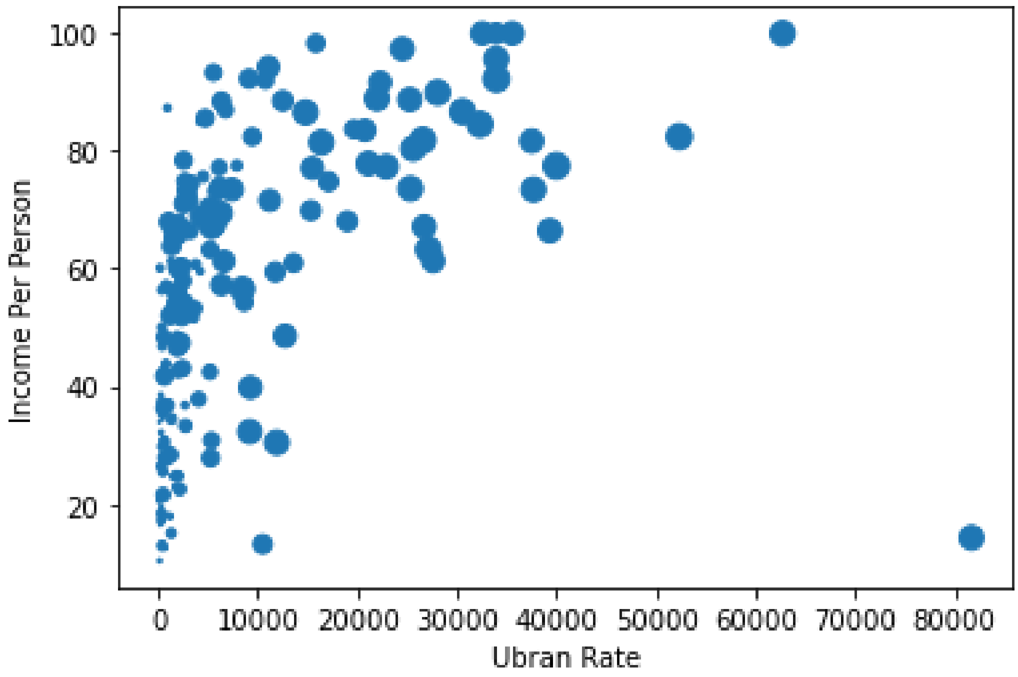
49 bubble size = internetuserate

[35]:

```
%matplotlib inline
fig = plt.figure()
plt.scatter(gapminder_clean['incomeperperson'], gapminder_clean['urbanrate'],␣
 ↪gapminder_clean['internetuserate'])
plt.xlabel('Ubran Rate')
plt.ylabel('Income Per Person')
```

[35]:

```
Text(0, 0.5, 'Income Per Person')
```


Income Per Person
Ubran Rate