In [1]:

```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
```

In [2]:

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

From Prac 1

Columns/Data used in Prac 1

In [3]:

```
nesarc['S2AQ5B'] = pd.to_numeric(nesarc['S2AQ5B'], errors='coerce') #convert variable t
o numeric
nesarc['S2AQ5D'] = pd.to_numeric(nesarc['S2AQ5D'], errors='coerce') #convert variable t
o numeric
nesarc['S2AQ5A'] = pd.to_numeric(nesarc['S2AQ5A'], errors='coerce') #convert variable t
o 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 nume
ric
```

From Prac 2

A subset of nesarc data, with the following criteria

Age from 26 to 50

Beer drinking status - S2AQ5A = Y

In [4]:

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

From Prac 2

SETTING MISSING DATA

In [5]:

```
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)
```

From Prac 2

Recode data

In [6]:

```
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)
```

From Prac 2

Create secondary variables

In [7]:

```
# A secondary variable multiplying the number of days beer consumed/month and the appro
x number of
# beer consumed/day
sub2['NUMBEERMO_EST']=sub2['BEER_FEQMO'] * sub2['S2AQ5D']
```

Draw a Line chart

Age vs Number of beer consumed per month (NUMBEERMO_EST)

a) mean number of beer consumed

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

In [8]:

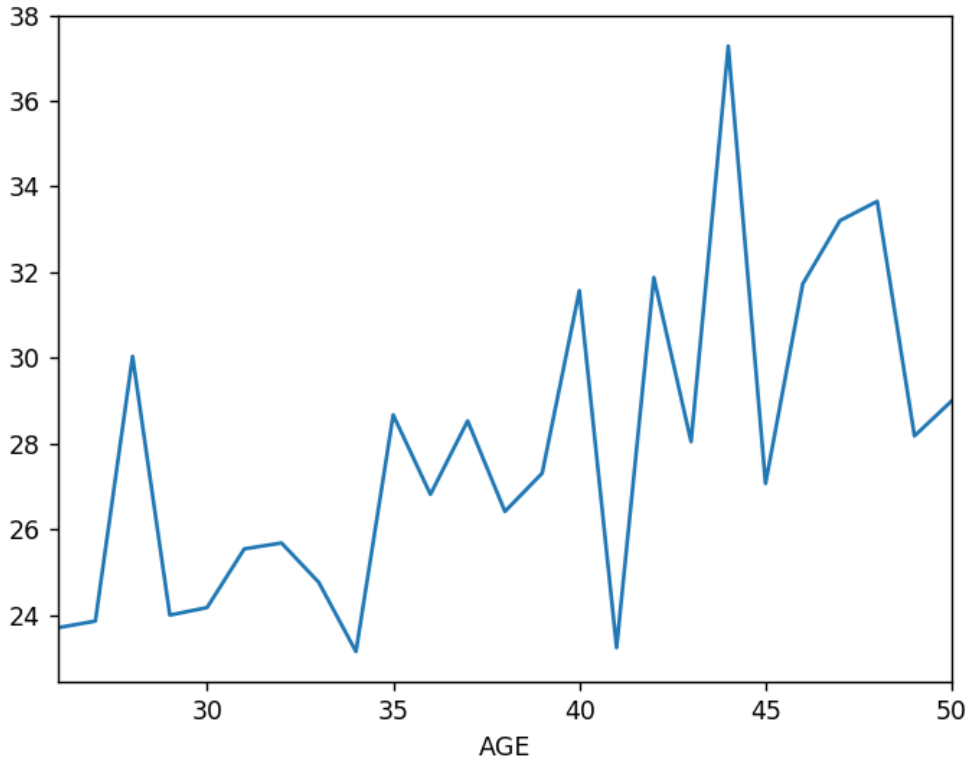
```
var = sub2.groupby(['AGE']).NUMBEERMO_EST.mean()
print(var)
```

```
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
```

Result shows the average amount of beer interviewee drinks, from age 26 to 50.
Most interviewees from this age group drinks 20 to 30 beers per month.

In [30]:

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



Line graph that describes how many bottles of beer interviewee from age 26 to 50 drank every month.

Out[30]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x6c1b8552b0>
```

b) total number of beer consumed

var2 = sum number of beers consumed a month,

In [10]:

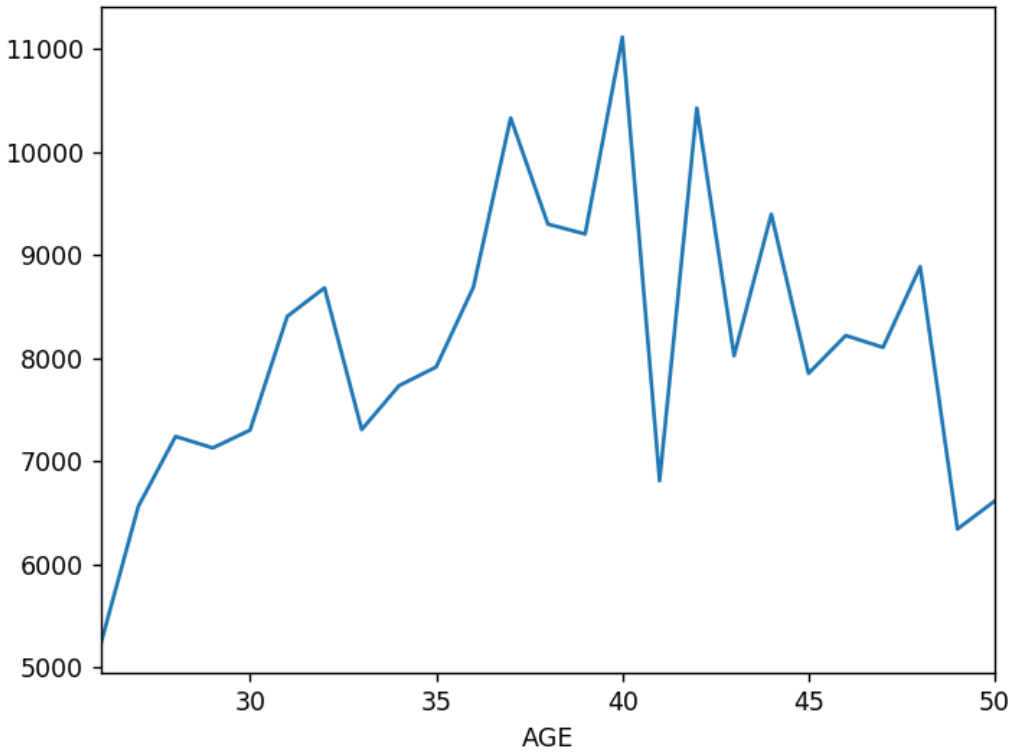
```
var2 = sub2.groupby(['AGE']).NUMBEERMO_EST.sum()
print(var2)
```

```
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
```

Result shows the total amount of beer interviewee drinks every month, from age 26 to 50.
Interviewees that are 40, 42, and 37 years old rank top three with 11113, 10424, and 10328 bottles of beer per month.
Age 26 group has the lowest amount, 5238 bottles consumed every month.

In [31]:

```
fig = plt.figure()
var2.plot(kind='line')
```



Line graph that shows how many bottles of beer each age group drank every month, from age 26 to age 50

Out[31]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x6c06ab77f0>
```

Draw a stacked Column Chart

x = age (AGE)

y = number of beers consumed per month (NUMBEERMO_EST)

stack is based on depedency on beer (S2BQ1B1)

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

In [12]:

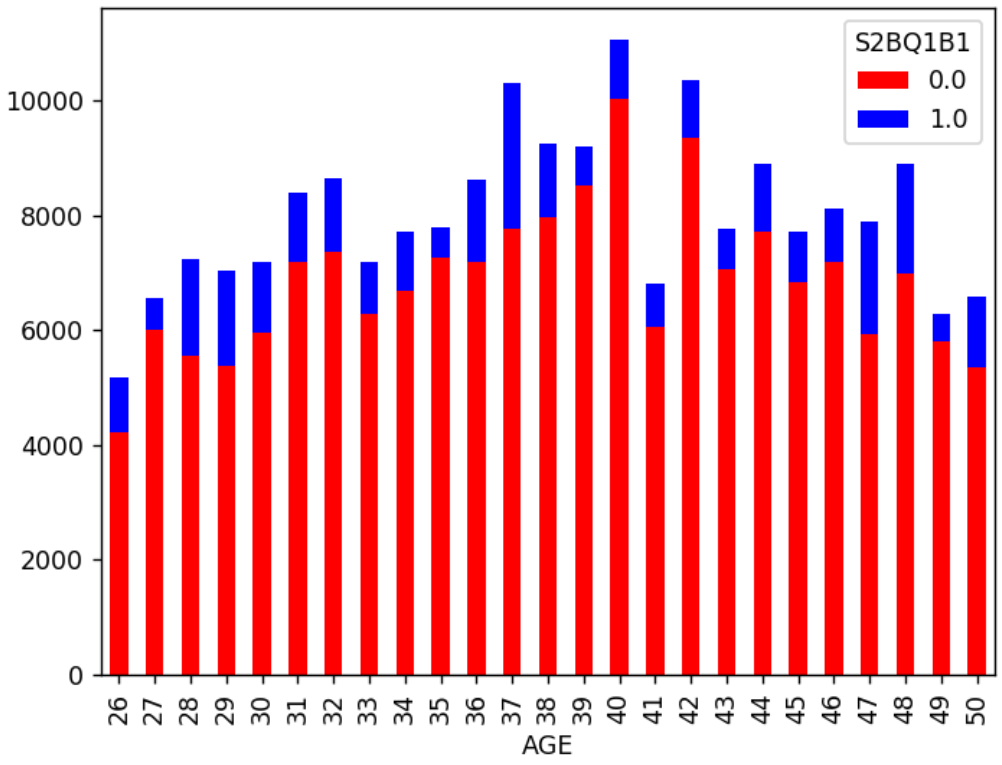
```
var3 = sub2.groupby(['AGE','S2BQ1B1']).NUMBEERMO_EST.sum()
print(var3)
```

```
AGE  S2BQ1B1
26   0.000000    4225.500000
     1.000000     949.000000
27   0.000000    6000.000000
     1.000000     560.000000
28   0.000000    5542.500000
     1.000000    1686.500000
29   0.000000    5363.500000
     1.000000    1675.000000
30   0.000000    5942.500000
     1.000000    1244.000000
31   0.000000    7185.500000
     1.000000    1212.500000
32   0.000000    7352.500000
     1.000000    1288.500000
33   0.000000    6279.000000
     1.000000     901.500000
34   0.000000    6672.000000
     1.000000    1039.500000
35   0.000000    7264.500000
     1.000000     518.000000
36   0.000000    7190.000000
     1.000000    1420.000000
37   0.000000    7765.000000
     1.000000    2531.000000
38   0.000000    7962.000000
     1.000000    1294.000000
39   0.000000    8519.000000
     1.000000     667.500000
40   0.000000   10030.500000
     1.000000    1022.500000
41   0.000000    6047.000000
     1.000000     755.500000
42   0.000000    9352.500000
     1.000000     986.500000
43   0.000000    7061.000000
     1.000000     695.000000
44   0.000000    7711.500000
     1.000000    1186.000000
45   0.000000    6839.000000
     1.000000     865.500000
46   0.000000    7180.000000
     1.000000     925.500000
47   0.000000    5938.500000
     1.000000    1949.000000
48   0.000000    6971.500000
     1.000000    1913.500000
49   0.000000    5799.500000
     1.000000     491.000000
50   0.000000    5341.500000
     1.000000    1230.500000
Name: NUMBEERMO_EST, dtype: float64
```

Result shows the sum amount of beer interviewees drink per month, from age 26 to 50.
And within each age group, interviewees are divided by whether they are alcohol dependent.

In [32]:

```
var3.unstack().plot(kind='bar', stacked=True,  color=['red','blue'], grid=False)
```



Bar plot that describes the sum amount of beer drank every month, by interviewees from age 26 to age 50, and whether they are alcohol independent.

Out[32]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x6c0885bd68>
```

Draw a horizontal stacked Column Chart

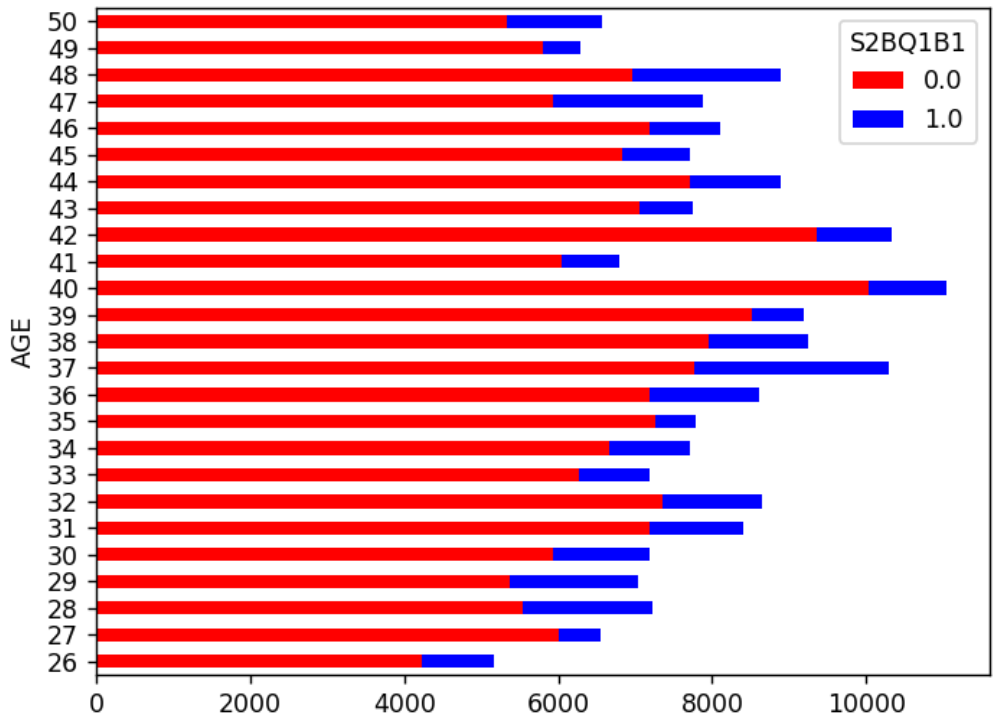
x = age (AGE)

y = number of beers consumed per month (NUMBEERMO_EST)

stack is based on depedency on beer (S2BQ1B1)

In [33]:

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



Same bar plot that is draw horizontally (switched x, y axes)

Out[33]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x6c14836588>
```

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

hint use var2

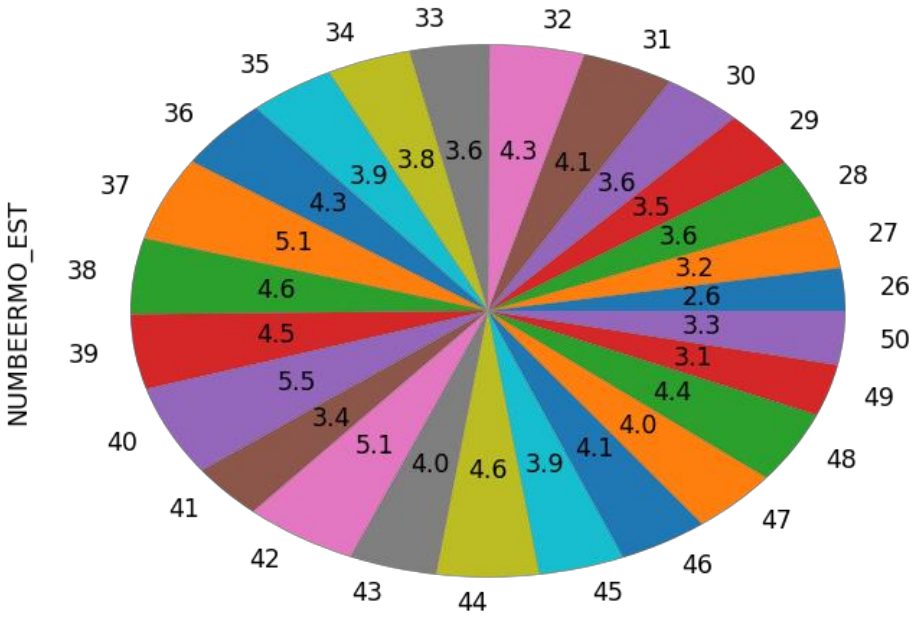
In [15]:

```
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
```

In [34]:

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



A pie chart that shows the amount of beer drank by interviewees within age 26 to 50, what percentage each age group is compare to total.

Out[34]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x6c14d07e10>
```

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

convert income (S1Q10A) to numeric

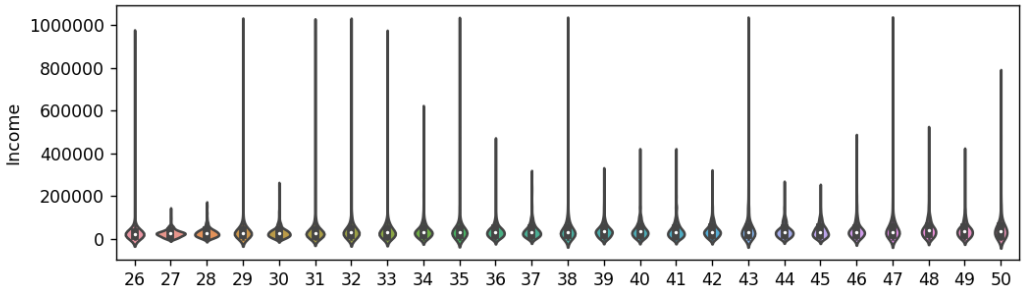
In [17]:

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

Plot violin plot

In [35]:

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



A violin plot that shows incomes from age 26 to age 50, we can see a slight increase in income as interviewee grew older.

Out[35]:

Text(0,0.5,'Income')

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

Rename Race - From Module 4

In [19]:

```
# you can rename categorical variable values for graphing if original values are not in
formative
# first change the variable format to categorical if you haven't already done so
sub2['ETHRACE2A'] = sub2['ETHRACE2A'].astype('category')

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

Create a new variable CARTON_ADAY using CARTON_ADAY function provided

In [20]:

```
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)
```

Print the size of CARTON_ADAY, grouped by category

In [21]:

```
c4= sub2.groupby('CARTON_ADAY').size()
print(c4)
```

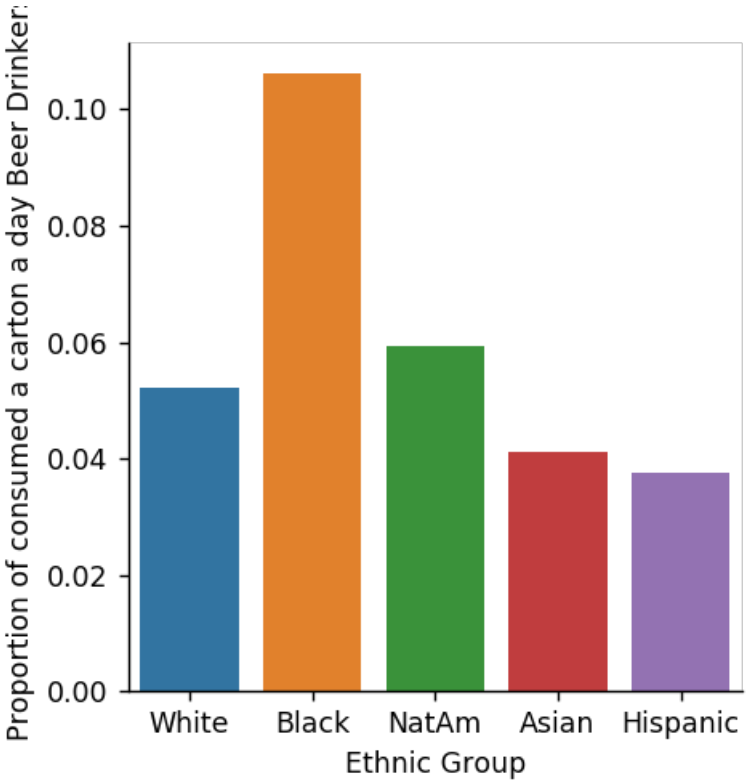
```
CARTON_ADAY
0.000000    6897
1.000000     417
dtype: int64
```

Results shows whether interviewee drank more than a carton beer per day.

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

In [36]:

```
# bivariate bar graph C->C
%matplotlib notebook
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')
```



Result shows for 5 different ethnic groups: white, black, Native American, Asian, and Hispanic, what percentage of that ethnic group drank more than a carton of beer everyday.
Black people has the highest percentage - 0.105, Asians and Hispanics are two of lowest, around 0.04.

Out[36]:

```
Text(0.694444,0.5,'Proportion of consumed a carton a day Beer Drinkers')
```

Make copy of just race (ETHRACE2A) and CARTON_ADAY

In [23]:

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

Out[23]:

	ETHRACE2A	CARTON_ADAY
1	Hispanic	nan
8	White	nan
12	Asian	0.000000
16	White	nan
24	Hispanic	nan

The first 5 rows of ethnic groups and carton beer per day drinking values.

Create pivot table of race (ETHRACE2A) and CARTON_ADAY

In [24]:

```
table = pd.pivot_table(sub3, index=['ETHRACE2A'], columns=['CARTON_ADAY'], aggfunc=np.s
ize)
print(table)
```

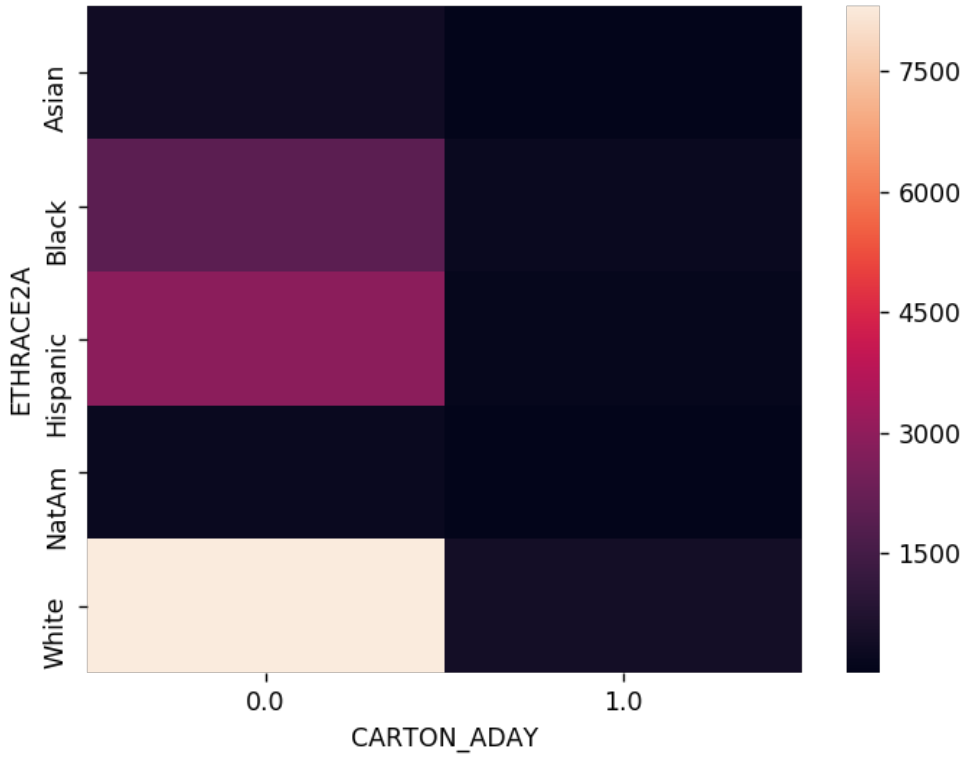
```
CARTON_ADAY  0.000000  1.000000
ETHRACE2A
Asian             374        16
Black            1972       234
Hispanic         2914       114
NatAm             222        14
White            8312       456
```

A pivot table that describe five ethnic groups and number of a carton of beer drinkers for each group.

Draw heat map

In [37]:

```
fig = plt.figure()
sns.heatmap(table)
```



A heat map that shows number of a-carton-beer-per-day drinkers among five different ethnic groups.

Out[37]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x6c05e2a9e8>
```

Draw a bubble Chart

Read in gapminder.csv

In [26]:

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

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

Out[26]:

	country	incomeperperson	alcconsumption	armedforcesrate	breastcancerper
0	Afghanistan		.03	.5696534	26.8
1	Albania	1914.99655094922	7.29	1.0247361	57.4
2	Algeria	2231.99333515006	.69	2.306817	23.5
3	Andorra	21943.3398976022	10.17		
4	Angola	1381.00426770244	5.57	1.4613288	23.1

Convert internetuserate, urbanrate and incomeperperson to numeric

First five rows of a table that describes several information, such as income rate, breast cancer rate, alcohol consumption, etc. among different countries

In [27]:

```
gapminder['internetuserate'] = pd.to_numeric(gapminder['internetuserate'],errors='coerc
e')
gapminder['urbanrate'] = pd.to_numeric(gapminder['urbanrate'],errors='coerce')
gapminder['incomeperperson'] = pd.to_numeric(gapminder['incomeperperson'],errors='coerc
e')
```

In [28]:

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

Draw a bubble Chart

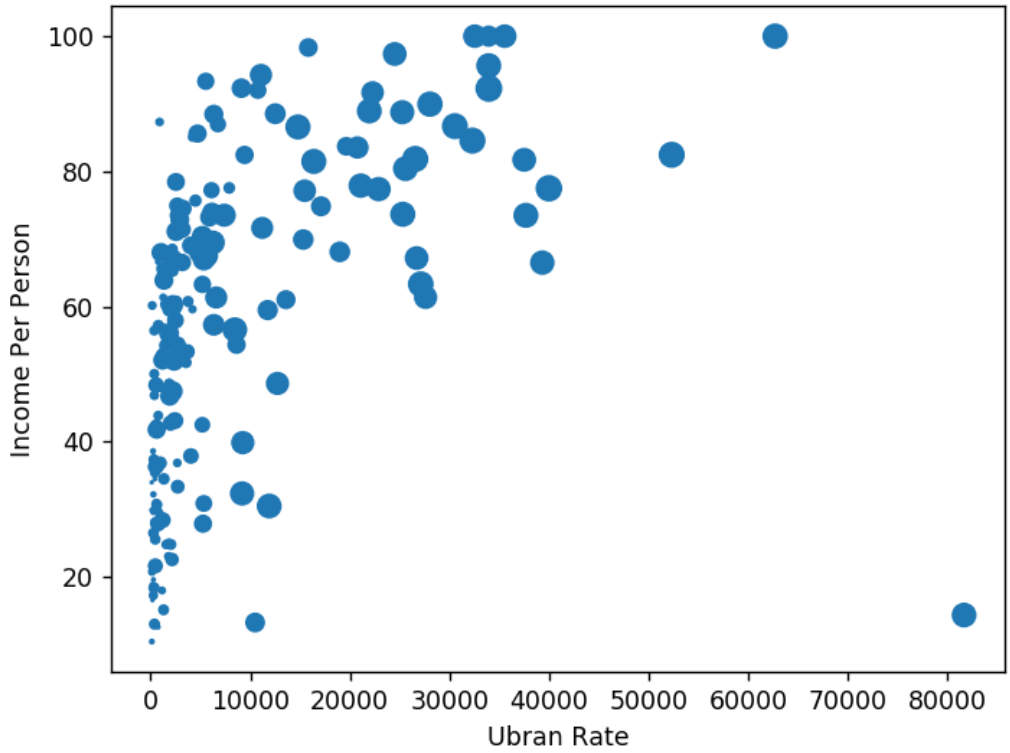
x = urbanrate

y = income per person

bubble size = internetuserate

In [29]:

```
# x = internetuserate
# y = incomeperperson
# Added third variable income as size of the bubble
%matplotlib notebook
fig = plt.figure()
plt.scatter(gapminder_clean['incomeperperson'],gapminder_clean['urbanrate'], s=gapminde
r_clean['internetuserate'])
plt.xlabel('Ubran Rate')
plt.ylabel('Income Per Person')
```



A bubble chart that describes relation between urban rate and income per person.

We can observe there is a positive correlation between two variables.

Out[29]:

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