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)
```

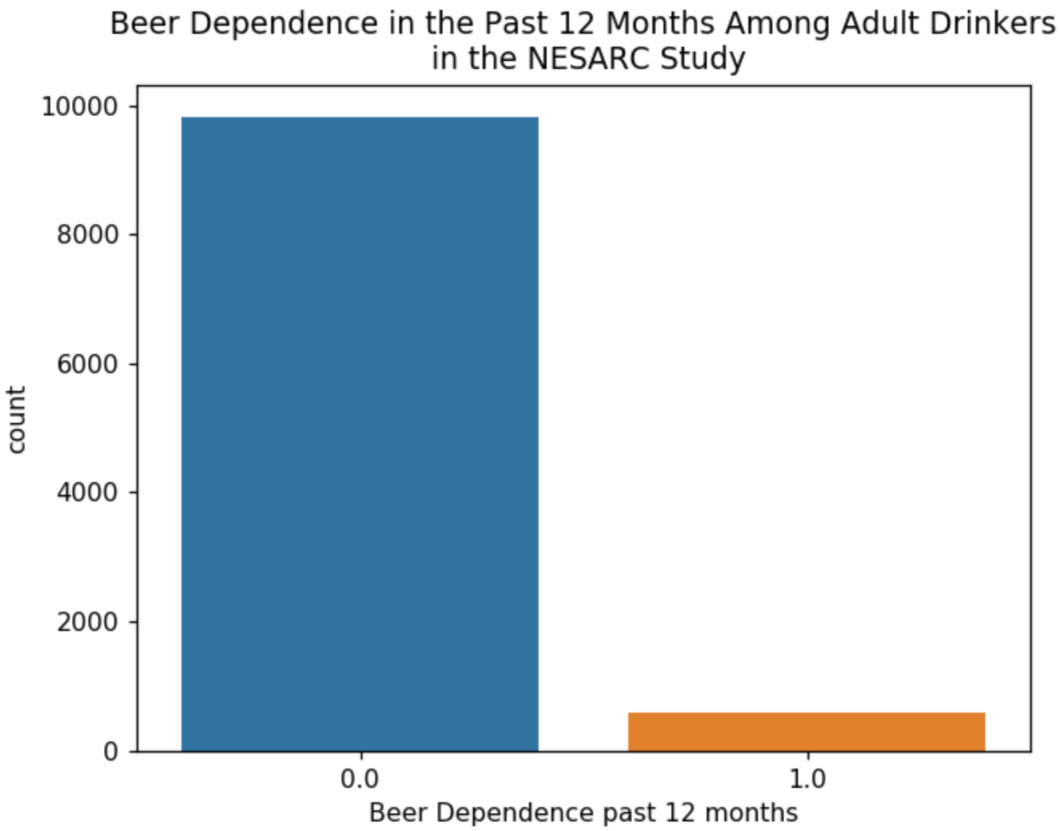
Plot bar chart for S2BQ1B1

In [7]:

```
sub2["S2BQ1B1"] = sub2["S2BQ1B1"].astype('category')
```

In [8]:

```
%matplotlib notebook
sns.countplot(x="S2BQ1B1", data=sub2)
plt.xlabel('Beer Dependence past 12 months')
plt.title('Beer Dependence in the Past 12 Months Among Adult Drinkers'+ '\n' + ' in the
 NESARC Study')
```



The bar chart shows in the past 12 months, majority of adult drinkers do not have beer dependence, and adults who do have beer dependency are less than 1000, which is less than 10% of sample size.

Out[8]:

```
Text(0.5,1,'Beer Dependence in the Past 12 Months Among Adult Drinkers\n i
n the NESARC Study')
```

Visualizing Quantitative Variable - histogram

From Prac 2

Create a secondary variable to estimate the number of beer consumed per month

NUMBEERMO_EST

In [9]:

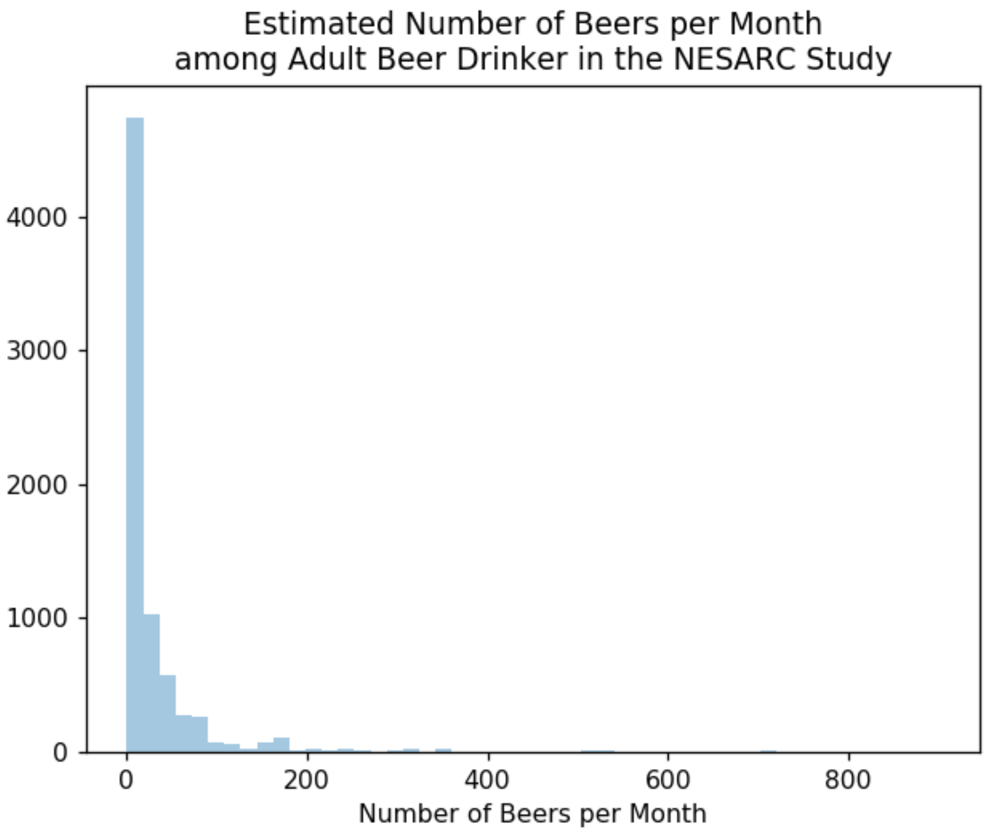
```
# A secondary variable multiplying the number of beers comsumed and the approx number o
f beers consumed/day
sub2['NUMBEERMO_EST']=sub2['BEER_FEQMO'] * sub2['S2AQ5D']
```

Visualise the number of beers consumed per month (NUMBEERMO_EST) using a histogram

In [5]:

```
%matplotlib notebook

#Univariate histogram for quantitative variable:
sns.distplot(sub2["NUMBEERMO_EST"].dropna(), kde=False);
plt.xlabel('Number of Beers per Month')
plt.title('Estimated Number of Beers per Month' + '\n' + 'among Adult Beer Drinker in t
he NESARC Study')
```



The histogram shows among all interviewees, most people do not drink any beer at all. And for those interviewees who do drink, most will not drink more than 200 bottles per month, with few except outliers.

Out[10]:

```
Text(0.5,1,'Estimated Number of Beers per Month\namong Adult Beer Drinker
in the NESARC Study')
```

Calculate the spread and centre of NUMBEERMO_EST

Use describe()

In [6]:

```
# standard deviation and other descriptive statistics for quantitative variables
print('describe number of beers drinking per month')
desc1 = sub2['NUMBEERMO_EST'].describe()
print (desc1)
```

```
describe number of beers drinking per month
count   7303.000000
mean      27.765713
std       49.201312
min        1.000000
25%        4.000000
50%       12.000000
75%       28.000000
max      900.000000
Name: NUMBEERMO_EST, dtype: float64
```

The dataset description indicates there are 7303 interviewees, and they drink 27.77 bottles of beer per month on average. There is a lot of variance in the observed data, as standard deviation is 49.2. The minimum and maximum value are 1 and 900 respectively. And quarterly percentiles are 4, 12, and 28 bottles.

Alternative method

Calculate descriptive statistics of NUMBEERMO_EST

Use mean(), std(), min(), max(), median(), mode()

In [7]:

```
print('mean')
mean1 = sub2['NUMBEERMO_EST'].mean()
print (mean1)

print('std')
std1 = sub2['NUMBEERMO_EST'].std()
print (std1)

print('min')
min1 = sub2['NUMBEERMO_EST'].min()
print (min1)

print ('max')
max1 = sub2['NUMBEERMO_EST'].max()
print (max1)

print ('median')
median1 = sub2['NUMBEERMO_EST'].median()
print (median1)

print ('mode')
mode1 = sub2['NUMBEERMO_EST'].mode()
print (mode1)
```

```
mean
27.765712720799673
std
49.201312205771465
min
1.0
max
900.0
median
12.0
mode
0    8.000000
dtype: float64
```

An alternative method to calculate descriptive statistics. Two new statistics are median and mode of given dataset, and they are 12 and 8 respectively.

Calculate descriptive statistics for categorical data

S2BQ1B1 - Beer Dependence

Use describe()

In [13]:

```
print ('describe beer dependence')
desc2 = sub2['S2BQ1B1'].describe()
print (desc2)
```

```
describe beer dependence
count     10406.000000
unique        2.000000
top           0.000000
freq       9829.000000
Name: S2BQ1B1, dtype: float64
```

The descriptive statistics for beer dependency, there are 10406 interviewees, as there are only two values ‘0’ and ‘1’, so unique values is two. Top value is 0, and it has frequency of 9829.

What if categorical data was considered as quantitative data

S2BQ1B1 - Beer Dependence

Convert S2BQ1B1 to quantitative data and

Calculate descriptive statistics

Use describe()

In [14]:

```
sub2['S2BQ1B1'] = pd.to_numeric(sub2['S2BQ1B1']) # convert a numerical variable to quan
titatie
```

In [15]:

```
print ('describe beer dependence')
desc3 = sub2['S2BQ1B1'].describe()
print (desc3) #descriptor don't have sense
```

```
describe beer dependence
count    10406.000000
mean         0.055449
std          0.228865
min          0.000000
25%          0.000000
50%          0.000000
75%          0.000000
max          1.000000
Name: S2BQ1B1, dtype: float64
```

The descriptive statistics for beer dependency (quantitative) shows the mean value is 0.055, with standard deviation of 0.23. The minimum and maximum value are 0 and 1. All quarterly percentiles are 0.

Visualising 2 variable

Categorical -> Quantitative - Bar chart

Create a secondary variable

CARTONPERMONTH - number of beer carton consumed per month

assume that there is 24 beer cans in a carton

In [16]:

```
sub2['CARTONPERMONTH']=sub2['NUMBEERMO_EST'] / 24
```

In [17]:

```
c2= sub2.groupby('CARTONPERMONTH').size()
print (c2)
```

```
CARTONPERMONTH
0.041667     477
0.083333     407
0.104167     414
0.125000     172
0.166667     429
0.208333     623
0.250000      36
0.291667       5
0.312500     267
0.333333     635
0.416667     119
0.500000     296
0.520833      48
0.583333     160
0.625000      87
0.666667     561
0.729167       5
0.750000       1
0.833333      81
0.937500       3
1.000000     410
1.041667       6
1.083333      51
1.145833       1
1.166667     242
1.250000      62
1.333333     168
1.458333       1
1.500000       3
1.562500       2
            ...
4.083333       9
4.333333      37
4.666667      21
5.000000      39
5.416667      13
5.833333       5
6.000000       2
6.250000      18
6.500000      54
7.000000      27
7.500000      77
7.583333       6
8.000000       3
8.666667      10
8.750000       5
9.750000       2
10.000000     13
10.500000      5
10.833333      3
11.250000      4
12.500000      6
13.000000     14
15.000000     25
19.500000      1
21.250000      1
21.666667      1
22.500000      2
26.000000      1
30.000000      2
```

The data shows how many cartons of beer interviewees drink per month in the last 12 months. Grouping interviewees by their consumptions, from 0.042 cartons per month to 30 cartons per month.

```
37.500000      1
Length: 75, dtype: int64
```

Group CARTONPERMONTH into 7 groups

1 - 5 cartons

6 - 10 cartons

11 - 15 cartons

16 - 20 cartons

21 - 25 cartons

26 - 30 cartons

31 - max cartons

In [23]:

```
sub2['CARTONCATEGORY'] = pd.cut(sub2.CARTONPERMONTH, [0, 5, 10, 15, 20, 25, 30, 38])
```

In [24]:

```
# change format from numeric to categorical
sub2['CARTONCATEGORY'] = sub2['CARTONCATEGORY'].astype('category')
```

Print describe of CARTONCATEGORY

In [25]:

```
print('describe CARTONCATEGORY')
desc3 = sub2['CARTONCATEGORY'].describe()
print (desc3)
```

```
describe CARTONCATEGORY
count       7303
unique         7
top       (0, 5]
freq        7002
Name: CARTONCATEGORY, dtype: object
```

Discrete interviewees into 7 groups by their monthly beer consumption (in cartons). With 7303 data observed, most people (7002) drink 0 to 5 cartons per month.

Print carton category counts

In [26]:

```
print('carton category counts')
c7 = sub2['CARTONCATEGORY'].value_counts(sort=False, dropna=True)
print(c7)
```

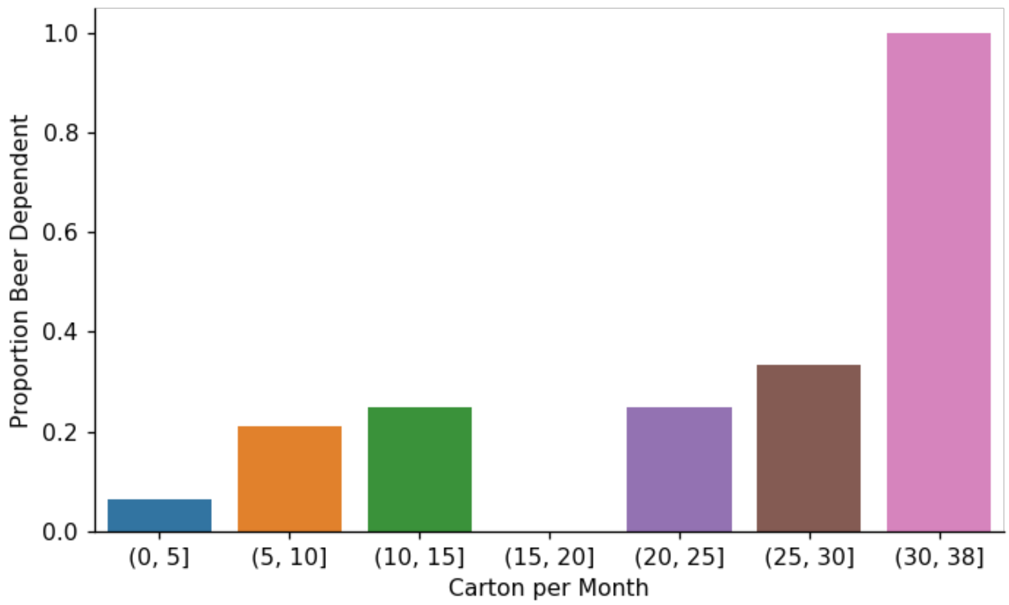
```
carton category counts
(0, 5]      7002
(5, 10]      235
(10, 15]      57
(15, 20]       1
(20, 25]       4
(25, 30]       3
(30, 38]       1
Name: CARTONCATEGORY, dtype: int64
```

A specific value counts for number of interviewees in each data group. Most people drink 0 to 5 cartons per month, and only single digit number of people drink more than 15 cartons per month.

Chart of bar chart showing the relationship between carton of beer consumed per month (CARTONCATEGORY) and Beer Dependent (S2BQ1B1)

In [27]:

```
# bivariate bar graph C->Q
sns.factorplot(x="CARTONCATEGORY", y="S2BQ1B1", data=sub2, kind="bar", ci=None)
plt.xlabel('Carton per Month')
plt.ylabel('Proportion Beer Dependent')
```



Out[27]:

```
Text(9.44444,0.5,'Proportion Beer Dependent')
```

A bar chart that shows there is a positive correlation between beer consumption and beer dependency. The more beer one drink, the more likely one becomes an alcoholic.

Visualising 2 variable

Categorical -> Categorical - Bar chart

Rename race from 1-5 to "White", "Black", "NatAm", "Asian", "Hispanic"

In [28]:

```
# 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')
# second create a new variable (PACKCAT) that has the new variable value labels
sub2['ETHRACE2A']=sub2['ETHRACE2A'].cat.rename_categories(["White", "Black", "NatAm",
"Asian", "Hispanic"])
```

Function to get 'CARTON_ADAY)

In [29]:

```
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)

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

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

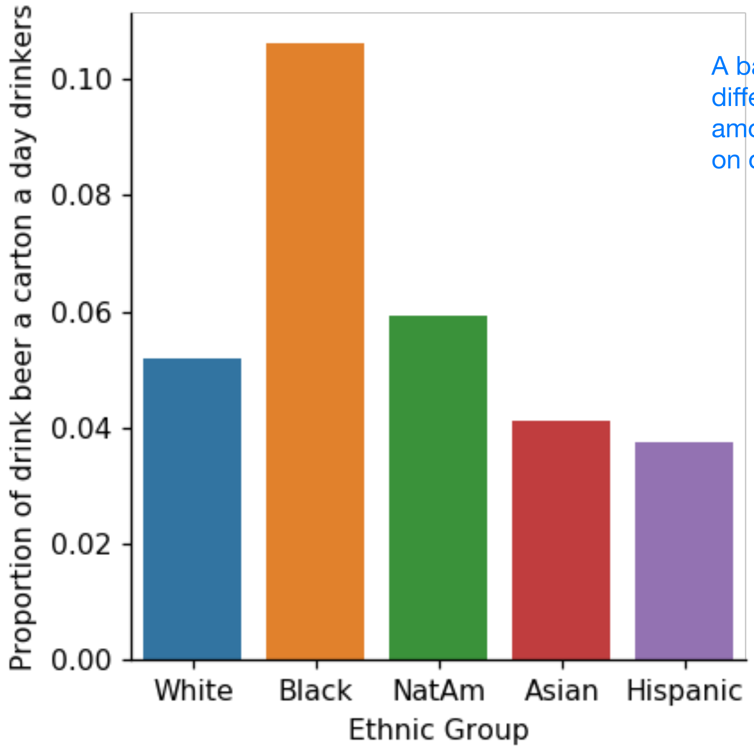
For over 7000 interviewees, 6897 of them do not drink more than a carton of beer per day, while the rest 417 interviewees do.

Bar Graph showing the relationship between race (ETHRACE2A) and

CARTON_ADAY

In [30]:

```
# bivariate bar graph C->C
sns.factorplot(x='ETHRACE2A', y='CARTON_ADAY', data=sub2, kind="bar", ci=None)
plt.xlabel('Ethnic Group')
plt.ylabel('Proportion of drink beer a carton a day drinkers')
```



A bar chart gives information for interviewees from five different ethnic groups, what percentage interviewees among those ethnic groups drink more a carton of beer on daily basis.

Out[30]:

```
Text(0.694444,0.5,'Proportion of drink beer a carton a day drinkers')
```

Visualising 2 variable

Categorical -> Quantitative - box plot

convert age to category data type

convert income (S1Q10A) to numeric data type

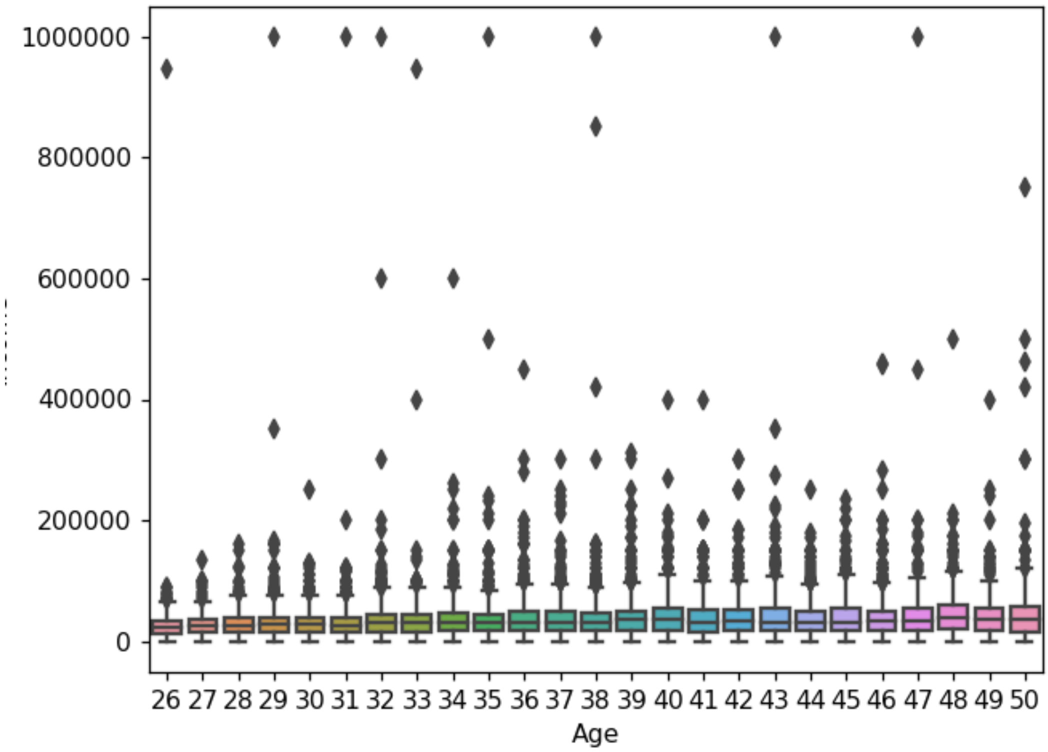
In [31]:

```
sub2['AGE'] = sub2['AGE'].astype('category')
sub2['S1Q10A'] = pd.to_numeric(sub2['S1Q10A'])
```

Box plot to show the relationship between age and income (S1Q10A) among adults aged 26 - 50 years old.

In [32]:

```
%matplotlib notebook
sns.boxplot(x='AGE', y='S1Q10A', data=sub2)
plt.xlabel('Age')
plt.ylabel('Income')
```



The box plot shows interviewees' income from age of 26 to age of 50. As interviewees grow older, their income gradually increase.

Out[32]:

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

Visualising 2 variable

Quantitative -> Quantitative - scatter plot

Read in gapminder.csv

In [33]:

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

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

Out[33]:

	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

The first 5 rows of csv file 'gapminder'

convert 'oilperperson' and 'relectricperperson' to numeric

In [34]:

```
gapminder['oilperperson'] = pd.to_numeric(gapminder['oilperperson'],errors='coerce')
gapminder['relectricperperson'] = pd.to_numeric(gapminder['relectricperperson'],errors=
'coerce')
```

drop NAN data

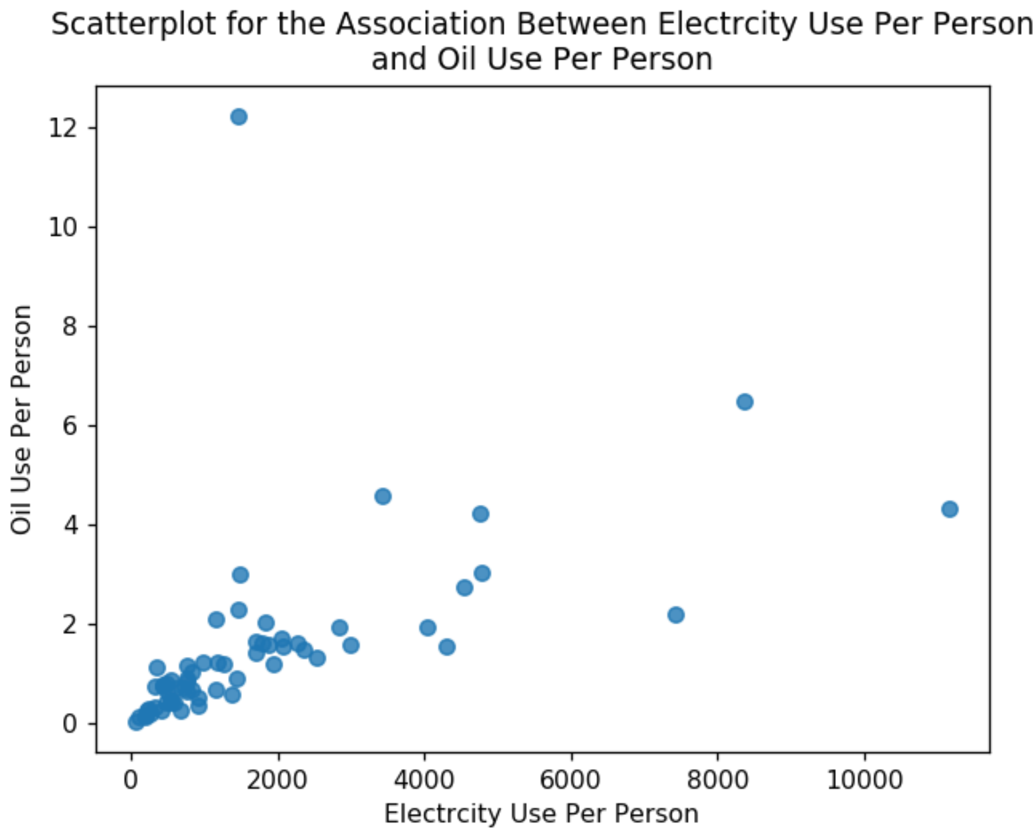
In [35]:

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

Scatter plot to show the relationship between Electrcity Use Per Person (relectricperperson) and Oil Use Per Person (oilperperson)

In [36]:

```
%matplotlib notebook
plt.figure()
scat1 = sns.regplot(x="relectricperperson", y="oilperperson", fit_reg=False, data=gapmi
nder_clean)
plt.xlabel('Electrcity Use Per Person')
plt.ylabel('Oil Use Per Person')
plt.title('Scatterplot for the Association Between Electrcity Use Per Person' + '\n' +
'and Oil Use Per Person')
```



Scatterplot indicates despite few outliers, there is a positive correlation between a person's oil usage and a person's electricity usage.

Out[36]:

```
Text(0.5,1,'Scatterplot for the Association Between Electrcity Use Per Per
son\nand Oil Use Per Person')
```