import pandas as pd  
import os  
import numpy as np  
import matplotlib.pyplot as plt   
%matplotlib inline  
from sklearn.cluster import KMeans  
from sklearn import datasets

pwd = os.getcwd() #capturing the present working directory

#Converting the data as it was recieved into   
read\_file = pd.read\_csv (pwd + r'\Data\adult.csv') # by using pwd + I am able to shorten the path required.  
df = pd.DataFrame(read\_file)  
df.head()

age workclass fnlwgt education education-num \  
0 39 State-gov 77516 Bachelors 13   
1 50 Self-emp-not-inc 83311 Bachelors 13   
2 38 Private 215646 HS-grad 9   
3 53 Private 234721 11th 7   
4 28 Private 338409 Bachelors 13   
  
 marital-status occupation relationship race sex \  
0 Never-married Adm-clerical Not-in-family White Male   
1 Married-civ-spouse Exec-managerial Husband White Male   
2 Divorced Handlers-cleaners Not-in-family White Male   
3 Married-civ-spouse Handlers-cleaners Husband Black Male   
4 Married-civ-spouse Prof-specialty Wife Black Female   
  
 capital-gain capital-loss hours-per-week country salary   
0 2174 0 40 United-States <=50K   
1 0 0 13 United-States <=50K   
2 0 0 40 United-States <=50K   
3 0 0 40 United-States <=50K   
4 0 0 40 Cuba <=50K

df.dtypes

age int64  
workclass object  
fnlwgt int64  
education object  
education-num int64  
marital-status object  
occupation object  
relationship object  
race object  
sex object  
capital-gain int64  
capital-loss int64  
hours-per-week int64  
country object  
salary object  
dtype: object

df\_mod = df.copy()  
df\_mod.head()

age workclass fnlwgt education education-num \  
0 39 State-gov 77516 Bachelors 13   
1 50 Self-emp-not-inc 83311 Bachelors 13   
2 38 Private 215646 HS-grad 9   
3 53 Private 234721 11th 7   
4 28 Private 338409 Bachelors 13   
  
 marital-status occupation relationship race sex \  
0 Never-married Adm-clerical Not-in-family White Male   
1 Married-civ-spouse Exec-managerial Husband White Male   
2 Divorced Handlers-cleaners Not-in-family White Male   
3 Married-civ-spouse Handlers-cleaners Husband Black Male   
4 Married-civ-spouse Prof-specialty Wife Black Female   
  
 capital-gain capital-loss hours-per-week country salary   
0 2174 0 40 United-States <=50K   
1 0 0 13 United-States <=50K   
2 0 0 40 United-States <=50K   
3 0 0 40 United-States <=50K   
4 0 0 40 Cuba <=50K

I'm first going to drop any columns I know I won't need at this point.

col\_to\_drop = ['workclass', 'fnlwgt', 'education-num', 'occupation', 'race', 'capital-gain', 'capital-loss', 'country']  
col\_to\_drop

['workclass',  
 'fnlwgt',  
 'education-num',  
 'occupation',  
 'race',  
 'capital-gain',  
 'capital-loss',  
 'country']

df\_mod = df\_mod.drop(columns=col\_to\_drop)  
df\_mod.head()

age education marital-status relationship sex hours-per-week \  
0 39 Bachelors Never-married Not-in-family Male 40   
1 50 Bachelors Married-civ-spouse Husband Male 13   
2 38 HS-grad Divorced Not-in-family Male 40   
3 53 11th Married-civ-spouse Husband Male 40   
4 28 Bachelors Married-civ-spouse Wife Female 40   
  
 salary   
0 <=50K   
1 <=50K   
2 <=50K   
3 <=50K   
4 <=50K

With this result I can see that I still need to sort the data into more of a binary approach if the clustering is going to work.

Now that I have made a copy of the edited dataset I can start to pick points of interest.

df\_mod\_copy = df\_mod.copy()  
col\_to\_drop=['education', 'marital-status', 'relationship', 'salary']  
df\_mod\_copy = df\_mod\_copy.drop(columns=col\_to\_drop)  
df\_mod\_copy.head()

age sex hours-per-week  
0 39 Male 40  
1 50 Male 13  
2 38 Male 40  
3 53 Male 40  
4 28 Female 40

As you can see I am now left with three columns or points of interest. age, sex and hours-per-week. I will be able to build the scatter graph showing how age relates to how many hours people work. But I also have the possibility of seperating the data into two scatter graphs which would also show not just how age affects, but also sex.

Now I am going to split this data into Male and Female and change the value of the cells to 1 or 0.

final\_result = df\_mod\_copy.copy()  
  
for x in final\_result.index:  
 if final\_result.loc[x, "sex"] == "Male":  
 final\_result.at[x, "sex"] = 1  
 else:  
 final\_result.at[x, "sex"] = 2  
 """In this case we do not need to check whether the other   
 value would be anything but what we expected (Male/ Female) because the 'sex'  
 column has no missing data or unexpected values. However, in other situations  
 I may have had to check for unknowns."""  
   
final\_result.head()

age sex hours-per-week  
0 39 1 40  
1 50 1 13  
2 38 1 40  
3 53 1 40  
4 28 2 40

df\_male = final\_result.copy()  
  
for x in df\_male.index:  
 if df\_male.loc[x, "sex"] == 2:  
 df\_male.drop(x, inplace = True)  
 """In this case we do not need to check whether the other   
 value would be anything but what we expected (Male/ Female) because the 'sex'  
 column has no missing data or unexpected values. However, in other situations  
 I may have had to check for unknowns."""  
   
df\_male.head()

age sex hours-per-week  
0 39 1 40  
1 50 1 13  
2 38 1 40  
3 53 1 40  
7 52 1 45

df\_female = final\_result.copy()  
  
for x in df\_female.index:  
 if df\_female.loc[x, "sex"] == 1:   
 df\_female.drop(x, inplace = True)  
 """In this case we do not need to check whether the other   
 value would be anything but what we expected (Male/ Female) because the 'sex'  
 column has no missing data or unexpected values. However, in other situations  
 I may have had to check for unknowns."""  
   
df\_female.head()

age sex hours-per-week  
4 28 2 40  
5 37 2 40  
6 49 2 16  
8 31 2 50  
12 23 2 30

writer = pd.ExcelWriter(pwd + "/Data/data\_output.xlsx", engine = 'xlsxwriter')  
df.to\_excel(writer, sheet\_name = 'adult\_data')  
df\_mod.to\_excel(writer, sheet\_name = 'data\_FirstEdit')  
df\_mod\_copy.to\_excel(writer, sheet\_name = 'data\_SecondEdit')  
df\_male.to\_excel(writer, sheet\_name = 'Final\_MaleData')  
df\_female.to\_excel(writer, sheet\_name = 'Final\_FemaleData')  
final\_result.to\_excel(writer, sheet\_name = 'Final\_Result')  
writer.save()