H. Diana McSpadden (hdm5s)

Lab Assignment 2: How to Load CSV, ASCII, and other data into Python

DS 6001: Practice and Application of Data Science

Instructions

Please answer the following questions as completely as possible using text, code, and the results of code as needed. Format your answers in a Jupyter notebook. To receive full credit, make sure you address every part of the problem, and make sure your document is formatted in a clean and professional way.

There are 11 data files attached to this lab assignment, with different extensions. First, download all of these data files, and save them in the same folder on your local machine. Your task in the following questions is to load each file into Python correctly, so that you can begin the process of data cleaning. If the variable names are included in the file, use those names to name the columns. If the variable names are not included, use these names in order:

In [ ]:
```
column_names = ["Country", "Happiness score", "Whisker-high", "Whisker-low",
  "Dystopia (1.92) + residual", "Explained by: GDP per capita",
  "Explained by: Social support", "Explained by: Healthy life expectancy",
  "Explained by: Freedom to make life choices", "Explained by: Generosity",
  "Explained by: Perceptions of corruption" ]
```

If you loaded the data correctly, it will look like `data_clean.csv`, which is also attached to this lab.

Problem 0

Import the libraries you will need. Then write code to change the working directory to the folder in which you saved the data files, run the code displayed above to create the `column_names` list, load `data_clean.csv`, and display the output of the `.info()` method of `data_clean`. (1 point)

In [ ]:
```
import os
import pandas as pd
import numpy as np
import xlrd # to read excel files
# import openpyxl # if you get an error, uncomment this line
```

In [ ]:
```
os.getcwd()
```

Out[ ]:
```
'c:\\Users\\dianam\\Documents\\jlab_datascience\\PlayGround\\UVa\\ds6001\\mod2'
```

In [ ]:
```
# load the data_clean.csv file and display the output of the .info() method
df_data_clean = pd.read_csv("data_clean.csv")
df_data_clean.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 156 entries, 0 to 155
Data columns (total 11 columns):
 #   Column                                    Non-Null Count  Dtype  
---  ------                                    --------------  -----  
 0   Country                                   156 non-null    object 
 1   Happiness score                           156 non-null    float64
 2   Whisker-high                              156 non-null    float64
 3   Whisker-low                               156 non-null    float64
 4   Dystopia (1.92) + residual                156 non-null    float64
 5   Explained by: GDP per capita              156 non-null    float64
 6   Explained by: Social support              156 non-null    float64
 7   Explained by: Healthy life expectancy     156 non-null    float64
 8   Explained by: Freedom to make life choices  156 non-null    float64
 9   Explained by: Generosity                  156 non-null    float64
 10  Explained by: Perceptions of corruption   156 non-null    float64
dtypes: float64(10), object(1)
memory usage: 13.5+ KB
```

In [ ]:
```
df_data_clean.describe()
```

Out[ ]:

	Happiness score	Whisker-high	Whisker-low	Dystopia (1.92) + residual	Explained by: GDP per capita	Explained by: Social support	Explained by: Healthy life expectancy	Expla Free to m cho
count	156.000000	156.000000	156.000000	156.000000	156.000000	156.000000	156.000000	156.00
mean	5.375878	5.478654	5.273090	1.922891	0.887417	1.216571	0.598026	0.45
std	1.119507	1.107522	1.132544	0.513760	0.382573	0.300380	0.247925	0.16
min	2.905000	3.074000	2.735000	0.292000	0.000000	0.000000	0.000000	0.00
25%	4.453750	4.589750	4.344750	1.654250	0.616250	1.076750	0.422250	0.35
50%	5.378000	5.477500	5.284500	1.908500	0.949500	1.261500	0.644000	0.49
75%	6.168500	6.260000	6.051250	2.270250	1.197750	1.463000	0.777250	0.58
max	7.632000	7.695000	7.569000	2.961000	1.649000	1.644000	1.030000	0.72

Problem 1

Load `data1.csv`. Use the tools we discussed in class to decide whether the data file loaded correctly, and include that code in your lab report. In one or two sentences, describe how you decided on the right combination of parameters needed to load the data. (1 point)

Answer:

I used the skiprows = 2 because I needed to ignore the metadata row at the top of the file. The column names were already included in the file and were the same as in the data_clean.csv file. The df_data1.info() is the same as the same as the df_clean.info().

In [ ]:
```
# load the data_clean.csv file and display the output of the .info() method
# need to ignore the first two rows which have meta data
df_data1 = pd.read_csv("data1.csv", skiprows=2)
df_data1.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 156 entries, 0 to 155
Data columns (total 11 columns):
 #   Column                                    Non-Null Count  Dtype  
---  ------                                    --------------  -----  
 0   Country                                   156 non-null    object 
 1   Happiness score                           156 non-null    float64
 2   Whisker-high                              156 non-null    float64
 3   Whisker-low                               156 non-null    float64
 4   Dystopia (1.92) + residual                156 non-null    float64
 5   Explained by: GDP per capita              156 non-null    float64
 6   Explained by: Social support              156 non-null    float64
 7   Explained by: Healthy life expectancy     156 non-null    float64
 8   Explained by: Freedom to make life choices  156 non-null    float64
 9   Explained by: Generosity                  156 non-null    float64
 10  Explained by: Perceptions of corruption   156 non-null    float64
dtypes: float64(10), object(1)
memory usage: 13.5+ KB
```

In [ ]:
```
np.round(df_data_clean.describe(),6) - np.round(df_data1.describe(),6)
```

Out[ ]:

	Happiness score	Whisker-high	Whisker-low	Dystopia (1.92) + residual	Explained by: GDP per capita	Explained by: Social support	Explained by: Healthy life expectancy	Explained by: Freedom to make life choices	Ex Ge
count	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
mean	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
std	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
min	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
25%	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
50%	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
75%	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
max	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	

Problem 2

Load `data2.txt`. Use the tools we discussed in class to decide whether the data file loaded correctly, and include that code in your lab report. In one or two sentences, describe how you decided on the right combination of parameters needed to load the data. (1 point)

Answer:

I hadn't used the pandas read_table() method before, but I found the documentation for pd.read_table() very helpful when I googled 'python load txt file remove lines starting with character':

https://pandas.pydata.org/docs/reference/api/pandas.read_table.html

I looked at the .txt file and found the rows to skip that had meta data in them and included those row indices in the skiprows list. I also skipped the row with the column names and re-added the column names using the names parameter, and set the separator character to the comma.

In [ ]:
```
# load the data2.txt, but I first need to remove lines that start with '/The '
df_data2 = pd.read_table("data2.txt", sep=',', skiprows=lambda x: x in [0,1,2,3,17,
df_data2.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 156 entries, 0 to 155
Data columns (total 11 columns):
 #   Column                                     Non-Null Count  Dtype
---  ------                                     --------------  -----
 0   Country                                    156 non-null    object
 1   Happiness score                            156 non-null    float64
 2   Whisker-high                               156 non-null    float64
 3   Whisker-low                                156 non-null    float64
 4   Dystopia (1.92) + residual                 156 non-null    float64
 5   Explained by: GDP per capita               156 non-null    float64
 6   Explained by: Social support               156 non-null    float64
 7   Explained by: Healthy life expectancy      156 non-null    float64
 8   Explained by: Freedom to make life choices  156 non-null    float64
 9   Explained by: Generosity                   156 non-null    float64
 10  Explained by: Perceptions of corruption    156 non-null    float64
dtypes: float64(10), object(1)
memory usage: 13.5+ KB
```

In [ ]:
```
np.round(df_data_clean.describe(),6) - np.round(df_data2.describe(),6)
```

Out[ ]:

	Happiness score	Whisker-high	Whisker-low	Dystopia (1.92) + residual	Explained by: GDP per capita	Explained by: Social support	Explained by: Healthy life expectancy	Explained by: Freedom to make life choices	Ex Ge
count	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
mean	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
std	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
min	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
25%	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
50%	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
75%	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
max	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	

Problem 3

Load `data3.txt`. Use the tools we discussed in class to decide whether the data file loaded correctly, and include that code in your lab report. In one or two sentences, describe how you decided on the right combination of parameters needed to load the data. (1 point)

Answer:

I could still use pd.read_csv, but I needed to use the sep="\t" for tab separated, and I needed to skip two rows of metadata. The column names were included in this file.

In [ ]:
```
# load the data3.txt file and display the output of the .info() method
df_data3 = pd.read_csv("data3.txt", sep="\t", skiprows=2)
df_data3.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 156 entries, 0 to 155
Data columns (total 11 columns):
 #   Column                                     Non-Null Count  Dtype  
---  ------                                     --------------  -----  
 0   Country                                    156 non-null    object 
 1   Happiness score                            156 non-null    float64
 2   Whisker-high                               156 non-null    float64
 3   Whisker-low                                156 non-null    float64
 4   Dystopia (1.92) + residual                 156 non-null    float64
 5   Explained by: GDP per capita               156 non-null    float64
 6   Explained by: Social support               156 non-null    float64
 7   Explained by: Healthy life expectancy      156 non-null    float64
 8   Explained by: Freedom to make life choices 156 non-null    float64
 9   Explained by: Generosity                   156 non-null    float64
 10  Explained by: Perceptions of corruption    156 non-null    float64
dtypes: float64(10), object(1)
memory usage: 13.5+ KB
```

In [ ]:
```
np.round(df_data_clean.describe(),6) - np.round(df_data3.describe(),6)
```

Out[ ]:

	Happiness score	Whisker-high	Whisker-low	Dystopia (1.92) + residual	Explained by: GDP per capita	Explained by: Social support	Explained by: Healthy life expectancy	Explained by: Freedom to make life choices	Ex Ge
count	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
mean	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
std	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
min	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
25%	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
50%	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
75%	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
max	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	

Problem 4

Load `data4.txt`. Use the tools we discussed in class to decide whether the data file loaded correctly, and include that code in your lab report. In one or two sentences, describe how you decided on the right combination of parameters needed to load the data. (1 point)

Answer:

I used read_csv(), and I set the seperator to '$' and header to None (to get all 156 entries). I set the column names to the list column_names.

In [ ]:
```
# load the data3.txt file and display the output of the .info() method
df_data4 = pd.read_csv("data4.txt", sep="$", header=None)
df_data4.columns = column_names
df_data4.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 156 entries, 0 to 155
Data columns (total 11 columns):
 #   Column                                     Non-Null Count  Dtype
---  ------                                     --------------  -----
 0   Country                                    156 non-null    object
 1   Happiness score                            156 non-null    float64
 2   Whisker-high                               156 non-null    float64
 3   Whisker-low                                156 non-null    float64
 4   Dystopia (1.92) + residual                 156 non-null    float64
 5   Explained by: GDP per capita               156 non-null    float64
 6   Explained by: Social support               156 non-null    float64
 7   Explained by: Healthy life expectancy      156 non-null    float64
 8   Explained by: Freedom to make life choices  156 non-null    float64
 9   Explained by: Generosity                   156 non-null    float64
 10  Explained by: Perceptions of corruption    156 non-null    float64
dtypes: float64(10), object(1)
memory usage: 13.5+ KB
```

In [ ]:
```
np.round(df_data_clean.describe(),6) - np.round(df_data4.describe(),6)
```

Out[ ]:

	Happiness score	Whisker-high	Whisker-low	Dystopia (1.92) + residual	Explained by: GDP per capita	Explained by: Social support	Explained by: Healthy life expectancy	Explained by: Freedom to make life choices	Ex Ge
count	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
mean	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
std	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
min	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
25%	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
50%	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
75%	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
max	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	

Problem 5

Load `data5.csv`. Use the tools we discussed in class to decide whether the data file loaded correctly, and include that code in your lab report. In one or two sentences, describe how you decided on the right combination of parameters needed to load the data. (1 point)

Answer:

The read_csv() method works perfectly here, the default separator is a comma, so I don't even need to add that; however, there were two extra footer rows with metadata about the dataset, so I used to the skipfooter argument to skip those two rows.

In [ ]:
```
df_data5 = pd.read_csv("data5.csv", skipfooter=2)
df_data5.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 156 entries, 0 to 155
Data columns (total 11 columns):
 #   Column                                    Non-Null Count  Dtype  
---  ------                                    --------------  -----  
 0   Country                                   156 non-null    object 
 1   Happiness score                           156 non-null    float64
 2   Whisker-high                              156 non-null    float64
 3   Whisker-low                               156 non-null    float64
 4   Dystopia (1.92) + residual                156 non-null    float64
 5   Explained by: GDP per capita              156 non-null    float64
 6   Explained by: Social support              156 non-null    float64
 7   Explained by: Healthy life expectancy     156 non-null    float64
 8   Explained by: Freedom to make life choices  156 non-null    float64
 9   Explained by: Generosity                  156 non-null    float64
 10  Explained by: Perceptions of corruption   156 non-null    float64
dtypes: float64(10), object(1)
memory usage: 13.5+ KB
```

```
C:\Users\dianam\AppData\Local\Temp\ipykernel_18744\1010595436.py:1: ParserWarning: 
Falling back to the 'python' engine because the 'c' engine does not support skipfo
oter; you can avoid this warning by specifying engine='python'.
  df_data5 = pd.read_csv("data5.csv", skipfooter=2)
```

In [ ]:
```
np.round(df_data_clean.describe(),6) - np.round(df_data5.describe(),6)
```

Out[ ]:

	Happiness score	Whisker-high	Whisker-low	Dystopia (1.92) + residual	Explained by: GDP per capita	Explained by: Social support	Explained by: Healthy life expectancy	Explained by: Freedom to make life choices	Ex Ge
count	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
mean	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
std	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
min	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
25%	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
50%	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
75%	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
max	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	

Problem 6

Load `data6.dat`. Use the tools we discussed in class to decide whether the data file loaded correctly, and include that code in your lab report. In one or two sentences, describe how you decided on the right combination of parameters needed to load the data. (1 point)

Answer:

The read_table() with a comma separator works for this file. I did set an NaN value of 999 since there were so many 999 values in this data file, which is very different from the other data files. A google search indicated that 999 is commonly used in .dat files for missing values. I decided to set the mean value for each column to any NaN's. This is the only file in the lab assignment with this issue and, unlike the other data files, the `df_data_clean.describe() - df_data6.describe()` does not return all zeros because of these missing values.

In [ ]:
```
df_data6 = pd.read_table("data6.dat", sep=",", na_values="999")
# set NaN values to the mean for the column
for col in df_data6.columns:
    # if the column is numeric, then set NaN values to the mean
    if df_data6[col].dtype == np.float64:
        df_data6[col] = df_data6[col].fillna(df_data6[col].mean())

df_data6.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 156 entries, 0 to 155
Data columns (total 11 columns):
 #   Column                                     Non-Null Count  Dtype
---  ------                                     --------------  -----
 0   Country                                    145 non-null    object
 1   Happiness score                            156 non-null    float64
 2   Whisker-high                               156 non-null    float64
 3   Whisker-low                                156 non-null    float64
 4   Dystopia (1.92) + residual                 156 non-null    float64
 5   Explained by: GDP per capita               156 non-null    float64
 6   Explained by: Social support               156 non-null    float64
 7   Explained by: Healthy life expectancy      156 non-null    float64
 8   Explained by: Freedom to make life choices 156 non-null    float64
 9   Explained by: Generosity                   156 non-null    float64
 10  Explained by: Perceptions of corruption    156 non-null    float64
dtypes: float64(10), object(1)
memory usage: 13.5+ KB
```

In [ ]:
```
df_data_clean.head()
```

Out[ ]:

	Country	Happiness score	Whisker-high	Whisker-low	Dystopia (1.92) + residual	Explained by: GDP per capita	Explained by: Social support	Explained by: Healthy life expectancy	Explai Freed to m choi
0	Finland	7.632	7.695	7.569	2.595	1.305	1.592	0.874	0.
1	Norway	7.594	7.657	7.530	2.383	1.456	1.582	0.861	0.
2	Denmark	7.555	7.623	7.487	2.370	1.351	1.590	0.868	0.
3	Iceland	7.495	7.593	7.398	2.426	1.343	1.644	0.914	0.
4	Switzerland	7.487	7.570	7.405	2.320	1.420	1.549	0.927	0.

In [ ]:
```
df_data6.head()
```

Out[ ]:

	Country	Happiness score	Whisker-high	Whisker-low	Dystopia (1.92) + residual	Explained by: GDP per capita	Explained by: Social support	Explained by: Healthy life expectancy	Explai Freed to m choi
0	Finland	7.632	7.695	7.569000	2.595000	0.900577	1.226739	0.590803	0.
1	Norway	7.594	7.657	7.530000	1.922241	0.900577	1.582000	0.590803	0.
2	Denmark	7.555	7.623	7.487000	2.370000	1.351000	1.590000	0.590803	0.
3	Iceland	7.495	7.593	5.345412	2.426000	1.343000	1.644000	0.914000	0.
4	Switzerland	7.487	7.570	7.405000	2.320000	1.420000	1.549000	0.927000	0.

In [ ]:
```
np.round(df_data_clean.describe(),6) - np.round(df_data6.describe(),6)
```

Out[ ]:

	Happiness score	Whisker-high	Whisker-low	Dystopia (1.92) + residual	Explained by: GDP per capita	Explained by: Social support	Explained by: Healthy life expectancy	Explained by: Freedom to make life choices	G
count	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	
mean	-0.083418	-0.022872	-0.072322	0.000650	-0.013160	-0.010168	0.007223	0.001531	
std	0.051106	0.080321	0.074211	0.017528	0.031349	0.023641	0.009149	0.007300	
min	0.000000	0.000000	-0.204000	0.000000	0.000000	0.000000	0.000000	0.000000	
25%	-0.136750	-0.121000	-0.239000	-0.018750	-0.072750	-0.072750	-0.016750	-0.013750	
50%	-0.081296	-0.024026	-0.060912	-0.013741	0.048923	0.034131	0.032500	0.035000	
75%	0.000000	0.084000	0.019750	0.022000	0.046000	0.011250	0.032250	0.005000	
max	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	

Problem 7

Load `data7.xlsx`, which is an Excel file. Keep only the sheet named “Data”. Use the tools we discussed in class to decide whether the data file loaded correctly, and include that code in your lab report. In one or two sentences, describe how you decided on the right combination of parameters needed to load the data. (2 points)

Answer:

The read_excel() document was helpful: https://pandas.pydata.org/docs/reference/api/pandas.read_excel.html.

I needed to `conda install -c anaconda openpyxl` to my kernel in order to be able to read the excel file. The reading recommended a different library, the xlrd library, which I installed with `conda install -c anaconda xlrd` and demonstrate.

In [ ]:
```
df_data7 = pd.read_excel("data7.xlsx", 'Data')
df_data7.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 156 entries, 0 to 155
Data columns (total 11 columns):
 #   Column                                     Non-Null Count  Dtype
---  ------                                     --------------  -----
 0   Country                                    156 non-null    object
 1   Happiness score                            156 non-null    float64
 2   Whisker-high                               156 non-null    float64
 3   Whisker-low                                156 non-null    float64
 4   Dystopia (1.92) + residual                 156 non-null    float64
 5   Explained by: GDP per capita               156 non-null    float64
 6   Explained by: Social support               156 non-null    float64
 7   Explained by: Healthy life expectancy      156 non-null    float64
 8   Explained by: Freedom to make life choices  156 non-null    float64
 9   Explained by: Generosity                   156 non-null    float64
 10  Explained by: Perceptions of corruption    156 non-null    float64
dtypes: float64(10), object(1)
memory usage: 13.5+ KB
```

In [ ]:
```
np.round(df_data_clean.describe(),6) - np.round(df_data7.describe(),6)
```

Out[ ]:

	Happiness score	Whisker-high	Whisker-low	Dystopia (1.92) + residual	Explained by: GDP per capita	Explained by: Social support	Explained by: Healthy life expectancy	Explained by: Freedom to make life choices	Ex Ge
count	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
mean	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
std	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
min	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
25%	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
50%	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
75%	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
max	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	

Problem 8

Load `data8.dta`, which is a Stata 13 file. Use the tools we discussed in class to decide whether the data file loaded correctly, and include that code in your lab report. In one or two sentences, describe how you decided on the right combination of parameters needed to load the data. (2 points)

Answer:

The read_stata document was helpful: https://pandas.pydata.org/docs/reference/api/pandas.read_stata.html. This one took a few steps for me. I read the file, set the floats from float32's to float64 to match the clean data, set the column names using the list, and reset the index to get the RangeIndex. (I used StackOverflow to find a way to do this.)

In [ ]:
```
df_data8 = pd.read_stata("data8.dta")
# set the float format to float64 for the float32 columns
df_data8[df_data8.select_dtypes(np.float32).columns] = df_data8.select_dtypes(np.fl
df_data8.columns = column_names
# need to reset the index to get a RangeIndex (otherwise it will be an IntIndex whi
# needed to use drop=True otherwise I get another column with the old index
df_data8.reset_index(inplace=True, drop=True)
df_data8.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 156 entries, 0 to 155
Data columns (total 11 columns):
 #   Column                                     Non-Null Count  Dtype  
---  ------                                     --------------  -----  
 0   Country                                    156 non-null    object 
 1   Happiness score                            156 non-null    float64
 2   Whisker-high                               156 non-null    float64
 3   Whisker-low                                156 non-null    float64
 4   Dystopia (1.92) + residual                 156 non-null    float64
 5   Explained by: GDP per capita               156 non-null    float64
 6   Explained by: Social support               156 non-null    float64
 7   Explained by: Healthy life expectancy      156 non-null    float64
 8   Explained by: Freedom to make life choices 156 non-null    float64
 9   Explained by: Generosity                   156 non-null    float64
 10  Explained by: Perceptions of corruption    156 non-null    float64
dtypes: float64(10), object(1)
memory usage: 13.5+ KB
```

In [ ]:
```
np.round(df_data_clean.describe(),6) - np.round(df_data8.describe(),6)
```

Out[ ]:

	Happiness score	Whisker-high	Whisker-low	Dystopia (1.92) + residual	Explained by: GDP per capita	Explained by: Social support	Explained by: Healthy life expectancy	Explained by: Freedom to make life choices	Ex Ge
count	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
mean	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
std	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
min	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
25%	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
50%	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
75%	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
max	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	

Problem 9

Load `data9.sav`, which is an SPSS file. Use the tools we discussed in class to decide whether the data file loaded correctly, and include that code in your lab report. In one or two sentences, describe how you decided on the right combination of parameters needed to load the data. (2 points)

Answer:

I needed to run `conda install -c conda-forge pyreadstat` in order to run pd.read_spss(). I also set the column headers using the column_names list.

In [ ]:
```
df_data9 = pd.read_spss("data9.sav")
df_data9.columns = column_names
df_data9.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 156 entries, 0 to 155
Data columns (total 11 columns):
 #   Column                                     Non-Null Count  Dtype
---  ------                                     --------------  -----
 0   Country                                    156 non-null    object
 1   Happiness score                            156 non-null    float64
 2   Whisker-high                               156 non-null    float64
 3   Whisker-low                                156 non-null    float64
 4   Dystopia (1.92) + residual                 156 non-null    float64
 5   Explained by: GDP per capita               156 non-null    float64
 6   Explained by: Social support               156 non-null    float64
 7   Explained by: Healthy life expectancy      156 non-null    float64
 8   Explained by: Freedom to make life choices 156 non-null    float64
 9   Explained by: Generosity                   156 non-null    float64
 10  Explained by: Perceptions of corruption    156 non-null    float64
dtypes: float64(10), object(1)
memory usage: 13.5+ KB
```

In [ ]:
```
np.round(df_data_clean.describe(),6) - np.round(df_data9.describe(),6)
```

Out[ ]:

	Happiness score	Whisker-high	Whisker-low	Dystopia (1.92) + residual	Explained by: GDP per capita	Explained by: Social support	Explained by: Healthy life expectancy	Explained by: Freedom to make life choices	Ex Ge
count	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
mean	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
std	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
min	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
25%	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
50%	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
75%	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
max	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	

Problem 10

Load `data10.xpt`, which is a SAS file. Use the tools we discussed in class to decide whether the data file loaded correctly, and include that code in your lab report. In one or two sentences, describe how you decided on the right combination of parameters needed to load the data. (If some of the country names display as `b'Finland'`, don't worry aout that.) (2 points)

In [ ]:
```
df_data10 = pd.read_sas("data10.xpt")
df_data10.columns = column_names
```

```
df_data10.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 156 entries, 0 to 155
Data columns (total 11 columns):
 #   Column                                     Non-Null Count  Dtype
---  ------                                     --------------  -----
 0   Country                                    156 non-null    object
 1   Happiness score                            156 non-null    float64
 2   Whisker-high                               156 non-null    float64
 3   Whisker-low                                156 non-null    float64
 4   Dystopia (1.92) + residual                 156 non-null    float64
 5   Explained by: GDP per capita               156 non-null    float64
 6   Explained by: Social support               156 non-null    float64
 7   Explained by: Healthy life expectancy      156 non-null    float64
 8   Explained by: Freedom to make life choices 156 non-null    float64
 9   Explained by: Generosity                   156 non-null    float64
 10  Explained by: Perceptions of corruption    156 non-null    float64
dtypes: float64(10), object(1)
memory usage: 13.5+ KB
```

In [ ]:
```
np.round(df_data_clean.describe(),6) - np.round(df_data10.describe(),6)
```

Out[ ]:

	Happiness score	Whisker-high	Whisker-low	Dystopia (1.92) + residual	Explained by: GDP per capita	Explained by: Social support	Explained by: Healthy life expectancy	Explained by: Freedom to make life choices	Ex Ge
count	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
mean	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
std	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
min	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
25%	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
50%	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
75%	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
max	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	

Problem 11

Please load the `data11.txt` file, which is a fixed width file. The columns are defined as follows:

Variable	Width	Start	End
Country	24	1	24
Happiness score	5	25	29

Variable	Width	Start	End
Whisker-high	5	30	34
Whisker-low	5	35	39
Dystopia (1.92) + residual	5	40	44
Explained by: GDP per capita	5	45	49
Explained by: Social support	5	50	54
Explained by: Healthy life expectancy	5	55	59
Explained by: Freedom to make life choices	5	60	64
Explained by: Generosity	5	65	69
Explained by: Perceptions of corruption	5	70	74

Then save the this loaded data frame as a CSV file on your local machine. Be sure to use a unique filename so as not to overwrite any existing files. (5 points)

Answer:

I used read_fwf() the pandas method to read fixed width files. I needed to use the header=None argument because otherwise I only retrieved 155 rows and the first row was treated as the header row. I also needed to set the column names using the list column_names.

In [ ]:
```
# load the fixed width data txt file data11.txt
# creating a "the_widths" list of a 24 then 10 5's which are the fixed widths of th
the_widths = [24] + [5] * 10
df_data11 = pd.read_fwf("data11.txt", widths=the_widths, header=None)
df_data11.columns = column_names
df_data11.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 156 entries, 0 to 155
Data columns (total 11 columns):
 #   Column                                    Non-Null Count  Dtype  
---  ------                                    --------------  -----  
 0   Country                                   156 non-null    object 
 1   Happiness score                           156 non-null    float64
 2   Whisker-high                              156 non-null    float64
 3   Whisker-low                               156 non-null    float64
 4   Dystopia (1.92) + residual                156 non-null    float64
 5   Explained by: GDP per capita              156 non-null    float64
 6   Explained by: Social support              156 non-null    float64
 7   Explained by: Healthy life expectancy     156 non-null    float64
 8   Explained by: Freedom to make life choices  156 non-null    float64
 9   Explained by: Generosity                  156 non-null    float64
 10  Explained by: Perceptions of corruption   156 non-null    float64
dtypes: float64(10), object(1)
memory usage: 13.5+ KB
```

In [ ]:
```
np.round(df_data_clean.describe(),6) - np.round(df_data11.describe(),6)
```

Out[ ]:

	Happiness score	Whisker-high	Whisker-low	Dystopia (1.92) + residual	Explained by: GDP per capita	Explained by: Social support	Explained by: Healthy life expectancy	Explained by: Freedom to make life choices	Ex Ge
count	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
mean	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
std	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
min	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
25%	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
50%	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
75%	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	
max	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	

In [ ]:
```
# save the data_clean.csv file
# using the index=False argument to set index to False to not add an additional col
df_data11.to_csv("data11_ascsv.csv", index=False)
```