Assignment 2

May 10, 2020

---

*You are currently looking at **version 1.2** of this notebook. To download notebooks and datafiles, as well as get help on Jupyter notebooks in the Coursera platform, visit the Jupyter Notebook FAQ course resource.*

---

1 Assignment 2 - Pandas Introduction

All questions are weighted the same in this assignment. ## Part 1 The following code loads the olympics dataset (olympics.csv), which was derrived from the Wikipedia entry on All Time Olympic Games Medals, and does some basic data cleaning.

The columns are organized as # of Summer games, Summer medals, # of Winter games, Winter medals, total # number of games, total # of medals. Use this dataset to answer the questions below.

In [3]:
```
import pandas as pd

df = pd.read_csv('olympics.csv', index_col=0, skiprows=1)

for col in df.columns:
    if col[:2]=='01':
        df.rename(columns={col:'Gold'+col[4:]}, inplace=True)
    if col[:2]=='02':
        df.rename(columns={col:'Silver'+col[4:]}, inplace=True)
    if col[:2]=='03':
        df.rename(columns={col:'Bronze'+col[4:]}, inplace=True)
    if col[:1]=='':
        df.rename(columns={col:'#'+col[1:]}, inplace=True)

names_ids = df.index.str.split('\s\(') # split the index by '('

df.index = names_ids.str[0] # the [0] element is the country name (new index)
df['ID'] = names_ids.str[1].str[:3] # the [1] element is the abbreviation or ID (take fi

df = df.drop('Totals')
df
```

Out[3]:

	# Summer	Gold	Silver	Bronze	Total	\
Afghanistan	13	0	0	2	2	
Algeria	12	5	2	8	15	
Argentina	23	18	24	28	70	
Armenia	5	1	2	9	12	
Australasia	2	3	4	5	12	
Australia	25	139	152	177	468	
Austria	26	18	33	35	86	
Azerbaijan	5	6	5	15	26	
Bahamas	15	5	2	5	12	
Bahrain	8	0	0	1	1	
Barbados	11	0	0	1	1	
Belarus	5	12	24	39	75	
Belgium	25	37	52	53	142	
Bermuda	17	0	0	1	1	
Bohemia	3	0	1	3	4	
Botswana	9	0	1	0	1	
Brazil	21	23	30	55	108	
British West Indies	1	0	0	2	2	
Bulgaria	19	51	85	78	214	
Burundi	5	1	0	0	1	
Cameroon	13	3	1	1	5	
Canada	25	59	99	121	279	
Chile	22	2	7	4	13	
China	9	201	146	126	473	
Colombia	18	2	6	11	19	
Costa Rica	14	1	1	2	4	
Ivory Coast	12	0	1	0	1	
Croatia	6	6	7	10	23	
Cuba	19	72	67	70	209	
Cyprus	9	0	1	0	1	
...	...	...	...	...	...	
Spain	22	37	59	35	131	
Sri Lanka	16	0	2	0	2	
Sudan	11	0	1	0	1	
Suriname	11	1	0	1	2	
Sweden	26	143	164	176	483	
Switzerland	27	47	73	65	185	
Syria	12	1	1	1	3	
Chinese Taipei	13	2	7	12	21	
Tajikistan	5	0	1	2	3	
Tanzania	12	0	2	0	2	
Thailand	15	7	6	11	24	
Togo	9	0	0	1	1	
Tonga	8	0	1	0	1	
Trinidad and Tobago	16	2	5	11	18	
Tunisia	13	3	3	4	10	
Turkey	21	39	25	24	88	

Uganda	14	2	3	2	7
Ukraine	5	33	27	55	115
United Arab Emirates	8	1	0	0	1
United States	26	976	757	666	2399
Uruguay	20	2	2	6	10
Uzbekistan	5	5	5	10	20
Venezuela	17	2	2	8	12
Vietnam	14	0	2	0	2
Virgin Islands	11	0	1	0	1
Yugoslavia	16	26	29	28	83
Independent Olympic Participants	1	0	1	2	3
Zambia	12	0	1	1	2
Zimbabwe	12	3	4	1	8
Mixed team	3	8	5	4	17

	# Winter	Gold.1	Silver.1	Bronze.1	\
Afghanistan	0	0	0	0	
Algeria	3	0	0	0	
Argentina	18	0	0	0	
Armenia	6	0	0	0	
Australasia	0	0	0	0	
Australia	18	5	3	4	
Austria	22	59	78	81	
Azerbaijan	5	0	0	0	
Bahamas	0	0	0	0	
Bahrain	0	0	0	0	
Barbados	0	0	0	0	
Belarus	6	6	4	5	
Belgium	20	1	1	3	
Bermuda	7	0	0	0	
Bohemia	0	0	0	0	
Botswana	0	0	0	0	
Brazil	7	0	0	0	
British West Indies	0	0	0	0	
Bulgaria	19	1	2	3	
Burundi	0	0	0	0	
Cameroon	1	0	0	0	
Canada	22	62	56	52	
Chile	16	0	0	0	
China	10	12	22	19	
Colombia	1	0	0	0	
Costa Rica	6	0	0	0	
Ivory Coast	0	0	0	0	
Croatia	7	4	6	1	
Cuba	0	0	0	0	
Cyprus	10	0	0	0	
...	...	...	...	...	
Spain	19	1	0	1	

Sri Lanka	0	0	0	0
Sudan	0	0	0	0
Suriname	0	0	0	0
Sweden	22	50	40	54
Switzerland	22	50	40	48
Syria	0	0	0	0
Chinese Taipei	11	0	0	0
Tajikistan	4	0	0	0
Tanzania	0	0	0	0
Thailand	3	0	0	0
Togo	1	0	0	0
Tonga	1	0	0	0
Trinidad and Tobago	3	0	0	0
Tunisia	0	0	0	0
Turkey	16	0	0	0
Uganda	0	0	0	0
Ukraine	6	2	1	4
United Arab Emirates	0	0	0	0
United States	22	96	102	84
Uruguay	1	0	0	0
Uzbekistan	6	1	0	0
Venezuela	4	0	0	0
Vietnam	0	0	0	0
Virgin Islands	7	0	0	0
Yugoslavia	14	0	3	1
Independent Olympic Participants	0	0	0	0
Zambia	0	0	0	0
Zimbabwe	1	0	0	0
Mixed team	0	0	0	0

	Total.1	# Games	Gold.2	Silver.2	\
Afghanistan	0	13	0	0	
Algeria	0	15	5	2	
Argentina	0	41	18	24	
Armenia	0	11	1	2	
Australasia	0	2	3	4	
Australia	12	43	144	155	
Austria	218	48	77	111	
Azerbaijan	0	10	6	5	
Bahamas	0	15	5	2	
Bahrain	0	8	0	0	
Barbados	0	11	0	0	
Belarus	15	11	18	28	
Belgium	5	45	38	53	
Bermuda	0	24	0	0	
Bohemia	0	3	0	1	
Botswana	0	9	0	1	
Brazil	0	28	23	30	

British West Indies	0	1	0	0
Bulgaria	6	38	52	87
Burundi	0	5	1	0
Cameroon	0	14	3	1
Canada	170	47	121	155
Chile	0	38	2	7
China	53	19	213	168
Colombia	0	19	2	6
Costa Rica	0	20	1	1
Ivory Coast	0	12	0	1
Croatia	11	13	10	13
Cuba	0	19	72	67
Cyprus	0	19	0	1
...	...	...	...	...
Spain	2	41	38	59
Sri Lanka	0	16	0	2
Sudan	0	11	0	1
Suriname	0	11	1	0
Sweden	144	48	193	204
Switzerland	138	49	97	113
Syria	0	12	1	1
Chinese Taipei	0	24	2	7
Tajikistan	0	9	0	1
Tanzania	0	12	0	2
Thailand	0	18	7	6
Togo	0	10	0	0
Tonga	0	9	0	1
Trinidad and Tobago	0	19	2	5
Tunisia	0	13	3	3
Turkey	0	37	39	25
Uganda	0	14	2	3
Ukraine	7	11	35	28
United Arab Emirates	0	8	1	0
United States	282	48	1072	859
Uruguay	0	21	2	2
Uzbekistan	1	11	6	5
Venezuela	0	21	2	2
Vietnam	0	14	0	2
Virgin Islands	0	18	0	1
Yugoslavia	4	30	26	32
Independent Olympic Participants	0	1	0	1
Zambia	0	12	0	1
Zimbabwe	0	13	3	4
Mixed team	0	3	8	5

	Bronze.2	Combined total	ID
Afghanistan	2	2	AFG
Algeria	8	15	ALG

Argentina	28	70	ARG
Armenia	9	12	ARM
Australasia	5	12	ANZ
Australia	181	480	AUS
Austria	116	304	AUT
Azerbaijan	15	26	AZE
Bahamas	5	12	BAH
Bahrain	1	1	BRN
Barbados	1	1	BAR
Belarus	44	90	BLR
Belgium	56	147	BEL
Bermuda	1	1	BER
Bohemia	3	4	BOH
Botswana	0	1	BOT
Brazil	55	108	BRA
British West Indies	2	2	BWI
Bulgaria	81	220	BUL
Burundi	0	1	BDI
Cameroon	1	5	CMR
Canada	173	449	CAN
Chile	4	13	CHI
China	145	526	CHN
Colombia	11	19	COL
Costa Rica	2	4	CRC
Ivory Coast	0	1	CIV
Croatia	11	34	CRO
Cuba	70	209	CUB
Cyprus	0	1	CYP
...	...	...	...
Spain	36	133	ESP
Sri Lanka	0	2	SRI
Sudan	0	1	SUD
Suriname	1	2	SUR
Sweden	230	627	SWE
Switzerland	113	323	SUI
Syria	1	3	SYR
Chinese Taipei	12	21	TPE
Tajikistan	2	3	TJK
Tanzania	0	2	TAN
Thailand	11	24	THA
Togo	1	1	TOG
Tonga	0	1	TGA
Trinidad and Tobago	11	18	TRI
Tunisia	4	10	TUN
Turkey	24	88	TUR
Uganda	2	7	UGA
Ukraine	59	122	UKR
United Arab Emirates	0	1	UAE

```
United States                          750            2681  USA
Uruguay                                  6              10  URU
Uzbekistan                              10              21  UZB
Venezuela                                8              12  VEN
Vietnam                                  0               2  VIE
Virgin Islands                           0               1  ISV
Yugoslavia                              29              87  YUG
Independent Olympic Participants         2               3  IOP
Zambia                                   1               2  ZAM
Zimbabwe                                 1               8  ZIM
Mixed team                               4              17  ZZX

[146 rows x 16 columns]
```

1.0.1 Question 0 (Example)

What is the first country in df?

This function should return a Series.

```
In [3]: # You should write your whole answer within the function provided. The autograder will c
        # this function and compare the return value against the correct solution value
        def answer_zero():
            # This function returns the row for Afghanistan, which is a Series object. The assig
            # question description will tell you the general format the autograder is expecting
            return df.iloc[0]

        # You can examine what your function returns by calling it in the cell. If you have ques
        # about the assignment formats, check out the discussion forums for any FAQs
        answer_zero()
```

```
Out[3]: # Summer           13
        Gold                0
        Silver              0
        Bronze              2
        Total               2
        # Winter           0
        Gold.1              0
        Silver.1            0
        Bronze.1            0
        Total.1             0
        # Games            13
        Gold.2              0
        Silver.2            0
        Bronze.2            2
        Combined total      2
        ID                AFG
        Name: Afghanistan, dtype: object
```

1.0.2 Question 1

Which country has won the most gold medals in summer games?
This function should return a single string value.

```
In [28]: def answer_one():
             return df[df['Gold']==df['Gold'].max()].index[0]
         answer_one()

Out[28]: 'United States'
```

1.0.3 Question 2

Which country had the biggest difference between their summer and winter gold medal counts?
This function should return a single string value.

```
In [30]: def answer_two():
             return df[abs(df['Gold']-df['Gold.1'])==abs(df['Gold']-df['Gold.1']).max()].index[0
         answer_two()

Out[30]: 'United States'
```

1.0.4 Question 3

Which country has the biggest difference between their summer gold medal counts and winter gold medal counts relative to their total gold medal count?

$$\frac{Summer\ Gold - Winter\ Gold}{Total\ Gold}$$

Only include countries that have won at least 1 gold in both summer and winter.
This function should return a single string value.

```
In [31]: def answer_three():
             df1 = df[(df['Gold'] > 0) & (df['Gold.1'] > 0)].copy()
             df1['diff'] = abs(df1['Gold'] - df1['Gold.1'])
             df1['rel'] = df1['diff'] / df1['Gold.2']
             return df1[df1['rel'] == df1['rel'].max()].index[0]
         answer_three()

Out[31]: 'Bulgaria'
```

1.0.5 Question 4

Write a function that creates a Series called "Points" which is a weighted value where each gold medal (`Gold.2`) counts for 3 points, silver medals (`Silver.2`) for 2 points, and bronze medals (`Bronze.2`) for 1 point. The function should return only the column (a Series object) which you created, with the country names as indices.
This function should return a Series named `Points` of length 146

```
In [35]: def answer_four():
             df['Points'] = df['Gold.2'] * 3 + df['Silver.2'] * 2 + df['Bronze.2'] * 1
             return df['Points']
         answer_four()
```

```
Out[35]: Afghanistan                   2
         Algeria                      27
         Argentina                   130
         Armenia                      16
         Australasia                  22
         Australia                   923
         Austria                     569
         Azerbaijan                   43
         Bahamas                      24
         Bahrain                       1
         Barbados                      1
         Belarus                     154
         Belgium                     276
         Bermuda                       1
         Bohemia                       5
         Botswana                      2
         Brazil                      184
         British West Indies           2
         Bulgaria                    411
         Burundi                       3
         Cameroon                     12
         Canada                      846
         Chile                        24
         China                      1120
         Colombia                     29
         Costa Rica                    7
         Ivory Coast                   2
         Croatia                      67
         Cuba                        420
         Cyprus                        2
                                     ...
         Spain                       268
         Sri Lanka                     4
         Sudan                         2
         Suriname                      4
         Sweden                     1217
         Switzerland                 630
         Syria                         6
         Chinese Taipei               32
         Tajikistan                    4
         Tanzania                      4
         Thailand                     44
         Togo                          1
```

```
Tonga                                2
Trinidad and Tobago                 27
Tunisia                             19
Turkey                             191
Uganda                              14
Ukraine                            220
United Arab Emirates                 3
United States                     5684
Uruguay                             16
Uzbekistan                          38
Venezuela                           18
Vietnam                              4
Virgin Islands                       2
Yugoslavia                         171
Independent Olympic Participants     4
Zambia                               3
Zimbabwe                            18
Mixed team                          38
Name: Points, dtype: int64
```

1.1 Part 2

For the next set of questions, we will be using census data from the United States Census Bureau. Counties are political and geographic subdivisions of states in the United States. This dataset contains population data for counties and states in the US from 2010 to 2015. See this document for a description of the variable names.

The census dataset (census.csv) should be loaded as census_df. Answer questions using this as appropriate.

1.1.1 Question 5

Which state has the most counties in it? (hint: consider the sumlevel key carefully! You'll need this for future questions too...)

This function should return a single string value.

```
In [4]: census_df = pd.read_csv('census.csv')
        census_df
        census_df.head()
```

```
Out[4]:    SUMLEV  REGION  DIVISION  STATE  COUNTY   STNAME         CTYNAME  \
        0      40       3         6      1       0  Alabama         Alabama
        1      50       3         6      1       1  Alabama  Autauga County
        2      50       3         6      1       3  Alabama  Baldwin County
        3      50       3         6      1       5  Alabama  Barbour County
        4      50       3         6      1       7  Alabama     Bibb County

           CENSUS2010POP  ESTIMATESBASE2010  POPESTIMATE2010     ...       \
        0        4779736            4780127          4785161     ...
```

```
1          54571             54571             54660     ...
2         182265            182265            183193     ...
3          27457             27457             27341     ...
4          22915             22919             22861     ...

   RDOMESTICMIG2011  RDOMESTICMIG2012  RDOMESTICMIG2013  RDOMESTICMIG2014  \
0          0.002295         -0.193196          0.381066          0.582002
1          7.242091         -2.915927         -3.012349          2.265971
2         14.832960         17.647293         21.845705         19.243287
3         -4.728132         -2.500690         -7.056824         -3.904217
4         -5.527043         -5.068871         -6.201001         -0.177537

   RDOMESTICMIG2015  RNETMIG2011  RNETMIG2012  RNETMIG2013  RNETMIG2014  \
0         -0.467369     1.030015     0.826644     1.383282     1.724718
1         -2.530799     7.606016    -2.626146    -2.722002     2.592270
2         17.197872    15.844176    18.559627    22.727626    20.317142
3        -10.543299    -4.874741    -2.758113    -7.167664    -3.978583
4          0.177258    -5.088389    -4.363636    -5.403729     0.754533

   RNETMIG2015
0     0.712594
1    -2.187333
2    18.293499
3   -10.543299
4     1.107861

[5 rows x 100 columns]
```

```
In [42]: def answer_five():
             return census_df.groupby('STNAME')['COUNTY'].count().idxmax()
         answer_five()
```

```
Out[42]: 'Texas'
```

1.1.2 Question 6

Only looking at the three most populous counties for each state, what are the three most populous states (in order of highest population to lowest population)? Use `CENSUS2010POP`.

This function should return a list of string values.

```
In [46]: def answer_six():
             df1 = pd.DataFrame(census_df.where(census_df['SUMLEV'] == 50).groupby(['STNAME'])['
             df1 = df1.reset_index()

             return list(df1.groupby(['STNAME']).sum()['CENSUS2010POP'].nlargest(3).index)
         answer_six()
```

```
Out[46]: ['California', 'Texas', 'Illinois']
```

1.1.3 Question 7

Which county has had the largest absolute change in population within the period 2010-2015? (Hint: population values are stored in columns POPESTIMATE2010 through POPESTIMATE2015, you need to consider all six columns.)

e.g. If County Population in the 5 year period is 100, 120, 80, 105, 100, 130, then its largest change in the period would be |130-80| = 50.

This function should return a single string value.

```
In [5]: def answer_seven():
            d = census_df[census_df.SUMLEV==50].copy()
            d['max'] = d[['POPESTIMATE2010','POPESTIMATE2011','POPESTIMATE2012','POPESTIMATE2013
            d['min'] = d[['POPESTIMATE2010','POPESTIMATE2011','POPESTIMATE2012','POPESTIMATE2013
            d['diff'] = d['max'] - d['min']

            return d[d['diff'] == d['diff'].max()].iloc[0]['CTYNAME']
        answer_seven()

Out[5]: 'Harris County'
```

1.1.4 Question 8

In this datafile, the United States is broken up into four regions using the "REGION" column.

Create a query that finds the counties that belong to regions 1 or 2, whose name starts with 'Washington', and whose POPESTIMATE2015 was greater than their POPESTIMATE 2014.

This function should return a 5x2 DataFrame with the columns = ['STNAME', 'CTYNAME'] and the same index ID as the census_df (sorted ascending by index).

```
In [22]: def answer_eight():
             return census_df[((census_df['REGION'] == 1) | (census_df['REGION'] == 2)) & (censu

         answer_eight()

Out[22]:              STNAME            CTYNAME
         703        Illinois  Washington County
         799         Indiana  Washington County
         896            Iowa  Washington County
         1005         Kansas  Washington County
         1211          Maine  Washington County
         1419      Minnesota  Washington County
         1618       Missouri  Washington County
         1770       Nebraska  Washington County
         1918       New York  Washington County
         2162           Ohio  Washington County
         2345   Pennsylvania  Washington County
         2355   Rhode Island  Washington County
         2863        Vermont  Washington County
         3163      Wisconsin  Washington County

In [ ]:
```