Unit_Testing

January 7, 2022

0.1 Unit Testing

While we will not cover the unit testing library that python has, we wanted to introduce you to a simple way that you can test your code.

Unit testing is important because it the only way you can be sure that your code is do what you think it is doing.

Remember, just because ther are no errors does not mean your code is correct.

```
In [1]: import numpy as np
        import pandas as pd
        import matplotlib as plt
        pd.set_option('display.max_columns', 100) # Show all columns when looking at dataframe
```

```
In [25]: # Download NHANES 2015-2016 data
         df = pd.read_csv("nhanes_2015_2016.csv")
         df.index = range(1,df.shape[0]+1)
```

```
In [26]: df.head()
```

Out[26]:

	SEQN	ALQ101	ALQ110	ALQ130	SMQ020	RIAGENDR	RIDAGEYR	RIDRETH1	\
1	83732	1.0	NaN	1.0	1	1	62	3	
2	83733	1.0	NaN	6.0	1	1	53	3	
3	83734	1.0	NaN	NaN	1	1	78	3	
4	83735	2.0	1.0	1.0	2	2	56	3	
5	83736	2.0	1.0	1.0	2	2	42	4	

	DMDCITZN	DMDEDUC2	DMDMARTL	DMDHHSIZ	WTINT2YR	SDMVPSU	SDMVSTRA	\
1	1.0	5.0	1.0	2	134671.37	1	125	
2	2.0	3.0	3.0	1	24328.56	1	125	
3	1.0	3.0	1.0	2	12400.01	1	131	
4	1.0	5.0	6.0	1	102718.00	1	131	
5	1.0	4.0	3.0	5	17627.67	2	126	

	INDFMPIR	BPXSY1	BPXDI1	BPXSY2	BPXDI2	BMXWT	BMXHT	BMXBMI	BMXLEG	\
1	4.39	128.0	70.0	124.0	64.0	94.8	184.5	27.8	43.3	
2	1.32	146.0	88.0	140.0	88.0	90.4	171.4	30.8	38.0	
3	1.51	138.0	46.0	132.0	44.0	83.4	170.1	28.8	35.6	
4	5.00	132.0	72.0	134.0	68.0	109.8	160.9	42.4	38.5	

```
5      1.23   100.0    70.0   114.0    54.0   55.2  164.9    20.3    37.4

   BMXARML  BMXARMC  BMXWAIST  HIQ210
1     43.6     35.9     101.1     2.0
2     40.0     33.2     107.9     NaN
3     37.0     31.0     116.5     2.0
4     37.7     38.3     110.1     2.0
5     36.0     27.2      80.4     2.0
```

0.1.1 Goal

We want to find the mean of first 100 rows of 'BPXSY1' when 'RIDAGEYR' > 60

```
In [27]: # One possible way of doing this is:
         pd.Series.mean(df[df.RIDAGEYR > 60].loc[range(0,100), 'BPXSY1'])
         # Current version of python will include this warning, older versions will not

Out[27]: 139.57142857142858

In [30]: # test our code on only ten rows so we can easily check
         test = pd.DataFrame({'col1': np.repeat([3,1],5), 'col2': range(3,13)}, index=range(1,1
         test

Out[30]:     col1  col2
         1      3     3
         2      3     4
         3      3     5
         4      3     6
         5      3     7
         6      1     8
         7      1     9
         8      1    10
         9      1    11
         10     1    12

In [55]: # pd.Series.mean(df[df.RIDAGEYR > 60].loc[range(0,5), 'BPXSY1'])
         # should return 5

         pd.Series.mean(test[test.col1 > 2].loc[range(1,6), 'col2'])

Out[55]: 5.0
```

What went wrong?

```
In [37]: test[test.col1 > 2].loc[range(0,5), 'col2']
         # 0 is not in the row index labels because the second row's value is < 2. For now, pa
         # with NaN
```

```
Out[37]: 0    NaN
         1    3.0
         2    4.0
         3    5.0
         4    6.0
         Name: col2, dtype: float64

In [38]: # Using the .iloc method instead, we are correctly choosing the first 5 rows, regardl
         test[test.col1 >2].iloc[range(0,5), 1]

Out[38]: 1    3
         2    4
         3    5
         4    6
         5    7
         Name: col2, dtype: int64

In [39]: pd.Series.mean(test[test.col1 >2].iloc[range(0,5), 1])

Out[39]: 5.0

In [40]: # We can compare what our real dataframe looks like with the incorrect and correct me
         df[df.RIDAGEYR > 60].loc[range(0,5), :] # Filled with NaN whenever a row label does n

Out[40]:       SEQN  ALQ101  ALQ110  ALQ130  SMQ020  RIAGENDR  RIDAGEYR  RIDRETH1  \
         0      NaN     NaN     NaN     NaN     NaN       NaN       NaN       NaN
         1  83732.0     1.0     NaN     1.0     1.0       1.0      62.0       3.0
         2      NaN     NaN     NaN     NaN     NaN       NaN       NaN       NaN
         3  83734.0     1.0     NaN     NaN     1.0       1.0      78.0       3.0
         4      NaN     NaN     NaN     NaN     NaN       NaN       NaN       NaN

            DMDCITZN  DMDEDUC2  DMDMARTL  DMDHHSIZ   WTINT2YR  SDMVPSU  SDMVSTRA  \
         0       NaN       NaN       NaN       NaN        NaN      NaN       NaN
         1       1.0       5.0       1.0       2.0  134671.37      1.0     125.0
         2       NaN       NaN       NaN       NaN        NaN      NaN       NaN
         3       1.0       3.0       1.0       2.0   12400.01      1.0     131.0
         4       NaN       NaN       NaN       NaN        NaN      NaN       NaN

            INDFMPIR  BPXSY1  BPXDI1  BPXSY2  BPXDI2  BMXWT  BMXHT  BMXBMI  BMXLEG  \
         0       NaN     NaN     NaN     NaN     NaN    NaN    NaN     NaN     NaN
         1      4.39   128.0    70.0   124.0    64.0   94.8  184.5    27.8    43.3
         2       NaN     NaN     NaN     NaN     NaN    NaN    NaN     NaN     NaN
         3      1.51   138.0    46.0   132.0    44.0   83.4  170.1    28.8    35.6
         4       NaN     NaN     NaN     NaN     NaN    NaN    NaN     NaN     NaN

            BMXARML  BMXARMC  BMXWAIST  HIQ210
         0      NaN      NaN       NaN     NaN
         1     43.6     35.9     101.1     2.0
         2      NaN      NaN       NaN     NaN
```

```
3     37.0     31.0     116.5     2.0
4      NaN      NaN       NaN     NaN
```

```
In [41]: df[df.RIDAGEYR > 60].iloc[range(0,5), :] # Correct picks the first fice rows such tha
```

Out[41]:

	SEQN	ALQ101	ALQ110	ALQ130	SMQ020	RIAGENDR	RIDAGEYR	RIDRETH1	\
1	83732	1.0	NaN	1.0	1	1	62	3	
3	83734	1.0	NaN	NaN	1	1	78	3	
6	83737	2.0	2.0	NaN	2	2	72	1	
14	83754	2.0	1.0	1.0	2	2	67	2	
15	83755	1.0	NaN	3.0	2	1	67	4	

	DMDCITZN	DMDEDUC2	DMDMARTL	DMDHHSIZ	WTINT2YR	SDMVPSU	SDMVSTRA	\
1	1.0	5.0	1.0	2	134671.37	1	125	
3	1.0	3.0	1.0	2	12400.01	1	131	
6	2.0	2.0	4.0	5	11252.31	1	128	
14	1.0	5.0	1.0	7	10495.87	1	128	
15	1.0	5.0	2.0	1	14080.10	1	126	

	INDFMPIR	BPXSY1	BPXDI1	BPXSY2	BPXDI2	BMXWT	BMXHT	BMXBMI	BMXLEG	\
1	4.39	128.0	70.0	124.0	64.0	94.8	184.5	27.8	43.3	
3	1.51	138.0	46.0	132.0	44.0	83.4	170.1	28.8	35.6	
6	2.82	116.0	58.0	122.0	58.0	64.4	150.0	28.6	34.4	
14	0.89	124.0	76.0	116.0	64.0	117.8	164.1	43.7	34.8	
15	2.04	132.0	84.0	136.0	82.0	97.4	183.8	28.8	42.5	

	BMXARML	BMXARMC	BMXWAIST	HIQ210
1	43.6	35.9	101.1	2.0
3	37.0	31.0	116.5	2.0
6	33.5	31.4	92.9	NaN
14	38.6	42.7	123.0	2.0
15	40.6	34.2	106.3	2.0

```
In [42]: # Applying the correct method to the original question about BPXSY1
         print(pd.Series.mean(df[df.RIDAGEYR > 60].iloc[range(0,100), 16]))

         # Another way to reference the BPXSY1 variable
         print(pd.Series.mean(df[df.RIDAGEYR > 60].iloc[range(0,100), df.columns.get_loc('BPXS
```

```
136.29166666666666
136.29166666666666
```

```
In [57]: print(pd.Series.mean(df[df.RIDAGEYR > 60].loc[range(1,101), 'BPXSY1']))
```

```
139.57142857142858
```

```
In [64]: df.columns.get_loc('BPXSY1')
```

Out[64]: 16