

Asg 19.4

Working with Pandas DataFrames -- The Essentials (Part Two)

(Coding)



Files needed for this assignment:

[marketing_campaign.csv](https://drive.google.com/file/d/1Lx2V9-9j_t_9hTSdFpLmmP_5M_1sE4Wf/view?usp=share_link) (https://drive.google.com/file/d/1Lx2V9-9j_t_9hTSdFpLmmP_5M_1sE4Wf/view?usp=share_link)

[houses_train.txt](https://drive.google.com/file/d/1E1SzyR0dhIOrMloKs3UBL8pibFpVwLis/view?usp=share_link) (https://drive.google.com/file/d/1E1SzyR0dhIOrMloKs3UBL8pibFpVwLis/view?usp=share_link)

```
In [4]: 1 # set up notebook to display multiple output in one cell
        2
        3 from IPython.core.interactiveshell import InteractiveShell
        4 InteractiveShell.ast_node_interactivity = "all"
        5
        6 print('The notebook is set up to display multiple output in one cell.')
```

The notebook is set up to display multiple output in one cell.

```
In [3]: 1 # conventional way to import pandas and numpy
        2
        3 import pandas as pd
        4 import numpy as np
```

PART ONE

For Questions 1-7: We will be using the **'marketing_campaign.csv'** dataset and the **customers DataFrame**.

Question 1:

Read in the dataset **'marketing_campaign.csv'** and store the results in a DataFrame named **customers**.

[marketing_campaign.csv \(https://drive.google.com/file/d/1Lx2V9-9j_t_9hTSdFpLmmP_5M_1sE4Wf/view?usp=share_link\)](https://drive.google.com/file/d/1Lx2V9-9j_t_9hTSdFpLmmP_5M_1sE4Wf/view?usp=share_link)

Question 2:

a. Read in the dataset **'marketing_campaign.csv'** and store the results in a DataFrame named **customers**.

Note: This is the same as Question 1.

b. Use the appropriate attributes or methods to inspect the **customers** DataFrame.

In [6]:

```
1 customers = pd.read_csv('marketing_campaign.csv', sep=';')
2 print(customers.index)
3 print(customers.columns)
4 print(customers.shape)
5 print(customers.dtypes)

RangeIndex(start=0, stop=2240, step=1)
Index(['ID', 'Year_Birth', 'Education', 'Marital_Status', 'Income', 'Kidhome',
      'Teenhome', 'Dt_Customer', 'Recency', 'MntWines', 'MntFruits',
      'MntMeatProducts', 'MntFishProducts', 'MntSweetProducts',
      'MntGoldProds', 'NumDealsPurchases', 'NumWebPurchases',
      'NumCatalogPurchases', 'NumStorePurchases', 'NumWebVisitsMonth',
      'AcceptedCmp3', 'AcceptedCmp4', 'AcceptedCmp5', 'AcceptedCmp1',
      'AcceptedCmp2', 'Complain', 'Z_CostContact', 'Z_Revenue', 'Response'],
      dtype='object')
(2240, 29)
ID                int64
Year_Birth        int64
Education         object
Marital_Status    object
Income            float64
Kidhome           int64
Teenhome          int64
Dt_Customer       object
Recency           int64
MntWines          int64
MntFruits         int64
MntMeatProducts   int64
MntFishProducts   int64
MntSweetProducts  int64
MntGoldProds      int64
NumDealsPurchases int64
NumWebPurchases   int64
NumCatalogPurchases int64
NumStorePurchases int64
NumWebVisitsMonth int64
AcceptedCmp3      int64
AcceptedCmp4      int64
AcceptedCmp5      int64
AcceptedCmp1      int64
AcceptedCmp2      int64
Complain          int64
Z_CostContact     int64
Z_Revenue         int64
Response          int64
dtype: object
```

Question 3:

a. Sort the **'Year_Birth'** Series of the **customers** DataFrame in ascending order ... (this should return a Series).

b. Use an appropriate method to check that the Series was sorted properly.

```
In [11]: 1 customers['Year_Birth'].sort_values()
```

```
Out[11]: 239      1893
          339      1899
          192      1900
          1950     1940
          424      1941
          ...
          747      1995
          1850     1995
          696      1995
          1170     1996
          46       1996
Name: Year_Birth, Length: 2240, dtype: int64
```

Question 4:

a. Sort the **'Income'** Series of the **customers** DataFrame in descending order ... (this should return a Series).

b. Use an appropriate method to check that the Series was sorted properly.

```
In [12]: 1 customers['Income'].sort_values(ascending=False)
```

```
Out[12]: 2233      666666.0
          617      162397.0
          687      160803.0
          1300     157733.0
          164      157243.0
          ...
          2078      NaN
          2079      NaN
          2081      NaN
          2084      NaN
          2228      NaN
Name: Income, Length: 2240, dtype: float64
```

Question 5:

a. Sort the entire **customers** DataFrame by the **'Year_Birth'** Series in ascending order ... (this should return a DataFrame).

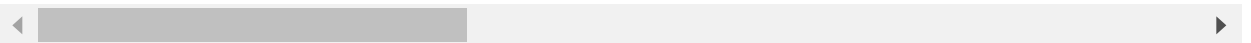
b. Use an appropriate method to check that the DataFrame was sorted properly.

```
In [13]: 1 customers.sort_values(by=['Year_Birth'])
```

Out[13]:

	ID	Year_Birth	Education	Marital_Status	Income	Kidhome	Teenhome	Dt_Customer	R
239	11004	1893	2n Cycle	Single	60182.0	0	1	2014-05-17	
339	1150	1899	PhD	Together	83532.0	0	0	2013-09-26	
192	7829	1900	2n Cycle	Divorced	36640.0	1	0	2013-09-26	
1950	6663	1940	PhD	Single	51141.0	0	0	2013-07-08	
424	6932	1941	PhD	Married	93027.0	0	0	2013-04-13	
...
747	10548	1995	Graduation	Single	71163.0	0	0	2014-03-09	
1850	4427	1995	2n Cycle	Single	83257.0	0	0	2012-09-18	
696	8315	1995	Graduation	Single	34824.0	0	0	2014-03-26	
1170	193	1996	Basic	Married	14421.0	0	0	2014-02-17	
46	9909	1996	2n Cycle	Married	7500.0	0	0	2012-11-09	

2240 rows × 29 columns



Question 6:

a. Sort the entire **customers** DataFrame by the **'Income'** Series in descending order ... (this should return a DataFrame).

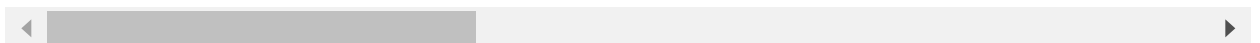
b. Use an appropriate method to check that the DataFrame was sorted properly.

```
In [14]: 1 customers.sort_values(by=['Income'])
```

Out[14]:

	ID	Year_Birth	Education	Marital_Status	Income	Kidhome	Teenhome	Dt_Customer	Re
1245	6862	1971	Graduation	Divorced	1730.0	0	0	2014-05-18	
21	5376	1979	Graduation	Married	2447.0	1	0	2013-01-06	
1524	11110	1973	Graduation	Single	3502.0	1	0	2013-04-13	
1846	9931	1963	PhD	Married	4023.0	1	1	2014-06-23	
1975	10311	1969	Graduation	Married	4428.0	0	1	2013-10-05	
...
2078	5079	1971	Graduation	Married	NaN	1	1	2013-03-03	
2079	10339	1954	Master	Together	NaN	0	1	2013-06-23	
2081	3117	1955	Graduation	Single	NaN	0	1	2013-10-18	
2084	5250	1943	Master	Widow	NaN	0	0	2013-10-30	
2228	8720	1978	2n Cycle	Together	NaN	0	0	2012-08-12	

2240 rows × 29 columns



Question 7:

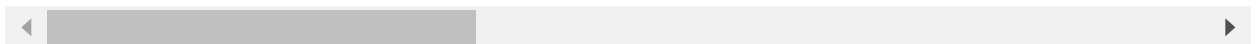
- Sort the entire **customers** DataFrame first by '**Income**', then by '**Year_Birth**'.
- Use an appropriate method to check that the DataFrame was sorted properly.

```
In [15]: 1 customers = customers.sort_values(by=['Income'])
         2 customers.sort_values(by=['Year_Birth'])
```

Out[15]:

	ID	Year_Birth	Education	Marital_Status	Income	Kidhome	Teenhome	Dt_Customer	R
239	11004	1893	2n Cycle	Single	60182.0	0	1	2014-05-17	
339	1150	1899	PhD	Together	83532.0	0	0	2013-09-26	
192	7829	1900	2n Cycle	Divorced	36640.0	1	0	2013-09-26	
1950	6663	1940	PhD	Single	51141.0	0	0	2013-07-08	
424	6932	1941	PhD	Married	93027.0	0	0	2013-04-13	
...
1850	4427	1995	2n Cycle	Single	83257.0	0	0	2012-09-18	
747	10548	1995	Graduation	Single	71163.0	0	0	2014-03-09	
2213	3661	1995	2n Cycle	Single	80617.0	0	0	2012-10-12	
1170	193	1996	Basic	Married	14421.0	0	0	2014-02-17	
46	9909	1996	2n Cycle	Married	7500.0	0	0	2012-11-09	

2240 rows × 29 columns



Question 8:

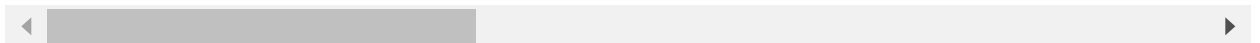
- Use the `sort_index()` method to sort the entire **customers** DataFrame in reverse order.
- Use an appropriate method to check that the DataFrame was sorted properly.

In [16]: 1 customers.sort_index(ascending=False)

Out[16]:

	ID	Year_Birth	Education	Marital_Status	Income	Kidhome	Teenhome	Dt_Customer	R
2239	9405	1954	PhD	Married	52869.0	1	1	2012-10-15	
2238	8235	1956	Master	Together	69245.0	0	1	2014-01-24	
2237	7270	1981	Graduation	Divorced	56981.0	0	0	2014-01-25	
2236	4001	1946	PhD	Together	64014.0	2	1	2014-06-10	
2235	10870	1967	Graduation	Married	61223.0	0	1	2013-06-13	
...
4	5324	1981	PhD	Married	58293.0	1	0	2014-01-19	
3	6182	1984	Graduation	Together	26646.0	1	0	2014-02-10	
2	4141	1965	Graduation	Together	71613.0	0	0	2013-08-21	
1	2174	1954	Graduation	Single	46344.0	1	1	2014-03-08	
0	5524	1957	Graduation	Single	58138.0	0	0	2012-09-04	

2240 rows × 29 columns



Question 9:

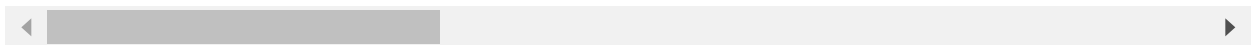
- Use the `sort_index()` method to sort the columns of the entire **customers** DataFrame in reverse order.
- Use an appropriate method to check that the DataFrame was sorted properly.


```
In [17]: 1 customers.sort_index(axis=1, ascending=False)
```

Out[17]:

	Z_Revenue	Z_CostContact	Year_Birth	Teenhome	Response	Recency	NumWebVisitsMonth
1245	11	3	1971	0	0	65	20
21	11	3	1979	0	0	42	1
1524	11	3	1973	0	0	56	14
1846	11	3	1963	1	0	29	19
1975	11	3	1969	1	0	0	1
...
2078	11	3	1971	1	0	82	8
2079	11	3	1954	1	0	83	6
2081	11	3	1955	1	0	95	7
2084	11	3	1943	0	1	75	1
2228	11	3	1978	0	0	53	0

2240 rows × 29 columns



Question 10:

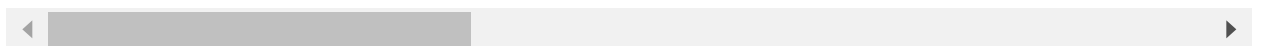
- Use the `set_index()` method to set the **ID** column as the row index.
- Use an appropriate method to check that the `DataFrame` was sorted properly.

```
In [18]: 1 customers.set_index(['ID'])
```

```
Out[18]:
```

	Year_Birth	Education	Marital_Status	Income	Kidhome	Teenhome	Dt_Customer	Recency
ID								
6862	1971	Graduation	Divorced	1730.0	0	0	2014-05-18	65
5376	1979	Graduation	Married	2447.0	1	0	2013-01-06	42
11110	1973	Graduation	Single	3502.0	1	0	2013-04-13	56
9931	1963	PhD	Married	4023.0	1	1	2014-06-23	29
10311	1969	Graduation	Married	4428.0	0	1	2013-10-05	0
...
5079	1971	Graduation	Married	NaN	1	1	2013-03-03	82
10339	1954	Master	Together	NaN	0	1	2013-06-23	83
3117	1955	Graduation	Single	NaN	0	1	2013-10-18	95
5250	1943	Master	Widow	NaN	0	0	2013-10-30	75
8720	1978	2n Cycle	Together	NaN	0	0	2012-08-12	53

2240 rows × 28 columns



Question 11:

Read in the dataset '**marketing_campaign.csv**', but this time set the **ID** column as the row index in your `read_csv` statement. Store the results in a DataFrame named **customers**.

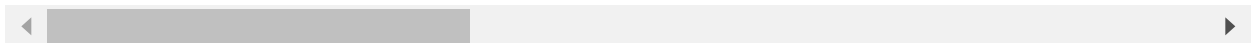
[marketing_campaign.csv](https://drive.google.com/file/d/1Lx2V9-9j_t_9hTSdFpLmmP_5M_1sE4Wf/view?usp=share_link) (https://drive.google.com/file/d/1Lx2V9-9j_t_9hTSdFpLmmP_5M_1sE4Wf/view?usp=share_link).

```
In [19]: 1 customers = pd.read_csv('marketing_campaign.csv', sep=';', index_col='ID')
         2 customers
```

Out[19]:

	Year_Birth	Education	Marital_Status	Income	Kidhome	Teenhome	Dt_Customer	Recency
ID								
5524	1957	Graduation	Single	58138.0	0	0	2012-09-04	58
2174	1954	Graduation	Single	46344.0	1	1	2014-03-08	38
4141	1965	Graduation	Together	71613.0	0	0	2013-08-21	26
6182	1984	Graduation	Together	26646.0	1	0	2014-02-10	26
5324	1981	PhD	Married	58293.0	1	0	2014-01-19	94
...
10870	1967	Graduation	Married	61223.0	0	1	2013-06-13	46
4001	1946	PhD	Together	64014.0	2	1	2014-06-10	56
7270	1981	Graduation	Divorced	56981.0	0	0	2014-01-25	91
8235	1956	Master	Together	69245.0	0	1	2014-01-24	8
9405	1954	PhD	Married	52869.0	1	1	2012-10-15	40

2240 rows × 28 columns



In []:

1