DSO 545: Statistical Computing and Data Visualization

Abbass Al Sharif

Fall 2019

Lab 02: Data Manipulation using Pandas

Contents

---

1. Import the `pandas` Python package. If you don't have the package installed, then you might use the following line to install it using the terminal (Mac) or command prompt (windows):

```
conda install pandas
```

```
import pandas as pd
```

Pandas Series Data Structure

2. Create two series `name` ("Dona", "James", "Alice", "USC Marshall") and `age` (25, 35, 19, 99).

```
import pandas as pd

name = pd.Series(["Dona", "James", "Alice", "USC Marshall"])
age =pd.Series([25, 35, 19, 99]) # USC Marshall is turning 100 this year! (2020)

name
```

```
## 0            Dona
## 1           James
## 2           Alice
## 3    USC Marshall
## dtype: object
```

```
age
```

```
## 0    25
## 1    35
## 2    19
## 3    99
## dtype: int64
```

3. Update the last element of the `age` series to 100.

```
age[3]
```

```
## 99
```

```
age[3] = 100 #update USC Marshall's age to 100
age
```

```
## 0     25
## 1     35
## 2     19
## 3    100
## dtype: int64
```

4. Find the average, min, and max age.

```
age.mean()
```

```
## 44.75
```

```
age.max()
```

```
## 100
```

```
age.min()
```

```
## 19
```

5. Get the summary statistics for age.

```
age.describe()
```

```
## count      4.000000
## mean      44.750000
## std       37.419914
## min       19.000000
## 25%       23.500000
## 50%       30.000000
## 75%       51.250000
## max      100.000000
## dtype: float64
```

6. Find all ages that are below the average age.

```
age[age < age.mean()]
```

```
## 0    25
## 1    35
## 2    19
## dtype: int64
```

Pandas DataFrame Data Structure

7. Create a dataframe called `data` that has the two series `name` and `age` as its columns.

```
import pandas as pd

data = pd.DataFrame({"Name":name, "Age":age})
data

### OR

# You can also create a dataframe using lists as follows
```

```
##            Name  Age
## 0          Dona   25
## 1         James   35
## 2         Alice   19
## 3  USC Marshall  100
```

```
name = ["Dona", "James", "Alice", "USC Marshall"]
age =[25, 35, 19, 100] # USC Marshall is turning 100 this year! (2020)

name
```

```
## ['Dona', 'James', 'Alice', 'USC Marshall']
```

```
age
```

```
## [25, 35, 19, 100]
```

```
data = pd.DataFrame({"Name":name, "Age":age})
data
```

```
##            Name  Age
## 0          Dona   25
## 1         James   35
## 2         Alice   19
## 3  USC Marshall  100
```

8. Add a third column ("Person") to the dataframe defined before. The cells in this column has a value of "Yes" if the corresponding entity is a person, and "No" otherwise.

```
data["Person"] = ["Yes","Yes", "Yes", "No"]

data
```

```
##            Name  Age Person
## 0          Dona   25    Yes
## 1         James   35    Yes
## 2         Alice   19    Yes
## 3  USC Marshall  100     No
```

9. Rearrange the columns in the dataframe as follows: Name, Person, and Age.

```
cols= ["Name", "Person", "Age"]
data = data[cols]

data
```

```
##            Name Person  Age
## 0          Dona    Yes   25
## 1         James    Yes   35
## 2         Alice    Yes   19
## 3  USC Marshall     No  100
```

10. Load the dataset `orders.csv` into a Pandas dataframe called `orders`.

```
orders = pd.read_csv("orders.csv", delimiter=",", header = 0)
```

11. Print the first five observations in the orders dataset.

```
orders.head()
```

```
##            customer     products     sales_person sales_region sales_date  \
```

```
## 0  LONG ISLANDS INC  SOFT DRINKS  Michael Jackson     AMERICAS    4/13/12
## 1  LONG ISLANDS INC  SOFT DRINKS  Michael Jackson     AMERICAS   12/21/12
## 2  LONG ISLANDS INC  SOFT DRINKS  Michael Jackson     AMERICAS   12/24/12
## 3  LONG ISLANDS INC  SOFT DRINKS  Michael Jackson     AMERICAS   12/24/12
## 4  LONG ISLANDS INC  SOFT DRINKS  Michael Jackson     AMERICAS   12/29/12
##
##   quarter  sales  costs
## 0      Q2  24640  16999
## 1      Q4  24640  13059
## 2      Q4  29923  13826
## 3      Q4  66901  18658
## 4      Q4  63116  19949
```

12. Print the last 10 observations in the orders dataset.

```
orders.tail(10)
```

```
##               customer products   sales_person sales_region sales_date  \
## 566  GIN ON THE RUN CO    TONIC  Homer Simpson       AFRICA    12/8/14
## 567  GIN ON THE RUN CO    TONIC  Homer Simpson       AFRICA    12/8/14
## 568  GIN ON THE RUN CO    TONIC  Homer Simpson       AFRICA    12/8/14
## 569  GIN ON THE RUN CO    TONIC  Homer Simpson       AFRICA   12/17/14
## 570  GIN ON THE RUN CO    TONIC  Homer Simpson       AFRICA   12/17/14
## 571  GIN ON THE RUN CO    TONIC  Homer Simpson       AFRICA   12/17/14
## 572  GIN ON THE RUN CO    TONIC  Homer Simpson       AFRICA   12/17/14
## 573  GIN ON THE RUN CO    TONIC  Homer Simpson       AFRICA     1/5/14
## 574  GIN ON THE RUN CO    TONIC  Homer Simpson       AFRICA     1/5/14
## 575  GIN ON THE RUN CO    TONIC  Homer Simpson       AFRICA     1/5/14
##
##     quarter  sales  costs
## 566      Q4  50033  16007
## 567      Q4  50577  19377
## 568      Q4  54040  10098
## 569      Q4  45057  14462
## 570      Q4  35558  13330
## 571      Q4  21217  14334
## 572      Q4  60244  18345
## 573      Q1  76362  15399
## 574      Q1  60119  15703
## 575      Q1  45139  13952
```

13. For the `orders` dataframe, run the two methods `.describe()` and `.info()`. What does each method do?

```
orders.describe() # it will only give summary stats to numerical variables
```

```
##              sales         costs
## count    576.000000    576.000000
## mean   55664.314236  15085.576389
## std    25887.689778   2868.591122
## min    10014.000000  10006.000000
## 25%    32562.750000  12708.750000
## 50%    56809.000000  15033.500000
## 75%    76732.500000  17493.250000
## max    99878.000000  19994.000000
```

```
orders.info()
```

```
## <class 'pandas.core.frame.DataFrame'>
## RangeIndex: 576 entries, 0 to 575
## Data columns (total 8 columns):
## customer        576 non-null object
## products        576 non-null object
## sales_person    576 non-null object
## sales_region    576 non-null object
## sales_date      576 non-null object
## quarter         576 non-null object
## sales           576 non-null int64
## costs           576 non-null int64
## dtypes: int64(2), object(6)
## memory usage: 36.1+ KB
```

14. Extract the customer column from the orders dataframe.

```
customers = orders['customer']
customers.head()

# OR
```

```
## 0    LONG ISLANDS INC
## 1    LONG ISLANDS INC
## 2    LONG ISLANDS INC
## 3    LONG ISLANDS INC
## 4    LONG ISLANDS INC
## Name: customer, dtype: object
```

```
customers = orders.customer
```

15. What are the different products in the orders dataset?

```
orders['products'].unique()
```

```
## array(['SOFT DRINKS', 'BOTTLES', 'ICE CUBES', 'TONIC'], dtype=object)
```

16. Create a dataset (called data) that has only the customers, products, and sales information from the orders dataset.

```
data = orders[['customer', "products", "sales"]] ## include the columns as a list
data.head()
```

```
##            customer     products  sales
## 0  LONG ISLANDS INC  SOFT DRINKS  24640
## 1  LONG ISLANDS INC  SOFT DRINKS  24640
## 2  LONG ISLANDS INC  SOFT DRINKS  29923
## 3  LONG ISLANDS INC  SOFT DRINKS  66901
## 4  LONG ISLANDS INC  SOFT DRINKS  63116
```

Indexing Techniques

There are two indexers that are very useful in slicing and dicing your dataset.

- loc[ ] : subset using index value/label
- iloc[ ] : subset using index position (integer)

17. Print the sales value for the first column in the orders dataset.

```
orders.iloc[0,6] #DATAFRAME.iloc[ROW_INDEX, COL_INDEX]
```

```
## 24640
```

18. Print the sales, costs, and sales date for the second order in the orders dataset.

```
orders.iloc[1,[6,7,4]]
```

```
## sales            24640
## costs            13059
## sales_date    12/21/12
## Name: 1, dtype: object
```

19. Print the sales, costs, and sales date for the **second and fifth** orders:

```
orders.iloc[[1,4],[4,6,7]]
```

```
##   sales_date  sales  costs
## 1   12/21/12  24640  13059
## 4   12/29/12  63116  19949
```

20. Print all information for the first order.

```
orders.iloc[0,:]
```

```
## customer        LONG ISLANDS INC
## products             SOFT DRINKS
## sales_person     Michael Jackson
## sales_region            AMERICAS
## sales_date               4/13/12
## quarter                       Q2
## sales                      24640
## costs                      16999
## Name: 0, dtype: object
```

21. Print the sales for the first 15 observations using both `iloc[]` and `.loc[]`.

```
orders.iloc[0:14, 6]

#OR
```

```
## 0     24640
## 1     24640
## 2     29923
## 3     66901
## 4     63116
## 5     38281
## 6     57650
## 7     90967
## 8     11910
## 9     59531
## 10    88297
## 11    87868
## 12    95527
## 13    90599
## Name: sales, dtype: int64
```

```
orders.loc[0:14, 'sales']
```

```
## 0     24640
## 1     24640
## 2     29923
## 3     66901
## 4     63116
## 5     38281
## 6     57650
## 7     90967
## 8     11910
## 9     59531
## 10    88297
## 11    87868
## 12    95527
## 13    90599
## 14    17030
## Name: sales, dtype: int64
```

Vectorized Computations in Python with Pandas

22. Create a list called ages (20, 45, 23, 40, 26).

```
ages = [20, 45, 23, 40, 26]
ages
```

```
## [20, 45, 23, 40, 26]
```

23. Add one year to each age in the `ages` list created in the previous question.

```
new_ages = [] # initialize an empty list to hold the updated ages

for i in ages:
  new_ages.append(i+1)

new_ages
```

```
## [21, 46, 24, 41, 27]
```

24. Create a pandas Series called ages (20, 45, 23, 40, 26).

```
ages = pd.Series([20, 45, 23, 40, 26])
```

25. Add one year to each age in the `ages` pandas series created in the previous question.

```
new_ages = ages + 1
new_ages
```

```
## 0    21
## 1    46
## 2    24
## 3    41
## 4    27
## dtype: int64
```

Conditional Subsetting the Data

26. Create a variable called `myage` which stores `78`. What is the output of the following Python code (`age < 90`)?

```
myage = 24
myage < 30 # comparison
```

```
## True
```

27. What is the output of comparting the `ages` pandas series to 30 (`ages <30`)?

```
ages = pd.Series([20, 45, 23, 40, 26])
ages < 30
```

```
## 0     True
## 1    False
## 2     True
## 3    False
## 4     True
## dtype: bool
```

28. Find all ages in the `ages` pandas series defined earlier that are less than 30.

```
# this will selecet all ages except the last one in the list
ages[[True, True, True, True, False]]

# this will select all ages less than 30
```

```
## 0    20
## 1    45
## 2    23
## 3    40
## dtype: int64
```

```
ages[ages < 30]
```

```
## 0    20
## 2    23
## 4    26
## dtype: int64
```

29. Which orders in the `orders` dataset has sales greater than or equal to $97,000?

```
orders.loc[data.sales >= 97000, :]
```

```
##               customer     products      sales_person sales_region sales_date  \
## 115   LONG ISLANDS INC       BOTTLES   Michael Jackson     AMERICAS   10/14/14
## 129   LONG ISLANDS INC     ICE CUBES   Michael Jackson     AMERICAS    9/21/14
## 165        MOJITOS R US       BOTTLES        Ian Wright       EUROPE   10/11/12
## 184        MOJITOS R US         TONIC        Ian Wright       EUROPE   11/22/12
## 185        MOJITOS R US         TONIC        Ian Wright       EUROPE   11/11/12
## 273        MOJITOS R US     ICE CUBES        Ian Wright       EUROPE     4/1/14
## 304  TEQUILA TACOS LTD       BOTTLES  John Michaloudis         ASIA   11/14/12
## 387  TEQUILA TACOS LTD   SOFT DRINKS  John Michaloudis         ASIA    8/4/14
## 425  TEQUILA TACOS LTD         TONIC  John Michaloudis         ASIA    6/23/14
## 440  GIN ON THE RUN CO   SOFT DRINKS     Homer Simpson       AFRICA   12/14/12
```

```
## 470  GIN ON THE RUN CO        TONIC     Homer Simpson       AFRICA     6/8/12
## 491  GIN ON THE RUN CO  SOFT DRINKS     Homer Simpson       AFRICA    12/9/13
## 492  GIN ON THE RUN CO      BOTTLES     Homer Simpson       AFRICA    5/18/13
## 552  GIN ON THE RUN CO    ICE CUBES     Homer Simpson       AFRICA     4/1/14
##
##     quarter  sales  costs
## 115      Q4  97950  11999
## 129      Q3  98236  19955
## 165      Q4  99220  10667
## 184      Q4  99542  14950
## 185      Q4  99202  16237
## 273      Q2  97708  14793
## 304      Q4  98116  15526
## 387      Q3  97319  18234
## 425      Q2  99878  12500
## 440      Q4  99101  11736
## 470      Q2  98483  19217
## 491      Q4  97854  17861
## 492      Q2  98852  10634
## 552      Q2  97314  17060
```

30. Create a subset dataset called (orders_GIN) where the customer is `GIN ON THE RUN CO` and include only the columns: products, sales, and costs.

```
orders_2012 = orders.loc[orders.customer == 'GIN ON THE RUN CO', orders.columns[[1,4,5]]]

#OR
orders_2012 = orders.loc[orders.customer == 'GIN ON THE RUN CO',].iloc[:,[1,4,5]]
```

Numerical Summary Statistics

31. What are the average, median, min, and max sales for "SOFT DRINKS" for all years.

```
# Average
orders.loc[orders.products == "SOFT DRINKS", "sales"].mean()
```

```
## 54227.680555555555
```

```
#median
orders.loc[orders.products == "SOFT DRINKS", "sales"].median()
```

```
## 56873.0
```

```
#min
orders.loc[orders.products == "SOFT DRINKS", "sales"].min()

#max
```

```
## 10690
```

```
orders.loc[orders.products == "SOFT DRINKS", "sales"].max()
```

```
## 99101
```

32. What is the 25th percentile for all the sales for "SOFT DRINKS" sold by "Michael Jackson"?

```
orders.loc[(orders.products == "SOFT DRINKS") & (orders.sales_person == "Michael Jackson"), "sales"].qua
```

```
## 22582.5
```