Lecture-10 (Pandas)

June 11, 2021

```
In [1]: import pandas as pd
```

```
In [2]: df = pd.read_csv('nyc_weather.csv')
        df
```

Out[2]:

	EST	Temperature	DewPoint	Humidity	Sea Level PressureIn	\
0	1/1/2016	38	23	52	30.03	
1	1/2/2016	36	18	46	30.02	
2	1/3/2016	40	21	47	29.86	
3	1/4/2016	25	9	44	30.05	
4	1/5/2016	20	-3	41	30.57	
5	1/6/2016	33	4	35	30.50	
6	1/7/2016	39	11	33	30.28	
7	1/8/2016	39	29	64	30.20	
8	1/9/2016	44	38	77	30.16	
9	1/10/2016	50	46	71	29.59	
10	1/11/2016	33	8	37	29.92	
11	1/12/2016	35	15	53	29.85	
12	1/13/2016	26	4	42	29.94	
13	1/14/2016	30	12	47	29.95	
14	1/15/2016	43	31	62	29.82	
15	1/16/2016	47	37	70	29.52	
16	1/17/2016	36	23	66	29.78	
17	1/18/2016	25	6	53	29.83	
18	1/19/2016	22	3	42	30.03	
19	1/20/2016	32	15	49	30.13	
20	1/21/2016	31	11	45	30.15	
21	1/22/2016	26	6	41	30.21	
22	1/23/2016	26	21	78	29.77	
23	1/24/2016	28	11	53	29.92	
24	1/25/2016	34	18	54	30.25	
25	1/26/2016	43	29	56	30.03	
26	1/27/2016	41	22	45	30.03	
27	1/28/2016	37	20	51	29.90	
28	1/29/2016	36	21	50	29.58	
29	1/30/2016	34	16	46	30.01	
30	1/31/2016	46	28	52	29.90	

	VisibilityMiles	WindSpeedMPH	PrecipitationIn	CloudCover	Events	WindDirDegrees
0	10	8.0	0	5	NaN	281
1	10	7.0	0	3	NaN	275
2	10	8.0	0	1	NaN	277
3	10	9.0	0	3	NaN	345
4	10	5.0	0	0	NaN	333
5	10	4.0	0	0	NaN	259
6	10	2.0	0	3	NaN	293
7	10	4.0	0	8	NaN	79
8	9	8.0	T	8	Rain	76
9	4	NaN	1.8	7	Rain	109
10	10	NaN	0	1	NaN	289
11	10	6.0	T	4	NaN	235
12	10	10.0	0	0	NaN	284
13	10	5.0	T	7	NaN	266
14	9	5.0	T	2	NaN	
15	8	7.0	0.24	7	Rain	
16	8	6.0	0.05	6	Fog-Snow	
17	9	12.0	T	2	Snow	
18	10	11.0	0	1	NaN	
19	10	6.0	0	2	NaN	
20	10	6.0	0	1	NaN	
21	9	NaN	0.01	3	Snow	
22	1	16.0	2.31	8	Fog-Snow	
23	8	6.0	T	3	Snow	
24	10	3.0	0	2	NaN	
25	10	7.0	0	2	NaN	
26	10	7.0	T	3	Rain	
27	10	5.0	0	1	NaN	
28	10	8.0	0	4	NaN	
29	10	7.0	0	0	NaN	
30	10	5.0	0	0	NaN	

```
14              101
15              340
16              345
17              293
18              293
19              302
20              312
21               34
22               42
23              327
24              286
25              244
26              311
27              234
28              298
29              257
30              241
```

```
In [3]: #Find maximum temperature of the month
        print(df['Temperature'].max())
```

```
50
```

```
In [4]: #find the days, when its rains

        print(df['EST'][df['Events'] == 'Rain'])
```

```
8      1/9/2016
9     1/10/2016
15    1/16/2016
26    1/27/2016
Name: EST, dtype: object
```

```
In [5]: #Find the day when temperature of the month is heighest
        print(df['EST'][df['Temperature'] == df['Temperature'].max()])
```

```
9    1/10/2016
Name: EST, dtype: object
```

```
In [6]: #Find average windspeed

        print(df['WindSpeedMPH'].mean())
```

```
6.892857142857143
```

In [7]:
```
#Making dataFrame using list of tuples

weather_data = [
    ('1/1/2017', 32, 6, 'Rain'),
    ('1/2/2017', 35, 7, 'Sunny'),
    ('1/3/2017', 28, 2, 'Snow'),
    ('1/4/2017', 24, 7, 'Snow'),
    ('1/5/2017', 32, 4, 'Rain'),
    ('1/6/2017', 31, 2, 'Sunny')
]

df = pd.DataFrame(weather_data, columns=['day', 'temperature', 'wind_speed', 'event'])

df
```

Out[7]:

	day	temperature	wind_speed	event
0	1/1/2017	32	6	Rain
1	1/2/2017	35	7	Sunny
2	1/3/2017	28	2	Snow
3	1/4/2017	24	7	Snow
4	1/5/2017	32	4	Rain
5	1/6/2017	31	2	Sunny

In [8]:
```
#get the dimension of a dataFrame: (rows, columns)

print(df.shape)
```

```
(6, 4)
```

In [9]:
```
#df.head() return first 5 rows Similarly df.tail() return last 5 rows
df.head()
```

Out[9]:

	day	temperature	wind_speed	event
0	1/1/2017	32	6	Rain
1	1/2/2017	35	7	Sunny
2	1/3/2017	28	2	Snow
3	1/4/2017	24	7	Snow
4	1/5/2017	32	4	Rain

In [10]:
```
df[2:5] #it's returns the row id 2 to (5-1) or 4.
```

Out[10]:

	day	temperature	wind_speed	event
2	1/3/2017	28	2	Snow
3	1/4/2017	24	7	Snow
4	1/5/2017	32	4	Rain

In [11]:
```
df.columns #returns all the columns name in a list
```

Out[11]:
```
Index(['day', 'temperature', 'wind_speed', 'event'], dtype='object')
```

```
In [12]: df.day #df.day and df['day'] return same value
```

```
Out[12]: 0    1/1/2017
         1    1/2/2017
         2    1/3/2017
         3    1/4/2017
         4    1/5/2017
         5    1/6/2017
         Name: day, dtype: object
```

```
In [13]: #get data from more then one column

         df[['day', 'event']]
```

```
Out[13]:         day  event
         0  1/1/2017   Rain
         1  1/2/2017  Sunny
         2  1/3/2017   Snow
         3  1/4/2017   Snow
         4  1/5/2017   Rain
         5  1/6/2017  Sunny
```

```
In [14]: #find statistical values of a dataFrame

         df['temperature'].describe()
```

```
Out[14]: count     6.000000
         mean     30.333333
         std       3.829708
         min      24.000000
         25%      28.750000
         50%      31.500000
         75%      32.000000
         max      35.000000
         Name: temperature, dtype: float64
```

```
In [15]: #Find the row which temperature of the month is heighest

         df[df.temperature == df.temperature.max()]
```

```
Out[15]:         day  temperature  wind_speed  event
         1  1/2/2017           35           7  Sunny
```

```
In [16]: #Find the day when temperature of the month is heighest
         df.day[df.temperature == df.temperature.max()]
```

```
Out[16]: 1    1/2/2017
         Name: day, dtype: object
```

In [17]:
```
#Read Execl File using Panda
# Install the package pip3 install xlrd

df = pd.read_excel('weather_data.xlsx')

df
```

Out[17]:

	day	temperature	wind_speed	event
0	1/1/20217	32	6	Rain
1	1/2/20217	35	7	Sunny
2	1/3/20217	28	2	Snow
3	1/4/20217	24	7	Snow
4	1/5/20217	32	4	Rain
5	1/6/20217	31	2	Sunny

In [18]:
```
#Store data in csv_file and excel_file

student_data = [
    ('101', 'Rahat', 'CSE', 2.90),
    ('102', 'Dider', 'CSE', 2.75),
    ('103', 'SAM',   'ECE', 3.60),
    ('104', 'Aziz',  'CSE', 3.23),
    ('105', 'AB',    'ECE', 4.00),
    ('106', 'Munna', 'CSE', 3.90)
]

df = pd.DataFrame(student_data, columns=['id', 'name', 'dept', 'result'])
```

In [19]:
```
#save the to csv file
df.to_csv('student_document.csv', index=False) #if we put index=True or leave the
parameter empty the file contains the auto ids
```

In [20]:
```
#installation for excel write pip3 install openpyxl
#save the to excel file
df.to_excel('student_document.xlsx', sheet_name='basic_info', index=False)
```

In [21]:
```
#GROUP-BY

df = pd.read_csv('weather_data_cities.csv')

df
```

Out[21]:

	day	city	temperature	windspeed	event
0	1/1/2017	new york	32	6	Rain
1	1/2/2017	new york	36	7	Sunny
2	1/3/2017	new york	28	12	Snow
3	1/4/2017	new york	33	7	Sunny
4	1/1/2017	mumbai	90	5	Sunny

```
5   1/2/2017    mumbai          85         12     Fog
6   1/3/2017    mumbai          87         15     Fog
7   1/4/2017    mumbai          92          5    Rain
8   1/1/2017     paris          45         20   Sunny
9   1/2/2017     paris          50         13  Cloudy
10  1/3/2017     paris          54          8  Cloudy
11  1/4/2017     paris          42         10  Cloudy
```

In [22]:
```
#now we can group the data using their column value

g = df.groupby('city')

for city, city_df in g:
    print('City Name: ' + city)
    print('Weather Information')
    print(city_df)
    print()
```

```
City Name: mumbai
Weather Information
        day    city  temperature  windspeed  event
4  1/1/2017  mumbai           90          5  Sunny
5  1/2/2017  mumbai           85         12    Fog
6  1/3/2017  mumbai           87         15    Fog
7  1/4/2017  mumbai           92          5   Rain

City Name: new york
Weather Information
        day      city  temperature  windspeed  event
0  1/1/2017  new york           32          6   Rain
1  1/2/2017  new york           36          7  Sunny
2  1/3/2017  new york           28         12   Snow
3  1/4/2017  new york           33          7  Sunny

City Name: paris
Weather Information
         day   city  temperature  windspeed   event
8   1/1/2017  paris           45         20   Sunny
9   1/2/2017  paris           50         13  Cloudy
10  1/3/2017  paris           54          8  Cloudy
11  1/4/2017  paris           42         10  Cloudy
```

In [23]:
```
#All rows of a specific group
g.get_group('paris')
```

Out[23]:
```
         day   city  temperature  windspeed   event
8   1/1/2017  paris           45         20   Sunny
```

```
9   1/2/2017  paris           50         13  Cloudy
10  1/3/2017  paris           54          8  Cloudy
11  1/4/2017  paris           42         10  Cloudy
```

In [24]:
```
#Find the maximum temperature and windSpeed of each city

g.max()
```

Out[24]:
```
               day  temperature  windspeed  event
city
mumbai    1/4/2017           92         15  Sunny
new york  1/4/2017           36         12  Sunny
paris     1/4/2017           54         20  Sunny
```

In [25]:
```
#Find the average temp. and windspeed of each city

g.mean()
```

Out[25]:
```
          temperature  windspeed
city
mumbai          88.50       9.25
new york        32.25       8.00
paris           47.75      12.75
```

In [26]:
```
#Find important data of each city

g.describe()
```

Out[26]:
```
         temperature                                                    \
               count   mean       std   min    25%   50%    75%   max
city
mumbai           4.0  88.50  3.109126  85.0  86.50  88.5  90.50  92.0
new york         4.0  32.25  3.304038  28.0  31.00  32.5  33.75  36.0
paris            4.0  47.75  5.315073  42.0  44.25  47.5  51.00  54.0

         windspeed
             count   mean       std  min   25%   50%    75%   max
city
mumbai         4.0   9.25  5.057997  5.0  5.00   8.5  12.75  15.0
new york       4.0   8.00  2.708013  6.0  6.75   7.0   8.25  12.0
paris          4.0  12.75  5.251984  8.0  9.50  11.5  14.75  20.0
```

In [27]:
```
#Creating Data Frame using Hash-Tables [Dictionary]

bd_wd = pd.DataFrame({
    "city" : ["Dhaka", "Sylhet", "Rajshahi"],
    "temp" : [40, 32, 44],
    "humidity" : [80, 75, 65]
})

bd_wd
```

Out[27]:

	city	temp	humidity
0	Dhaka	40	80
1	Sylhet	32	75
2	Rajshahi	44	65

In [28]:
```
us_wd = pd.DataFrame({
    "city" : ["New York", "Chicago", "Michigan"],
    "temp" : [21, 14, 35],
    "humidity" : [68, 65, 75]
})

us_wd
```

Out[28]:

	city	temp	humidity
0	New York	21	68
1	Chicago	14	65
2	Michigan	35	75

In [29]:
```
#Concate two data frame

df = pd.concat([bd_wd, us_wd])

df
```

Out[29]:

	city	temp	humidity
0	Dhaka	40	80
1	Sylhet	32	75
2	Rajshahi	44	65
0	New York	21	68
1	Chicago	14	65
2	Michigan	35	75

In [30]:
```
#Concatenate two data frame with ignoring the index will create a data frame with
with completely new index

df = pd.concat([bd_wd, us_wd], ignore_index=True)

df
```

Out[30]:

	city	temp	humidity
0	Dhaka	40	80
1	Sylhet	32	75
2	Rajshahi	44	65
3	New York	21	68
4	Chicago	14	65
5	Michigan	35	75

In [31]:
```
#Merge two data frame in a same row

df = pd.concat([bd_wd, us_wd], axis=1)
df
```

Out[31]:

	city	temp	humidity	city	temp	humidity
0	Dhaka	40	80	New York	21	68
1	Sylhet	32	75	Chicago	14	65
2	Rajshahi	44	65	Michigan	35	75

In [32]:
```
#Joining two data frame
```

In [33]:
```
#data frame that contains onlu temperature data

temp_df = pd.DataFrame({
    "city" : ["Dhaka", "Sylhet", "Rajshahi", "Chittagong"],
    "temp" : [40, 32, 44, 38],
})

temp_df
```

Out[33]:

	city	temp
0	Dhaka	40
1	Sylhet	32
2	Rajshahi	44
3	Chittagong	38

In [34]:
```
#data frame that contains only humidity data

hum_df = pd.DataFrame({
    "city" : ["Dhaka", "Sylhet", "Chittagong"],
    "humidity" : [80, 70, 85],
})

hum_df
```

Out[34]:

	city	humidity
0	Dhaka	80
1	Sylhet	70
2	Chittagong	85

In [35]:
```
#Join two df using one common column city

df = pd.merge(temp_df, hum_df, on='city')

df
```

Out[35]:

	city	temp	humidity
0	Dhaka	40	80
1	Sylhet	32	70
2	Chittagong	38	85

In [36]:
```
#In the previous example Rajshahi is missing as it has no value
of humidity. To get is we have to join them as outer.
```

```
df = pd.merge(temp_df, hum_df, on='city', how='outer')

df
```

Out[36]:

	city	temp	humidity
0	Dhaka	40	80.0
1	Sylhet	32	70.0
2	Rajshahi	44	NaN
3	Chittagong	38	85.0

```
In [37]: #Indexing: Pandas always take the indices from 0 to (n-1). But we can override
the indices by our own value

         us_wd = pd.DataFrame({
             "city" : ["New York", "Chicago", "Michigan"],
             "temp" : [21, 14, 35],
             "humidity" : [68, 65, 75]
         }, index = [4, 2, 9])

         us_wd
```

Out[37]:

	city	temp	humidity
4	New York	21	68
2	Chicago	14	65
9	Michigan	35	75

```
In [38]: #In pandas we can get any data from their index or their row number. If we use custom
indexing the tow number may be not equal to the index number, otherwise both will be same

         #Using Index number
         us_wd.loc[9]
```

```
Out[38]: city        Michigan
         temp              35
         humidity          75
         Name: 9, dtype: object
```

```
In [39]: #Using row number
         us_wd.iloc[2]
```

```
Out[39]: city        Michigan
         temp              35
         humidity          75
         Name: 9, dtype: object
```