Lecture 7 – Pandas

Mr. Yousif G. Arshak

University of Zakho

Computer Science Department

yousif.arshak@uoz.edu.krd

OUTLINES

- Pandas
  - Series
  - DataFrames
  - GroupBy
  - Data Input and output

Series

- Series is a one-dimensional labeled array capable of holding data of any type (integer, string, float, python objects, etc.). The axis labels are collectively called index.
- pandas.Series(data, index, dtype, copy)

1	**data** data takes various forms like ndarray, list, constants
2	**index** Index values must be unique and hashable, same length as data. Default **np.arrange(n)** if no index is passed.
3	**dtype** dtype is for data type. If None, data type will be inferred
4	**copy** Copy data. Default False

Create a Series from ndarray with default index

```
import pandas as pd
import numpy as np
data = np.array(['a','b','c','d'])
s = pd.Series(data)
print s
```

```
Output:
0 a
1 b
2 c
3 d
dtype: object
```

Series with specific indexes

```
import pandas as pd
import numpy as np
data = np.array(['a','b','c','d’])
s = pd.Series(data,index=[100,101,102,103])
print s
```

```
Output
100 a
101 b
102 c
103 d
dtype: object
```

DataFrame

- A Data frame is a two-dimensional data structure, i.e., data is aligned in a tabular fashion in rows and columns.
- `pandas.DataFrame( data, index, columns, dtype, copy)`

Sr.No	Parameter & Description
1	**data** data takes various forms like ndarray, series, map, lists, dict, constants and also another DataFrame.
2	**index** For the row labels, the Index to be used for the resulting frame is Optional Default np.arange(n) if no index is passed.
3	**columns** For column labels, the optional default syntax is - np.arange(n). This is only true if no index is passed.
4	**dtype** Data type of each column.
5	**copy** This command (or whatever it is) is used for copying of data, if the default is False.

example

```
import pandas as pd
data = [['Alex',10],['Bob',12],['Clarke',13]]
df = pd.DataFrame(data,columns=['Name','Age’])
print df
```

```
Output
   Name Age
0 Alex 10
1 Bob 12
2 Clarke 13
```

Example

```
import pandas as pd
import numpy as np
from numpy.random import randn
np.random.seed(101)   # to get the same random numbers
df = pd.DataFrame(randn(5,4),['A','B','C','D','E'], ['W','X','Y','Z'])
print(df)
```

```
          W         X         Y         Z
A  2.706850  0.628133  0.907969  0.503826
B  0.651118 -0.319318 -0.848077  0.605965
C -2.018168  0.740122  0.528813 -0.589001
D  0.188695 -0.758872 -0.933237  0.955057
E  0.190794  1.978757  2.605967  0.683509
```

To access specific Column

- df = pd.DataFrame(randn(5,4),['A','B','C','D','E'], ['W','X','Y','Z'])
- df[‘W’] #Return Column W
- df[‘W’,’Z’] #Return Column W and Z
- df.loc['A’] #Return row A
- df.iloc[2] #Return row index 2
- df.loc[‘A’,’W’] #Return item in row A, Column W
- df.iloc[[0,1,2],[0,1,3]]

	W	X	Z
A	2.706850	0.628133	0.503826
B	0.651118	-0.319318	0.605965
C	-2.018168	0.740122	-0.589001

Filter with True or False

Df > 0

Output:	**W**	**X**	**Y**	**Z**
A	True	True	True	True
B	True	False	False	True
C	False	True	True	False
D	True	False	False	True
E	True	True	True	True

GroupBy

- Any **groupby** operation involves one of the following operations on the original object. They are −
- **Splitting** the Object
- **Applying** a function
- **Combining** the results

Example

```
import pandas as pd
ipl_data = {'Team': ['Riders', 'Riders', 'Devils', 'Devils', 'Kings',        'kings', 'Kings',
'Kings', 'Riders', 'Royals', 'Royals', 'Riders'],        'Rank': [1, 2, 2, 3, 3,4 ,1 ,1,2 ,
4,1,2], 'Year’:
[2014,2015,2014,2015,2014,2015,2016,2017,2016,2014,2015,2017],
'Points':[876,789,863,673,741,812,756,788,694,701,804,690]}df =
pd.DataFrame(ipl_data)
print df.groupby('Team').groups
```

Data Input and output

```
import pandas as pd
a = pd.read_csv('Salaries.csv’)
b = a.to_csv('olympics2.csv’)
a.to_csv('olympics2.csv',index=false)
```