**Data Science and Data Visualization in Python using numpy, pandas and matplotlib**

**Raman Deep Singh**

Data Science means manipulating large set of data in python using python modules like numpy and pandas.

Numpy is a python library which include ndarray , a two dimensional data structure to store and manipulate large sets of data.

Using numpy we can perform operations on ndarray like mathematical operations like add subtract, multiply and many more complex operations like mean etc.

Following are the commands to install numpy, pandas and matplotlib

pip install numpy

pip install pandas

pip install matplotlib

---

Few examples of numpy programs

create a numpy array from a list

```
import numpy as np
List=[1,2,3,4]
a1=np.array(List)
print(a1)
```

---

access individual elements from array

```
import numpy as np
List=[1,2,3,4]
a1=np.array(List)
print(a1[0])
```

---

another example of accessing elements from numpy array

```
import numpy as np
a7=np.array([[10,11,12,13],[21,22,23,24]])
print(a7[1,3])
print(a7[1][3])
```

---

type of ndarray, shape and itemsize of ndarray

```
import numpy as np
a1=np.array([1,2,3,4])
a2=np.array([[2,4,6],[1,3,5]])
print(type(a1),type(a2))
print(a1.shape)
print(a2.shape)
print(a2.itemsize)
```

---

adding number to each element of ndarray

```
import numpy as np
a1=np.array([1,2,3,4])
a1=a1+2
print(a1)
```

---

create ndarray from tuple

```
import numpy as np
a1=np.array((1,2,3,4))
print(a1)
```

---

create ndarray from dictionary

```
import numpy as np
dict1={1:'A',2:'B',3:'C'}
a1=np.array(dict1)
print(a1)
```

---

create ndarray using fromiter function

```
import numpy as np
dict1={1:'A',2:'B',3:'C'}
```

```
a1=np.fromiter(dict1,dtype=np.int32)
print(a1[0])
```

---

create ndarray from string using fromiter function

```
import numpy as np
str="HelloWorld"
a1=np.fromiter(str,dtype="U2")
print(a1)
```

---

create ndarray with a numerical range using arange()

```
import numpy as np
ar5=np.arange(7)
print(ar5)
```

---

create array with a numerical range using linspace()

```
import numpy as np
a1=np.linspace(2,10,5)
print(a1)
```

---

create 2d ndarray using array()

```
import numpy as np
List1=[[1,3,5],[2,4,6]]
nd2=np.array(List1)
print(nd2)
print(nd2.shape)
```

---

create 2d ndarray using arrange and reshape the ndarray

```
import numpy as np
ar=np.arange(10)
print(ar)
ar1=ar.reshape(5,2)
```

```
print(ar1)
ar2=ar.reshape(2,5)
print(ar2)
```

---

create ndarray and reshape in single statement

```
import numpy as np
ar=np.arange(10).reshape(5,2)
print(ar)
```

---

create empty array using empty() function array will contain garbage values

```
import numpy as np
ar=np.empty(3)
print(ar)
```

---

create ndarray containing zero values using zeros function

```
import numpy as np
ar=np.zeros(3)
print(ar)
```

---

crate ndarray containing one using ones function

```
import numpy as np
ar=np.ones(3)
print(ar)
```

---

access 3rd element from ndarray

```
import numpy as np
ar=np.array([3,4,5,7,8])
print(ar[2])
```

---

access individual elements in 2d array

```
import numpy as np
```

```
ar=np.array([[3,4,5,7,8],[1,2,3,4,5]])
print(ar[1][2])
```

---

array slices

```
import numpy as np
ar=np.array([3,4,5,7,8,1,2,3,4,5])
print(ar[2:5])
```

---

array slice step=2

```
import numpy as np
ar=np.array([3,4,5,7,8,1,2,3,4,5])
print(ar[2:5:2])
```

---

combine two ndarray horizontally using hstack function

```
import numpy as np
ar=np.array([1,2,3])
ar1=np.array([4,5,6])
ar2=np.hstack((ar,ar1))
print(ar2)
```

---

combine two ndarray vertically using vstack function

```
import numpy as np
ar=np.array([1,2,3])
ar1=np.array([4,5,6])
ar2=np.vstack((ar,ar1))
print(ar2)
```

---

concatenate two ndarray using concatenate function

```
import numpy as np
ar=np.array([1,2,3])
ar1=np.array([4,5,6])
```

```
ar2=np.concatenate((ar,ar1))
print(ar2)
```

---

hsplit function

```
import numpy as np
ar=np.arange(24.0).reshape(4,6)
print(ar)
ar1=np.hsplit(ar,2)
print(ar1)
```

---

vsplit function

```
import numpy as np
ar=np.arange(24.0).reshape(4,6)
print(ar)
ar1=np.vsplit(ar,2)
print(ar1)
```

---

split function

```
import numpy as np
ar=[10,11,12,13,14,15,16,17,18,19]
ar1=np.split(ar,[2,6])
print(ar1)
```

---

extract elements using a condition

```
import numpy as np
ar=[10,11,12,13,14,15,16,17,18,19]
cond1=np.mod(ar,5)==0
ar1=np.extract(cond1,ar)
print(ar1)
```

---

add two ndarrays using + operator

```
import numpy as np
ar1=np.array([10,11,12,13,14])
ar2=np.array([15,16,17,18,19])
ar3=ar1+ar2
print(ar3)
```

---

-,*,/,% operators

numpy.add

numpy.subtract

multiply

divide

mod

remainder

---

Pandas is a python library which stores , manipulates and analyses data in dataframes and series.

---

Few examples of programs in Dataframe and Series.

creating empty dataframe

```
#import the pandas library and aliasing as pd
import pandas as pd
df = pd.DataFrame()
print df
```

---

create dataframe from list

```
import pandas as pd
data = [1,2,3,4,5]
df = pd.DataFrame(data)
print df
```

---

create dataframe from list example 2

```
import pandas as pd
data = [['Alex',10],['Bob',12],['Clarke',13]]
```

```
df = pd.DataFrame(data,columns=['Name','Age'])
print df
```

---

example 3

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

---

Create a DataFrame from Dict of ndarrays / Lists

```
import pandas as pd
data = {'Name':['Tom', 'Jack', 'Steve', 'Ricky'],'Age':[28,34,29,42]}
df = pd.DataFrame(data)
print df
```

---

example 2

```
import pandas as pd
data = {'Name':['Tom', 'Jack', 'Steve', 'Ricky'],'Age':[28,34,29,42]}
df = pd.DataFrame(data, index=['rank1','rank2','rank3','rank4'])
print df
```

---

Create a DataFrame from List of Dicts

example 1

```
import pandas as pd
data = [{'a': 1, 'b': 2},{'a': 5, 'b': 10, 'c': 20}]
df = pd.DataFrame(data)
print df
```

---

example 2

```
import pandas as pd
data = [{'a': 1, 'b': 2},{'a': 5, 'b': 10, 'c': 20}]
```

```
df = pd.DataFrame(data, index=['first', 'second'])
print df
```

---

example 3

```
import pandas as pd
data = [{'a': 1, 'b': 2},{'a': 5, 'b': 10, 'c': 20}]
#With two column indices, values same as dictionary keys
df1 = pd.DataFrame(data, index=['first', 'second'], columns=['a', 'b'])
#With two column indices with one index with other name
df2 = pd.DataFrame(data, index=['first', 'second'], columns=['a', 'b1'])
print df1
print df2
```

---

Create a DataFrame from Dict of Series

```
import pandas as pd
d = {'one' : pd.Series([1, 2, 3], index=['a', 'b', 'c']),
   'two' : pd.Series([1, 2, 3, 4], index=['a', 'b', 'c', 'd'])}
df = pd.DataFrame(d)
print df
```

---

Column Selection

```
import pandas as pd
d = {'one' : pd.Series([1, 2, 3], index=['a', 'b', 'c']),
   'two' : pd.Series([1, 2, 3, 4], index=['a', 'b', 'c', 'd'])}
df = pd.DataFrame(d)
print df ['one']
```

---

adding a new column

```
import pandas as pd
d = {'one' : pd.Series([1, 2, 3], index=['a', 'b', 'c']),
   'two' : pd.Series([1, 2, 3, 4], index=['a', 'b', 'c', 'd'])}
```

```
df = pd.DataFrame(d)


# Adding a new column to an existing DataFrame object with column label by passing new series

print ("Adding a new column by passing as Series:")

df['three']=pd.Series([10,20,30],index=['a','b','c'])

print df

print ("Adding a new column using the existing columns in DataFrame:")

df['four']=df['one']+df['three']

print df
```

---

Column Deletion

```
# Using the previous DataFrame, we will delete a column

# using del function

import pandas as pd

d = {'one' : pd.Series([1, 2, 3], index=['a', 'b', 'c']),

  'two' : pd.Series([1, 2, 3, 4], index=['a', 'b', 'c', 'd']),

  'three' : pd.Series([10,20,30], index=['a','b','c'])}

df = pd.DataFrame(d)

print ("Our dataframe is:")

print df

# using del function

print ("Deleting the first column using DEL function:")

del df['one']

print df

# using pop function

print ("Deleting another column using POP function:")

df.pop('two')

print df
```

---

Row Selection, Addition and Deletion

Selection by label

```
import pandas as pd
d = {'one' : pd.Series([1, 2, 3], index=['a', 'b', 'c']),
  'two' : pd.Series([1, 2, 3, 4], index=['a', 'b', 'c', 'd'])}
df = pd.DataFrame(d)
print df.loc['b']
```

---

Selection by integer location

```
import pandas as pd
d = {'one' : pd.Series([1, 2, 3], index=['a', 'b', 'c']),
  'two' : pd.Series([1, 2, 3, 4], index=['a', 'b', 'c', 'd'])}

df = pd.DataFrame(d)
print df.iloc[2]
```

---

Slice Rows

```
import pandas as pd
d = {'one' : pd.Series([1, 2, 3], index=['a', 'b', 'c']),
  'two' : pd.Series([1, 2, 3, 4], index=['a', 'b', 'c', 'd'])}

df = pd.DataFrame(d)
print df[2:4]
```

---

Addition of Rows

```
import pandas as pd
df = pd.DataFrame([[1, 2], [3, 4]], columns = ['a','b'])
df2 = pd.DataFrame([[5, 6], [7, 8]], columns = ['a','b'])
df = df.append(df2)
print df
```

---

Deletion of Rows

```
import pandas as pd

df = pd.DataFrame([[1, 2], [3, 4]], columns = ['a','b'])

df2 = pd.DataFrame([[5, 6], [7, 8]], columns = ['a','b'])

df = df.append(df2)

# Drop rows with label 0

df = df.drop(0)

print df
```

---

iteritems()

```
import pandas as pd

import numpy as np

dc={2015:{'Qtr1':34500,'Qtr2':56000,'Qtr3':47000,'Qtr4':49000},2016:{'Qtr1':44900,'Qtr2':46100,'
Qtr3':57000,'Qtr4':59000},2017:{'Qtr1':54500,'Qtr2':51000,'Qtr3':57000,'Qtr4':58500}}

df=pd.DataFrame(dc)

for key,value in df.iteritems():

  print(key,value)
```

---

iterrows

```
import pandas as pd

import numpy as np

dc={2015:{'Qtr1':34500,'Qtr2':56000,'Qtr3':47000,'Qtr4':49000},2016:{'Qtr1':44900,'Qtr2':46100,'
Qtr3':57000,'Qtr4':59000},2017:{'Qtr1':54500,'Qtr2':51000,'Qtr3':57000,'Qtr4':58500}}

df=pd.DataFrame(dc)

for key,value in df.iterrows():

  print(key,value)
```

---

loc function to get a value from dataframe

```
# importing pandas as pd

import pandas as pd

index_ = ['Row_1', 'Row_2', 'Row_3', 'Row_4', 'Row_5']

# Creating the DataFrame

df = pd.DataFrame({'Weight':[45, 88, 56, 15, 71],
```

```
            'Name':['Sam', 'Andrea', 'Alex', 'Robin', 'Kia'],
            'Age':[14, 25, 55, 8, 21]},index=index_)
print(df)
# return the value
result = df.loc['Row_2', 'Name']
# Print the result
print(result)
```

---

loc function to access multiple rows

```
# importing pandas as pd
import pandas as pd
index_ = ['Row_1', 'Row_2', 'Row_3', 'Row_4', 'Row_5']
# Creating the DataFrame
df = pd.DataFrame({'Weight':[45, 88, 56, 15, 71],
            'Name':['Sam', 'Andrea', 'Alex', 'Robin', 'Kia'],
            'Age':[14, 25, 55, 8, 21]},index=index_)
print(df)
# return the value
result = df.loc['Row_2':'Row_4',:]
# Print the result
print(result)
```

---

loc function to access selective columns

```
# importing pandas as pd
import pandas as pd
index_ = ['Row_1', 'Row_2', 'Row_3', 'Row_4', 'Row_5']
# Creating the DataFrame
df = pd.DataFrame({'Weight':[45, 88, 56, 15, 71],
            'Name':['Sam', 'Andrea', 'Alex', 'Robin', 'Kia'],
            'Age':[14, 25, 55, 8, 21]},index=index_)
print(df)
```

```
# return the value
result = df.loc[:,'Name':'Age',]
# Print the result
print(result)
```

---

loc function to access range of columns from a range of rows

```
# importing pandas as pd
import pandas as pd
index_ = ['Row_1', 'Row_2', 'Row_3', 'Row_4', 'Row_5']
# Creating the DataFrame
df = pd.DataFrame({'Weight':[45, 88, 56, 15, 71],
                   'Name':['Sam', 'Andrea', 'Alex', 'Robin', 'Kia'],
                   'Age':[14, 25, 55, 8, 21]},index=index_)
print(df)
# return the value
result = df.loc["Row_1":"Row_3",'Name':'Age',]
# Print the result
print(result)
```

---

iloc function to access rows and columns from dataframe

```
# importing pandas as pd
import pandas as pd
index_ = ['Row_1', 'Row_2', 'Row_3', 'Row_4', 'Row_5']
# Creating the DataFrame
df = pd.DataFrame({'Weight':[45, 88, 56, 15, 71],
                   'Name':['Sam', 'Andrea', 'Alex', 'Robin', 'Kia'],
                   'Age':[14, 25, 55, 8, 21]},index=index_)
print(df)
# return the value
result = df.iloc[0:2,1:3]
# Print the result
```

```
print(result)
```

---

select or accessing individual value using column name

```
# importing pandas as pd
import pandas as pd
index_ = ['Row_1', 'Row_2', 'Row_3', 'Row_4', 'Row_5']
# Creating the DataFrame
df = pd.DataFrame({'Weight':[45, 88, 56, 15, 71],
                   'Name':['Sam', 'Andrea', 'Alex', 'Robin', 'Kia'],
                   'Age':[14, 25, 55, 8, 21]},index=index_)
print(df)
# return the value
result = df.Weight[0]
# Print the result
print(result)
```

---

access individual element using index and column name

```
# importing pandas as pd
import pandas as pd
index_ = ['Row_1', 'Row_2', 'Row_3', 'Row_4', 'Row_5']
# Creating the DataFrame
df = pd.DataFrame({'Weight':[45, 88, 56, 15, 71],
                   'Name':['Sam', 'Andrea', 'Alex', 'Robin', 'Kia'],
                   'Age':[14, 25, 55, 8, 21]},index=index_)
print(df)
# return the value
result = df.Weight['Row_1']
# Print the result
print(result)
```

---

modify column value in a dataframe

```
# importing pandas as pd
import pandas as pd
index_ = ['Row_1', 'Row_2', 'Row_3', 'Row_4', 'Row_5']
# Creating the DataFrame
df = pd.DataFrame({'Weight':[45, 88, 56, 15, 71],
                   'Name':['Sam', 'Andrea', 'Alex', 'Robin', 'Kia'],
                   'Age':[14, 25, 55, 8, 21]},index=index_)
print(df)
# return the value
df.Weight['Row_1']=90
print(df)
```

---

modify row in a dataframe using at function

```
# importing pandas as pd
import pandas as pd
index_ = ['Row_1', 'Row_2', 'Row_3', 'Row_4', 'Row_5']
# Creating the DataFrame
df = pd.DataFrame({'Weight':[45, 88, 56, 15, 71],
                   'Name':['Sam', 'Andrea', 'Alex', 'Robin', 'Kia'],
                   'Age':[14, 25, 55, 8, 21]},index=index_)
print(df)
# return the value
df.at['Row_1']=99
print(df)
```

---

modify row in a dataframe using loc function

```
# importing pandas as pd
import pandas as pd
index_ = ['Row_1', 'Row_2', 'Row_3', 'Row_4', 'Row_5']
# Creating the DataFrame
df = pd.DataFrame({'Weight':[45, 88, 56, 15, 71],
```

```
        'Name':['Sam', 'Andrea', 'Alex', 'Robin', 'Kia'],
        'Age':[14, 25, 55, 8, 21]},index=index_)
print(df)
# return the value
df.loc['Row_1']=99
print(df)
```

---

adding column to a dataframe

```
# importing pandas as pd
import pandas as pd
index_ = ['Row_1', 'Row_2', 'Row_3', 'Row_4', 'Row_5']
# Creating the DataFrame
df = pd.DataFrame({'Weight':[45, 88, 56, 15, 71],
        'Name':['Sam', 'Andrea', 'Alex', 'Robin', 'Kia'],
        'Age':[14, 25, 55, 8, 21]},index=index_)
print(df)
# return the value
df['Class']=[12,11,10,9,9]
print(df)
```

---

delete a column from a dataframe using del statement

```
# importing pandas as pd
import pandas as pd
index_ = ['Row_1', 'Row_2', 'Row_3', 'Row_4', 'Row_5']
# Creating the DataFrame
df = pd.DataFrame({'Weight':[45, 88, 56, 15, 71],
        'Name':['Sam', 'Andrea', 'Alex', 'Robin', 'Kia'],
        'Age':[14, 25, 55, 8, 21]},index=index_)
print(df)
# return the value
del df["Weight"]
```

```
print(df)
```

---

min function to get minimum values from a dataframe

```
# importing pandas as pd
import pandas as pd
index_ = ['Row_1', 'Row_2', 'Row_3', 'Row_4', 'Row_5']
# Creating the DataFrame
df = pd.DataFrame({'Weight':[45, 88, 56, 15, 71],
        'Name':['Sam', 'Andrea', 'Alex', 'Robin', 'Kia'],
        'Age':[14, 25, 55, 8, 21]},index=index_)
print(df)
print(df.min())
```

---

max function to get maximum values in a dataframe

```
# importing pandas as pd
import pandas as pd
index_ = ['Row_1', 'Row_2', 'Row_3', 'Row_4', 'Row_5']
# Creating the DataFrame
df = pd.DataFrame({'Weight':[45, 88, 56, 15, 71],
        'Name':['Sam', 'Andrea', 'Alex', 'Robin', 'Kia'],
        'Age':[14, 25, 55, 8, 21]},index=index_)
print(df)
print(df.max())
```

---

mode function to sort the records

```
# importing pandas as pd
import pandas as pd
index_ = ['Row_1', 'Row_2', 'Row_3', 'Row_4', 'Row_5']
# Creating the DataFrame
df = pd.DataFrame({'Weight':[45, 88, 56, 15, 71],
        'Name':['Sam', 'Andrea', 'Alex', 'Robin', 'Kia'],
```

```
        'Age':[14, 25, 55, 8, 21]},index=index_)
print(df)
print(df.mode())
```

---

mean function to find mean of values

```
# importing pandas as pd
import pandas as pd
index_ = ['Row_1', 'Row_2', 'Row_3', 'Row_4', 'Row_5']
# Creating the DataFrame
df = pd.DataFrame({'Weight':[45, 88, 56, 15, 71],
        'Name':['Sam', 'Andrea', 'Alex', 'Robin', 'Kia'],
        'Age':[14, 25, 55, 8, 21]},index=index_)
print(df)
print(df.mean())
```

---

median function to get median value

```
# importing pandas as pd
import pandas as pd
index_ = ['Row_1', 'Row_2', 'Row_3', 'Row_4', 'Row_5']
# Creating the DataFrame
df = pd.DataFrame({'Weight':[45, 88, 56, 15, 71],
        'Name':['Sam', 'Andrea', 'Alex', 'Robin', 'Kia'],
        'Age':[14, 25, 55, 8, 21]},index=index_)
print(df)
print(df.median())
```

---

count function

```
# importing pandas as pd
import pandas as pd
index_ = ['Row_1', 'Row_2', 'Row_3', 'Row_4', 'Row_5']
# Creating the DataFrame
```

```
df = pd.DataFrame({'Weight':[45, 88, 56, 15, 71],
                   'Name':['Sam', 'Andrea', 'Alex', 'Robin', 'Kia'],
                   'Age':[14, 25, 55, 8, 21]},index=index_)
print(df)
print(df.count())
```

---

sum function

```
# importing pandas as pd
import pandas as pd
index_ = ['Row_1', 'Row_2', 'Row_3', 'Row_4', 'Row_5']
# Creating the DataFrame
df = pd.DataFrame({'Weight':[45, 88, 56, 15, 71],
                   'Name':['Sam', 'Andrea', 'Alex', 'Robin', 'Kia'],
                   'Age':[14, 25, 55, 8, 21]},index=index_)
print(df)
print(df.sum())
```

---

apply a function on a column

```
# importing pandas as pd
import pandas as pd
index_ = ['Row_1', 'Row_2', 'Row_3', 'Row_4', 'Row_5']
# Creating the DataFrame
df = pd.DataFrame({'Weight':[45, 88, 56, 15, 71],
                   'Name':['Sam', 'Andrea', 'Alex', 'Robin', 'Kia'],
                   'Age':[14, 25, 55, 8, 21]},index=index_)
print(df)
print(df['Weight'].min())
```

---

apply functions on multiple columns

```
# importing pandas as pd
import pandas as pd
```

```
index_ = ['Row_1', 'Row_2', 'Row_3', 'Row_4', 'Row_5']
# Creating the DataFrame
df = pd.DataFrame({'Weight':[45, 88, 56, 15, 71],
        'Name':['Sam', 'Andrea', 'Alex', 'Robin', 'Kia'],
        'Age':[14, 25, 55, 8, 21]},index=index_)
print(df)
print(df[['Weight','Age']].min())
```

---

example 2 apply function on multiple columns

```
# importing pandas as pd
import pandas as pd
index_ = ['Row_1', 'Row_2', 'Row_3', 'Row_4', 'Row_5']
# Creating the DataFrame
df = pd.DataFrame({'Weight':[45, 88, 56, 15, 71],
        'Name':['Sam', 'Andrea', 'Alex', 'Robin', 'Kia'],
        'Age':[14, 25, 55, 8, 21]},index=index_)
print(df)
print(df[['Weight','Age']].count())
```

---

apply functions on a row of a dataframe

```
# importing pandas as pd
import pandas as pd
index_ = ['Row_1', 'Row_2', 'Row_3', 'Row_4', 'Row_5']
# Creating the DataFrame
df = pd.DataFrame({'Weight':[45, 88, 56, 15, 71],
        'Name':['Sam', 'Andrea', 'Alex', 'Robin', 'Kia'],
        'Age':[14, 25, 55, 8, 21]},index=index_)
print(df)
print(df.loc['Row_1':].max())
```

---

apply function on a range of rows of a dataframe

```
# importing pandas as pd
import pandas as pd
index_ = ['Row_1', 'Row_2', 'Row_3', 'Row_4', 'Row_5']
# Creating the DataFrame
df = pd.DataFrame({'Weight':[45, 88, 56, 15, 71],
                   'Name':['Sam', 'Andrea', 'Alex', 'Robin', 'Kia'],
                   'Age':[14, 25, 55, 8, 21]},index=index_)
print(df)
print(df.loc['Row_1':'Row_3'].count())
```

---

apply function to subset of dataframe

```
# importing pandas as pd
import pandas as pd
index_ = ['Row_1', 'Row_2', 'Row_3', 'Row_4', 'Row_5']
# Creating the DataFrame
df = pd.DataFrame({'Weight':[45, 88, 56, 15, 71],
                   'Name':['Sam', 'Andrea', 'Alex', 'Robin', 'Kia'],
                   'Age':[14, 25, 55, 8, 21]},index=index_)
print(df)
print(df.loc['Row_1':'Row_2','Weight':'Age'].max())
```

---

pivot function

```
# importing pandas as pd
import pandas as pd
d1={'Tutor':['Tahira','Gurjyot','Anusha','Jacob','Venkat'],
'Classes':[28,36,41,32,40],
'Country':['USA','UK','Japan','USA','Brazil']
}
dfd=pd.DataFrame(d1)
print(dfd)
dfd2=dfd.pivot(index='Country',columns='Tutor',values='Classes')
```

```
print(dfd2)
```

---

sort_values function

```
# importing pandas as pd
import pandas as pd
d1={'Tutor':['Tahira','Gurjyot','Anusha','Jacob','Venkat'],
'Classes':[28,36,41,32,40],
'Country':['USA','UK','Japan','USA','Brazil']
}
dfd=pd.DataFrame(d1)
print(dfd)
dfd2=dfd.sort_values('Country')
print(dfd2)
```

---

create histogram

```
# importing pandas as pd
import pandas as pd
index_ = ['Row_1', 'Row_2', 'Row_3', 'Row_4', 'Row_5']
# Creating the DataFrame
df = pd.DataFrame({'Weight':[45, 88, 56, 15, 71],
            'Name':['Sam', 'Andrea', 'Alex', 'Robin', 'Kia'],
            'Age':[14, 25, 55, 8, 21]},index=index_)
print(df)
import matplotlib.pyplot as plt
plt.interactive(True)
plt.show(df.hist(column='Age'))
```

---

pipe function

```
import pandas as pd
import numpy as np
index_ = ['Row_1', 'Row_2', 'Row_3', 'Row_4', 'Row_5']
```

```
# Creating the DataFrame
df = pd.DataFrame([45, 88, 56, 15, 71])
print(df)
df=df.pipe(np.add,30).pipe(np.multiply,3)
print(df)
```

---

apply function

```
import pandas as pd;
def fun(num):
  if num<200:
    return "Low"
  elif num>= 200 and num<400:
    return "Normal"
  else:
    return "High"
src=pd.Series([1000,20,300,400]);
print(src);
new=src.apply(fun)
print(new)
```

---

applymap

```
import pandas as pd;
src=pd.DataFrame(['abc','xyx','xyz']);
src=src.applymap(lambda x: str(x) + '_X')
print(src)
```

---

group by function

```
import pandas as pd
# Define a dictionary containing employee data
data1 = {'Name':['Jai', 'Anuj', 'Jai', 'Princi',
          'Gaurav', 'Anuj', 'Princi', 'Abhi'],
```

```
    'Age':[27, 24, 22, 32,
           33, 36, 27, 32],
    'Address':['Nagpur', 'Kanpur', 'Allahabad', 'Kannuaj',
               'Jaunpur', 'Kanpur', 'Allahabad', 'Aligarh'],
    'Qualification':['Msc', 'MA', 'MCA', 'Phd',
                     'B.Tech', 'B.com', 'Msc', 'MA']}
# Convert the dictionary into DataFrame
df = pd.DataFrame(data1)

print(df)
df.groupby('Name')
print(df.groupby('Name').groups)
```

---

agg function

```
# importing pandas module
import pandas as pd
# importing numpy module
import numpy as np
# creating random arr of 10 elements
arr=np.random.randn(10)
# creating series from array
series=pd.Series(arr)
# calling .agg() method
result=series.agg(lambda num : num + 2)
# display
print('Array before operation: \n', series,
    '\n\nArray after operation: \n',result)
```

---

transform function

```
import pandas as pd
# Creating the DataFrame
```

```
df = pd.DataFrame({"A":[12, 4, 5, None, 1],
                   "B":[7, 2, 54, 3, None],
                   "C":[20, 16, 11, 3, 8],
                   "D":[14, 3, None, 2, 6]})
# Create the index
index_ = ['Row_1', 'Row_2', 'Row_3', 'Row_4', 'Row_5']
# Set the index
df.index = index_
# Print the DataFrame
print(df)
# add 10 to each element of the dataframe
result = df.transform(func = lambda x : x + 10)
# Print the result
print(result)
```

---

rename function

```
import pandas as pd
d = {'one' : pd.Series([1, 2, 3], index=['a', 'b', 'c']),
  'two' : pd.Series([1, 2, 3, 4], index=['a', 'b', 'c', 'd']),
  'three' : pd.Series([10,20,30], index=['a','b','c'])}
df = pd.DataFrame(d)
print ("Our dataframe is:")
print(df)
df.rename(index = {"a": "A",
                   "b":"B","c":"C","d":"D"},
                         inplace = True)
print(df)
```

---

reindex function

```
# importing pandas as pd
import pandas as pd
```

```
# Creating the dataframe
df = pd.DataFrame({"A":[1, 5, 3, 4, 2],
                   "B":[3, 2, 4, 3, 4],
                   "C":[2, 2, 7, 3, 4],
                   "D":[4, 3, 6, 12, 7]},
                   index =["first", "second", "third", "fourth", "fifth"])
# Print the dataframe
print(df)
df=df.reindex(["first", "dues", "trois", "fourth", "fifth"])
print(df)
```

---

reindex_like function

```
# importing pandas as pd
import pandas as pd
# Creating the first dataframe
df1 = pd.DataFrame({"A":[1, 5, 3, 4, 2],
                    "B":[3, 2, 4, 3, 4],
                    "C":[2, 2, 7, 3, 4],
                    "D":[4, 3, 6, 12, 7]},
                    index =["A1", "A2", "A3", "A4", "A5"])
# Creating the second dataframe
df2 = pd.DataFrame({"A":[10, 11, 7, 8, 5],
                    "B":[21, 5, 32, 4, 6],
                    "C":[11, 21, 23, 7, 9],
                    "D":[1, 5, 3, 8, 6]},
                     index =["A1", "A3", "A4", "A7", "A8"])
# Print the first dataframe
print(df1)
# Print the second dataframe
print(df2)
# find matching indexes
```

```
df1=df1.reindex_like(df2)
print(df1)
```

---

Few examples of programs in Series

numpy arrays

creating a numpy array

```
import numpy as np
lst=[1,2,3,4];
a1=np.array(lst);
print(a1);
```

---

creating series object from a sequence

```
import pandas as pd;
src=pd.Series([1,2,3,4]);
print(src);
```

---

creating series object from ndarray

```
import pandas as pd;
import numpy as np;
nda1=np.arange(3,13,3.5);
print(nda1);
ser1=pd.Series(nda1);
print(ser1);
```

---

creating series object from dictionary

```
import pandas as pd;
ser1=pd.Series({"1":"raman","2":"gagan","3":"aman"});
print(ser1);
```

---

specifying NaN values in a Series object

```
import pandas as pd;
import numpy as np;
```

```
ser1=pd.Series([2,3,np.NaN,4]);
print(ser1);
```

---

creating series with specifying data and indexes

```
import pandas as pd;
import numpy as np;
arr=[31,28,31,30];
mon=["Jan","Feb","Mar","Apr"];
ser1=pd.Series(data=arr,index=mon);
print(ser1);
```

---

creating series object by specifying datatype,data and index

```
import pandas as pd;
import numpy as np;
arr=[31,28,31,30];
mon=["Jan","Feb","Mar","Apr"];
ser1=pd.Series(data=arr,index=mon,dtype=np.float64);
print(ser1);
```

---

creating series obect using a mathematical function or expression

```
import pandas as pd;
import numpy as np;
a=np.arange(9,13);
ob7=pd.Series(index=a,data=a*2);
print(ob7);
```

---

create a series  by adding data to a series

```
import pandas as pd;
lst=[1,2,3,4];
ob7=pd.Series(data=(2*lst));
print(ob7);
```

---

modify a value in series object

```
import pandas as pd;
ob7=pd.Series([1,2,3,4]);
print(ob7);
ob7[2]=12;
print(ob7);
```

---

access individual element in series object

```
import pandas as pd;
ob7=pd.Series([1,2,3,4]);
print(ob7[2]);
```

---

extract slices from series obbject

```
import pandas as pd;
ob7=pd.Series([1,2,3,4]);
print(ob7[2:4]);
```

---

head function of series object

```
import pandas as pd;
ob7=pd.Series([1,2,3,4,5,6,7,8,9,10]);
print(ob7.head());
```

---

tail function of series object

```
import pandas as pd;
ob7=pd.Series([1,2,3,4,5,6,7,8,9,10]);
print(ob7.tail());
```

---

vector operation on series object (+) operation

```
import pandas as pd;
ob7=pd.Series([1,2,3,4,5,6,7,8,9,10]);
```

```
print(ob7+2);
```

---

* operation

```
import pandas as pd;
ob7=pd.Series([1,2,3,4,5,6,7,8,9,10]);
print(ob7*2);
```

---

** operation

```
import pandas as pd;
ob7=pd.Series([1,2,3,4,5,6,7,8,9,10]);
print(ob7**2);
```

---

arithmetic operation add(+) on series objects

```
import pandas as pd;
ob7=pd.Series([1,2,3,4,5,6,7,8,9,10]);
ob8=pd.Series([2,3,4,5,6,7,8,9,10,11]);
ob9=ob7+ob8;
print(ob9);
```

---

- operation on series objects

```
import pandas as pd;
ob7=pd.Series([1,2,3,4,5,6,7,8,9,10]);
ob8=pd.Series([2,3,4,5,6,7,8,9,10,11]);
ob9=ob7-ob8;
print(ob9);
```

---

* operation on series object

```
import pandas as pd;
ob7=pd.Series([1,2,3,4,5,6,7,8,9,10]);
ob8=pd.Series([2,3,4,5,6,7,8,9,10,11]);
ob9=ob7*ob8;
```

```
print(ob9);
```

---

/ operation on series object

```
import pandas as pd;
ob7=pd.Series([1,2,3,4,5,6,7,8,9,10]);
ob8=pd.Series([2,3,4,5,6,7,8,9,10,11]);
ob9=ob7/ob8;
print(ob9);
```

---

filtering entries

```
import pandas as pd;
ob7=pd.Series([1,2,3,4,5,6,7,8,9,10]);
print(ob7>2);
```

---

Data Visualization means to display data in the form of graphs or charts like bar graph, scatter chart, line chart etc. matplotlib is a python module to implement data visualization.

Few examples of programs using matplotlib

plot a line chart

```
import matplotlib.pyplot as plt
a=[1,2,3,4]
b=[2,4,6,8]
plt.plot(a,b)
plt.show()
```

---

plot a line chart and giving xlabel and ylabel

```
import matplotlib.pyplot as plt
a=[1,2,3,4]
b=[2,4,6,8]
plt.xlabel('X Values')
plt.ylabel('Y Values')
plt.plot(a,b)
```

```
plt.show()
```

---

plot a bar graph

```
import matplotlib.pyplot as plt; plt.rcdefaults()
import numpy as np
import matplotlib.pyplot as plt

objects = ('Python', 'C++', 'Java', 'Perl', 'Scala', 'Lisp')
y_pos = np.arange(len(objects))
performance = [10,8,6,4,2,1]

plt.bar(y_pos, performance)
plt.xticks(y_pos, objects)
plt.ylabel('Usage')
plt.title('Programming language usage')

plt.show()
```

---

scatter plot

```
import matplotlib.pyplot as plt
a=[1,2,3,4]
b=[2,4,6,8]
plt.scatter(a,b)
plt.show()
```

---

plot a bar graph

```
import matplotlib.pyplot as plt;
a=["java","c++","python","html"];
b=[10,8,6,7];
plt.bar(a,b);
plt.show()
```

---

plot a bar graph with ylabel

```
import matplotlib.pyplot as plt;
a=["java","c++","python","html"];
b=[10,8,6,7];
plt.ylabel("Users in Millions")
plt.bar(a,b);
plt.show()
```

---

line chart with title

```
import matplotlib.pyplot as plt
a=[1,2,3,4]
b=[2,4,6,8]
plt.title("My Chart 1")
plt.plot(a,b)
plt.show()
```

---

bar chart with yticks

```
import matplotlib.pyplot as plt;
a=["java","c++","python","html"];
b=[10,8,6,7];
plt.yticks([0,1,2,3,4,5,6,7,8,9,10])
plt.bar(a,b);
plt.show()
```

---

savefigure

```
import matplotlib.pyplot as plt;
a=["java","c++","python","html"];
b=[10,8,6,7];
plt.yticks([0,1,2,3,4,5,6,7,8,9,10])
plt.bar(a,b);
```

```
plt.savefig("my chart 1.pdf");
plt.show()
```

---

bar chart with y lim

```
import matplotlib.pyplot as plt;
a=["java","c++","python","html"];
b=[10,8,6,7];
plt.ylim(0,12);
plt.bar(a,b);
plt.show()
```

---

pie chart

```
import matplotlib.pyplot as plt;
a=["java","c++","python","html"];
b=[20,30,10,40];
plt.pie(b,labels=a);
plt.show()
```

---

pie chart with autopct

```
import matplotlib.pyplot as plt;
a=["java","c++","python","html"];
b=[20,30,10,40];
plt.pie(b,labels=a,autopct="%1.1f%%");
plt.show()
```

---

pie chart with explode

```
import matplotlib.pyplot as plt;
a=["java","c++","python","html"];
b=[20,30,10,40];
expl=[0,0.2,0,0]
plt.pie(b,labels=a,explode=expl,autopct="%1.1f%%");
```

```
plt.show()
```

---

pie chart with colours

```
import matplotlib.pyplot as plt;
a=["java","c++","python","html"];
b=[20,30,10,40];
expl=[0,0.2,0,0]
colr=["red","blue","green","yellow"]
plt.pie(b,labels=a,explode=expl,colors=colr,autopct="%1.1f%%");
plt.show()
```

---

plot more than one graph

```
import matplotlib.pyplot as plt
t1=[1,2,3,4]
t2=[2,4,6,8]
t3=[3,7,8,9]
t4=[4,5,6,7]
plt.plot(t1,t2,"r",t3,t4,"b")
plt.show()
```

---