CS601 Machine learning

A

CS601 Machine Learning Lab File

Submitted in partial fulfillment of the

requirements for the degree of

BACHELOR OF TECHNOLOGY

In

COMPUTER SCIENCE & ENGINEERING

By

Ritik Raghuwanshi

0187Cs191137

Under the guidance of

Mr. Rahul Dubey

(Assistant Professor)

May-2022

Department of COMPUTER SCIENCE &
ENGINEERING Sagar Institute of Science & Technology
(SISTec) Bhopal (M.P.)

Approved by AICTE, New Delhi & Govt. of M.P.

Affiliated to Rajiv Gandhi Proudyogiki Vishwavidyalaya, Bhopal (M.P.)

INDEX

LAB No.: 01 Feb 09, 2022

Problem Statement:

Write a program to implement the various function of NumPy

Solution:

```
import numpy as np

arr_1d=np.array([1,2,3,4])
print(arr_1d)
print(type(arr_1d))
```

```
[1 2 3 4]
<class 'numpy.ndarray'>
```

```
arr_1d=([1,2,3,4])
print(arr_1d)
print(type(arr_1d))
```

```
[1, 2, 3, 4]
<class 'list'>
```

Ndim

```
arr_1d=np.array([1,2,3,4])
print(arr_1d)
print(type(arr_1d))
```

```
[1 2 3 4]
<class 'numpy.ndarray'>
```

```
arr_1d.ndim
1
arr_2d=np.array([[1,2,3],[4,5,6],[7,8,9]])
print(arr_2d)
print(type(arr_2d))
arr_2d.ndim
[[1 2 3]
 [4 5 6]
 [7 8 9]]
<class 'numpy.ndarray'>
2
arr_md=np.array([[[1,2,3,4],[11,22,33,44],[111,222,333,444],[22,33,44,55]],
              [[12,13,14,15],[22,23,24,25],[1,1,1,1],[2,2,2,2]]])
print(arr_md)
print(type(arr_md))
arr_md.ndim
```

```
[[[  1   2   3   4]
  [ 11  22  33  44]
  [111 222 333 444]
  [ 22  33  44  55]]

 [[ 12  13  14  15]
  [ 22  23  24  25]
  [  1   1   1   1]
  [  2   2   2   2]]]
<class 'numpy.ndarray'>
3
```

Size

```
arr_1d.size
4
arr_2d.size
9
arr_md.size
32
```

Dtype

```
arr_2d.dtype

dtype('int32')

arr_1d.dtype
dtype('int32')
arr_md.dtype
dtype('int32')
arr_oned=np.array([[1,1,1],[1,1,1],[1,1,1]])
print(arr_oned)
[[1 1 1]
 [1 1 1]
 [1 1 1]]
max_1one=np.ones(5)
print(max_1one)

[1. 1. 1. 1. 1.]

max_2one=np.ones((3,4))
print(max_2one)
[[1. 1. 1. 1.]
 [1. 1. 1. 1.]
 [1. 1. 1. 1.]]
max_2one=np.ones((3,4),dtype=int)
print(max_2one)
[[1 1 1 1]
 [1 1 1 1]
 [1 1 1 1]]
```

Empty

```
em_mx=np.empty((5,5))
print(em_mx)
```

```
[[6.23042070e-307 4.67296746e-307 1.69121096e-306 1.24610994e-306
  1.33511018e-306]
 [1.33511969e-306 6.23037996e-307 6.23053954e-307 9.34609790e-307
  8.45593934e-307]
 [9.34600963e-307 1.86921143e-306 6.23061763e-307 8.90104239e-307
  6.89804132e-307]
 [9.34605716e-307 1.37962456e-306 1.42418172e-306 2.04712906e-306
  7.56589622e-307]
 [1.11258277e-307 8.90111708e-307 3.22643519e-307 9.79103798e-307
  2.46155235e-312]]
```

Arrange()

```
ar_1d=np.arange(1,13)
print(ar_1d)
```

```
[ 1  2  3  4  5  6  7  8  9 10 11 12]
```

```
ar_1d=np.arange(1,12,2)
print(ar_1d)
```

```
[ 1  3  5  7  9 11]
```

Inspace()

```
np.linspace(1,5,50)
```

```
array([1.        , 1.08163265, 1.16326531, 1.24489796, 1.32653061,
       1.40816327, 1.48979592, 1.57142857, 1.65306122, 1.73469388,
       1.81632653, 1.89795918, 1.97959184, 2.06122449, 2.14285714,
       2.2244898 , 2.30612245, 2.3877551 , 2.46938776, 2.55102041,
       2.63265306, 2.71428571, 2.79591837, 2.87755102, 2.95918367,
       3.04081633, 3.12244898, 3.20408163, 3.28571429, 3.36734694,
       3.44897959, 3.53061224, 3.6122449 , 3.69387755, 3.7755102 ,
       3.85714286, 3.93877551, 4.02040816, 4.10204082, 4.18367347,
       4.26530612, 4.34693878, 4.42857143, 4.51020408, 4.59183673,
       4.67346939, 4.75510204, 4.83673469, 4.91836735, 5.        ])
```

Reshape()

```
ar_1=np.arange(1,13)
print(ar_1)
print(type(ar_1))
```

```
[ 1  2  3  4  5  6  7  8  9 10 11 12]
<class 'numpy.ndarray'>
```

```
ar_2=ar_1.reshape(3,4)
print(ar_2)
```

```
[[ 1  2  3  4]
 [ 5  6  7  8]
 [ 9 10 11 12]]
```

```
ar_3=ar_1.reshape(2,3,2)
print(ar_3)
```

```
[[[ 1  2]
  [ 3  4]
  [ 5  6]]

 [[ 7  8]
  [ 9 10]
  [11 12]]]
```

Reval()

ar_2.ravel()

```
array([ 1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12])
```

ar_3.ravel()

```
array([ 1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12])
```

ar_2.transpose()

```
array([[ 1,  5,  9],
       [ 2,  6, 10],
       [ 3,  7, 11],
       [ 4,  8, 12]])
```

ar_2.T

```
array([[ 1,  5,  9],
       [ 2,  6, 10],
       [ 3,  7, 11],
       [ 4,  8, 12]])
```

arr1=np.arange(1,10).reshape(3,3)
arr2=np.arange(1,10).reshape(3,3)
print(arr1)
print(arr2)

```
[[1 2 3]
 [4 5 6]
 [7 8 9]]
[[1 2 3]
 [4 5 6]
 [7 8 9]]
```

print(arr1+arr2)

```
[[ 2  4  6]
 [ 8 10 12]
 [14 16 18]]
```

print(arr1-arr2)

```
[[0 0 0]
 [0 0 0]
 [0 0 0]]
```

In [39]:

print(arr1)
print(arr2)

```
[[1 2 3]
 [4 5 6]
 [7 8 9]]
[[1 2 3]
 [4 5 6]
 [7 8 9]]
```

print(arr1*arr2)

```
[[ 1  4  9]
 [16 25 36]
 [49 64 81]]
```

print(arr1 @ arr2) # cross product

```
[[ 30  36  42]
 [ 66  81  96]
 [102 126 150]]
```

In [41]:

arr1.dot(arr2)

```
array([[ 30,  36,  42],
       [ 66,  81,  96],
       [102, 126, 150]])
```

Max()

arr1.max()

9

arr1.argmax()

8

arr1.max(axis=0)

```
array([7, 8, 9])
```

arr1.max(axis=1)

```
array([3, 6, 9])
```

min()

arr1.min()

1

arr1.argmin()

0

arr1.min(axis=0)

```
array([1, 2, 3])
```

arr1.min(axis=1)

```
array([1, 4, 7])
```

np.sum(arr1)

```
45
```

np.sum(arr1,axis=0)

```
array([12, 15, 18])
```

np.sum(arr1,axis=1)

```
array([ 6, 15, 24])
```

np.mean(arr1)

5.0

np.std(arr1)

```
2.581988897471611
```

np.sqrt(arr1)

```
array([[1.        , 1.41421356, 1.73205081],
       [2.        , 2.23606798, 2.44948974],
       [2.64575131, 2.82842712, 3.        ]])
```

Python Numpy array concatenation and split

```
arr1=np.arange(1,17).reshape(4,4)
print(arr1)
[[ 1  2  3  4]
 [ 5  6  7  8]
 [ 9 10 11 12]
 [13 14 15 16]]
```

```
arr2=np.arange(17,33).reshape(4,4)
print(arr2)
[[17 18 19 20]
 [21 22 23 24]
 [25 26 27 28]
 [29 30 31 32]]
```

```
l1=[1,2,3,4,5]
l2=[6,7,8,9,10]
l1+l2
[1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
```

```
arr1+arr2
array([[18, 20, 22, 24],
       [26, 28, 30, 32],
       [34, 36, 38, 40],
       [42, 44, 46, 48]])
```

```
np.concatenate((arr1,arr2))
array([[ 1,  2,  3,  4],
       [ 5,  6,  7,  8],
       [ 9, 10, 11, 12],
       [13, 14, 15, 16],
       [17, 18, 19, 20],
       [21, 22, 23, 24],
       [25, 26, 27, 28],
       [29, 30, 31, 32]])
```

```
np.concatenate((arr1,arr2),axis=1)
array([[ 1,  2,  3,  4, 17, 18, 19, 20],
       [ 5,  6,  7,  8, 21, 22, 23, 24],
       [ 9, 10, 11, 12, 25, 26, 27, 28],
       [13, 14, 15, 16, 29, 30, 31, 32]])
```

```
np.hstack((arr1,arr2))
array([[ 1,  2,  3,  4, 17, 18, 19, 20],
       [ 5,  6,  7,  8, 21, 22, 23, 24],
       [ 9, 10, 11, 12, 25, 26, 27, 28],
       [13, 14, 15, 16, 29, 30, 31, 32]])
```

```
np.vstack((arr1,arr2))
array([[ 1,  2,  3,  4],
       [ 5,  6,  7,  8],
       [ 9, 10, 11, 12],
       [13, 14, 15, 16],
       [17, 18, 19, 20],
       [21, 22, 23, 24],
       [25, 26, 27, 28],
       [29, 30, 31, 32]])
```

```
np.hstack((arr1,arr2,arr1,arr2))
array([[ 1,  2,  3,  4, 17, 18, 19, 20,  1,  2,  3,  4, 17, 18, 19, 20],
       [ 5,  6,  7,  8, 21, 22, 23, 24,  5,  6,  7,  8, 21, 22, 23, 24],
       [ 9, 10, 11, 12, 25, 26, 27, 28,  9, 10, 11, 12, 25, 26, 27, 28],
       [13, 14, 15, 16, 29, 30, 31, 32, 13, 14, 15, 16, 29, 30, 31, 32]])
```

```
arr1
array([[ 1,  2,  3,  4],
       [ 5,  6,  7,  8],
       [ 9, 10, 11, 12],
       [13, 14, 15, 16]])
np.split(arr1,2)
[array([[1, 2, 3, 4],
        [5, 6, 7, 8]]),
 array([[ 9, 10, 11, 12],
        [13, 14, 15, 16]])]
d1=np.split(arr1,2)
print(type(d1))
<class 'list'>
```

Random

```
import random
np.random.random(5)
array([0.47825224, 0.1485511 , 0.16867452, 0.82786752, 0.88526434])

np.random.random(5)
array([0.63965221, 0.45245917, 0.11260697, 0.33661331, 0.64716726])
np.random.random((4,4))
array([[0.84406397, 0.59000831, 0.39176282, 0.81001232],
       [0.63185124, 0.66292243, 0.81106677, 0.8115016 ],
       [0.64453667, 0.41220153, 0.29552743, 0.98091944],
       [0.50680513, 0.67169127, 0.29159428, 0.79581387]])
np.random.randint(1,8)
6
np.random.randint(1,8,(4,4))
array([[4, 1, 4, 2],
       [6, 4, 2, 1],
       [1, 5, 3, 3],
       [3, 1, 5, 5]])

x=[1,2,3,4,5,6,7,8,9,10]
np.random.choice(x)
8
```

LAB No.: 02 Apr 04, 2022

Problem Statement:

2.Write a program to read the weather data set and perform the following queries using panda’s library.
a) Find maximum temperature of dataset.
b) Find the temperature and EST where event is rain.
c) Find the temperature and EST from the data set where temperature is maximum and event is rain.
3. Create two data set india_weather and US_weather and perform the concatenation and merge operation to create the joint data set also perform the left outer, right outer and outer join.

Solution:

```
import pandas as pd
import numpy as np

df =  pd.DataFrame()

print(df)
```

```
Empty DataFrame
Columns: []
Index: []
```

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

```
   0
0  1
1  2
2  3
3  4
4  5
```

```
data = [['A',10],['B',20]]
df =  pd.DataFrame(data)
print(df)
```

```
   0   1
0  A  10
1  B  20
```

```
data = [['A',10],['B',20]]
df =  pd.DataFrame(data,columns=['Name','Age'])
print(df)
```

```
  Name  Age
0    A   10
1    B   20
```

```
data = [['A',10],['B',20]]
df =  pd.DataFrame(data,columns=['Name','Age'],dtype=float)
print(df)
```

```
  Name   Age
0    A  10.0
1    B  20.0
```

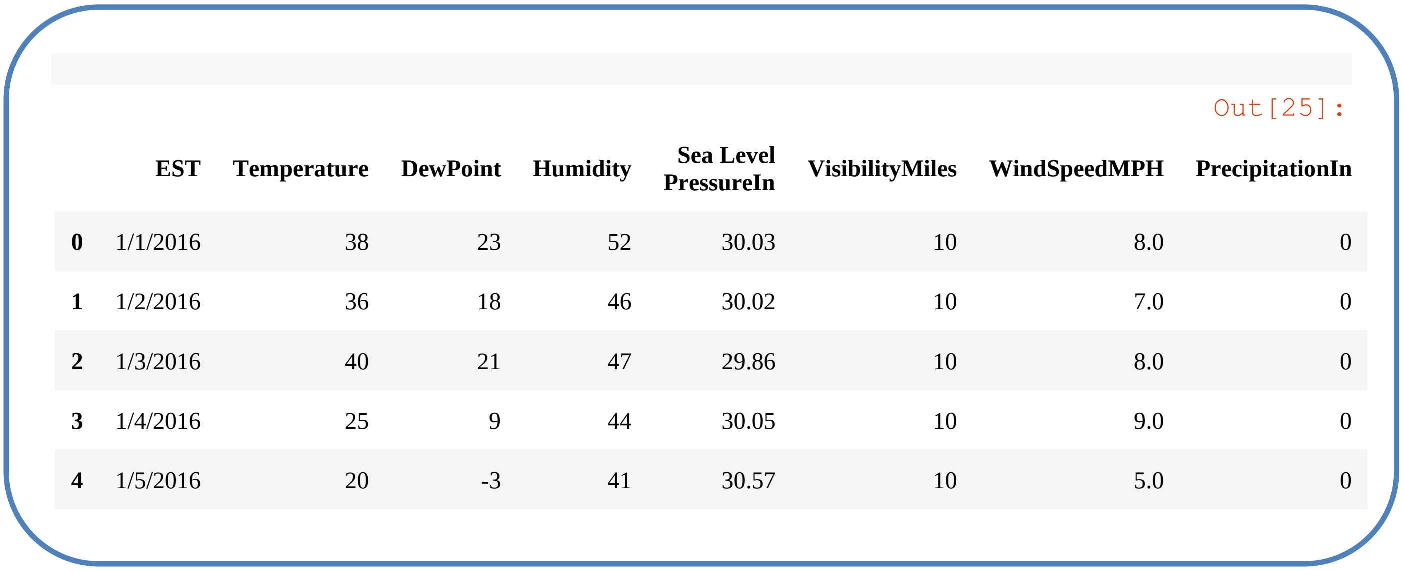
```
data = [{'a':1,'b':2},{'a':3,'b':4,'c':5}]
df =  pd.DataFrame(data,dtype=float)
print(df)
```

```
     a    b    c
0  1.0  2.0  NaN
1  3.0  4.0  5.0
```

```
df =  pd.read_csv('nyc.weather.csv')
print(df)
```

```
          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
```

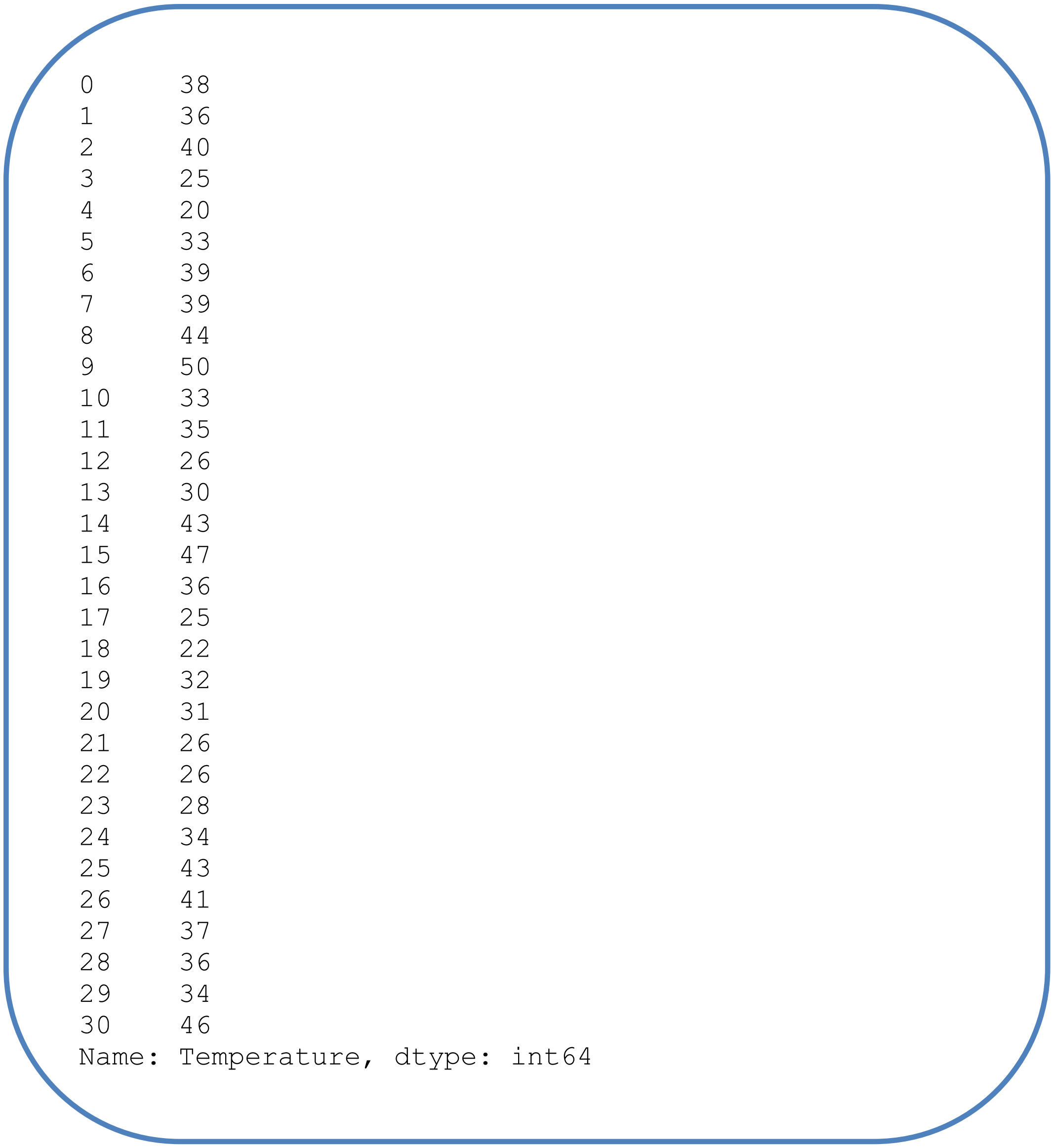
```
df.head(5)
```



Out[25]:

	EST	Temperature	DewPoint	Humidity	Sea Level PressureIn	VisibilityMiles	WindSpeedMPH	PrecipitationIn
0	1/1/2016	38	23	52	30.03	10	8.0	0
1	1/2/2016	36	18	46	30.02	10	7.0	0
2	1/3/2016	40	21	47	29.86	10	8.0	0
3	1/4/2016	25	9	44	30.05	10	9.0	0
4	1/5/2016	20	-3	41	30.57	10	5.0	0

```
x = df['Temperature']
x
```



```
0     38
1     36
2     40
3     25
4     20
5     33
6     39
7     39
8     44
9     50
10    33
11    35
12    26
13    30
14    43
15    47
16    36
17    25
18    22
19    32
20    31
21    26
22    26
23    28
24    34
25    43
26    41
27    37
28    36
29    34
30    46
Name: Temperature, dtype: int64
```

df[df['Events']=='Rain']

	EST	Temperature	Dew Point	Humidity	Sea Level PressureIn	VisibilityMiles	WindSpeedMPH	PrecipitationIn	Cloud Cover	Events	WindDirDegrees
8	1/9/2016	44	38	77	30.16	9	8.0	T	8	Rain	76
9	1/10/2016	50	46	71	29.59	4	NaN	1.8	7	Rain	109
15	1/16/2016	47	37	70	29.52	8	7.0	0.24	7	Rain	340
26	1/27/2016	41	22	45	30.03	10	7.0	T	3	Rain	311

df[['Temperature','EST']][df['Events']=='Rain']

	Temperature	**EST**
8	44	1/9/2016
9	50	1/10/2016
15	47	1/16/2016
26	41	1/27/2016

df[df['Temperature'] == df['Temperature'].max()]

	EST	Temperature	DewPoint	Humidity	Sea Level PressureIn	Visibility Miles	Wind Speed MPH	Precipitation In	CloudCover	Events	WindDirDegrees
9	1/10/2016	50	46	71	29.59	4	NaN	1.8	7	Rain	109

df[['Temperature','EST']][(df['Events']=='Rain') & (df['Temperature'] == df['Temperature'].max())]

	Temperature	**EST**
9	50	1/10/2016

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
df = pd.read_excel('weather_data.xlsx')
df.head()
```

```
day     temperature   windspeed     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
```

```
df.to_csv('r.csv')
df.to_csv('r1.csv',index=False)
```

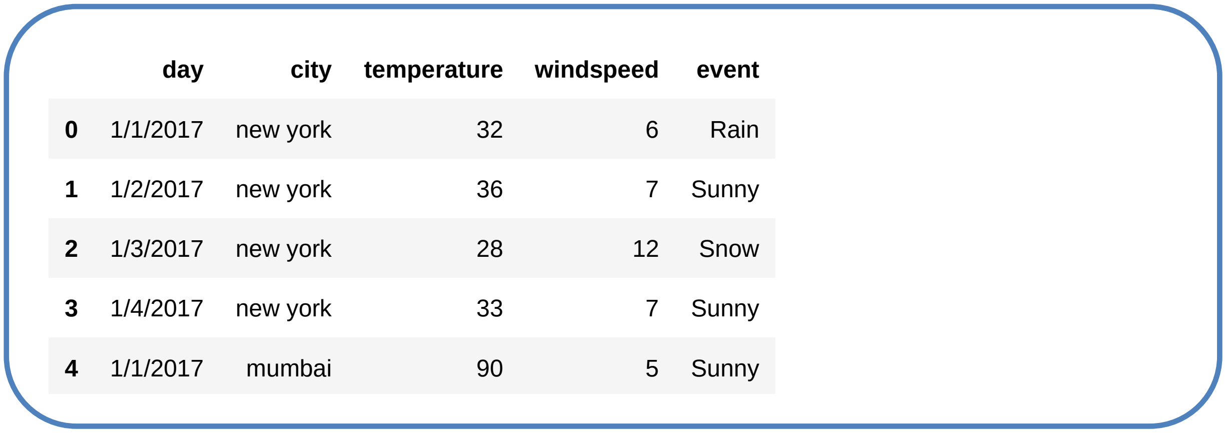
LAB No.: 03 Apr 06, 2022

Problem Statement:

4. Write a program to read the weather data from csv file and store it in xls file after performing the required modification with given sheet name
5. Write a program to extract the Toyota car data set and perform the data preprocessing to deal with continuous and categorical missing value also store the preprocessed file in new file that contain no missing values.

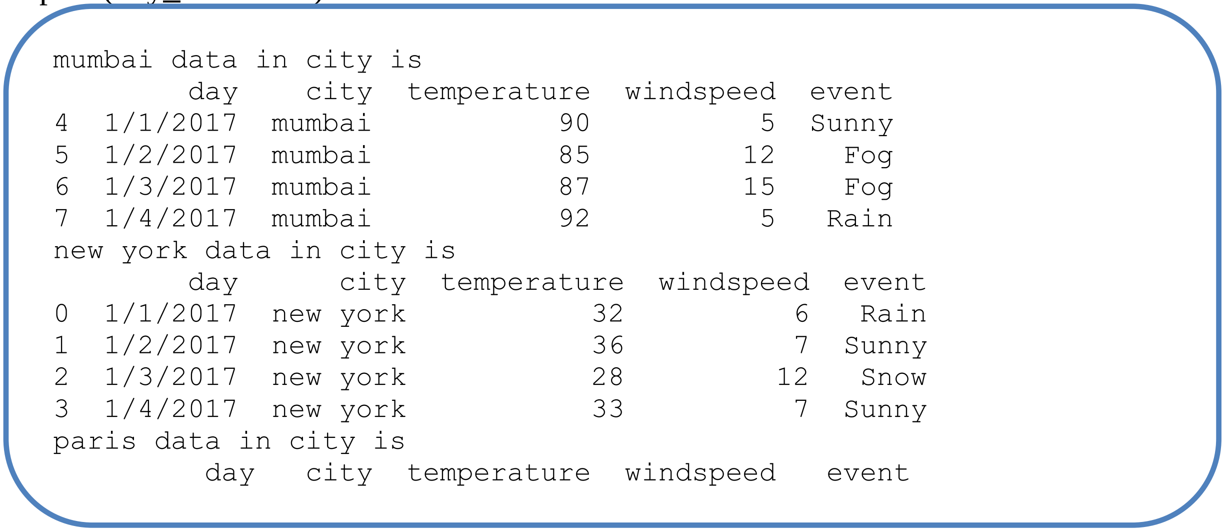
Solution:

```
# new excel file
df.to_excel('x.xls',sheet_name="r",index=False)
df.to_excel('x.xlsx',sheet_name="r",index=False)
df.to_excel('x.xlsx',sheet_name="d",index=False)
df = pd.read_csv('weather_data_cities.csv')
df.head()
```



	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

```
for city, city_dataframe in g:
    print(city,'data in city is')
    print(city_dataframe)
```



```
mumbai data in city is
        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
new york data in city is
        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
paris data in city is
         day   city  temperature  windspeed   event
```

g.get_group('paris')

```
         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
```

g.mean()

```
        temperature  windspeed
city
mumbai        88.50  9.25
new york      32.25  8.00
paris   47.75  12.75
```

g.median()

```
          temperature  windspeed
     city
   mumbai        88.5        8.5
 new york        32.5        7.0
    paris        47.5       11.5
```

g.std()

```
        temperature    windspeed
city
mumbai        3.109126       5.057997
new york      3.304038       2.708013
paris   5.315073       5.251984
```

```
g.describe()
```

```
temperature     windspeed
count   mean    std     min     25%     50%     75%     max     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    4.0
        9.25    5.057997        5.0     5.00    8.5     12.75   15.0
new york        4.0     32.25   3.304038        28.0    31.00   32.5    33.75   36.0    4.0
```

```
# concatination and mergging
indian_weather = pd.DataFrame({
   "city":["hy","ba","pu"],
   "temp":["10","20","30"],
   "wind":["90","80","70"]
})

usa_weather = pd.DataFrame({
   "city":["a","b","c"],
   "temp":["40","50","60"],
   "wind":["60","50","40"]
})

indian_weather
```

```
        city    temp    wind
0       hy      10      90
1       ba      20      80
2       pu      30      70
```

```
Usa_weather
```

```
        city    temp    wind
0       a       40      60
1       b       50      50
2       c       60      40
```

```
df = pd.concat([indian_weather,usa_weather], ignore_index=True )
df
```

	city	temp	wind
0	hy	10	90
1	ba	20	80
2	pu	30	70
3	a	40	60
4	b	50	50
5	c	60	40

```
temp = pd.DataFrame({
    "city":["hy","ba","pu","bp"],
    "temp":[10,20,30,40]
})
wind = pd.DataFrame({
    "city":["hy","ba","pu","kt"],
    "wind":[90,80,70,60]
})

df = pd.merge(temp,wind,on = 'city')

df
```

```
        city    temp    wind
0       hy      10      90
1       ba      20      80
2       pu      30      70
```

```
df = pd.merge(temp,wind,on = 'city', how='outer')

df
```

	city	temp	wind
0	hy	10.0	90.0
1	ba	20.0	80.0
2	pu	30.0	70.0
3	bp	40.0	NaN
4	kt	NaN	60.0

```
df = pd.merge(temp,wind,on = 'city', how='left')
df
```

	city	temp	wind
0	hy	10	90.0
1	ba	20	80.0
2	pu	30	70.0

	city	temp	wind
3	bp	40	NaN

```
df = pd.merge(temp,wind,on = 'city', how='right')
df = pd.DataFrame([1,2,3,4],index=[20,21,1,2])
df
```

```
        0
20      1
21      2
1       3
2       4
```

```
df.loc[2]
```

```
0    4
Name: 2, dtype: int64
```

```
df.iloc[3]
```

```
0    4
Name: 2, dtype: int64
```

```
df.iloc[:3]
```

	0
20	1
21	2
1	3

LAB No.: 04 Apr 08, 2022

Problem Statement:

6. Write a program to create a single plot using matplotlib which contain title, X axis, Y axis and data (note use NumPy library to create synthetic data set)

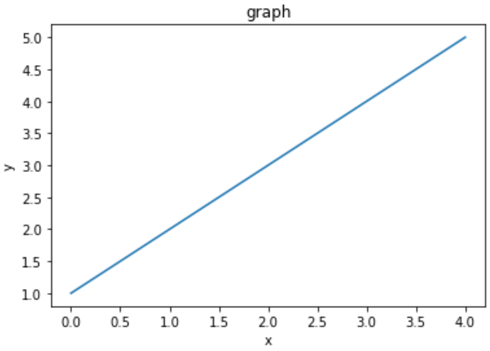
7. Write a program to create a multiple plot on single canvas for dependent and independent variable. Dependent variable should be calculated by user defined function.

Solution:

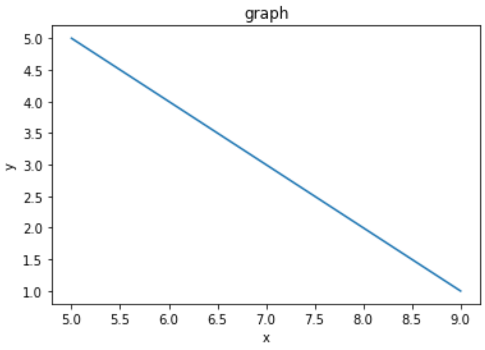
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import matplotlib
print(matplotlib.__version__)
```

```
3.4.3
```

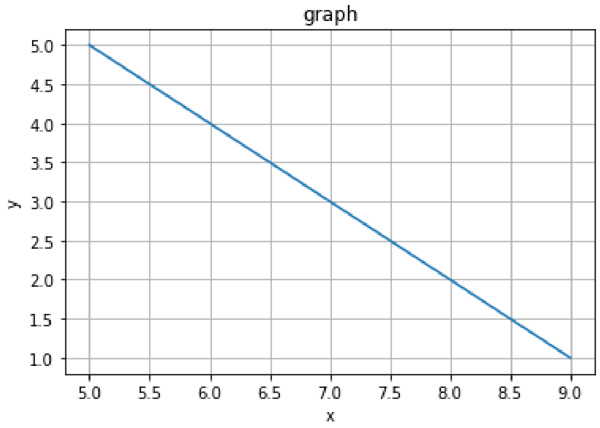
```
# Uses :
#     visualization:
#         know about data
#         to present summary of data analytics to the customer

 plt.plot([1,2,3,4,5])
plt.xlabel("x")
plt.ylabel("y")
plt.title("graph")
plt.show()
```



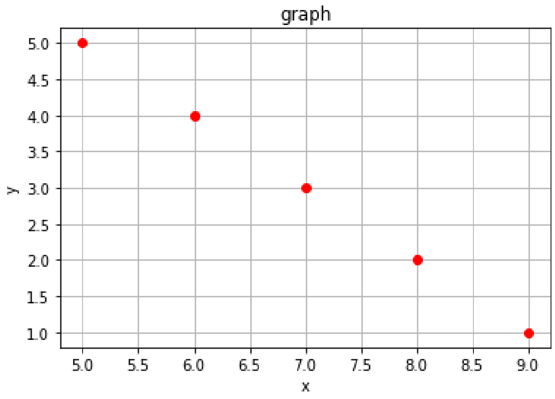
```
plt.plot([9,8,7,6,5],[1,2,3,4,5])
plt.xlabel("x")
plt.ylabel("y")
plt.title("graph")
plt.show()
```



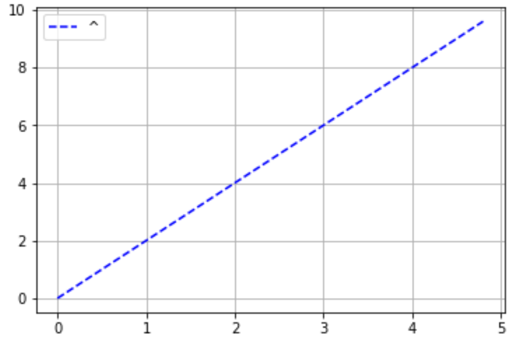
```
plt.plot([9,8,7,6,5],[1,2,3,4,5])
plt.xlabel("x")
plt.ylabel("y")
plt.title("graph")
plt.grid()
plt.show()
```



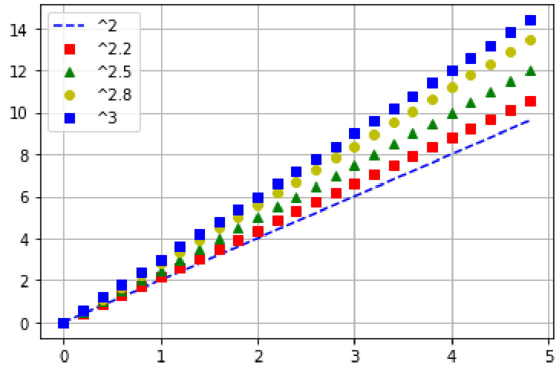
```
plt.plot([9,8,7,6,5],[1,2,3,4,5],'ro')
plt.xlabel("x")
plt.ylabel("y")
plt.title("graph")
plt.grid()
plt.show()
```



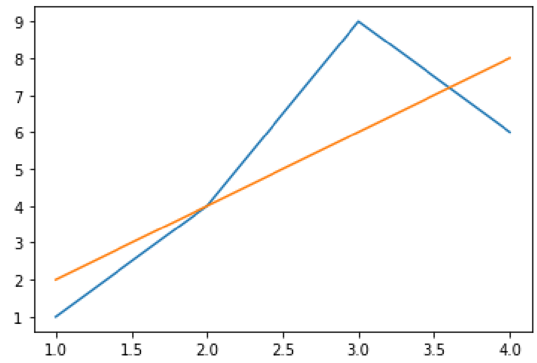
```
# multiple plot on same canvas
t = np.arange(0,5.0,0.2)
plt.plot(t,t*2,'b--',label='^')
plt.grid()
plt.legend()
plt.show()
```



```
t = np.arange(0,5.0,0.2)
plt.plot(t,t*2,'b--',label='^2')
plt.plot(t,t*2.2,'rs',label='^2.2')
plt.plot(t,t*2.5,'g^',label='^2.5')
plt.plot(t,t*2.8,'yo',label='^2.8')
plt.plot(t,t*3,'bs',label='^3')
plt.grid()
plt.legend()
plt.show()
```



```
x1 = [1,2,3,4]
y1 = [1,4,9,6]
x2 = [1,2,3,4]
y2 = [2,4,6,8]
lines = plt.plot(x1,y1,x2,y2)
```



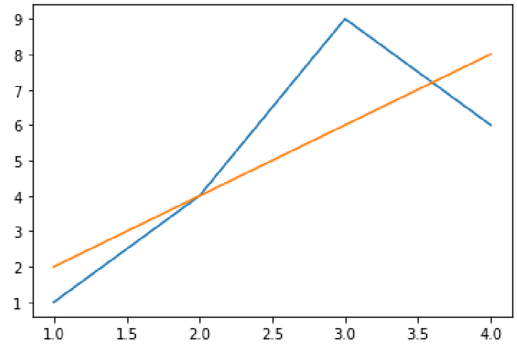
```
print(lines)
```

```
[<matplotlib.lines.Line2D object at 0x00000205692ADBE0>, <matplotlib.lines
.Line2D object at 0x00000205692ADC10>]
```

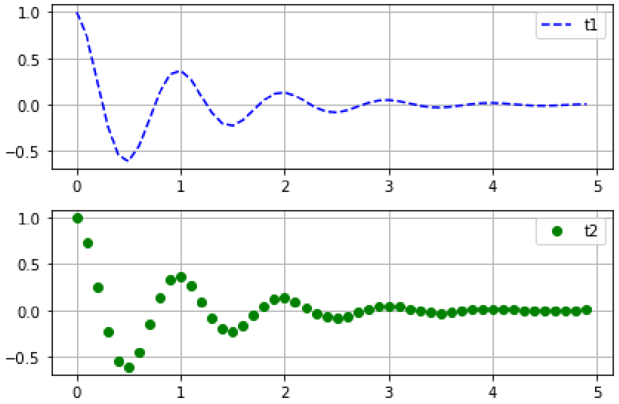
```
plt.setp(lines[0],color='b',linewidth=2.0)
plt.setp(lines[1],color='g',linewidth=5.0)
```

```
[None, None]
```

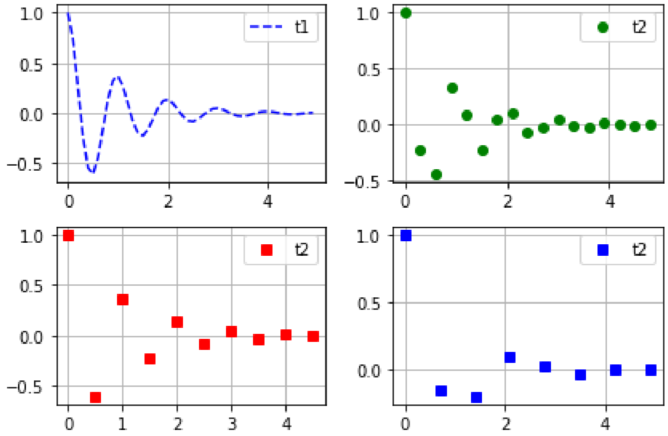
```
x1 = [1,2,3,4]
y1 = [1,4,9,6]
x2 = [1,2,3,4]
y2 = [2,4,6,8]
plt.setp(lines[0],color='b',linewidth=2.0)
plt.setp(lines[1],color='g',linewidth=5.0)
lines = plt.plot(x1,y1,x2,y2)
```



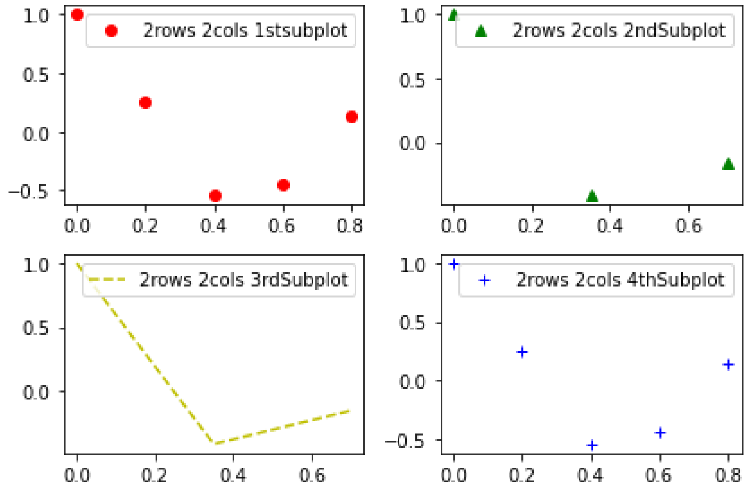
```
def f(t):
   return (np.exp(-t)*np.cos(2*np.pi*t))

t1 = np.arange(0.,5.,0.1)
t2 = np.arange(0.,5.,0.3)
plt.figure(1)
plt.subplot(211)
plt.grid()
plt.plot(t1,f(t1),'b--',label='t1')
plt.legend()
plt.subplot(212)
plt.grid()
plt.plot(t1,f(t1),'go',label='t2')
plt.legend()

plt.tight_layout()
plt.show()
```



```
def f(t):
    return (np.exp(-t)*np.cos(2*np.pi*t))
t1 = np.arange(0.,5.,0.1)
t2 = np.arange(0.,5.,0.3)
t3 = np.arange(0.,5.,0.5)
t4 = np.arange(0.,5.,0.7)
plt.figure(1)

plt.subplot(221)
plt.grid()
plt.plot(t1,f(t1),'b--',label='t1')
plt.legend()

plt.subplot(222)
plt.grid()
plt.plot(t2,f(t2),'go',label='t2')
plt.legend()

plt.subplot(223)
plt.grid()
plt.plot(t3,f(t3),'rs',label='t2')
plt.legend()

plt.subplot(224)
plt.grid()
plt.plot(t4,f(t4),'bs',label='t2')
plt.legend()

plt.tight_layout()
plt.show()
```



```
def f(t):
    return np.exp(-t) * np.cos(2*np.pi*t);

t1=np.arange(0,1,0.2);
t2=np.arange(0,1,0.35);

plt.figure(1);

plt.subplot(221);
plt.plot(t1,f(t1),'ro',label='2rows 2cols 1stsubplot')
plt.legend();

plt.subplot(222);
plt.plot(t2,f(t2),'g^',label='2rows 2cols 2ndSubplot')
plt.legend();

plt.subplot(223);
plt.plot(t2,f(t2),'y--',label='2rows 2cols 3rdSubplot')
plt.legend();

plt.subplot(224);
plt.plot(t1,f(t1),'b+',label='2rows 2cols 4thSubplot');

plt.legend();
plt.tight_layout()
plt.show()
```


2rows 2cols 1stsubplot
2rows 2cols 2ndSubplot
2rows 2cols 3rdSubplot
2rows 2cols 4thSubplot
1.0
0.5
0.0
−0.5
0.0
0.2
0.4
0.6
0.8