Machine Learning
Assignment – 2
Student Name: Sowjanya Sunkavalli
Student ID: 700731896

NumPy

Using NumPy create random vector of size 15 having only Integers in the range 1-20.

1 . Numpy:

Using NumPy create random vector of size 15 having only Integers in the range 1-20.

In [1]:
```
import pandas as pd # Importing Pandas Library
import numpy as np # Importing NumPy Library
```

In [2]:
```
arr = np.random.randint(1,20,15) # array with range 1 to 20 with vector Size as 15
arr
```

Out[2]:
```
array([10, 18, 13,  5,  3,  2, 10, 13, 10, 15,  9, 16,  6, 15,  1])
```

Firstly, Import pandas and NumPy Libraries as pd and np respectively
Create an NumPy array **arr** with vector size of 15 of values ranging from 1 to 20.
random function is used to give random values and randint take the range and size of the array to be printed as output.

1. Reshape the array to 3 by 5

1. Reshape the array to 3 by 5

In [3]:
```
arr.resize((3,5)) # resizing the array with 3by5 size
arr
```

Out[3]:
```
array([[10, 18, 13,  5,  3],
       [ 2, 10, 13, 10, 15],
       [ 9, 16,  6, 15,  1]])
```

resize () function is used to resize the array for the required mxn size, here we reshaped arr to 3 by 5.

2. Print array shape.

2. Print array shape.

In [4]:
```
arr.shape # Getting the array shape
```

Out[4]:
```
(3, 5)
```

shape is used to find the shape of the given array.

3. Replace the max in each row by 0

3. Replace the max in each row by 0

In [5]:
```
# Iterating throw all the elements and replacing maximun values with 0
for i in arr:
    m = i.argmax()
    i[m] = 0
arr
```

Out[5]:
```
array([[10,  0, 13,  5,  3],
       [ 2, 10, 13, 10,  0],
       [ 9,  0,  6, 15,  1]])
```

In the above we are iterating through all the elements in the array and getting index of max value in each row and assigning 0 to that position's value.

PANDAS

1. Read the provided CSV file 'data.csv'.

2. Pandas

1. Read the provided CSV file 'data.csv'

In [17]:
```
df = pd.read_csv('data.csv') # reads the csv file mentioned
#print(df.to_string())
df # Prints the dataframe
```

Out[17]:

	Duration	Pulse	Maxpulse	Calories
0	60	110	130	409.1
1	60	117	145	479.0
2	60	103	135	340.0
3	45	109	175	282.4
4	45	117	148	406.0
...	...	...	...	...
164	60	105	140	290.8
165	60	110	145	300.0
166	60	115	145	310.2
167	75	120	150	320.4
168	75	125	150	330.4

169 rows × 4 columns

In the above code, we read the data.csv file by using read_csv function and assigned that output to dataframe df.

printf(df.to_string) – shows all the values

Simple df shows only few values but briefly gives the rows and columns values.

2. Show the basic statistical description about the data.

2. Show the basic statistical description about the data

```
In [7]:  1 df.describe() # describe() function gives the statistical description
```

Out[7]:

	Duration	Pulse	Maxpulse	Calories
count	169.000000	169.000000	169.000000	164.000000
mean	63.846154	107.461538	134.047337	375.790244
std	42.299949	14.510259	16.450434	266.379919
min	15.000000	80.000000	100.000000	50.300000
25%	45.000000	100.000000	124.000000	250.925000
50%	60.000000	105.000000	131.000000	318.600000
75%	60.000000	111.000000	141.000000	387.600000
max	300.000000	159.000000	184.000000	1860.400000

describe () function gives the all the statistical description of the given dataframe.

3. Check if the data has null values.

a. Replace the null values with the mean

3. Check if the data has null values.

a. Replace the null values with the mean

```
In [11]:  1 df.isnull().any() #isnull is used to check null values
```

```
Out[11]: Duration    False
         Pulse       False
         Maxpulse    False
         Calories     True
         dtype: bool
```

```
In [12]:  1 df.fillna(df.mean(),axis=0,inplace=True) # replacing Calories null values with its mean value
          2 print(df.to_string()) # to_string() is used to view all the values
```

```
     Duration  Pulse  Maxpulse    Calories
0          60    110       130  409.100000
1          60    117       145  479.000000
2          60    103       135  340.000000
3          45    109       175  282.400000
4          45    117       148  406.000000
5          60    102       127  300.000000
6          60    110       136  374.000000
7          45    104       134  253.300000
8          30    109       133  195.100000
9          60     98       124  269.000000
10         60    103       147  329.300000
11         60    100       120  250.700000
12         60    106       128  345.300000
13         60    104       132  379.300000
14         60     98       123  275.000000
15         60     98       120  215.200000
16         60    100       120  300.000000
17         45     90       112  375.790244
```

isnull (). any () shows the columns having Null Values.
fillna is used to fill the null values with defined values passed in the function
Here we have passed mean () value of Calories to all the Null values in Calories column.

4. Select at least two columns and aggregate the data using: min, max, count, mean.

4. Select at least two columns and aggregate the data using: min, max, count, mean.

In [13]:
```
#considering first two coloumns and getting their aggregate values
df[['Duration','Pulse']].aggregate(func=['min','max','count','mean'],axis=0,)
```

Out[13]:

	Duration	Pulse
min	15.000000	80.000000
max	300.000000	159.000000
count	169.000000	169.000000
mean	63.846154	107.461538

We are using aggregate function on two columns Duration and Pulse and getting min, max, count, mean across columns levels.

5. Filter the dataframe to select the rows with calories values between 500 and 1000.

5. Filter the dataframe to select the rows with calories values between 500 and 1000.

In [14]:
```
df[(df['Calories']>=500) & (df['Calories']<=1000)] # filtering dataframe as per given conditions
```

Out[14]:

	Duration	Pulse	Maxpulse	Calories
51	80	123	146	643.1
62	160	109	135	853.0
65	180	90	130	800.4
66	150	105	135	873.4
67	150	107	130	816.0
72	90	100	127	700.0
73	150	97	127	953.2
75	90	98	125	563.2
78	120	100	130	500.4
83	120	100	130	500.0
90	180	101	127	600.1
99	90	93	124	604.1
101	90	90	110	500.0
102	90	90	100	500.0
103	90	90	100	500.4
106	180	90	120	800.3
108	90	90	120	500.3

Filtering the dataframe with the given conditions for getting values between 500 and 1000 by using >=, & and <= conditions.

6. Filter the dataframe to select the rows with calories values > 500 and pulse < 100.

6. Filter the dataframe to select the rows with calories values > 500 and pulse < 100.

In [15]:
```
# Filtering Dataframe as per given conditions
df[(df['Calories']>500)&(df['Pulse']<100)]
```

Out[15]:

	Duration	Pulse	Maxpulse	Calories
65	180	90	130	800.4
70	150	97	129	1115.0
73	150	97	127	953.2
75	90	98	125	563.2
99	90	93	124	604.1
103	90	90	100	500.4
106	180	90	120	800.3
108	90	90	120	500.3

Filtering the dataframe with the given conditions for getting values between 500 and 1000 by using >, & and < conditions.

7. Create a new "df_modified" dataframe that contains all the columns from df except for "Maxpulse"

7. Create a new "df_modified" dataframe that contains all the columns from df except for "Maxpulse".

In [16]:
```
df_modified = df[['Duration','Pulse','Calories']] # creating new dataframe
df_modified
```

Out[16]:

	Duration	Pulse	Calories
0	60	110	409.1
1	60	117	479.0
2	60	103	340.0
3	45	109	282.4
4	45	117	406.0
...	...	...	...
164	60	105	290.8
165	60	110	300.0
166	60	115	310.2
167	75	120	320.4
168	75	125	330.4

169 rows × 3 columns

Creating new dataframe df_modified with columns Durations, Pulse and Calories except MaxPulse by passing these parameters into new dataframe and assigning that dataframe to new dataframe df_modified.

8. Delete the “Maxpulse” column from the main df dataframe

8. Delete the “Maxpulse” column from the main df dataframe

```
In [18]:
1 df.drop(labels = ['Maxpulse'],axis=1,inplace=True) # drop function is used to remove specified values from dataframe
2 df
```

Out[18]:

	Duration	Pulse	Calories
0	60	110	409.1
1	60	117	479.0
2	60	103	340.0
3	45	109	282.4
4	45	117	406.0
...	...	...	...
164	60	105	290.8
165	60	110	300.0
166	60	115	310.2
167	75	120	320.4
168	75	125	330.4

169 rows × 3 columns

drop() function is used to delete the specified value from the dataframe.
Here we are deleting column Maxpulse from the dataframe.

9. Convert the datatype of Calories column to int datatype.

9. Convert the datatype of Calories column to int datatype.

```
In [25]:
1 df['Calories'] = df['Calories'].astype(int) # converting to int datatype
2 df.dtypes #showing the datatype of df dataframe
```

```
Out[25]: Duration    int64
         Pulse       int64
         Calories    int32
         dtype: object
```

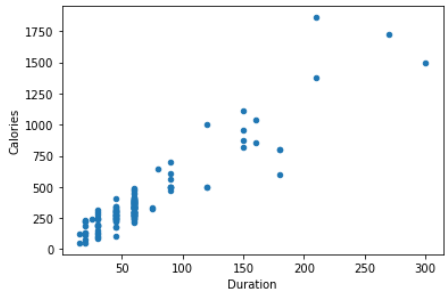
astype(int) is used to convert the datatype to int.
df.dtypes is used to get the datatype of all columns in dataframe df.

10. Using pandas create a scatter plot for the two columns (Duration and Calories).

10. Using pandas create a scatter plot for the two columns (Duration and Calories).

```
In [20]:  1 df.plot.scatter(x='Duration',y='Calories') # plotting the graph from the values
```

```
Out[20]: <AxesSubplot:xlabel='Duration', ylabel='Calories'>
```



Getting Scatter plot from the existing dataframe df by providing x-axis parameter column as Duration and y-axis parameter column as Calories.

3. Matplotlib

1. Write a Python programming to create a below chart of the popularity of programming Languages.

2. Sample data: Programming languages: Java, Python, PHP, JavaScript, C#, C++
Popularity: 22.2, 17.6, 8.8, 8, 7.7, 6.7

3. Matplotlib

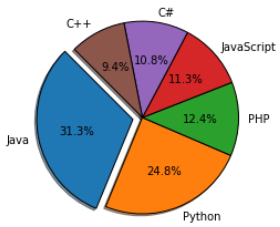
1. Write a Python programming to create a below chart of the popularity of programming Languages.

2. Sample data:

Programming languages: Java, Python, PHP, JavaScript, C#, C++

Popularity: 22.2, 17.6, 8.8, 8, 7.7, 6.7

```
In [21]:  1 %matplotlib inline
          2 from matplotlib import pyplot as plt # importing pyplot from matlab
          3 programming_languages = ['Java','Python','PHP','JavaScript','C#','C++']
          4 popularity = [22.2,17.6,8.8,8,7.7,6.7]
          5 explode = [0.1, 0, 0, 0, 0, 0]
          6 plt.pie(popularity, labels=programming_languages, explode=explode, shadow=True,startangle=135,
          7         autopct='%1.1f%%',
          8         wedgeprops={'edgecolor': 'black'}) # passing the attributes in pie chart
          9 plt.show() # showing the pie chart
```



%matplotlib inline gives the output plotting in the same cell.
Next, we imported pyplot as plt from matplotlib library

Then we have initialized list of Programming languages as programming_languages
And their popularity values as list popularity.
And then as per list to make java sector explode, we have made its value as 0.1 and remaining values as 0 only.
And then we have passed all the lists and set the parameters for color, shadow to the pie function from the plot class.
And finally, we have shown the pie chart using plt.show() function.

Demonstration Video Link: https://youtu.be/40PgOvrEPkI

GitHub Link : https://github.com/sunkavallisowjanya/MachineLearning_Assignment2