# Lab2. Red Wine Quality Data Analysis using NumPy Part-II

# Rollno:225229104

In [3]:
```
import numpy as np
```

In [4]:
```
wines = np.genfromtxt("winequality-red.csv", delimiter=";", skip_header=1)
```

# NumPy Aggregation Methods

## Find sum of all residual sugar values

In [5]:
```
p=wines[:,3]
sum(p)
```

Out[5]:
```
4059.550000000003
```

## Find sums of every feature value. There are 12 features altogether

In [6]:
```
sum(wines)
```

Out[6]:
```
array([13303.1    ,   843.985  ,   433.29   ,  4059.55   ,   139.859  ,
       25384.     , 74302.     ,  1593.79794,  5294.47   ,  1052.38   ,
       16666.35   ,  9012.     ])
```

## Find sum of every row

In [7]:
```
a=wines[:,:].sum(axis=1)
a
```

Out[7]:
```
array([ 74.5438 , 123.0548 ,  99.699  , ..., 100.48174, 105.21547,
        92.49249])
```

## What is its size?

In [8]:
```
len(wines)
```

Out[8]:
```
1599
```

## What is the maximum residual sugar value in red wines data?

In [9]:
```
z=wines[:,3]
z=z.astype('int32')
z
```

Out[9]:
```
array([1, 2, 2, ..., 2, 2, 3])
```

## find its maximum residual sugar value

In [10]:
```
max(z)
```

Out[10]:
```
15
```

## What is the minimum residual sugar value in red wines data?

In [11]:
```
min(z)
```

Out[11]:
```
0
```

## What is the average residual sugar value in red wines data?

In [12]:
```
np.mean(p)
```

Out[12]:
```
2.53880550343965
```

## What is 25 percentile residual sugar value?

In [13]:
```
np.percentile(p,25)
```

Out[13]:
```
1.9
```

## What is 75 percentile residual sugar value?

In [14]:
```
np.percentile(p,75)
```

Out[14]:
```
2.6
```

## Find the average of each feature value

In [15]:
```
x=wines[:,:]
x
x.mean(axis=0)
```

Out[15]:
```
array([ 8.31963727,  0.52782051,  0.27097561,  2.5388055 ,  0.08746654,
       15.87492183, 46.46779237,  0.99674668,  3.3111132 ,  0.65814884,
       10.42298311,  5.63602251])
```

# NumPy Array Comparisons

## Show all wines with quality > 5

In [16]:
```
wines[:,11]>5
```

Out[16]:
```
array([False, False, False, ...,  True, False,  True])
```

## Show all wines with quality > 7

In [17]:
```
wines[:,11]>7
```

Out[17]:
```
array([False, False, False, ..., False, False, False])
```

## check if any wines value is True for the condition quality > 7

In [18]:
```
x=wines[:,11]>7
True in x
```

Out[18]:
```
True
```

## Show first 3 rows where wine quality > 7, call it high_quality

In [19]:
```
high_quality =wines[:,11]>7
high_quality
```

Out[19]:
```
array([False, False, False, ..., False, False, False])
```

## Show only top 3 rows and all columns of high_quality wines data

In [20]:
```
wines[high_quality][0:3]
```

Out[20]:
```
array([[7.900e+00, 3.500e-01, 4.600e-01, 3.600e+00, 7.800e-02, 1.500e+01,
        3.700e+01, 9.973e-01, 3.350e+00, 8.600e-01, 1.280e+01, 8.000e+00],
       [1.030e+01, 3.200e-01, 4.500e-01, 6.400e+00, 7.300e-02, 5.000e+00,
        1.300e+01, 9.976e-01, 3.230e+00, 8.200e-01, 1.260e+01, 8.000e+00],
       [5.600e+00, 8.500e-01, 5.000e-02, 1.400e+00, 4.500e-02, 1.200e+01,
        8.800e+01, 9.924e-01, 3.560e+00, 8.200e-01, 1.290e+01, 8.000e+00]])
```

### Show wines with a lot of alcohol > 10 and high wine quality > 7

In [21]:
```
xy=(wines[:,10]>10)&(wines[:,11]>7)
xy
```

Out[21]:
```
array([False, False, False, ..., False, False, False])
```

### show only alcohol and wine quality columns

In [22]:
```
wines[xy,10:]
```

Out[22]:
```
array([[12.8,  8. ],
       [12.6,  8. ],
       [12.9,  8. ],
       [13.4,  8. ],
       [11.7,  8. ],
       [11. ,  8. ],
       [11. ,  8. ],
       [14. ,  8. ],
       [12.7,  8. ],
       [12.5,  8. ],
       [11.8,  8. ],
       [13.1,  8. ],
       [11.7,  8. ],
       [14. ,  8. ],
       [11.3,  8. ],
       [11.4,  8. ]])
```

# Combining NumPy Arrays

## Combine red wine and white wine data

#### Open white wine dataset

In [23]:
```
white_wines = np.genfromtxt("winequality-white.csv", delimiter=";", skip_header=1
```

### Show size of white_wines

In [24]:
```
white_wines.shape
```

Out[24]:
```
(4898, 12)
```

### combine wines and white_wines data frames using vstack and call it all_wines

In [25]:
```
all_wines=np.vstack((wines,white_wines))
```

In [26]:
```
all_wines.shape
```

Out[26]:
```
(6497, 12)
```

### Combine wines and white_wines data frames using concatenate method

In [27]:
```
all_wines1=np.concatenate((wines,white_wines),axis=0)
all_wines1.shape
```

Out[27]:
```
(6497, 12)
```

## Matrix Operations and Reshape

### Find Transpose of wines and print its size

In [28]:
```
tranpose=wines.T
tranpose.shape
```

Out[28]:
```
(12, 1599)
```

### Convert wines data into 1D array

In [29]:
```
wines.ravel()
```

Out[29]:
```
array([ 7.4 ,  0.7 ,  0.  , ...,  0.66, 11.  ,  6.  ])
```

In [30]:
```
wines.ravel().shape
```

Out[30]:
```
(19188,)
```

### Reshape second row of wines into a 2-dimensional array with 2 rows and 6 columns

In [31]:
```
wines[1].reshape((2,6))
```

Out[31]:
```
array([[ 7.8   ,  0.88  ,  0.    ,  2.6   ,  0.098 , 25.    ],
       [67.    ,  0.9968,  3.2   ,  0.68  ,  9.8   ,  5.    ]])
```

## Sort alcohol column Ascending Order

In [34]:
```
sorted_alcohol=np.sort(wines[:,-2])
sorted_alcohol
```

Out[34]:
```
array([ 8.4,  8.4,  8.5, ..., 14. , 14. , 14.9])
```

## Make sorting to take place in-place

In [35]:
```
wines[:,-2].sort()
```

In [36]:
```
wines[:,-2]
```

Out[36]:
```
array([ 8.4,  8.4,  8.5, ..., 14. , 14. , 14.9])
```

## Sort alcohol column Descending Order

In [38]:
```
sorted_alcohol_desc=np.sort(wines[:,10])[::-1]
sorted_alcohol_desc
```

Out[38]:
```
array([14.9, 14. , 14. , ...,  8.5,  8.4,  8.4])
```

## Will original data be modified?. Check top 10 rows

In [39]:
```
wines[:,-2]
```

Out[39]:
```
array([ 8.4,  8.4,  8.5, ..., 14. , 14. , 14.9])
```