0.1 Red Wine Quality Data Analysis using NumPy Part-I

In [37]:

```
'''
Wine quality dataset 11 input features and 1 output feature
1 - fixed acidity
2 - volatile acidity
3 - citric acid
4 - residual sugar
5 - chlorides
6 - free sulfur dioxide
7 - total sulfur dioxide
8 - density
9 - pH
10 - sulphates
11 - alcohol
Output variable (based on sensory data):
12 - quality (score between 0 and 10)'''
```

Out[37]:

```
'\nWine quality dataset 11 input features and 1 output feature\n1 - fixed
acidity\n2 - volatile acidity\n3 - citric acid\n4 - residual sugar\n5 - ch
lorides\n6 - free sulfur dioxide\n7 - total sulfur dioxide\n8 - density\n9
- pH\n10 - sulphates\n11 - alcohol\nOutput variable (based on sensory dat
a):\n12 - quality (score between 0 and 10)'
```

In [2]:

```
import numpy as np
```

In [3]:

```
wines = np.genfromtxt("winequality-red.csv", delimiter=";", skip_header=1)
```

What is its shape?

In [4]:

```
 wines.shape
```

Out[4]:

```
(1599, 12)
```

How many wine data rows here?

In [5]:

```
 wines.shape[0]
```

Out[5]:

```
1599
```

How many wine data columns here?

In [6]:

```
 wines.shape[1]
```

Out[6]:

```
12
```

How many dimensions?

In [7]:

```
wines.ndim
```

Out[7]:

```
2
```

What is the type of data?

In [8]:

```
 type(wines)
```

Out[8]:

```
numpy.ndarray
```

What is the data type of wines data?

In [9]:

```
 wines.dtype
```

Out[9]:

```
dtype('float64')
```

Show top 5 rows.

In [12]:

```
wines[:5, :]
```

Out[12]:

```
array([[7.400e+00, 7.000e-01, 0.000e+00, 1.900e+00, 7.600e-02, 1.100e+01,
        3.400e+01, 9.978e-01, 3.510e+00, 5.600e-01, 9.400e+00, 5.000e+00],
       [7.800e+00, 8.800e-01, 0.000e+00, 2.600e+00, 9.800e-02, 2.500e+01,
        6.700e+01, 9.968e-01, 3.200e+00, 6.800e-01, 9.800e+00, 5.000e+00],
       [7.800e+00, 7.600e-01, 4.000e-02, 2.300e+00, 9.200e-02, 1.500e+01,
        5.400e+01, 9.970e-01, 3.260e+00, 6.500e-01, 9.800e+00, 5.000e+00],
       [1.120e+01, 2.800e-01, 5.600e-01, 1.900e+00, 7.500e-02, 1.700e+01,
        6.000e+01, 9.980e-01, 3.160e+00, 5.800e-01, 9.800e+00, 6.000e+00],
       [7.400e+00, 7.000e-01, 0.000e+00, 1.900e+00, 7.600e-02, 1.100e+01,
        3.400e+01, 9.978e-01, 3.510e+00, 5.600e-01, 9.400e+00, 5.000e+0
0]])
```

What is the value at 3rd row, 4th column of wine data?

In [13]:

```
wines[2,3]
```

Out[13]:

```
2.3
```

Select first 3 items in 4th column.

In [14]:

```
wines[:3, 3]
```

Out[14]:

```
array([1.9, 2.6, 2.3])
```

Show 1st column.

In [15]:

```
 wines[:, 0]
```

Out[15]:

```
array([7.4, 7.8, 7.8, ..., 6.3, 5.9, 6. ])
```

Show 2nd row.

In [16]:

```
wines[1, :]
```

Out[16]:

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

Select items from rows 1 to 3 and 5th column.

In [17]:

```
wines[1:4, 4]
```

Out[17]:

```
array([0.098, 0.092, 0.075])
```

Select entire array.

In [18]:

```
wines[:,:]
```

Out[18]:

```
array([[ 7.4  ,  0.7  ,  0.   , ...,  0.56 ,  9.4  ,  5.   ],
       [ 7.8  ,  0.88 ,  0.   , ...,  0.68 ,  9.8  ,  5.   ],
       [ 7.8  ,  0.76 ,  0.04 , ...,  0.65 ,  9.8  ,  5.   ],
       ...,
       [ 6.3  ,  0.51 ,  0.13 , ...,  0.75 , 11.   ,  6.   ],
       [ 5.9  ,  0.645,  0.12 , ...,  0.71 , 10.2  ,  5.   ],
       [ 6.   ,  0.31 ,  0.47 , ...,  0.66 , 11.   ,  6.   ]])
```

Change 1st value in wines to 100.

In [19]:

```
wines[0,0]
```

Out[19]:

```
7.4
```

In [20]:

```
wines[0,0] = 100
```

In [21]:

```
 wines[0,0]
```

Out[21]:

```
100.0
```

In [22]:

```
wines[0,0] = 7.4     # change it back
```

In [23]:

```
wines[0,0]
```

Out[23]:

```
7.4
```

0.1.1 1-Dimensional Numpy Arrays.

Select 4th row all column values.

In [24]:

```
third_wine = wines[3, :]
```

In [25]:

```
 third_wine
```

Out[25]:

```
array([11.2  ,  0.28 ,  0.56 ,  1.9  ,  0.075, 17.   , 60.   ,  0.998,
        3.16 ,  0.58 ,  9.8  ,  6.   ])
```

In [26]:

```
third_wine[1]
```

Out[26]:

```
0.28
```

Convert one datatype to another.

In [27]:

```
wines.astype(int)     #convert to int
```

Out[27]:

```
array([[ 7,  0,  0, ...,  0,  9,  5],
       [ 7,  0,  0, ...,  0,  9,  5],
       [ 7,  0,  0, ...,  0,  9,  5],
       ...,
       [ 6,  0,  0, ...,  0, 11,  6],
       [ 5,  0,  0, ...,  0, 10,  5],
       [ 6,  0,  0, ...,  0, 11,  6]])
```

0.1.2 Vectorization Operations.

Increase wine quality score (output variable) by 10.

In [28]:

```
wines[:, 11]      # check values first
```

Out[28]:

```
array([5., 5., 5., ..., 6., 5., 6.])
```

In [29]:

```
wines[:, 11] += 10
```

In [30]:

```
 wines[:, 11]
```

Out[30]:

```
array([15., 15., 15., ..., 16., 15., 16.])
```

Multiply alcohol of all wine data by 3 times.

In [31]:

```
wines[:, 10] *= 3
```

In [32]:

```
wines[:, 10]
```

Out[32]:

```
array([28.2, 29.4, 29.4, ..., 33. , 30.6, 33. ])
```

Add quality column by itselt.

In [33]:

```
wines[:, 11] + wines[:, 11]    # It will produce a new array
```

Out[33]:

```
array([30., 30., 30., ..., 32., 30., 32.])
```

Multiply alcohol and wine quality columns. It will perform element wise multiplication.

In [38]:

```
 wines[:,10] * wines[:,11]
```

Out[38]:

```
array([423., 441., 441., ..., 528., 459., 528.])
```

0.1.3 Broadcasting.

Add every row of wines data with a random array of values.

In [34]:

```
 rand_array = np.random.rand(12)   #here 12, becos there are 12 columns
```

In [35]:

```
 rand_array
```

Out[35]:

```
array([0.65370756, 0.43076319, 0.5491816 , 0.72269602, 0.41190467,
       0.18686757, 0.70195118, 0.31917323, 0.68695962, 0.11152442,
       0.86673255, 0.89851711])
```

In [36]:

```
wines + rand_array
```

Out[36]:

```
array([[ 8.05370756,  1.13076319,  0.5491816 , ...,  0.67152442,
        29.06673255, 15.89851711],
       [ 8.45370756,  1.31076319,  0.5491816 , ...,  0.79152442,
        30.26673255, 15.89851711],
       [ 8.45370756,  1.19076319,  0.5891816 , ...,  0.76152442,
        30.26673255, 15.89851711],
       ...,
       [ 6.95370756,  0.94076319,  0.6791816 , ...,  0.86152442,
        33.86673255, 16.89851711],
       [ 6.55370756,  1.07576319,  0.6691816 , ...,  0.82152442,
        31.46673255, 15.89851711],
       [ 6.65370756,  0.74076319,  1.0191816 , ...,  0.77152442,
        33.86673255, 16.89851711]])
```