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

Name:Harshavardhini K

Rollno:235229108

In [19]:
```
import numpy as np
```

In [20]:
```
wines = np.genfromtxt("winequality-red.csv", delimiter=";", skip_header=1)
```

NumPy Aggregation Methods

Find sum of all residual sugar values

In [21]:
```
x=wines[:,3]
sum(x)
```

Out[21]:
```
4059.550000000003
```

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

In [22]:
```
sum(wines)
```

Out[22]:
```
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 [23]:
```
z=wines[:,:].sum(axis=1)
z
```

Out[23]:
```
array([ 74.5438 , 123.0548 ,  99.699  , ..., 100.48174, 105.21547,
        92.49249])
```

What is its size?

In [24]:
```
len(wines)
```

Out[24]:
```
1599
```

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

In [25]:
```
a=wines[:,3]
a=a.astype('int32')
a
```

Out[25]:
```
array([1, 2, 2, ..., 2, 2, 3])
```

find its maximum residual sugar value

In [26]:
```
max(a)
```

Out[26]:
```
15
```

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

In [27]:
```
min(a)
```

Out[27]:
```
0
```

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

In [28]:
```
np.mean(x)
```

Out[28]:
```
2.53880550343965
```

What is 25 percentile residual sugar value?

In [29]:
```
np.percentile(x,25)
```

Out[29]:
```
1.9
```

What is 75 percentile residual sugar value?

In [30]:
```
np.percentile(x,75)
```

Out[30]:
```
2.6
```

Find the average of each feature value

In [31]:
```
y=wines[:,:]
y
y.mean(axis=0)
```

Out[31]:
```
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 [32]:
```
wines[:,11]>5
```

Out[32]:
```
array([False, False, False, ...,  True, False,  True])
```

Show all wines with quality > 7

In [33]:
```
f=wines[:,11]>7
f
```

Out[33]:
```
array([False, False, False, ..., False, False, False])
```

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

In [34]:
```
True in f
```

Out[34]:
```
True
```

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

In [35]:
```
high_quality=[wines[:,11]>7][:3]
high_quality
```

Out[35]:
```
[array([False, False, False, ..., False, False, False])]
```

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

In [36]:
```
wines[high_quality][0:3]
```

```
C:\Users\1mscds08\AppData\Local\Temp\ipykernel_15692\4022932719.py:1: FutureWarn
ing: Using a non-tuple sequence for multidimensional indexing is deprecated; use
`arr[tuple(seq)]` instead of `arr[seq]`. In the future this will be interpreted
as an array index, `arr[np.array(seq)]`, which will result either in an error or
a different result.
  wines[high_quality][0:3]
```

Out[36]:
```
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 [37]:
```
b=(wines[:,10]>10)&(wines[:,11]>7)
b
```

Out[37]:
```
array([False, False, False, ..., False, False, False])
```

show only alcohol and wine quality columns

In [ ]:
```
wines[b,10:]
```

Combining NumPy Arrays

Combine red wine and white wine data

Open white wine dataset

In [39]:
```
white_wines = np.genfromtxt("winequality-white.csv", delimiter=";", skip_header=1
```

Show size of white_wines

In [40]:
```
white_wines.shape
```

Out[40]:
```
(4898, 12)
```

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

In [41]:
```
all_wines=np.vstack((wines,white_wines))
```

In [42]:
```
all_wines.shape
```

Out[42]:
```
(6497, 12)
```

Combine wines and white_wines data frames using concatenate method

In [43]:
```
all_wines1=np.concatenate((wines,white_wines),axis=0)
```

In [44]:
```
all_wines1.shape
```

Out[44]:
```
(6497, 12)
```

Matrix Operations and Reshape

Find Transpose of wines and print its size

In [45]:
```
tran=wines.T
tran.shape
```

Out[45]:
```
(12, 1599)
```

Convert wines data into 1D array

In [46]:
```
wines.ravel()
```

Out[46]:
```
array([ 7.4 ,  0.7 ,  0.  , ...,  0.66, 11.  ,  6.  ])
```

Reshape second row of wines into a 2-dimensional array with 2 rows an

In [47]:
```
wines[1].reshape((2,6))
```

Out[47]:
```
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 [48]:
```
sorted_alcohol=np.sort(wines[:,-2])
```

In [49]:
```
sorted_alcohol
```

Out[49]:
```
array([ 8.4,  8.4,  8.5, ..., 14. , 14. , 14.9])
```

Make sorting to take place in-place

In [50]:
```
wines[:,-2].sort()
```

In [51]:
```
wines[:,-2]
```

Out[51]:
```
array([ 8.4,  8.4,  8.5, ..., 14. , 14. , 14.9])
```

Show top 10 rows

In [52]:
```
wines[:,-2]
```

Out[52]:
```
array([ 8.4,  8.4,  8.5, ..., 14. , 14. , 14.9])
```

Sort alcohol column Descending Order

In [53]:
```
sorted_alcohol_desc=np.sort(wines[:,10])[::-1]
```

In [54]:
```
sorted_alcohol_desc
```

Out[54]:
```
array([14.9, 14. , 14. , ...,  8.5,  8.4,  8.4])
```

will original data be modified?. Check top 10 rows

In [55]:
```
wines[:,-2]
```

Out[55]:
```
array([ 8.4,  8.4,  8.5, ..., 14. , 14. , 14.9])
```

In [ ]:
```

```