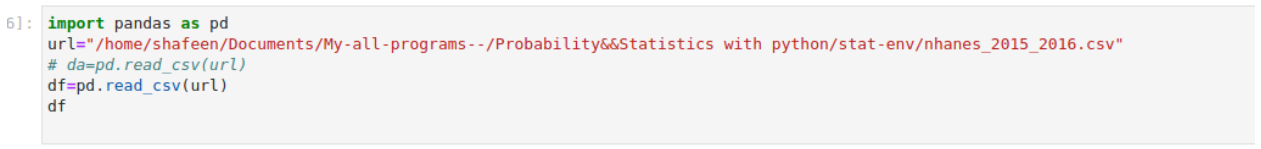
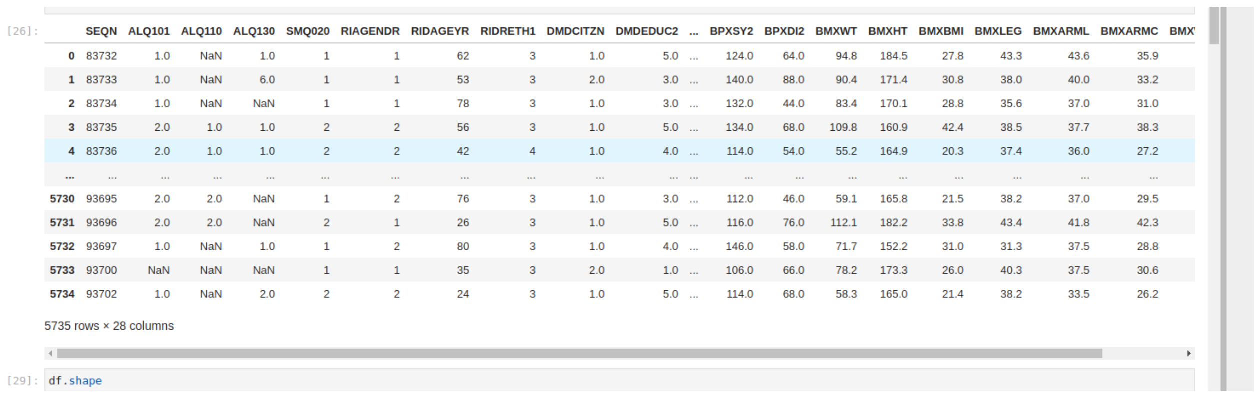
Probability and Statistics with PYTHON lecture 2

- Visualizing the data at first and how to deal with NAN values , after that finding the average
- First of all we use pandas for dealing with the dataframe
- Then we read the data using CSV



```
6]: import pandas as pd
    url="/home/shafeen/Documents/My-all-programs--/Probability&&Statistics with python/stat-env/nhanes_2015_2016.csv"
    # da=pd.read_csv(url)
    df=pd.read_csv(url)
    df
```

-
- After that we are gonna display the data frame



[26]:

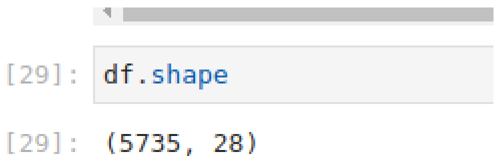
	SEQN	ALQ101	ALQ110	ALQ130	SMQ020	RIAGENDR	RIDAGEYR	RIDRETH1	DMDCITZN	DMDEDUC2	...	BPXSY2	BPXDI2	BMXWT	BMXHT	BMXBMI	BMXLEG	BMXARML	BMXARMC	BMX'
0	83732	1.0	NaN	1.0	1	1	62	3	1.0	5.0	...	124.0	64.0	94.8	184.5	27.8	43.3	43.6	35.9	
1	83733	1.0	NaN	6.0	1	1	53	3	2.0	3.0	...	140.0	88.0	90.4	171.4	30.8	38.0	40.0	33.2	
2	83734	1.0	NaN	NaN	1	1	78	3	1.0	3.0	...	132.0	44.0	83.4	170.1	28.8	35.6	37.0	31.0	
3	83735	2.0	1.0	1.0	2	2	56	3	1.0	5.0	...	134.0	68.0	109.8	160.9	42.4	38.5	37.7	38.3	
4	83736	2.0	1.0	1.0	2	2	42	4	1.0	4.0	...	114.0	54.0	55.2	164.9	20.3	37.4	36.0	27.2	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
5730	93695	2.0	2.0	NaN	1	2	76	3	1.0	3.0	...	112.0	46.0	59.1	165.8	21.5	38.2	37.0	29.5	
5731	93696	2.0	2.0	NaN	2	1	26	3	1.0	5.0	...	116.0	76.0	112.1	182.2	33.8	43.4	41.8	42.3	
5732	93697	1.0	NaN	1.0	1	2	80	3	1.0	4.0	...	146.0	58.0	71.7	152.2	31.0	31.3	37.5	28.8	
5733	93700	NaN	NaN	NaN	1	1	35	3	2.0	1.0	...	106.0	66.0	78.2	173.3	26.0	40.3	37.5	30.6	
5734	93702	1.0	NaN	2.0	2	2	24	3	1.0	5.0	...	114.0	68.0	58.3	165.0	21.4	38.2	33.5	26.2	

5735 rows × 28 columns

```
[29]: df.shape
```

-

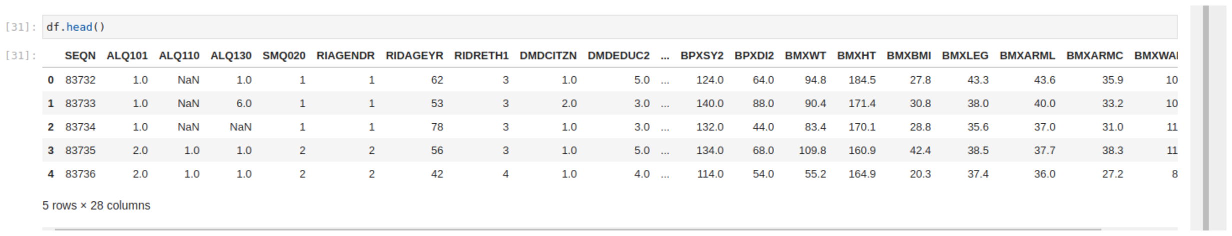
- To know how many rows and columns are there / how much is the data spread
- We use “ ***df.shape*** “



```
[29]: df.shape

[29]: (5735, 28)
```

-

- For looking at the first five rows inlcuding (0) we use “ ***df.head*** “



```
[31]: df.head()
```

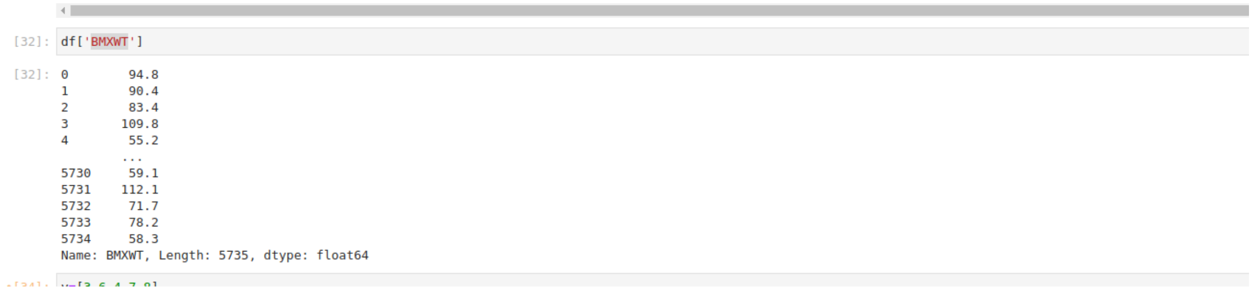
[31]:

	SEQN	ALQ101	ALQ110	ALQ130	SMQ020	RIAGENDR	RIDAGEYR	RIDRETH1	DMDCITZN	DMDEDUC2	...	BPXSY2	BPXDI2	BMXWT	BMXHT	BMXBMI	BMXLEG	BMXARML	BMXARMC	BMXWAI
0	83732	1.0	NaN	1.0	1	1	62	3	1.0	5.0	...	124.0	64.0	94.8	184.5	27.8	43.3	43.6	35.9	10
1	83733	1.0	NaN	6.0	1	1	53	3	2.0	3.0	...	140.0	88.0	90.4	171.4	30.8	38.0	40.0	33.2	10
2	83734	1.0	NaN	NaN	1	1	78	3	1.0	3.0	...	132.0	44.0	83.4	170.1	28.8	35.6	37.0	31.0	11
3	83735	2.0	1.0	1.0	2	2	56	3	1.0	5.0	...	134.0	68.0	109.8	160.9	42.4	38.5	37.7	38.3	11
4	83736	2.0	1.0	1.0	2	2	42	4	1.0	4.0	...	114.0	54.0	55.2	164.9	20.3	37.4	36.0	27.2	8

5 rows × 28 columns

-

- Here we wanted to find the average WEIGHT from the data set so we display the weight column with



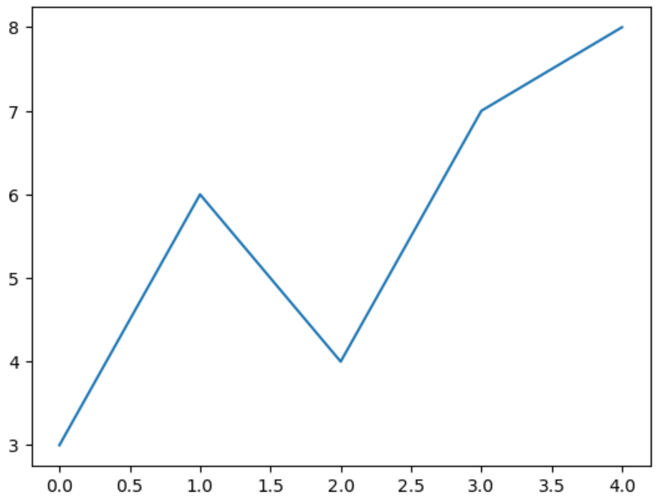
```
[32]: df['BMXWT']

[32]: 0        94.8
      1        90.4
      2        83.4
      3       109.8
      4        55.2
               ...
      5730     59.1
      5731    112.1
      5732     71.7
      5733     78.2
      5734     58.3
      Name: BMXWT, Length: 5735, dtype: float64
```

- DATA VISUALIZATION

    - We are gonna use dummy values to understand how to visualize our data with MATPLOTLIB
    - First we import the library then we import the directory **PYPLOT** from it and we use " **%matplotlib inline** " to plot the graph inline in the jupyter lab
    - We use the plot function and give it our list
    - Then use the show function ot plot it

```
[34]: y=[3,6,4,7,8]
      import matplotlib
      import matplotlib.pyplot as plt
      %matplotlib inline
```

```
[36]: plt.plot(y)
      plt.show()
```

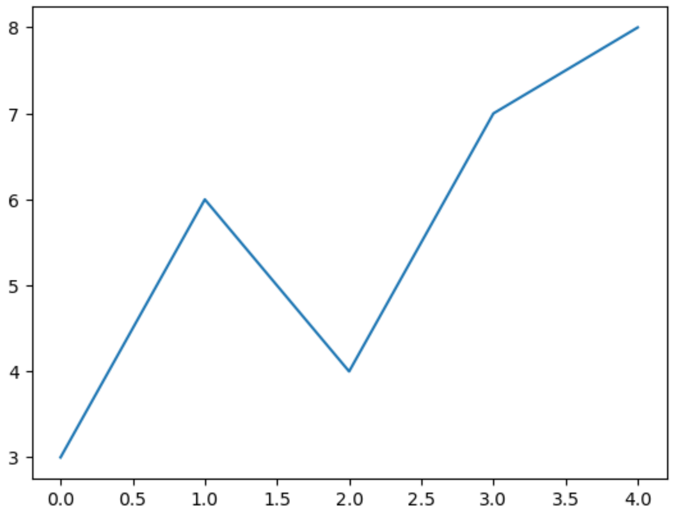


```
•[38]: #create a figure
       plt.figure(figsize=(10,5))
```

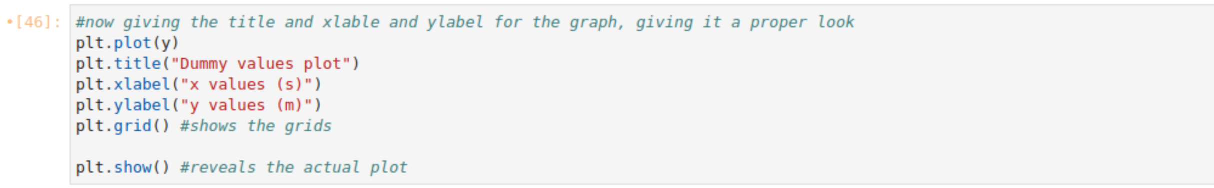
```
[38]: <Figure size 1000x500 with 0 Axes>
      <Figure size 1000x500 with 0 Axes>
```

```
•[41]: #simple plot with no modification and proper labels
       plt.plot(y)
```

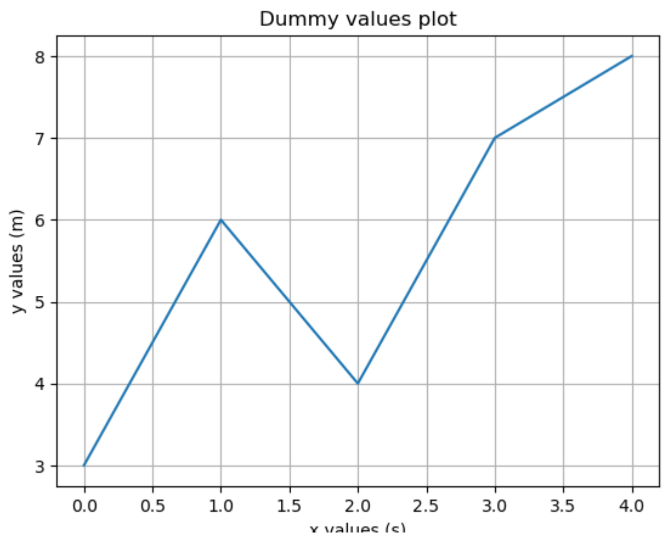
```
[41]: [<matplotlib.lines.Line2D at 0x7fef2f06a010>]
```



- As we can see the above plot is without any labels and not so understanding we use more function fo matplotlib to properly visualize it
- We give it a title , x-label , y-label , we also give it a grid for more better understanding

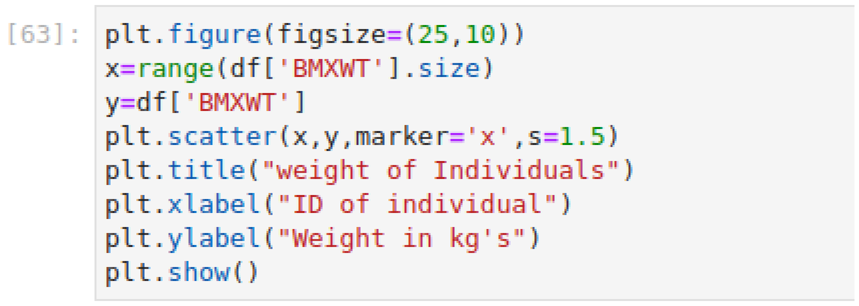


```
•[46]: #now giving the title and xlable and ylabel for the graph, giving it a proper look
       plt.plot(y)
       plt.title("Dummy values plot")
       plt.xlabel("x values (s)")
       plt.ylabel("y values (m)")
       plt.grid() #shows the grids

       plt.show() #reveals the actual plot
```

- 

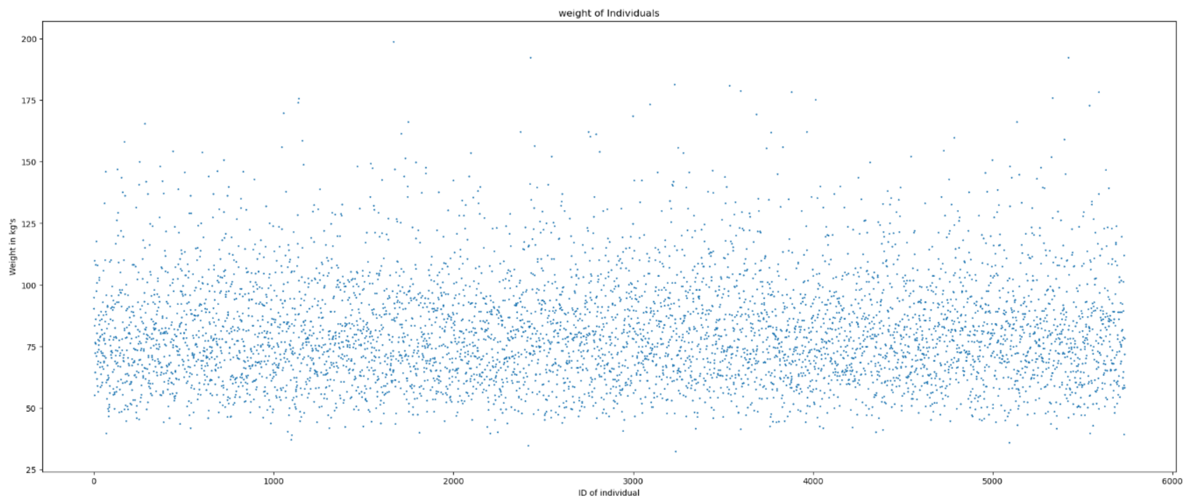
- Now as you can see this plot is more understandable than the previous one

- NOW WE USE SIMILAR LEARNING TO PLOT THE WEIGHT OF ALL THE INDIVIDUALS IN A SCATTER PLOT STYLE BECASUE WE DONOT HAVE TO RELATE ONE DATA WITH ANOTHER WHICH IS LINE WE ARE NOT USING THE ABOVE PLOT x values (s)



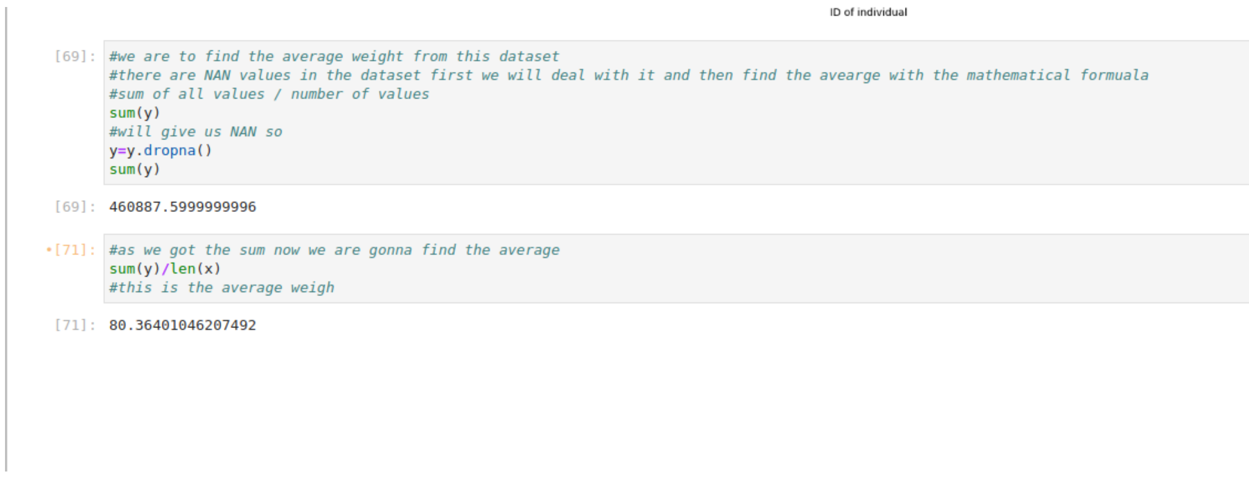
```
[63]: plt.figure(figsize=(25,10))
      x=range(df['BMXWT'].size)
      y=df['BMXWT']
      plt.scatter(x,y,marker='x',s=1.5)
      plt.title("weight of Individuals")
      plt.xlabel("ID of individual")
      plt.ylabel("Weight in kg's")
      plt.show()
```

-



-

- NOW WE FIND THE AVERAGE



```
[69]: #we are to find the average weight from this dataset
      #there are NAN values in the dataset first we will deal with it and then find the avaerge with the mathematical formuala
      #sum of all values / number of values
      sum(y)
      #will give us NAN so
      y=y.dropna()
      sum(y)

[69]: 460887.5999999996

•[71]: #as we got the sum now we are gonna find the average
       sum(y)/len(x)
       #this is the average weigh

[71]: 80.36401046207492
```

-
- Simple
-