

Probability and Statistics with PYTHON lecture 5

- **Central Tendency**

- Studying different averages
- First we plot the graph of weight to find a visual average

```
[ ]: plt.figure(figsize=(20, 5))

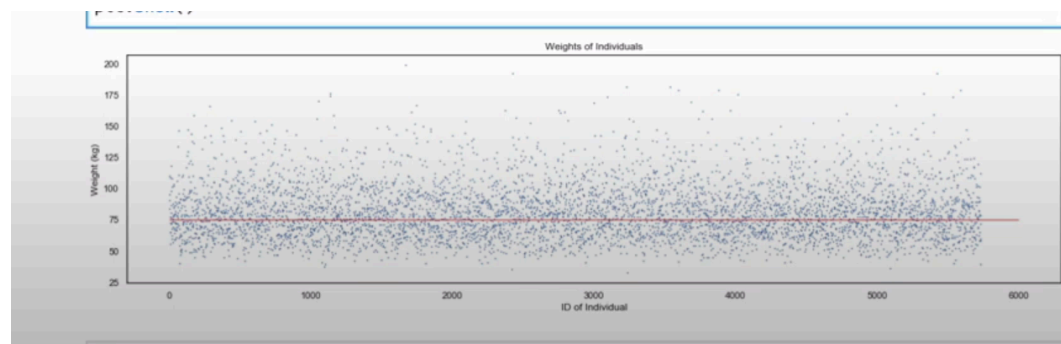
x = range(da['BMXWT'].size)
y = da['BMXWT']

plt.scatter(x, y, marker='x', s=0.5)

plt.title("Weights of Individuals")
plt.xlabel("ID of Individual")
plt.ylabel("Weight (kg)")

# plot the estimate line
p1 = (0, 75)
p2 = (6000, 75)
x_c, y_c = zip(p1, p2)
plt.plot(x_c, y_c, color='red', linewidth=0.75)
plt.show()
```

○



○

- It is the same graph we drew in last lecture just the red line added for visualizing the average on the graph

- **Plotting the mean**

- Now we are gonna plot two lines to show the mean and the average (the one we visualized before)

```
[ ]: plt.figure(figsize=(20, 5))

x = range(da['BMXWT'].size)
y = da['BMXWT']

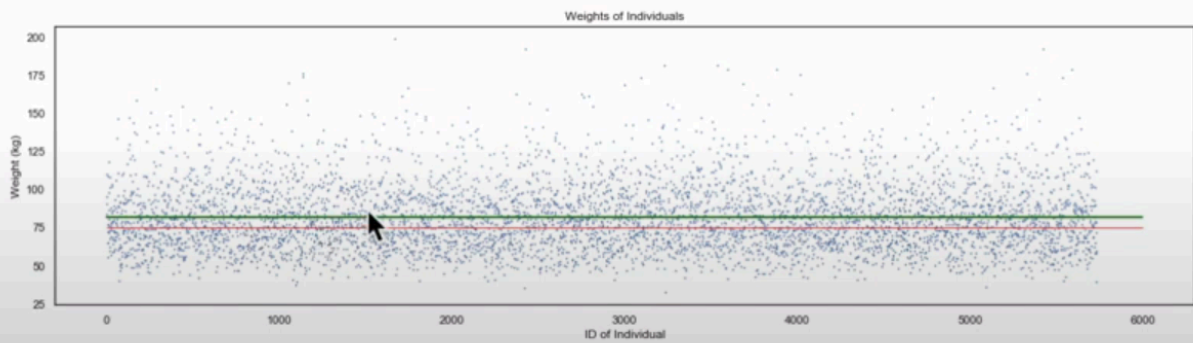
plt.scatter(x, y, marker='x', s=0.5)

plt.title("Weights of Individuals")
plt.xlabel("ID of Individual")
plt.ylabel("Weight (kg)")

# plot the estimate line
x_c, y_c = ([0, 6000], [75, 75])
plt.plot(x_c, y_c, color='red', linewidth=0.75)

# plot the estimate line
x_c, y_c = ([0, 6000], [wt.mean(), wt.mean()])
plt.plot(x_c, y_c, color='green', linewidth=2)

plt.show()
```



Problem with Arithmetic Mean

- Now as you can see the actual mean is in green line and the average we found out was the red line
- Why is there a difference in the average we visualize and the mean that the system finds out?
 - Problem with Arithmetic Mean

- Now we calculate mean of some random data points we get

```
Problem with Arithmetic Mean

[41]: import numpy as np

[42]: nums = np.array([3, 4, 9, 17, 18, 20, 22, 23, 29, 32, 39, 42, 42, 42, 51, 97])

[43]: np.mean(nums)

[43]: 30.625

[ ]: # let's try to do a different kind of average: the median!

Visualizing Weighted Data
```

○

- Now we are gonna change only one value and see if it makes any changes

```
Problem with Arithmetic Mean

[41]: import numpy as np

[44]: nums = np.array([3, 4, 9, 17, 18, 20, 22, 23, 29, 32, 39, 42, 42, 42, 51, 197])

[45]: np.mean(nums)

[45]: 36.875

[ ]: # let's try to do a different kind of average: the median!

Visualizing Weighted Data
```

○

○ And it did

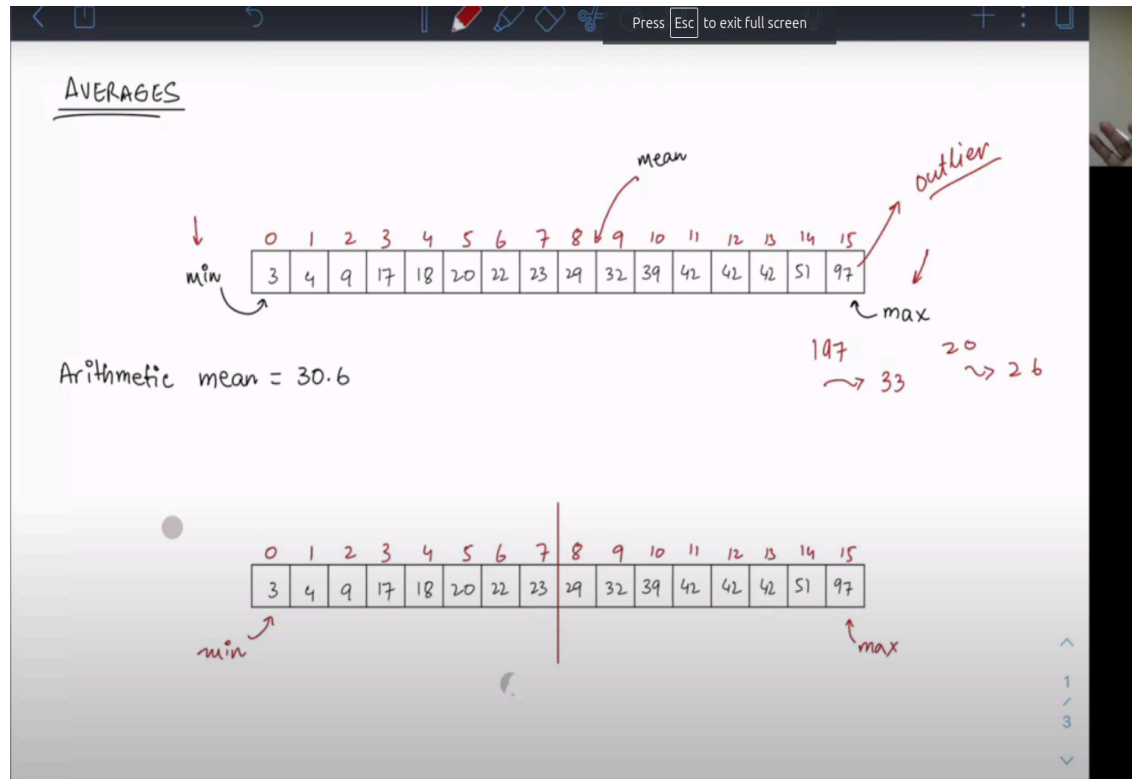
- Conclusion : THE PROBLEM WITH ARITHMETIC MEAN IS THAT IT IS GREATLY AFFECTED BY AN OUTLIER

- Outlier : An outlier is a data point in a data set that is unique from all the other data points or differs greatly from all data points

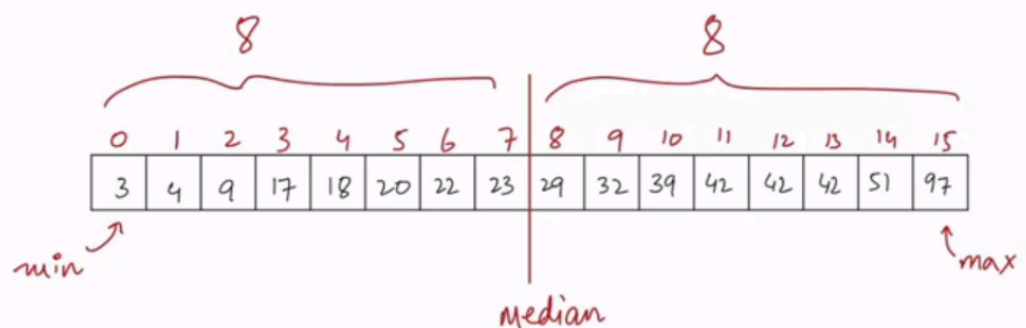
○

- Understanding the problem :

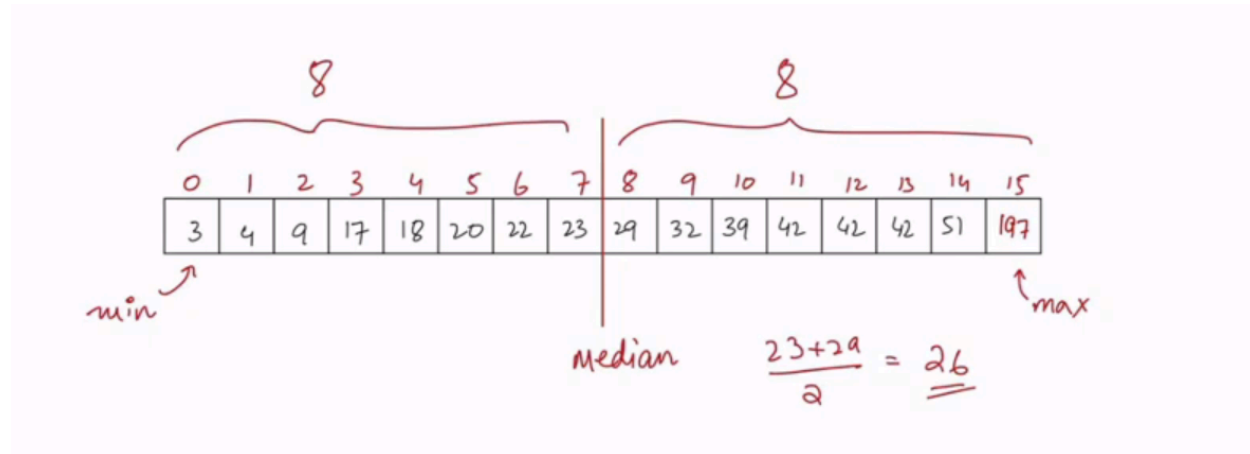
- This is arithmetic mean



- As we discussed arithmetic mean gives us problem with calculating mean as it is easily disturbed by an outlier just like ADHD patients that lose focus with minor distraction
- So here is what we are gonna do to fix it

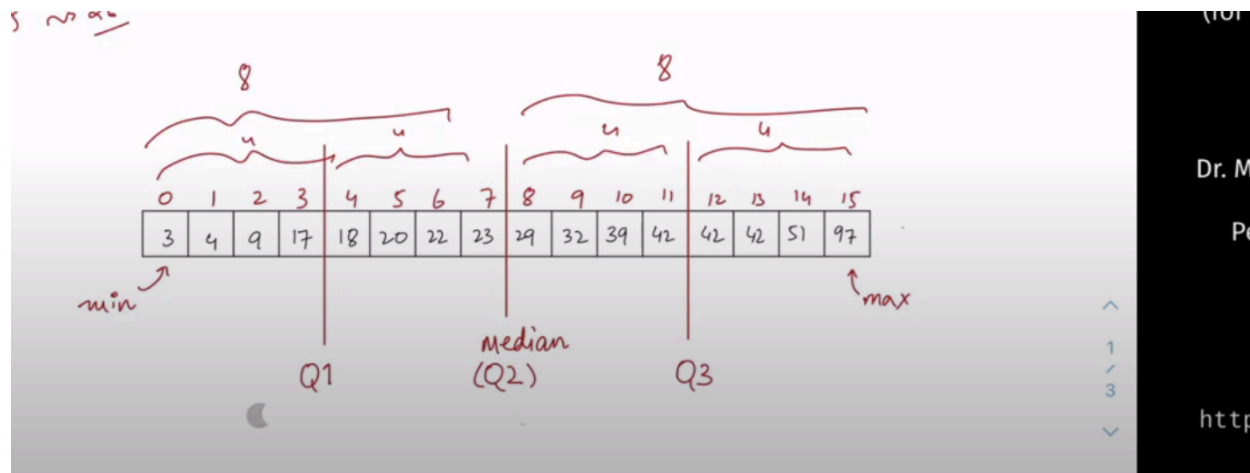


- We are gonna split the data set in two and call it MEDIAN
- The median here is 26
 - We add the value to the left of the slice and the value to the right of the slice and divide it by two to find the median.... (batch mein cut kai left right wali value add karein aur divide by 2 kardein)
- Now change the last value to 197 (an outlier)



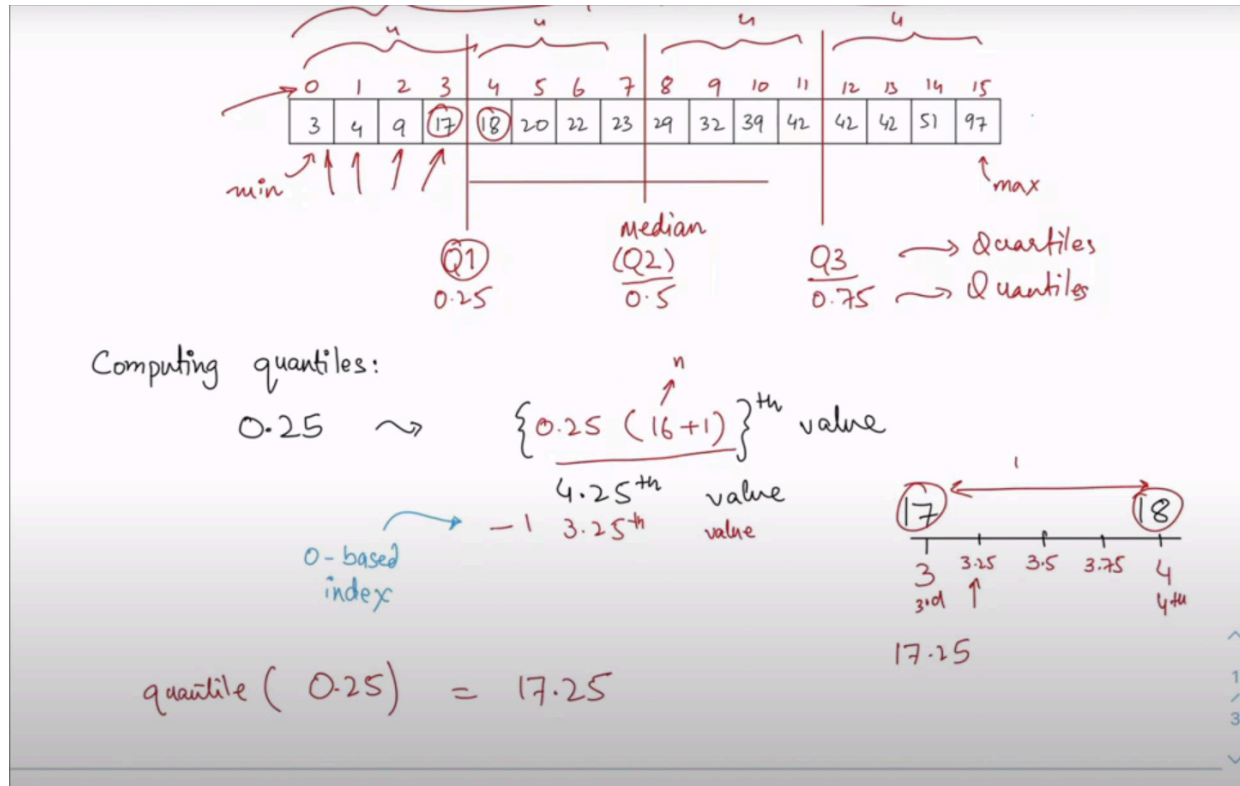
- As you can see there is no effect on the median which means we are dealing with the outliers
- **MEDIANS ARE ROBUST TO OUTLIERS**

- **Three types of averages**
 - The arithmetic mean
 - Mode
 - Median
- **Now taking another step forward**



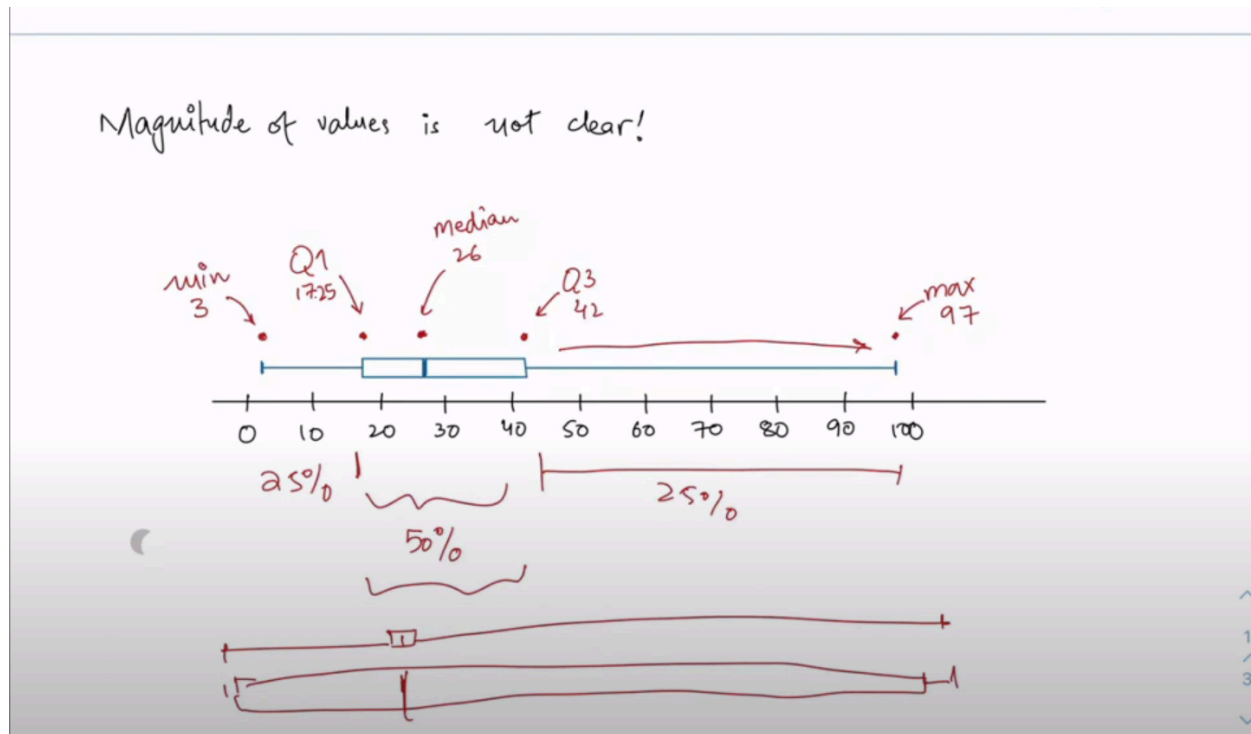
- **Now what we did was that we splitted the rest of the data into two as well and gave the first slice a name of Q1 and the second slice a name of Q2 (MEDIAN) and the third slice a name of Q3**
- **Quartile - 1 , 2 & 3**
- **So now we know if say what is the median we can say this middle point is the median**

- If someone say what is q1 we can say the first slice which means 25% people are below q2 the rest are above or greater vice versa
- Quartile 1 - 0.25 known as QUANTILES
- Quartile 2 - 0.50 known as QUANTILES
- Quartile 3 - 0.75 known as QUANTILES



- Now to find the quantile we use the formule
 - $\{ 0.25 (n + 1) \}$ th value
 - It means we are gonna take whatever the answer is “ th “ value
 - In this case the value is 4.25 th value now we can only take either 4th value or fifth value so what we do is we create number line for the both 4th and fifth value
 - But as we are using 0-based index we subtract one so the final value is 3.25th value
 - After that we divide it in further q's and take the first value i.e 17 and add 0.25 decimal with it so
 - The first quantile is 17.25
 - Some libraries will give us different values but it is fine for-example some libraries here would give 17.5 some would give 17.25 so both are fine
 - Now this is fine for small data set but if the data is too large we wont be able to compute and remember instead we are gonna do this

- **Box Plots**



-
- Now to understand our median properly and our data we use this number line the maximum range our dataset can take
- Then we plot all the information we extracted the median Q1 Q2 Q3
- Then we make a box from q1 to q3 covering the median
- This separates all the other data
- Now our 50% data will always lie in the box the rest 25% and 25% will lie else wise
- Now by looking at this plot we can easily say that 97 is an outlier because all the data similar data that is close to the median is q1 and q3 but the rest is just outliers biased data
- The 50% means half of the people are there while the rest are in 25%



-
- In such a box plot we say most people are closer to the median while some people are far



-
- But if the box plot is like this then we compute that the data is quite spread and have many variations and are not similar but unique

-

- **CONCLUSION :**

We have three types of averages as we discussed , average is nothing but central tendency meaning how many data points are near the center

Arithmetic mean (not robust to outlier)

Mode (most common) (used in some cases)

Median (robust ot outlier)

Related to median we saw quantiles

Further taken to percentile , 100 pieces , a specific percentile mean that the specific area is greater than the percentile it is in , forexample if data point is in 80th percentile then it means you are better than the rest 80%

- The rest is just the code for it


```
1 Click here to ask Blackbox to help you code faster
2 # understanding mean
3 import numpy as np
4 x=np.array([3,4,9,17,18,20,22,23,29,32,39,42,42,42,51,97])
5 np.mean(x)

[59] ✓ 0.0s Python
... 30.625

1 Click here to ask Blackbox to help you code faster
2 # different kind of average which is called the median

[60] ✓ 0.0s Python
...

1 Click here to ask Blackbox to help you code faster
2 # finding the median to prove that outlier does not affect it
3 np.median(x)
4 #median is same even if we change values in the array in the start or in the end

[61] ✓ 0.0s Python
... 26.0

1 Click here to ask Blackbox to help you code faster
2 # visualizing the data set weights

[62] ✓ 0.0s Python
...

1 Click here to ask Blackbox to help you code faster
2 wgt.min()

[63] ✓ 0.0s Python
... 32.4

1 Click here to ask Blackbox to help you code faster
2 wgt.mean()

[64] ✓ 0.0s Python
... 81.34267560889516
```

Code Comment Code Chat Ln 1, Col 13 Spaces: 4 LF Cell 21 of 21 Go Live Blackbox

Quartiles and Boxplots

```
1 Click here to ask Blackbox to help you code faster
2 num=np.array([3, 4, 9, 17, 18, 20, 22, 23, 29, 32, 39, 42, 42, 42, 51, 97])
3 np.median(num)

[66] ✓ 0.0s Python
... 26.0

1 Click here to ask Blackbox to help you code faster
2 import statistics "/home/shafeen/Documents/My-all-programs--/Probability&Statistics with python/stat-env/lib/python3.11/statistics.py" is overriding the stdlib modul
3 statistics.quantiles(num,n=4)

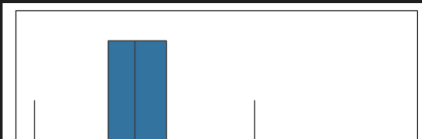
[67] ✓ 0.0s Python
... [17.25, 26.0, 42.0]

1 Click here to ask Blackbox to help you code faster
2 statistics.quantiles(wgt,n=4)

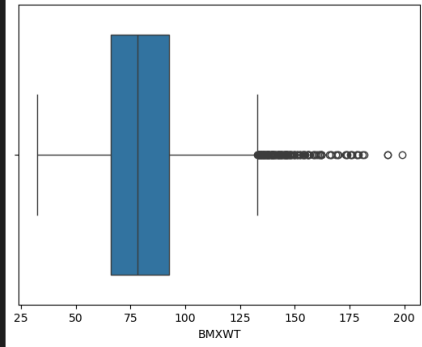
[68] ✓ 0.0s Python
... [43.7, 52.5, 50.1]

1 Click here to ask Blackbox to help you code faster
2 # %pip install seaborn
3 import seaborn as sns
4 ax=sns.boxplot(x=wgt)

[71] ✓ 0.0s Python
... 
```



Code Comment Code Chat Ln 1, Col 13 Spaces: 4 LF Cell 21 of 21 Go Live Blackbox



```

Click here to ask Blackbox to help you code faster |
1 male_wgt=df.loc[df['RIAGENDR']==2,'BMXWT']
2 male_wgt
[80] ✓ 0.0s Python
...
3      189.8
4      55.2
5      64.4
7      64.5
12     71.2
...
5724    58.2
5727    64.8
5730    59.1
5732    71.7
5734    58.3
Name: BMXWT, Length: 2976, dtype: float64
Code Comment Code Chat Ln 1, Col 13 Spaces: 4 LF Cell 21 of 21 Go Live Blackbox

```

```

Click here to ask Blackbox to help you code faster |
1 female_wgt=df.loc[df['RIAGENDR']==1,'BMXWT']
2 female_wgt
[81] ✓ 0.0s Python
...
0      94.8
1      90.4
2      83.4
6      76.6
8      72.4
...
5726    78.7
5728    89.5
5729    39.2
5731   112.1
5733    78.2
Name: BMXWT, Length: 2759, dtype: float64

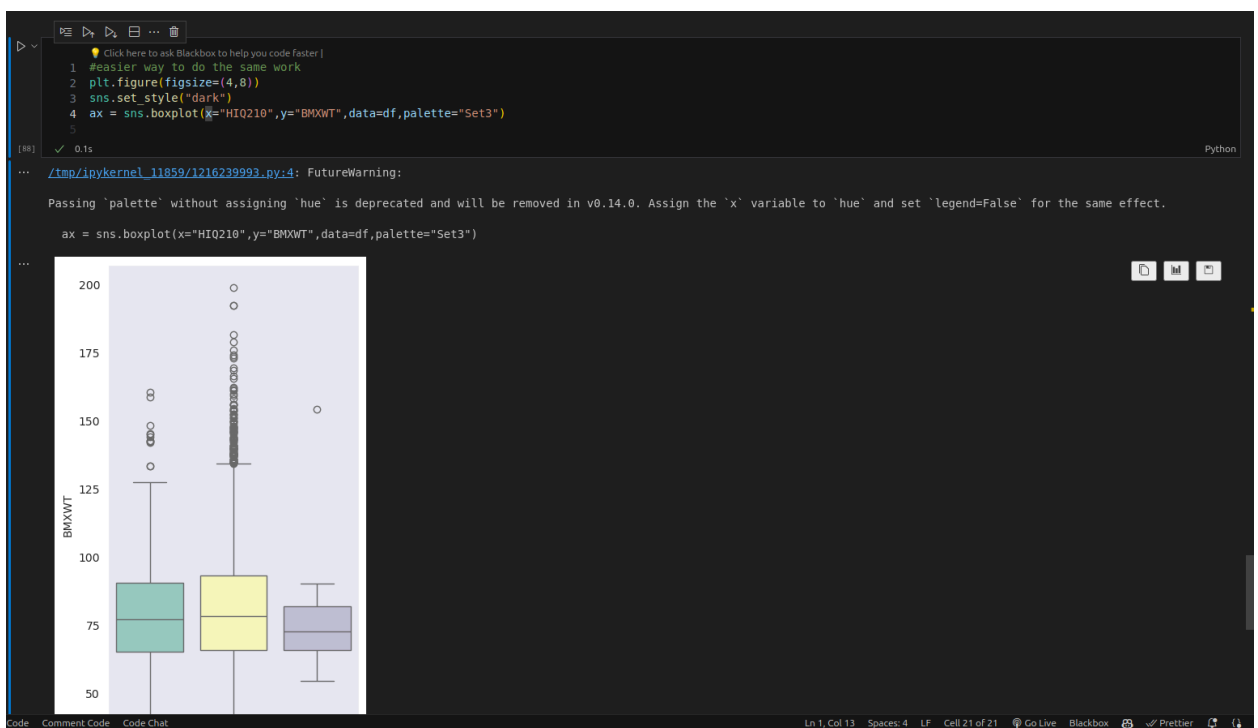
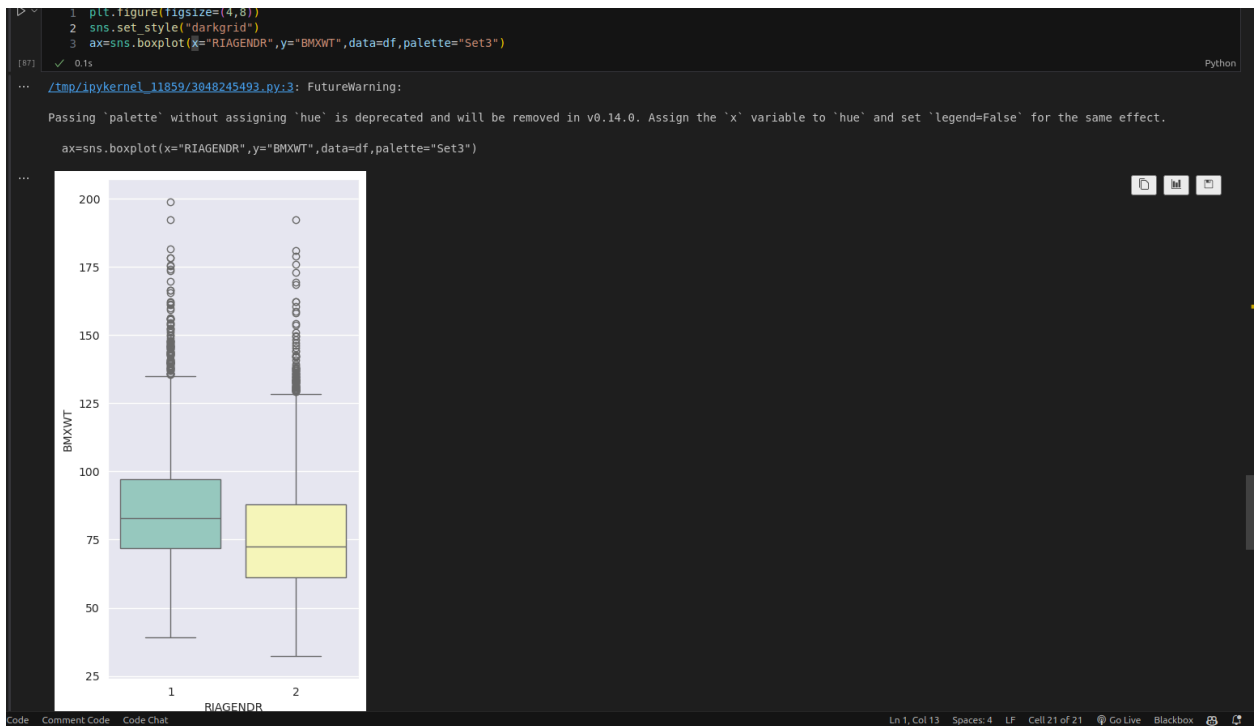
Click here to ask Blackbox to help you code faster |
1 plt.figure(figsize=(4,8))
2 sns.set_style("darkgrid")
3 ax=sns.boxplot(x="RIAGENDR",y="BMXWT",data=df,palette="Set3")
[87] ✓ 0.1s Python

/tmp/ipykernel_11859/3048245493.py:3: FutureWarning:
Passing 'palette' without assigning 'hue' is deprecated and will be removed in v0.14.0. Assign the 'x' variable to 'hue' and set 'legend=False' for the same effect.

ax=sns.boxplot(x="RIAGENDR",y="BMXWT",data=df,palette="Set3")
...

```





- This is all the code for it