Outlier

An outlier is a dat point in a data set that is distant from all other obeservation. A data point that lies outside the overall distribution of the dataset.

Criteria to identify an outlier

1) Data point that falls outside of 1.5 times f an interquantile range above te 3rd quartile and below the 1st quartile.

2) Data pont that falls outside of 3 standard deviations , we can use a z-score and if the z-score falls otuside of the 3 standard deviation.

Reason for outlier to exist in dataset

1) Variablitiy in the data

2) An Experimental measurement error

Wha are the impact of having outlier ?

1) It causes various problems during our statistical analysis.

2) It may cause a signigicant impact on the mean and the standard deviation.

various ways of finding the outliers.

1) Using scatter plots. 2) box plot 3) using z-score 4) Using the IQR interquantile range

'Mean' is the only measure of central tendency that is affected by the outliers which in turn impacts Standard deviation. outliers will note affect medain and mode much

Handling Outliers

Below are some of the methods of treating the outliers

Trimming/removing the outlier--in this technique, we remove the outliers from the dataset. Although it is not a good practice to follow ,dropping outlier is always a harsh step and should be taken only in extreme conditions when we're very sure that the outlier is due to a measurement error , ex - in human data where age is more 200 or weight is more 300 which is not possible.

Quantile based flooring and capping- we can replace outliers with upper and lower bound.

Replacing with Mean/Median/mode -As the mean value is highly influenced by the outliers, it is advised to replace the outliers with the median value

catagorical values can be replaced by mode

In [1]:
```
import numpy as np
from matplotlib import pyplot as plt
import seaborn as sns
import pandas as pd
```

In [51]:
```
d=[11,12,13,14,16,18,10,13,12,19,12,14,15,20,101,106,12,10]
```

Detecting using Z-score.

z= (x-mean)/std

In [3]:
```
def detect_outliers(data):
    quartile_1, quartile_3 = np.percentile(data, [25, 75])
    iqr = quartile_3 - quartile_1
    lower_bound = quartile_1 - (iqr * 1.5)
    upper_bound = quartile_3 + (iqr * 1.5)
    return np.where((data > upper_bound) | (data < lower_bound))
```

In [52]:
```
detect_outliers(d)

#14 and and 15 indexes are outlier which is 101 ,106
```

Out[52]:
```
(array([14, 15], dtype=int64),)
```

In [53]:
```
print(d[14])
print(d[15])
```

```
101
106
```

interquantile range

75%-25%

steps

1) Arrange data in assending order

2) calcuate first(q1) and third quartile(q3)

3) Find interquantile range(q3-q1)

4) Find lower bound q1*1.5.

5) Find upper bound q3*1.5.

anything lies outside of lower and upper bound is and outliers.

In [28]:
```
d
```

Out[28]:
```
[11, 12, 13, 14, 16, 18, 10, 13, 12, 19, 12, 14, 15, 20, 101, 106, 12, 10]
```

In [29]:
```
sorted(d)
```

Out[29]:
```
[10, 10, 11, 12, 12, 12, 12, 13, 13, 14, 14, 15, 16, 18, 19, 20, 101, 106]
```

In [30]:
```
#finding q1 and q3 using numpy inbuild function called percentile

q1 , q3= np.percentile(d,[25,75])
print(q1,q3)
```

```
12.0 17.5
```

In [31]:
```
#IQR
IQR=q3-q1
print(IQR)
```

```
5.5
```

In [32]:
```
#finding lower and upper bound

lower_bound = q1-(1.5*IQR)
upper_bound= q3+(1.5*IQR)
print(lower_bound,upper_bound)
```

```
3.75 25.75
```

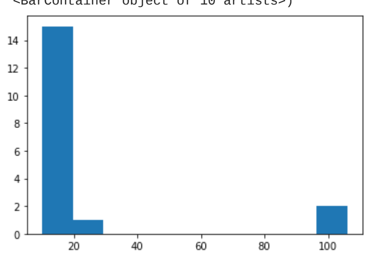
In [33]:
```
for i in d:
    if i<3.25 or i>25.75:
        print(i)
```

```
101
106
```

101 and 106 are outliers

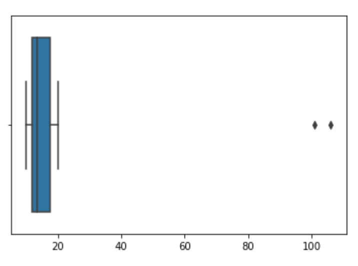
In [38]:
```
plt.hist(d)
```

Out[38]:
```
(array([15.,  1.,  0.,  0.,  0.,  0.,  0.,  0.,  0.,  2.]),
 array([ 10. ,  19.6,  29.2,  38.8,  48.4,  58. ,  67.6,  77.2,  86.8,
         96.4, 106. ]),
 <BarContainer object of 10 artists>)
```



In [51]:
```
sns.boxplot(d)
```

Out[51]:
```
<AxesSubplot:>
```



In [18]:
```
#droping outliers

# we have remove 101 and 106 from list

d.remove(106)
```

In [19]:
```
d
```

Out[19]:
```
[11, 12, 13, 14, 16, 18, 10, 13, 12, 19, 12, 14, 15, 20, 12, 10]
```

working with iris dataset

In [2]:
```
df=pd.read_csv('Iris.csv')
df
```

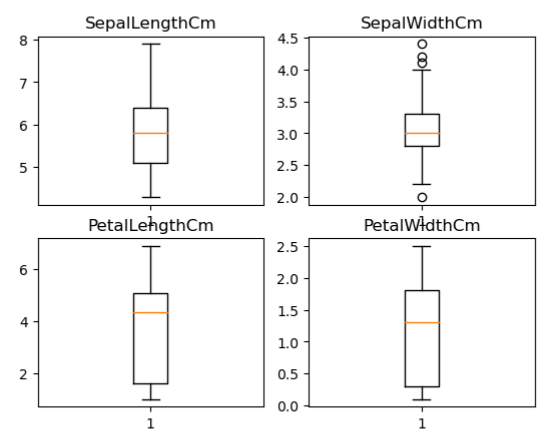
Out[2]:

	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Species
0	1	5.1	3.5	1.4	0.2	Iris-setosa
1	2	4.9	3.0	1.4	0.2	Iris-setosa
2	3	4.7	3.2	1.3	0.2	Iris-setosa
3	4	4.6	3.1	1.5	0.2	Iris-setosa
4	5	5.0	3.6	1.4	0.2	Iris-setosa
...	...	...	...	...	...	...
145	146	6.7	3.0	5.2	2.3	Iris-virginica
146	147	6.3	2.5	5.0	1.9	Iris-virginica
147	148	6.5	3.0	5.2	2.0	Iris-virginica
148	149	6.2	3.4	5.4	2.3	Iris-virginica
149	150	5.9	3.0	5.1	1.8	Iris-virginica

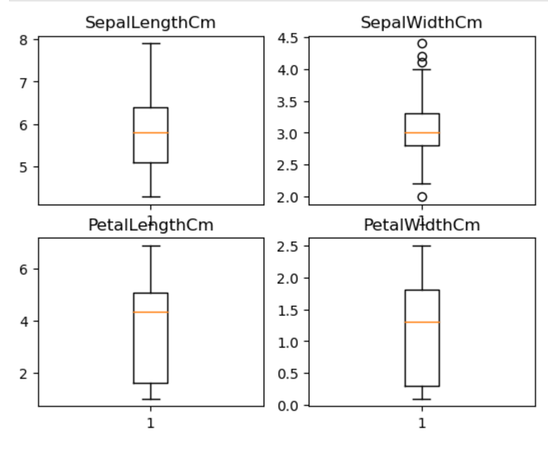
150 rows × 6 columns

In [3]:
```
fig,ax =plt.subplots(2,2)
ax[0,0].boxplot(x='SepalLengthCm' ,data= df)
ax[0,0].set_title('SepalLengthCm')
ax[0,1].boxplot(x='SepalWidthCm' ,data= df)
ax[0,1].set_title('SepalWidthCm')
ax[1,0].boxplot(x='PetalLengthCm' ,data= df)
ax[1,0].set_title('PetalLengthCm')
ax[1,1].boxplot(x='PetalWidthCm' ,data= df)
ax[1,1].set_title('PetalWidthCm')
```

Out[3]:
```
Text(0.5, 1.0, 'PetalWidthCm')
```



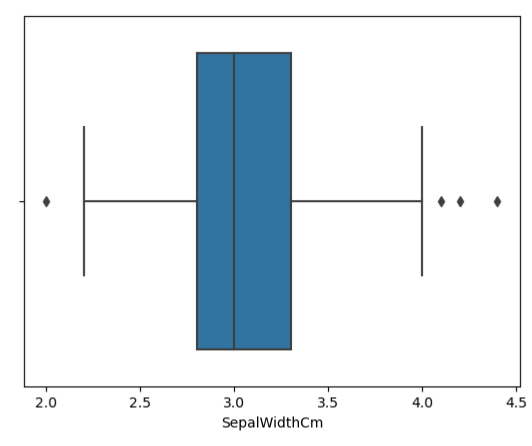
In [9]:
```
plt.subplot(2,2,1)
plt.boxplot('SepalLengthCm' ,data= df)
plt.title('SepalLengthCm')

plt.subplot(2,2,2)
plt.boxplot('SepalWidthCm' ,data= df)
plt.title('SepalWidthCm')

plt.subplot(2,2,3)
plt.boxplot('PetalLengthCm' ,data= df)
plt.title('PetalLengthCm')

plt.subplot(2,2,4)
plt.boxplot('PetalWidthCm' ,data= df)
plt.title('PetalWidthCm')

plt.show()
```



from here we can see column sepal width has outliers

In [16]:
```
sns.boxplot(x='SepalWidthCm' , data=df )
```

Out[16]:
```
<AxesSubplot:xlabel='SepalWidthCm'>
```



In [17]:
```
q1 , q3= np.percentile(df.SepalWidthCm,[25,75])
print(q1,q3)
```

```
2.8 3.3
```

In [18]:
```
#second method
q1=df.SepalWidthCm.quantile(0.25)
q3=df.SepalWidthCm.quantile(0.75)
print(q1,q3)
```

```
2.8 3.3
```

In [19]:
```
iqr=q3-q1
iqr
```

Out[19]:
```
0.5
```

First method -Capping(using qualtile based flooring method)

Here we will replace outliers with upper and lower_bound

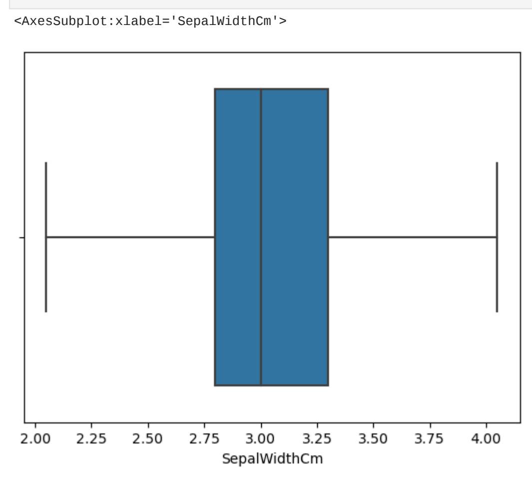
In [20]:
```
lower_bound = q1-(1.5*iqr)
upper_bound= q3+(1.5*iqr)
print(lower_bound,upper_bound)
```

```
2.05 4.05
```

In [21]:
```
df['SepalWidthCm']=df['SepalWidthCm'].apply(lambda x:lower_bound if x < lower_bound else x)

df['SepalWidthCm']=df['SepalWidthCm'].apply(lambda x:upper_bound if x > upper_bound else x)
```

In [22]:
```
sns.boxplot(x='SepalWidthCm' , data=df)
```

Out[22]:
```
<AxesSubplot:xlabel='SepalWidthCm'>
```



we have replace outliers with upper_bound and lower bound

Second method - Trimming(Removing outliers)

In [27]:
```
df_new=df[(df.SepalWidthCm > 2.05) & (df.SepalWidthCm <4.05)]
df_new
```

Out[27]:

	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Species
0	1	5.1	3.5	1.4	0.2	Iris-setosa
1	2	4.9	3.0	1.4	0.2	Iris-setosa
2	3	4.7	3.2	1.3	0.2	Iris-setosa
3	4	4.6	3.1	1.5	0.2	Iris-setosa
4	5	5.0	3.6	1.4	0.2	Iris-setosa
...	...	...	...	...	...	...
145	146	6.7	3.0	5.2	2.3	Iris-virginica
146	147	6.3	2.5	5.0	1.9	Iris-virginica
147	148	6.5	3.0	5.2	2.0	Iris-virginica
148	149	6.2	3.4	5.4	2.3	Iris-virginica
149	150	5.9	3.0	5.1	1.8	Iris-virginica

146 rows × 6 columns

In [28]:
```
print('row in orignal data:', len(df))
print('row in new data:', len(df_new))
print('Difference:', len(df)-len(df_new))
```

```
row in orignal data: 150
row in new data: 146
Difference: 4
```

We can clearely see now we had 4 rows with outliers and we have succesfully remove it

conclusion:

Don't drop an outlier if: -When your results are critical, then even minor changes will matter a lot.

Valuing the outliers: If there is a valid reason for the outlier to exist and it is a part of our natural process, we should investigate the cause of the outlier as it can provide valuable clues that can help you better understand your process performance. Outliers may be hiding precious information that could be invaluable to improve your process performance. You need to take the time to understand the special causes that contributed to these outliers. Fixing these special causes can give you significant boost in your process performance and improve customer satisfaction. For example, normal delivery of orders takes 1-2 days, but a few orders took more than a month to complete. Understanding the reason why it took a month and fixing this process can help future customers as they would not be impacted by such large wait times.

ex2.Let's have a use case of credit card fraud detection, outlier analysis becomes important because here, the exception rather than the rule may be of interest to the analyst.

In [ ]: