

<b>Project Name</b>	Global Sales Data Analytics
<b>Name</b>	GOMATHI.P
<b>Roll No</b>	620619104004
<b>Team ID</b>	PNT2022TMID41454

## Assignment -4

### Import Libraries

```
import pandas as pd
import
matplotlib.pyplot as
pltimport numpy as
np
import seaborn as sns
from sklearn.model_selection import
train_test_splitfrom
sklearn.preprocessing import
StandardScaler
from sklearn.linear_model import
LinearRegressionfrom sklearn.tree
import DecisionTreeRegressor
```

### Import Dataset

```
data =
pd.read_csv('abalone.c
sv')data
```



	w e i g h t	w e i g h t		
--	----------------------------	----------------------------	--	--

					weight		
				0 . 0 9 5 . 0 . 5 1 4 0	0 . 2 2 4 5  0 . 1 0 1 0		
				0 . 0 9 0 . 0 . 2 2 5 5	0 . 0 9 9 5  0 . 0 4 8 5		
				0 . 1 3 5 . 0 . 6 7 7 0	0 . 2 5 6 5  0 . 1 4 1 5		
				0 . 1 2 5	0 . 2 1 5 5		

			0 . 5 1 6 0 0	0 . 1 1 4 0		
			0 . 0 8 0 0 . 2 0 5 0	0 . 0 8 9 5 0 . 0 3 9 5		
...			. . . . . . . . .	. . . . . . . . .		
4172			0 . 1 6 5 0 . 8 8 7 0	0 . 3 7 0 0 0 . 2 3 9 0		
4173			0 . 1 3 5 0 . 9 6 6	0 . 4 3 9 0 0 . 2 1		

			0	4		
				5		
4174			0	0		
			.	.		
			2	5		
			0	2		
			5	5		
				5		
			1			
			.	0		
			1	.		
			7	2		
			6	8		
			0	7		
				5		
4175			0	0		
			.	.		
			1	5		
			5	3		
			0	1		
				0		
			1			
			.	0		
			0	.		
			9	2		
			4	6		
			5	1		
				0		
4176			0	0		
			.	.		
			1	9		
			9	4		
			5	5		
				5		
			1			
			.	0		
			9	.		
			4	3		
			8	7		
			5	6		
				5		

Sex Length Diameter Height Whole

Shucked

Viscera

Shell

## Rings

4177 rows x 9 columns

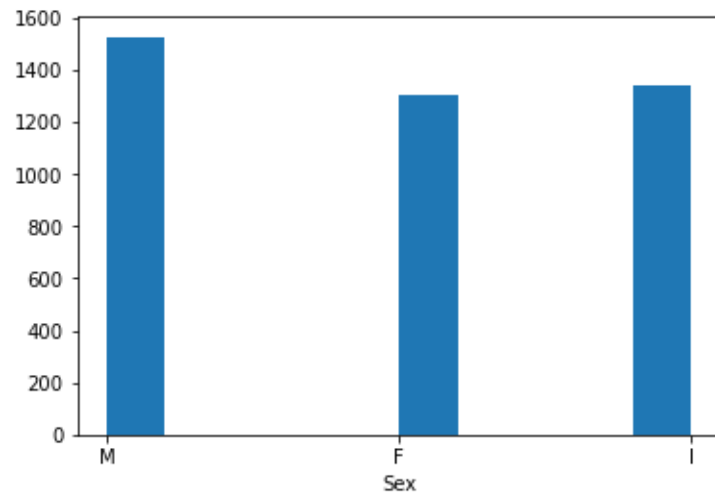
data.info()

```
<class
'pandas.core.frame.DataFrame'>RangeIndex: 4177
entries, 0 to 4176
Data columns (total 9 columns):
#   Column          Non-Null Count  Dtype
•   Sex              4177 non-null   object
•   Length           4177 non-null   float64

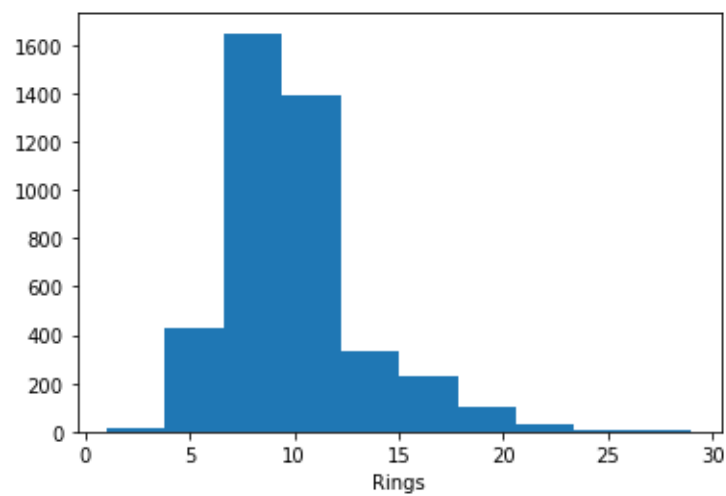
•   Diameter         4177 non-null   float64
•   Height           4177 non-null   float64
•   Whole weight     4177 non-null   float64
•   Shucked weight   4177 non-null   float64
•   Viscera weight    4177 non-null   float64
•   Shell weight     4177 non-null   float64
•   Rings            4177 non-
null int64dtypes: float64(7),
int64(1), object(1)
memory usage: 293.8+ KB
```

## Univariate Analysis

```
plt.hist(data['Sex']);  
plt.xlabel('Sex');
```



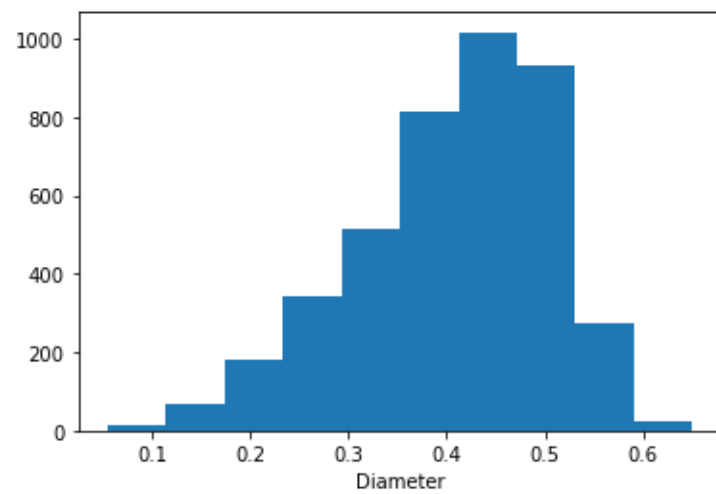
```
plt.hist(data  
['Rings']);  
plt.xlabel('R  
ings');
```



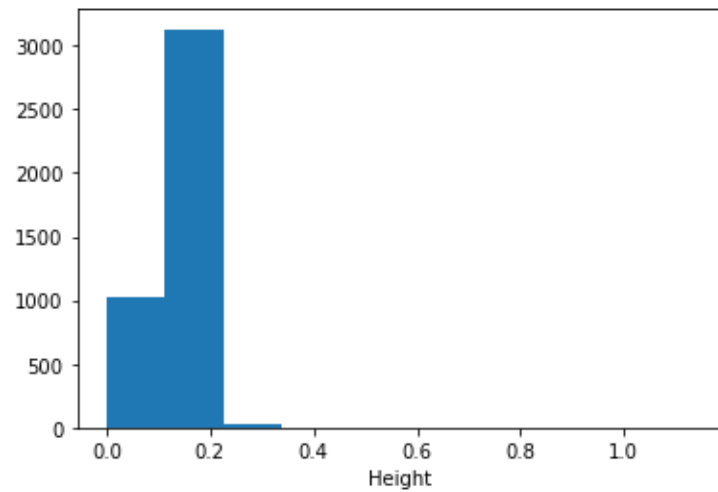
```
sns.boxplot(x=data  
['Length'])
```

```
plt.xlabel('Length  
' );
```

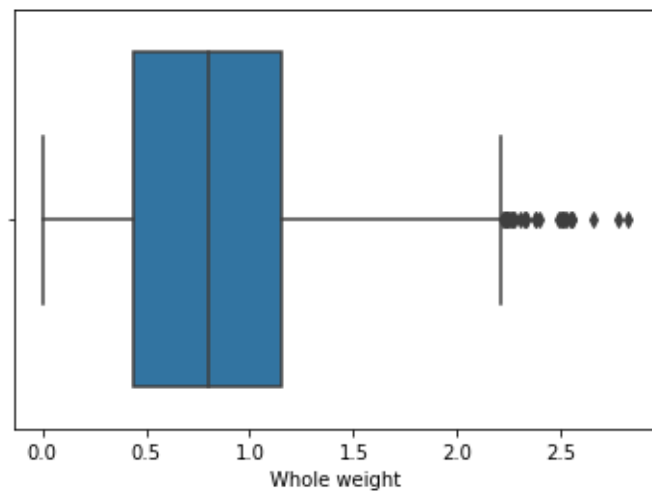
```
plt.hist(data['D  
iameter']);  
plt.xlabel('Diam  
eter');
```



```
plt.hist(data[  
'Height']);  
plt.xlabel('He  
ight');
```

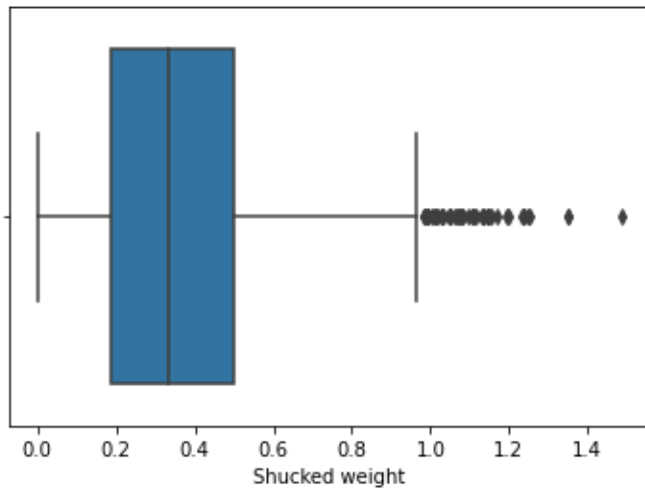


```
sns.boxplot(x=data['Whole weight'])
plt.xlabel('Whole weight');
```

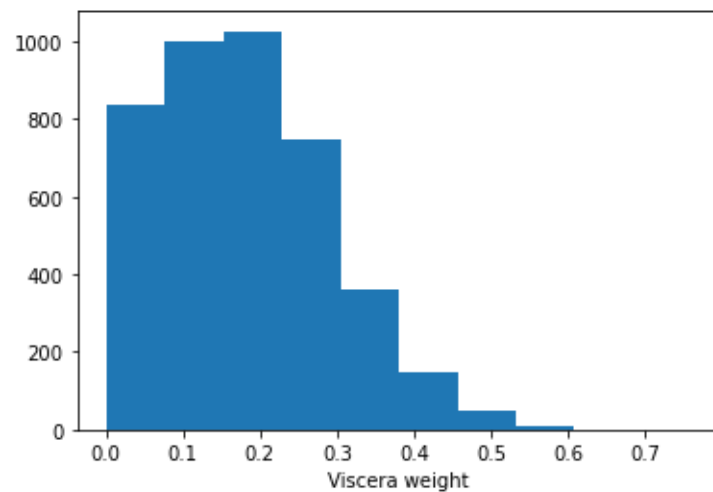


```
sns.boxplot(x=data['Shucked weight'])
plt.xlabel('Shucked weight');
```

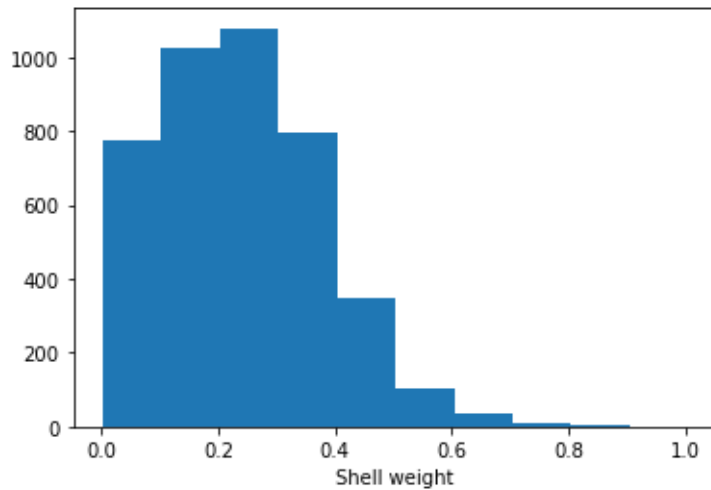




```
plt.hist(data['Viscera
weight']);
plt.xlabel('Viscera
weight');
```

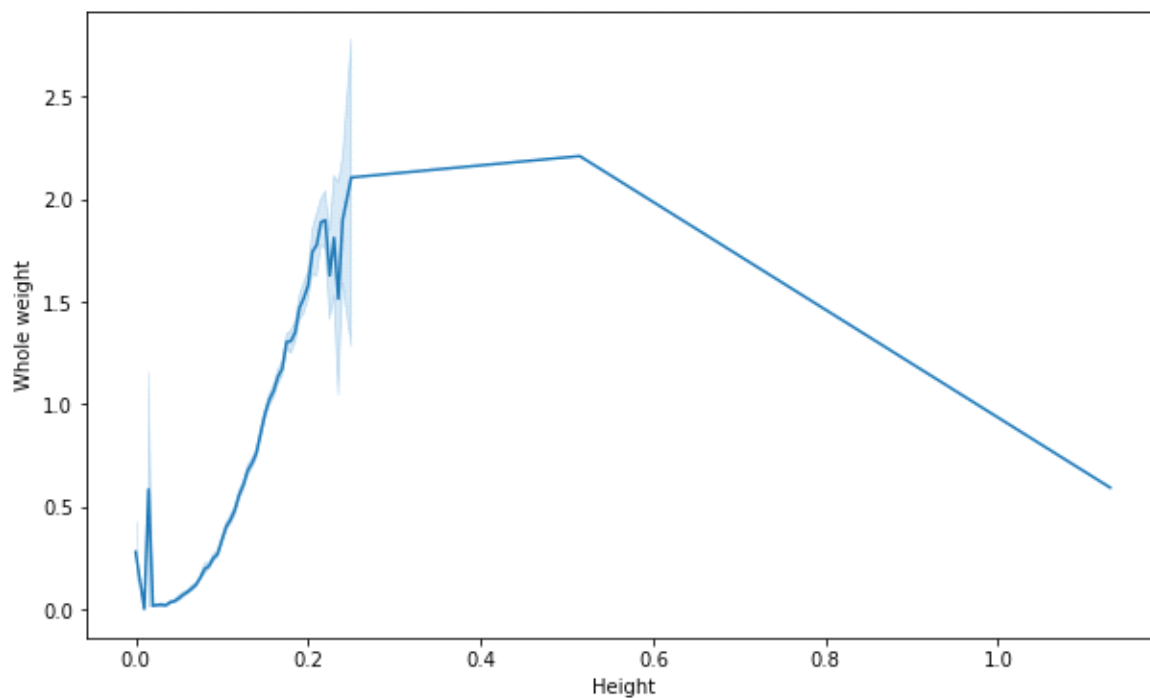


```
plt.hist(data['Shell
weight']);
plt.xlabel('Shell
weight');
```



## Bivariate Analysis

```
plt.figure(figsize=(10, 6))  
sns.lineplot(x=data["Height"], y=data["Whole  
weight"]);plt.xlabel('Height');  
plt.ylabel('Whole weight');
```



```
plt.figure(figsize=(10, 6))
```

```
sns.lineplot(x=data["Length"],  
y=data["Height"]);  
plt.xlabel('Length');  
plt.ylabel('Height');
```

```
plt.figure(figsize=(10, 6))  
sns.lineplot(x=data["Diameter"],  
y=data["Height"]);  
plt.xlabel('Diameter');  
plt.ylabel('Height');
```

```
plt.figure(figsize=(10, 6))
sns.lineplot(x=data["Length"],
y=data["Diameter"]);
plt.xlabel('Length');
plt.ylabel('Diameter');
```

```
plt.figure(figsize=(10, 6))
plt.scatter(x=data["Shucked weight"], y=data["Whole
weight"]);plt.xlabel('Shucked weight');
plt.ylabel('Whole weight');
```

```
plt.figure(figsize=(10, 6))
plt.scatter(x=data["Viscera weight"], y=data["Whole
weight"]);plt.xlabel('Viscera weight');
plt.ylabel('Whole weight');
```

## Multi-variate Analysis

```
sns.pairplot(data, hue='Sex');
```

```
plt.figure(figsize=(10, 6));
sns.heatmap(data.corr(), annot=True);
```

				weight	weight	weight	
count	4177.000000 0	4177.000000 0	4177.000000 0	4177.000000 0	4177.000000 0	4177.000000 0	41
mean	0.523992	0.407881	0.139516	0.828742	0.359367	0.180594	
std	0.120093	0.099240	0.041827	0.490389	0.221963	0.109614	
min	0.075000	0.055000	0.000000	0.002000	0.001000	0.000500	
25%	0.450000	0.350000	0.115000	0.441500	0.186000	0.093500	
50%	0.545000	0.425000	0.140000	0.799500	0.336000	0.171000	

<b>75%</b>	0.615000	0.480000	0.165000	1.153000	0.502000	0.253000	
<b>max</b>	0.815000	0.650000	1.130000	2.825500	1.488000	0.760000	

## Handling Missing Values

```
data.isna().sum()
```

```

Sex                0
Length             0
Diameter           0
Height             0
Whole weight       0
Shucked weight     0
Viscera weight     0
Shell weight       0
Rings              0
dtype: int64

```

## Outlier Handling

```
numeric_cols = ['Length', 'Diameter', 'Height', 'Whole weight', 'Shucked weight', 'Viscera weight']
```

```
def boxplots(cols):
```

```
    fig, axes = plt.subplots(4, 2, figsize=(15, 20))
```

```
    t=0
```

```
    for i in range(4):
```

```
        for j in range(2):
```

```
            sns.boxplot(ax=axes[i][j], data=data,
                        x=cols[t])t+=1
```

```
    plt.show()
```

```
def Flooring_outlier(col):
```

```
    Q1 =
```

```
    data[col].quantile
```

```
    (0.25)Q3 =
```

```

data[col].quantile
(0.75)
IQR = Q3 - Q1
whisker_width = 1.5
lower_whisker = Q1 -
(whisker_width*IQR) upper_whisker =
Q3 + (whisker_width*IQR)
data[col]=np.where(data[col]>upper_whisker,upper_whisker,np.where(data[col]<
lower_whis

```

```

print('Before Outliers
Handling')
print('='*100)
boxplots(nume
ric_cols) for
col in
numeric_cols:
    Flooring_outlier(col)
print('\n\n\nAfter Outliers
Handling')print('='*100)
boxplots(numeric_cols)

```

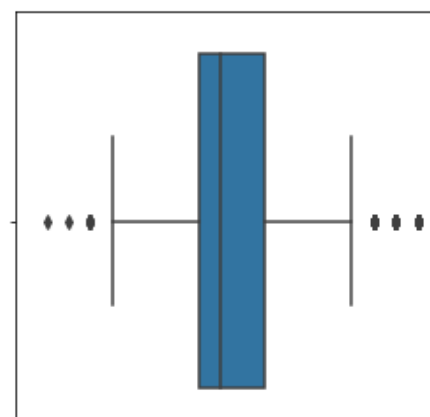
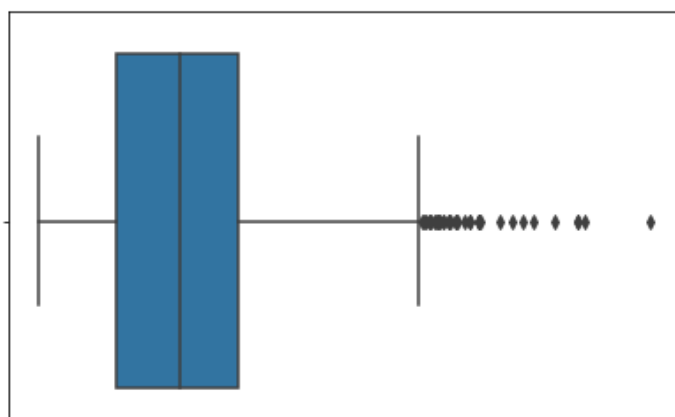
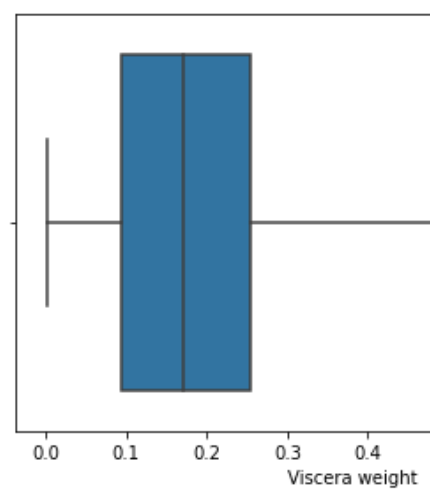
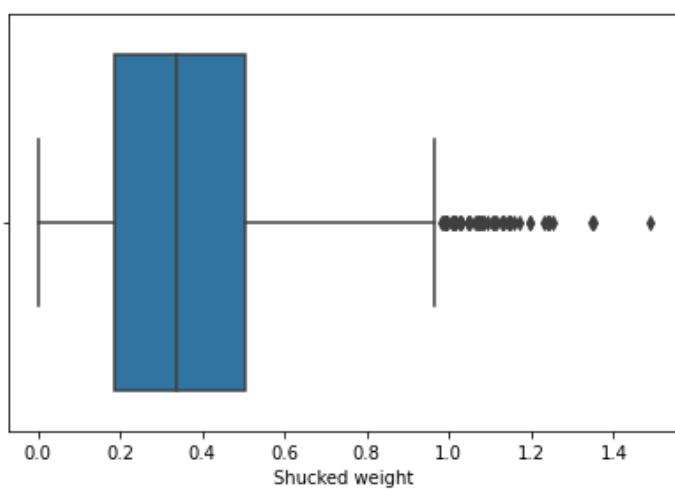
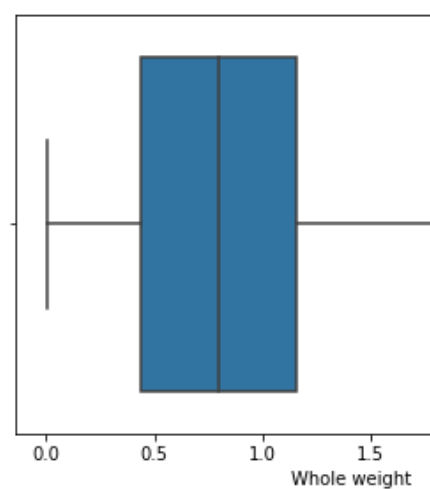
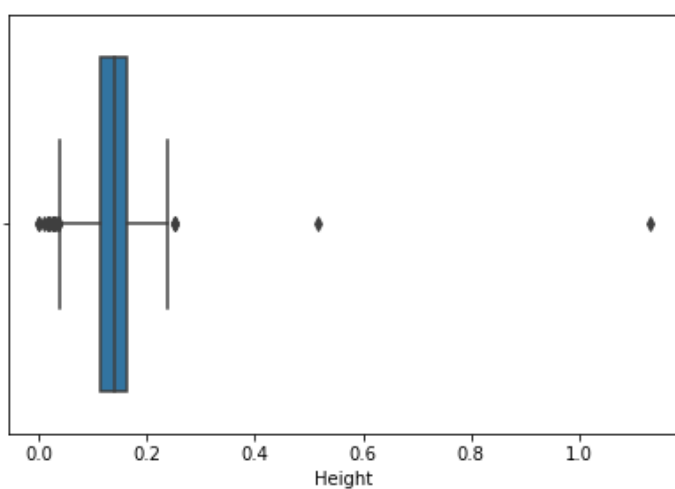
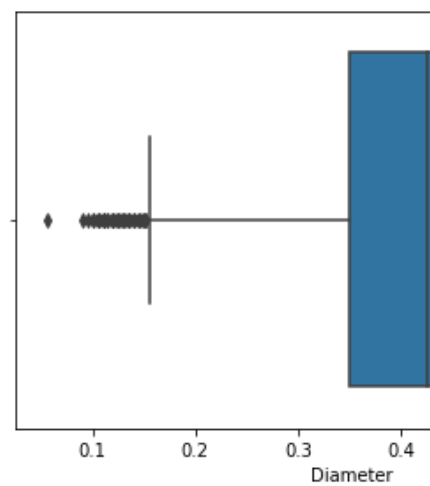
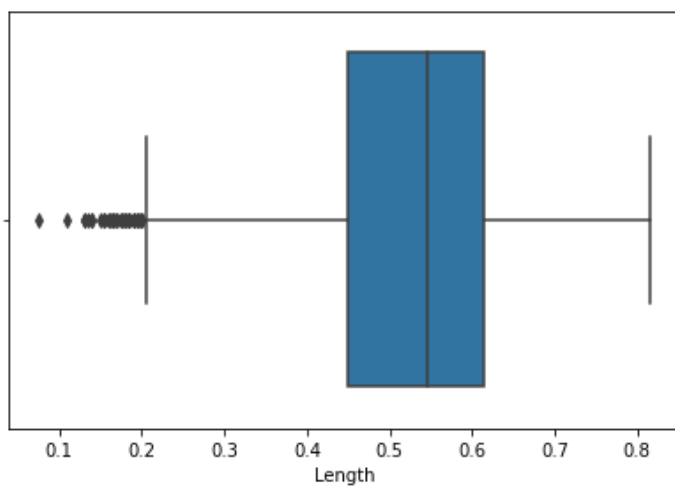
Before Outliers Handling

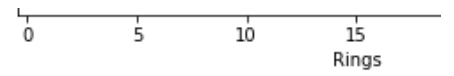
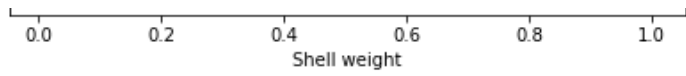
```

=====
=====

```







After Outliers Handling

=====

## Encode Categorical Columns

```
data = pd.get_dummies(data, columns
= ['Sex'])data
```

			weight	weight	weight	weight	
<b>0</b>	0.455	0.365	0.095 0.514 0	0.2245	0.1010	0.1500	15.0
<b>1</b>	0.350	0.265	0.090 0.225 5	0.0995	0.0485	0.0700	7.0
<b>2</b>	0.530	0.420	0.135 0.677 0	0.2565	0.1415	0.2100	9.0
<b>3</b>	0.440	0.365	0.125 0.516 0	0.2155	0.1140	0.1550	10.0
<b>4</b>	0.330	0.255	0.080 0.205 0	0.0895	0.0395	0.0550	7.0
...	...	...	...	...	...	...	...
<b>4172</b> 0.56 5	0.450		0.165 0.887 0	0.3700	0.2390	0.2490	11.0
<b>4173</b> 0.59 0	0.440		0.135 0.966 0	0.4390	0.2145	0.2605	10.0
<b>4174</b> 0.60 0	0.475		0.205 1.176 0	0.5255	0.2875	0.3080	9.0
<b>4175</b> 0.62 5	0.485		0.150 1.094 5	0.5310	0.2610	0.2960	10.0
<b>4176</b> 0.71	0.555		0.195 1.948	0.9455	0.3765	0.4950	12.0

0		5				
---	--	---	--	--	--	--

```
Y = data[['Rings']]
X = data.drop(['Rings'], axis=1)
```

## Scale the independent Variables

```
scaler =
StandardScaler()
X =
scaler.fit_trans
form(X)X
```

array([[ -0.58311728, -0.44088378, -1.15809314, ..., -0.67483383,		
-0.68801788, 1.31667716],		
[ -1.46569411, -1.45976205, -1.28875125, ..., -0.67483383,		
-0.68801788, 1.31667716],		
[ 0.04729474, 0.11949927, -0.1128283 , ..., 1.48184628,		
-0.68801788, -0.75948762],		
...,		
[ 0.63567929, 0.67988232, 1.71638519, ..., -0.67483383,		
-0.68801788, 1.31667716],		
[ 0.84581663, 0.78177015, 0.27914602, ..., 1.48184628,		
-0.68801788, -0.75948762],		
[ 1.56028358, 1.49498494, 1.45506898, ..., -0.67483383,		
-0.68801788, 1.31667716]])		

## Train Test Split

```
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.2,
random_state=42)
```

```
X_train.shape, X_test.shape, Y_train.shape,  
Y_test.shape((3341, 10), (836, 10),  
(3341, 1), (836, 1))
```

## Model Training & Testing

```
model =  
LinearRegression  
(  
model.fit(X_train,  
n, Y_train)  
model.score(X_train, Y_train),  
  
model.score(X_test, Y_test)  
  
(0.5743537797259437, 0.574066914479568)  
  
model = DecisionTreeRegressor(max_depth=15,  
max_leaf_nodes=40)model.fit(X_train, Y_train)  
model.score(X_train, Y_train),  
  
model.score(X_test, Y_test)  
  
(0.6299341126842184, 0.5533377990647702)
```

Colab [HYPERLINK](#)

"https://colab.research.google.com/signup?utm\_source=footer&utm\_medium=link&utm\_campaign=footer\_links"

[HYPERLINK](#)

"https://colab.research.google.com/signup?utm\_source=footer&utm\_medium=link&utm\_campaign=footer\_links" [paid](#) [HYPERLINK](#)

"https://colab.research.google.com/signup?utm\_source=footer&utm\_medium=link&utm\_campaign=footer\_links"

[HYPERLINK](#)

"https://colab.research.google.com/signup?utm\_source=footer&utm\_medium=link&utm\_campaign=footer\_links" [products](#) - [Cancel](#)

[HYPERLINK](#)

"https://colab.research.google.com/cancel-subscription" [HYPERLINK](#)

"https://colab.research.google.com/cancel-subscription" [contracts](#)

[HYPERLINK](#)

"https://colab.research.google.com/cancel-subscription" [HYPERLINK](#)

"https://colab.research.google.com/cancel-subscription" [here](#)

