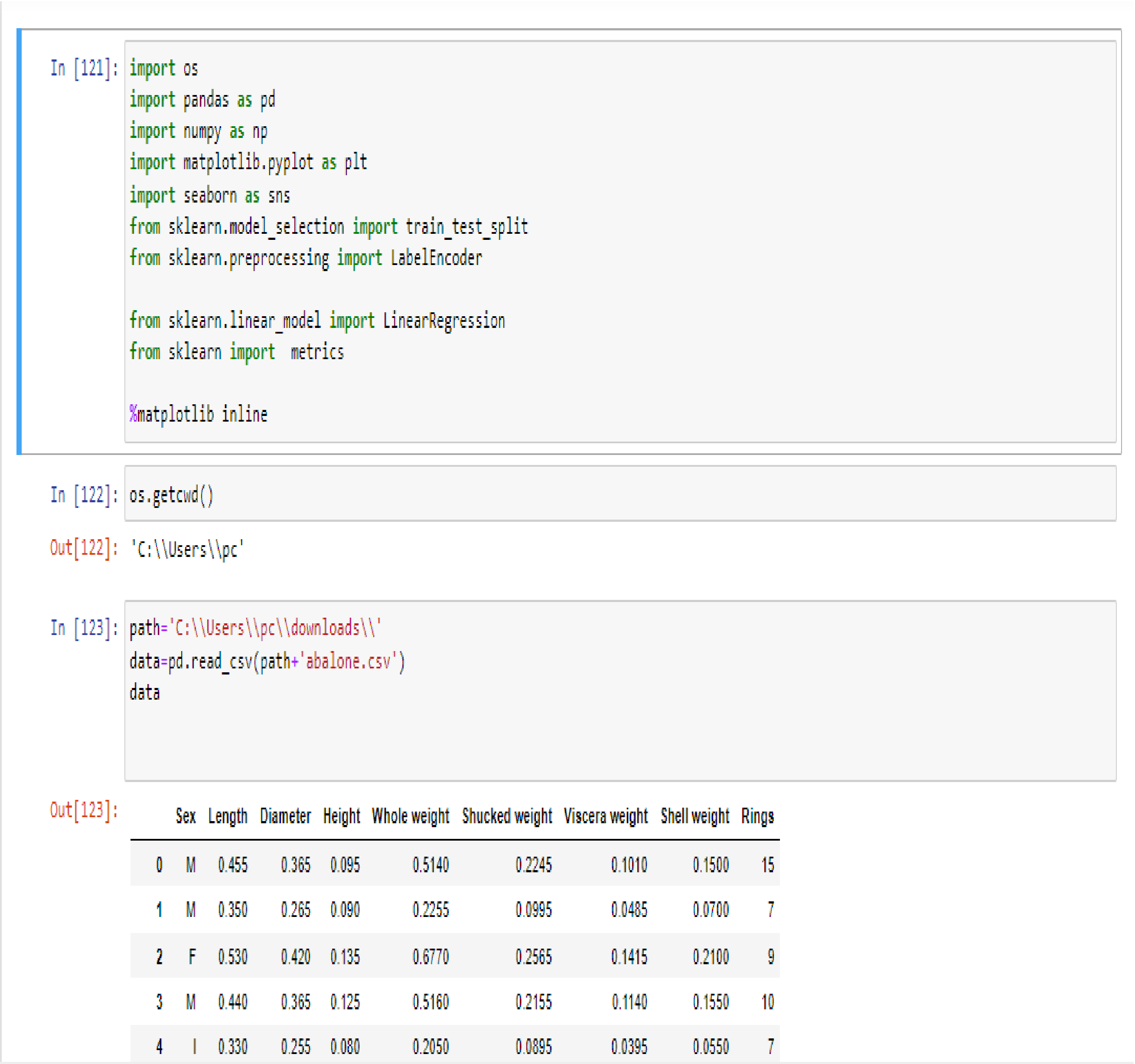
Assignment -3

Python Programming

Assignment Date	7 October 2022
Student Name	Rasika S
Student Roll Number	812419106036
Maximum Marks	2 Marks



In [121]:
```
import os
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder

from sklearn.linear_model import LinearRegression
from sklearn import  metrics

%matplotlib inline
```

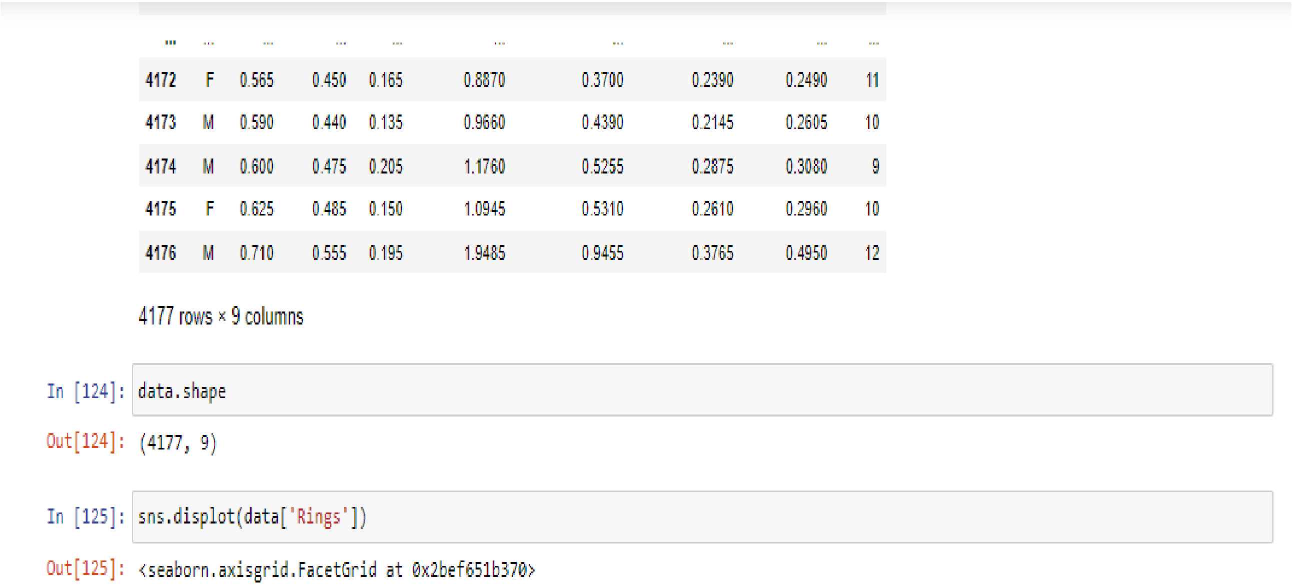
In [122]:
```
os.getcwd()
```

Out[122]:
```
'C:\\Users\\pc'
```

In [123]:
```
path='C:\\Users\\pc\\downloads\\'
data=pd.read_csv(path+'abalone.csv')
data
```

Out[123]:

	Sex	Length	Diameter	Height	Whole weight	Shucked weight	Viscera weight	Shell weight	Rings
0	M	0.455	0.365	0.095	0.5140	0.2245	0.1010	0.1500	15
1	M	0.350	0.265	0.090	0.2255	0.0995	0.0485	0.0700	7
2	F	0.530	0.420	0.135	0.6770	0.2565	0.1415	0.2100	9
3	M	0.440	0.365	0.125	0.5160	0.2155	0.1140	0.1550	10
4	I	0.330	0.255	0.080	0.2050	0.0895	0.0395	0.0550	7

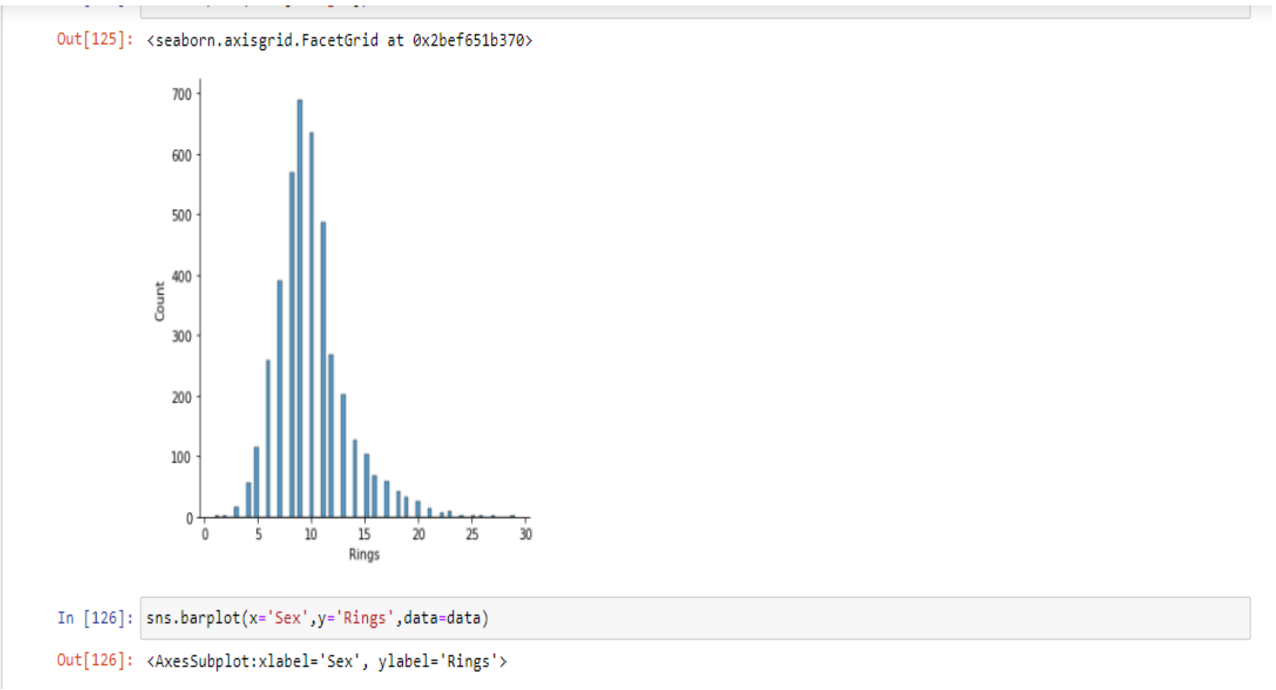


...	...	...	...	...	...	...	...	...	...
4172	F	0.565	0.450	0.165	0.8870	0.3700	0.2390	0.2490	11
4173	M	0.590	0.440	0.135	0.9660	0.4390	0.2145	0.2605	10
4174	M	0.600	0.475	0.205	1.1760	0.5255	0.2875	0.3080	9
4175	F	0.625	0.485	0.150	1.0945	0.5310	0.2610	0.2960	10
4176	M	0.710	0.555	0.195	1.9485	0.9455	0.3765	0.4950	12

4177 rows × 9 columns

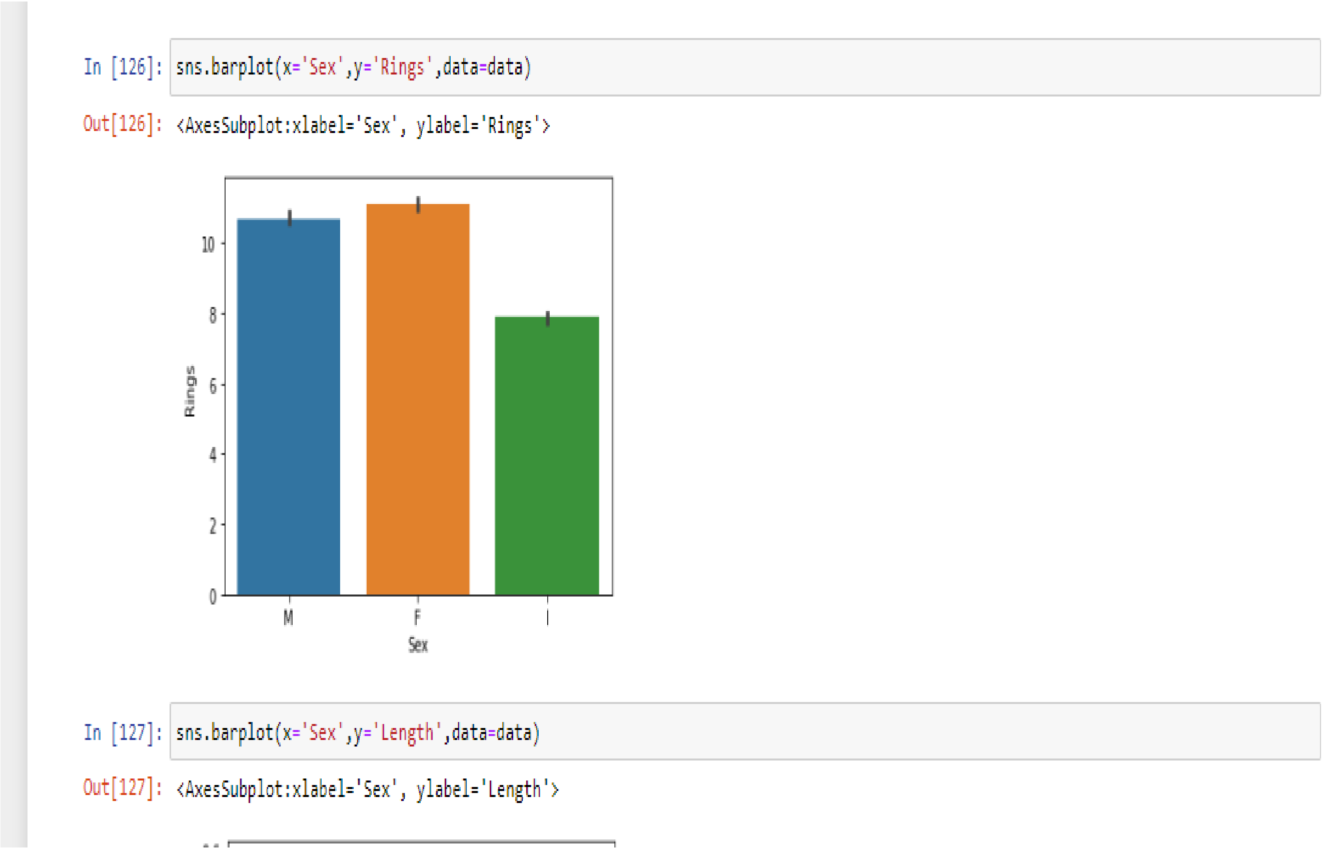
```
In [124]: data.shape
Out[124]: (4177, 9)
```

```
In [125]: sns.displot(data['Rings'])
Out[125]: <seaborn.axisgrid.FacetGrid at 0x2bef651b370>
```



```
Out[125]: <seaborn.axisgrid.FacetGrid at 0x2bef651b370>
```

```
In [126]: sns.barplot(x='Sex',y='Rings',data=data)
Out[126]: <AxesSubplot:xlabel='Sex', ylabel='Rings'>
```

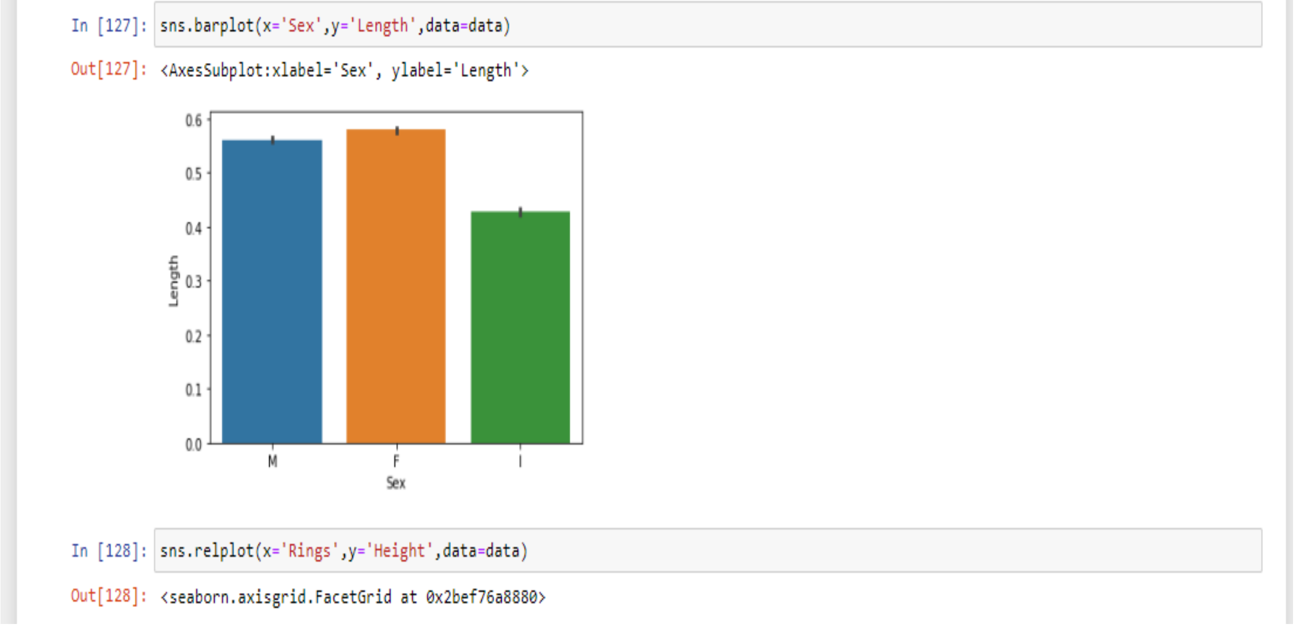


```
In [126]: sns.barplot(x='Sex',y='Rings',data=data)
```

```
Out[126]: <AxesSubplot:xlabel='Sex', ylabel='Rings'>
```

```
In [127]: sns.barplot(x='Sex',y='Length',data=data)
```

```
Out[127]: <AxesSubplot:xlabel='Sex', ylabel='Length'>
```

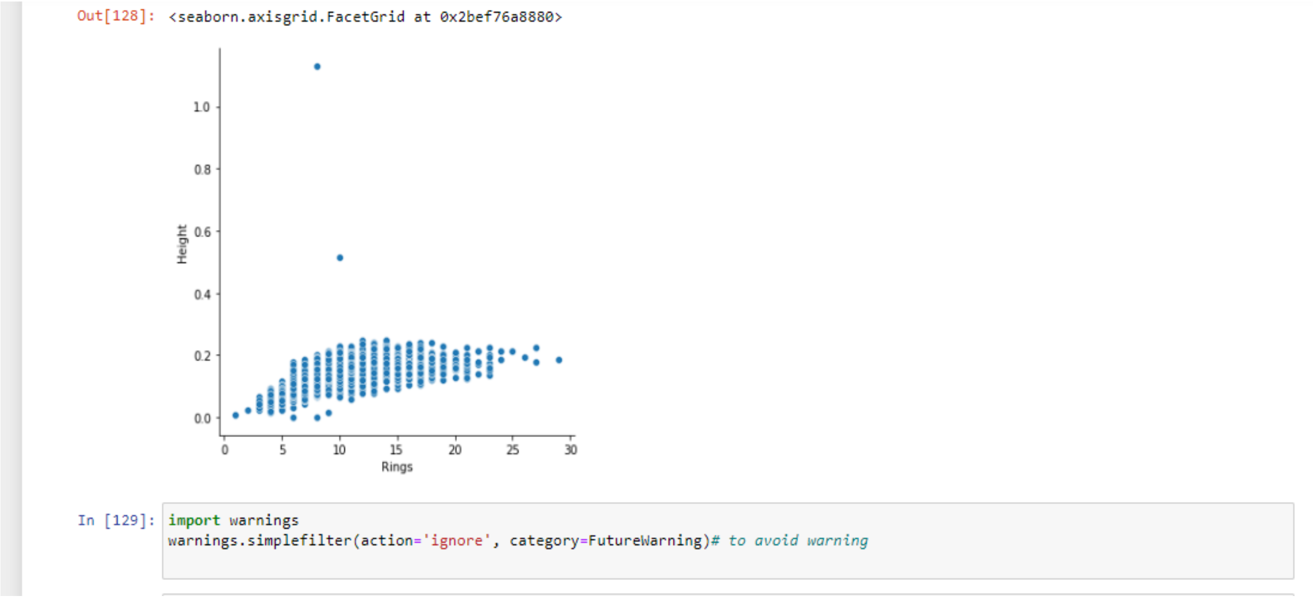


```
In [127]: sns.barplot(x='Sex',y='Length',data=data)
```

```
Out[127]: <AxesSubplot:xlabel='Sex', ylabel='Length'>
```

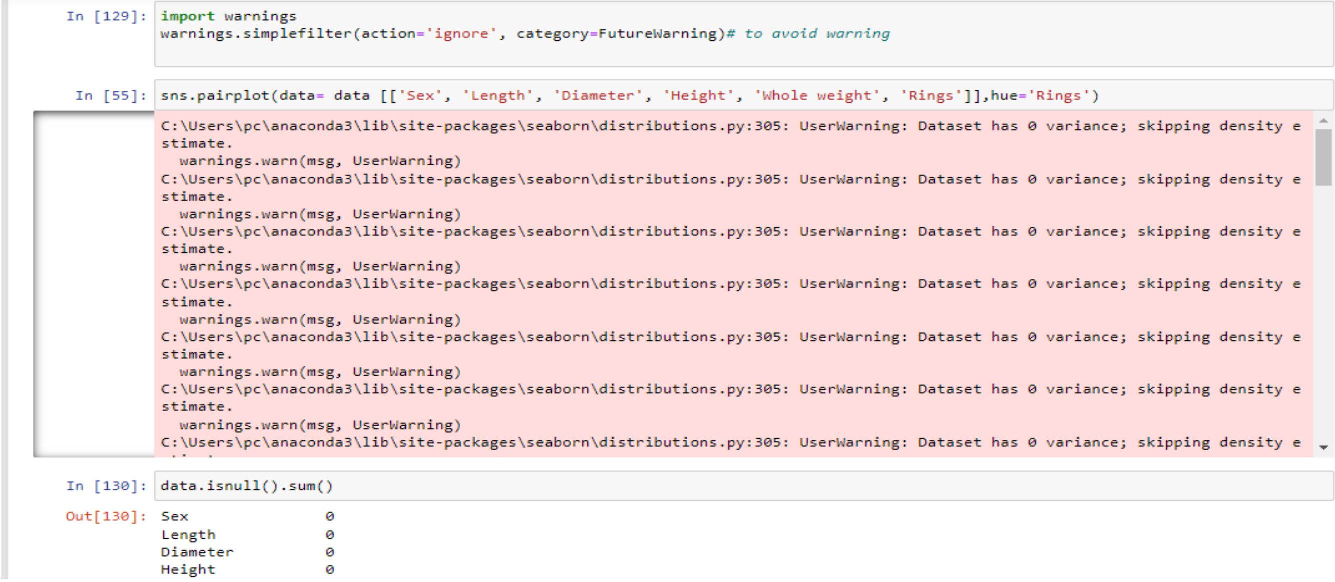
```
In [128]: sns.relplot(x='Rings',y='Height',data=data)
```

```
Out[128]: <seaborn.axisgrid.FacetGrid at 0x2bef76a8880>
```



```
Out[128]: <seaborn.axisgrid.FacetGrid at 0x2bef76a8880>
```

```
In [129]: import warnings
          warnings.simplefilter(action='ignore', category=FutureWarning)# to avoid warning
```



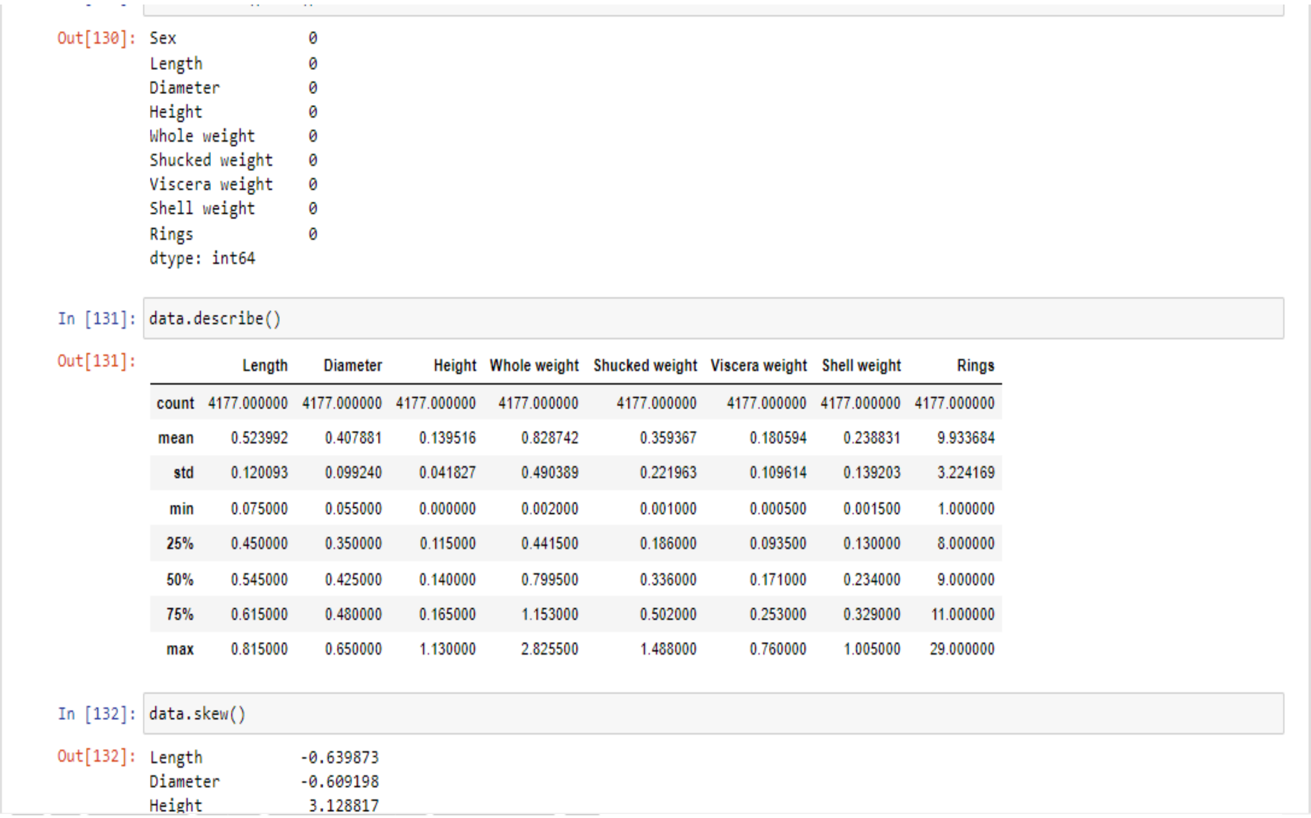
```
In [129]: import warnings
          warnings.simplefilter(action='ignore', category=FutureWarning)# to avoid warning
```

```
In [55]: sns.pairplot(data= data [['Sex', 'Length', 'Diameter', 'Height', 'Whole weight', 'Rings']],hue='Rings')
```

```
C:\Users\pc\anaconda3\lib\site-packages\seaborn\distributions.py:305: UserWarning: Dataset has 0 variance; skipping density e
stimate.
  warnings.warn(msg, UserWarning)
C:\Users\pc\anaconda3\lib\site-packages\seaborn\distributions.py:305: UserWarning: Dataset has 0 variance; skipping density e
stimate.
  warnings.warn(msg, UserWarning)
C:\Users\pc\anaconda3\lib\site-packages\seaborn\distributions.py:305: UserWarning: Dataset has 0 variance; skipping density e
stimate.
  warnings.warn(msg, UserWarning)
C:\Users\pc\anaconda3\lib\site-packages\seaborn\distributions.py:305: UserWarning: Dataset has 0 variance; skipping density e
stimate.
  warnings.warn(msg, UserWarning)
C:\Users\pc\anaconda3\lib\site-packages\seaborn\distributions.py:305: UserWarning: Dataset has 0 variance; skipping density e
stimate.
  warnings.warn(msg, UserWarning)
C:\Users\pc\anaconda3\lib\site-packages\seaborn\distributions.py:305: UserWarning: Dataset has 0 variance; skipping density e
stimate.
  warnings.warn(msg, UserWarning)
C:\Users\pc\anaconda3\lib\site-packages\seaborn\distributions.py:305: UserWarning: Dataset has 0 variance; skipping density e
```

```
In [130]: data.isnull().sum()
```

```
Out[130]: Sex          0
          Length       0
          Diameter     0
          Height       0
```



```
Out[130]: Sex               0
          Length            0
          Diameter          0
          Height            0
          Whole weight      0
          Shucked weight    0
          Viscera weight    0
          Shell weight      0
          Rings             0
          dtype: int64
```

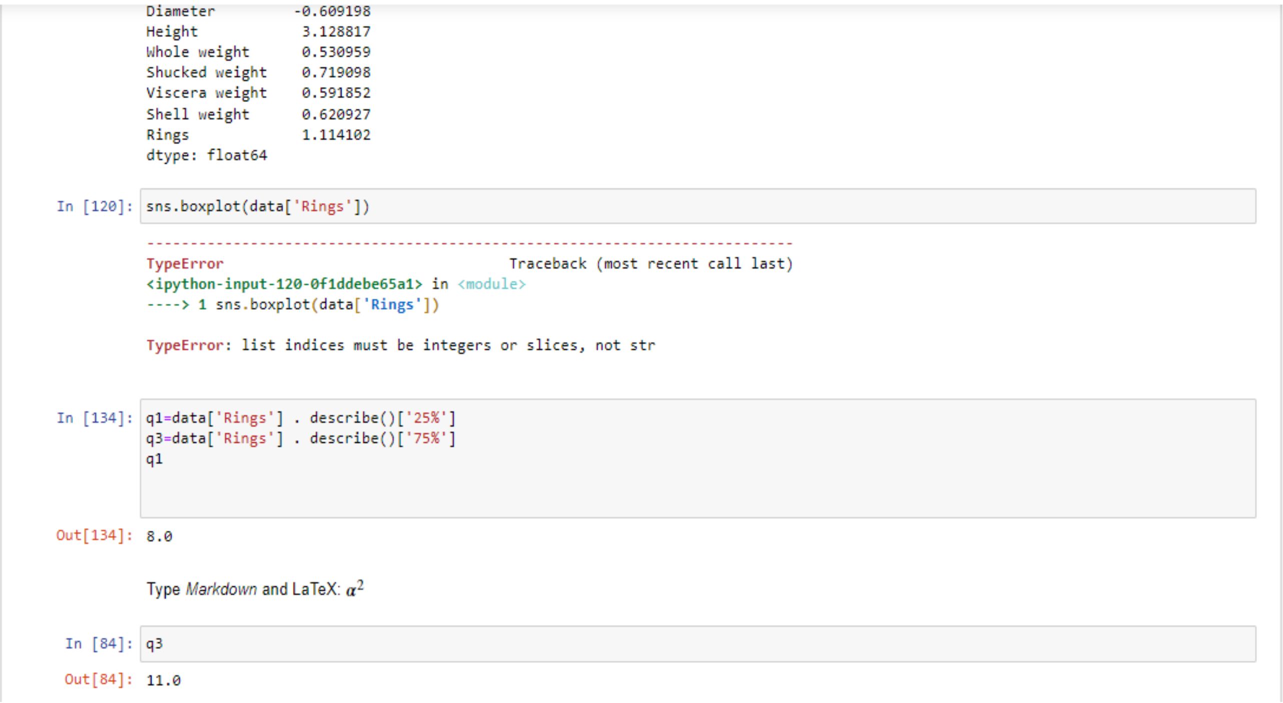
```
In [131]: data.describe()
```

Out[131]:

	Length	Diameter	Height	Whole weight	Shucked weight	Viscera weight	Shell weight	Rings
count	4177.000000	4177.000000	4177.000000	4177.000000	4177.000000	4177.000000	4177.000000	4177.000000
mean	0.523992	0.407881	0.139516	0.828742	0.359367	0.180594	0.238831	9.933684
std	0.120093	0.099240	0.041827	0.490389	0.221963	0.109614	0.139203	3.224169
min	0.075000	0.055000	0.000000	0.002000	0.001000	0.000500	0.001500	1.000000
25%	0.450000	0.350000	0.115000	0.441500	0.186000	0.093500	0.130000	8.000000
50%	0.545000	0.425000	0.140000	0.799500	0.336000	0.171000	0.234000	9.000000
75%	0.615000	0.480000	0.165000	1.153000	0.502000	0.253000	0.329000	11.000000
max	0.815000	0.650000	1.130000	2.825500	1.488000	0.760000	1.005000	29.000000

```
In [132]: data.skew()

Out[132]: Length           -0.639873
          Diameter         -0.609198
          Height            3.128817
```



```
Diameter         -0.609198
Height            3.128817
```

```
Whole weight      0.530959
Shucked weight    0.719098
Viscera weight    0.591852
Shell weight      0.620927
Rings             1.114102
dtype: float64
```

```
In [120]: sns.boxplot(data['Rings'])

          ---------------------------------------------------------------------------
          TypeError                                 Traceback (most recent call last)
          <ipython-input-120-0f1ddebe65a1> in <module>
          ----> 1 sns.boxplot(data['Rings'])

          TypeError: list indices must be integers or slices, not str
```

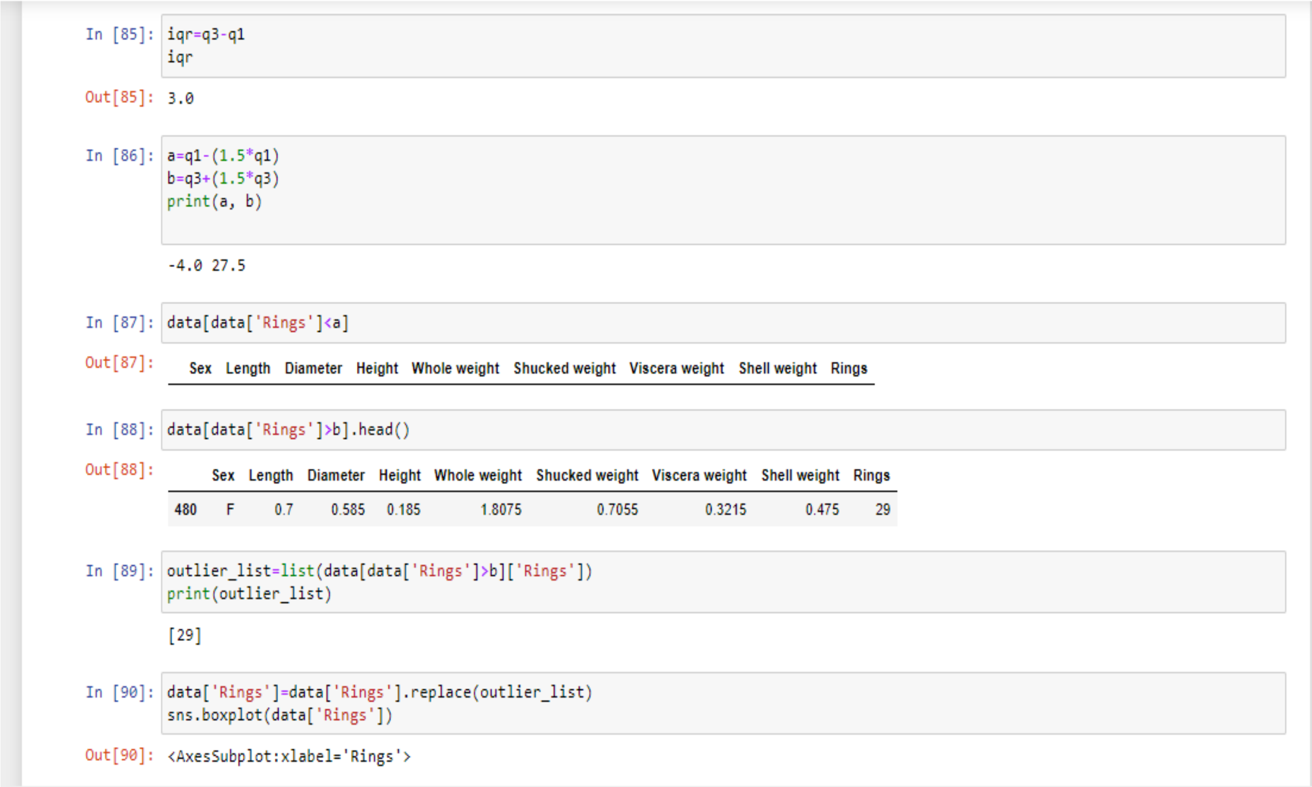
```
In [134]: q1=data['Rings'] . describe()['25%']
          q3=data['Rings'] . describe()['75%']
          q1

Out[134]: 8.0
```

Type *Markdown* and LaTeX: α^2

```
In [84]: q3

Out[84]: 11.0
```



```
In [85]: iqr=q3-q1
         iqr
Out[85]: 3.0
```

```
In [86]: a=q1-(1.5*q1)
         b=q3+(1.5*q3)
         print(a, b)

         -4.0 27.5
```

```
In [87]: data[data['Rings']<a]
```

Out[87]:

	Sex	Length	Diameter	Height	Whole weight	Shucked weight	Viscera weight	Shell weight	Rings

```
In [88]: data[data['Rings']>b].head()
```

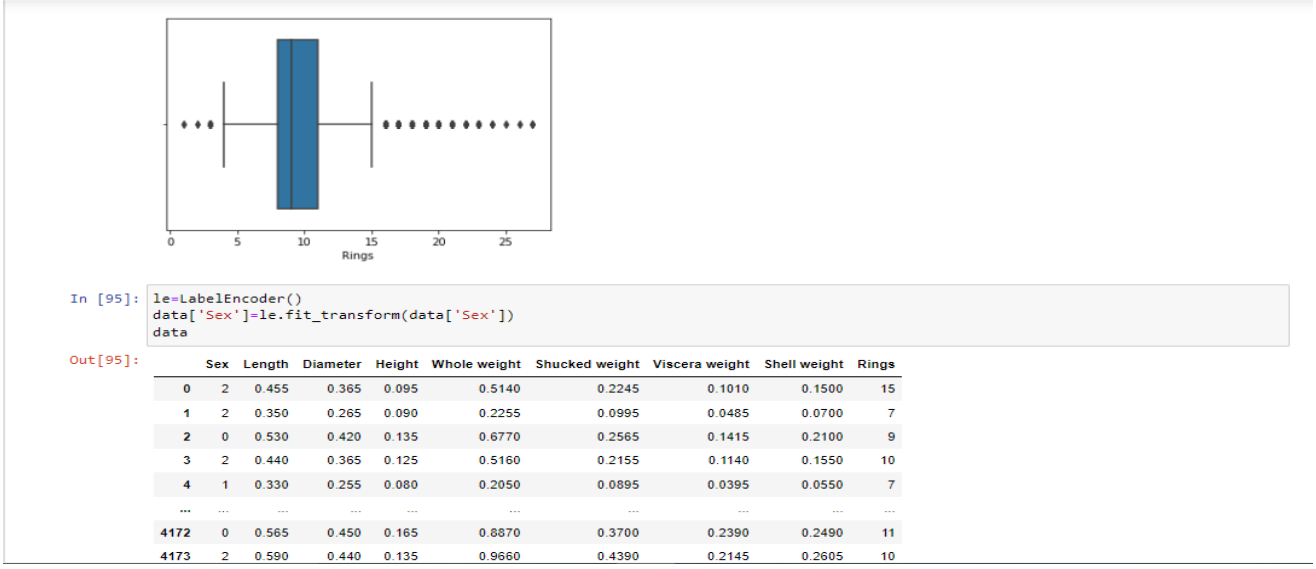
Out[88]:

	Sex	Length	Diameter	Height	Whole weight	Shucked weight	Viscera weight	Shell weight	Rings
480	F	0.7	0.585	0.185	1.8075	0.7055	0.3215	0.475	29

```
In [89]: outlier_list=list(data[data['Rings']>b]['Rings'])
         print(outlier_list)

         [29]
```

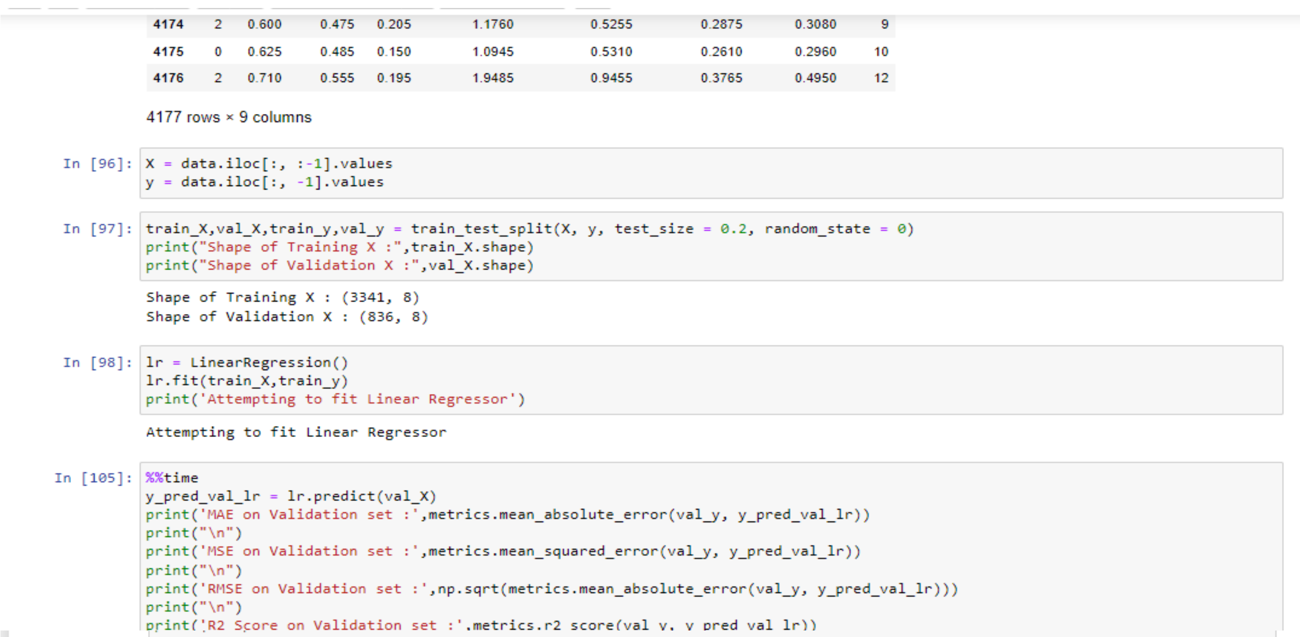
```
In [90]: data['Rings']=data['Rings'].replace(outlier_list)
         sns.boxplot(data['Rings'])
Out[90]: <AxesSubplot:xlabel='Rings'>
```



```
In [95]: le=LabelEncoder()
         data['Sex']=le.fit_transform(data['Sex'])
         data
```

Out[95]:

	Sex	Length	Diameter	Height	Whole weight	Shucked weight	Viscera weight	Shell weight	Rings
0	2	0.455	0.365	0.095	0.5140	0.2245	0.1010	0.1500	15
1	2	0.350	0.265	0.090	0.2255	0.0995	0.0485	0.0700	7
2	0	0.530	0.420	0.135	0.6770	0.2565	0.1415	0.2100	9
3	2	0.440	0.365	0.125	0.5160	0.2155	0.1140	0.1550	10
4	1	0.330	0.255	0.080	0.2050	0.0895	0.0395	0.0550	7
...	...	...	...	...	...	...	...	...	...
4172	0	0.565	0.450	0.165	0.8870	0.3700	0.2390	0.2490	11
4173	2	0.590	0.440	0.135	0.9660	0.4390	0.2145	0.2605	10



4174	2	0.600	0.475	0.205	1.1760	0.5255	0.2875	0.3080	9
4175	0	0.625	0.485	0.150	1.0945	0.5310	0.2610	0.2960	10
4176	2	0.710	0.555	0.195	1.9485	0.9455	0.3765	0.4950	12

4177 rows × 9 columns

```
In [96]: X = data.iloc[:, :-1].values
         y = data.iloc[:, -1].values
```

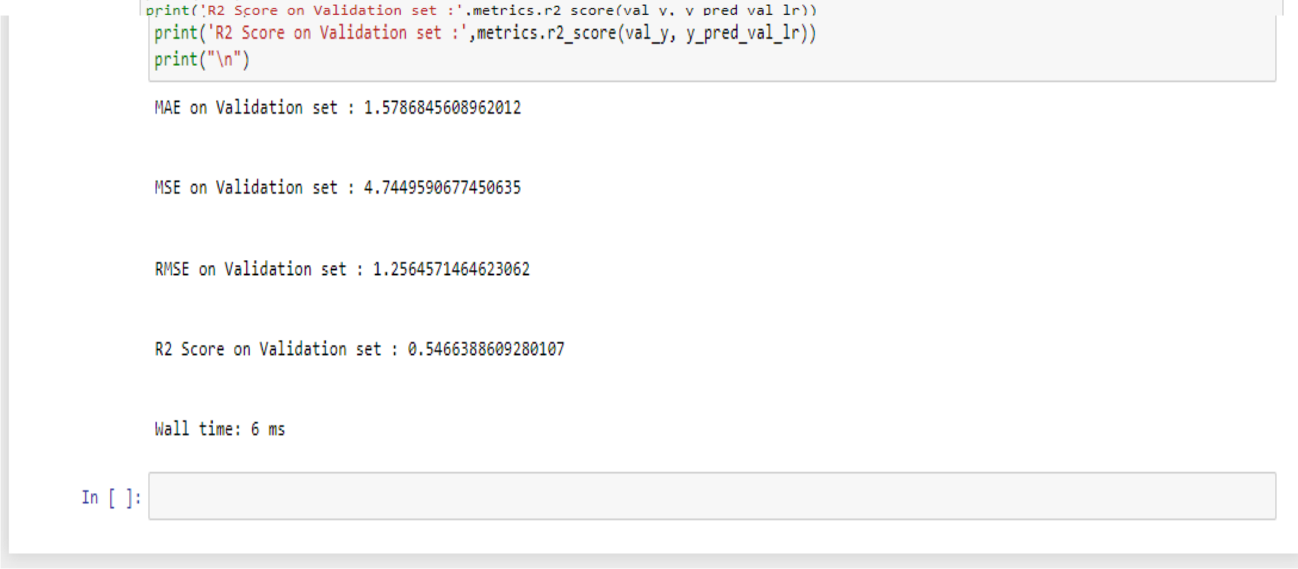
```
In [97]: train_X,val_X,train_y,val_y = train_test_split(X, y, test_size = 0.2, random_state = 0)
         print("Shape of Training X :",train_X.shape)
         print("Shape of Validation X :",val_X.shape)
```

```
Shape of Training X : (3341, 8)
Shape of Validation X : (836, 8)
```

```
In [98]: lr = LinearRegression()
         lr.fit(train_X,train_y)
         print('Attempting to fit Linear Regressor')
```

```
Attempting to fit Linear Regressor
```

```
In [105]: %%time
          y_pred_val_lr = lr.predict(val_X)
          print('MAE on Validation set :',metrics.mean_absolute_error(val_y, y_pred_val_lr))
          print("\n")
          print('MSE on Validation set :',metrics.mean_squared_error(val_y, y_pred_val_lr))
          print("\n")
          print('RMSE on Validation set :',np.sqrt(metrics.mean_absolute_error(val_y, y_pred_val_lr)))
          print("\n")
          print('R2 Score on Validation set :',metrics.r2_score(val_y, y_pred_val_lr))
```



```
print('R2 Score on Validation set :',metrics.r2_score(val_y, y_pred_val_lr))
```

```
print("\n")
```

```
MAE on Validation set : 1.5786845608962012


MSE on Validation set : 4.7449590677450635


RMSE on Validation set : 1.2564571464623062


R2 Score on Validation set : 0.5466388609280107


Wall time: 6 ms
```

```
In [ ]:
```