Data Preprocessing

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from matplotlib import rcParams
%matplotlib inline
import seaborn as sns
pd.options.display.max_rows = None
pd.options.display.max_columns = None

ab = pd.read_csv("/content/abalone.csv")
ab.head()
```

```
  Sex  Length  Diameter  Height  Whole weight  Shucked weight  Viscera
weight  \
0   M   0.455     0.365   0.095        0.5140          0.2245
0.1010
1   M   0.350     0.265   0.090        0.2255          0.0995
0.0485
2   F   0.530     0.420   0.135        0.6770          0.2565
0.1415
3   M   0.440     0.365   0.125        0.5160          0.2155
0.1140
4   I   0.330     0.255   0.080        0.2050          0.0895
0.0395

   Shell weight  Rings
0         0.150     15
1         0.070      7
2         0.210      9
3         0.155     10
4         0.055      7
```

```
ab.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4177 entries, 0 to 4176
Data columns (total 9 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   Sex             4177 non-null   object
 1   Length          4177 non-null   float64
 2   Diameter        4177 non-null   float64
 3   Height          4177 non-null   float64
 4   Whole weight    4177 non-null   float64
 5   Shucked weight  4177 non-null   float64
 6   Viscera weight  4177 non-null   float64
 7   Shell weight    4177 non-null   float64
 8   Rings           4177 non-null   int64
```

```
dtypes: float64(7), int64(1), object(1)
memory usage: 293.8+ KB
```

```
ab.describe()
```

```
            Length     Diameter       Height  Whole weight  Shucked
weight  \
count  4177.000000  4177.000000  4177.000000   4177.000000
4177.000000
mean      0.523992     0.407881     0.139516      0.828742
0.359367
std       0.120093     0.099240     0.041827      0.490389
0.221963
min       0.075000     0.055000     0.000000      0.002000
0.001000
25%       0.450000     0.350000     0.115000      0.441500
0.186000
50%       0.545000     0.425000     0.140000      0.799500
0.336000
75%       0.615000     0.480000     0.165000      1.153000
0.502000
max       0.815000     0.650000     1.130000      2.825500
1.488000

       Viscera weight  Shell weight        Rings
count     4177.000000   4177.000000  4177.000000
mean         0.180594      0.238831     9.933684
std          0.109614      0.139203     3.224169
min          0.000500      0.001500     1.000000
25%          0.093500      0.130000     8.000000
50%          0.171000      0.234000     9.000000
75%          0.253000      0.329000    11.000000
max          0.760000      1.005000    29.000000
```

```
ab.isnull().any()
```

```
Sex               False
Length            False
Diameter          False
Height            False
Whole weight      False
Shucked weight    False
Viscera weight    False
Shell weight      False
Rings             False
dtype: bool
```

Encoding Categorical Data

```
from sklearn.preprocessing import LabelEncoder
le = LabelEncoder()
```

```
ab.Sex = le.fit_transform(ab.Sex)
```

```
ab.head()
```

```
   Sex  Length  Diameter  Height  Whole weight  Shucked weight  \
0    2   0.455     0.365   0.095        0.5140          0.2245
1    2   0.350     0.265   0.090        0.2255          0.0995
2    0   0.530     0.420   0.135        0.6770          0.2565
3    2   0.440     0.365   0.125        0.5160          0.2155
4    1   0.330     0.255   0.080        0.2050          0.0895

   Viscera weight  Shell weight  Rings
0          0.1010         0.150     15
1          0.0485         0.070      7
2          0.1415         0.210      9
3          0.1140         0.155     10
4          0.0395         0.055      7
```

Splitting Independent and Dependant variable

```
X = ab.drop(columns=['Rings','Sex'],axis=1)
y = ab.Rings
```

```
X.head()
```

```
   Length  Diameter  Height  Whole weight  Shucked weight  Viscera
weight  \
0   0.455     0.365   0.095        0.5140          0.2245
0.1010
1   0.350     0.265   0.090        0.2255          0.0995
0.0485
2   0.530     0.420   0.135        0.6770          0.2565
0.1415
3   0.440     0.365   0.125        0.5160          0.2155
0.1140
4   0.330     0.255   0.080        0.2050          0.0895
0.0395

   Shell weight
0         0.150
1         0.070
2         0.210
3         0.155
4         0.055
```

```
y.head()
```

```
0    15
1     7
2     9
3    10
```

```
4     7
Name: Rings, dtype: int64
```

```
X.describe()
```

```
            Length     Diameter       Height  Whole weight  Shucked
weight  \
count  4177.000000  4177.000000  4177.000000   4177.000000
4177.000000
mean      0.523992     0.407881     0.139516      0.828742
0.359367
std       0.120093     0.099240     0.041827      0.490389
0.221963
min       0.075000     0.055000     0.000000      0.002000
0.001000
25%       0.450000     0.350000     0.115000      0.441500
0.186000
50%       0.545000     0.425000     0.140000      0.799500
0.336000
75%       0.615000     0.480000     0.165000      1.153000
0.502000
max       0.815000     0.650000     1.130000      2.825500
1.488000

       Viscera weight  Shell weight
count     4177.000000   4177.000000
mean         0.180594      0.238831
std          0.109614      0.139203
min          0.000500      0.001500
25%          0.093500      0.130000
50%          0.171000      0.234000
75%          0.253000      0.329000
max          0.760000      1.005000
```

```
y.describe()
```

```
count    4177.000000
mean        9.933684
std         3.224169
min         1.000000
25%         8.000000
50%         9.000000
75%        11.000000
max        29.000000
Name: Rings, dtype: float64
```

Univariate Analysis

```
ab.head()
```

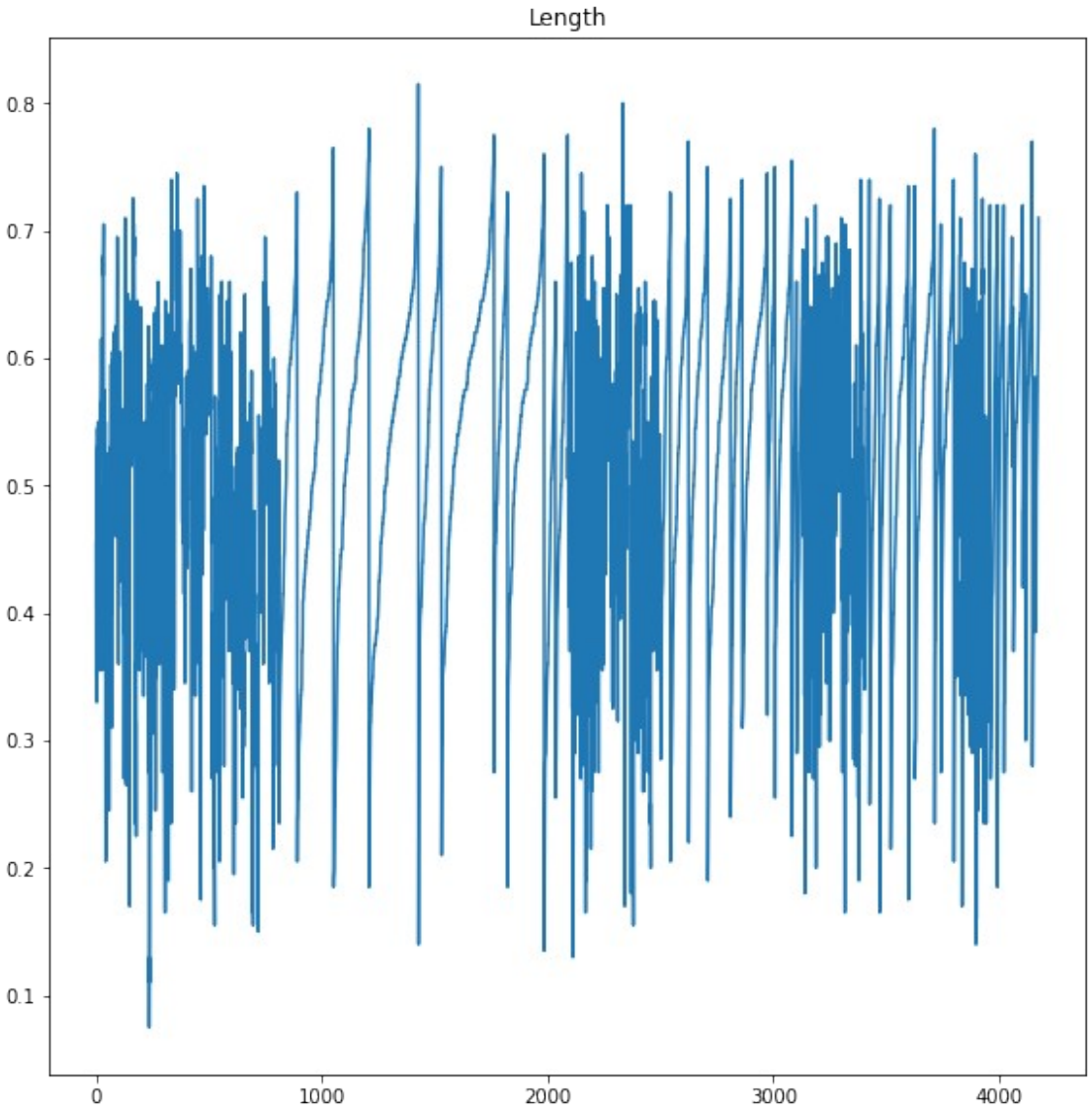
```
   Sex  Length  Diameter  Height  Whole weight  Shucked weight  \
0    2   0.455     0.365   0.095        0.5140          0.2245
```

```
1    2   0.350     0.265   0.090        0.2255          0.0995
2    0   0.530     0.420   0.135        0.6770          0.2565
3    2   0.440     0.365   0.125        0.5160          0.2155
4    1   0.330     0.255   0.080        0.2050          0.0895
```

```
   Viscera weight  Shell weight  Rings
0          0.1010         0.150     15
1          0.0485         0.070      7
2          0.1415         0.210      9
3          0.1140         0.155     10
4          0.0395         0.055      7
```

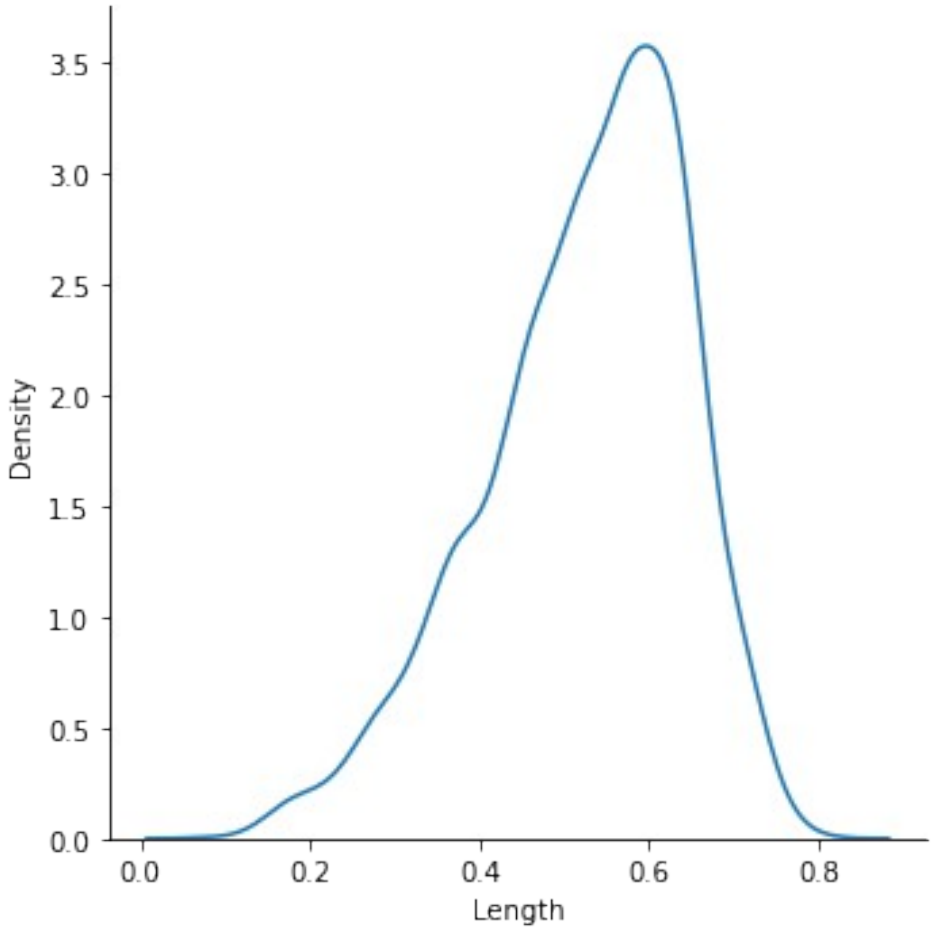
```
fig = plt.figure(figsize=(5,5))
cx = fig.add_axes([0,0,1.5,1.5])
cx.set_title('Length')
cx.plot(ab.Length)
```

```
[<matplotlib.lines.Line2D at 0x7fd21eeaa9d0>]
```



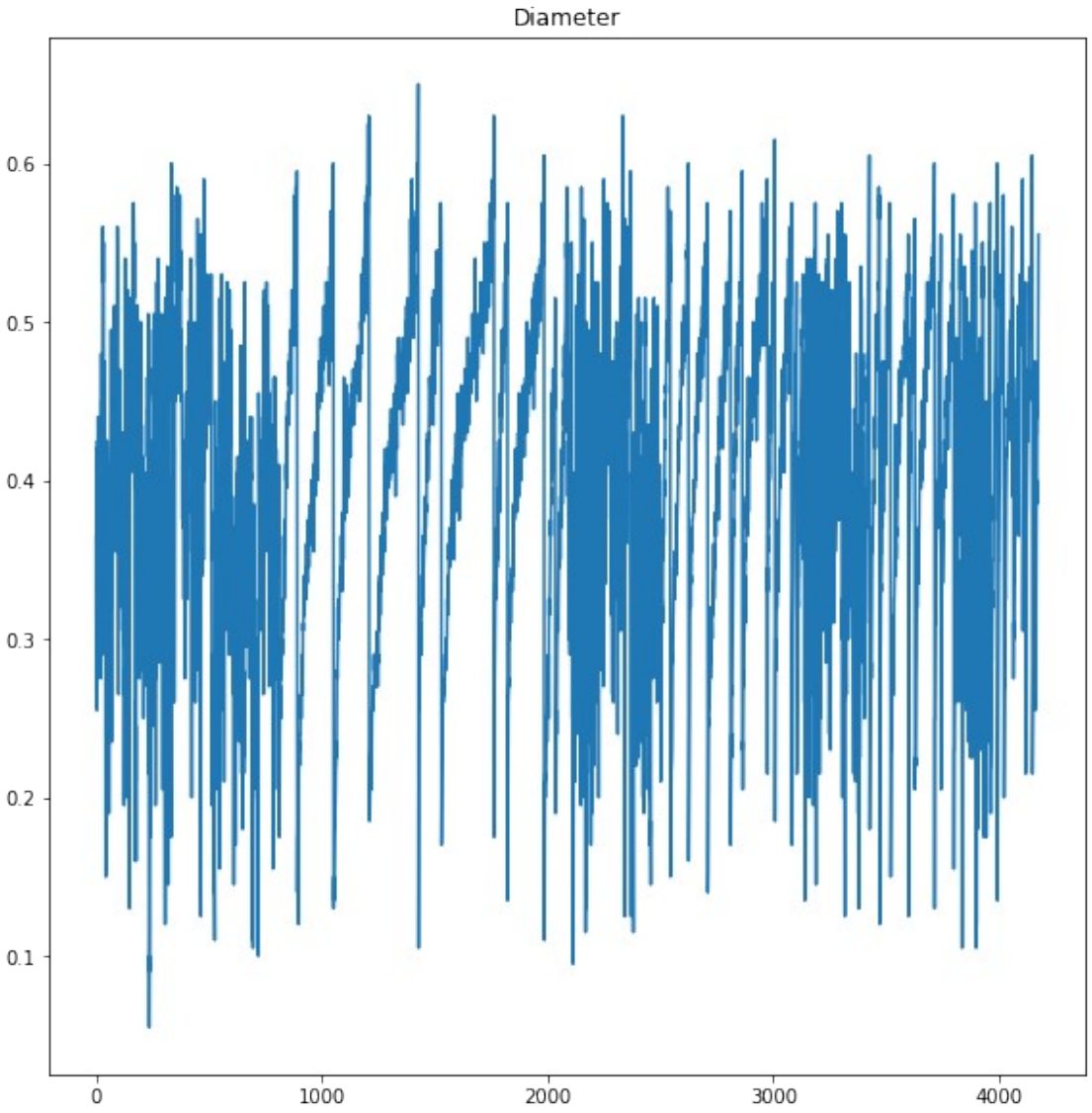

```
sns.displot(ab.Length,kind='kde')
```

```
<seaborn.axisgrid.FacetGrid at 0x7fd21ede5810>
```



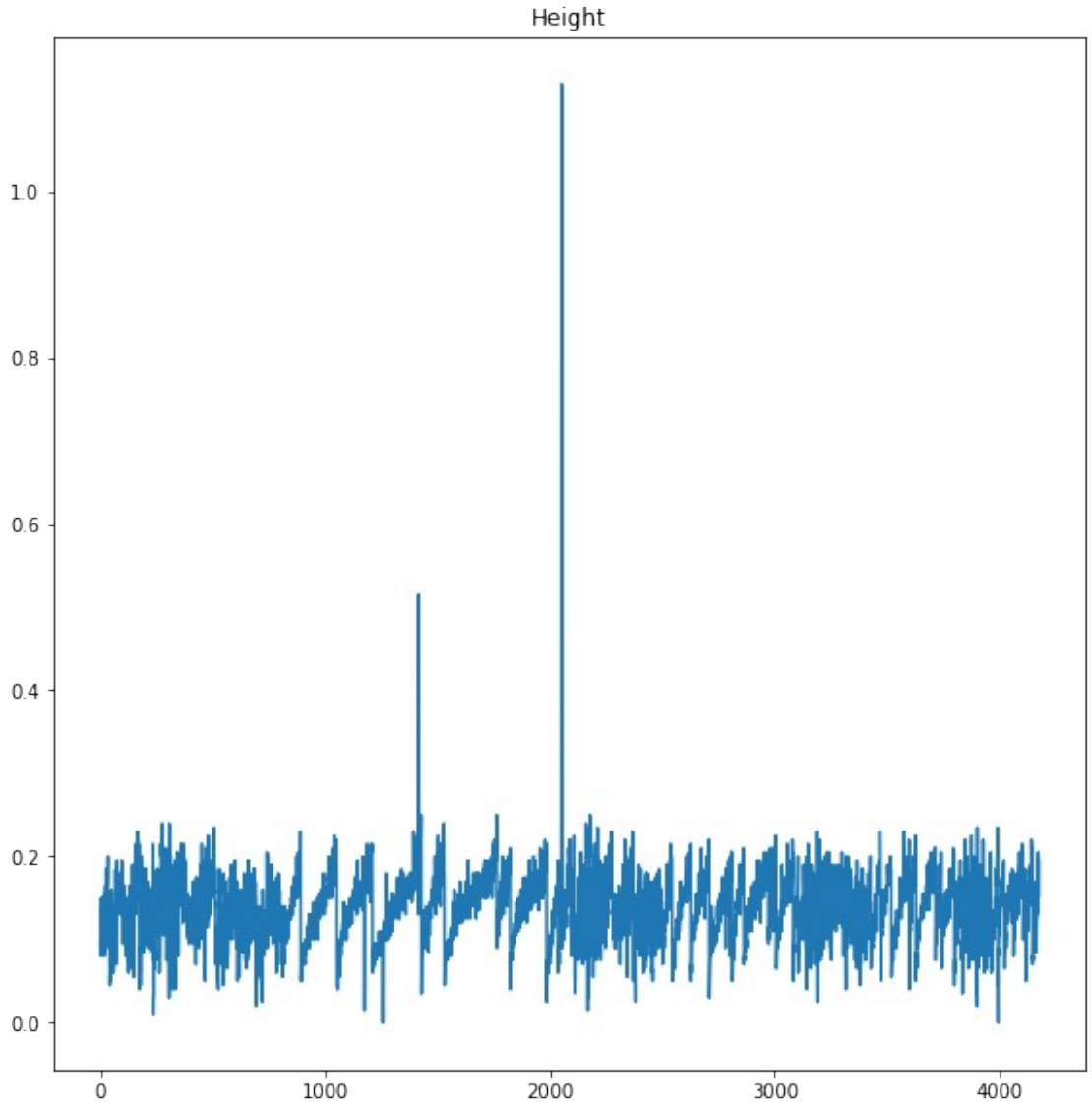

```
fig = plt.figure(figsize=(5,5))
cx = fig.add_axes([0,0,1.5,1.5])
cx.set_title('Diameter')
cx.plot(ab.Diameter)
```

```
[<matplotlib.lines.Line2D at 0x7fd21a9031d0>]
```



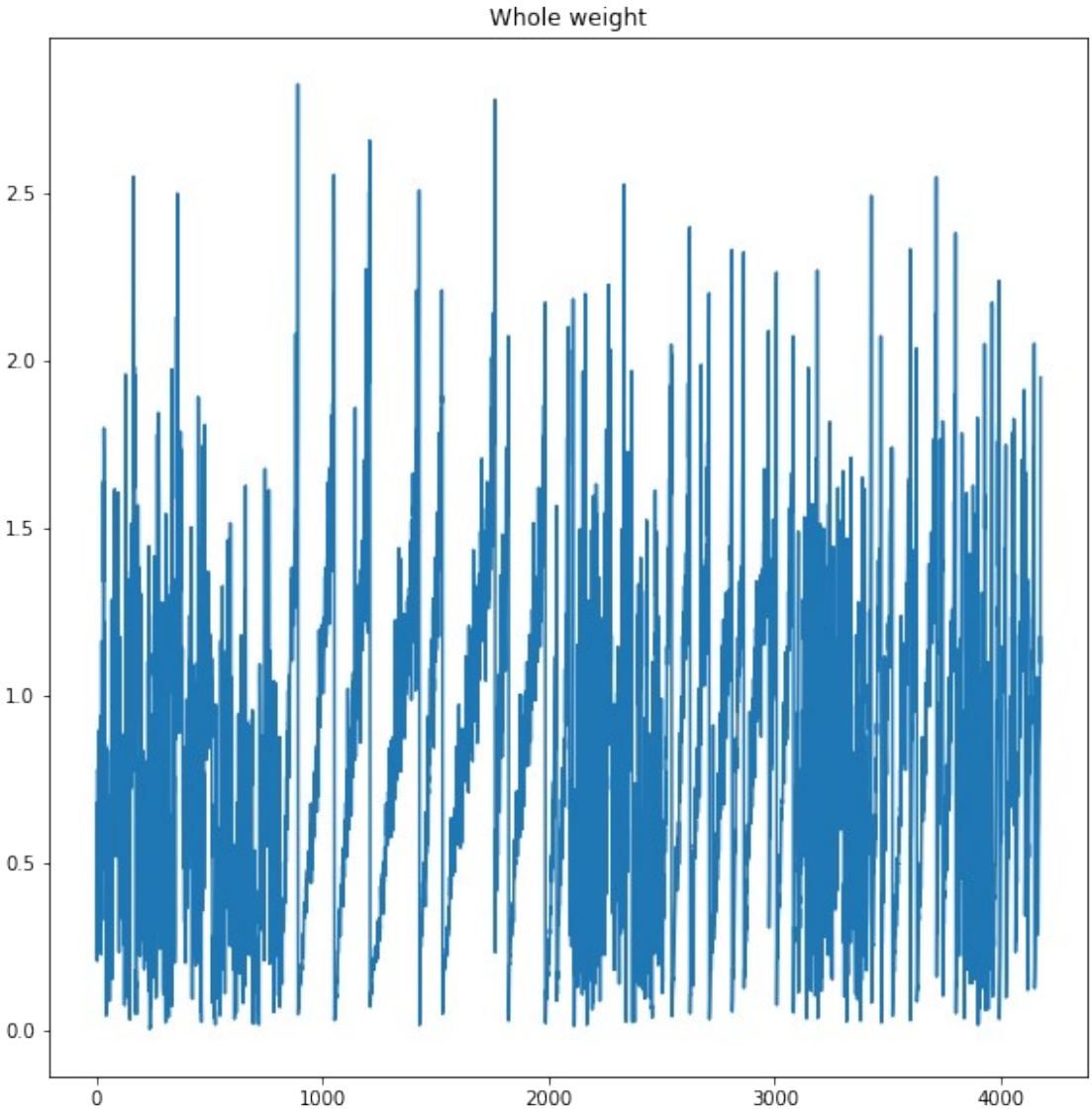

```
fig = plt.figure(figsize=(5,5))
cx = fig.add_axes([0,0,1.5,1.5])
cx.set_title('Height')
cx.plot(ab.Height)
```

```
[<matplotlib.lines.Line2D at 0x7fd21a86e290>]
```



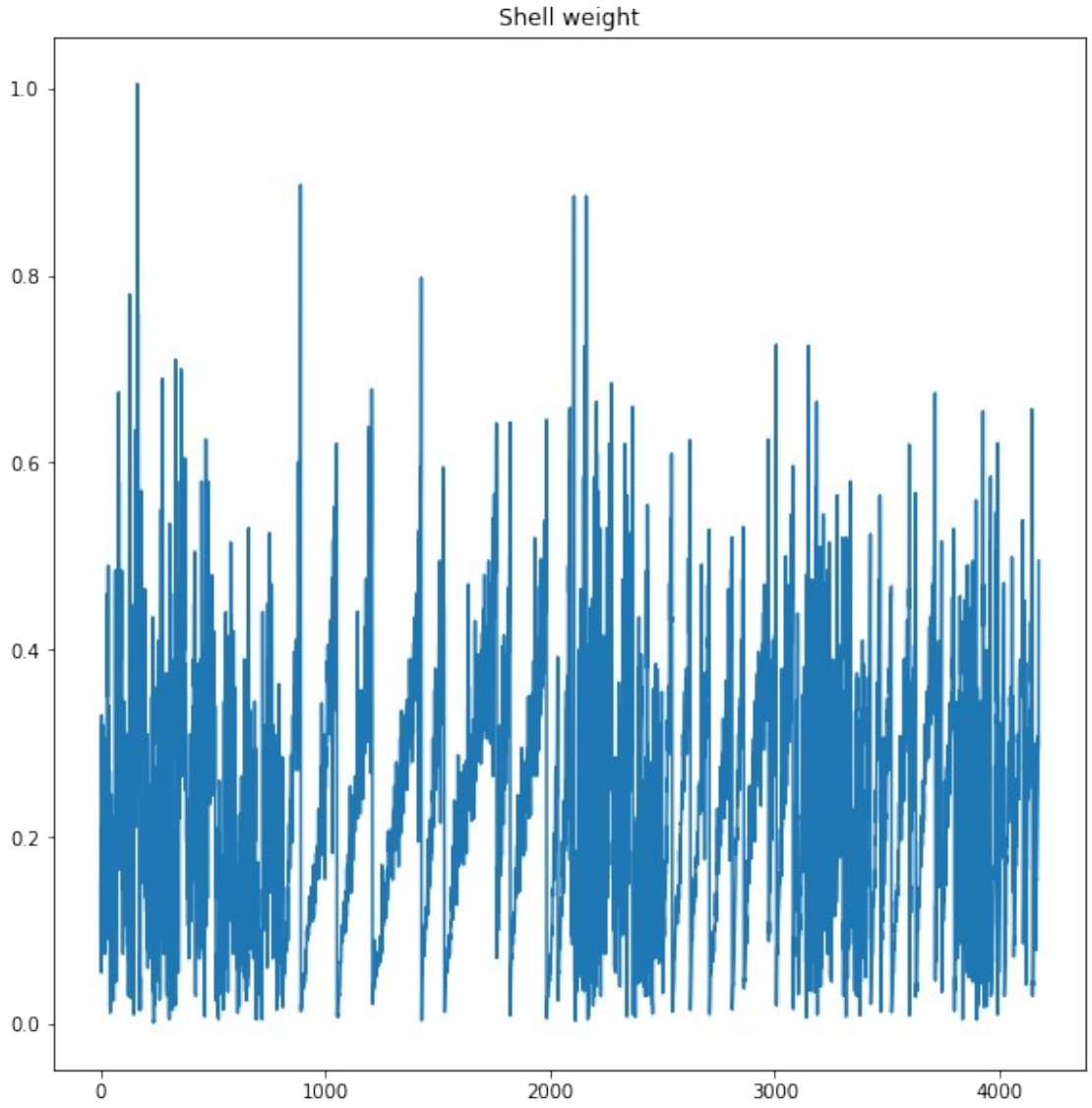

```
fig = plt.figure(figsize=(5,5))
cx = fig.add_axes([0,0,1.5,1.5])
cx.set_title('Whole weight')
cx.plot(ab['Whole weight'])
```

```
[<matplotlib.lines.Line2D at 0x7fd21a7d1610>]
```



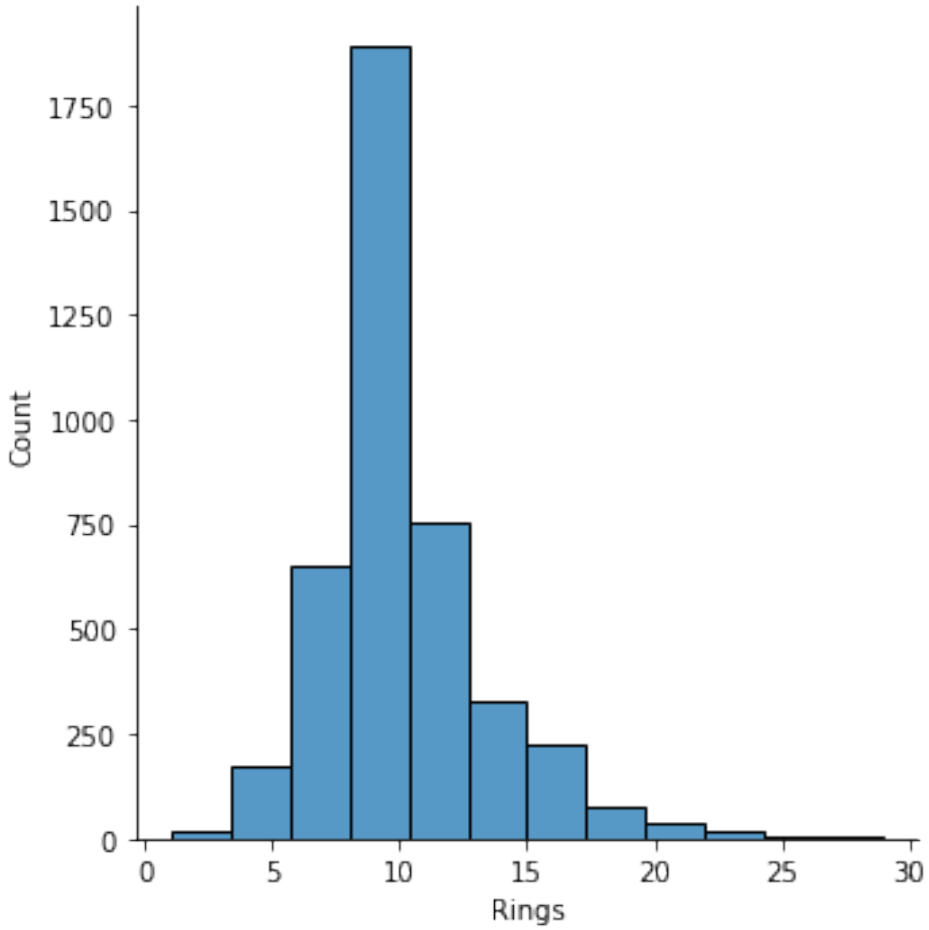

```
fig = plt.figure(figsize=(5,5))
cx = fig.add_axes([0,0,1.5,1.5])
cx.set_title('Shell weight')
cx.plot(ab['Shell weight'])
```

```
[<matplotlib.lines.Line2D at 0x7fd21a7b8990>]
```



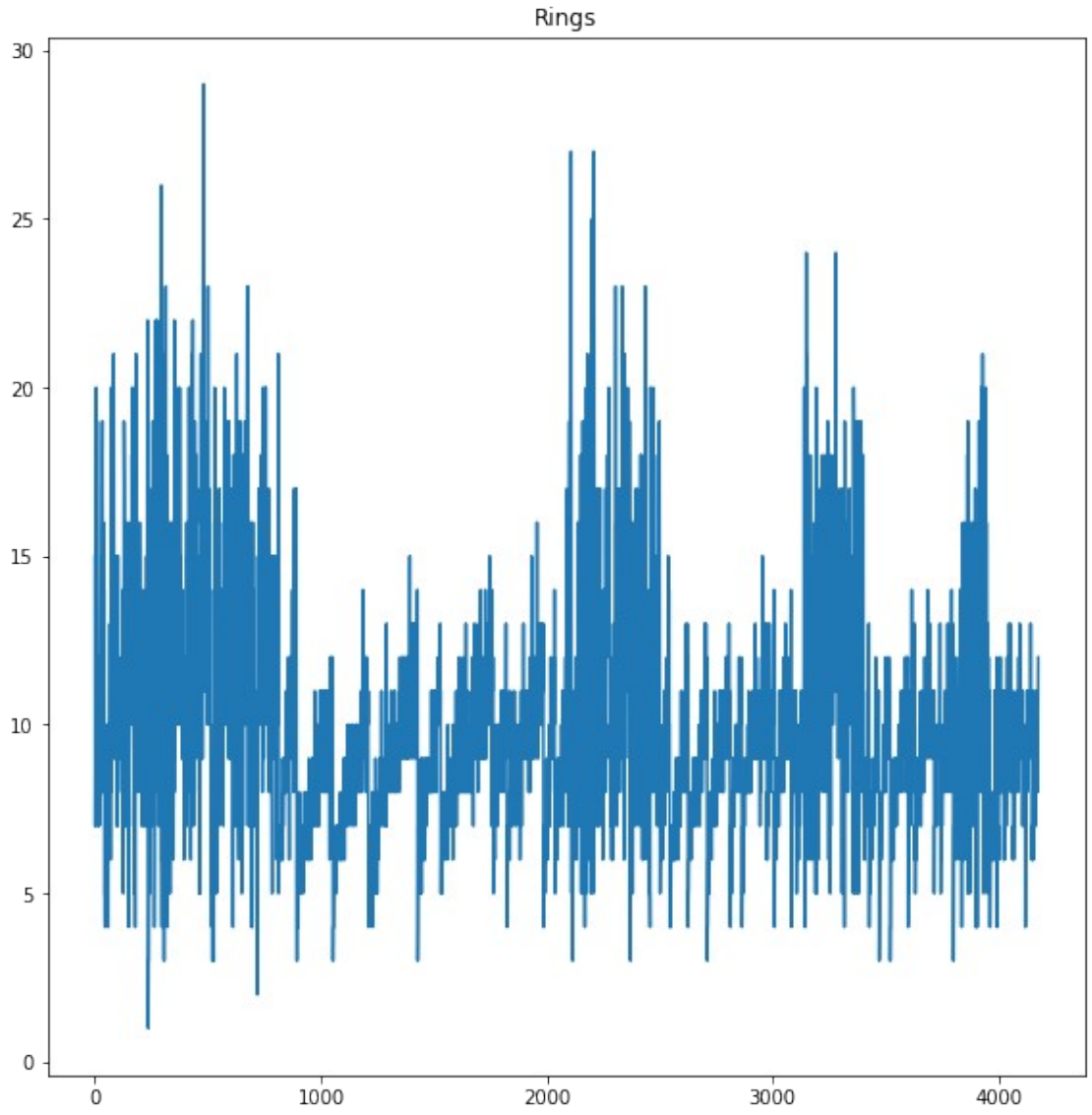

```
sns.displot(ab.Rings,bins=12)
```

```
<seaborn.axisgrid.FacetGrid at 0x7fd21a7b87d0>
```




```
fig = plt.figure(figsize=(5,5))
cx = fig.add_axes([0,0,1.5,1.5])
cx.set_title('Rings')
cx.plot(ab['Rings'])
```

```
[<matplotlib.lines.Line2D at 0x7fd21a6c0e50>]
```



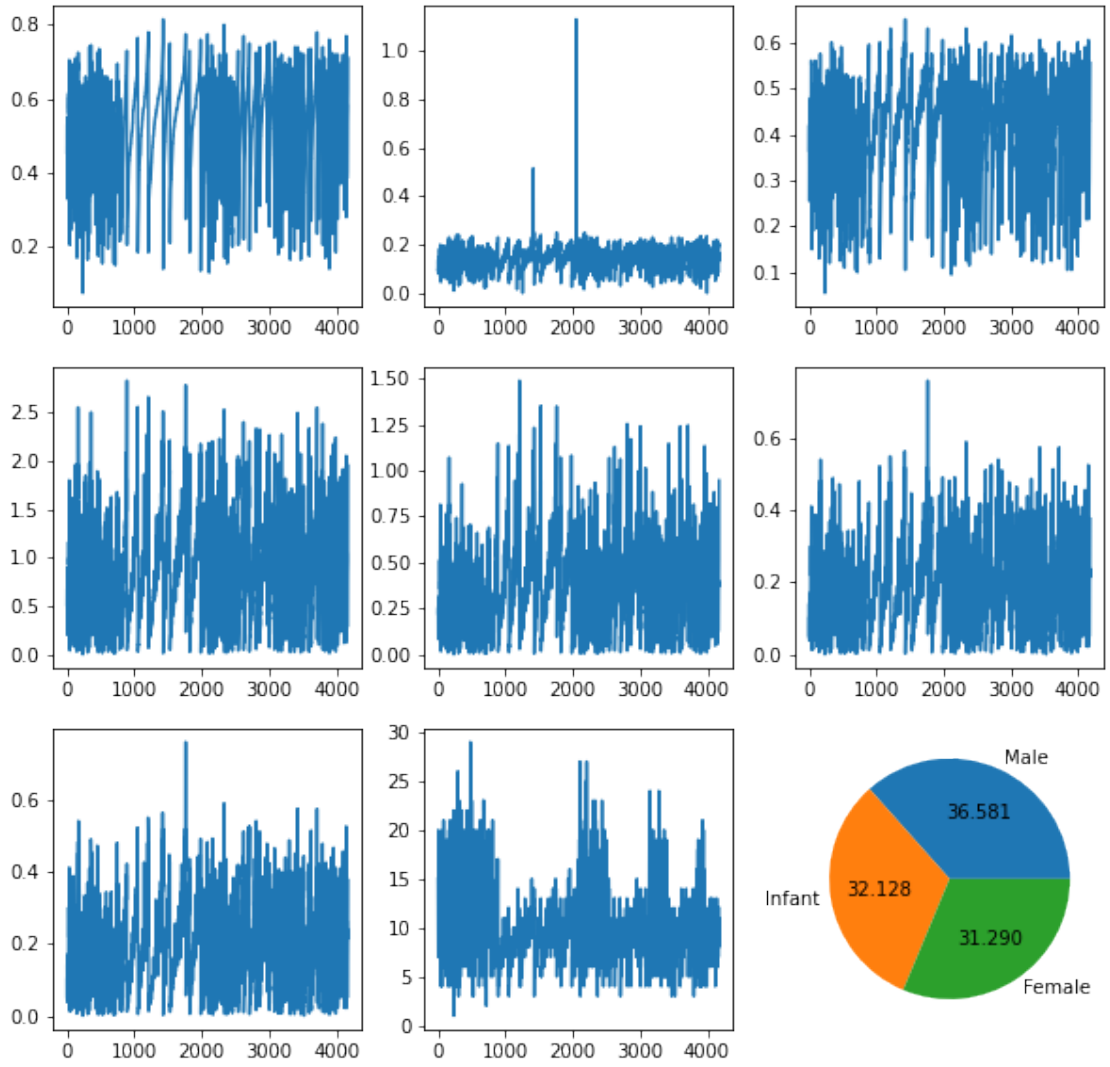

```
sex_count = pd.value_counts(ab.Sex)
sex_count
```

```
2    1528
1    1342
0    1307
Name: Sex, dtype: int64
```

```
rcParams['figure.figsize']= 10,10
```

```
fig,ax = plt.subplots(3,3)
ax[0,0].plot(ab.Length)
ax[0,1].plot(ab.Height)
ax[0,2].plot(ab.Diameter)
ax[1,0].plot(ab['Whole weight'])
ax[1,1].plot(ab['Shucked weight'])
ax[1,2].plot(ab['Viscera weight'])
```

```
ax[2,0].plot(ab['Viscera weight'])
ax[2,1].plot(ab.Rings)
ax[2,2].pie(sex_count,labels=['Male','Infant','Female'],autopct='%0.3f
')
plt.show()
```



Bi-Variate Analysis

```
ab.head()
```

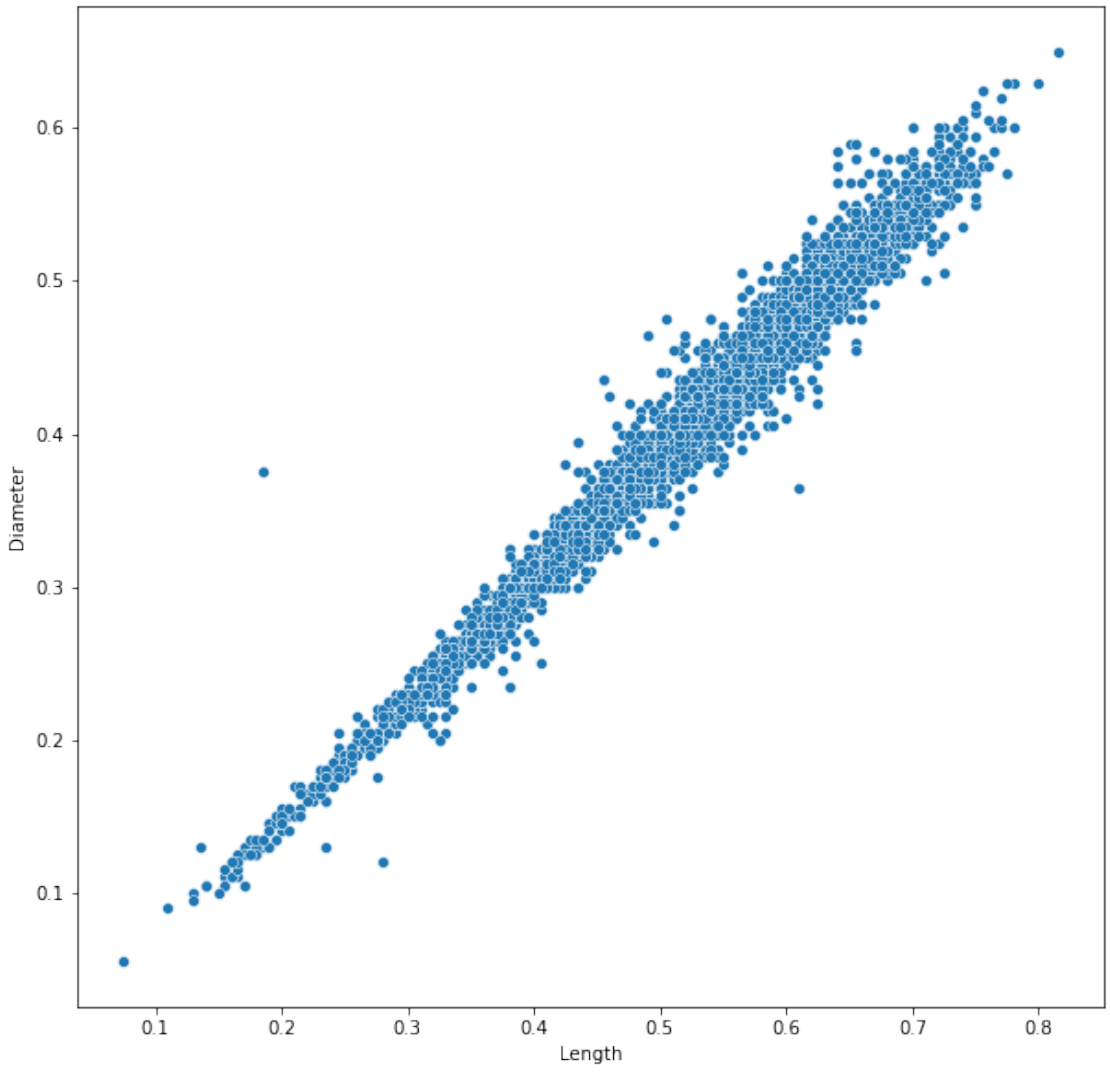
```
   Sex  Length  Diameter  Height  Whole weight  Shucked weight  \
0    2   0.455     0.365   0.095        0.5140          0.2245
1    2   0.350     0.265   0.090        0.2255          0.0995
2    0   0.530     0.420   0.135        0.6770          0.2565
3    2   0.440     0.365   0.125        0.5160          0.2155
4    1   0.330     0.255   0.080        0.2050          0.0895

   Viscera weight  Shell weight  Rings
```

```
0          0.1010          0.150     15
1          0.0485          0.070      7
2          0.1415          0.210      9
3          0.1140          0.155     10
4          0.0395          0.055      7
```

```
sns.scatterplot(ab.Length,ab.Diameter)
```

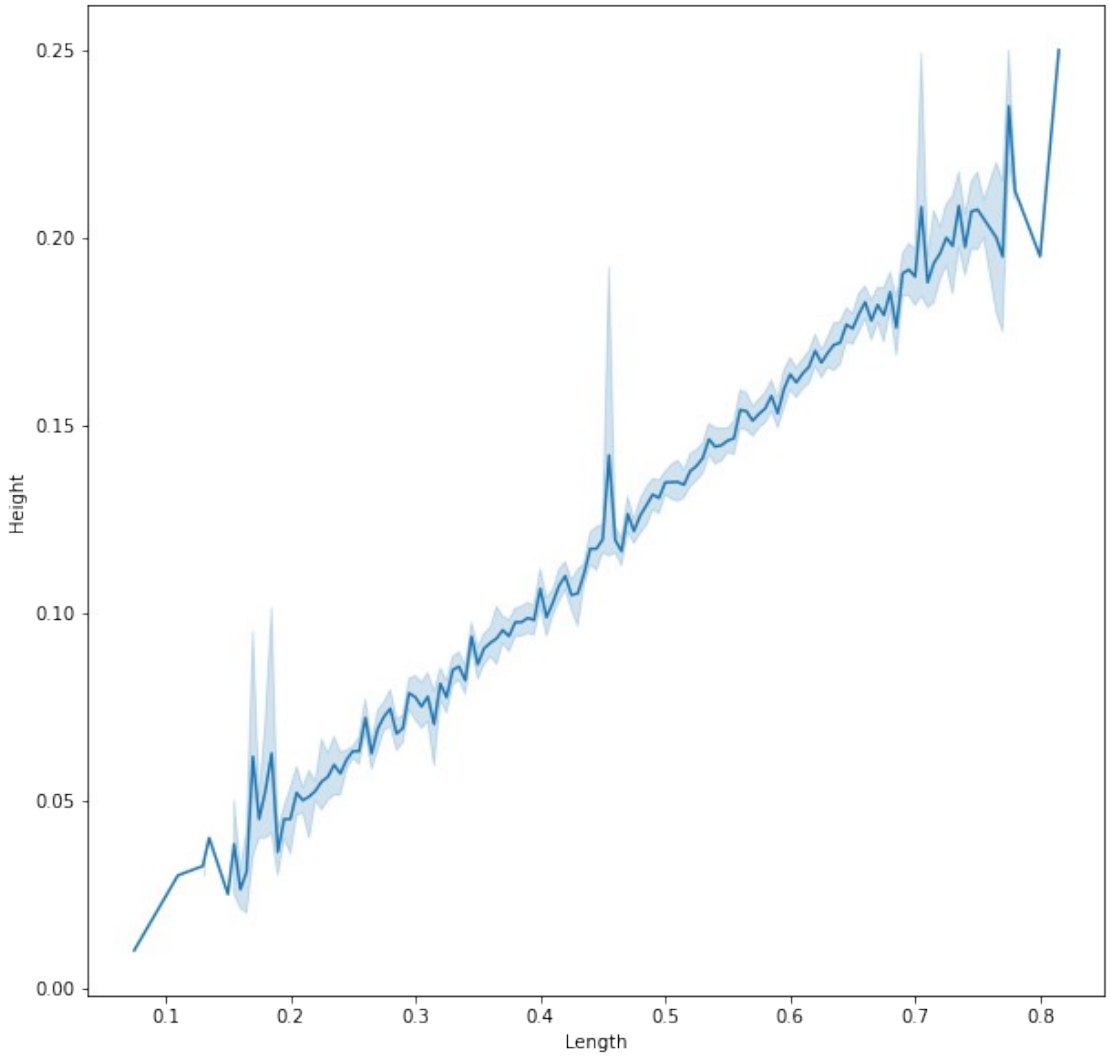
```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43:
FutureWarning: Pass the following variables as keyword args: x, y.
From version 0.12, the only valid positional argument will be `data`,
and passing other arguments without an explicit keyword will result in
an error or misinterpretation.
  FutureWarning
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fd21a60f150>
```

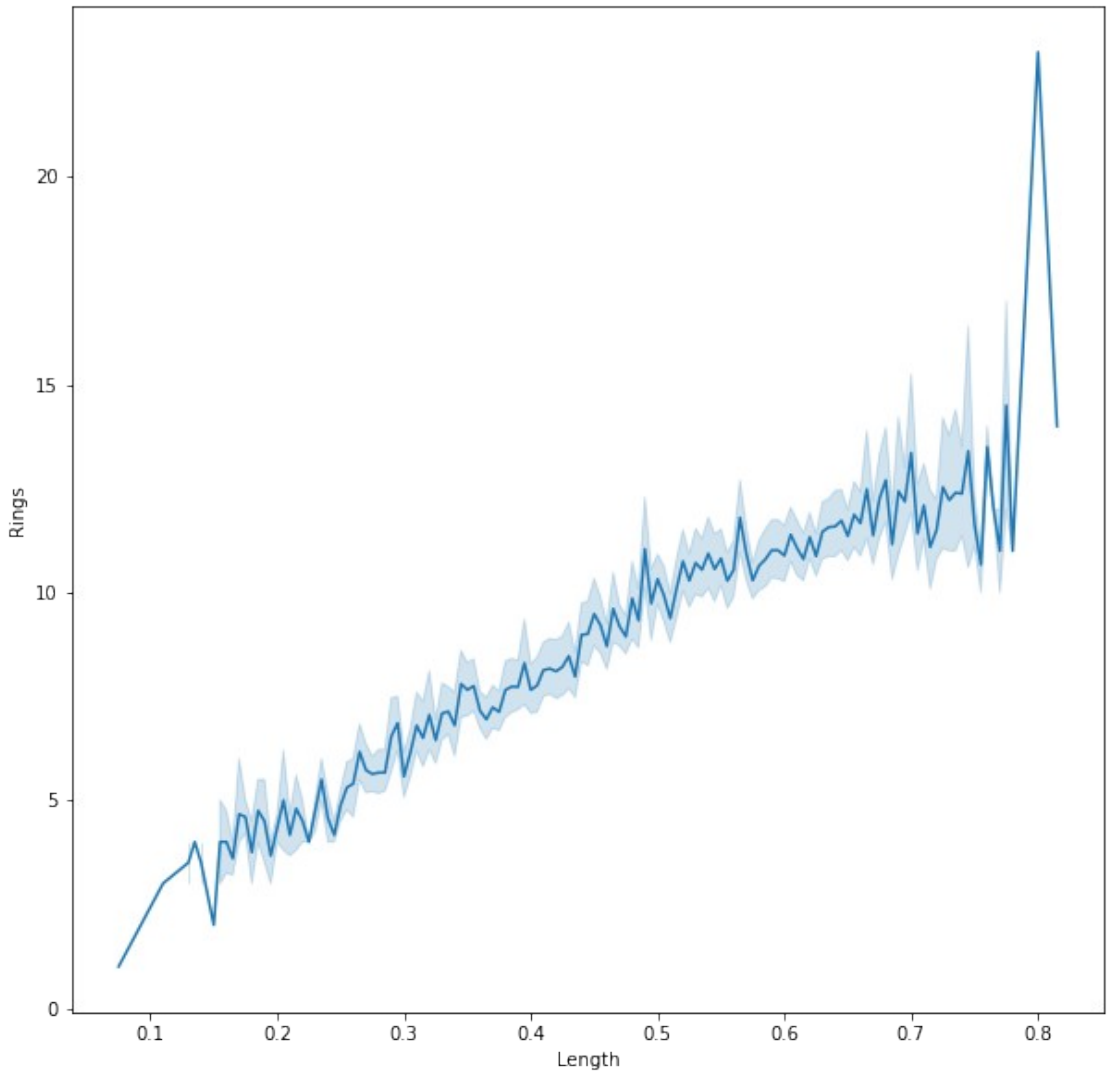



```
sns.lineplot(ab.Length,ab.Height)
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43:
FutureWarning: Pass the following variables as keyword args: x, y.
From version 0.12, the only valid positional argument will be `data`,
and passing other arguments without an explicit keyword will result in
an error or misinterpretation.
  FutureWarning

<matplotlib.axes._subplots.AxesSubplot at 0x7fd21a60fad0>
```



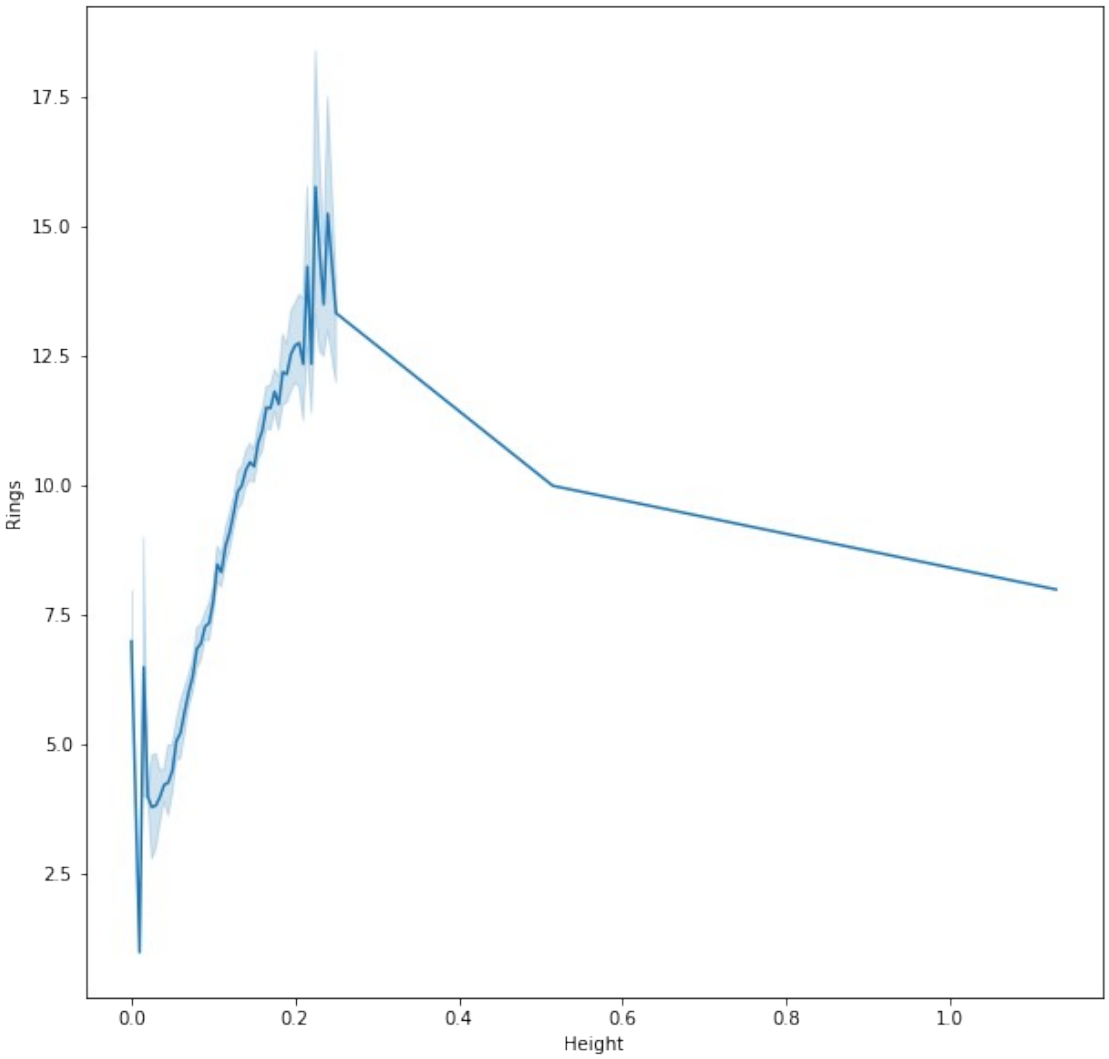

```
sns.lineplot(ab.Length,ab.Rings)

/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43:
FutureWarning: Pass the following variables as keyword args: x, y.
From version 0.12, the only valid positional argument will be `data`,
and passing other arguments without an explicit keyword will result in
an error or misinterpretation.
  FutureWarning

<matplotlib.axes._subplots.AxesSubplot at 0x7fd21a275450>
```




```
sns.lineplot(ab.Height,ab.Rings)
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43:
FutureWarning: Pass the following variables as keyword args: x, y.
From version 0.12, the only valid positional argument will be `data`,
and passing other arguments without an explicit keyword will result in
an error or misinterpretation.
  FutureWarning
```

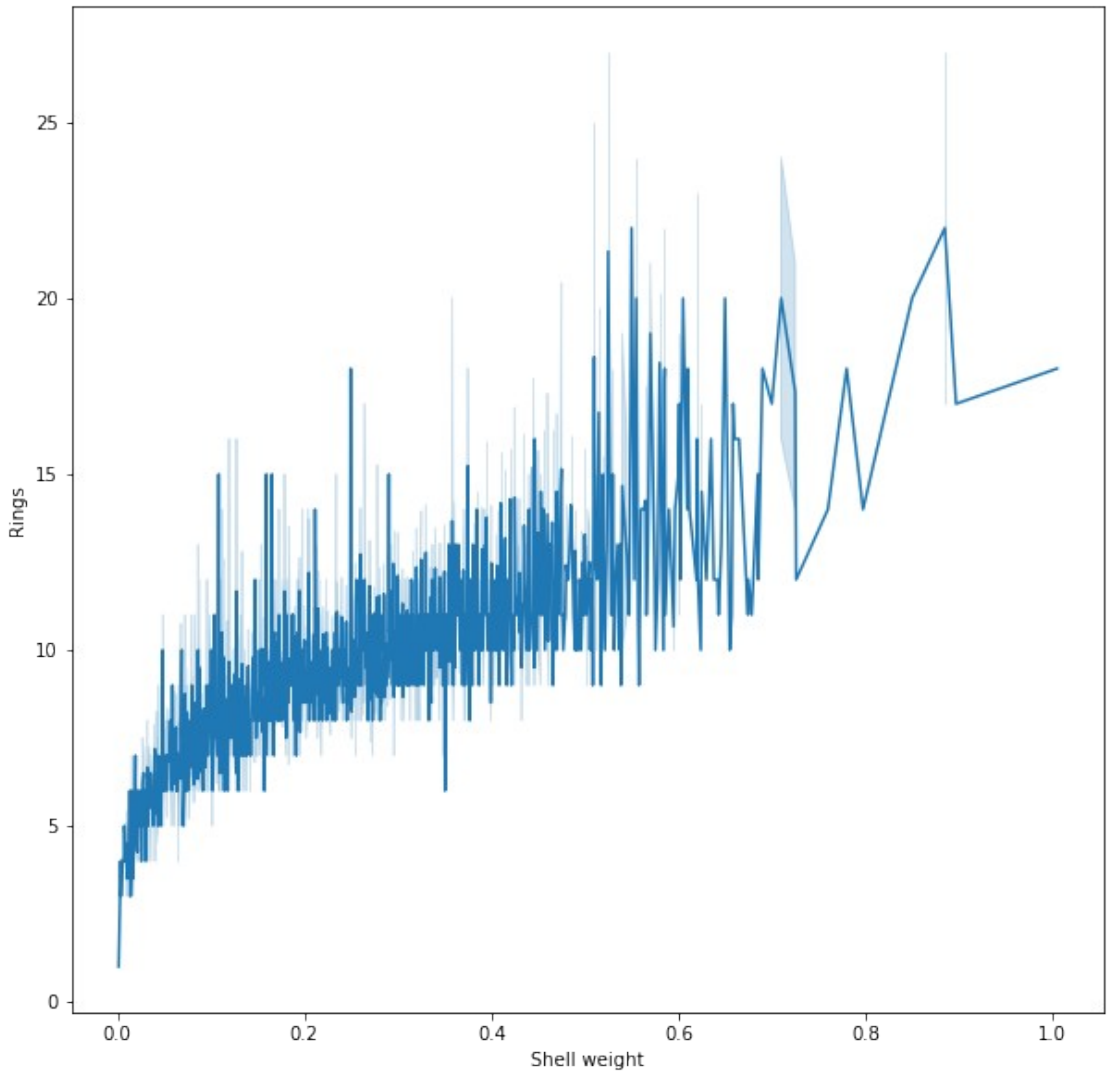
```
<matplotlib.axes._subplots.AxesSubplot at 0x7fd21a130710>
```




```
sns.lineplot(ab['Shell weight'],ab['Rings'])
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43:
FutureWarning: Pass the following variables as keyword args: x, y.
From version 0.12, the only valid positional argument will be `data`,
and passing other arguments without an explicit keyword will result in
an error or misinterpretation.
  FutureWarning
```

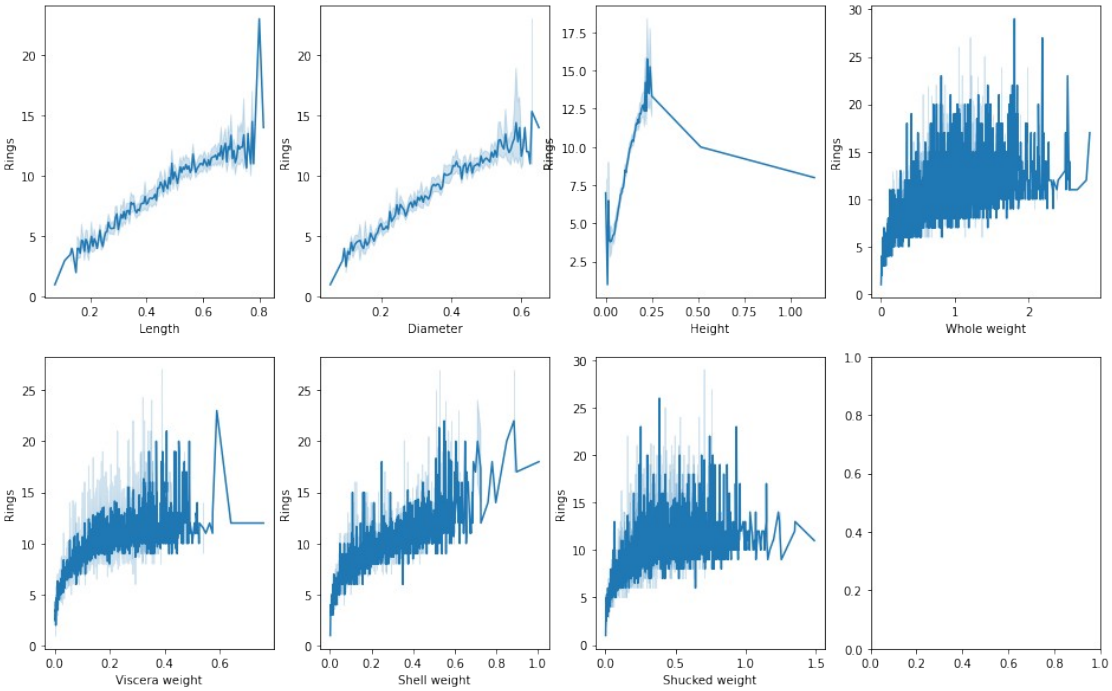
```
<matplotlib.axes._subplots.AxesSubplot at 0x7fd21a219150>
```




```
fig,axes = plt.subplots(2,4,figsize=(16,10))
sns.lineplot(ab.Length,ab.Rings,ax=axes[0,0])
sns.lineplot(ab.Diameter,ab.Rings,ax=axes[0,1])
sns.lineplot(ab.Height,ab.Rings,ax=axes[0,2])
sns.lineplot(ab['Whole weight'],ab.Rings,ax=axes[0,3])
sns.lineplot(ab['Viscera weight'],ab.Rings,ax=axes[1,0])
sns.lineplot(ab['Shell weight'],ab.Rings,ax=axes[1,1])
sns.lineplot(ab['Shucked weight'],ab.Rings,ax=axes[1,2])
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43:
FutureWarning: Pass the following variables as keyword args: x, y.
From version 0.12, the only valid positional argument will be `data`,
and passing other arguments without an explicit keyword will result in
an error or misinterpretation.
  FutureWarning
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43:
FutureWarning: Pass the following variables as keyword args: x, y.
From version 0.12, the only valid positional argument will be `data`,
```

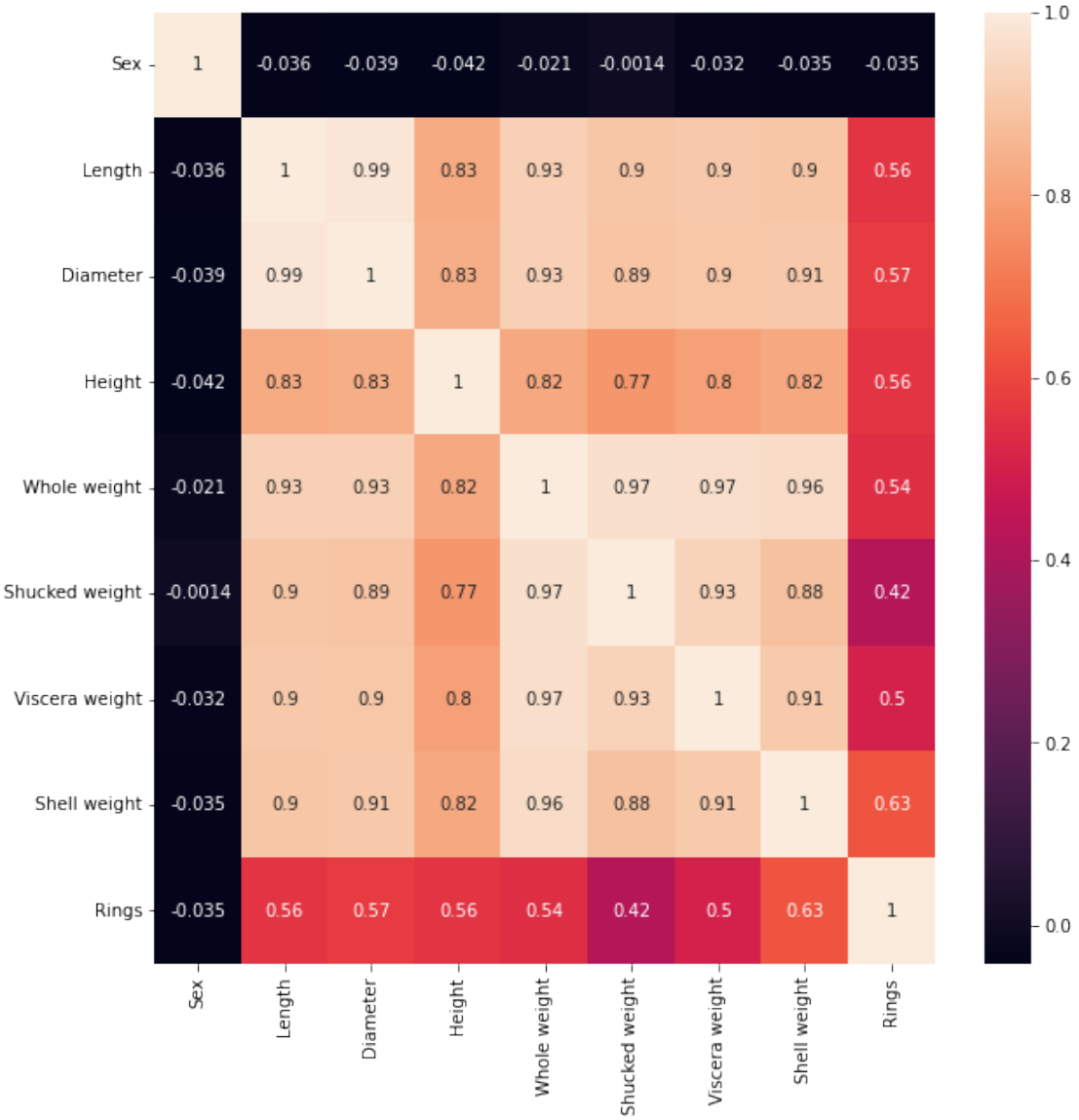
```
and passing other arguments without an explicit keyword will result in
an error or misinterpretation.
  FutureWarning
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43:
FutureWarning: Pass the following variables as keyword args: x, y.
From version 0.12, the only valid positional argument will be `data`,
and passing other arguments without an explicit keyword will result in
an error or misinterpretation.
  FutureWarning
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43:
FutureWarning: Pass the following variables as keyword args: x, y.
From version 0.12, the only valid positional argument will be `data`,
and passing other arguments without an explicit keyword will result in
an error or misinterpretation.
  FutureWarning
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43:
FutureWarning: Pass the following variables as keyword args: x, y.
From version 0.12, the only valid positional argument will be `data`,
and passing other arguments without an explicit keyword will result in
an error or misinterpretation.
  FutureWarning
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43:
FutureWarning: Pass the following variables as keyword args: x, y.
From version 0.12, the only valid positional argument will be `data`,
and passing other arguments without an explicit keyword will result in
an error or misinterpretation.
  FutureWarning
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43:
FutureWarning: Pass the following variables as keyword args: x, y.
From version 0.12, the only valid positional argument will be `data`,
and passing other arguments without an explicit keyword will result in
an error or misinterpretation.
  FutureWarning

<matplotlib.axes._subplots.AxesSubplot at 0x7fd219f22a50>
```



Multivariate Analysis

```
corr = ab.corr()
sns.heatmap(corr,annot=True)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fd219d3f110>
```



Descriptive Statistics on Dataset

```
pd.value_counts(ab.Sex)
```

```
2    1528
1    1342
0    1307
Name: Sex, dtype: int64
```

```
ab.Length.describe()
```

```
count    4177.000000
mean        0.523992
std         0.120093
min         0.075000
25%         0.450000
```

```
50%         0.545000
75%         0.615000
max         0.815000
Name: Length, dtype: float64
```

```
ab.Height.describe()
```

```
count    4177.000000
mean        0.139516
std         0.041827
min         0.000000
25%         0.115000
50%         0.140000
75%         0.165000
max         1.130000
Name: Height, dtype: float64
```

```
ab.Diameter.describe()
```

```
count    4177.000000
mean        0.407881
std         0.099240
min         0.055000
25%         0.350000
50%         0.425000
75%         0.480000
max         0.650000
Name: Diameter, dtype: float64
```

```
ab.Rings.describe()
```

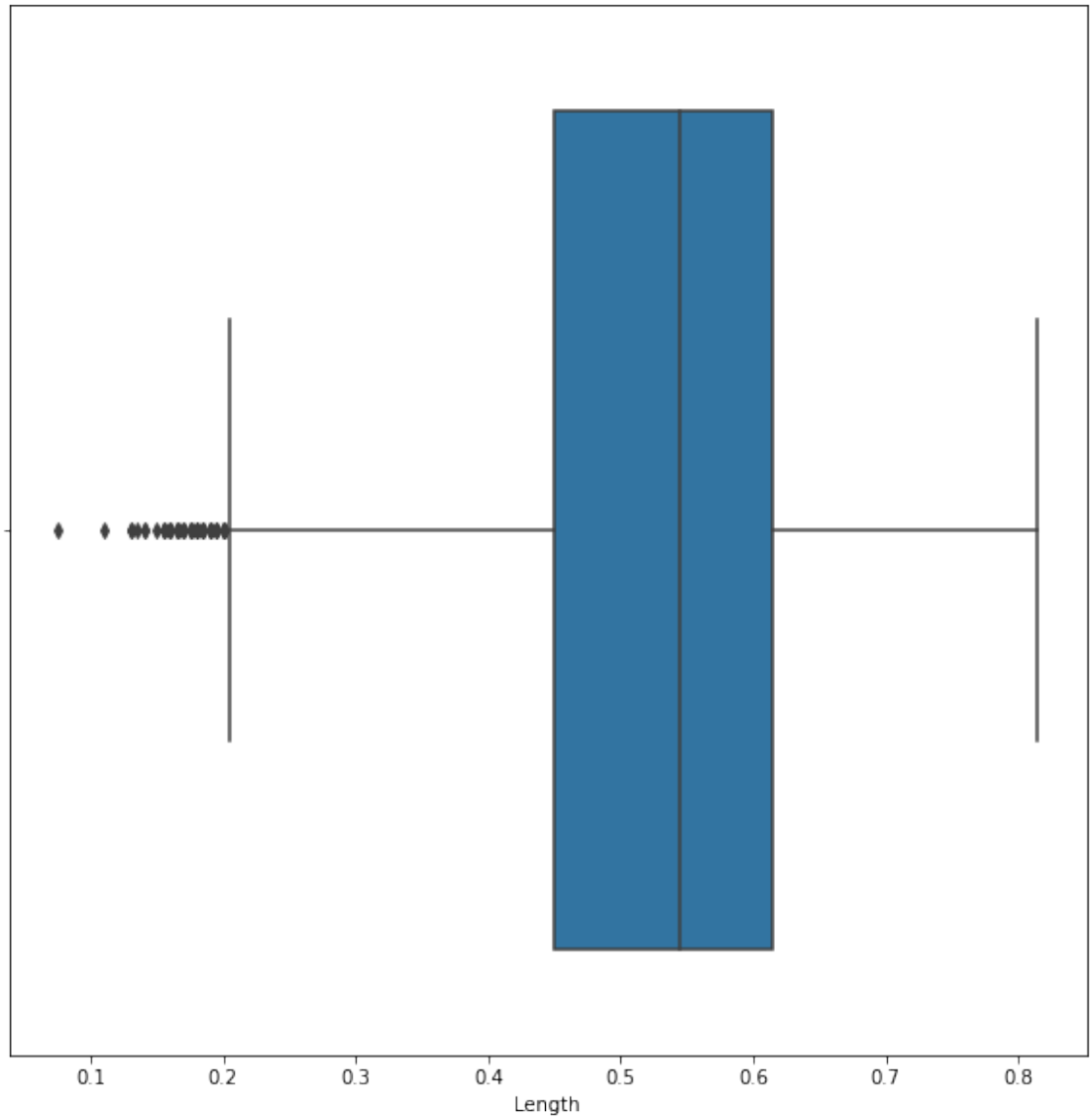
```
count    4177.000000
mean        9.933684
std         3.224169
min         1.000000
25%         8.000000
50%         9.000000
75%        11.000000
max        29.000000
Name: Rings, dtype: float64
```

Check for Outliers and Replace Them

```
sns.boxplot(ab.Length)
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43:
FutureWarning: Pass the following variable as a keyword arg: x. From
version 0.12, the only valid positional argument will be `data`, and
passing other arguments without an explicit keyword will result in an
error or misinterpretation.
  FutureWarning
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fd219ae5ad0>
```



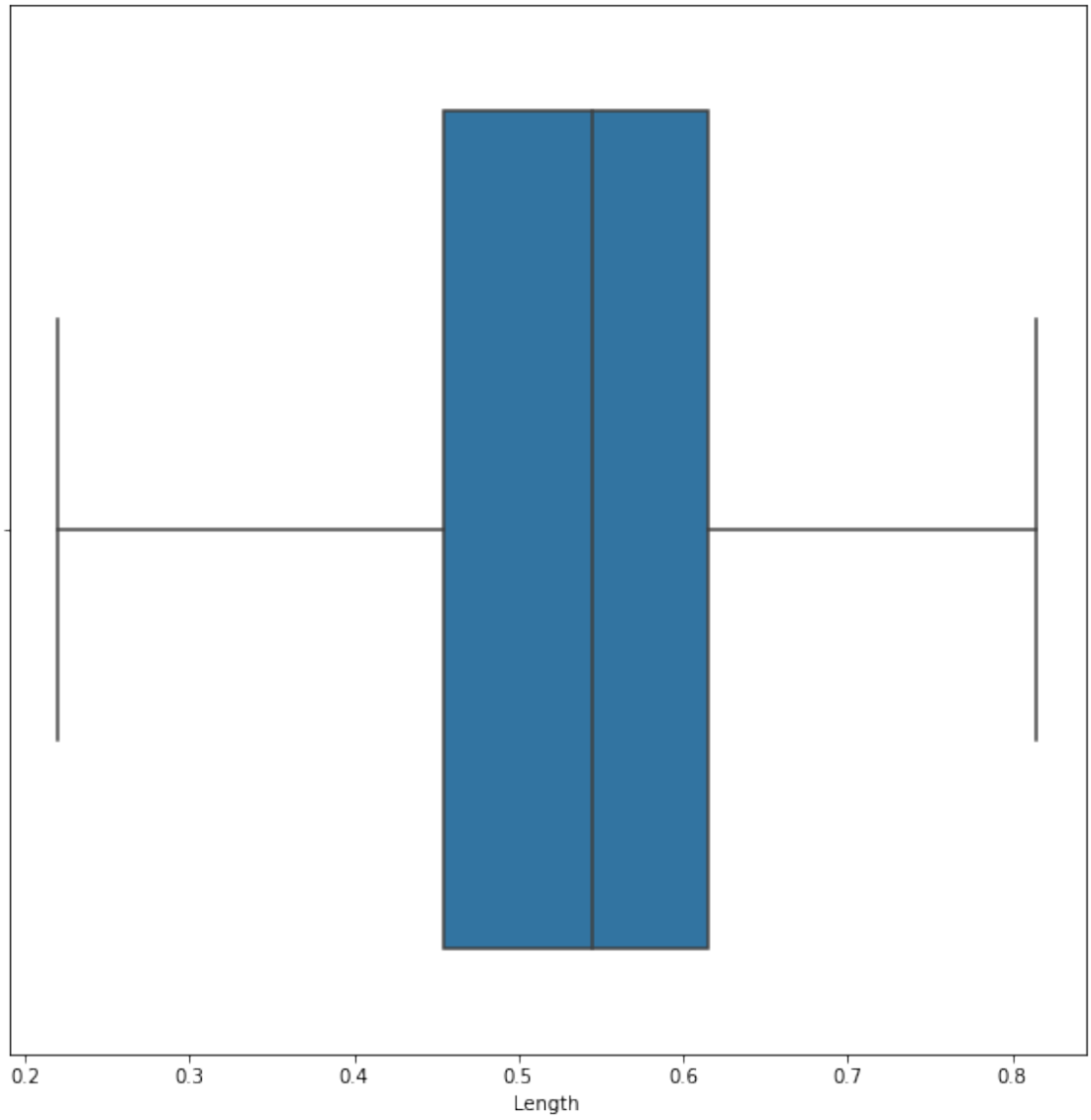

```
p01 = ab.Length.quantile(0.015)
p01
```

```
0.215
```

```
ab = ab[ab.Length > p01]
sns.boxplot(ab.Length)
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43:
FutureWarning: Pass the following variable as a keyword arg: x. From
version 0.12, the only valid positional argument will be `data`, and
passing other arguments without an explicit keyword will result in an
error or misinterpretation.
  FutureWarning
```

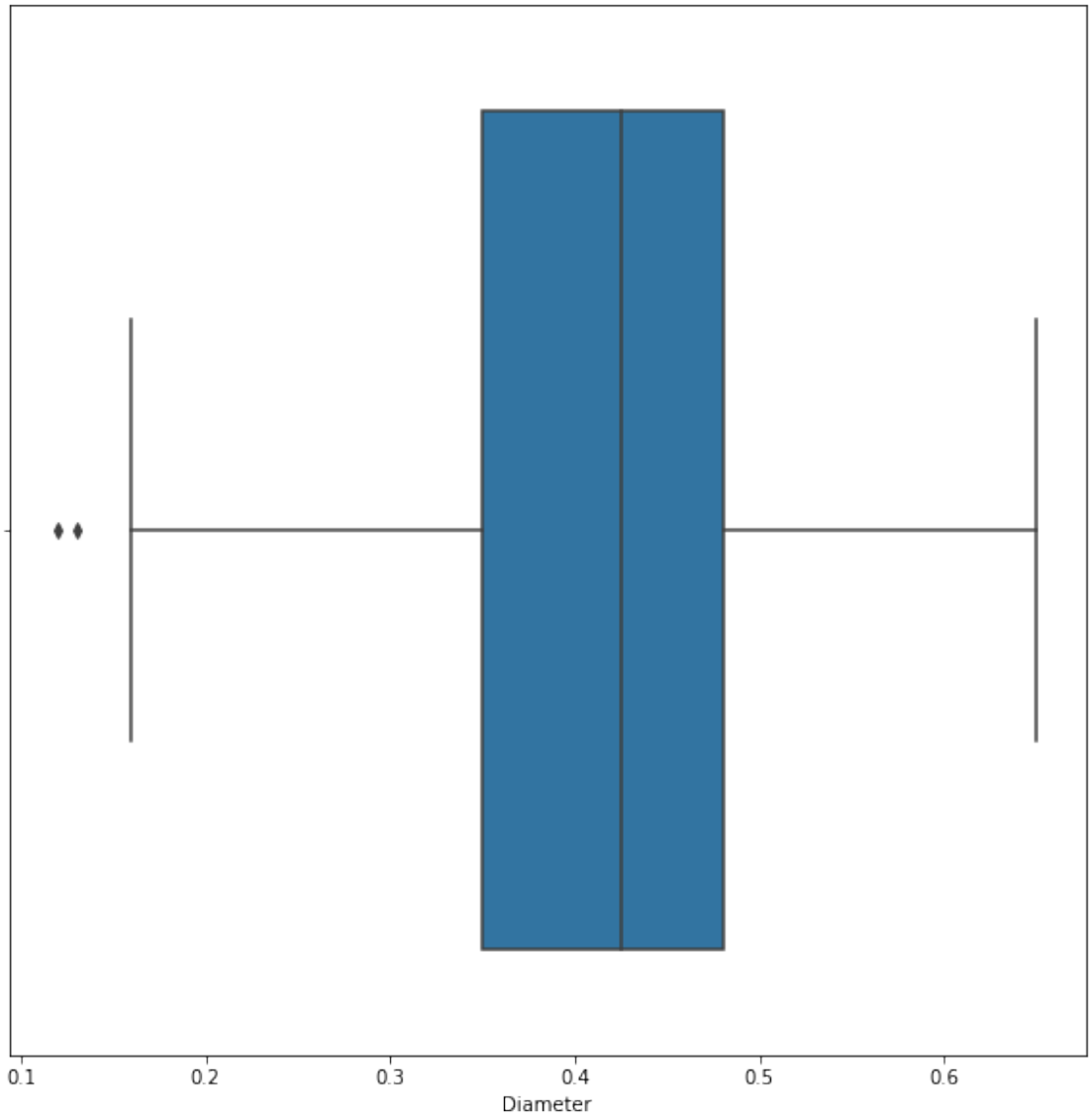
```
<matplotlib.axes._subplots.AxesSubplot at 0x7fd219a9cf10>
```




```
sns.boxplot(ab.Diameter)
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43:
FutureWarning: Pass the following variable as a keyword arg: x. From
version 0.12, the only valid positional argument will be `data`, and
passing other arguments without an explicit keyword will result in an
error or misinterpretation.
  FutureWarning
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fd219dfa510>
```



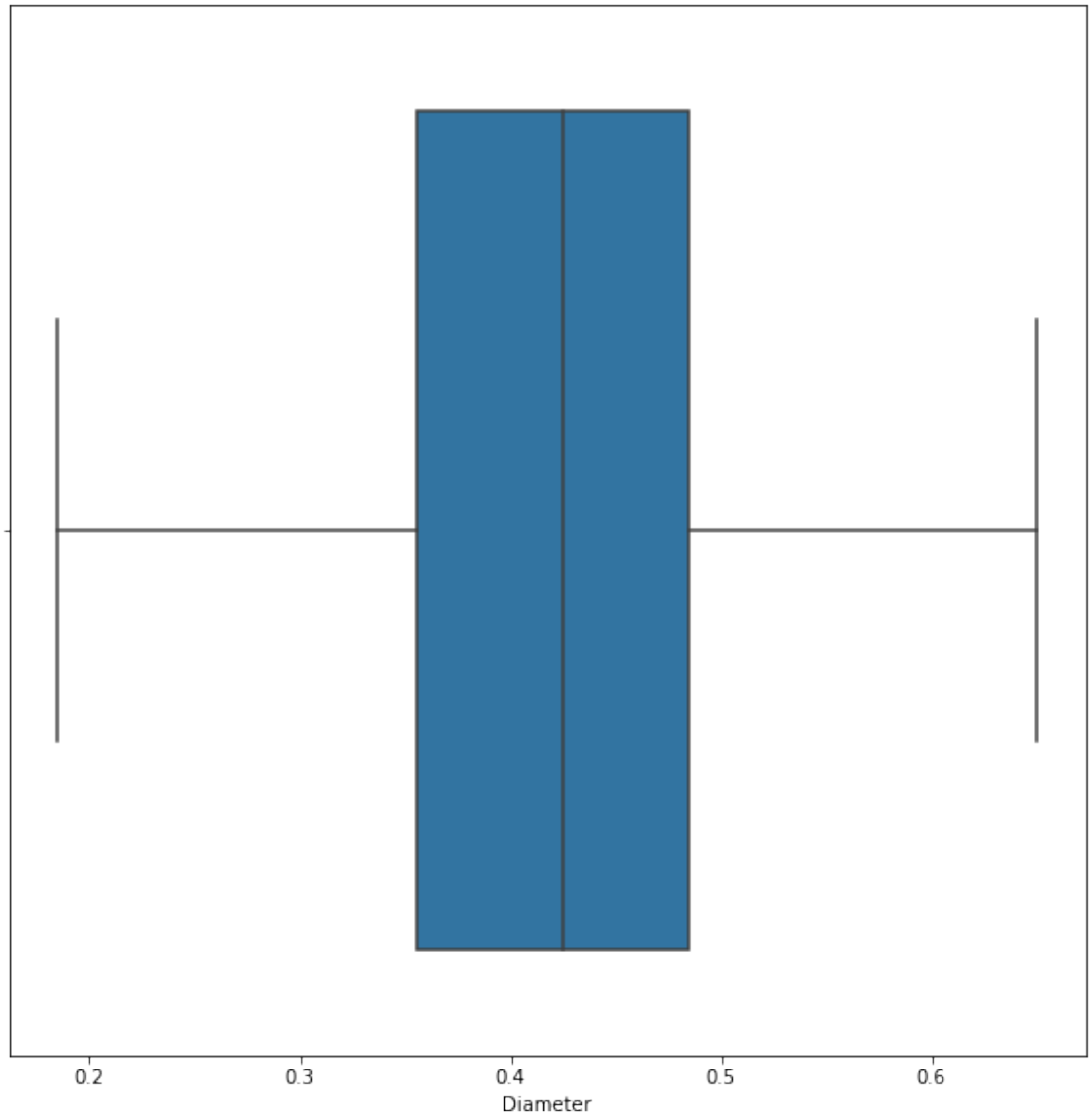

```
p01 = ab.Diameter.quantile(0.01)
p01
```

```
0.185
```

```
ab = ab[ab.Diameter >= p01]
sns.boxplot(ab.Diameter)
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43:
FutureWarning: Pass the following variable as a keyword arg: x. From
version 0.12, the only valid positional argument will be `data`, and
passing other arguments without an explicit keyword will result in an
error or misinterpretation.
  FutureWarning
```

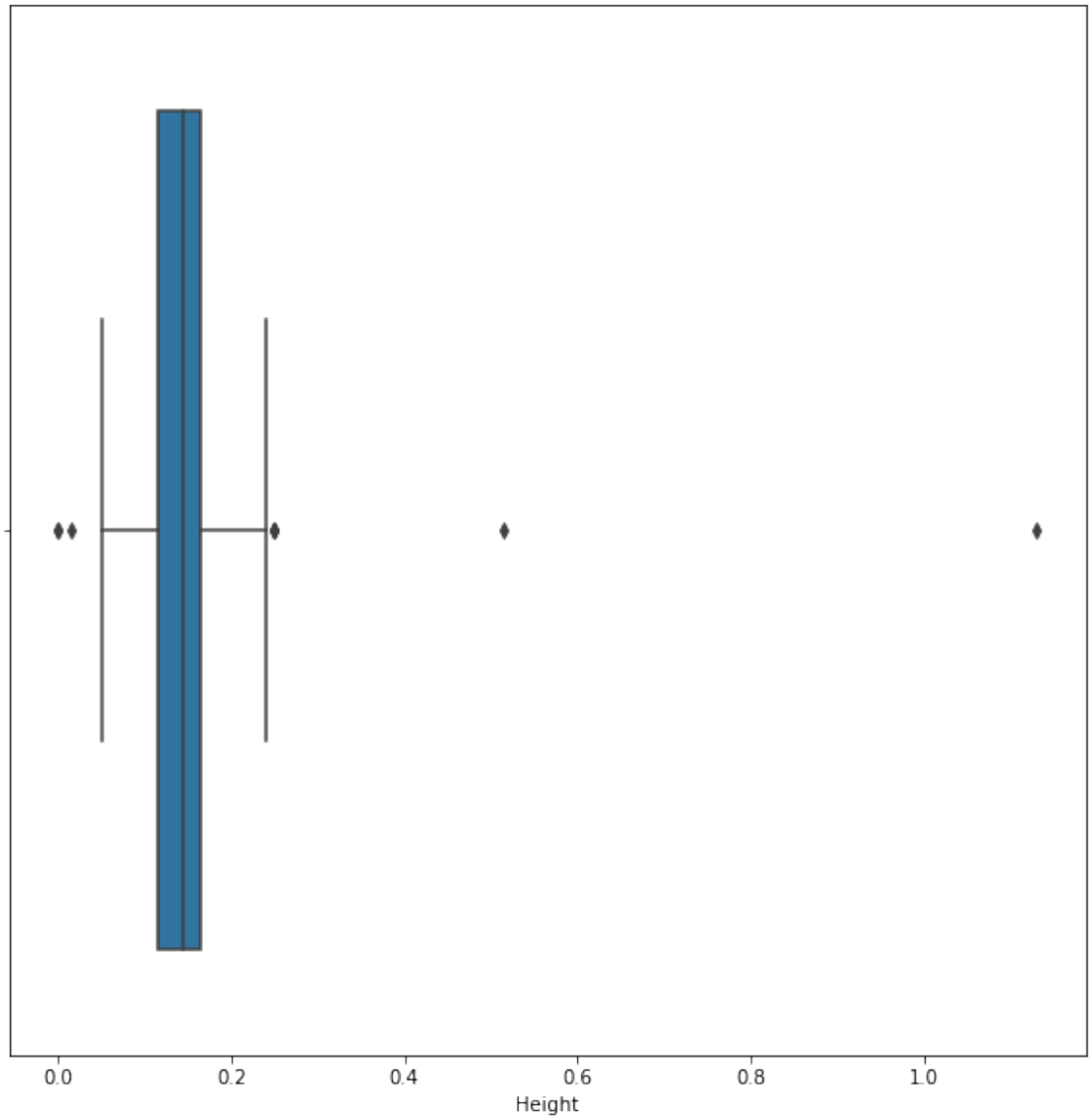
```
<matplotlib.axes._subplots.AxesSubplot at 0x7fd219b8ecd0>
```




```
sns.boxplot(ab.Height)
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43:
FutureWarning: Pass the following variable as a keyword arg: x. From
version 0.12, the only valid positional argument will be `data`, and
passing other arguments without an explicit keyword will result in an
error or misinterpretation.
  FutureWarning
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fd2199e80d0>
```



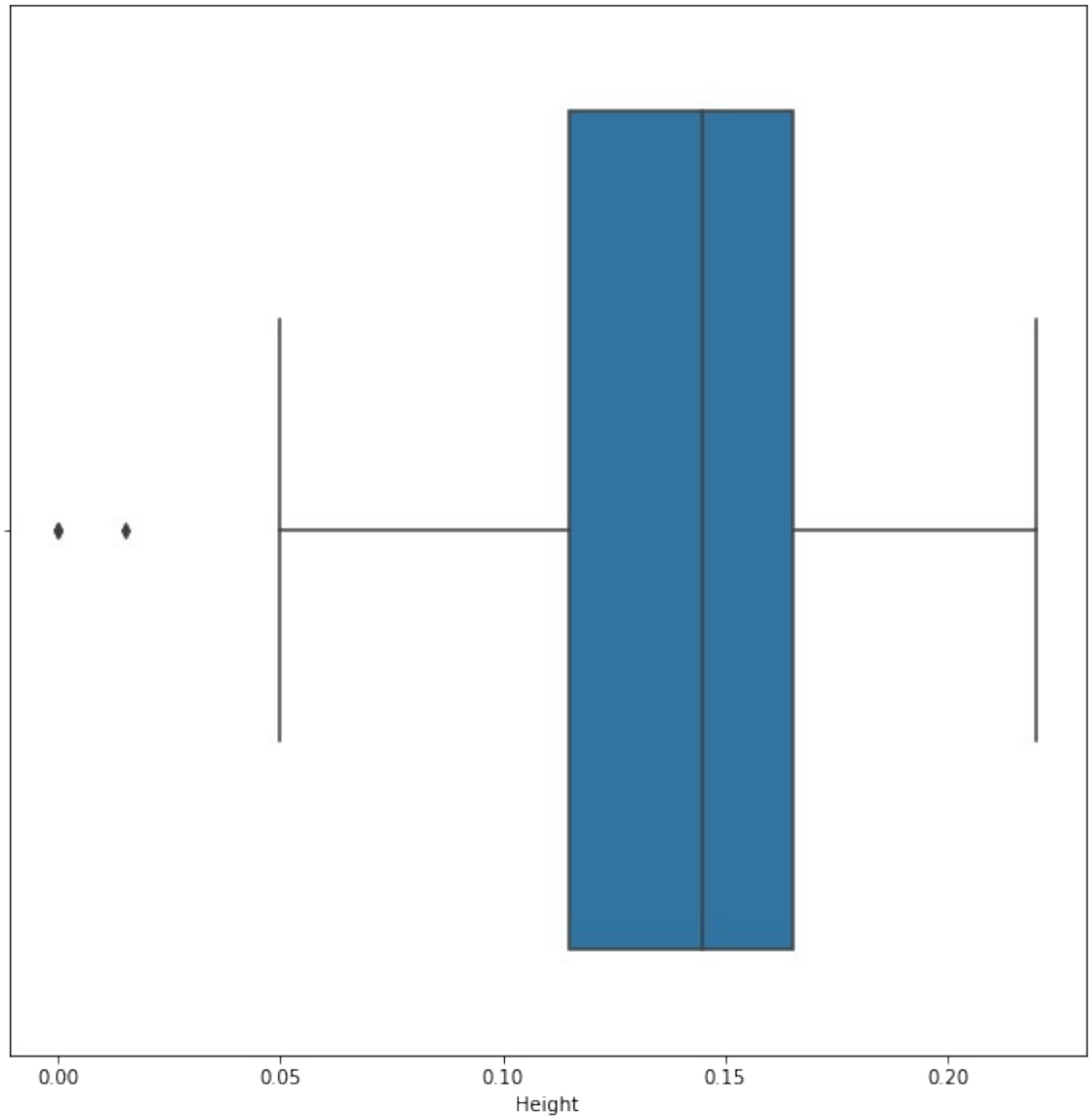

```
p99 = ab.Height.quantile(0.99)
p99
```

```
0.22
```

```
ab = ab[ab.Height <= p99]
sns.boxplot(ab.Height)
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43:
FutureWarning: Pass the following variable as a keyword arg: x. From
version 0.12, the only valid positional argument will be `data`, and
passing other arguments without an explicit keyword will result in an
error or misinterpretation.
  FutureWarning
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fd219dd76d0>
```



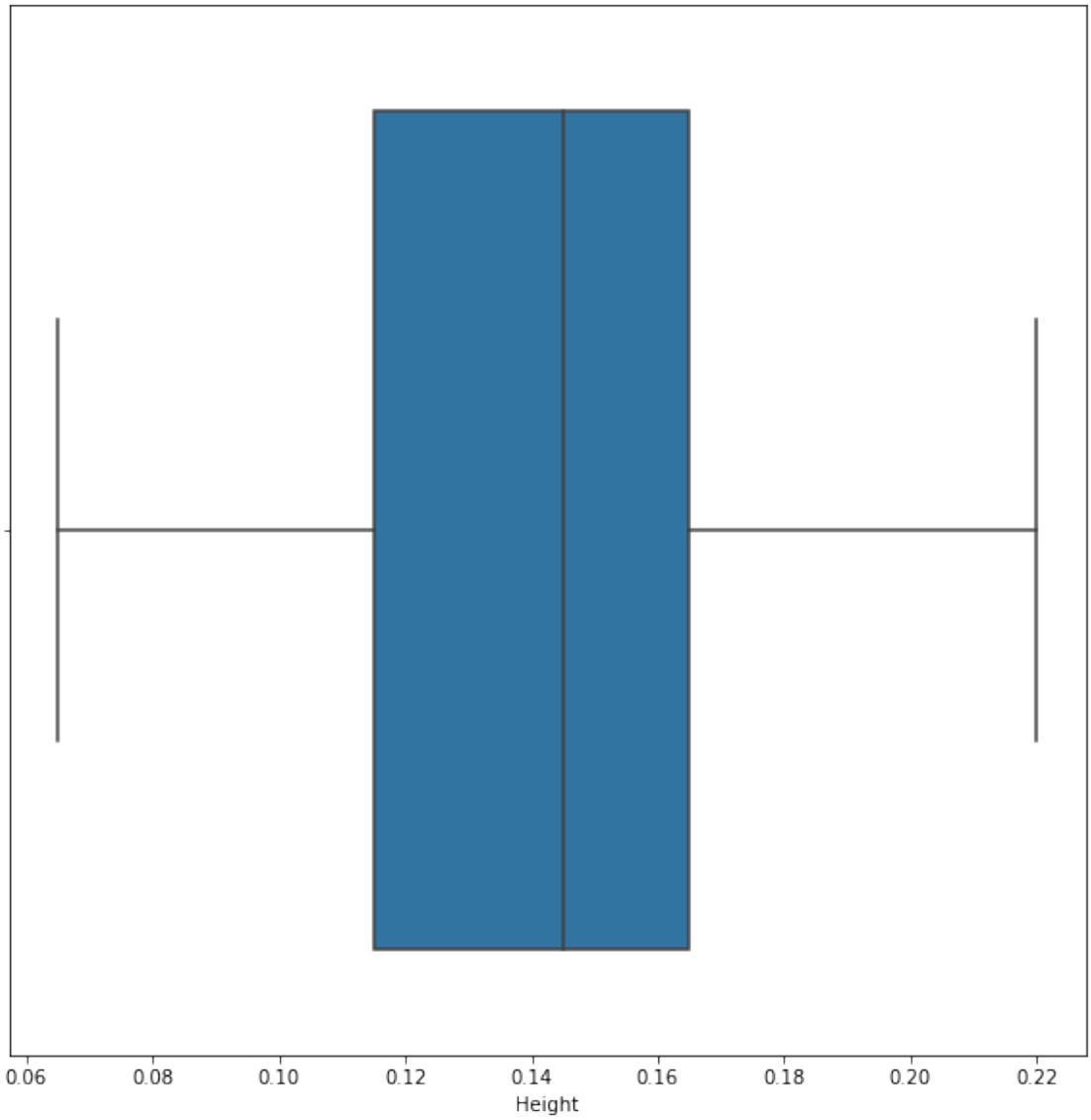

```
p01 = ab.Height.quantile(0.01)
p01
```

```
0.065
```

```
ab = ab[ab.Height >= p01]
sns.boxplot(ab.Height)
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43:
FutureWarning: Pass the following variable as a keyword arg: x. From
version 0.12, the only valid positional argument will be `data`, and
passing other arguments without an explicit keyword will result in an
error or misinterpretation.
  FutureWarning
```

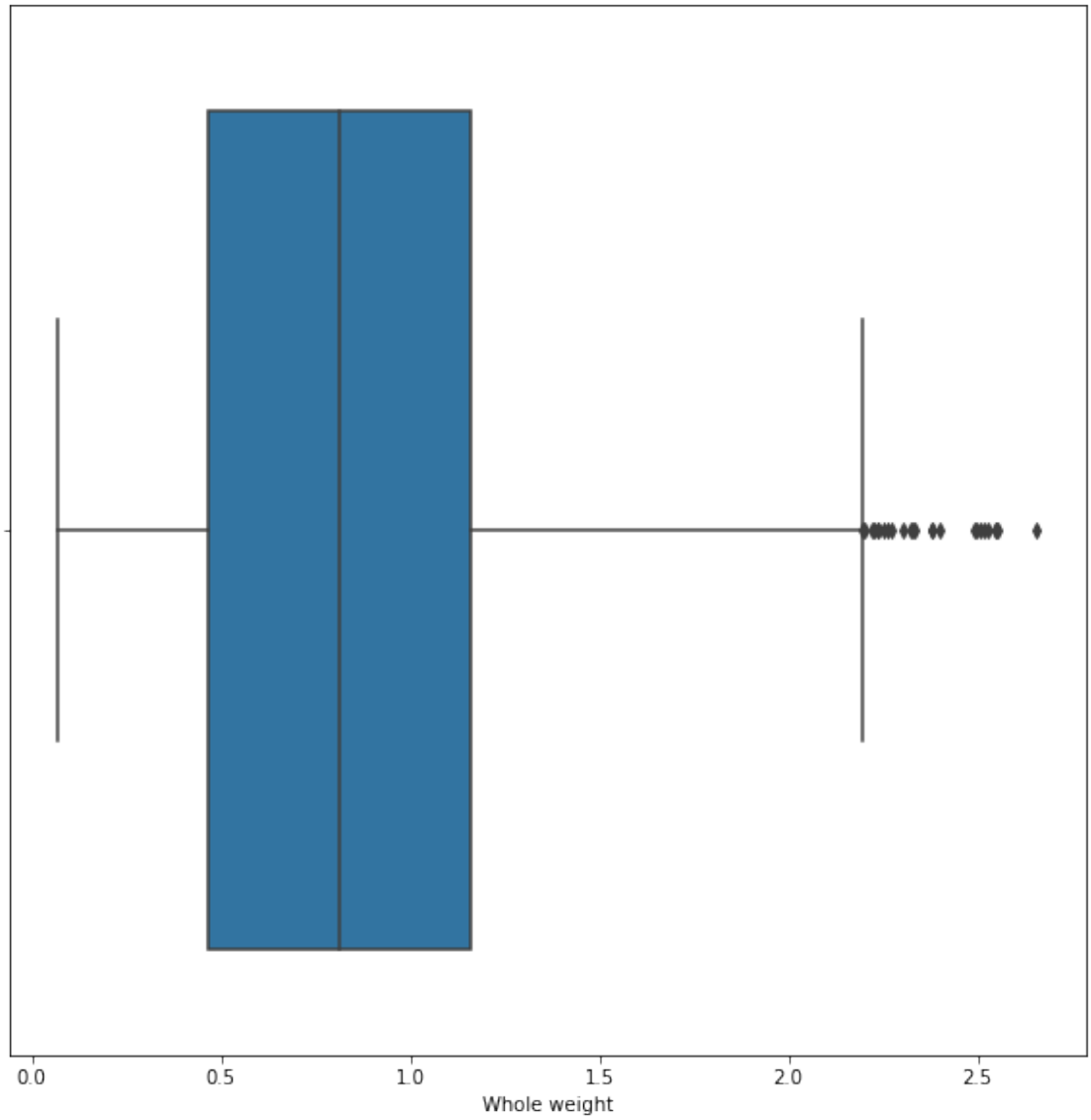
```
<matplotlib.axes._subplots.AxesSubplot at 0x7fd219942cd0>
```




```
sns.boxplot(ab['Whole weight'])
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43:
FutureWarning: Pass the following variable as a keyword arg: x. From
version 0.12, the only valid positional argument will be `data`, and
passing other arguments without an explicit keyword will result in an
error or misinterpretation.
  FutureWarning
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fd2198b6f90>
```



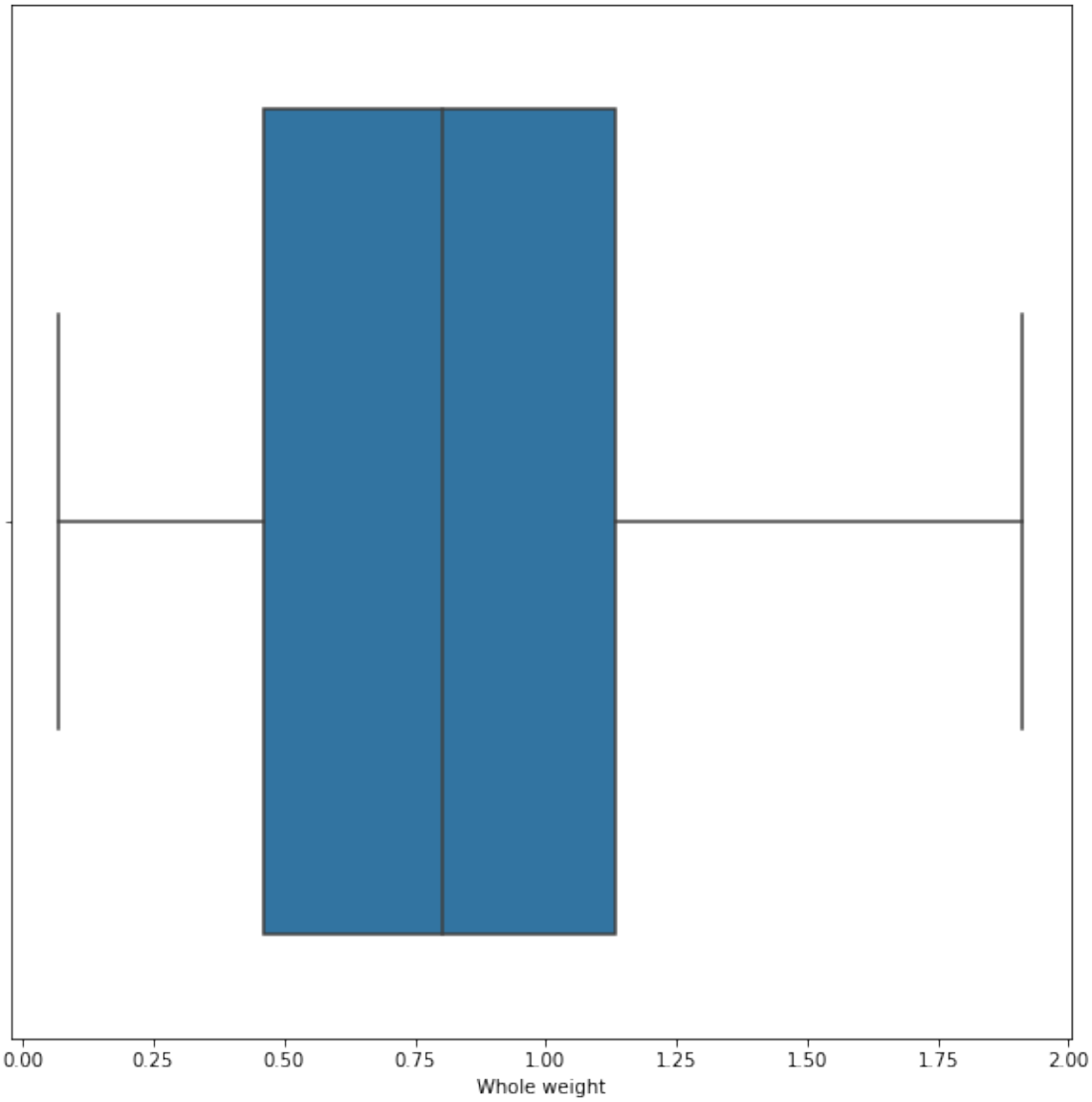

```
p99 = ab['Whole weight'].quantile(0.99)
p99
```

```
1.9130700000000003
```

```
ab = ab[ab['Whole weight'] <= p99]
sns.boxplot(ab['Whole weight'])
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43:
FutureWarning: Pass the following variable as a keyword arg: x. From
version 0.12, the only valid positional argument will be `data`, and
passing other arguments without an explicit keyword will result in an
error or misinterpretation.
  FutureWarning
```

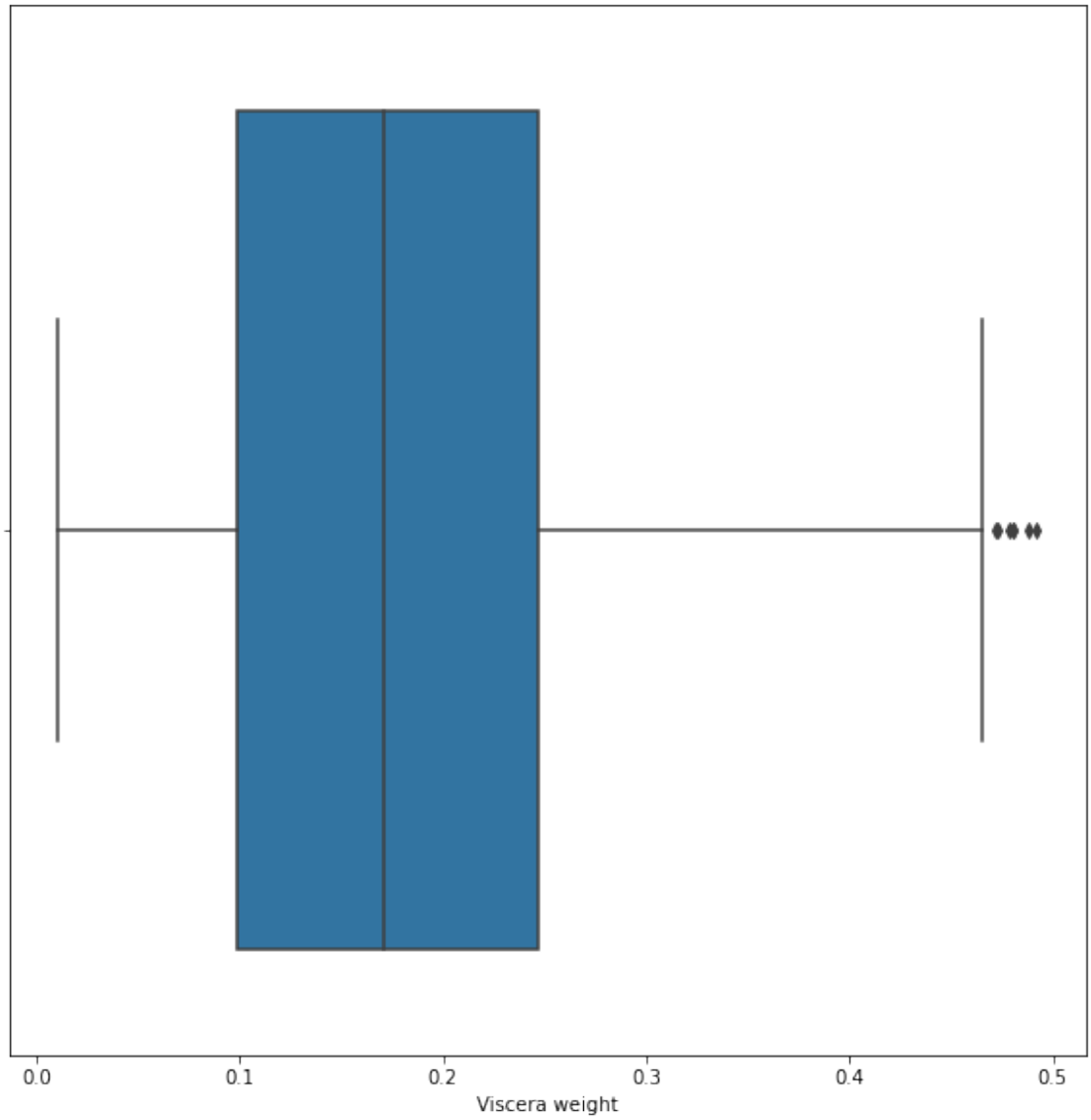
```
<matplotlib.axes._subplots.AxesSubplot at 0x7fd2197ac550>
```




```
sns.boxplot(ab['Viscera weight'])
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43:
FutureWarning: Pass the following variable as a keyword arg: x. From
version 0.12, the only valid positional argument will be `data`, and
passing other arguments without an explicit keyword will result in an
error or misinterpretation.
  FutureWarning
```

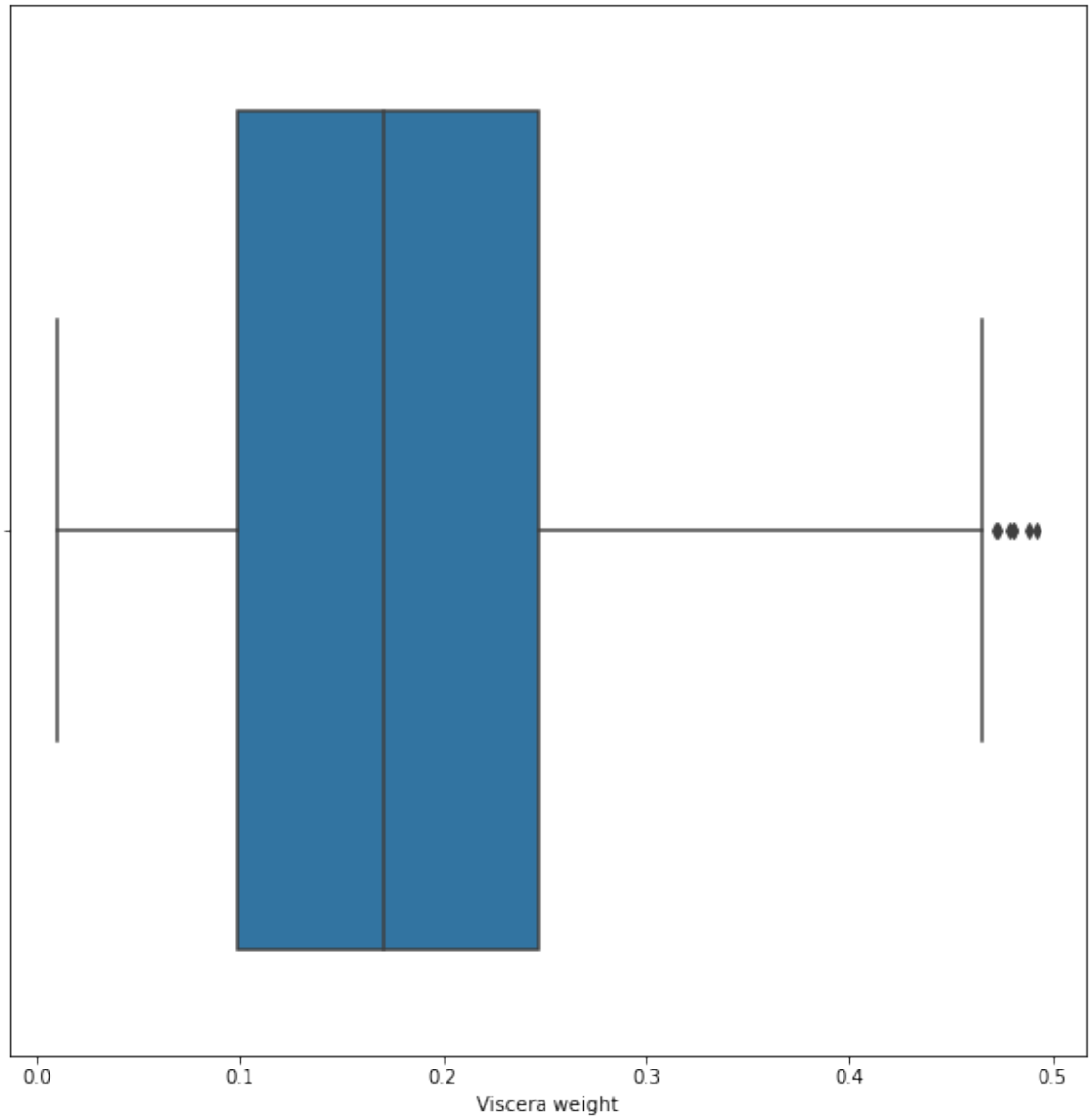
```
<matplotlib.axes._subplots.AxesSubplot at 0x7fd219714b90>
```




```
p99 = ab['Viscera weight'].quantile(0.99)
p99ab = ab[ab['Viscera weight'] <= p99]
sns.boxplot(ab['Viscera weight'])
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43:
FutureWarning: Pass the following variable as a keyword arg: x. From
version 0.12, the only valid positional argument will be `data`, and
passing other arguments without an explicit keyword will result in an
error or misinterpretation.
  FutureWarning
```

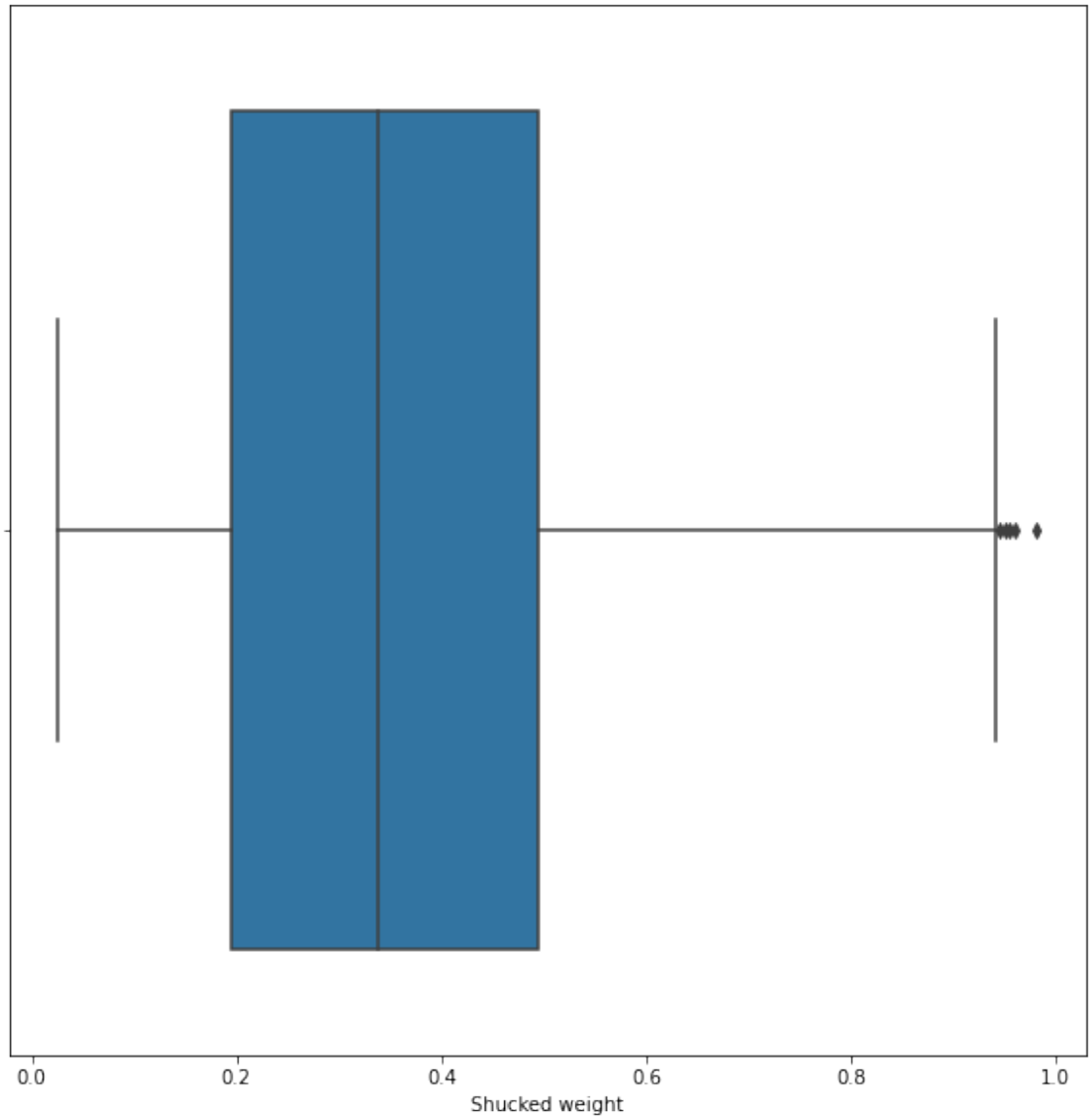
```
<matplotlib.axes._subplots.AxesSubplot at 0x7fd2196f1f10>
```




```
sns.boxplot(ab['Shucked weight'])
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43:
FutureWarning: Pass the following variable as a keyword arg: x. From
version 0.12, the only valid positional argument will be `data`, and
passing other arguments without an explicit keyword will result in an
error or misinterpretation.
  FutureWarning
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fd21966eb50>
```



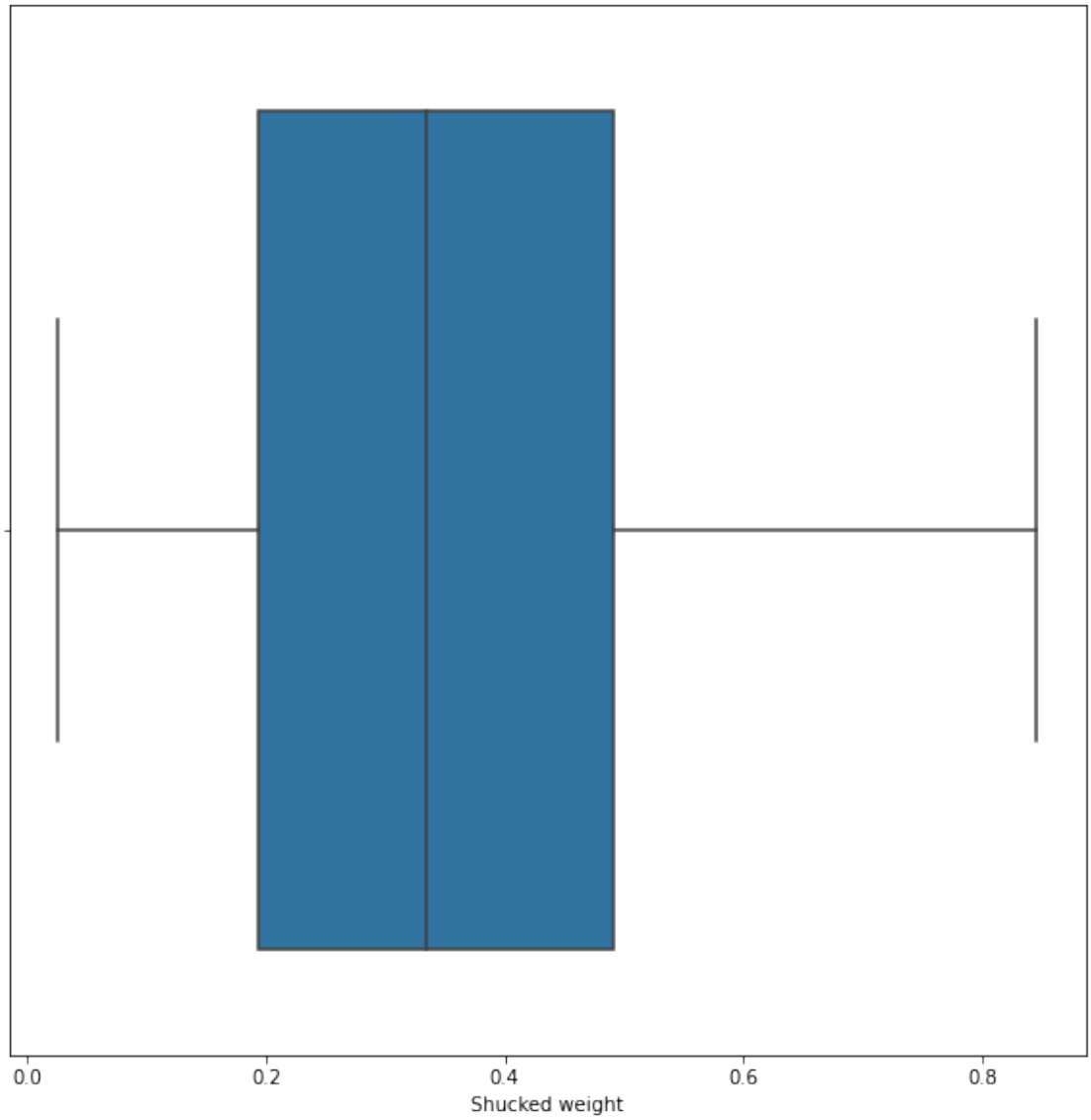

```
p99 = ab['Shucked weight'].quantile(0.99)
p99
```

```
0.8462799999999998
```

```
ab = ab[ab['Shucked weight'] <= p99]
sns.boxplot(ab['Shucked weight'])
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43:
FutureWarning: Pass the following variable as a keyword arg: x. From
version 0.12, the only valid positional argument will be `data`, and
passing other arguments without an explicit keyword will result in an
error or misinterpretation.
  FutureWarning
```

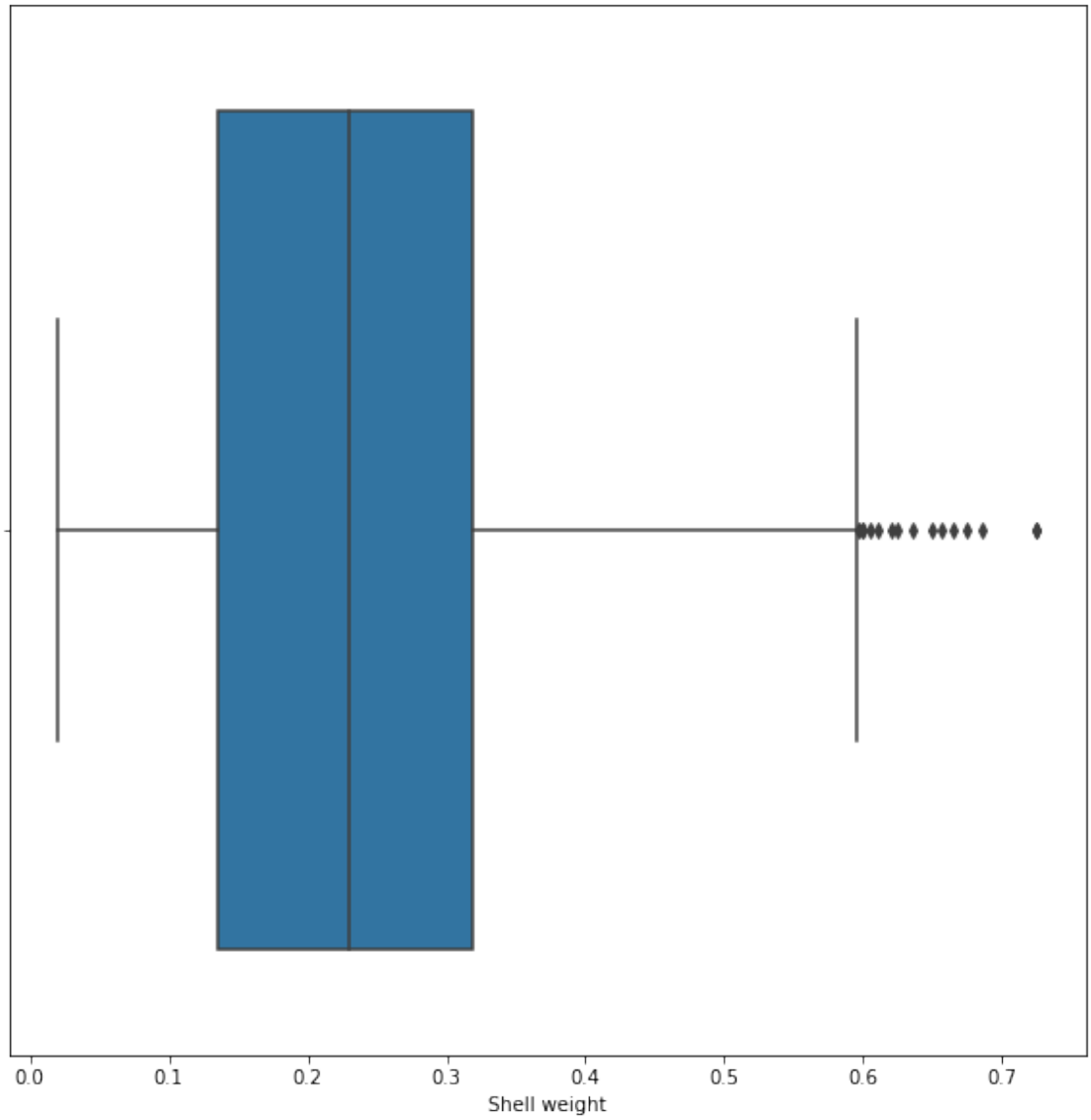
```
<matplotlib.axes._subplots.AxesSubplot at 0x7fd2195e6d90>
```




```
sns.boxplot(ab['Shell weight'])
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43:
FutureWarning: Pass the following variable as a keyword arg: x. From
version 0.12, the only valid positional argument will be `data`, and
passing other arguments without an explicit keyword will result in an
error or misinterpretation.
  FutureWarning
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fd219598310>
```



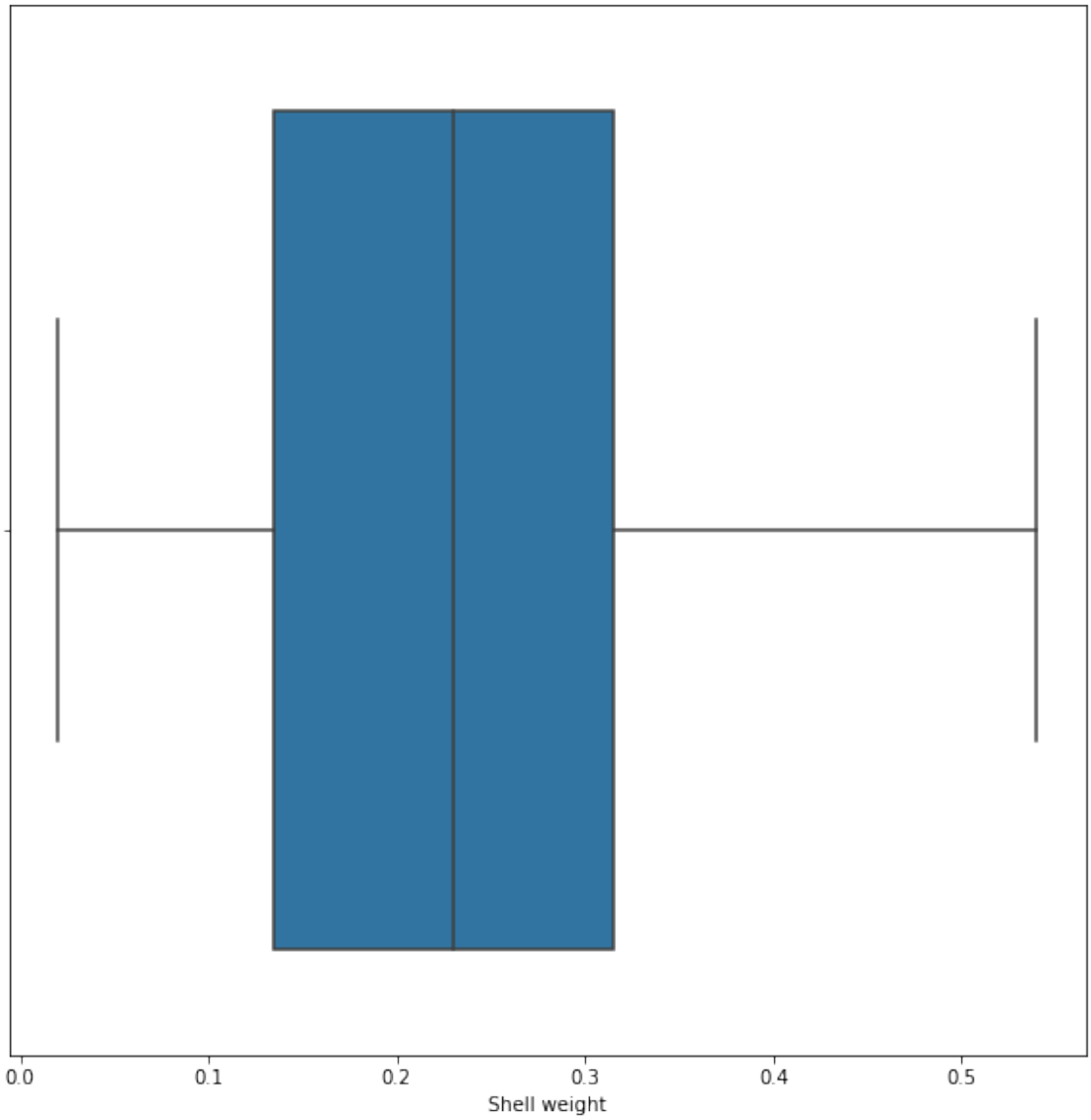

```
p99 = ab['Shell weight'].quantile(0.99)
p99
```

```
0.54
```

```
ab = ab[ab['Shell weight'] <= p99]
sns.boxplot(ab['Shell weight'])
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43:
FutureWarning: Pass the following variable as a keyword arg: x. From
version 0.12, the only valid positional argument will be `data`, and
passing other arguments without an explicit keyword will result in an
error or misinterpretation.
  FutureWarning
```

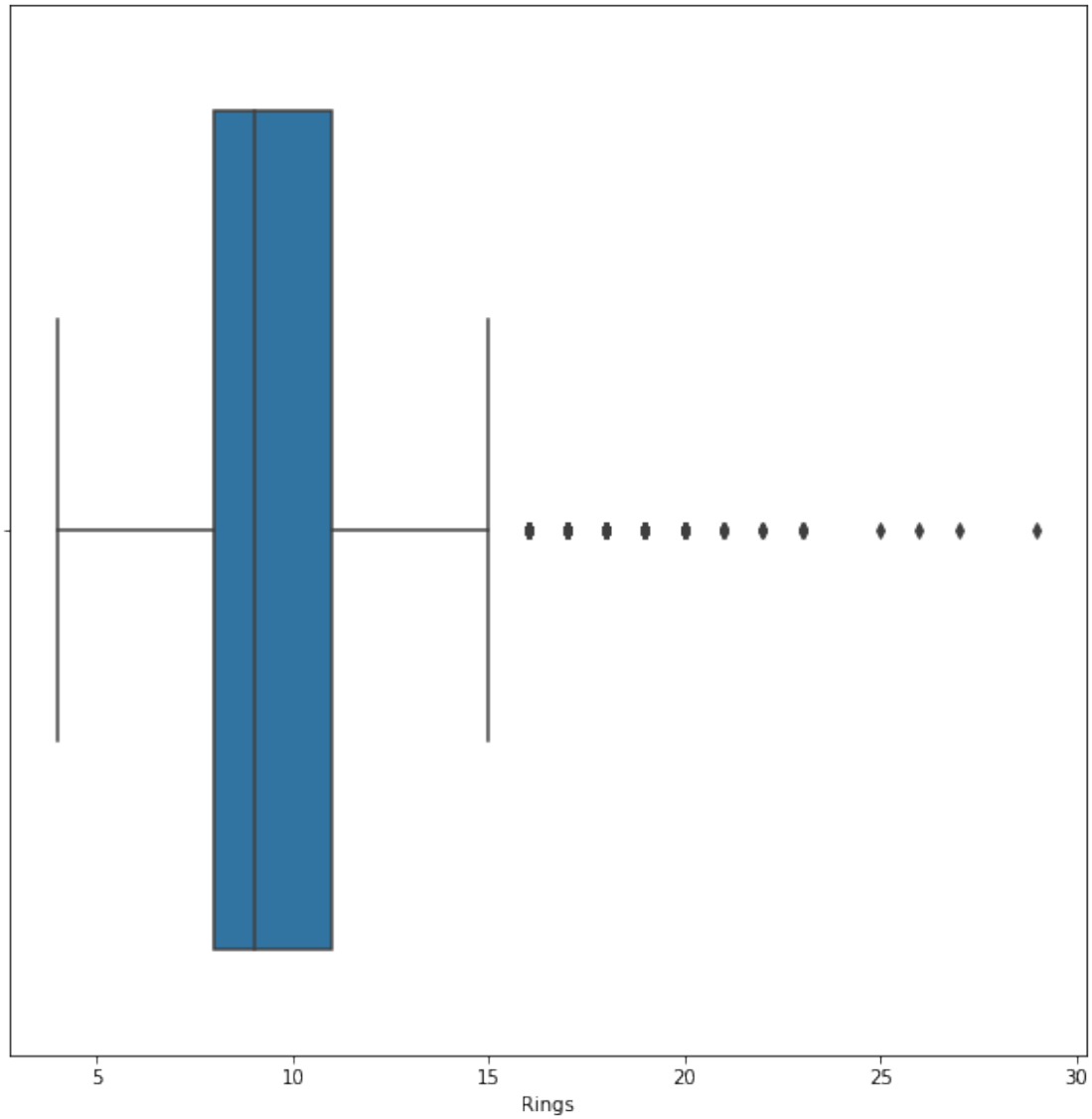
```
<matplotlib.axes._subplots.AxesSubplot at 0x7fd219530d10>
```




```
sns.boxplot(ab.Rings)
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43:
FutureWarning: Pass the following variable as a keyword arg: x. From
version 0.12, the only valid positional argument will be `data`, and
passing other arguments without an explicit keyword will result in an
error or misinterpretation.
  FutureWarning
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fd2194a5510>
```



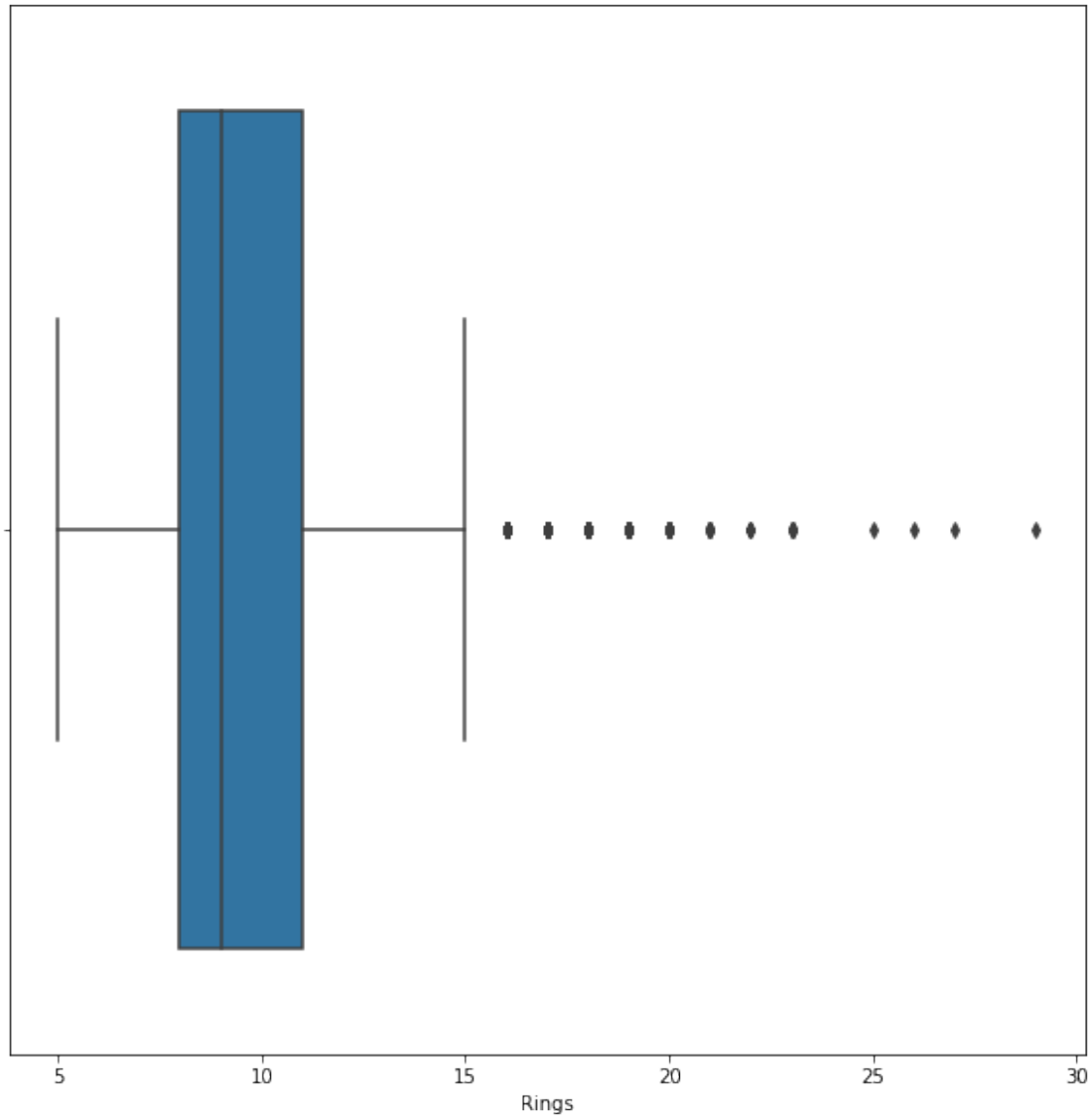

```
p01 = ab.Rings.quantile(0.01)
p01
```

```
5.0
```

```
ab = ab[ab.Rings >= p01]
sns.boxplot(ab.Rings)
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43:
FutureWarning: Pass the following variable as a keyword arg: x. From
version 0.12, the only valid positional argument will be `data`, and
passing other arguments without an explicit keyword will result in an
error or misinterpretation.
  FutureWarning
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fd219474810>
```



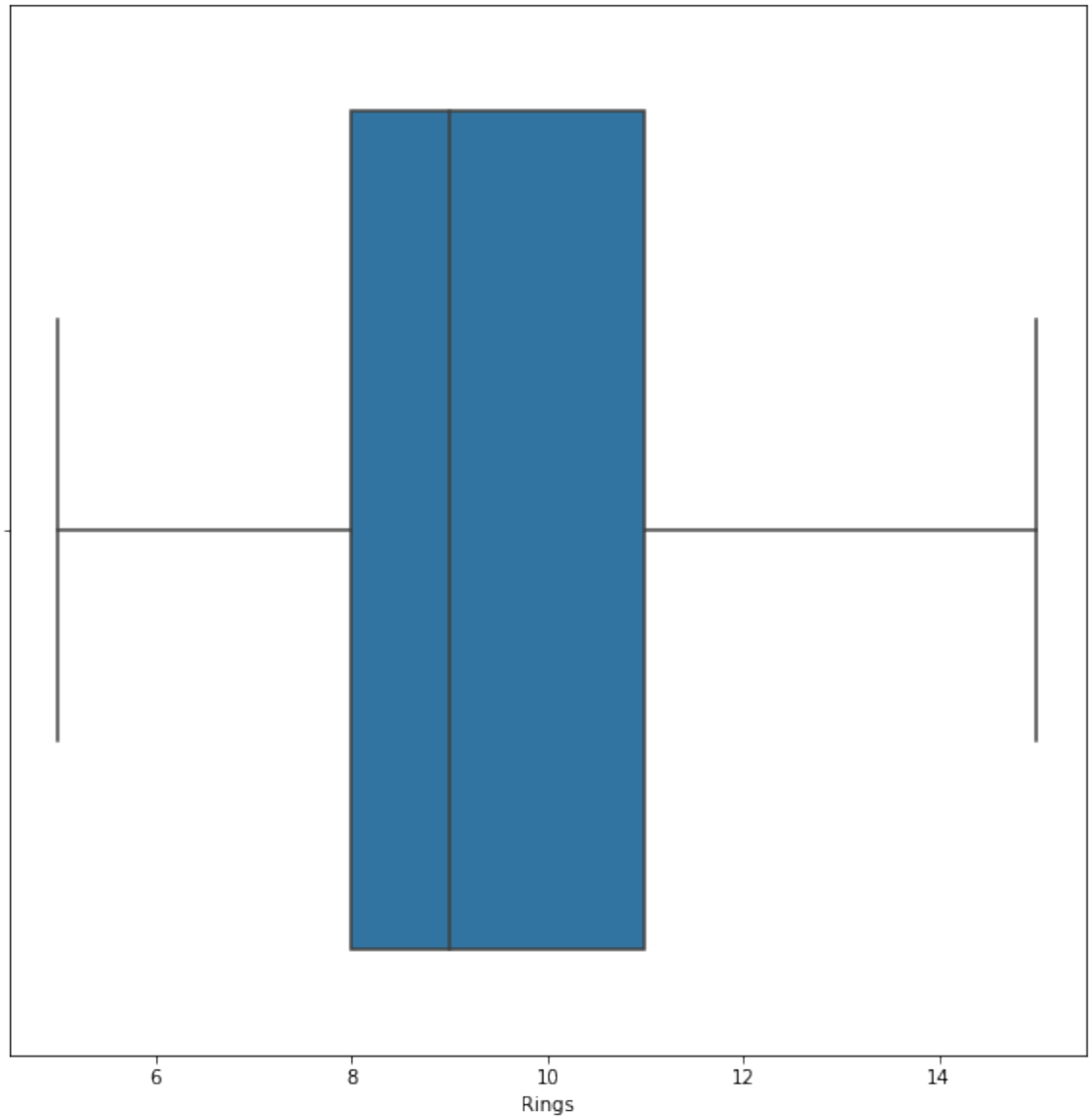

```
p99 = ab.Rings.quantile(0.92)
p99
```

```
15.0
```

```
ab = ab[ab.Rings <= p99]
sns.boxplot(ab.Rings)
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43:
FutureWarning: Pass the following variable as a keyword arg: x. From
version 0.12, the only valid positional argument will be `data`, and
passing other arguments without an explicit keyword will result in an
error or misinterpretation.
  FutureWarning
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fd21948ce50>
```




```
ab.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 3621 entries, 0 to 4175
Data columns (total 9 columns):
 #   Column          Non-Null Count  Dtype  
---  ------          --------------  -----  
 0   Sex             3621 non-null   int64  
 1   Length          3621 non-null   float64
 2   Diameter        3621 non-null   float64
 3   Height          3621 non-null   float64
 4   Whole weight    3621 non-null   float64
 5   Shucked weight  3621 non-null   float64
 6   Viscera weight  3621 non-null   float64
 7   Shell weight    3621 non-null   float64
 8   Rings           3621 non-null   int64  
```

```
dtypes: float64(7), int64(2)
memory usage: 282.9 KB
```

Feature Scaling

```
X.head()
```

```
   Length  Diameter  Height  Whole weight  Shucked weight  Viscera
weight  \
0   0.455     0.365   0.095        0.5140          0.2245
0.1010
1   0.350     0.265   0.090        0.2255          0.0995
0.0485
2   0.530     0.420   0.135        0.6770          0.2565
0.1415
3   0.440     0.365   0.125        0.5160          0.2155
0.1140
4   0.330     0.255   0.080        0.2050          0.0895
0.0395

   Shell weight
0         0.150
1         0.070
2         0.210
3         0.155
4         0.055
```

```
X.describe()
```

```
            Length     Diameter       Height  Whole weight  Shucked
weight  \
count  4177.000000  4177.000000  4177.000000   4177.000000
4177.000000
mean      0.523992     0.407881     0.139516      0.828742
0.359367
std       0.120093     0.099240     0.041827      0.490389
0.221963
min       0.075000     0.055000     0.000000      0.002000
0.001000
25%       0.450000     0.350000     0.115000      0.441500
0.186000
50%       0.545000     0.425000     0.140000      0.799500
0.336000
75%       0.615000     0.480000     0.165000      1.153000
0.502000
max       0.815000     0.650000     1.130000      2.825500
1.488000

       Viscera weight  Shell weight
count     4177.000000   4177.000000
mean         0.180594      0.238831
```

```
std          0.109614      0.139203
min          0.000500      0.001500
25%          0.093500      0.130000
50%          0.171000      0.234000
75%          0.253000      0.329000
max          0.760000      1.005000
```

```
from sklearn.preprocessing import MinMaxScaler
scale = MinMaxScaler()
```

```
X_scaled = pd.DataFrame(scale.fit_transform(X),columns= X.columns)
X_scaled.head()
```

```
     Length  Diameter    Height  Whole weight  Shucked weight  Viscera
weight  \
0  0.513514  0.521008  0.084071      0.181335        0.150303
0.132324
1  0.371622  0.352941  0.079646      0.079157        0.066241
0.063199
2  0.614865  0.613445  0.119469      0.239065        0.171822
0.185648
3  0.493243  0.521008  0.110619      0.182044        0.144250
0.149440
4  0.344595  0.336134  0.070796      0.071897        0.059516
0.051350

   Shell weight
0      0.147982
1      0.068261
2      0.207773
3      0.152965
4      0.053313
```

```
X.describe()
```

```
            Length     Diameter       Height  Whole weight  Shucked
weight  \
count  4177.000000  4177.000000  4177.000000   4177.000000
4177.000000
mean      0.523992     0.407881     0.139516      0.828742
0.359367
std       0.120093     0.099240     0.041827      0.490389
0.221963
min       0.075000     0.055000     0.000000      0.002000
0.001000
25%       0.450000     0.350000     0.115000      0.441500
0.186000
50%       0.545000     0.425000     0.140000      0.799500
0.336000
75%       0.615000     0.480000     0.165000      1.153000
0.502000
```

```
max       0.815000     0.650000     1.130000      2.825500
1.488000

       Viscera weight  Shell weight
count     4177.000000   4177.000000
mean         0.180594      0.238831
std          0.109614      0.139203
min          0.000500      0.001500
25%          0.093500      0.130000
50%          0.171000      0.234000
75%          0.253000      0.329000
max          0.760000      1.005000

X_scaled.describe()

            Length     Diameter       Height  Whole weight  Shucked
weight  \
count  4177.000000  4177.000000  4177.000000   4177.000000
4177.000000
mean      0.606746     0.593078     0.123466      0.292808
0.241000
std       0.162288     0.166790     0.037015      0.173681
0.149269
min       0.000000     0.000000     0.000000      0.000000
0.000000
25%       0.506757     0.495798     0.101770      0.155658
0.124412
50%       0.635135     0.621849     0.123894      0.282451
0.225286
75%       0.729730     0.714286     0.146018      0.407650
0.336920
max       1.000000     1.000000     1.000000      1.000000
1.000000

       Viscera weight  Shell weight
count     4177.000000   4177.000000
mean         0.237121      0.236503
std          0.144324      0.138717
min          0.000000      0.000000
25%          0.122449      0.128052
50%          0.224490      0.231689
75%          0.332456      0.326358
max          1.000000      1.000000

y.describe()

count    4177.000000
mean        9.933684
std         3.224169
min         1.000000
25%         8.000000
```

```
50%         9.000000
75%        11.000000
max        29.000000
Name: Rings, dtype: float64
```

Train Test Split

```
from sklearn.model_selection import train_test_split
X_train,X_test,y_train,y_test=train_test_split(X_scaled,y,test_size=0.
3,random_state=0)
```

```
X_train.shape
```

```
(2923, 7)
```

```
X_train.shape
```

```
(2923, 7)
```

```
y_train.shape
```

```
(2923,)
```

```
y_test.shape
```

```
(1254,)
```

Model Building

```
from sklearn.linear_model import Ridge
```

```
r=Ridge()
```

```
r.fit(X_train,y_train)
```

```
Ridge()
```

```
r_pred=r.predict(X_test)
r_pred
```

```
array([12.79666142,  9.73082729, 10.5759828 , ...,  9.67503523,
       17.62015894, 11.61193318])
```

```
r_pred_train = r.predict(X_train)
r_pred_train
```

```
array([12.10842881,  6.70483065,  7.40951508, ...,  9.67501971,
       11.49719609,  8.3484614 ])
```

```
from sklearn.linear_model import Lasso
```

```
l = Lasso()
```

```
l.fit(X_train,y_train)
```

```
Lasso()
```

```
l_pred = l.predict(X_test)
l_pred
```

```
array([9.94902497, 9.94902497, 9.94902497, ..., 9.94902497,
9.94902497,
       9.94902497])
```

```
l_pred_train = l.predict(X_train)
l_pred_train
```

```
array([9.94902497, 9.94902497, 9.94902497, ..., 9.94902497,
9.94902497,
       9.94902497])
```

```
Rings=pd.DataFrame({'Actual_y_value':y_test,'Ridge_pred':r_pred,'Lasso
_pred':l_pred})
Rings.head(10)
```

```
      Actual_y_value  Ridge_pred  Lasso_pred
668               13   12.796661    9.949025
1580               8    9.730827    9.949025
3784              11   10.575983    9.949025
463                5    5.689987    9.949025
2615              12   10.763115    9.949025
1399              11   12.091017    9.949025
2054               7    7.859295    9.949025
2058               8    9.422339    9.949025
217                7    8.353626    9.949025
1931               9   12.159195    9.949025
```

Model Evaluation Metrics

```
from sklearn import metrics
```

```
#Mean-Squared-Error
```

```
print(metrics.mean_squared_error(y_test,r_pred))
print(metrics.mean_squared_error(y_test,l_pred))
```

```
5.237501870845426
10.54562109553069
```

```
#Root Mean Squared Error
```

```
print(np.sqrt(metrics.mean_squared_error(y_test,r_pred)))
print(np.sqrt(metrics.mean_squared_error(y_test,l_pred)))
```

```
2.2885589070079506
3.2474022072312954
```

```
## R2-score

print(metrics.r2_score(y_test,r_pred))
print(metrics.r2_score(y_test,l_pred))

0.5032251848939031
-0.0002476560577491238
```