```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

Load the dataset

In [4]:

```
df=pd.read_csv("D:\\Users\ELCOT\Abalone_IBM.csv")
df
```

Out[4]:

	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 [5]:

```
df.head()
```

Out[5]:

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

	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	0.150	15
1	M	0.350	0.265	0.090	0.2255	0.0995	0.0485	0.070	7
2	F	0.530	0.420	0.135	0.6770	0.2565	0.1415	0.210	9
3	M	0.440	0.365	0.125	0.5160	0.2155	0.1140	0.155	10
4	I	0.330	0.255	0.080	0.2050	0.0895	0.0395	0.055	7

Univariate Analysis

In [6]:

```
sns.displot(df.Rings)
```

Out[6]:

In [7]:

```
df.hist('Rings')
```

Out[7]:

```
array([[]], dtype=object)
```

Bi-variate analysis

In [9]:

```
sns.scatterplot(x=df.Length,y=df.Height)
```

Out[9]:

Multivariate Analysis

In [10]:

```
sns.pairplot(df)
```

Out[10]:

Perform descriptive statistic on dataset

In [11]:

```
df.describe()
```

Out[11]:

	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

Check the missing values & deal with them

In [12]:

```
df.isna()
```

Out[12]:

	Sex	Length	Diameter	Height	Whole weight	Shucked weight	Viscera weight	Shell weight	Rings
0	False	False	False	False	False	False	False	False	False
1	False	False	False	False	False	False	False	False	False
2	False	False	False	False	False	False	False	False	False
3	False	False	False	False	False	False	False	False	False

	Sex	Length	Diameter	Height	Whole weight	Shucked weight	Viscera weight	Shell weight	Rings
4	False	False	False	False	False	False	False	False	False
...	...	...	...	...	...	...	...	...	...
4172	False	False	False	False	False	False	False	False	False
4173	False	False	False	False	False	False	False	False	False
4174	False	False	False	False	False	False	False	False	False
4175	False	False	False	False	False	False	False	False	False
4176	False	False	False	False	False	False	False	False	False

4177 rows × 9 columns

In [13]:

```
df.isnull().sum()
```

Out[13]:

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

Find & replace the outliers

In [14]:

```
sns.boxplot(x=df['Rings'])
```

Out[14]:

check for categorical columnsand perform encoding

In [15]:

```
df['Sex'].replace({'M':0,'F':1})
df.head()
```

Out[15]:

	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	0.150	15
1	M	0.350	0.265	0.090	0.2255	0.0995	0.0485	0.070	7
2	F	0.530	0.420	0.135	0.6770	0.2565	0.1415	0.210	9
3	M	0.440	0.365	0.125	0.5160	0.2155	0.1140	0.155	10
4	I	0.330	0.255	0.080	0.2050	0.0895	0.0395	0.055	7

Split the data into dependent & independent variables

In [16]:

```
y=df['Whole weight']
print(y)
```

```
0       0.5140
1       0.2255
2       0.6770
3       0.5160
4       0.2050
         ...  
4172    0.8870
4173    0.9660
4174    1.1760
4175    1.0945
4176    1.9485
Name: Whole weight, Length: 4177, dtype: float64
```

In [17]:

```
x=df.drop(columns=['Whole weight'])
print(x)
```

```
     Sex  Length  Diameter  Height  Shucked weight  Viscera weight  \
0      M   0.455     0.365   0.095          0.2245          0.1010   
1      M   0.350     0.265   0.090          0.0995          0.0485   
2      F   0.530     0.420   0.135          0.2565          0.1415   
3      M   0.440     0.365   0.125          0.2155          0.1140   
4      I   0.330     0.255   0.080          0.0895          0.0395   
...   ..     ...       ...     ...             ...             ...   
4172   F   0.565     0.450   0.165          0.3700          0.2390   
4173   M   0.590     0.440   0.135          0.4390          0.2145   
4174   M   0.600     0.475   0.205          0.5255          0.2875   
```

```
4175   F   0.625     0.485   0.150          0.5310          0.2610
4176   M   0.710     0.555   0.195          0.9455          0.3765

      Shell weight  Rings
0           0.1500     15
1           0.0700      7
2           0.2100      9
3           0.1550     10
4           0.0550      7
...            ...    ...
4172        0.2490     11
4173        0.2605     10
4174        0.3080      9
4175        0.2960     10
4176        0.4950     12

[4177 rows x 8 columns]
```

Scale the independent variables

In [18]:

```
x=df.drop(columns=['Viscera weight'])
print(x)
```

```
     Sex  Length  Diameter  Height  Whole weight  Shucked weight  \
0      M   0.455     0.365   0.095        0.5140          0.2245
1      M   0.350     0.265   0.090        0.2255          0.0995
2      F   0.530     0.420   0.135        0.6770          0.2565
3      M   0.440     0.365   0.125        0.5160          0.2155
4      I   0.330     0.255   0.080        0.2050          0.0895
...   ..     ...       ...     ...           ...             ...
4172   F   0.565     0.450   0.165        0.8870          0.3700
4173   M   0.590     0.440   0.135        0.9660          0.4390
4174   M   0.600     0.475   0.205        1.1760          0.5255
4175   F   0.625     0.485   0.150        1.0945          0.5310
4176   M   0.710     0.555   0.195        1.9485          0.9455

      Shell weight  Rings
0           0.1500     15
1           0.0700      7
2           0.2100      9
3           0.1550     10
4           0.0550      7
...            ...    ...
4172        0.2490     11
4173        0.2605     10
4174        0.3080      9
4175        0.2960     10
4176        0.4950     12

[4177 rows x 8 columns]
```

In [21]:

```
from sklearn import preprocessing
```

In [24]:

```
x=df.iloc[:,1:6].Height
```

```
print(x)
```

```
0       0.095
1       0.090
2       0.135
3       0.125
4       0.080
        ...
4172    0.165
4173    0.135
4174    0.205
4175    0.150
4176    0.195
Name: Height, Length: 4177, dtype: float64
```

split the data into training and testing

In [28]:

```
from sklearn import tree
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeRegressor
```

In [29]:

```
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2,random_sta
te=0)
x_train.shape
```

Out[29]:

```
(3341,)
```

In [30]:

```
x_test.shape
```

Out[30]:

```
(836,)
```

In [31]:

```
y_test.shape
```

Out[31]:

```
(836,)
```

In [32]:

```
x=df.drop('Sex',axis=1)
print(x)
```

```
      Length  Diameter  Height  Whole weight  Shucked weight  Viscera weigh
t  \
0      0.455     0.365   0.095        0.5140          0.2245          0.101
0
1      0.350     0.265   0.090        0.2255          0.0995          0.048
5
2      0.530     0.420   0.135        0.6770          0.2565          0.141
5
3      0.440     0.365   0.125        0.5160          0.2155          0.114
0
4      0.330     0.255   0.080        0.2050          0.0895          0.039
5
...      ...       ...     ...           ...             ...            ..
.
```

```
4172   0.565     0.450   0.165        0.8870          0.3700          0.239
0
4173   0.590     0.440   0.135        0.9660          0.4390          0.214
5
4174   0.600     0.475   0.205        1.1760          0.5255          0.287
5
4175   0.625     0.485   0.150        1.0945          0.5310          0.261
0
4176   0.710     0.555   0.195        1.9485          0.9455          0.376
5

      Shell weight  Rings
0           0.1500     15
1           0.0700      7
2           0.2100      9
3           0.1550     10
4           0.0550      7
...            ...    ...
4172        0.2490     11
4173        0.2605     10
4174        0.3080      9
4175        0.2960     10
4176        0.4950     12

[4177 rows x 8 columns]
```

In [37]:

```
x=pd.get_dummies(df,columns=["Length"])
print(x)
```

```
     Sex  Diameter  Height  Whole weight  Shucked weight  Viscera weight  \
0       M     0.365   0.095        0.5140          0.2245          0.1010
1       M     0.265   0.090        0.2255          0.0995          0.0485
2       F     0.420   0.135        0.6770          0.2565          0.1415
3       M     0.365   0.125        0.5160          0.2155          0.1140
4       I     0.255   0.080        0.2050          0.0895          0.0395
...   ..       ...     ...           ...             ...             ...
4172    F     0.450   0.165        0.8870          0.3700          0.2390
4173    M     0.440   0.135        0.9660          0.4390          0.2145
4174    M     0.475   0.205        1.1760          0.5255          0.2875
4175    F     0.485   0.150        1.0945          0.5310          0.2610
4176    M     0.555   0.195        1.9485          0.9455          0.3765

      Shell weight  Rings  Length_0.075  Length_0.11  ...  Length_0.745  \
0           0.1500     15             0            0  ...             0
1           0.0700      7             0            0  ...             0
2           0.2100      9             0            0  ...             0
3           0.1550     10             0            0  ...             0
4           0.0550      7             0            0  ...             0
...            ...    ...           ...          ...  ...           ...
4172        0.2490     11             0            0  ...             0
4173        0.2605     10             0            0  ...             0
4174        0.3080      9             0            0  ...             0
4175        0.2960     10             0            0  ...             0
4176        0.4950     12             0            0  ...             0

      Length_0.75  Length_0.755  Length_0.76  Length_0.765  Length_0.77  \
0               0             0            0             0            0
```

```
1              0             0            0             0            0
2              0             0            0             0            0
3              0             0            0             0            0
4              0             0            0             0            0
...          ...           ...          ...           ...          ...
4172           0             0            0             0            0
4173           0             0            0             0            0
4174           0             0            0             0            0
4175           0             0            0             0            0
4176           0             0            0             0            0

      Length_0.775  Length_0.78  Length_0.8  Length_0.815
0                0            0           0             0
1                0            0           0             0
2                0            0           0             0
3                0            0           0             0
4                0            0           0             0
...            ...          ...         ...           ...
4172             0            0           0             0
4173             0            0           0             0
4174             0            0           0             0
4175             0            0           0             0
4176             0            0           0             0

[4177 rows x 142 columns]
```

Bulid ,Test & Train the model

In [38]:

```
df.info()
```

```
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
```

In [39]:

```
df.shape
```

Out[39]:

```
(4177, 9)
```

In [41]:

```
df.corr()
```

Out[41]:

	Length	Diameter	Height	Whole weight	Shucked weight	Viscera weight	Shell weight	Rings
Length	1.000000	0.986812	0.827554	0.925261	0.897914	0.903018	0.897706	0.556720
Diameter	0.986812	1.000000	0.833684	0.925452	0.893162	0.899724	0.905330	0.574660
Height	0.827554	0.833684	1.000000	0.819221	0.774972	0.798319	0.817338	0.557467
Whole weight	0.925261	0.925452	0.819221	1.000000	0.969405	0.966375	0.955355	0.540390
Shucked weight	0.897914	0.893162	0.774972	0.969405	1.000000	0.931961	0.882617	0.420884
Viscera weight	0.903018	0.899724	0.798319	0.966375	0.931961	1.000000	0.907656	0.503819
Shell weight	0.897706	0.905330	0.817338	0.955355	0.882617	0.907656	1.000000	0.627574
Rings	0.556720	0.574660	0.557467	0.540390	0.420884	0.503819	0.627574	1.000000

In [47]:

```
x=df[['Shell weight','Diameter','Height']]
y=df[['Rings']]
```

In [48]:

```
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.4,random_sta
te=100)
```

In [52]:

```
from sklearn.linear_model import LogisticRegression
```

In [53]:

```
lr=LogisticRegression()
lr.fit(x_train,y_train)
```

```
D:\Education Content\anaconda3\lib\site-packages\sklearn\utils\validation.p
y:63: DataConversionWarning: A column-vector y was passed when a 1d array w
as expected. Please change the shape of y to (n_samples, ), for example usi
ng ravel().
  return f(*args, **kwargs)
```

Out[53]:

```
LogisticRegression()
```

In [54]:

```
y_pred=lr.predict(x_test)
print(x_test)
print(y_pred)
```

```
      Shell weight  Diameter  Height
551         0.3450     0.490   0.155
3245        0.4400     0.550   0.160
```

```
1418        0.5280     0.555   0.215
416         0.4000     0.500   0.170
1553        0.0730     0.290   0.100
...            ...       ...     ...
3056        0.2900     0.485   0.215
279         0.2500     0.425   0.135
3770        0.2100     0.430   0.125
106         0.2800     0.430   0.165
3709        0.4895     0.550   0.190

[1671 rows x 3 columns]
[10 11 11 ...  9  9 11]
```

Measure the performance metrics

In [55]:

```
from sklearn.metrics import r2_score
from sklearn.metrics import mean_absolute_error
from sklearn.metrics import mean_squared_error
```

In [56]:

```
x_actual=[5,1,2,9]
y_pred=[3.5,0.9,2,9.9]
```

In [57]:

```
print(r2_score(x_actual,y_pred))
```

```
0.9207741935483871
```

In [58]:

```
print(mean_absolute_error(x_actual,y_pred))
```

```
0.6250000000000001
```

In [59]:

```
print( mean_squared_error(x_actual,y_pred))
```