

### Assignment -3

#### Problem Statement: Abalone Age Prediction

ASSIGNMENT DATE	19 OCTOBER 2022
STUDENT NAME	S.DHINESHKUMAR
STUDENT ROLL NUMBER	CS19008
MAXIMUM MARKS	2 mark

*#1.Download the dataset*

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

*#2. Load the dataset into the tool*

```
df=pd.read_csv("abalone.csv")
df.head()
```

	Sex	Length	Diameter	Height	Whole weight	Shucked weight	Viscera weight	Shell 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 df.tail()

Whol

Shucke

Viscer Shell

	Sex	Length	Diameter	Height	Whole weight	Shucked weight	Vent weight	Total weight
4172	F	0.565	0.450	0.165	0.8870	0.3700	0.2390	0.2490
4173	M	0.590	0.440	0.135	0.9660	0.4390	0.2145	0.2605
4174	M	0.600	0.475	0.205	1.1760	0.5255	0.2875	0.3080
4175	F	0.625	0.485	0.150	1.0945	0.5310	0.2610	0.2960
4176	M	0.710	0.555	0.195	1.9485	0.9455	0.3765	0.4950

```
df.shape
(4177, 9) 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 #3. Perform Below Visualizations
#Univariate Analysis sns.boxplot(x=df['Height'])

```

```

#Bi-Variate Analysis sns.lineplot(df['Sex'],df['Length'])

```

C:\Users\shire\anaconda3\lib\site-packages\seaborn\\_decorators.py:36: 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.

```
warnings.warn(
```

```

#Multi-Variate Analysis sns.heatmap(df.corr(),annot=True)

```

```

#4. Perform descriptive statistics on the dataset df.describe()

```

	Shucke					
	Diameter	Whole Height	Viscera weight	Shell weight	weight	weight
cou	4177.00	4177.00	4177.00	4177.00	4177.00	4177.00
nt	0000	0000	0000	0000	0000	0000
Shucke	0000	0000	0000	0000	0000	0000
me	0.52399	0.40788	0.13951	0.82874	0.35936	0.18059
an	0.23883	9.93368	2	1	6	2
7	4	1	4			
std	0.12009	0.09924	0.04182	0.49038	0.22196	0.10961
	0.13920	3.22416				



4177.0 0.359367 0.221963 0.0010 0.1860 0.3360 0.502 1.4880

**weight**

**Viscera**

4177.0 0.180594 0.109614 0.0005 0.0935 0.1710 0.253 0.7600

**weight**

**Shell**

**weight** 4177.0 0.238831 0.139203 0.0015 0.1300 0.2340 0.329 1.0050

**Rings** 4177.0 9.933684 3.224169 1.0000 8.0000 9.0000 11.000 29.0000 #5. Check for Missing values and deal with them df.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

#6. Find the outliers and replace them outliers

df['Sex'].replace({'M':1, 'F':0, 'T':-1},inplace=True)

df.head()

		Sex	Diameter	Height	Whole weight	Shucked weight	Viscera weight	Shell weight	Rings
		Length							
0	1	0.455	0.365	0.095	0.5140	0.2245	0.1010	0.150	15
1	1	0.350	0.265	0.090	0.2255	0.0995	0.0485	0.070	7
2	0	0.530	0.420	0.135	0.6770	0.2565	0.1415	0.210	9

<b>3</b>	1	0.440	0.365	0.125	0.5160	0.2155	0.1140	0.155	10
			0.255	0.080	0.2050	0.0895	0.0395	0.055	7
<b>4</b>	-1	0.330							

```
df.Sex.unique()
```

```
array([ 1,  0, -1], dtype=int64)
```

```
sns.boxplot(x=df["Sex"])
```

```
sns.boxplot(x=df["Length"])
```

```
sns.boxplot(x=df["Diameter"])
```

```
sns.boxplot(x=df["Height"])
```

```
sns.boxplot(x=df["Whole weight"])
```

```
sns.boxplot(x=df["Shucked weight"])
```

```
sns.boxplot(x=df["Viscera weight"])
```

```
sns.boxplot(x=df["Shell weight"])
```

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

```
#handle outlier qnt=df.quantile(q=[0.25,0.75])
```

```
qnt
```

	Sex	Length	Diameter	Height	Whole weight	Shucked weight	Viscera weight	Shell weight	Rings
0.25	-	0.450	0.35	0.115	0.4415	0.186	0.0935	0.130	8.0

```
0.75
1.0
5 iqr=qnt.loc[0.75]-qnt.loc[0.25]
```

iqr

```
Sex      2.0000
Length   0.1650
Diameter  0.1300
Height   0.0500
Whole weight  0.7115
Shucked weight  0.3160
Viscera weight  0.1595
Shell weight  0.1990 Rings
3.0000 dtype: float64 #lower limit
lower=qnt.loc[0.25]-
(1.5*iqr) lower
```

```
Sex      -4.00000
Length   0.20250
Diameter  0.15500
Height   0.04000
Whole weight  -0.62575
Shucked weight  -0.28800
Viscera weight  -0.14575
```

```
Shell weight    -0.16850
Rings           3.50000 dtype:
float64
upper=qnt.loc[0.75]+(1.5*iqr)
upper
```

```
Sex             4.00000
Length          0.86250
Diameter        0.67500
Height          0.24000
Whole weight    2.22025
Shucked weight  0.97600
Viscera weight  0.49225
Shell weight    0.62750
Rings           15.50000 dtype:
float64 df.mean()
```

```
Sex             0.044530
Length          0.523992
Diameter        0.407881
Height          0.139516 Whole weight    0.828742
Shucked weight  0.359367
Viscera weight  0.180594
```

```
Shell weight    0.238831 Rings           9.933684 dtype: float64
```

```
#replace outlier df['Length']=np.where(df['Length']<0.22,0.52,df['Length'])
```

```
df['Diameter']=np.where(df['Diameter']<0.155,0.407,df['Diameter'])
```

```
df['Height']=np.where(df['Height']<0.04,0.13,df['Height'])
```

```
df['Height']=np.where(df['Height']>0.24,0.13,df['Height']) df['Whole
```

```
weight']=np.where(df['Whole weight']>2.18,0.83,df['Whole weight']) df['Shucked
```

```
weight']=np.where(df['Shucked weight']>0.958,0.359367,df['Shucked weight']) df['Viscera
```

```
weight']=np.where(df['Viscera weight']>0.478,0.18,df['Viscera weight']) df['Shell
```

```
weight']=np.where(df['Shell weight']>0.61,0.238831,df['Shell weight'])
```

```
df['Rings']=np.where(df['Rings']<3.5,9.93,df['Rings'])
```



```
df['Rings']=np.where(df['Rings']>15.5,9.93,df['Rings']) sns.boxplot(df['Length'])
```

C:\Users\shire\anaconda3\lib\site-packages\seaborn\\_decorators.py:36: 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.

```
warnings.warn(
```

```
sns.boxplot(df['Diameter'])
```

C:\Users\shire\anaconda3\lib\site-packages\seaborn\\_decorators.py:36: 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.

```
warnings.warn(
```

```
sns.boxplot(df['Height'])
```

C:\Users\shire\anaconda3\lib\site-packages\seaborn\\_decorators.py:36: 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.

```
warnings.warn(
```

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

C:\Users\shire\anaconda3\lib\site-packages\seaborn\\_decorators.py:36: 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.

```
warnings.warn(
```

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

C:\Users\shire\anaconda3\lib\site-packages\seaborn\\_decorators.py:36: 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.

ent will be `data`, and passing other arguments without an explicit keyword will result in an error or misinterpretation.

```
warnings.warn(
```

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

C:\Users\shire\anaconda3\lib\site-packages\seaborn\\_decorators.py:36: 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.

```
warnings.warn(
```

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

C:\Users\shire\anaconda3\lib\site-packages\seaborn\\_decorators.py:36: 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.

```
warnings.warn(
```

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

C:\Users\shire\anaconda3\lib\site-packages\seaborn\\_decorators.py:36: 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.

```
warnings.warn(
```

*#7. Check for Categorical columns and perform encoding df.head()*

*#sex is categorical and encoding is performed*

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

<b>0</b>	1	0.455	0.365	0.095	0.5140	0.2245	0.1010	0.150	15.0
<b>1</b>	1	0.350	0.265	0.090	0.2255	0.0995	0.0485	0.070	7.0
<b>2</b>	0	0.530	0.420	0.135	0.6770	0.2565	0.1415	0.210	9.0
<b>3</b>	1	0.440	0.365	0.125	0.5160	0.2155	0.1140	0.155	10.0
<b>4</b>	-1	0.330	0.255	0.080	0.2050	0.0895	0.0395	0.055	7.0

*#8. Split the data into dependent and independent variables*

*#independent variable* x=df.drop(columns=['Rings'],axis=1).values

x array([[1. , 0.455, 0.365 , ..., 0.2245, 0.101 , 0.15 ], [1. , 0.35 , 0.265 , ..., 0.0995, 0.0485, 0.07 ],

[0. , 0.53 , 0.42 , ..., 0.2565, 0.1415, 0.21 ],

...,

[1. , 0.6 , 0.475 , ..., 0.5255, 0.2875, 0.308 ],

[0. , 0.625 , 0.485 , ..., 0.531 , 0.261 , 0.296 ],

[1. , 0.71 , 0.555 , ..., 0.9455, 0.3765, 0.495 ]])

*#dependent variable*

y=df['Rings'].values y

array([15., 7., 9., ..., 9., 10., 12.]) *#9. Scale*

*the independent variables from*

sklearn.preprocessing **import** scale

x=scale(x) x

array([[ 1.15434629, -0.66347373, -0.50167301, ..., -0.61177023,  
-0.73234257, -0.64358992],

[ 1.15434629, -1.60127264, -1.57291477, ..., -1.21969385,  
-1.23612645, -1.25742425],

[-0.05379815, 0.00638264, 0.08750996, ..., -0.45614178,  
-0.34370929, -0.18321418],

...,

```
[ 1.15434629, 0.63158191, 0.67669293, ..., 0.85210986,
 1.05728969, 0.56873287],
[-0.05379815, 0.85486737, 0.78381711, ..., 0.8788585 ,
 0.80299878, 0.47665772],
[ 1.15434629, 1.61403792, 1.53368634, ..., 2.89473324,
1.91132331, 2.0035706 ]] #10. Split the data into training
and testing from sklearn.model_selection import
train_test_split
```

```
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2)
```

*#11. Build the Model*

```
from sklearn.linear_model import LinearRegression linreg=LinearRegression() #12. Train
the Model linreg.fit(x_train,y_train)
```

LinearRegression() *#13. Test the Model*

```
test_pred=linreg.predict(x_test)
test_pred
```

```
array([10.49406044, 14.63071197, 9.5052661 , 7.12175027, 9.59508595,
 9.4343576 , 8.77992046, 10.17130406, 7.45502774, 9.87604313,
 10.98645479, 7.5538718 , 8.87927518, 9.76382226 , 8.54256728,
 10.42387201, 9.10033819, 9.87903278, 11.42897247, 7.06763663,
 10.57223182, 9.76975285, 12.30738965, 8.55382376, 9.52374863,
 8.21301289, 6.25183627, 7.12742482, 9.74741373, 10.3017582 ,
 9.82726168, 9.84749184, 10.4108395 , 10.3081998 , 10.08572396,
 8.30245647, 7.235845 , 6.74452118, 10.42584137, 7.64274971,
 7.14405667, 9.16150599, 8.70935569, 10.74880185, 9.86452375,
 12.88609365, 6.57858505, 9.5398517 , 6.81250209, 10.60088961,
 10.58682023, 10.59758934, 10.63987208, 10.60373354, 9.03578911,
 8.62103663, 9.90652623, 7.02963956, 9.84641914, 8.62932278,
 7.71223792, 11.69923451, 11.10448696, 8.06123754, 8.55513658,
 13.39968976, 8.26727764, 9.52753025, 9.09315656, 12.58339768,
```

9.99703469, 10.24999324, 9.29384572, 10.84986883, 9.23432613,  
7.71248702, 10.82510489, 9.74110842, 10.18617001, 11.15757814,  
8.15589364, 7.74042932, 6.90572983, 10.00216891, 12.35623317,  
9.2594473 , 9.83903046, 8.79445305, 9.98771476, 10.72074918,  
5.76586411, 8.83952495, 7.82141633, 9.27397291, 10.08449131,  
7.97368561, 8.13133341, 10.58531402, 8.54116758, 8.87592087,  
10.27752815, 9.91826533, 7.35190815, 10.30758392, 7.30769068,  
10.3549833 , 10.69101603, 10.1181462 , 10.39559027, 11.9945787 ,  
10.05265786, 12.85497306, 11.33865314, 10.6160416 , 10.86643523,  
9.98776731, 10.07059534, 7.51087688, 8.81450733, 10.76394848,  
9.23449231, 8.9394567 , 11.35528501, 7.02952734, 8.22981655,  
7.39038626, 7.16648403, 7.72492669, 6.96924802, 7.78201642,  
7.17710403, 9.82222011, 9.56803182, 8.40217156, 8.3040808 ,  
9.19097285, 7.27282145, 8.7291546 , 8.02818234, 9.6287928 ,  
9.17367559, 10.67429449, 10.83594529, 10.03487667, 7.01082421,  
8.22106326, 9.52078398, 12.01200605, 7.0664238 , 7.02545033,  
6.38664272, 9.03716991, 9.89980919, 9.54143876, 10.48601031,  
7.89737086, 10.57993475, 12.60549688, 8.9722634 , 8.86375281,  
10.58737471, 8.23508559, 9.16831774, 11.32643922, 11.72162036,  
7.35637849, 7.57148604, 7.1648948 , 10.85620295, 9.55486626,  
10.68453461, 10.42003548, 9.94733416, 11.13891581, 9.01364719,  
7.82060141, 10.78208786, 7.46904197, 9.32761963, 7.78647994,  
10.75827275, 8.09475084, 9.26765508, 9.58812949, 7.26964315,  
8.97532078, 8.90396235, 6.62637508, 7.78750708, 8.243058 ,  
9.46740388, 8.01654749, 8.84610761, 12.06376478, 11.18458934,  
7.95791777, 8.73139889, 7.63438426, 10.19784773, 10.19657975,  
9.88547762, 8.18847269, 7.75134569, 7.93222173, 8.53043085,  
11.47767482, 11.63701859, 9.67054006, 7.15334679, 11.58254568,  
10.91672544, 10.65123953, 11.30462744, 8.01570854, 8.691925 ,  
6.99630889, 10.45505798, 11.08400844, 7.84853522, 7.89503444,  
10.36775292, 9.29193168, 8.45869519, 9.40891292, 8.71995183,  
10.41488943, 9.80584287, 9.40871844, 10.47585472, 6.77413109,  
10.07855451, 9.36989613, 12.40825012, 8.71057984, 9.97974427,  
9.26533226, 10.63083868, 9.49615866, 10.23657265, 11.25380255,

10.65503119, 7.22469252, 10.23933921, 11.66614343, 7.52501383,  
9.78137819, 11.74179743, 10.06569605, 7.59341194, 9.32548854,  
9.09407202, 10.37992831, 10.4198217 , 9.20540036, 13.37322348,  
7.04827246, 7.30060552, 7.76040817, 8.26405016, 8.37641501,  
7.98024139, 8.66106856, 10.29294231, 8.4533951 , 9.1029908 ,  
7.6728443 , 9.17493898, 11.3350483 , 8.14113401, 9.57990685,  
8.99792287, 7.81308267, 7.88056289, 9.71714644, 8.78928014,  
7.48733805, 9.29344547, 8.25005563, 6.32596886, 10.67952799,  
10.34988789, 10.61398995, 9.73029599, 10.61124145, 8.10672637,  
10.83303256, 10.58983644, 11.56224758, 11.51701776, 7.26264654,  
9.17142228, 5.94220242, 8.79721855, 10.20287693, 10.40251293,  
7.26467813, 11.44855319, 10.18314512, 11.56865106, 10.08095547,  
11.04935475, 8.88901813, 10.06455925, 8.2275154 , 11.38494403,  
10.46370124, 8.81517211, 8.07626049, 10.29997579, 10.70159463,  
9.52425275, 8.55212551, 11.63567264, 7.01687668, 10.64424025,  
11.65796361, 8.03040793, 8.99581481, 5.87918977, 7.22561493,  
8.64902765, 8.46282178, 10.26638935, 7.77541642, 10.48666402,  
10.97160807, 7.77090259, 6.95097016, 10.66867657, 9.81598811,  
8.86175523, 10.14390988, 10.13604128, 7.67979877, 8.32951005,  
10.52746288, 11.03253764, 9.72136409, 9.96003508, 10.72896737,  
9.69336726, 9.0723992 , 9.28253035, 7.15534276, 10.02260695,  
8.39025513, 9.17409245, 8.79400875, 8.03635255, 13.46848816,  
11.25697851, 7.00933557, 8.2469982 , 8.44066123, 12.07134675,  
7.86611644, 6.91634306, 10.45047036, 9.05831727, 7.61872774,  
12.12276476, 12.15336763, 10.21088672, 7.30640948, 11.7712247 ,  
8.22309031, 9.00229321, 12.56925984, 9.89227365, 9.12720821,  
9.92856998, 6.15308924, 9.65988046, 7.26498527, 8.69157712,  
10.66712505, 12.0903993 , 10.00895812, 8.32592796, 10.09475343,  
9.9180563 , 7.79788418, 8.227395 , 9.67655999, 8.49861084,  
10.68758867, 10.96226694, 9.31583533, 9.88280193, 12.49353697,  
9.68406614, 8.39535884, 9.21841136, 8.05640704, 7.73070397,  
10.69944854, 6.88662863, 10.44933744, 8.63070059, 9.5300456 ,  
7.17288396, 7.23961838, 8.2406796 , 12.33534676, 9.31607714,

8.5199514 , 7.7413676 , 9.75259252, 9.03154513, 8.99602774,  
10.86102952, 9.72912372, 11.45369535, 7.73315024, 8.72272879,  
12.62516987, 8.07189869, 8.610371 , 7.96393172, 10.52464571,  
10.2889487 , 9.77064556, 7.2484293 , 6.75663943, 7.31758214,  
6.86054595, 8.3547735 , 8.28097679, 9.20481118, 6.17213238,  
7.7808438 , 10.18114547, 8.28582292, 9.34694407, 9.87520219,  
7.50494736, 8.88762551, 9.97581207, 11.05621772, 8.42628309,  
10.45364784, 13.37825413, 6.81598561, 9.27660382, 9.93844627,  
9.48483861, 8.2229326 , 10.77401117, 8.49913631, 9.69420869,  
10.1276895 , 9.32434402, 9.03969924, 9.2090436 , 11.16679102,  
11.52625517, 9.11519373, 10.26990747, 10.25863036, 9.68179585,  
10.50254453, 8.4147771 , 9.44129266, 12.51844495, 11.28219296,  
9.28797035, 7.81873126, 11.23464985, 6.11774657, 7.48506274,  
10.47162748, 6.18121011, 10.64721055, 6.36753714, 7.43156506,  
10.74864602, 8.27254424, 10.53053898, 8.98245363, 7.14884702,  
7.9185863 , 11.07177623, 9.84080718, 9.7463575 , 12.36483198,  
9.59473052, 11.53900136, 12.11790837, 10.456185 , 9.38664402,  
8.72639235, 10.05931532, 10.69421662, 9.40048527, 7.89705404,  
7.528022 , 9.54735021, 9.53043136, 9.60930106, 10.966415 ,  
8.41651724, 10.80665528, 10.82306864, 9.64976908, 10.84804833,  
7.25708744, 10.75477728, 10.31460886, 8.66916119, 9.59165337,  
10.71167513, 7.44318757, 7.33630981, 8.26722774, 10.24637661,  
7.26052307, 8.64349261, 8.00969525, 9.44531738, 9.25336289,  
8.8173519 , 11.10766426, 11.42811883, 11.63225499, 12.2226451 ,  
9.55772247, 10.01762113, 8.49878439, 13.07325304, 10.22581239,  
9.68317467, 9.36644105, 13.91225633, 8.11861748, 11.22528029,  
6.41498717, 8.33450985, 9.29388748, 9.5787418 , 9.62165703,  
9.34579274, 9.37312624, 7.57787897, 8.25299105, 7.75499926,  
10.20611942, 8.96911496, 10.13240095, 9.40401846, 10.01532991,  
6.42889635, 11.0992981 , 9.81411854, 10.66690492, 9.77542891,  
10.7579419 , 11.4218958 , 9.17569936, 11.15119791, 8.00751085,  
11.0322202 , 9.70187637, 11.70832554, 6.53861204, 8.01923515,  
6.92318834, 8.59732263, 8.40056135, 7.30281719, 9.14429838,  
10.45215168, 10.2717039 , 10.67123959, 7.59738449, 10.61468447,

9.42817769, 7.65101852, 10.71028888, 8.43697555, 6.37462576,  
10.74709502, 9.62271633, 9.58770478, 9.24606927, 10.7670444 ,  
11.54380344, 8.06392442, 7.21039732, 6.1079307 , 6.93022124,  
11.10032424, 10.95983728, 9.75697819, 11.52451347, 10.70619701,  
8.81853322, 10.6684791 , 8.01063822, 8.90079346, 10.47543873,  
8.91318792, 9.95998451, 10.44139164, 10.51897794, 11.57882658,  
11.84456732, 10.35382107, 10.7726265 , 9.30094023, 9.25140448,  
10.4064332 , 10.66595425, 10.0278771 , 11.38272576, 10.0385854 ,  
10.42082743, 11.13369816, 7.28230658, 11.78485276, 12.16118565,  
9.89429226, 8.37477714, 8.11694818, 9.03563761, 10.98296898,  
9.93984071, 10.56108814, 9.64267504, 11.89934257, 8.11015003,  
10.80031907, 8.74094699, 10.97226077, 9.77774357, 10.43921997,  
6.49816132, 9.67821635, 10.28257624, 8.7629948 , 9.63511441,  
10.895716 , 7.45692129, 10.19374451, 9.22221708, 9.78041215,  
11.44760477, 10.41100083, 8.30416136, 11.68611051, 7.68861374,  
11.71009186, 8.69958294, 9.71949872, 10.71546544, 8.45414734,  
10.70782326, 9.69817259, 8.49010884, 11.20048579, 7.10417804,  
8.60959772, 10.33897341, 7.17493522, 9.45440014, 11.11859072,  
7.47758363, 9.19445458, 6.8574115 , 10.35808533, 9.28828818,  
10.45985457, 11.31466369, 11.4669934 , 11.68665773, 8.99993578,  
8.97952297, 7.78900018, 8.18450868, 7.95794984, 9.00396033,  
10.17991165, 11.09356031, 9.46220989, 10.7942435 , 11.84593758,  
11.40717735, 7.86855466, 9.27888622, 9.84437026, 7.75099815,  
10.19060927, 10.91126132, 7.79415953, 10.83764466, 11.13188532,  
9.26878746, 8.73638969, 11.64932488, 10.11452862, 9.03523089,  
12.91263157, 10.25364549, 9.4425625 , 11.00469399, 6.22828241,  
10.16979898, 10.60212058, 9.6577842 , 10.23672345, 10.83518908,  
9.19495763, 10.08590973, 6.08966019, 10.98866463, 9.45082712,  
8.46473954, 8.37293439, 8.07427585, 7.88270037, 10.85787601,  
9.6116211 , 9.75365118, 8.42323857, 8.42694641, 10.07460764,  
10.94404488, 10.76805849, 8.29820248, 8.70900854, 10.84338969,  
9.02533208, 7.51567397, 10.11480584, 11.19191832, 10.22483298,  
9.75858954, 11.01863784, 11.04401786, 9.82237998, 11.11036566,  
9.12657119, 9.9134041 , 9.96637254, 7.98851208, 8.76813529,



8.44222221, 9.8299799 , 11.43538793, 11.46727411, 9.58198696,  
 9.41516146, 6.04997802, 8.7382142 , 8.39811798, 8.89002643,  
 9.15796197, 9.72926536, 9.62033314, 9.80335615, 10.09973494,  
 10.45277764, 9.26207715, 6.99253028, 11.70117286, 9.10432505,  
 11.81609264, 11.15118373, 10.07627417, 7.47581497, 10.97754338,  
 11.23837153, 8.82525246, 11.42534581, 9.66056195, 10.662489 ,  
 10.08497982, 10.34406113, 8.7286661 , 11.0201679 , 10.54348108,  
 10.61281915, 9.54283796, 11.26940858, 11.40514896, 12.76704696,  
 10.95727404, 12.53310884, 10.53660577, 6.13174213, 8.22141479,  
 11.11480284, 7.86786733, 8.46800754, 8.51586463, 9.13625447,  
 9.34245631, 9.11866664, 9.495566 , 8.95509321, 10.63760259,  
 7.98105786, 9.62091863, 10.82472807, 10.27078687, 7.94382377,  
 12.27535451, 8.13050337, 7.74354516, 9.18857763, 8.50048233,  
 10.91163797, 9.5968706 , 9.38561823, 7.46683215, 10.37626812,  
 11.54365374, 9.26738518, 8.86708666, 7.51672746, 8.24395976,  
 6.40001408, 10.94290447, 9.54647858, 8.12274086, 9.33606364,  
 6.62231198, 9.09139994, 7.91419542, 9.40328905, 9.63625618,  
 10.84323552, 9.07989997, 9.43956278, 9.18170767, 10.22881222,  
 6.75527458])

*#14. Measure the performance using Metrics.*

```
from sklearn import metrics from sklearn.metrics import
mean_squared_error metrics.r2_score(y_test,test_pred)
```

0.4166836799902973

df.head()

					Whole	Shucked	Viscera	Shell
	Sex	Length			Diameter	Height	weight	weight
					weight	weight Rings		
0	1	0.455	0.365	0.095	0.5140	0.2245	0.1010	0.150 15.0

<b>1</b>	1	0.350	0.265	0.090	0.2255	0.0995	0.0485	0.070	7.0
<b>2</b>	0	0.530	0.420	0.135	0.6770	0.2565	0.1415	0.210	9.0
<b>3</b>	1	0.440	0.365	0.125	0.5160	0.2155	0.1140	0.155	10.0
<b>4</b>	-1	0.330	0.255	0.080	0.2050	0.0895	0.0395	0.055	7.0

```
linreg.predict([[0.455,0.365,0.095,0.5140,0.2245,0.1010,0.150,15.0]]) array([21.53400745])
```