RETAIL STORE STOCK INVENTORY ANALYTICS

ASSIGNMENT 4

```
from google.colab import files
uploaded = files.upload()
```

Choose Files abalone.csv

- **abalone.csv**(text/csv) - 191962 bytes, last modified: 11/05/2022 - 100% done

```
Saving abalone.csv to abalone (1).csv
```

```
import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
import warnings
import seaborn as sns
warnings.filterwarnings('ignore')
```

```
data = pd.read_csv("abalone.csv")
```

```
data
```

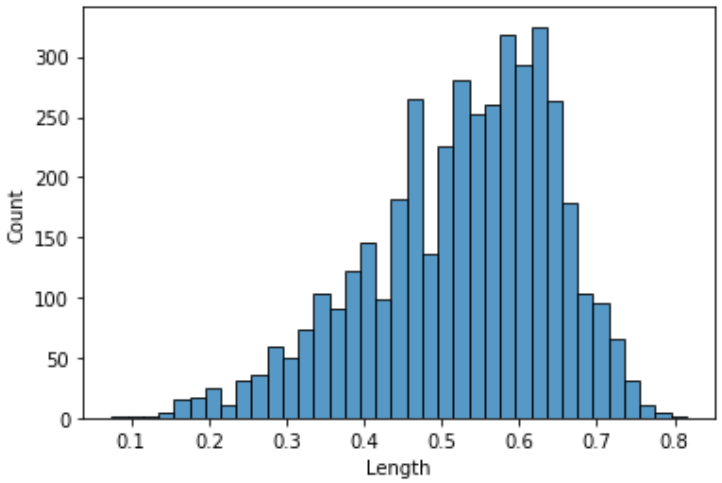
	Sex	Length	Diameter	Height	Whole weight	Shucked weight	Viscera weight
0	M	0.455	0.365	0.095	0.5140	0.2245	0.1010
1	M	0.350	0.265	0.090	0.2255	0.0995	0.0485
2	F	0.530	0.420	0.135	0.6770	0.2565	0.1415
3	M	0.440	0.365	0.125	0.5160	0.2155	0.1140
4	I	0.330	0.255	0.080	0.2050	0.0895	0.0395
...	...	...	...	...	...	...	...
4172	F	0.565	0.450	0.165	0.8870	0.3700	0.2390
4173	M	0.590	0.440	0.135	0.9660	0.4390	0.2145
4174	M	0.600	0.475	0.205	1.1760	0.5255	0.2875
4175	F	0.625	0.485	0.150	1.0945	0.5310	0.2610
4176	M	0.710	0.555	0.195	1.9485	0.9455	0.3765

4177 rows × 9 columns

```
data.head()
```

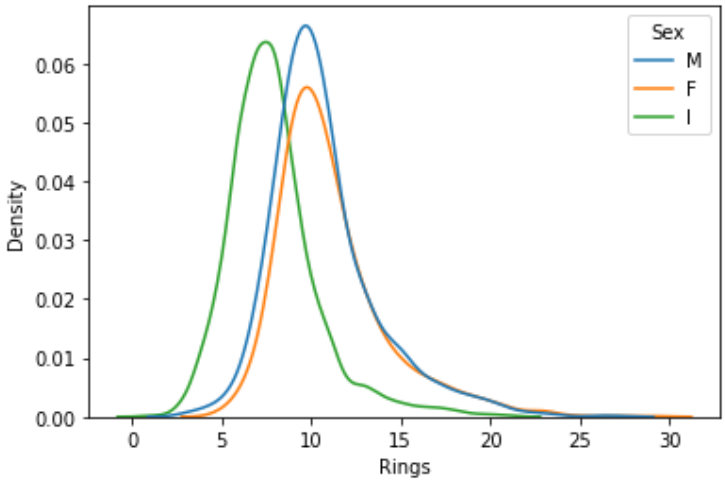
	Sex	Length	Diameter	Height	Whole weight	Shucked weight	Viscera weight	Shel
0	M	0.455	0.365	0.095	0.5140	0.2245	0.1010	
1	M	0.350	0.265	0.090	0.2255	0.0995	0.0485	
2	F	0.530	0.420	0.135	0.6770	0.2565	0.1415	
3	M	0.440	0.365	0.125	0.5160	0.2155	0.1140	
4	I	0.330	0.255	0.080	0.2050	0.0895	0.0395	

```
sns.histplot(x='Length', data = pd.read_csv("abalone.csv"));
```




```
sns.kdeplot(x='Rings', data = pd.read_csv("abalone.csv"), hue='Sex')
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fd3c1809310>
```

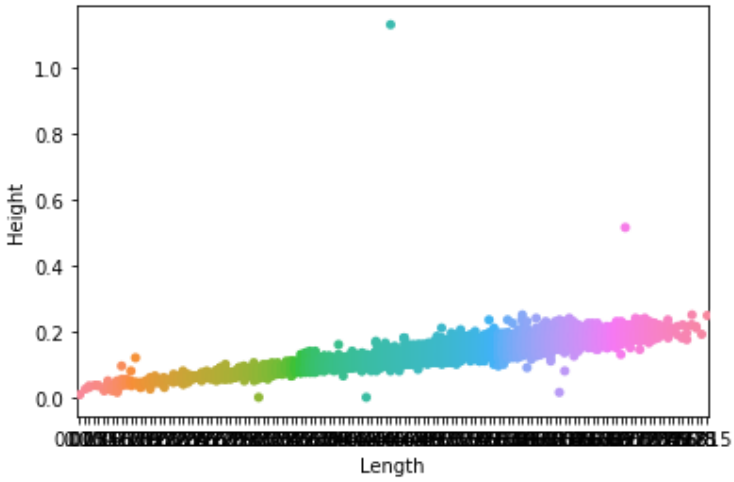



```
sns.boxplot(x='Length',y='Height',data = pd.read_csv("abalone.csv"))
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fd3c12bcc50>
```

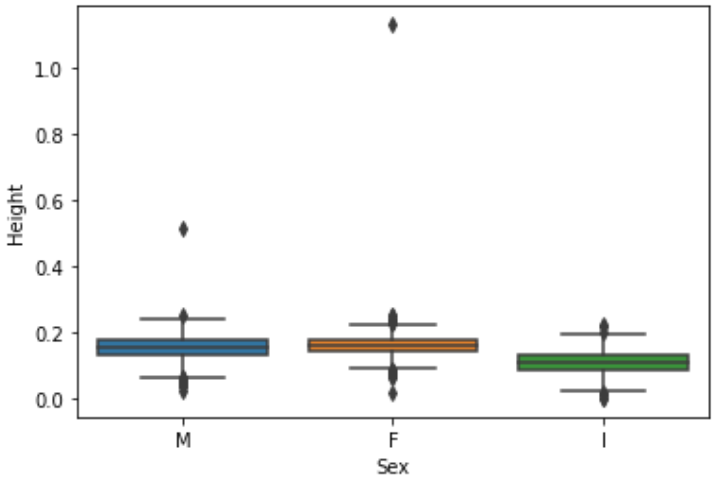

```
sns.stripplot(x="Length", y="Height", data = pd.read_csv("abalone.csv"))
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fd3c05c5890>
```



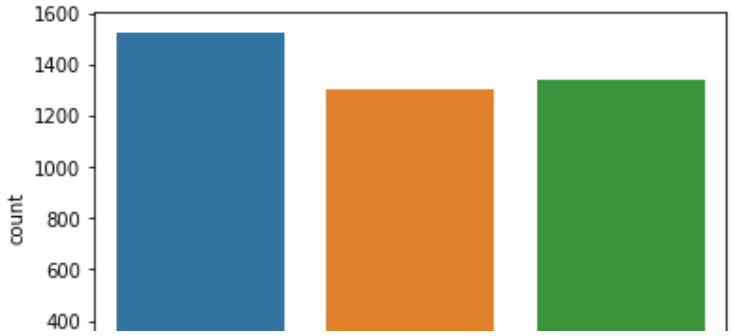

```
sns.boxplot(x="Sex", y="Height", data = pd.read_csv("abalone.csv"))
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fd3c0685190>
```



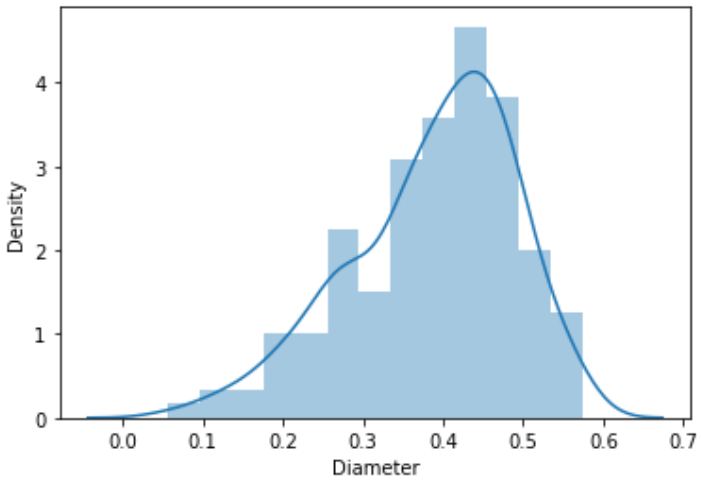

```
sns.barplot(x='Sex',y='Diameter',data = pd.read_csv("abalone.csv"))
sns.countplot(x='Sex',data = pd.read_csv("abalone.csv"))
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fd3c023afd0>
```



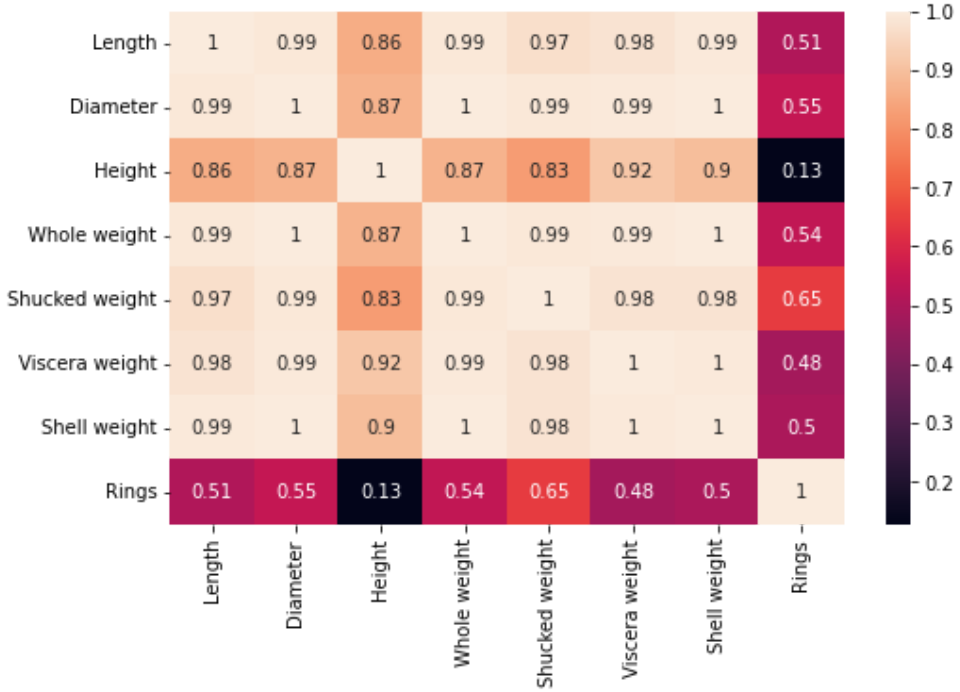

```
sns.distplot(data['Diameter'].head(300))
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fd3c01c06d0>
```




```
fig=plt.figure(figsize=(8,5))
sns.heatmap(data.head().corr(),annot=True)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fd3c0694550>
```

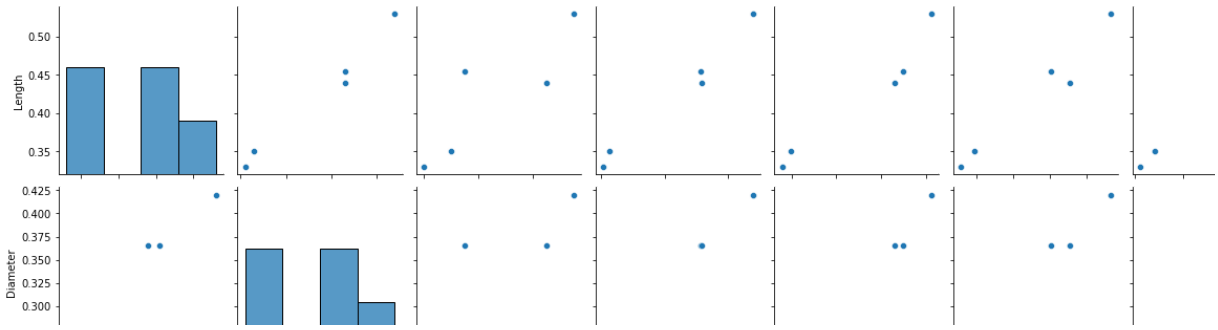


```
sns.pairplot(data.head(),hue='Height')
```

```
<seaborn.axisgrid.PairGrid at 0x7fd3c129b250>
```


```
sns.pairplot(data.head())
```

```
<seaborn.axisgrid.PairGrid at 0x7fd3bca8a350>
```

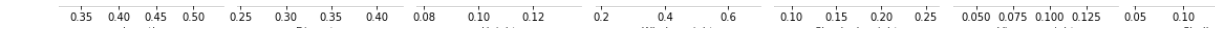



```
data.head()
```

	Sex	Length	Diameter	Height	Whole weight	Shucked weight	Viscera weight	Shel
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	

```
data.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
```



```
data.tail()
```

```
data.describe()
```

	Length	Diameter	Height	Whole weight	Shucked weight	Viscera weight	
count	4177.000000	4177.000000	4177.000000	4177.000000	4177.000000	4177.000000	4
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	
75%	0.615000	0.480000	0.165000	1.153000	0.502000	0.253000	
max	0.815000	0.650000	1.130000	2.825500	1.488000	0.760000	

```
data.mode().T
```

	0	1
Sex	M	NaN
Length	0.55	0.625
Diameter	0.45	NaN
Height	0.15	NaN
Whole weight	0.2225	NaN
Shucked weight	0.175	NaN
Viscera weight	0.1715	NaN
Shell weight	0.275	NaN
Rings	9.0	NaN

```
data.shape
```

```
(4177, 9)
```

```
data.skew()
```

```
Length           -0.639873
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
```

```
data.nunique()
```

```
Sex                 3
Length            134
Diameter          111
Height             51
Whole weight     2429
Shucked weight   1515
Viscera weight    880
Shell weight      926
Rings              28
dtype: int64
```

```
data.kurt()
```

```
Length            0.064621
Diameter         -0.045476
Height           76.025509
Whole weight     -0.023644
Shucked weight    0.595124
Viscera weight    0.084012
Shell weight      0.531926
Rings             2.330687
dtype: float64
```

```
data.var()
```

```
Length            0.014422
Diameter          0.009849
Height            0.001750
Whole weight      0.240481
Shucked weight    0.049268
Viscera weight    0.012015
Shell weight      0.019377
Rings            10.395266
dtype: float64
```

```
data.isna()
```

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

```
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
```

4177 rows × 9 columns

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

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

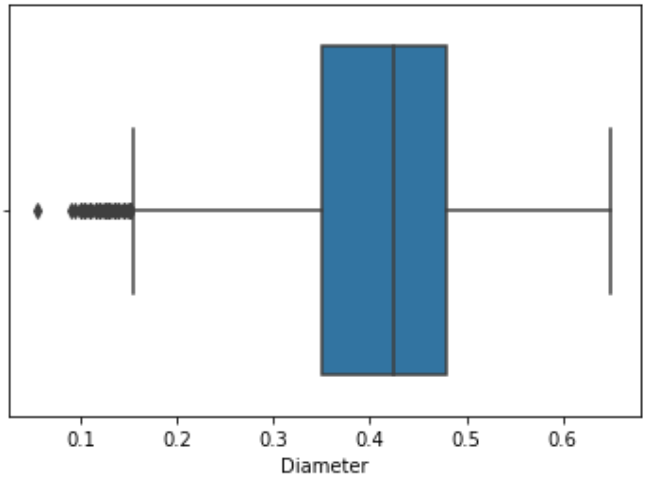
```
Sex               0
Length            0
Diameter          0
Height            0
Whole weight      0
Shucked weight    0
Viscera weight    0
Shell weight      0
Rings             0
dtype: int64
```

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

```
0
```

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

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fd3b9043710>
```




```
quant=data.quantile(q=[0.25,0.75])
quant
```

	Length	Diameter	Height	Whole weight	Shucked weight	Viscera weight	Shell
0.25	0.450	0.35	0.115	0.4415	0.186	0.0935	
0.75	0.615	0.48	0.165	1.1530	0.502	0.2530	

```
iqr=quant.loc[0.75]-quant.loc[0.25]
iqr
```

```
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
```

```
low=quant.loc[0.25]-(1.5*iqr)
low
```

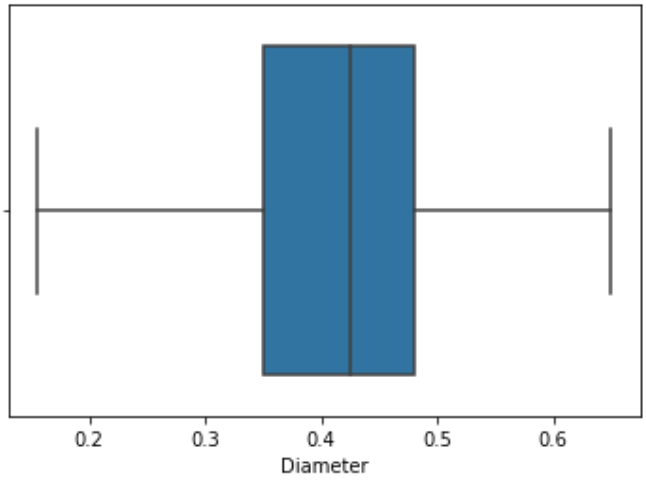
```
Length            0.20250
Diameter          0.15500
Height            0.04000
Whole weight     -0.62575
Shucked weight   -0.28800
Viscera weight   -0.14575
Shell weight     -0.16850
Rings             3.50000
dtype: float64
```

```
up=quant.loc[0.75]+(1.5*iqr)
up
```

```
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
```

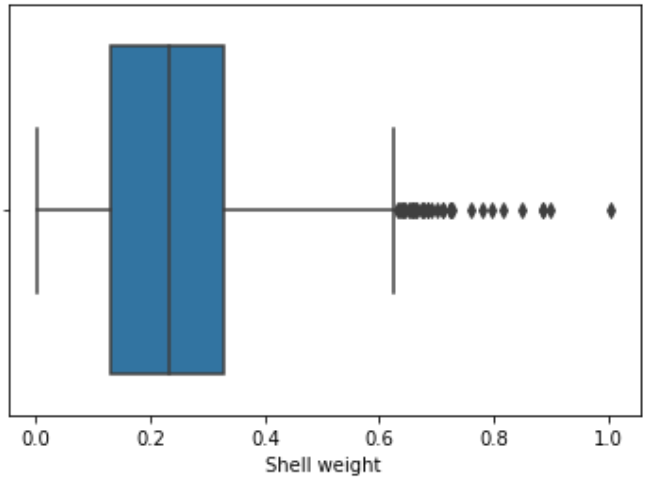
```
data['Diameter']=np.where(data['Diameter']<0.155,0.4078,data['Diameter'])
sns.boxplot(data['Diameter'])
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fd3c01c0f50>
```



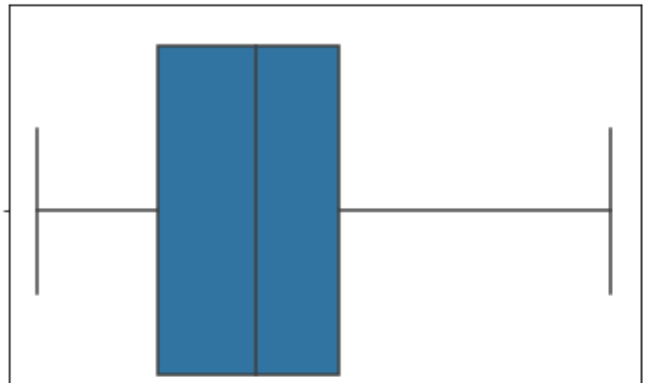

```
sns.boxplot(data['Shell weight'])
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fd3b8f59c90>
```




```
data['Shell weight']=np.where(data['Shell weight']>0.61,0.2388, data['Shell weight'])
sns.boxplot(data['Shell weight'])
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fd3b9102310>
```



```
data['Sex'].replace({'M':1,'F':0,'I':2},inplace=True)
data
```

	Sex	Length	Diameter	Height	Whole weight	Shucked weight	Viscera weight
0	1	0.455	0.365	0.095	0.5140	0.2245	0.1010
1	1	0.350	0.265	0.090	0.2255	0.0995	0.0485
2	0	0.530	0.420	0.135	0.6770	0.2565	0.1415
3	1	0.440	0.365	0.125	0.5160	0.2155	0.1140
4	2	0.330	0.255	0.080	0.2050	0.0895	0.0395
...	...	...	...	...	...	...	...
4172	0	0.565	0.450	0.165	0.8870	0.3700	0.2390
4173	1	0.590	0.440	0.135	0.9660	0.4390	0.2145
4174	1	0.600	0.475	0.205	1.1760	0.5255	0.2875
4175	0	0.625	0.485	0.150	1.0945	0.5310	0.2610
4176	1	0.710	0.555	0.195	1.9485	0.9455	0.3765

4177 rows × 9 columns

```
x=data.drop(columns= ['Rings'])
y=data['Rings']
x
```

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

```
y
```

```
0       15
1        7
2        9
3       10
4        7
        ..
4172    11
4173    10
4174     9
4175    10
4176    12
Name: Rings, Length: 4177, dtype: int64
```

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

```
array([[-0.0105225 , -0.57455813, -0.50179694, ..., -0.60768536,
        -0.72621157, -0.64358742],
       [-0.0105225 , -1.44898585, -1.57304487, ..., -1.17090984,
        -1.20522124, -1.25742181],
       [-1.26630752,  0.05003309,  0.08738942, ..., -0.4634999 ,
        -0.35668983, -0.18321163],
       ...,
       [-0.0105225 ,  0.6329849 ,  0.67657577, ...,  0.74855917,
         0.97541324,  0.56873549],
       [-1.26630752,  0.84118198,  0.78370057, ...,  0.77334105,
         0.73362741,  0.47666033],
       [-0.0105225 ,  1.54905203,  1.53357412, ...,  2.64099341,
         1.78744868,  2.00357336]])
```

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

```
array([[-0.0105225 , -0.57455813, -0.50179694, ..., -0.60768536,
        -0.72621157, -0.64358742],
       [-0.0105225 , -1.44898585, -1.57304487, ..., -1.17090984,
        -1.20522124, -1.25742181],
       [-1.26630752,  0.05003309,  0.08738942, ..., -0.4634999 ,
        -0.35668983, -0.18321163],
       ...,
       [-0.0105225 ,  0.6329849 ,  0.67657577, ...,  0.74855917,
         0.97541324,  0.56873549],
```

```
            [-1.26630752,  0.84118198,  0.78370057, ...,  0.77334105,
              0.73362741,  0.47666033],
            [-0.0105225 ,  1.54905203,  1.53357412, ...,  2.64099341,
              1.78744868,  2.00357336]])
```

```
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)
print(x_train.shape, x_test.shape)
```

```
     (3341, 8) (836, 8)
```

```
from sklearn.linear_model import LinearRegression
MLR=LinearRegression()
```

```
MLR.fit(x_train,y_train)
```

```
     LinearRegression()
```

```
y_pred=MLR.predict(x_test)
y_pred
```

```
            13.48490299, 10.12650933, 11.21370417,  9.78466293, 10.41119396,
            10.23921019,  6.36916583, 13.1372858 , 11.32054954,  9.80775249,
            11.57237095, 10.07203642,  7.43545573,  5.58346065,  9.74793101,
             7.31047123, 10.8820856 ,  6.19340828, 12.36225628,  6.39327262,
             8.81895465, 13.90591853,  8.80535546, 12.27266141, 10.26962165,
            10.85694506,  9.23548942,  8.18722783,  7.9314565 , 11.92528829,
             9.5247047 ,  9.45783571, 16.77713562,  9.81380449,  8.46038648,
            11.83738305, 12.71033273,  9.34405315,  7.54528966, 13.64794587,
            11.34200392, 10.97452307, 10.86062125,  9.48877503,  9.56273905,
             8.53527877, 17.40139364, 11.24973686, 10.19084319, 14.6810828 ,
            15.18292843,  7.20431777, 15.80313464,  9.32387206, 10.28861779,
            10.77799067,  4.96252544, 10.269737  , 10.43836225, 10.24428332,
            11.65337577, 10.64178254, 10.06404154,  8.96246351, 11.83909671,
            11.35437817,  8.95829941,  7.29838974, 12.37340949, 11.54228431,
             7.74176421, 10.78879979, 11.4815144 , 11.19751406,  9.86782995,
             7.30624898, 10.5956816 ,  8.97633041, 11.53955456, 11.88849179,
             6.88243609, 11.87817948, 11.17451193, 10.59086154,  8.92981204,
             9.15140208,  8.64904977, 10.76139589,  6.97769248,  9.09361918,
            10.84452403,  9.32738207,  8.52766816,  9.25076866, 10.53109827,
            10.46842039,  8.84043296, 10.36812729,  7.50780249, 14.35988376,
            11.73039012,  6.51835523, 10.89979546, 11.14016829, 12.24422863,
             7.86733091,  8.91842129, 13.15426474, 12.31895654, 11.46271619,
             8.39177521,  8.45906741, 12.9984135 , 11.77007517, 11.18372644,
             7.14572743, 11.02640894,  6.65127847,  9.85472883, 11.27038817,
             7.35953699, 10.44403379,  7.29859572,  8.90335212, 12.52643795,
            11.09625195,  7.84566972, 11.5852972 ,  7.4546814 , 14.86525194,
             9.29995656, 10.32784855,  5.80097179, 10.11812098, 10.77775104,
            11.9760842 ,  8.43311782, 10.63388666,  7.50089428, 11.0369102 ,
             7.09502581,  9.23035555,  9.76860135,  6.92151743, 12.01523362,
             8.81054443,  9.62877859,  7.96896058,  7.71984694,  7.68020093,
            10.23120865,  6.89906918,  7.32170166,  9.02583862,  6.48738961,
             6.93417081, 15.95467978, 10.6081471 , 14.49974856, 10.17002502,
            13.90249709,  8.07489324, 11.8946688 ,  9.94649337,  9.54068262,
            14.26789622, 14.36486553,  9.78681486,  6.93218256, 10.4955901 ,
            10.16767472,  9.26554889, 10.84529739, 11.44745539,  9.76441072,
             8.87609082, 13.02042454,  8.99605516, 11.53467843,  7.02093096,
```

```
            12.11393258,  7.56076648,  6.54986875, 13.02266295, 10.27119835,
             8.71903971, 10.45717161, 11.51333691,  8.894591  , 10.14035002,
            15.03489038,  7.9023417 , 10.35767151,  8.38191762,  4.99382965,
             9.01422232,  8.42087154, 11.36455901,  8.46780113,  8.59675494,
            10.14161831, 12.5837006 ,  9.90706593,  8.32828309, 10.5008917 ,
             9.70482555, 11.83506591, 12.70682392, 11.62235276,  6.823856  ,
             7.06326511,  6.87944689,  9.00322861,  8.20973818, 10.04175823,
             8.99111019,  9.38651656, 10.46032737, 11.939472  ,  8.48698233,
             8.13919833,  9.62546787,  6.45248215, 10.94000078, 11.8339691 ,
            12.66459793, 12.70517249,  9.25078847, 10.38310709,  9.04872162,
             7.50989425, 14.89629939,  8.60763701, 10.34795829,  9.51950782,
             5.19366173,  8.21412497,  9.56131108,  9.33066533,  6.06809077,
            13.68283263, 12.46155096, 10.18275967,  7.51857258,  7.15454444,
            10.74029367,  9.02599297,  6.82944699, 15.69511335, 11.75229526,
            11.63487837,  7.94134915,  6.11072922, 15.95079521,  8.32400194,
            10.85399816,  9.92283542,  7.23602627, 12.51782579,  7.80926347,
             9.18088859,  8.59544991, 11.47292828,  9.68565888,  8.58007892,
             8.46863237,  7.85634124,  7.78121552,  9.04662986, 13.60855079,
            10.96485212, 10.12910701,  7.99351895,  7.90389965,  8.40685848,
            11.9644303 , 11.25551129, 17.84613443,  6.44971756, 12.09302199,
            10.2539475 ,  6.5584797 , 12.44780802,  9.61483012, 10.14984344,
             8.66854509])
```

```
pred=MLR.predict(x_train)
pred
```

```
array([ 8.93054315,  7.83233303,  9.08534415, ..., 12.22316511,
        9.66287984,  9.20960728])
```

```
from sklearn.metrics import r2_score
accuracy=r2_score(y_test,y_pred)
accuracy
```

```
0.5406345167003427
```

```
MLR.predict([[1,0.455,0.365,0.095,0.5140,0.2245,0.1010,0.150]])
```

```
array([11.94235363])
```

```
from sklearn import metrics
from sklearn.metrics import mean_squared_error
np.sqrt(mean_squared_error(y_test,y_pred))
```

```
2.2428288608273217
```

```
from sklearn.linear_model import Lasso, Ridge
#intialising model
lso=Lasso(alpha=0.01,normalize=True)
#fit the model
lso.fit(x_train,y_train)
Lasso(alpha=0.01, normalize=True)
#prediction on test data
lso_pred=lso.predict(x_test)
```

```
#coef
coef=lso.coef_
coef
```

```
array([-0.        ,  0.        ,  0.        ,  0.25998603,  0.        ,
        0.        ,  0.        ,  1.16873649])
```

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

```
0.3228094234288654
```

```
np.sqrt(mean_squared_error(y_test,lso_pred))
```

```
2.72315529870534
```

```
#initialising model
rg=Ridge(alpha=0.01,normalize=True)
#fit the model
rg.fit(x_train,y_train)
Ridge(alpha=0.01, normalize=True)
#prediction
rg_pred=rg.predict(x_test)
rg_pred
```

```
           13.05922767, 10.07740919, 10.98550809,  9.92191713, 10.40272206,
           10.34033522,  6.28773065, 12.57615794, 11.54234252,  9.93311107,
           11.48331747, 10.38890668,  7.3007348 ,  5.46044182,  9.66565339,
            7.31294707, 10.8866748 ,  6.1434975 , 12.34159451,  6.30288007,
            8.88229062, 13.97696966,  8.69577194, 12.25278912, 10.03706181,
           11.04280787,  9.21551613,  8.17045263,  7.98250106, 12.04705276,
            9.59235621,  9.73317426, 16.55189621,  9.7202254 ,  8.59429731,
           11.37526054, 13.4590507 ,  9.41336747,  7.48318675, 13.35897892,
           11.33785195, 10.86083281, 10.92862271,  9.60315586,  9.78195658,
            8.69886947, 16.76150515, 11.30590633, 10.19853983, 14.38625663,
           14.63523841,  7.19323717, 15.15715082,  9.49576288, 10.59357647,
           10.77522826,  5.19163351, 10.35507626, 10.38841878, 10.29427328,
           11.69525547, 10.5623982 , 10.05323807,  8.92585431, 11.72910293,
           11.6631348 ,  8.92455148,  7.19330881, 12.37909569, 11.4605733 ,
            7.72978037, 10.97820651, 10.89320729, 11.31680001, 10.19804407,
            7.22586069, 10.47266094,  8.9157606 , 11.43407525, 11.44996836,
            6.82090967, 11.93230743, 11.26937635, 10.7569687 ,  9.33490233,
            9.41764715,  8.60875066, 11.12790702,  6.99099243,  9.00856694,
           11.14626974,  9.58772879,  8.8646477 ,  9.44872867, 10.77895884,
           10.73959181,  8.89249339, 10.0895149 ,  7.42941277, 13.94440489,
           11.57828937,  6.4785139 , 10.66900051, 11.05486698, 12.1543264 ,
            7.82491263,  9.01385332, 13.13998956, 12.42378003, 11.63715517,
            8.55651436,  8.30298752, 12.70739261, 11.40285281, 11.48053045,
            7.16766658, 10.85119887,  6.57289812,  9.57134011, 11.27060835,
            7.44593829, 10.59908067,  7.67599058,  9.61982825, 12.37234833,
           11.03016653,  7.82413361, 11.87098192,  7.48604497, 14.39433976,
            9.35372673, 10.51670145,  5.70958784,  9.9336436 , 10.89501108,
           11.87254717,  8.44364398, 10.75387196,  7.50409279, 11.2401361 ,
            7.04564057,  9.39025153, 10.23115972,  6.79321352, 11.78085371,
            8.86010019,  9.53665974,  7.87113557,  7.67096884,  7.63086303,
```

```
       10.37646256,  6.76562241,  7.22670986,  9.33338411,  6.46233997,
        6.83905009, 14.78116053, 10.85939905, 14.23059425, 10.38341675,
       13.76811886,  8.11957724, 11.70852822, 10.01599988,  9.50003005,
       13.59088743, 13.43599312, 10.14488342,  6.8545766 , 10.56344905,
       10.24165828,  9.300253  , 11.0721801 , 11.08511087,  9.82087237,
        9.00029956, 12.9520922 ,  9.1058268 , 11.63267643,  6.96362854,
       11.99927272,  9.90501054,  6.50268158, 12.59071959, 10.65676634,
        8.66630687, 10.45392216, 11.4678933 ,  8.77955971, 10.41148378,
       14.55053011,  7.95503552, 10.73593933,  8.27713745,  5.22484783,
        9.00794353,  8.49048587, 11.47252627,  8.45546467,  8.55121195,
       10.01199721, 12.51368957,  9.98949806,  8.3821651 , 10.7419689 ,
        9.72670851, 11.68365118, 12.1029351 , 11.51448086,  6.73888086,
        7.01394268,  6.80741666,  9.14757642,  8.29001683, 10.09373915,
        9.36391693,  9.29773783, 10.85964702, 12.08779329,  8.74502331,
        8.09459045,  9.51525423,  6.33277899, 11.04086756, 11.75107245,
       12.27723558, 12.42100228,  9.47646341, 10.8579674 ,  9.02111553,
        7.55515419, 14.1563324 ,  8.68599153, 10.27370728,  9.40634772,
        5.3821725 ,  8.20379804,  9.62208056,  9.50581514,  5.9342606 ,
       13.23587509, 12.53245039, 10.1738823 ,  7.43566258,  7.10693715,
       10.98973189,  9.01472461,  6.81512437, 14.69992271, 12.29547787,
       11.64462751,  8.19876166,  5.99254922, 15.20104749,  8.37736267,
       11.0041235 , 10.17324249,  7.25615289, 12.20976261,  7.83793831,
        9.30036041,  8.5661528 , 11.60267884, 10.27759708,  8.6213238 ,
        8.47074609,  7.79756423,  7.71772895,  9.01076046, 13.44135041,
       10.07296551, 10.21003258,  8.3198457 ,  7.94731013,  8.618621  ,
       12.10157668, 11.29114059, 16.32886707,  6.29359295, 11.8689986 ,
       10.42408341,  6.48698129, 12.28080793,  9.60772841, 10.45762507,
        8.63370565])
```

```
rg.coef_
```

```
array([-0.28253279,  0.72253142, -0.04237736,  0.55306056,  3.33012763,
       -3.41149089, -0.77137749,  1.30517786])
```

```
metrics.r2_score(y_test,rg_pred)
```

```
0.514945395267461
```

```
np.sqrt(mean_squared_error(y_test,rg_pred))
```

```
2.3046886990204123
```

Colab paid products - Cancel contracts here

✓ 0s completed at 9:55 AM