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

df=pd.read_csv('abalone.csv')

df.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
```

```
df.tail()
```

```
     Sex  Length  Diameter  Height  Whole weight  Shucked weight  \
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

      Viscera weight  Shell weight  Rings
4172          0.2390        0.2490     11
4173          0.2145        0.2605     10
4174          0.2875        0.3080      9
4175          0.2610        0.2960     10
4176          0.3765        0.4950     12
```

```
df.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
```

```
df.rename({'Rings':'Age'},axis=1, inplace=True)
```

```
df
```

```
     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

      Viscera weight  Shell weight  Age
0             0.1010        0.1500   15
1             0.0485        0.0700    7
2             0.1415        0.2100    9
3             0.1140        0.1550   10
4             0.0395        0.0550    7
...              ...           ...  ...
4172          0.2390        0.2490   11
4173          0.2145        0.2605   10
4174          0.2875        0.3080    9
4175          0.2610        0.2960   10
4176          0.3765        0.4950   12

[4177 rows x 9 columns]
```

```
hi= pd.Series(df.Age)
hi
```

```
0       15
1        7
2        9
3       10
4        7
        ..
4172    11
4173    10
4174     9
```

```
4175    10
4176    12
Name: Age, Length: 4177, dtype: int64
```

```
for i in hi:
    age=(hi+1.5)
age
```

```
0       16.5
1        8.5
2       10.5
3       11.5
4        8.5
        ...
4172    12.5
4173    11.5
4174    10.5
4175    11.5
4176    13.5
Name: Age, Length: 4177, dtype: float64
```

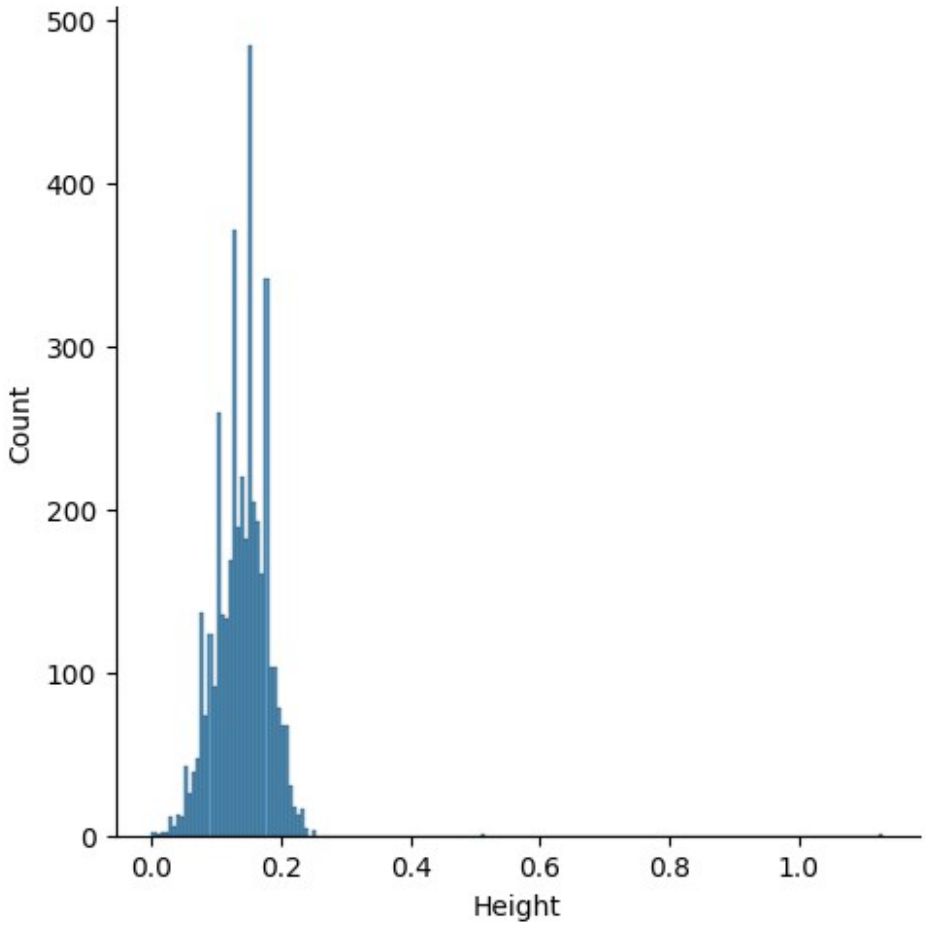
```
y=age
y
```

```
0       16.5
1        8.5
2       10.5
3       11.5
4        8.5
        ...
4172    12.5
4173    11.5
4174    10.5
4175    11.5
4176    13.5
Name: Age, Length: 4177, dtype: float64
```

Univariate analysis

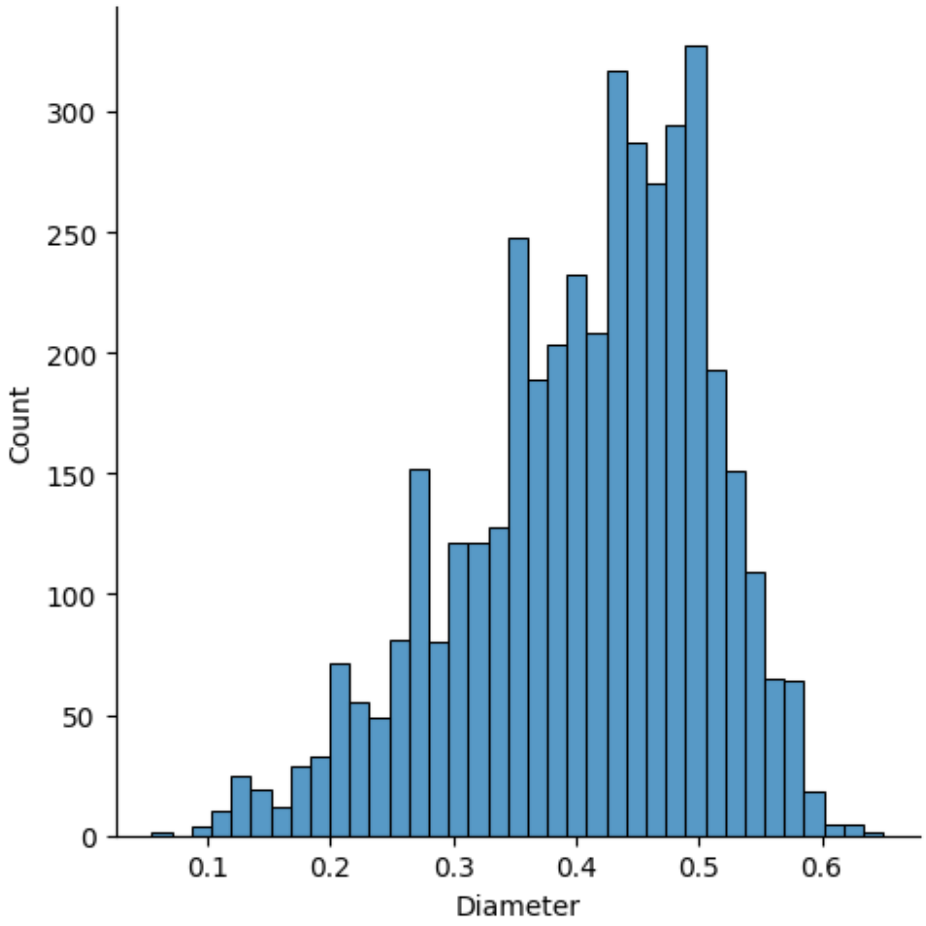
```
sns.displot(df.Height)
```

```
<seaborn.axisgrid.FacetGrid at 0x21f6c00ebc0>
```



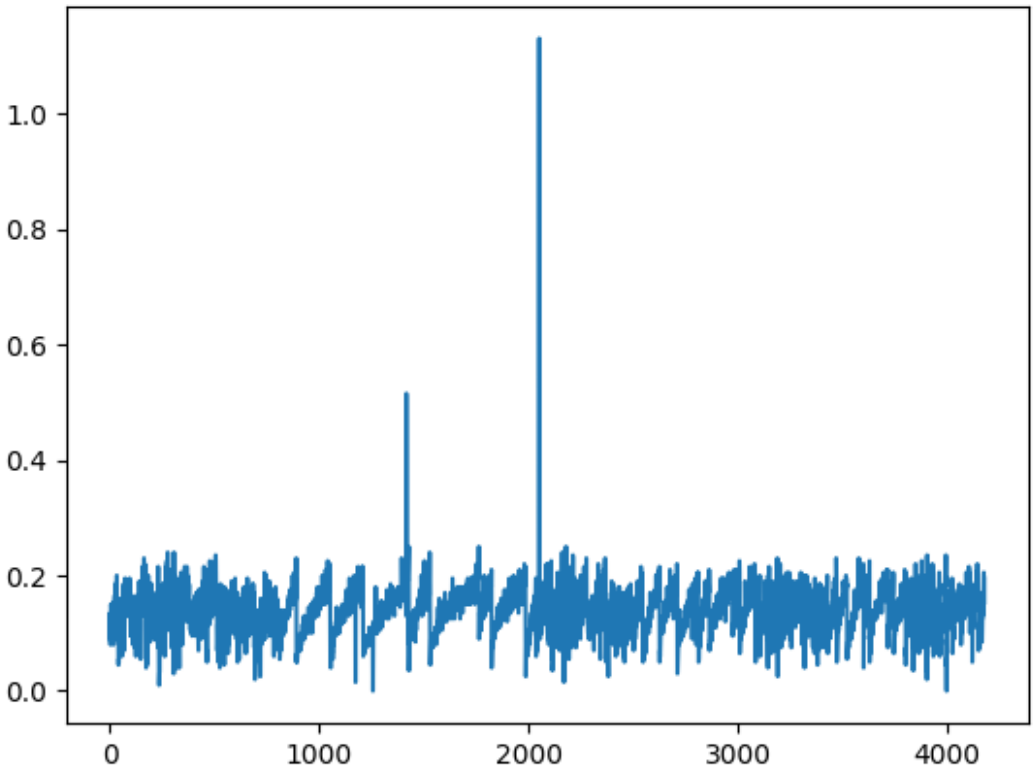

```
sns.displot(df.Diameter)
```

```
<seaborn.axisgrid.FacetGrid at 0x21f6e2cfdf0>
```




```
df.Height.plot()
```

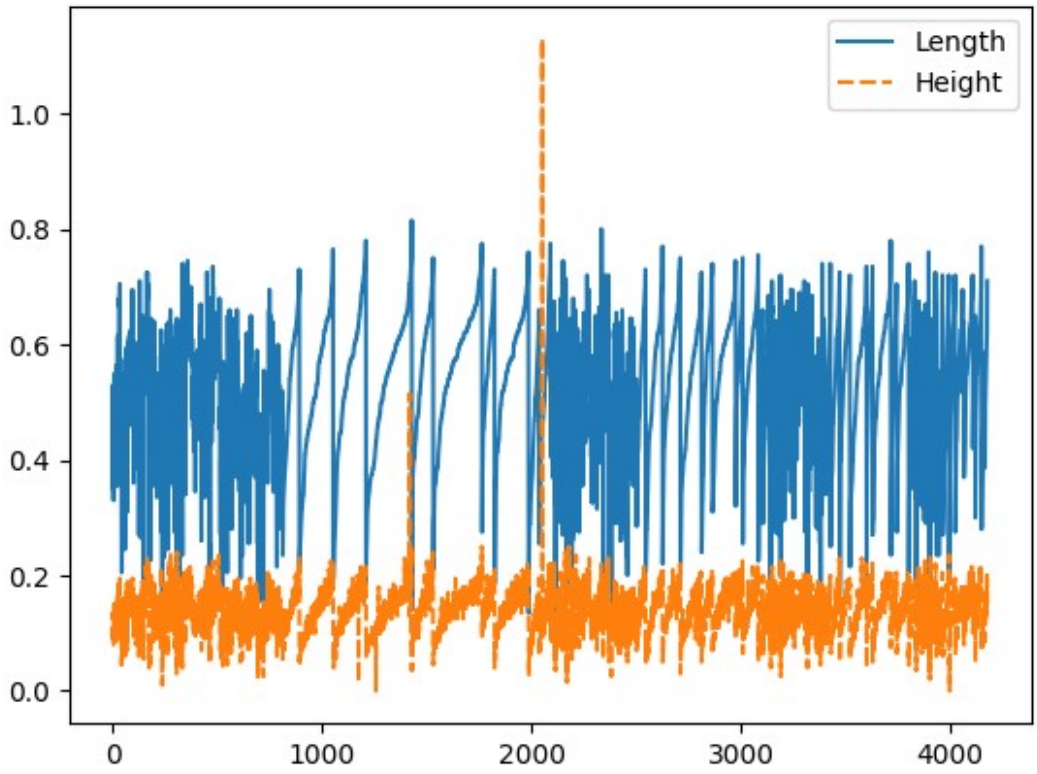
```
<AxesSubplot:>
```



Bi-Variate Analysis

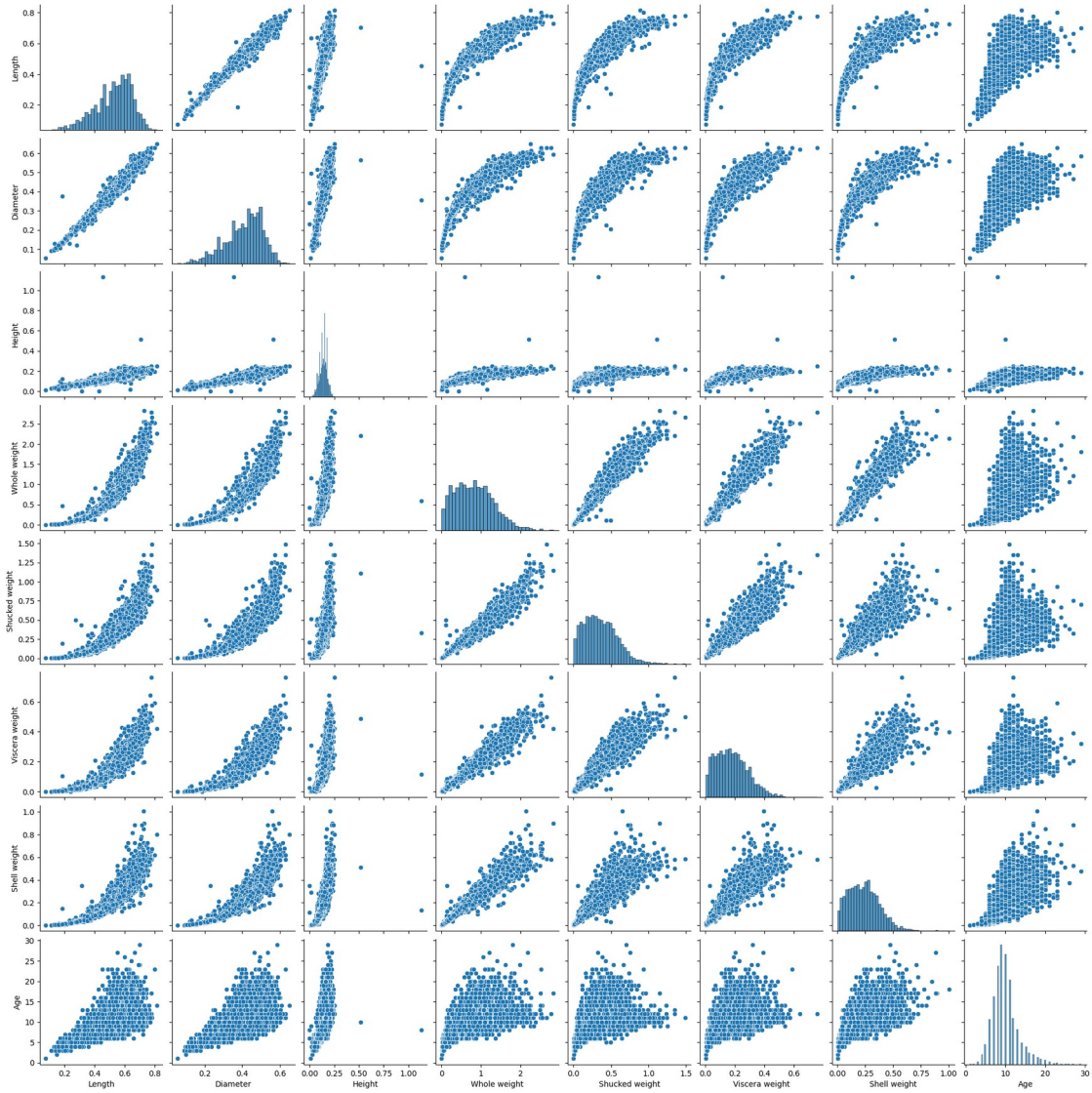
```
sns.lineplot([df.Length,df.Height])
```

```
<AxesSubplot:>
```



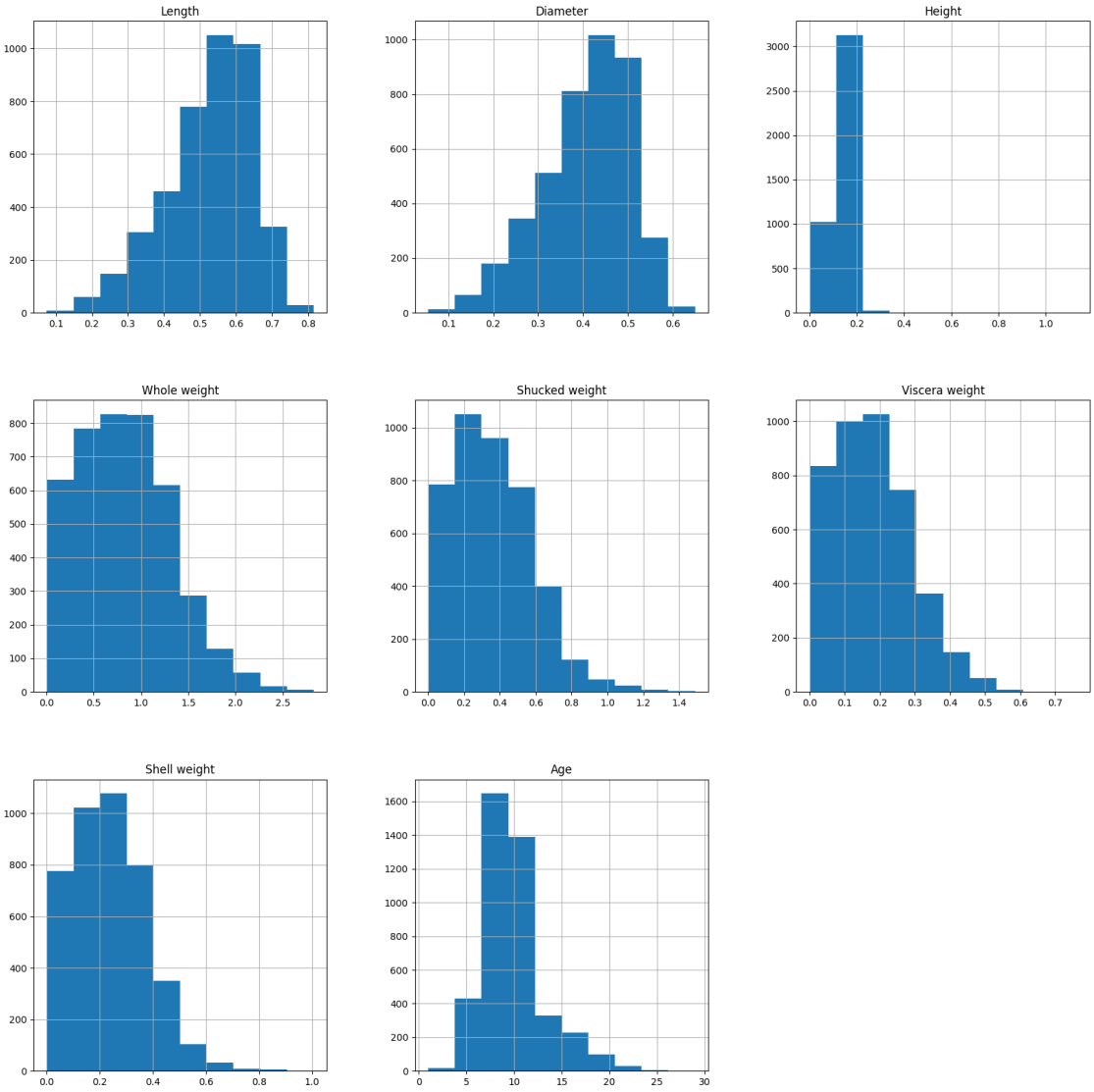

```
sns.pairplot(df)
```

```
<seaborn.axisgrid.PairGrid at 0x21f6e61a4a0>
```



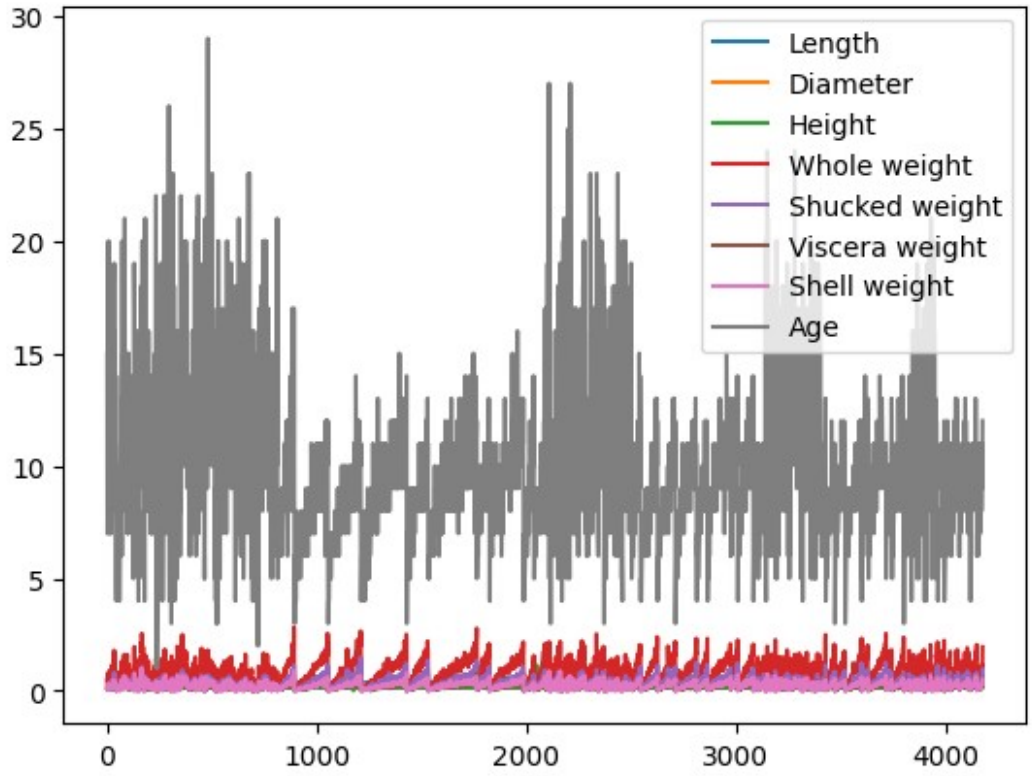

```
df.hist(figsize=(20,20))
```

```
array([[<AxesSubplot:title={'center':'Length'}>,
        <AxesSubplot:title={'center':'Diameter'}>,
        <AxesSubplot:title={'center':'Height'}>],
       [<AxesSubplot:title={'center':'Whole weight'}>,
        <AxesSubplot:title={'center':'Shucked weight'}>,
        <AxesSubplot:title={'center':'Viscera weight'}>],
       [<AxesSubplot:title={'center':'Shell weight'}>,
        <AxesSubplot:title={'center':'Age'}>, <AxesSubplot:>]],
      dtype=object)
```




```
df.plot()
```

```
<AxesSubplot:>
```




```
df.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          Age
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
```

```
df.isnull().any()
```

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

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

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

```
df.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   Age             4177 non-null   int64
dtypes: float64(7), int64(1), object(1)
memory usage: 293.8+ KB
```

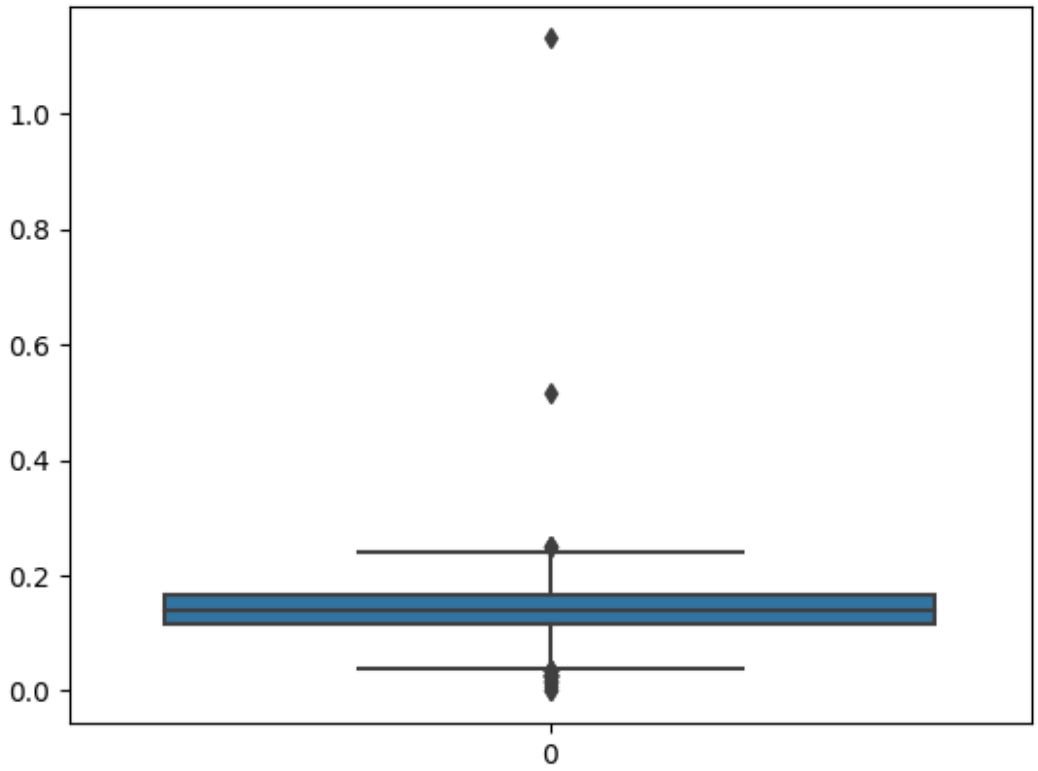
Outlier Detection

```
df.shape
```

```
(4177, 9)
```

```
sns.boxplot(df.Height)
```

```
<AxesSubplot:>
```




```
q1=df.Height.quantile(0.25)
q3=df.Height.quantile(0.75)
```

```
IQR=q3-q1
```

```
upper_limit= q3 + 1.5*IQR
```

```
lower_limit= q1 - 1.5*IQR
```

```
upper_limit
```

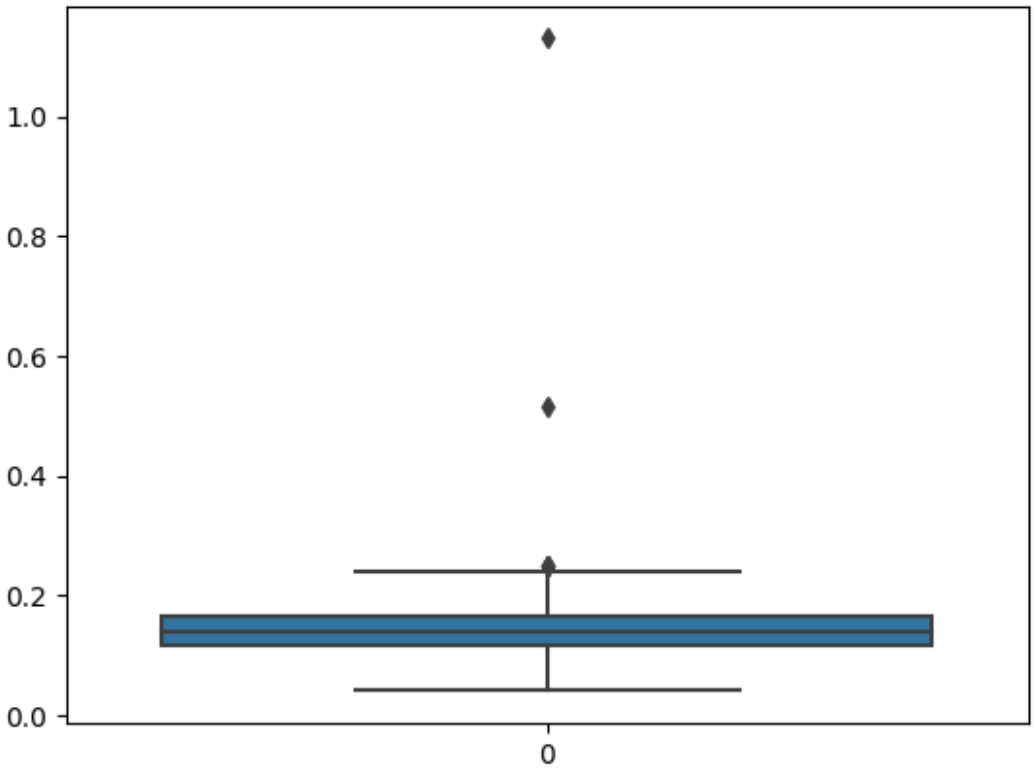
```
0.24000000000000002
```

```
lower_limit
```

```
0.039999999999999994
```

```
df.median()
```

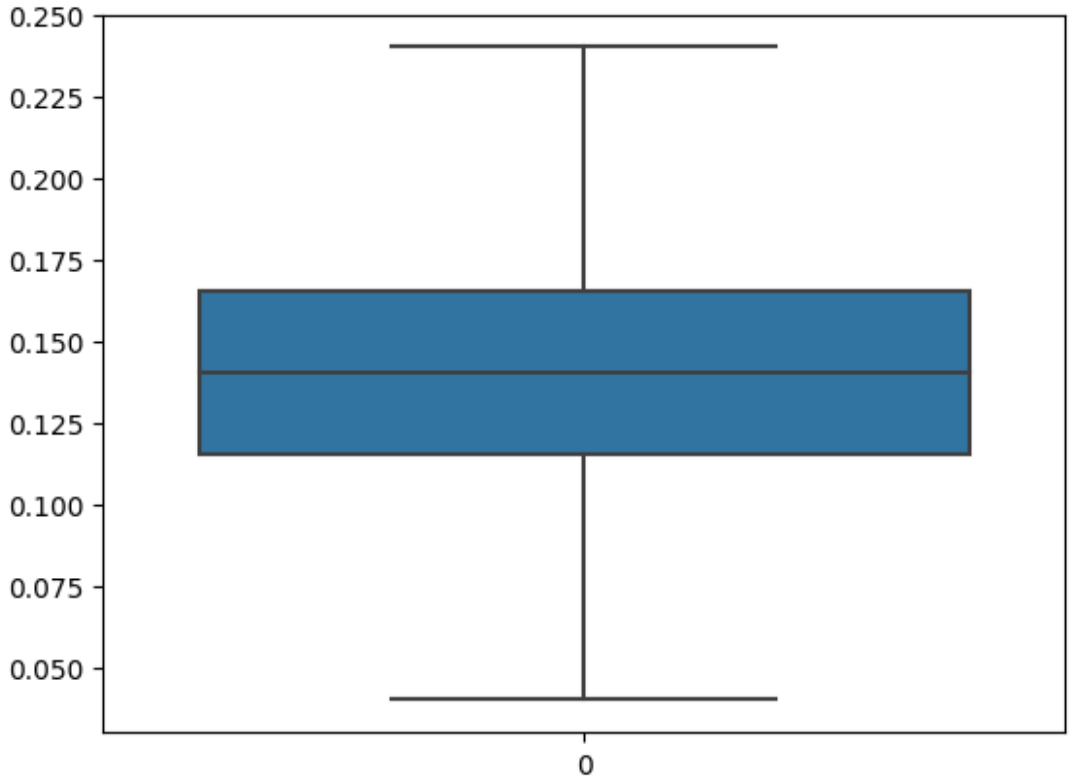
```
C:\Users\nojma\AppData\Local\Temp\ipykernel_21444\530051474.py:1:
FutureWarning: Dropping of nuisance columns in DataFrame reductions
(with 'numeric_only=None') is deprecated; in a future version this
will raise TypeError.  Select only valid columns before calling the
reduction.
  df.median()

Length            0.5450
Diameter          0.4250
Height            0.1400
Whole weight      0.7995
Shucked weight    0.3360
Viscera weight    0.1710
Shell weight      0.2340
Age               9.0000
dtype: float64

df['Height']=
np.where(df['Height']<lower_limit,0.039999999999999994 ,df['Height'])

sns.boxplot(df.Height)

<AxesSubplot:>
```




```
df['Height']=
np.where(df['Height']>upper_limit,0.24000000000000002 ,df['Height'])
```

```
sns.boxplot(df.Height)
```

```
<AxesSubplot:>
```




```
df.shape
```

```
(4177, 9)
```

The Categorical columns and perform Encoding.

```
df.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  Age
```

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

```
df.Sex.value_counts()
```

```
M    1528
I    1342
F    1307
Name: Sex, dtype: int64
```

```
from sklearn.preprocessing import LabelEncoder
```

```
le=LabelEncoder()
```

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

```
df.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  Age
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
```

```
df.tail()
```

```
      Sex  Length  Diameter  Height  Whole weight  Shucked weight  \
4172    0   0.565     0.450   0.165        0.8870          0.3700
4173    2   0.590     0.440   0.135        0.9660          0.4390
4174    2   0.600     0.475   0.205        1.1760          0.5255
4175    0   0.625     0.485   0.150        1.0945          0.5310
4176    2   0.710     0.555   0.195        1.9485          0.9455

      Viscera weight  Shell weight  Age
4172          0.2390        0.2490   11
4173          0.2145        0.2605   10
4174          0.2875        0.3080    9
4175          0.2610        0.2960   10
4176          0.3765        0.4950   12
```

Split the Data into Dependent and Independent variables.

```
x=df.drop(columns=['Age'],axis=1)
x
```

```
      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
...   ...     ...       ...     ...           ...             ...
4172    0   0.565     0.450   0.165        0.8870          0.3700
4173    2   0.590     0.440   0.135        0.9660          0.4390
4174    2   0.600     0.475   0.205        1.1760          0.5255
4175    0   0.625     0.485   0.150        1.0945          0.5310
4176    2   0.710     0.555   0.195        1.9485          0.9455

      Viscera weight  Shell weight
0             0.1010        0.1500
1             0.0485        0.0700
2             0.1415        0.2100
3             0.1140        0.1550
4             0.0395        0.0550
...              ...           ...
4172          0.2390        0.2490
4173          0.2145        0.2605
4174          0.2875        0.3080
4175          0.2610        0.2960
4176          0.3765        0.4950

[4177 rows x 8 columns]
```

```
y
```

```
0       16.5
1        8.5
2       10.5
3       11.5
4        8.5
        ...
4172    12.5
4173    11.5
4174    10.5
4175    11.5
4176    13.5
Name: Age, Length: 4177, dtype: float64
```

Scale the independent variables

```
from sklearn.preprocessing import scale
```

```
x_scaled=pd.DataFrame(scale(x),columns=x.columns)
x_scaled.head()
```

```
        Sex    Length  Diameter    Height  Whole weight  Shucked
weight  \
0  1.151980 -0.574558 -0.432149 -1.158093     -0.641898       -
0.607685
1  1.151980 -1.448986 -1.439929 -1.288751     -1.230277       -
1.170910
2 -1.280690  0.050033  0.122130 -0.112828     -0.309469       -
0.463500
3  1.151980 -0.699476 -0.432149 -0.374145     -0.637819       -
0.648238
4 -0.064355 -1.615544 -1.540707 -1.550067     -1.272086       -
1.215968

   Viscera weight  Shell weight
0       -0.726212     -0.638217
1       -1.205221     -1.212987
2       -0.356690     -0.207139
3       -0.607600     -0.602294
4       -1.287337     -1.320757
```

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_scaled,y,test_size=0.2,random_state=0)
```

```
X_train.shape
```

```
(3341, 8)
```

```
y_train.shape
```

```
(3341,)
```

```
X_test.shape
```

```
(836, 8)
```

```
y_test.shape
```

```
(836,)
```

Model building

```
from sklearn.linear_model import LinearRegression

model=LinearRegression()

model.fit(X_train,y_train)
```

```
LinearRegression()
```

Testing

```
pred_test=model.predict(X_test)
```

```
pred_test
```

```
array([14.53244534, 11.0078981 , 11.70625188,  7.02813239,
12.18242535,
       13.35584542,  9.28710687, 11.42212394, 10.12653741,
13.51180524,
       10.27844483,  7.98154103,  9.59514319, 10.44626088,
7.22370756,
       10.68720552,  9.24091016, 15.75581247, 12.64239931,
9.57397868,
        9.21366024,  8.56592733, 10.72833811,  8.97344174,
11.55892361,
       13.04994427,  6.25948905, 14.78324032, 12.25371328,
12.67496945,
        9.47615295,  6.19887287, 12.80393235, 14.62478044,
9.06495082,
       10.18031923, 10.66594652, 11.78933452, 10.18198705,
12.97094919,
       13.42820172, 10.72507517, 13.5003741 , 13.48852193,
13.96119858,
       10.96541652, 10.64272246, 13.17887707, 13.85346053,
9.69107776,
       12.98216095,  8.91133409, 10.40251492, 15.06229965,
10.60235007,
        9.05207143,  7.62747477,  8.75816059,  9.02348271,
8.76280894,
       11.08732622, 10.56218347, 11.79619028,  9.63045715,
9.82486158,
       13.78750076, 14.09087368, 13.99322601, 10.64057633,
15.83697549,
       11.2933687 , 20.47350024, 12.43234857, 11.59001976,
11.34988504,
       10.82440164, 11.15896341, 11.20527522, 12.69403663,
9.52758487,
       10.85907235,  7.67114484,  9.09760133, 13.93585886,
11.71409318,
        9.85798751, 11.43937101, 14.11714606,  6.51672826,
8.87371877,
       11.87465528, 12.12501048,  9.52022006,  4.00964382,
13.76887416,
        7.8371364 , 12.21012604,  9.17569268, 16.11145446,
10.98158594,
       11.34284257, 13.69497776, 11.32164788, 12.42320552,
7.15386018,
```

       11.888862 , 9.18449066, 8.53466049, 9.38646006,
14.97023344,
       10.5747346 , 12.5980386 , 12.81432213, 9.67064342,
15.60552497,
       11.6603456 , 13.27570247, 14.7189357 , 6.84372814,
11.60998243,
        8.86782202, 13.3099358 , 8.47605783, 11.09172386,
12.3761562 ,
       14.25380571, 12.08405342, 12.32891244, 9.57995662,
12.05635419,
       11.75046698, 9.42634719, 11.32432279, 12.7506026 ,
12.73823653,
       12.50471301, 11.83921499, 10.92067913, 9.0738368 ,
15.50115667,
       12.06204485, 13.03217064, 8.94525188, 9.67189824,
13.03917867,
       11.75642295, 10.97080939, 9.14406821, 9.75675431,
8.47569441,
       10.98926348, 18.92776394, 8.83827004, 12.32715664,
9.21560814,
        8.42775612, 13.13924886, 8.67259087, 14.86140354,
9.43480738,
       11.89566497, 8.62565569, 11.9546325 , 12.26800955,
6.72007654,
       14.63631512, 9.45543374, 8.25661349, 12.87098071,
11.49114466,
       10.48887513, 6.76977253, 9.80172366, 12.0937076 ,
14.16988145,
       11.72466997, 7.41598683, 10.31656596, 9.44969535,
13.1791082 ,
       11.3996152 , 12.04948826, 9.85994981, 10.7436796 ,
14.07361863,
       11.98125435, 10.40218538, 10.35634978, 8.34198772,
10.30304523,
       12.09259587, 11.31854318, 11.10907473, 11.73484321,
11.6942021 ,
       10.67334871, 9.75645553, 10.01288358, 6.70811431,
16.66985206,
       10.96322846, 12.17269595, 16.48579947, 11.28635242,
10.64728124,
       13.75748187, 7.7871626 , 13.82641821, 11.72114564,
13.89984085,
        7.36789009, 11.81986802, 12.39318903, 11.28190258,
13.06228982,
       12.143357 , 11.73621345, 12.51855138, 7.25569876,
14.03068684,
       13.54635237, 13.50386891, 9.99171305, 10.85760463,
13.31730598,
       15.71778649, 12.90307396, 10.8203456 , 13.48715389,
10.29961161,

```
       10.96002483, 10.00336493,  7.55924283, 12.8017881 ,
8.47091039,
       13.73403571,  9.64709623,  7.11196213,  9.92675198,
11.59328786,
       10.2751291 , 11.2107656 , 13.10280082, 11.22659975,
12.09112963,
       11.66904478, 13.04307667,  9.78935392, 12.70012771,
16.14996165,
       13.83052785,  9.30224518, 14.03595581,  9.76134506,
12.7805409 ,
       12.90428981,  9.14267407, 11.26826599, 12.00507241,
16.7313471 ,
       11.81696703, 10.74277241, 12.66214248,  8.68523556,
13.04725679,
       10.29424062, 10.33034869,  9.11815232,  9.64806126,
8.7985689 ,
       12.3271005 , 10.27733392,  9.88096756,  9.90607841,
8.05621571,
       12.28817417, 14.82613035, 10.53368251, 11.47606595,
11.44223583,
        8.97785412, 13.14634588,  9.35863532, 10.9952972 ,
11.42691368,
       13.09889687,  8.74544285, 12.08024064, 12.13045483,
14.47002341,
        9.20646477, 10.63893824,  7.47973256, 11.60957319,
9.08222752,
        9.31338168,  8.44986855,  7.90573357, 11.74114067,
16.7840848 ,
       10.41146936,  9.74343019, 14.06769462, 11.65339721,
10.53275913,
       10.53875286, 11.30995401, 11.52954947, 12.90110473,
10.6866098 ,
       10.80030225, 12.33074628,  9.88421546, 12.08292967,
18.68188504,
       13.48704046, 11.10041368,  8.35171557, 10.26829434,
11.9678599 ,
       12.36646872, 11.41422212, 11.07149653, 11.35340471,
13.93803813,
       10.99418782, 10.98822115, 12.1099718 , 17.51215828,
11.33130539,
       10.64923592, 12.95798921,  6.59506361, 12.88491434,
11.9613266 ,
       12.79203863,  8.74882679, 10.27351234, 12.65563604,
12.3092988 ,
       11.3923696 , 11.57184033, 11.9682194 , 12.89975525,
14.02164748,
        8.73043508,  7.60681346, 14.04892383, 14.72962318,
9.26562703,
        8.15863577, 10.39090267, 14.07176433, 11.65317097,
12.54070566,
```

```
       11.48555441, 12.23853577, 10.13462226,  9.91150534,
11.65965714,
       10.28485825, 11.54001118,  9.81709407, 10.63653115,
11.07401186,
        7.51402104,  8.16082003, 10.75265268,  9.36278012,
11.58984382,
       12.14925066,  8.99771109, 13.19704189,  9.7300532 ,
14.38018654,
       14.5990285 , 14.71724396, 10.07866002, 12.37471443,
6.59114425,
       10.40872379, 11.57766499, 11.3666394 , 14.4287674 ,
7.92578333,
       12.39753924, 17.001288  , 10.78155124,  9.55361564,
8.17748895,
       16.77915551, 11.55965325, 12.61125594, 11.40279445,
19.03085946,
        9.1974374 , 12.56834828, 10.86339483, 12.68562536,
12.13741854,
       10.20388759, 10.01405387,  8.36924958, 14.8441481 ,
9.90015183,
       11.44262366,  8.02316032, 13.18493306, 17.44919756,
11.27941733,
       11.72269422,  9.26902521, 11.75876981, 12.49828672,
15.89949294,
       12.81194268, 15.23265741, 13.12646364, 16.20431199,
11.54229386,
       10.52055193, 11.05148003, 12.27553724, 12.84343187,
7.73479626,
       10.93879109, 12.05629185, 13.03774149, 12.91363689,
15.70157825,
       11.29056019, 12.30560738, 10.36123108, 14.42541568,
10.68101781,
       10.80400139, 13.47263131, 11.07707206, 11.23080603,
12.31402767,
       10.01522273,  8.53743557, 14.7621652 , 10.13512612,
8.36290541,
        8.98170785, 15.17510849,  7.28852514, 12.19656398,
11.90511798,
        6.17321917,  8.62385721, 11.04741416,  8.62460938, -
1.47297544,
        9.95419162, 12.89211251, 11.24023027, 10.54271974,
13.55685826,
        8.90822473, 16.78956387,  9.93937733, 11.79317417,
11.1794635 ,
       13.6266044 ,  8.42754654, 21.36688423, 11.97210624,
10.29763889,
        9.26711611, 12.33505259,  9.37129839, 16.5749059 ,
10.44211327,
       10.67922834,  9.80101147,  9.71581467,  9.10780792,
12.38216486,
```

       11.90848986, 13.60040865, 10.40779974, 13.31542559,
15.39122244,
       13.60006414, 11.67899133, 11.36281632, 9.75447768,
12.31767215,
        6.49410505, 10.71006135, 14.06874949, 12.16349645,
8.78905515,
        9.12525815, 9.7320658 , 8.02209857, 12.90627833,
10.91071125,
        9.04966193, 13.97216624, 8.70568136, 13.59272751,
11.98728331,
       12.13772726, 13.10949811, 15.88972076, 9.67042181,
7.40711774,
       11.6273047 , 10.37835044, 16.21089949, 16.59849124,
11.11386189,
       10.03576924, 12.18992544, 11.60586954, 15.16949992,
11.9208423 ,
       12.29780979, 11.53540376, 12.10458326, 11.22214465,
9.03256482,
       14.61093967, 10.5722459 , 11.40050064, 10.38354932,
10.35606742,
        9.57959998, 9.36621381, 10.06218198, 11.85878282,
6.6759175 ,
       11.41453866, 13.78165367, 13.77621232, 9.33431658,
15.2380223 ,
        9.4378123 , 11.5315538 , 9.34981915, 14.07801547,
8.64742722,
       11.90734109, 9.98023159, 10.05349931, 12.29744499,
11.15067085,
       12.46772396, 15.31130349, 9.99155172, 16.25178226,
12.37445419,
        9.50924737, 9.9242075 , 10.93733949, 11.85357544,
11.72610291,
       13.4268923 , 10.94792578, 13.23018349, 10.42403757,
12.27863955,
        7.64552142, 9.98743011, 9.45625093, 11.63488557,
9.68133387,
       12.20397764, 9.9835451 , 10.13970686, 11.07580134,
10.5634322 ,
        9.74141341, 12.83625607, 11.50020725, 12.82380982,
6.34536263,
       14.44763715, 8.03398937, 7.21138956, 8.17068277,
9.79484807,
       10.61117355, 22.26035682, 11.76325478, 10.6032389 ,
13.98329507,
        7.57216955, 11.49228434, 11.35798073, 9.69807617,
13.77320211,
       11.87456673, 6.08722484, 10.13070558, 8.13271563,
11.55426638,
       10.53393164, 14.55693877, 11.64934356, 10.76502919,
13.64233197,

       11.48446041, 11.20665011, 8.70454581, 11.21603585,
13.41524187,
        9.15435852, 14.67089754, 11.17586771, 10.75017826,
12.4443991 ,
       14.90198919, 10.64477219, 9.46950572, 15.56130213,
11.98546967,
       11.55707512, 13.65822981, 9.00227742, 11.51153315,
12.64959113,
       13.14413401, 10.48760552, 14.77196813, 10.71500399,
11.95082094,
       11.86600702, 8.89458951, 9.82765902, 10.21972444,
9.83418416,
       11.59508749, 6.96283107, 16.52033076, 10.87007468,
11.67482471,
       11.78728465, 11.33986386, 12.93762116, 10.91346786,
11.15820943,
       11.17447772, 9.13333596, 11.39091376, 14.34494209,
12.81082189,
        9.74110154, 9.9875444 , 9.96278186, 10.92584332,
8.64925917,
       14.40628057, 9.3513612 , 10.21036434, 12.18792665,
14.7724984 ,
       12.95491527, 9.64140307, 14.68273119, 13.05481542,
13.53525668,
       10.28743488, 10.93367104, 7.60718771, 8.40076924,
12.96969653,
       11.19538403, 13.15981986, 16.86515026, 13.25325946,
7.36017467,
        9.44373172, 10.36749135, 8.01250773, 8.83867386,
17.44949175,
       15.26822303, 11.03949477, 12.27387613, 11.73810955,
10.72377222,
        9.38165472, 13.10917813, 12.76709793, 12.37796384,
18.37825831,
       10.86495486, 13.20101148, 13.30476443, 15.26466599,
9.5735314 ,
       11.75595797, 16.60509027, 14.45096879, 9.47512897,
9.69160693,
        8.54935647, 12.39902329, 10.72217414, 14.73829784,
10.79303989,
       12.99534963, 11.42641917, 10.11843876, 14.74614287,
13.87854227,
        7.6911517 , 6.97546365, 7.58372621, 12.34110329,
8.81401579,
       10.91473141, 12.85890858, 10.1137732 , 9.10523083,
10.05964618,
       15.02621266, 6.9446349 , 10.86458695, 8.66879173,
14.97963642,
        7.52614906, 9.96335965, 9.70658273, 13.70340806,
12.67256336,

```
        9.75930209, 12.87901199, 13.36454163, 14.33288441,
10.5833593 ,
       12.24976985,  8.95261265,  9.51815416, 10.40738817,
7.73876283,
       13.4790368 , 10.09713362,  8.73631873,  8.46152289,
12.06845303,
        9.11647045, 10.89287842, 10.09381971, 10.48160872,
10.66557638,
       11.41005482, 11.17882154,  9.27793561, 12.66536743,
9.73414768,
       13.38319446, 11.37257725,  8.93722555, 10.76511186,
15.70881856,
        8.6278767 ,  8.39895778,  9.36077029,  7.81331226,
12.81419264,
       10.87514152, 11.69582811,  8.27216684, 12.95952949,
9.6629573 ,
        8.76111272, 10.71023335, 15.21989454, 11.86739891,
10.51573331,
       12.22576065,  8.86097055, 15.16108111,  9.21950581,
11.67870977,
       16.22547646, 14.17701226,  9.39828153,  8.9573144 ,
10.50793761,
       10.2000563 , 10.93607193, 10.22054406, 11.08046947,
12.54039172,
       11.31575611, 11.99842664, 15.16688865, 12.95581089,
14.63051012,
       10.49806683, 13.04845619, 11.37493048,  6.22816992,
12.4270097 ,
       11.19134436,  9.86689109, 10.09837641,  7.78497896,
8.4308489 ,
       10.09392535, 10.7071512 , 12.7289974 ,  9.55080414,
10.06964749,
       10.68784557,  9.04165464, 11.47395606, 14.02838581,
12.40239912,
       10.92102594, 10.91191076, 13.56439785, 14.11253667,
11.73321252,
        7.57107634, 12.24330433, 12.4819393 ,  8.32934602,
9.8410542 ,
       10.37420246, 13.04819633, 11.66093331,  8.28699911,
8.6437147 ,
        9.94700644, 11.6921597 , 10.31474559,  9.96224932,
13.89646102,
        7.20203685])

pred_train = model.predict(X_train)
pred_train

array([ 7.19974119,  6.84237887, 15.18412216, ..., 10.94694691,
       13.11797394,  9.98079993])
```

```
Age= pd.DataFrame({'Actual_Age':y_test,'pred_Age':pred_test})
Age
```

```
      Actual_Age   pred_Age
668         14.5  14.532445
1580         9.5  11.007898
3784        12.5  11.706252
463          6.5   7.028132
2615        13.5  12.182425
...          ...        ...
575         12.5  11.692160
3231        13.5  10.314746
1084         8.5   9.962249
290         18.5  13.896461
2713         5.5   7.202037

[836 rows x 2 columns]
```

Performance using Metrics

```
from sklearn import metrics

# r2score

print(metrics.r2_score(y_test,pred_test)) # test accuracy
```

```
0.5418733297497837
```

```
print(metrics.r2_score(y_train,pred_train)) # train accuracy
```

```
0.531977913488948
```

```
#MSE

print(metrics.mean_squared_error(y_test,pred_test))   # test accuracy
```

```
4.975398170943831
```

```
print(metrics.mean_squared_error(y_train,pred_train))   # train
accuracy
```

```
4.807868180891602
```