```
import pandas as pd
import numpy as np
from matplotlib import pyplot as plt
import seaborn as sns
from sklearn.linear_model import LinearRegression
from google.colab import drive
from sklearn.preprocessing import LabelEncoder
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import r2_score
```

DATASET LOADED

```
drive.mount('/content/drive')
```

```
Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.mou
```

```
path='/content/drive/MyDrive/Colab Notebooks/IBM Project/abalone.csv'
```

+ Code + Text

```
df=pd.read_csv(path)
```

```
df.head()
```

1 to 5 of 5 entries Filter

index	Sex	Length	Diameter	Height	Whole weight	Shucked weight	Viscera weight	Shell weight	R
0	M	0.455	0.365	0.095	0.514	0.2245	0.101	0.15	
1	M	0.35	0.265	0.09	0.2255	0.0995	0.0485	0.07	
2	F	0.53	0.42	0.135	0.677	0.2565	0.1415	0.21	
3	M	0.44	0.365	0.125	0.516	0.2155	0.114	0.155	
4	I	0.33	0.255	0.08	0.205	0.0895	0.0395	0.055	

Show 25 per page

```
df.tail()
```

1 to 5 of 5 entries Filter

index	Sex	Length	Diameter	Height	Whole weight	Shucked weight	Viscera weight	Shell weight	a
4172	F	0.565	0.45	0.165	0.887	0.37	0.239	0.249	1

```
df.describe()
```

1 to 8 of 8 entries Filter

index	Length	Diameter	Height	Whole weight	S
count	4177.0	4177.0	4177.0	4177.0	
mean	0.5239920995930094	0.40788125448886764	0.13951639932966242	0.8287421594445774	0.35
std	0.12009291256479956	0.09923986613365945	0.041827056607257274	0.4903890182309977	0.22
min	0.075	0.055	0.0	0.002	
25%	0.45	0.35	0.115	0.4415	
50%	0.545	0.425	0.14	0.7995	
75%	0.615	0.48	0.165	1.153	
max	0.815	0.65	1.13	2.8255	

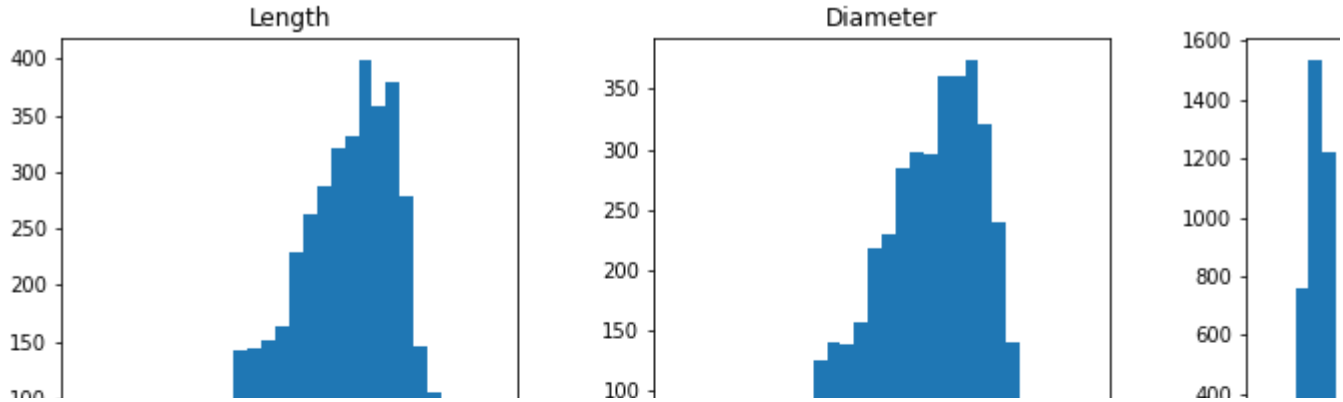
Show 25 per page

```
df['age'] = df['Rings']+1.5
df = df.drop('Rings', axis = 1)
```

Univariate Analysis

```
df.hist(figsize=(20,10), grid=False, layout=(2, 4), bins = 30)
```

```
array([[<matplotlib.axes._subplots.AxesSubplot object at 0x7f50c63ffc90>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7f50c77a4e50>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7f50c6311390>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7f50c62c8990>],
       [<matplotlib.axes._subplots.AxesSubplot object at 0x7f50c627ff90>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7f50c62405d0>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7f50c61f8c50>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7f50c61be1d0>]],
      dtype=object)
```



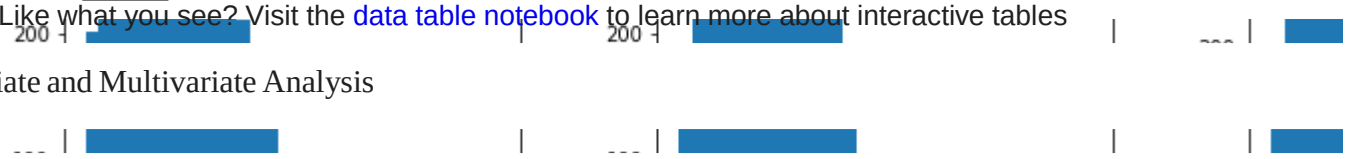

```
df.groupby('Sex')[['Length', 'Diameter', 'Height', 'Whole weight', 'Shucked weight',
       'Viscera weight', 'Shell weight', 'age']].mean().sort_values('age')
```

1 to 3 of 3 entries Filter

Sex	Length	Diameter	Height	Whole weight	Shu
I	0.42774590163934423	0.3264940387481371	0.10799552906110284	0.43136251862891206	0.19103
M	0.5613907068062827	0.4392866492146597	0.15138089005235603	0.9914594240837696	0.43294
F	0.5790933435348126	0.4547322111706198	0.15801071155317523	1.0465321346595258	0.44618

Show 25 per page

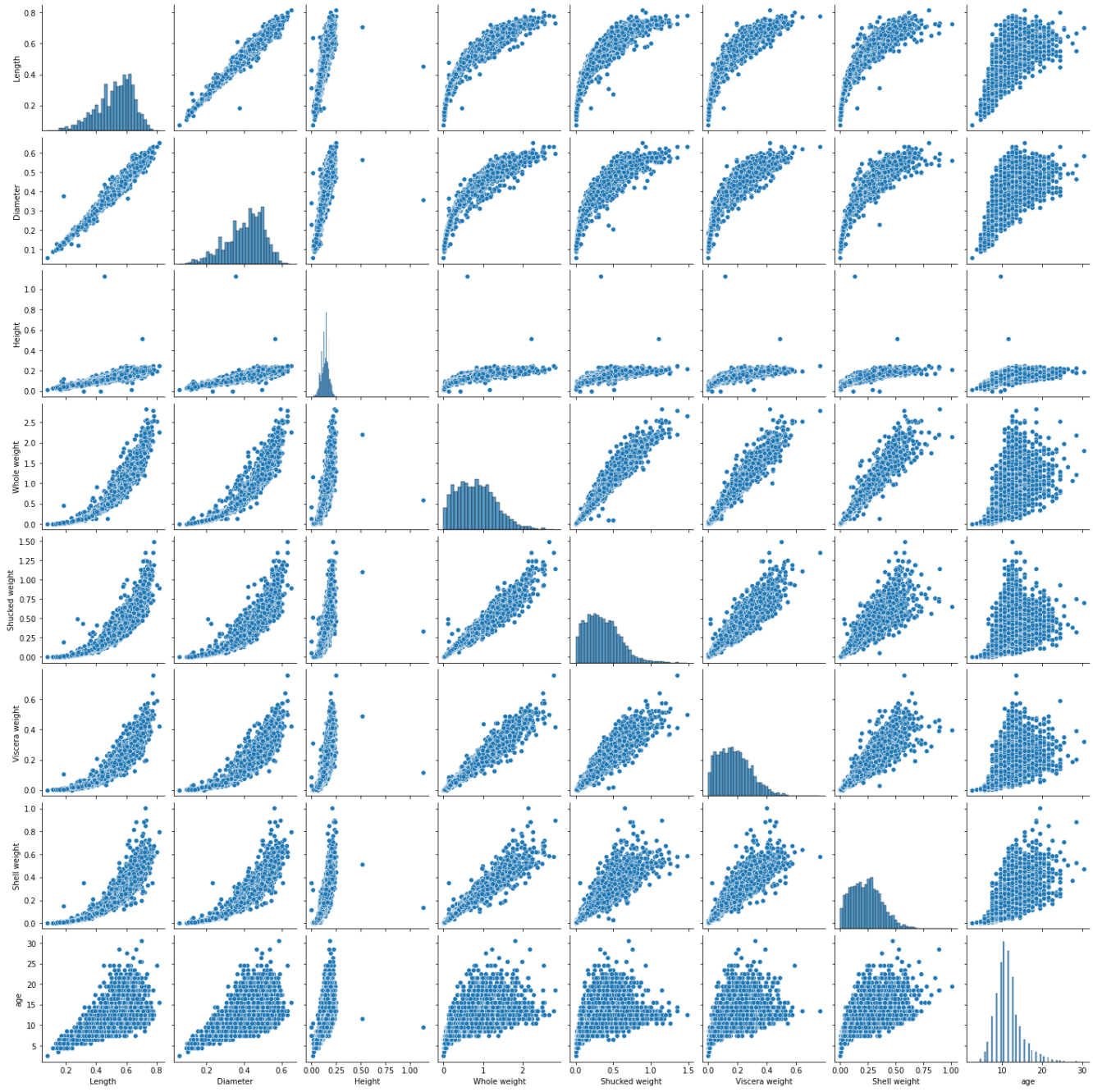
Like what you see? Visit the data table notebook to learn more about interactive tables



Bivariate and Multivariate Analysis


```
numerical_features = df.select_dtypes(include = [np.number]).columns
sns.pairplot(df[numerical_features])
```

```
<seaborn.axisgrid.PairGrid at 0x7f50c5cfee50>
```



Descriptive Statistics

```
df.describe()
```

1 to 8 of 8 entries Filter

index	Length	Diameter	Height	Whole weight	S
count	4177.0	4177.0	4177.0	4177.0	
mean	0.5239920995930094	0.40788125448886764	0.13951639932966242	0.8287421594445774	0.35
std	0.12009291256479956	0.09923986613365945	0.041827056607257274	0.4903890182309977	0.22
min	0.075	0.055	0.0	0.002	
25%	0.45	0.35	0.115	0.4415	
50%	0.545	0.425	0.14	0.7995	
75%	0.615	0.48	0.165	1.153	
max	0.815	0.65	1.13	2.8255	

Show 25 per page

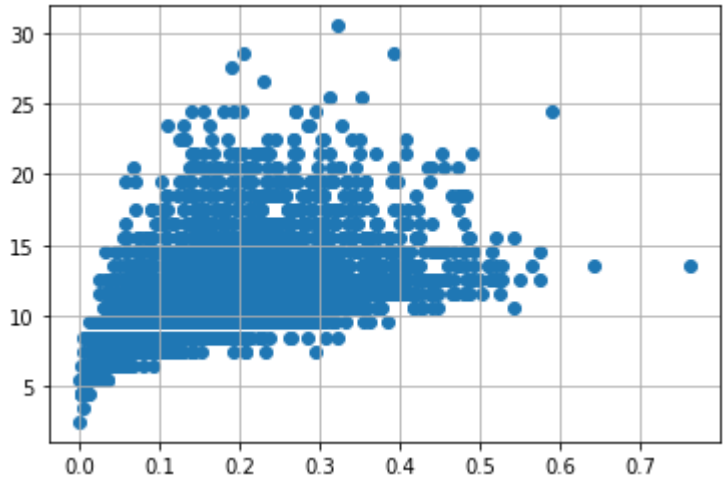
Check for missing values

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

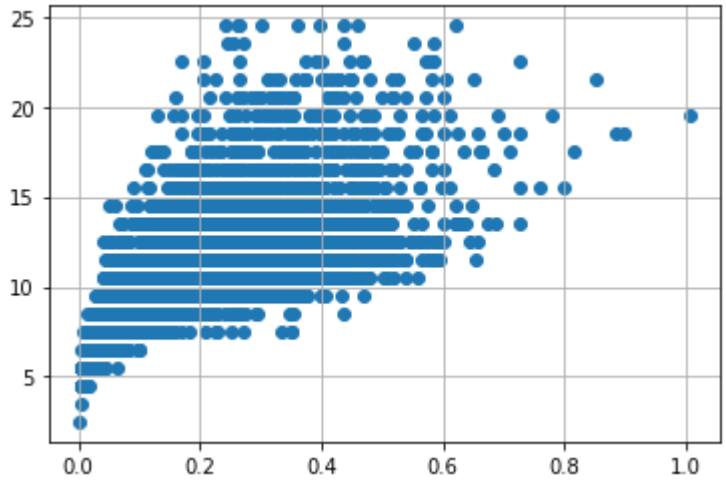
```
Sex               0
Length            0
Diameter          0
Height            0
Whole weight      0
Shucked weight    0
Viscera weight    0
Shell weight      0
age               0
dtype: int64
```

Outlier Handling

```
df = pd.get_dummies(df)
dummy_data = df.copy()


#outliers removal for viscera weight


var = 'Viscera weight'
plt.scatter(x = df[var], y = df['age'],)
plt.grid(True)
df.drop(df[(df['Viscera weight']> 0.5) & (df['age'] < 20)].index, inplace=True)
df.drop(df[(df['Viscera weight']<0.5) & (df['age'] > 25)].index, inplace=True)
```

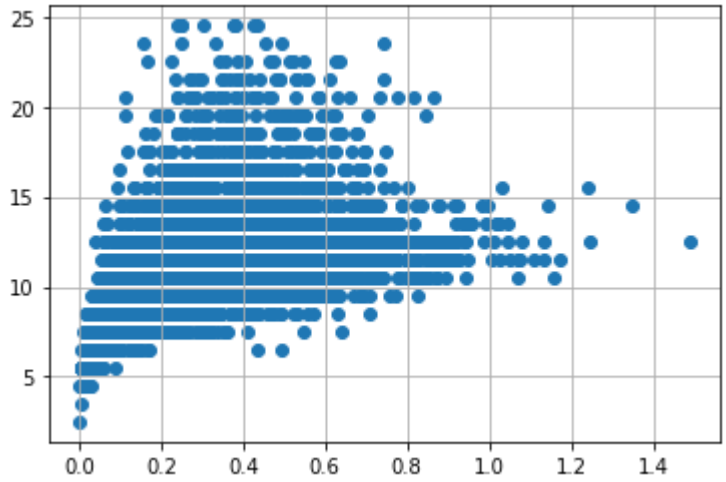



```
#outliers removal for shell weight


var = 'Shell weight'
plt.scatter(x = df[var], y = df['age'],)
plt.grid(True)
df.drop(df[(df['Shell weight']> 0.6) & (df['age'] < 25)].index, inplace=True)
df.drop(df[(df['Shell weight']<0.8) & (df['age'] > 25)].index, inplace=True)
```

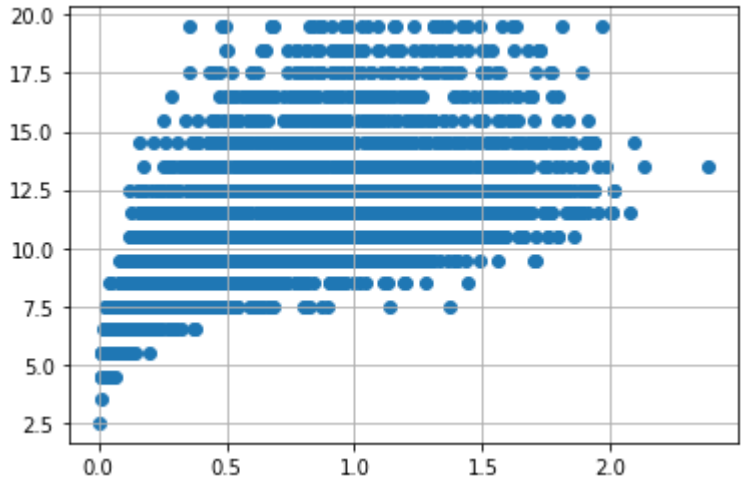



```
#Outliers removal for shuked weight
```

```
var = 'Shucked weight'
plt.scatter(x = df[var], y = df['age'],)
plt.grid(True)
df.drop(df[(df['Shucked weight']>= 1) & (df['age'] < 20)].index, inplace=True)
df.drop(df[(df['Shucked weight']<1) & (df['age'] > 20)].index, inplace=True)
```

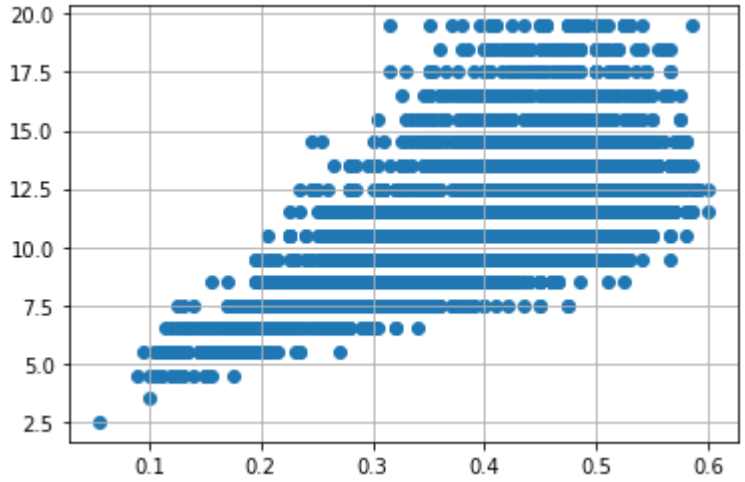



```
#outliers removal for whole weight


var = 'Whole weight'
plt.scatter(x = df[var], y = df['age'])
plt.grid(True)
df.drop(df[(df['Whole weight'] >= 2.5) &(df['age'] < 25)].index, inplace = True)
df.drop(df[(df['Whole weight']<2.5) & (df['age'] > 25)].index, inplace = True)
```

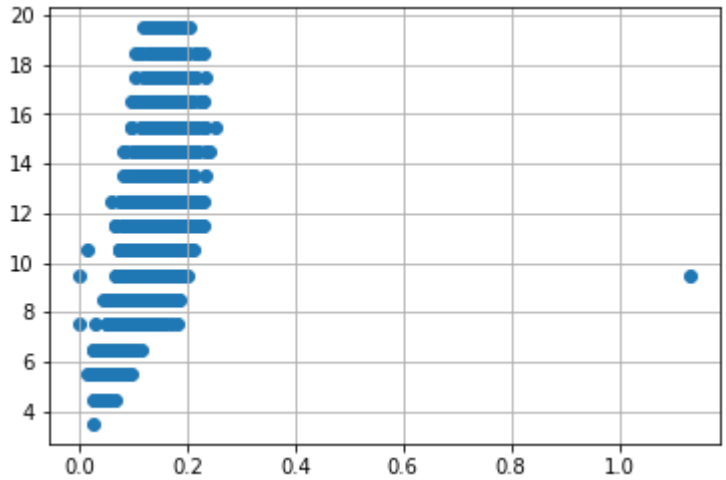



```
#outliers removal for diameters


var = 'Diameter'
plt.scatter(x = df[var], y = df['age'])
plt.grid(True)
```

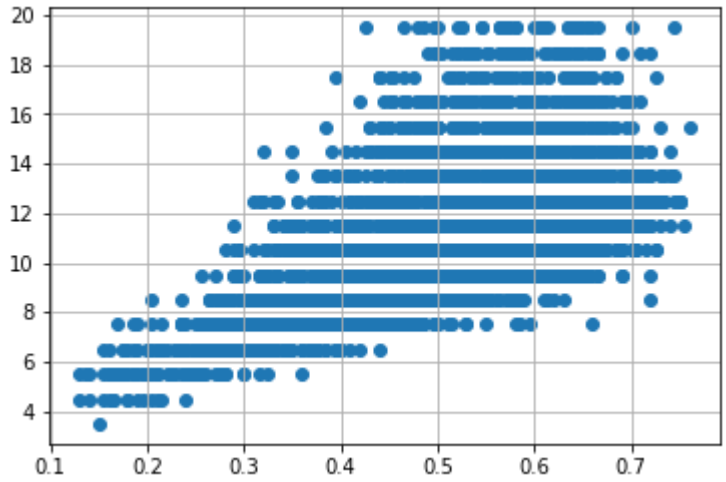
```
df.drop(df[(df['Diameter'] <0.1) &(df['age'] < 5)].index, inplace = True)
df.drop(df[(df['Diameter']<0.6) & (df['age'] > 25)].index, inplace = True)
df.drop(df[(df['Diameter']>=0.6) & (df['age'] < 25)].index, inplace = True)
```




```
#outliers removal for height


var = 'Height'
plt.scatter(x = df[var], y = df['age'])
plt.grid(True)
df.drop(df[(df['Height'] > 0.4) &(df['age'] < 15)].index, inplace = True)
df.drop(df[(df['Height']<0.4) & (df['age'] > 25)].index, inplace = True)
```




```
#outliers removal for length


var = 'Length'
plt.scatter(x = df[var], y = df['age'])
plt.grid(True)
df.drop(df[(df['Length'] <0.1) &(df['age'] < 5)].index, inplace = True)
df.drop(df[(df['Length']<0.8) & (df['age'] > 25)].index, inplace = True)
df.drop(df[(df['Length']>=0.8) & (df['age'] < 25)].index, inplace = True)
```



Categorical Columns

```
numerical_features = df.select_dtypes(include = [np.number]).columns
categorical_features = df.select_dtypes(include = [np.object]).columns
```

```
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:2: DeprecationWarning: `np
Deprecated in NumPy 1.20; for more details and guidance: https://numpy.org/devdocs/rele
```

```
numerical_features
```

```
Index(['Length', 'Diameter', 'Height', 'Whole weight', 'Shucked weight',
       'Viscera weight', 'Shell weight', 'age', 'Sex_F', 'Sex_I', 'Sex_M'],
      dtype='object')
```

```
categorical_features
```

```
Index([], dtype='object')
```

Split the dependent and independent variables

```
x=df.iloc[:,:5]
y=df.iloc[:,5:]
```

```
x
```

1 to 25 of 3995 entries Filter

index	Length	Diameter	Height	Whole weight	Shucked weight
0	0.455	0.365	0.095	0.514	0.2245
1	0.35	0.265	0.09	0.2255	0.0995
2	0.53	0.42	0.135	0.677	0.2565
3	0.44	0.365	0.125	0.516	0.2155
4	0.33	0.255	0.08	0.205	0.0895
5	0.425	0.3	0.095	0.3515	0.141
7	0.545	0.425	0.125	0.768	0.294
8	0.475	0.37	0.125	0.5095	0.2165
10	0.525	0.38	0.14	0.6065	0.194
11	0.43	0.35	0.11	0.406	0.1675
12	0.49	0.38	0.135	0.5415	0.2175
13	0.535	0.405	0.145	0.6845	0.2725
14	0.47	0.355	0.1	0.4755	0.1675
15	0.5	0.4	0.13	0.6645	0.258
16	0.355	0.28	0.085	0.2905	0.095
17	0.44	0.34	0.1	0.451	0.188
18	0.365	0.295	0.08	0.2555	0.097
19	0.45	0.32	0.1	0.381	0.1705
20	0.355	0.28	0.095	0.2455	0.0955
21	0.38	0.275	0.1	0.2255	0.08
22	0.565	0.44	0.155	0.9395	0.4275
23	0.55	0.415	0.135	0.7635	0.318
24	0.615	0.48	0.165	1.1615	0.513

y

1 to 25 of 3995 entries Filter

index	Viscera weight	Shell weight	age	Sex_F	Sex_I	Sex_M
0	0.101	0.15	16.5	0	0	1
1	0.0485	0.07	8.5	0	0	1
2	0.1415	0.21	10.5	1	0	0
3	0.114	0.155	11.5	0	0	1
4	0.0395	0.055	8.5	0	1	0
5	0.0775	0.12	9.5	0	1	0
7	0.1495	0.26	17.5	1	0	0
8	0.1125	0.165	10.5	0	0	1
10	0.1475	0.21	15.5	1	0	0
11	0.081	0.135	11.5	0	0	1
12	0.095	0.19	12.5	0	0	1

split the data (train and test)

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

17	0.087	0.13	11.5	1	0	0

Model Building

19	0.075	0.115	10.5	0	0	1

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

```
LinearRegression()
```

25	0.188	0.3	12.5	1	0	0

Train the model

Show 25 per page | 1 | 2 | 10 | 100 | 150 | 160

```
x_train[0:4]
```

1 to 4 of 4 entries Filter

index	Length	Diameter	Height	Whole weight	Shucked weight
2423	0.41	0.315	0.11	0.321	0.1255
1216	0.31	0.225	0.07	0.1055	0.435
3002	0.645	0.505	0.185	1.463	0.592
985	0.57	0.45	0.155	1.1935	0.513

Show 25 per page

Like what you see? Visit the data table notebook to learn more about interactive tables.

```
y_train[0:5]
```

1 to 5 of 5 entries Filter

index	Viscera weight	Shell weight	age	Sex_F	Sex_I	Sex_M
2423	0.0655	0.095	11.5	1	0	0
1216	0.015	0.04	6.5	0	1	0
3002	0.3905	0.416	11.5	0	0	1
985	0.21	0.343	11.5	0	0	1
2838	0.233	0.2595	10.5	0	0	1

```
x_test[0:4]
```

1 to 4 of 4 entries Filter

index	Length	Diameter	Height	Whole weight	Shucked weight
3006	0.7	0.545	0.185	1.6135	0.75
3817	0.475	0.385	0.12	0.562	0.289
4094	0.63	0.53	0.175	1.4135	0.667
402	0.435	0.325	0.11	0.4335	0.178

Show 25 per page
Like what you see? Visit the data table notebook to learn more about interactive tables.

```
y_test[0:5]
```

1 to 5 of 5 entries Filter

index	Viscera weight	Shell weight	age	Sex_F	Sex_I	Sex_M
3006	0.4035	0.3685	12.5	0	0	1
3817	0.0905	0.153	9.5	0	0	1
4094	0.2945	0.3555	14.5	0	0	1
402	0.0985	0.155	8.5	1	0	0
1396	0.2385	0.345	12.5	0	0	1

Show 25 per page
Like what you see? Visit the data table notebook to learn more about interactive tables.

```
ss=StandardScaler()
x_train=ss.fit_transform(x_train)
```

```
lrpred=lr.predict(x_test[0:9])
```

```
lrpred
```

```
array([[ 0.35064154,  0.42317517, 12.55339604,  0.50780283, -0.08545215,
         0.57764932],
       [ 0.11701718,  0.15625023,  9.84878154,  0.23508899,  0.45415266,
         0.31075835],
       [ 0.30007654,  0.37892926, 12.30238534,  0.50574715, -0.05317174,
         0.54742459],
       [ 0.09692013,  0.13181165,  9.95964476,  0.18232777,  0.5578356 ,
```

```
  0.25983664],
[ 0.25590426,  0.32122087, 11.92694455,  0.41939293,  0.12392858,
  0.45667849],
[ 0.15846252,  0.20923024, 11.29126176,  0.29014005,  0.36997235,
  0.33988761],
[ 0.28730637,  0.35538064, 12.37098073,  0.43130339,  0.09697514,
  0.47172147],
[ 0.15229535,  0.20263728, 10.84591436,  0.29722028,  0.34107547,
  0.36170425],
[ 0.05210596,  0.07789379,  9.1755676 ,  0.12539739,  0.65136117,
  0.22324144]])
```

Measure the performance using Metrics

```
r2_score(lr.predict(x_test),y_test)
```

```
-3.1758408437233587
```