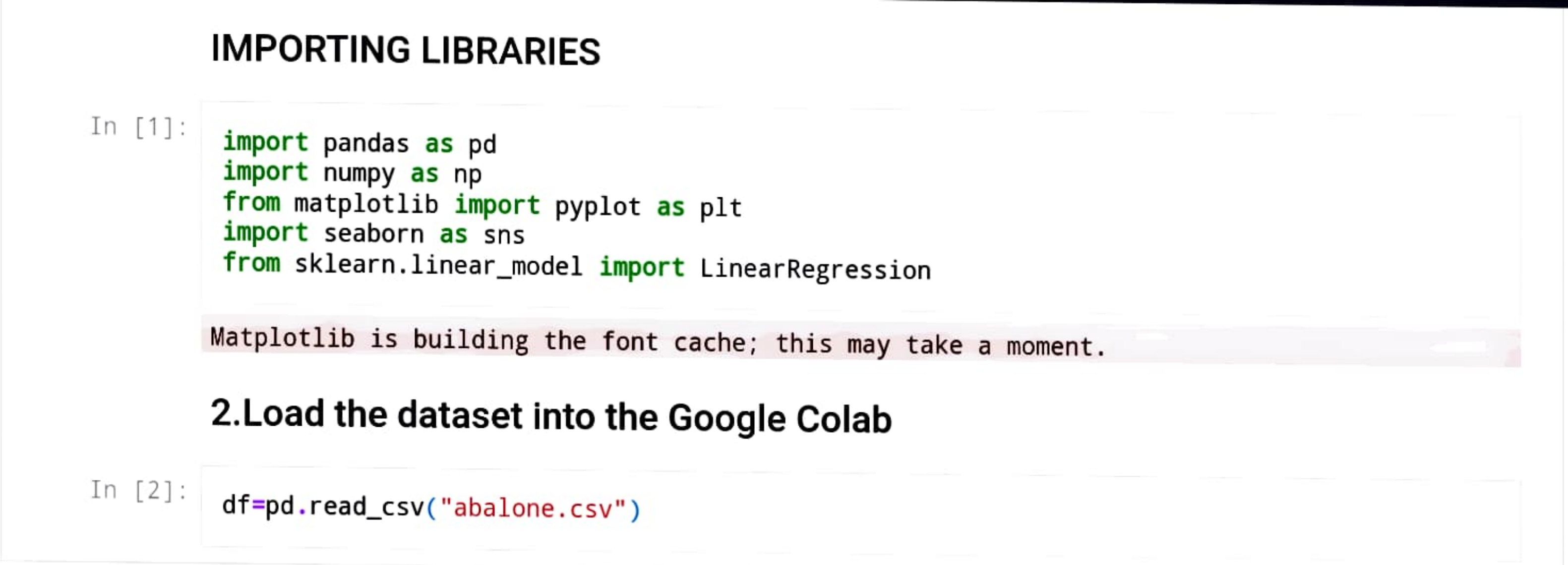


IMPORTING LIBRARIES

In [1]:

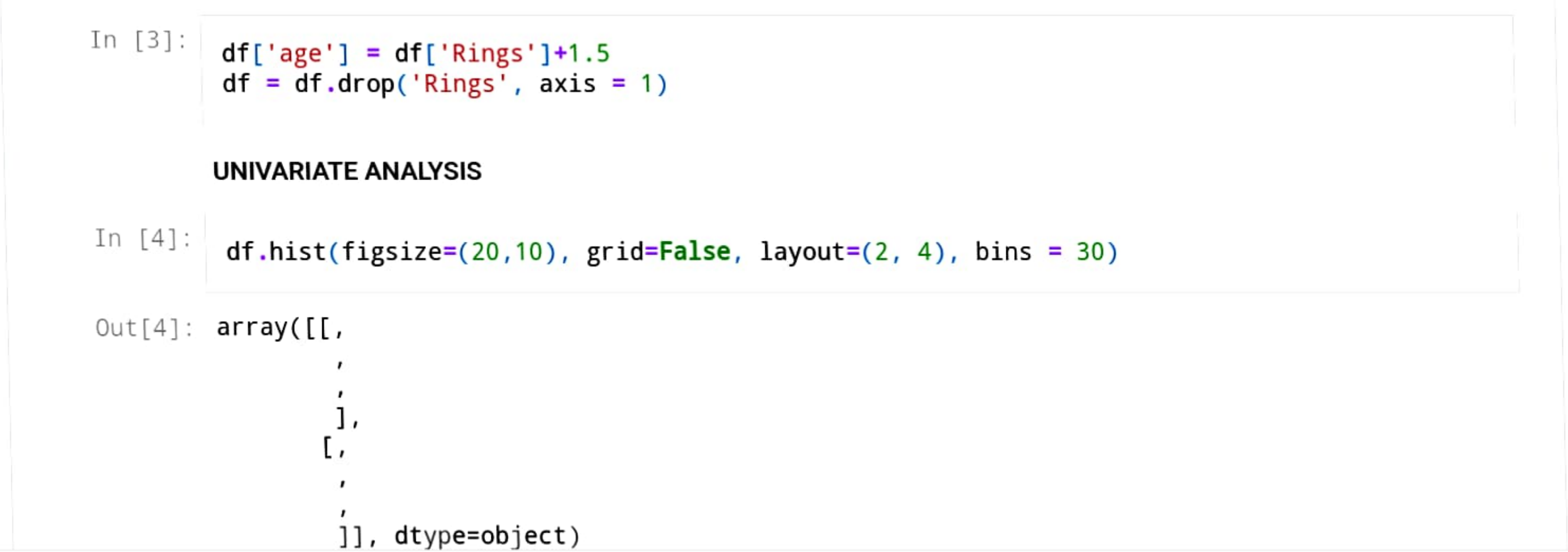
```
import pandas as pd
import numpy as np
from matplotlib import pyplot as plt
import seaborn as sns
from sklearn.linear_model import LinearRegression
```

```
Matplotlib is building the font cache; this may take a moment.
```

2.Load the dataset into the Google Colab

In [2]:

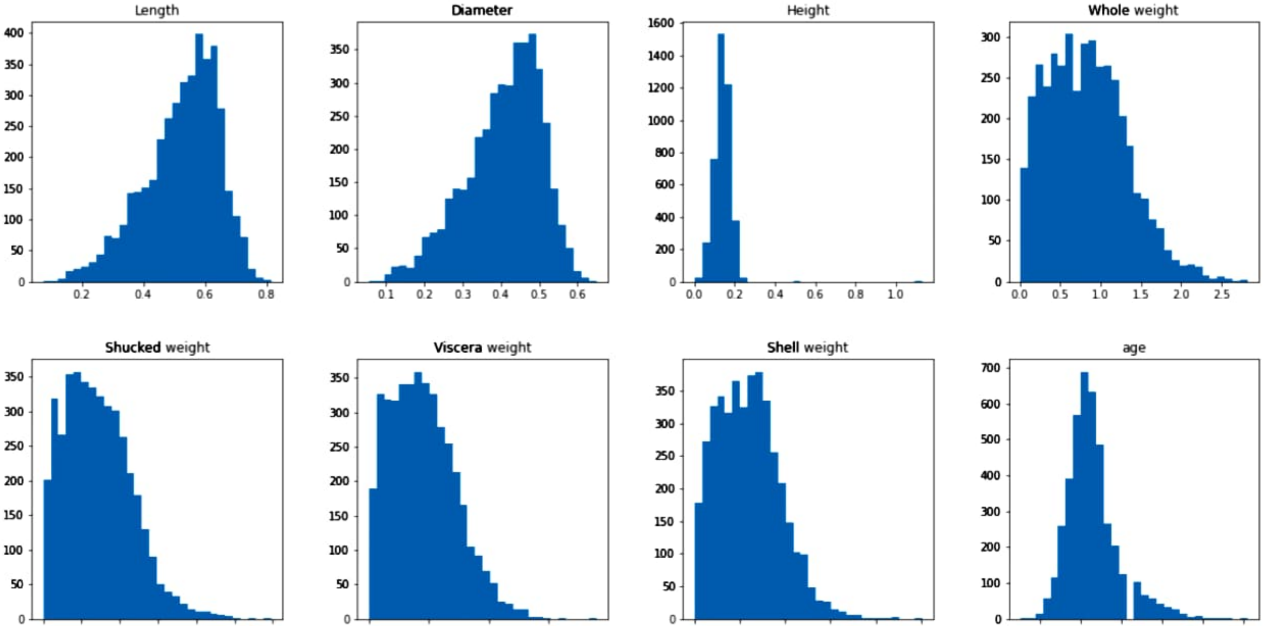
```
df=pd.read_csv("abalone.csv")
```



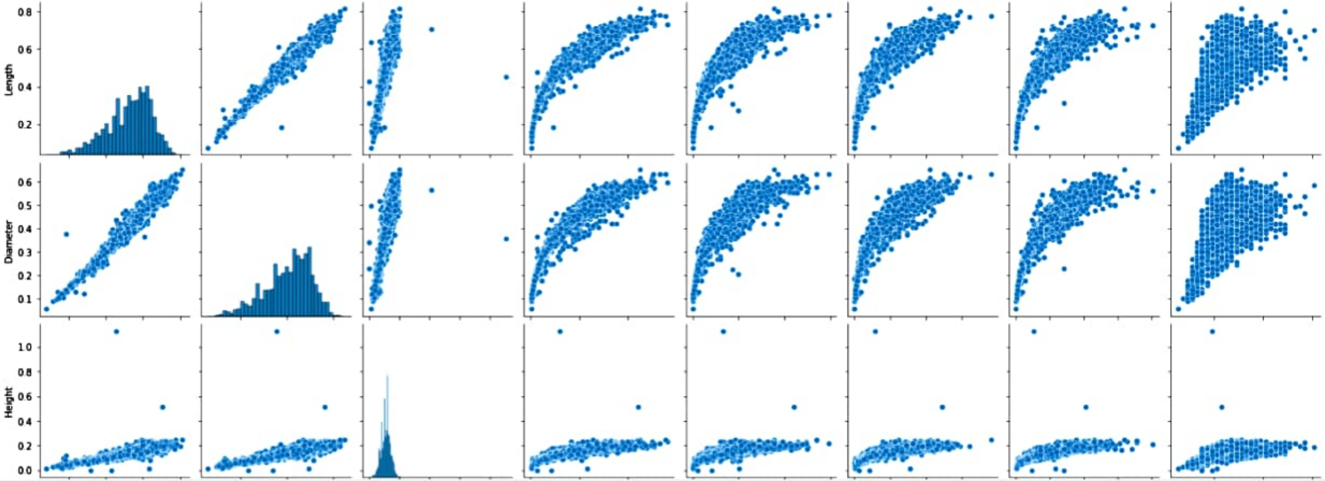
```
In [3]: df['age'] = df['Rings']+1.5
        df = df.drop('Rings', axis = 1)
```

UNIVARIATE ANALYSIS

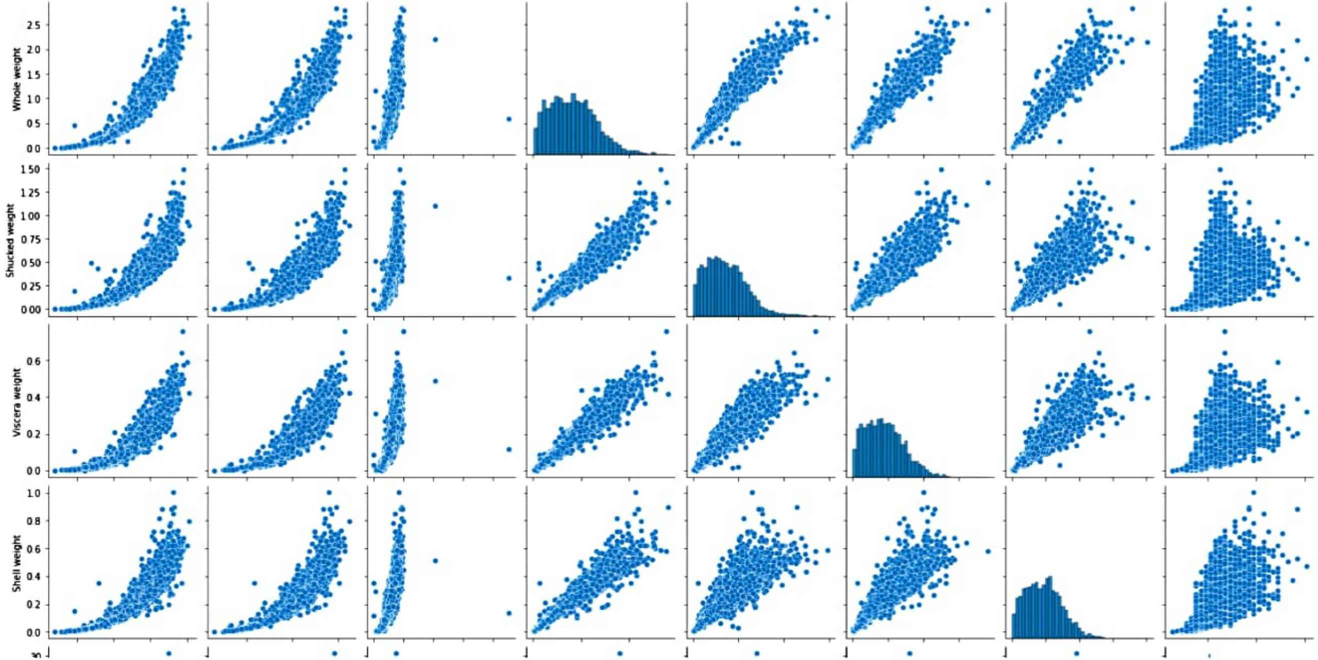
```
In [4]: df.hist(figsize=(20,10), grid=False, layout=(2, 4), bins = 30)

Out[4]: array([[,
                ,
                ,
                ],
               [,
                ,
                ,
                ]], dtype=object)
```


Length
0
50
100
150
200
250
300
350
400
0.2
0.4
0.6
0.8
Diameter
0
50
100
150
200
250
300
350
0.1
0.2
0.3
0.4
0.5
0.6
Height
0
200
400
600
800
1000
1200
1400
1600
0.0
0.2
0.4
0.6
0.8
1.0
Whole weight
0
50
100
150
200
250
300
0.0
0.5
1.0
1.5
2.0
2.5
Shucked weight
0
50
100
150
200
250
300
350
Viscera weight
0
50
100
150
200
250
300
350
Shell weight
0
50
100
150
200
250
300
350
age
0
100
200
300
400
500
600
700

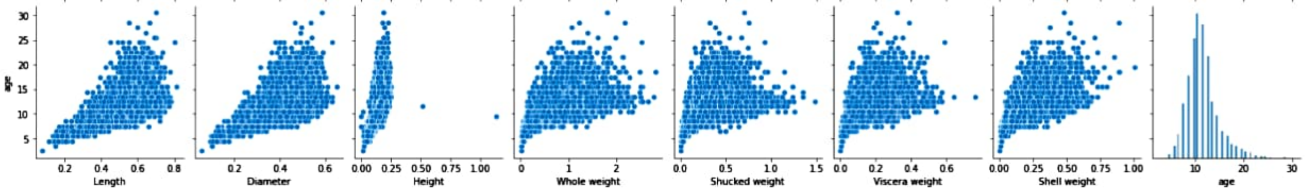
```
Out[6]:
```



Length
0.8
0.6
0.4
0.2
Diameter
0.6
0.5
0.4
0.3
0.2
0.1
Height
1.0
0.8
0.6
0.4
0.2
0.0



Whole weight
2.5
2.0
1.5
1.0
0.5
0.0
Shucked weight
1.50
1.25
1.00
0.75
0.50
0.25
0.00
Viscera weight
0.6
0.4
0.2
0.0
Shell weight
1.0
0.8
0.6
0.4
0.2
0.0
30



Descriptive statistics

```
In [7]: df.describe()
```

```
Out[7]:
            Length     Diameter       Height        Whole      Shucked      Viscera  Shell weight          age
                                                   weight       weight       weight
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    11.433684
  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     2.500000
```

25%	0.450000	0.350000	0.115000	0.441500	0.186000	0.093500	0.130000	9.500000
50%	0.545000	0.425000	0.140000	0.799500	0.336000	0.171000	0.234000	10.500000
75%	0.615000	0.480000	0.165000	1.153000	0.502000	0.253000	0.329000	12.500000
max	0.815000	0.650000	1.130000	2.825500	1.488000	0.760000	1.005000	30.500000

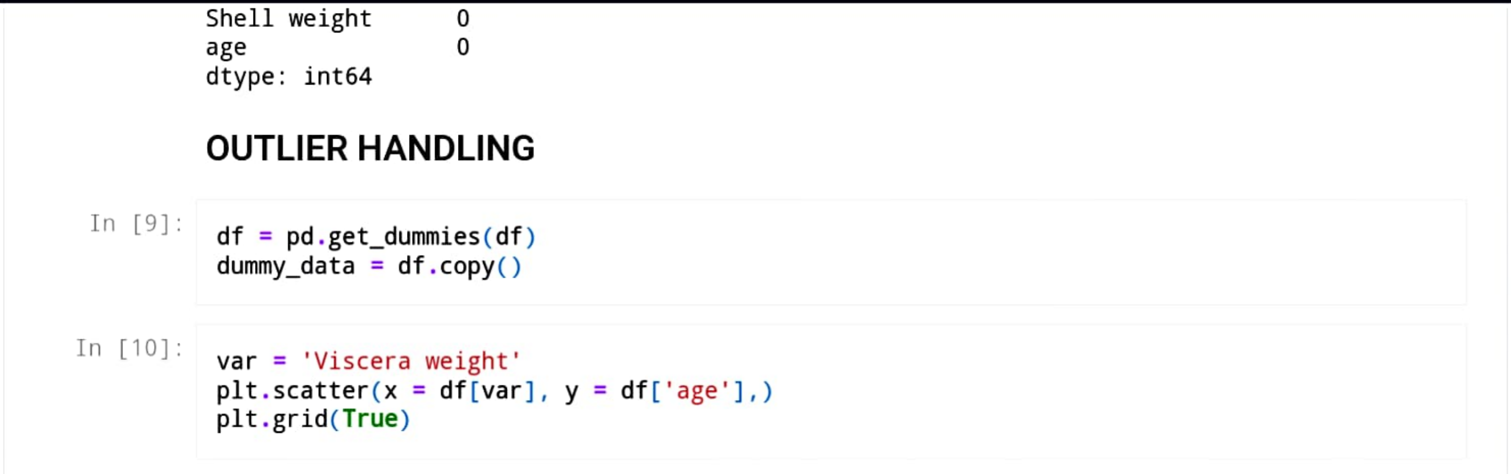
Check for Missing Values

In [8]:

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

Out[8]:

```
Sex               0
Length            0
Diameter          0
Height            0
Whole weight      0
Shucked weight    0
Viscera weight    0
Shell weight      0
```



```
Shell weight      0
age               0
dtype: int64
```

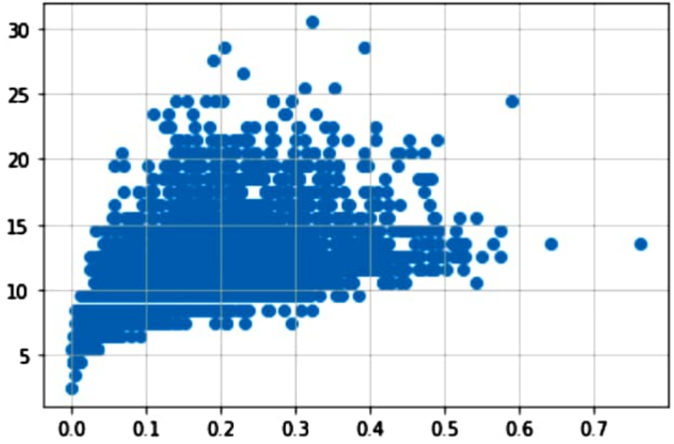
OUTLIER HANDLING

In [9]:

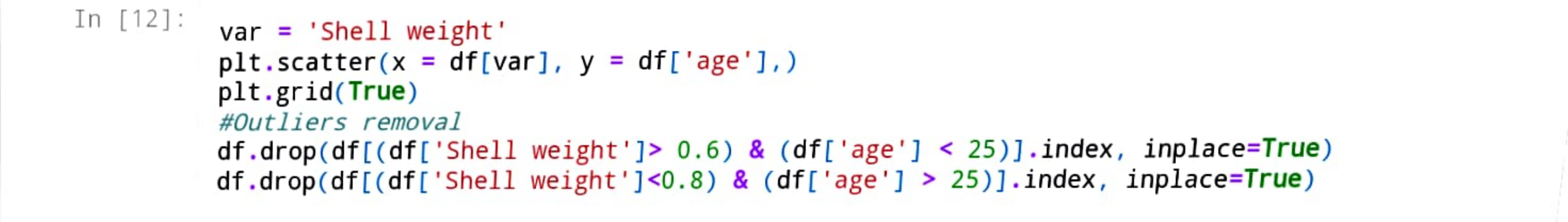
```
df = pd.get_dummies(df)
dummy_data = df.copy()
```

In [10]:

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

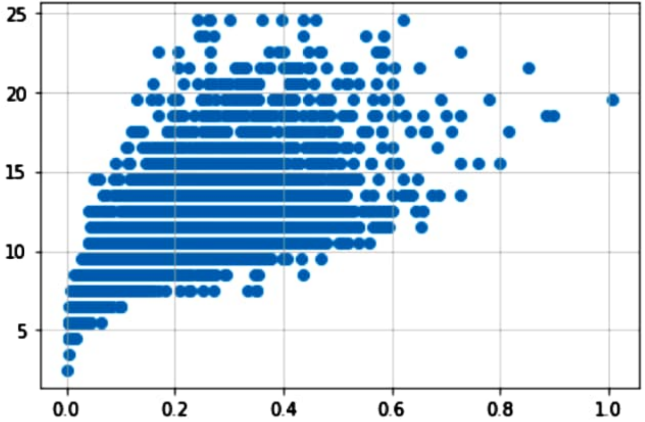


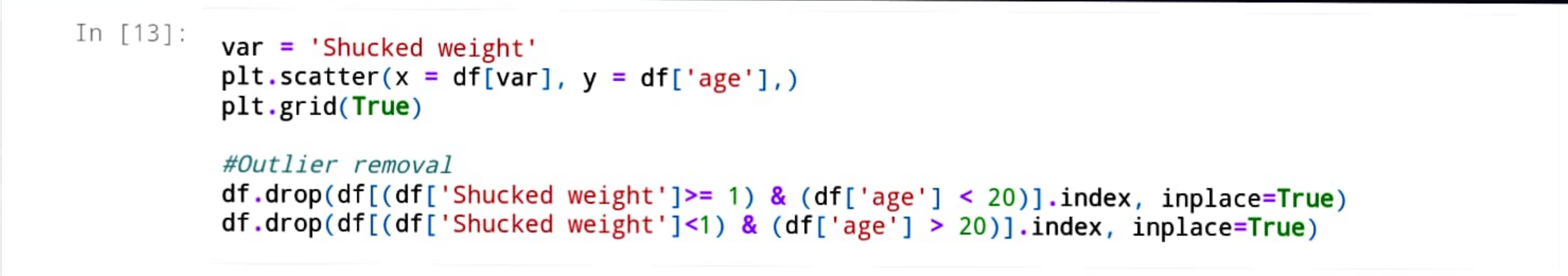
```
In [11]:

# outliers removal
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)
```



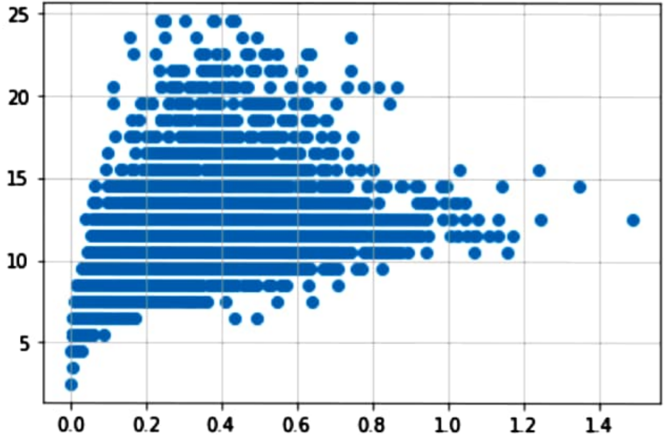
In [12]:

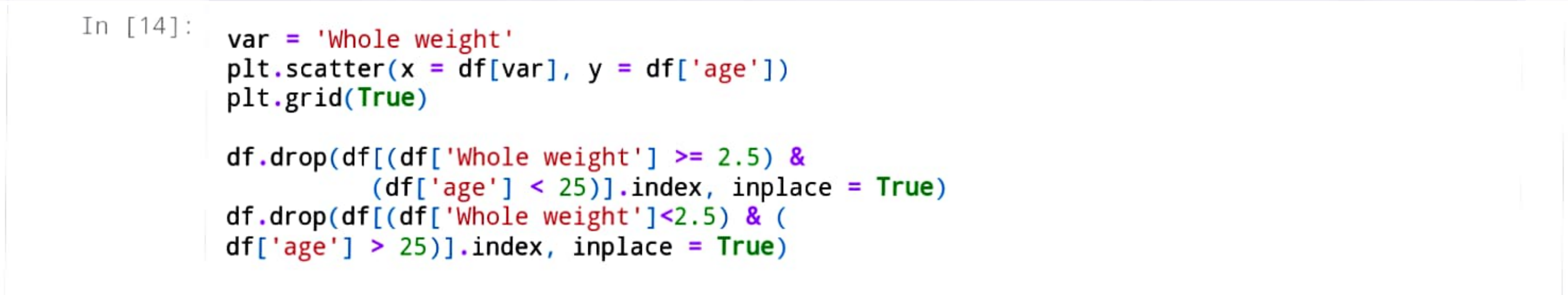
```
var = 'Shell weight'
plt.scatter(x = df[var], y = df['age'],)
plt.grid(True)
#Outliers removal
df.drop(df[(df['Shell weight']> 0.6) & (df['age'] < 25)].index, inplace=True)
df.drop(df[(df['Shell weight']<0.8) & (df['age'] > 25)].index, inplace=True)
```


25
20
15
10
5
0.0
0.2
0.4
0.6
0.8
1.0



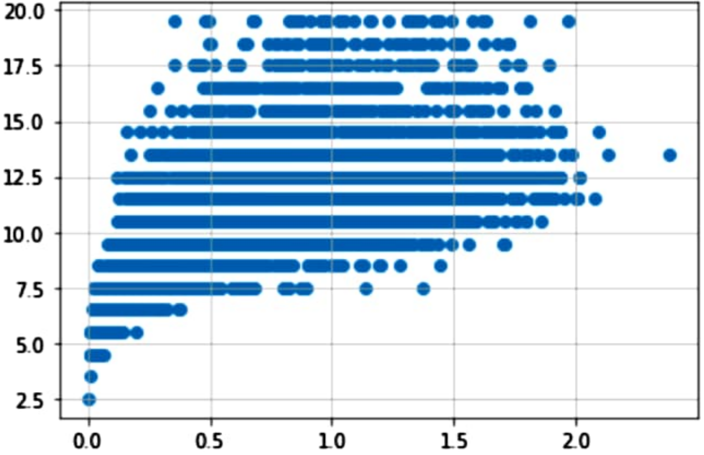
```
In [13]:  var = 'Shucked weight'
          plt.scatter(x = df[var], y = df['age'],)
          plt.grid(True)

          #Outlier removal
          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)
```

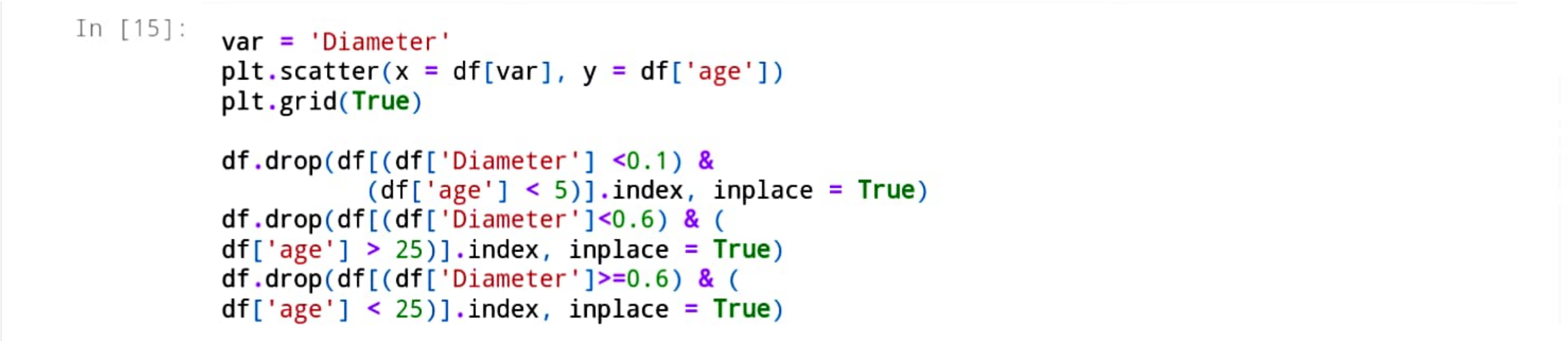

25
20
15
10
5
0.0
0.2
0.4
0.6
0.8
1.0
1.2
1.4



In [14]:

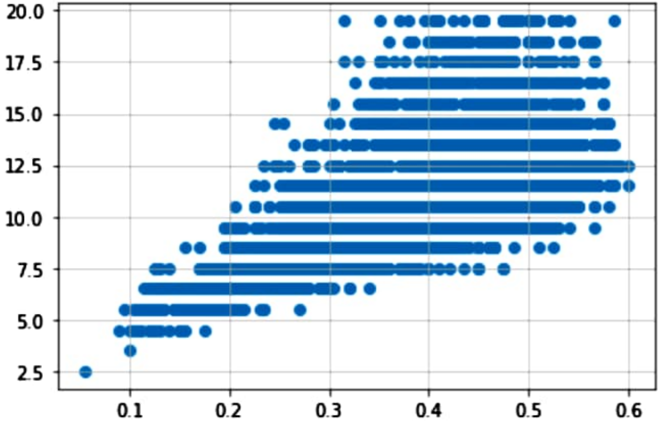
```
var = 'Whole weight'
plt.scatter(x = df[var], y = df['age'])
plt.grid(True)

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)
```

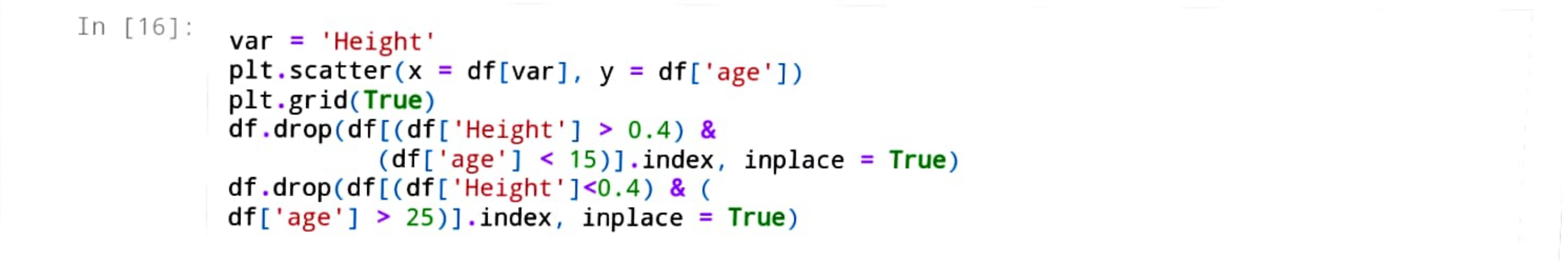

20.0
17.5
15.0
12.5
10.0
7.5
5.0
2.5
0.0
0.5
1.0
1.5
2.0



In [15]:

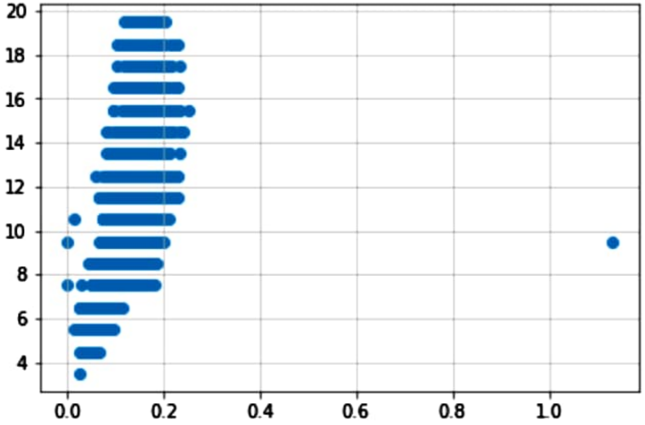
```
var = 'Diameter'
plt.scatter(x = df[var], y = df['age'])
plt.grid(True)

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)
```

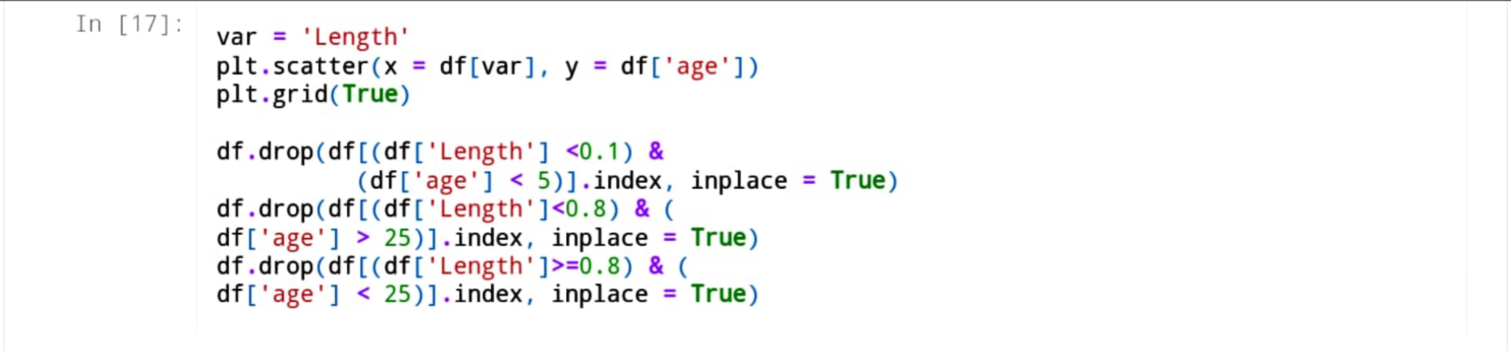

20.0
17.5
15.0
12.5
10.0
7.5
5.0
2.5
0.1
0.2
0.3
0.4
0.5
0.6



In [16]:

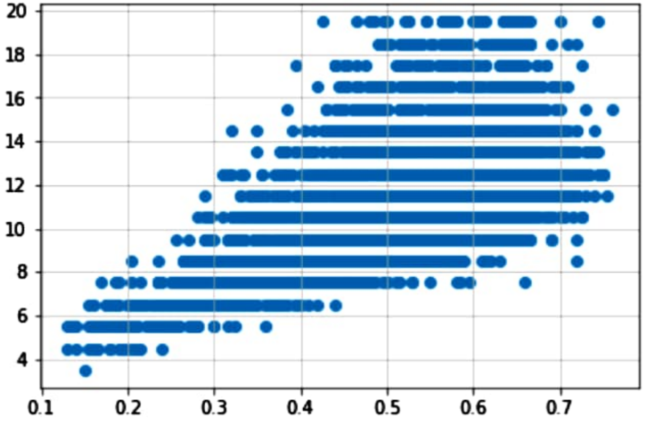
```
var = 'Height'
plt.scatter(x = df[var], y = df['age'])
plt.grid(True)
df.drop(df[(df['Height'] > 0.4) &
          (df['age'] < 15)].index, inplace = True)
df.drop(df[(df['Height']<0.4) & (
df['age'] > 25)].index, inplace = True)
```

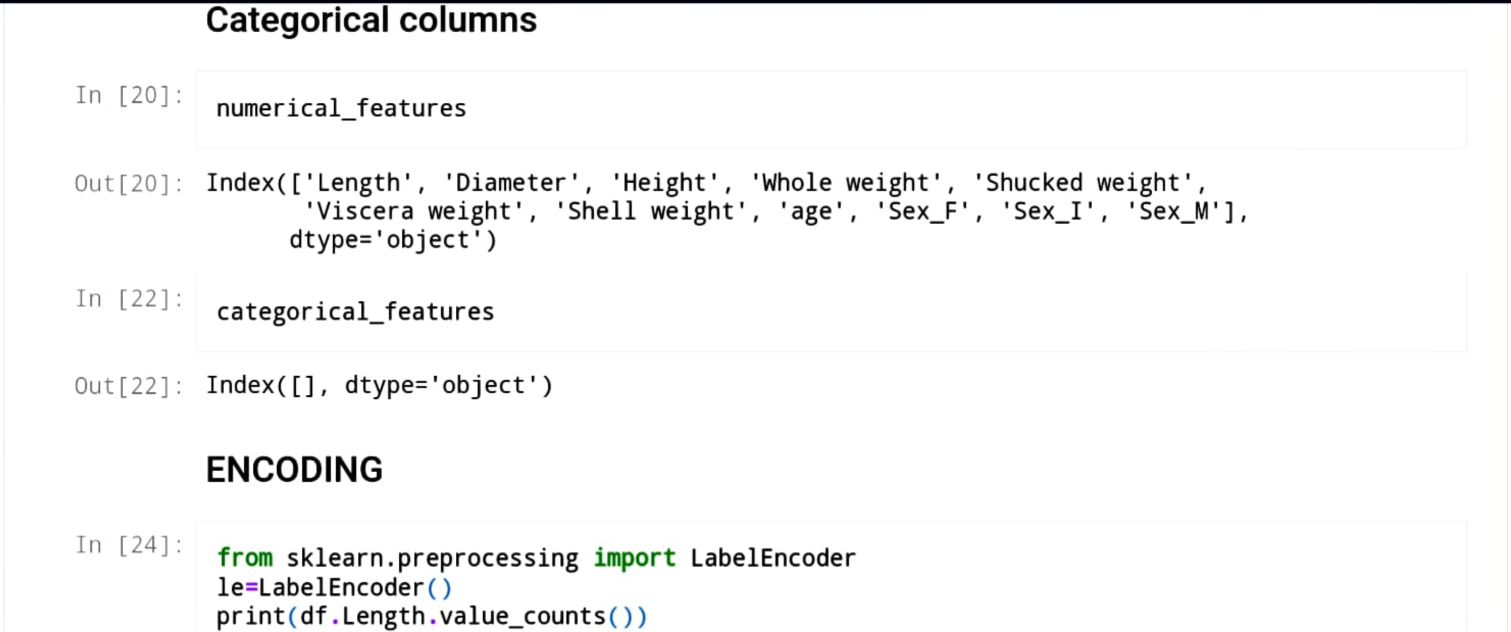

20
18
16
14
12
10
8
6
4
0.0
0.2
0.4
0.6
0.8
1.0



In [17]:

```
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)
```


20
18
16
14
12
10
8
6
4
0.1
0.2
0.3
0.4
0.5
0.6
0.7



Categorical columns

```
In [20]: numerical_features
```

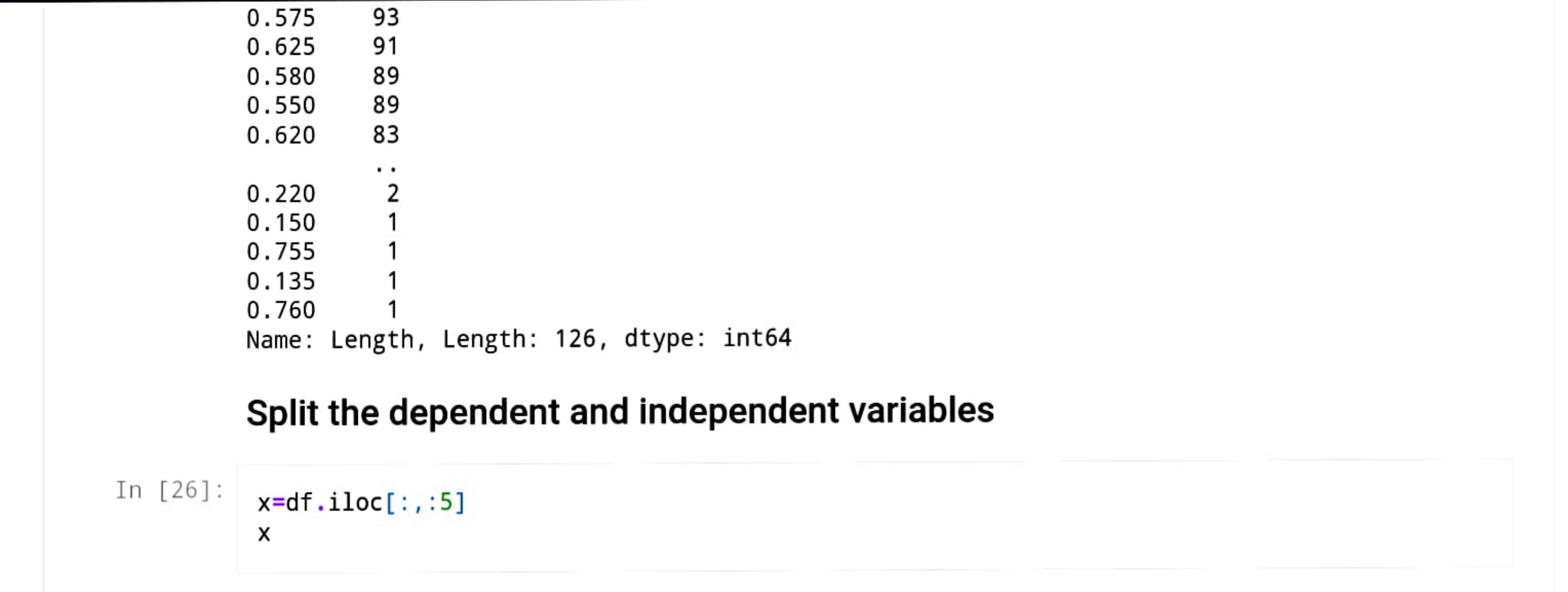
```
Out[20]: Index(['Length', 'Diameter', 'Height', 'Whole weight', 'Shucked weight',
                'Viscera weight', 'Shell weight', 'age', 'Sex_F', 'Sex_I', 'Sex_M'],
               dtype='object')
```

```
In [22]: categorical_features
```

```
Out[22]: Index([], dtype='object')
```

ENCODING

```
In [24]: from sklearn.preprocessing import LabelEncoder
         le=LabelEncoder()
         print(df.Length.value_counts())
```

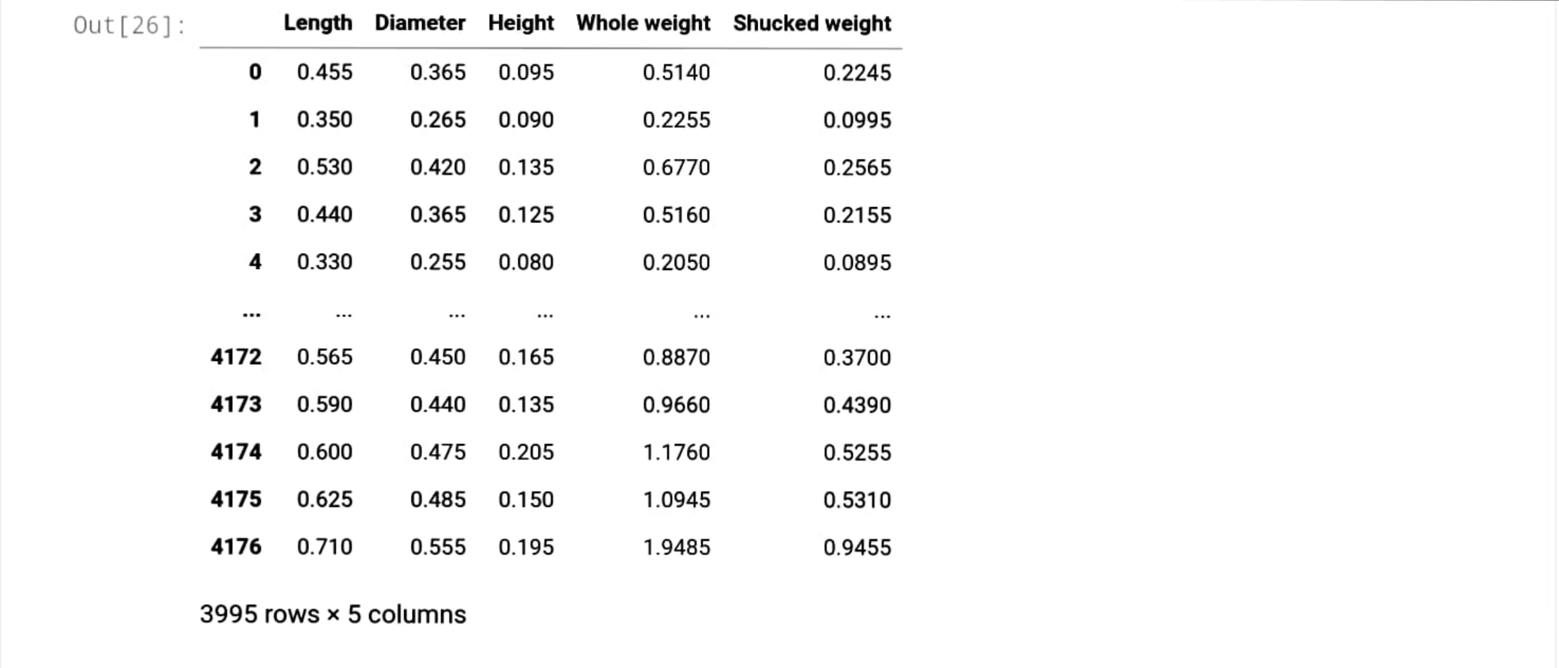


```
0.575    93
0.625    91
0.580    89
0.550    89
0.620    83
         ..
0.220     2
0.150     1
0.755     1
0.135     1
0.760     1
Name: Length, Length: 126, dtype: int64
```

Split the dependent and independent variables

In [26]:

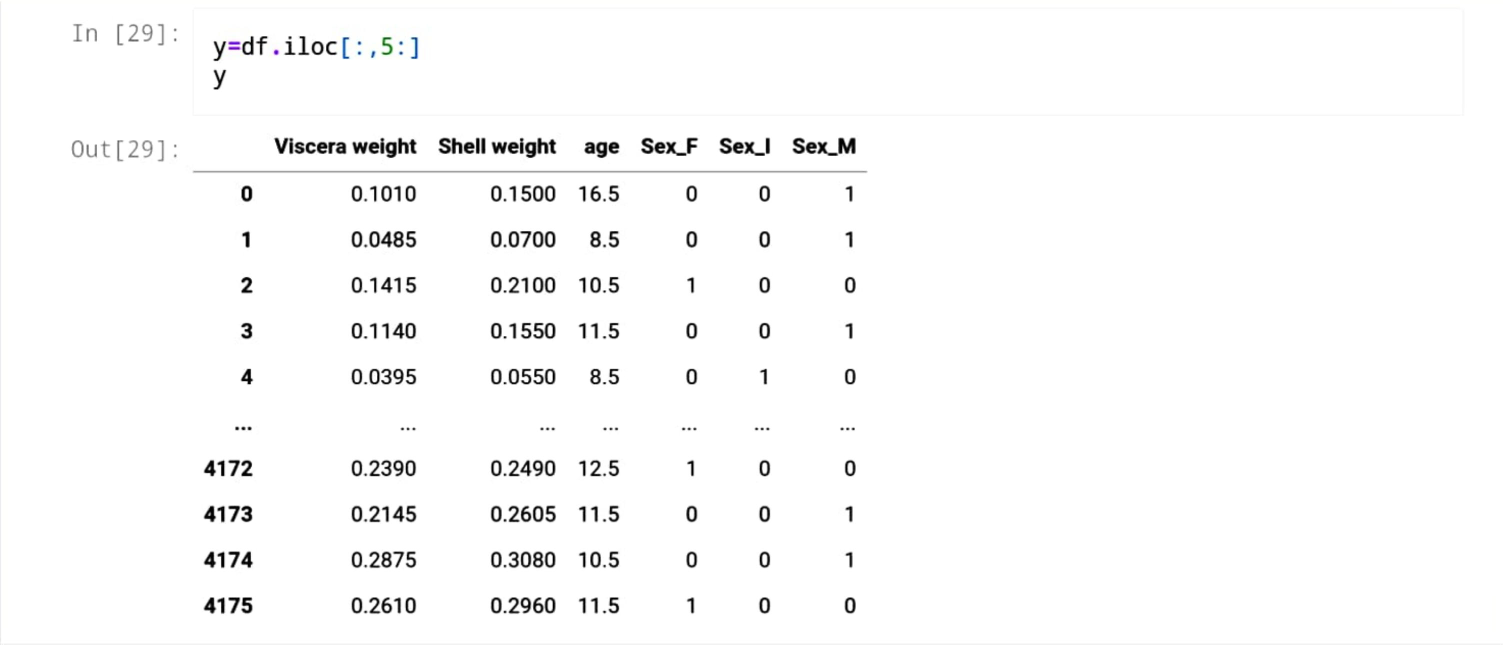
```
x=df.iloc[:,:5]
x
```



Out[26]:

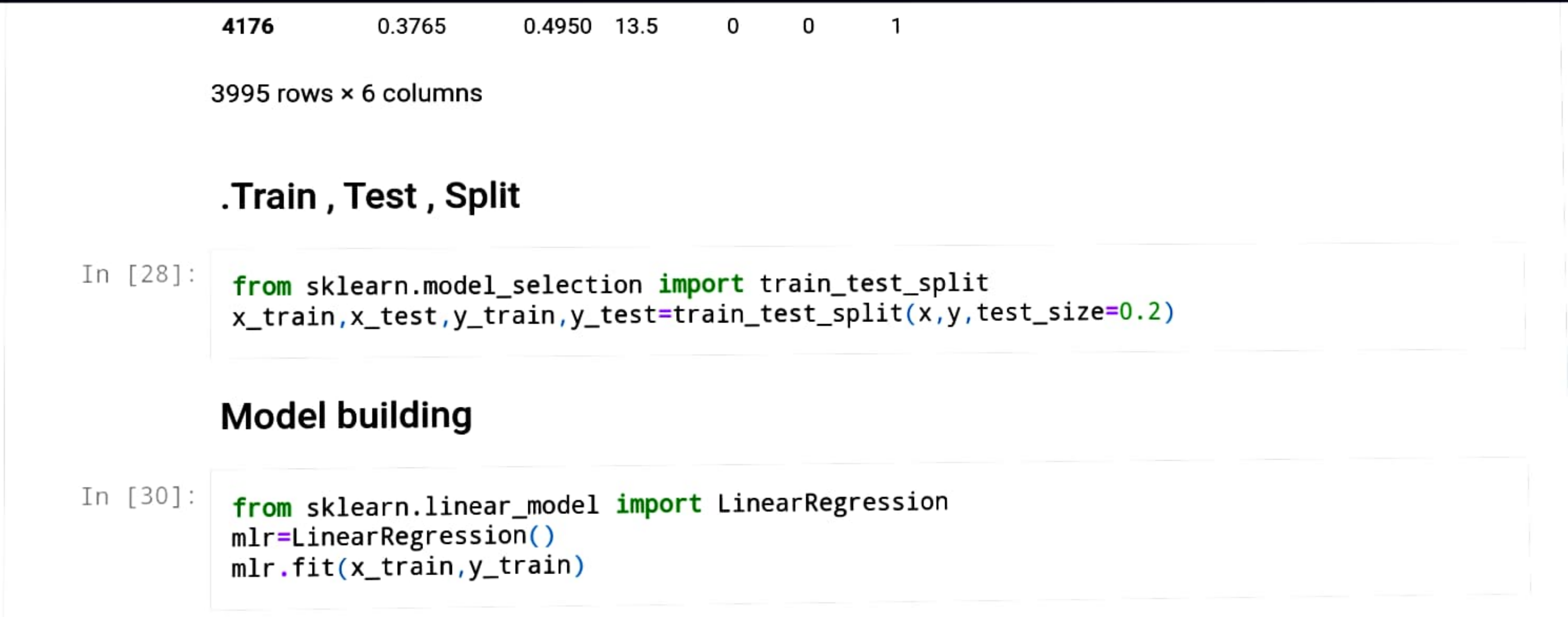
	Length	Diameter	Height	Whole weight	Shucked weight
0	0.455	0.365	0.095	0.5140	0.2245
1	0.350	0.265	0.090	0.2255	0.0995
2	0.530	0.420	0.135	0.6770	0.2565
3	0.440	0.365	0.125	0.5160	0.2155
4	0.330	0.255	0.080	0.2050	0.0895
...	...	...	...	...	...
4172	0.565	0.450	0.165	0.8870	0.3700
4173	0.590	0.440	0.135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

3995 rows × 5 columns



```
In [29]: y=df.iloc[:,5:]
         y
```

```
Out[29]:
         Viscera weight  Shell weight   age  Sex_F  Sex_I  Sex_M
      0          0.1010        0.1500  16.5      0      0      1
      1          0.0485        0.0700   8.5      0      0      1
      2          0.1415        0.2100  10.5      1      0      0
      3          0.1140        0.1550  11.5      0      0      1
      4          0.0395        0.0550   8.5      0      1      0
    ...             ...           ...   ...    ...    ...    ...
   4172          0.2390        0.2490  12.5      1      0      0
   4173          0.2145        0.2605  11.5      0      0      1
   4174          0.2875        0.3080  10.5      0      0      1
   4175          0.2610        0.2960  11.5      1      0      0
```



4176	0.3765	0.4950	13.5	0	0	1

3995 rows × 6 columns

.Train , Test , Split

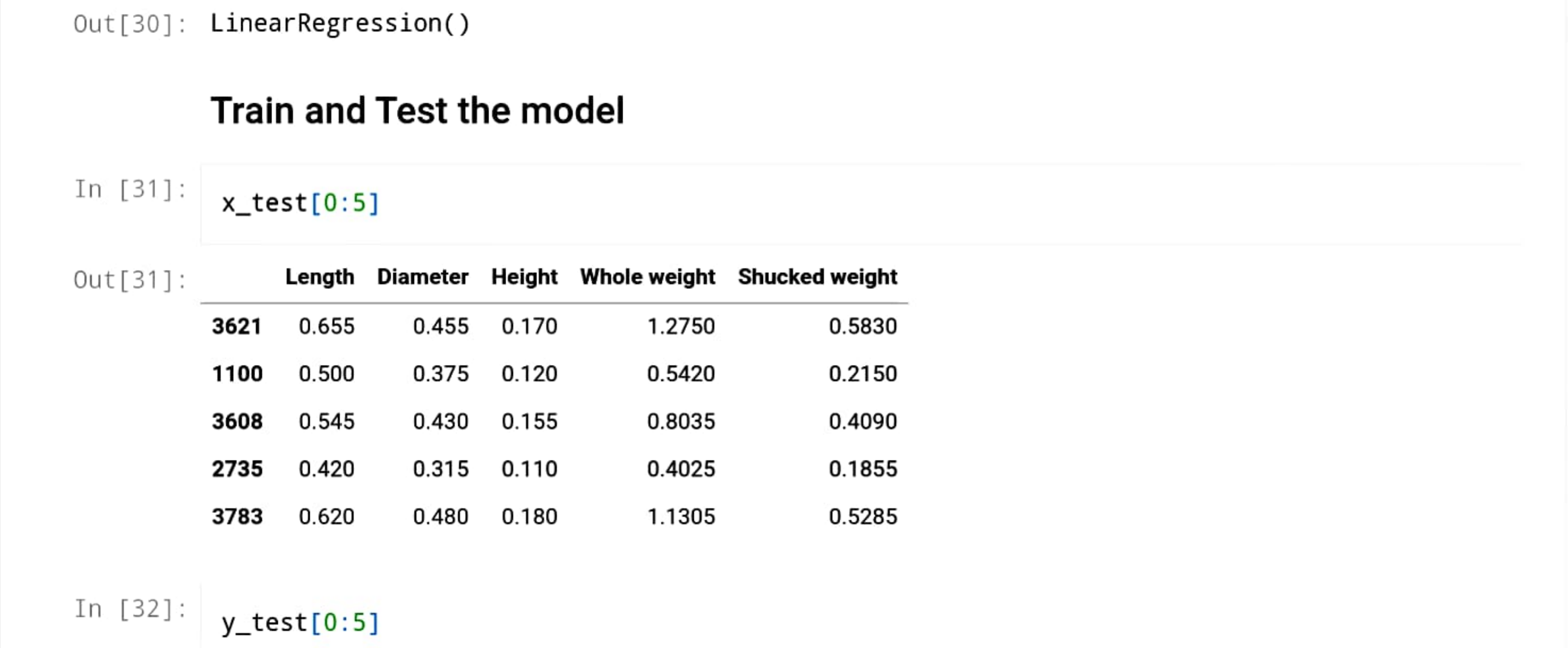
In [28]:

```
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)
```

Model building

In [30]:

```
from sklearn.linear_model import LinearRegression
mlr=LinearRegression()
mlr.fit(x_train,y_train)
```



```
Out[30]: LinearRegression()
```

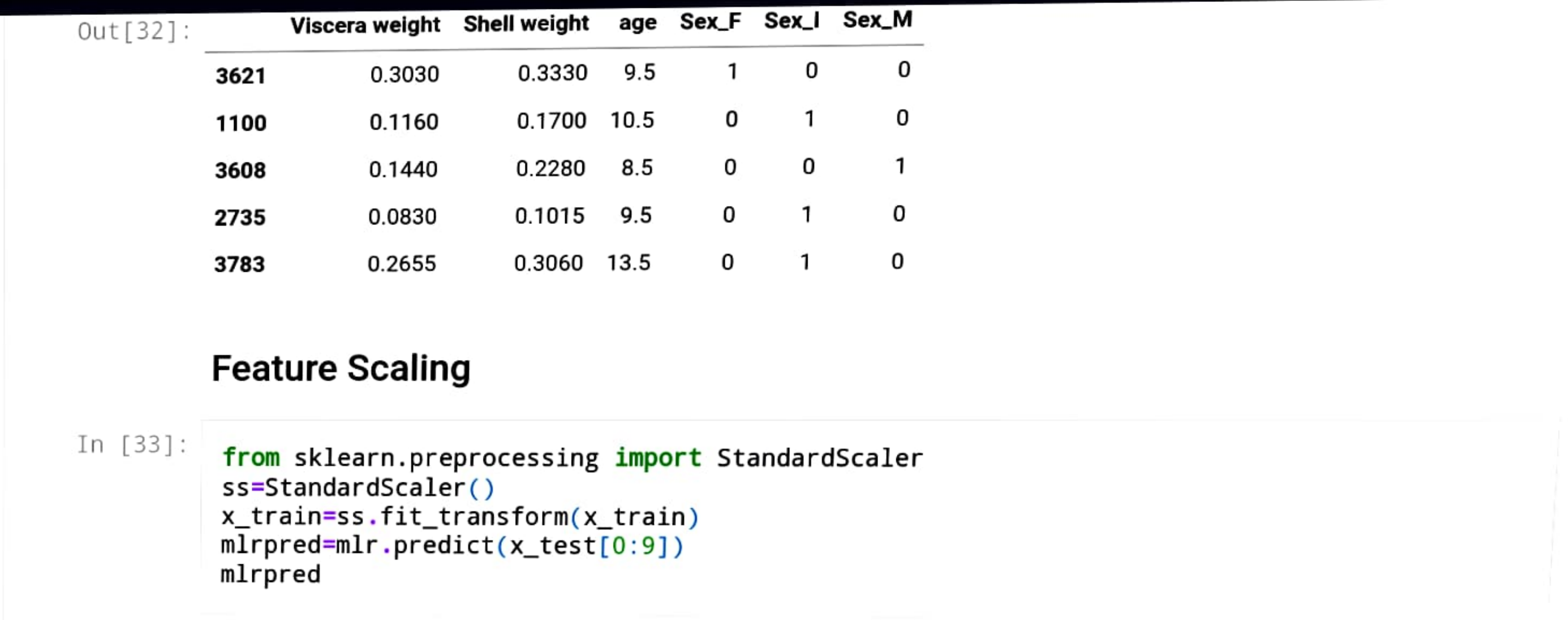
Train and Test the model

```
In [31]: x_test[0:5]
```

Out[31]:

	Length	Diameter	Height	Whole weight	Shucked weight
3621	0.655	0.455	0.170	1.2750	0.5830
1100	0.500	0.375	0.120	0.5420	0.2150
3608	0.545	0.430	0.155	0.8035	0.4090
2735	0.420	0.315	0.110	0.4025	0.1855
3783	0.620	0.480	0.180	1.1305	0.5285

```
In [32]: y_test[0:5]
```



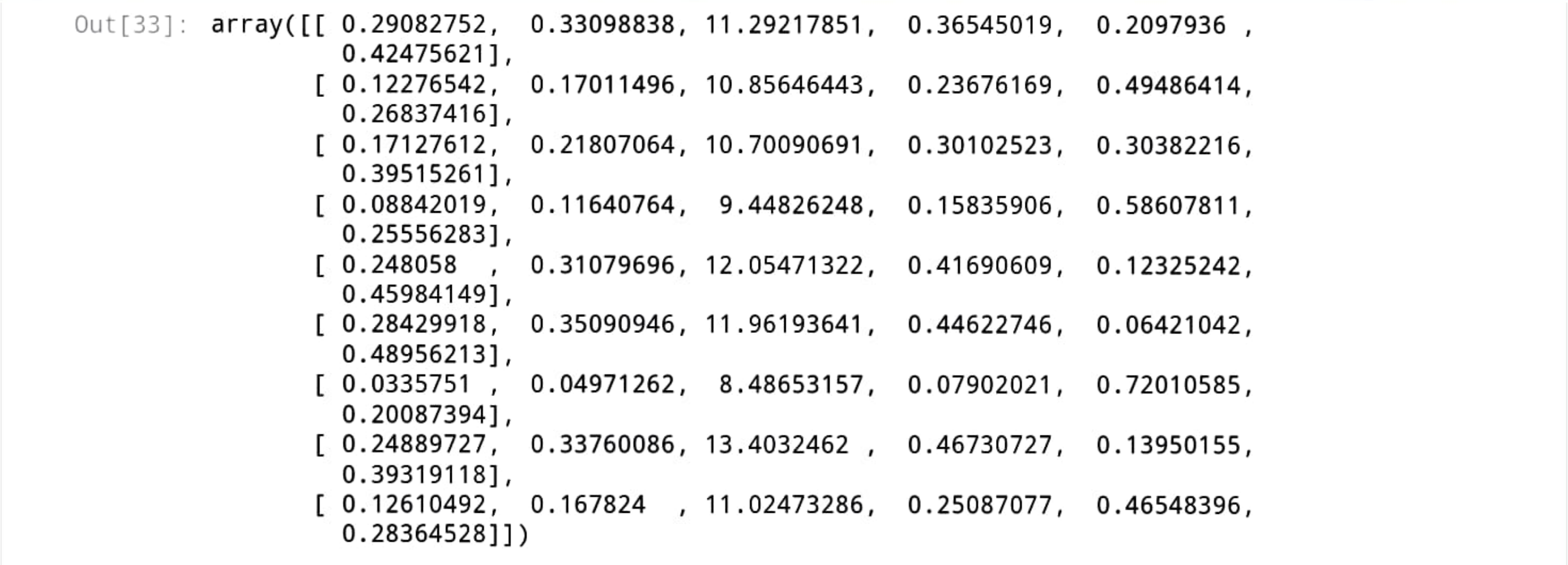
Out[32]:

	Viscera weight	Shell weight	age	Sex_F	Sex_I	Sex_M
3621	0.3030	0.3330	9.5	1	0	0
1100	0.1160	0.1700	10.5	0	1	0
3608	0.1440	0.2280	8.5	0	0	1
2735	0.0830	0.1015	9.5	0	1	0
3783	0.2655	0.3060	13.5	0	1	0

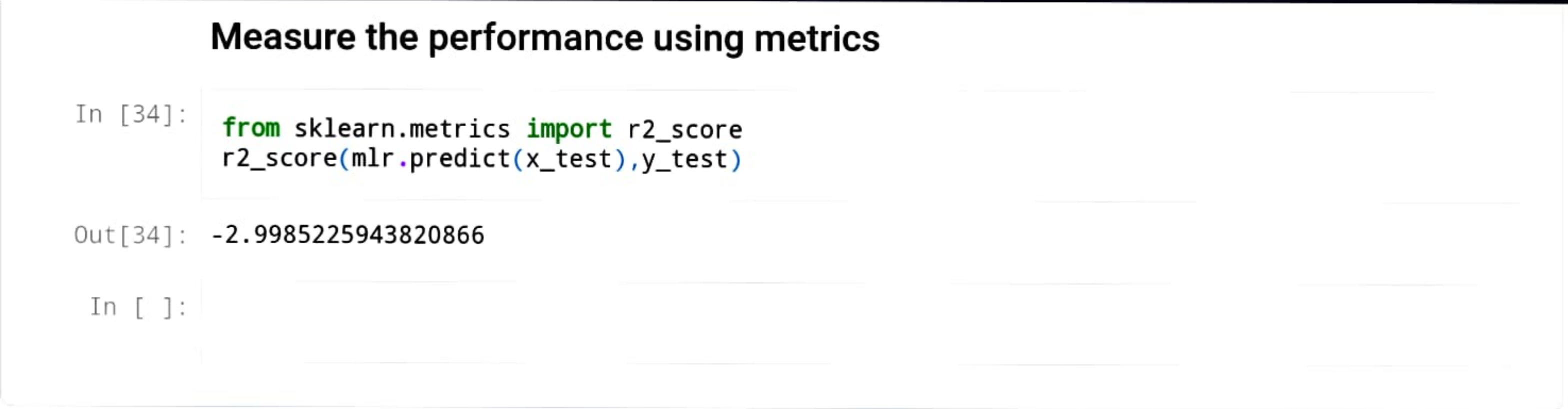
Feature Scaling

In [33]:

```
from sklearn.preprocessing import StandardScaler
ss=StandardScaler()
x_train=ss.fit_transform(x_train)
mlrpred=mlr.predict(x_test[0:9])
mlrpred
```



```
Out[33]: array([[ 0.29082752,  0.33098838, 11.29217851,  0.36545019,  0.2097936 ,
                  0.42475621],
                [ 0.12276542,  0.17011496, 10.85646443,  0.23676169,  0.49486414,
                  0.26837416],
                [ 0.17127612,  0.21807064, 10.70090691,  0.30102523,  0.30382216,
                  0.39515261],
                [ 0.08842019,  0.11640764,  9.44826248,  0.15835906,  0.58607811,
                  0.25556283],
                [ 0.248058  ,  0.31079696, 12.05471322,  0.41690609,  0.12325242,
                  0.45984149],
                [ 0.28429918,  0.35090946, 11.96193641,  0.44622746,  0.06421042,
                  0.48956213],
                [ 0.0335751 ,  0.04971262,  8.48653157,  0.07902021,  0.72010585,
                  0.20087394],
                [ 0.24889727,  0.33760086, 13.4032462 ,  0.46730727,  0.13950155,
                  0.39319118],
                [ 0.12610492,  0.167824  , 11.02473286,  0.25087077,  0.46548396,
                  0.28364528]])
```



Measure the performance using metrics

```
In [34]:  from sklearn.metrics import r2_score
          r2_score(mlr.predict(x_test),y_test)

Out[34]: -2.9985225943820866

In [ ]:
```