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

```
Drive already mounted at /content/drive; to attempt to forcibly
remount, call drive.mount("/content/drive", force_remount=True).
```

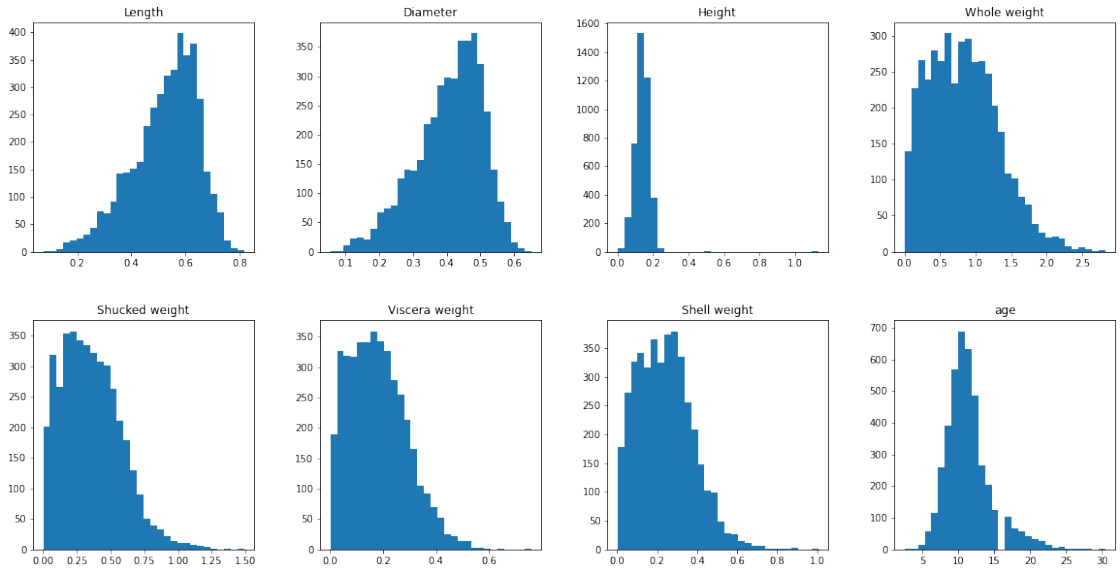
```
df=pd.read_csv("/content/drive/MyDrive/abalone.csv")

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  Rings
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]
```

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

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

```
array([[<matplotlib.axes._subplots.AxesSubplot object at
0x7fc787902410>,
        <matplotlib.axes._subplots.AxesSubplot object at
```

```
0x7fc787857b10>,
        <matplotlib.axes._subplots.AxesSubplot object at
0x7fc787820150>,
        <matplotlib.axes._subplots.AxesSubplot object at
0x7fc7877d7750>],
       [<matplotlib.axes._subplots.AxesSubplot object at
0x7fc78780dd50>,
        <matplotlib.axes._subplots.AxesSubplot object at
0x7fc7877cf390>,
        <matplotlib.axes._subplots.AxesSubplot object at
0x7fc787786a10>,
        <matplotlib.axes._subplots.AxesSubplot object at
0x7fc78773cf90>]],
      dtype=object)
```

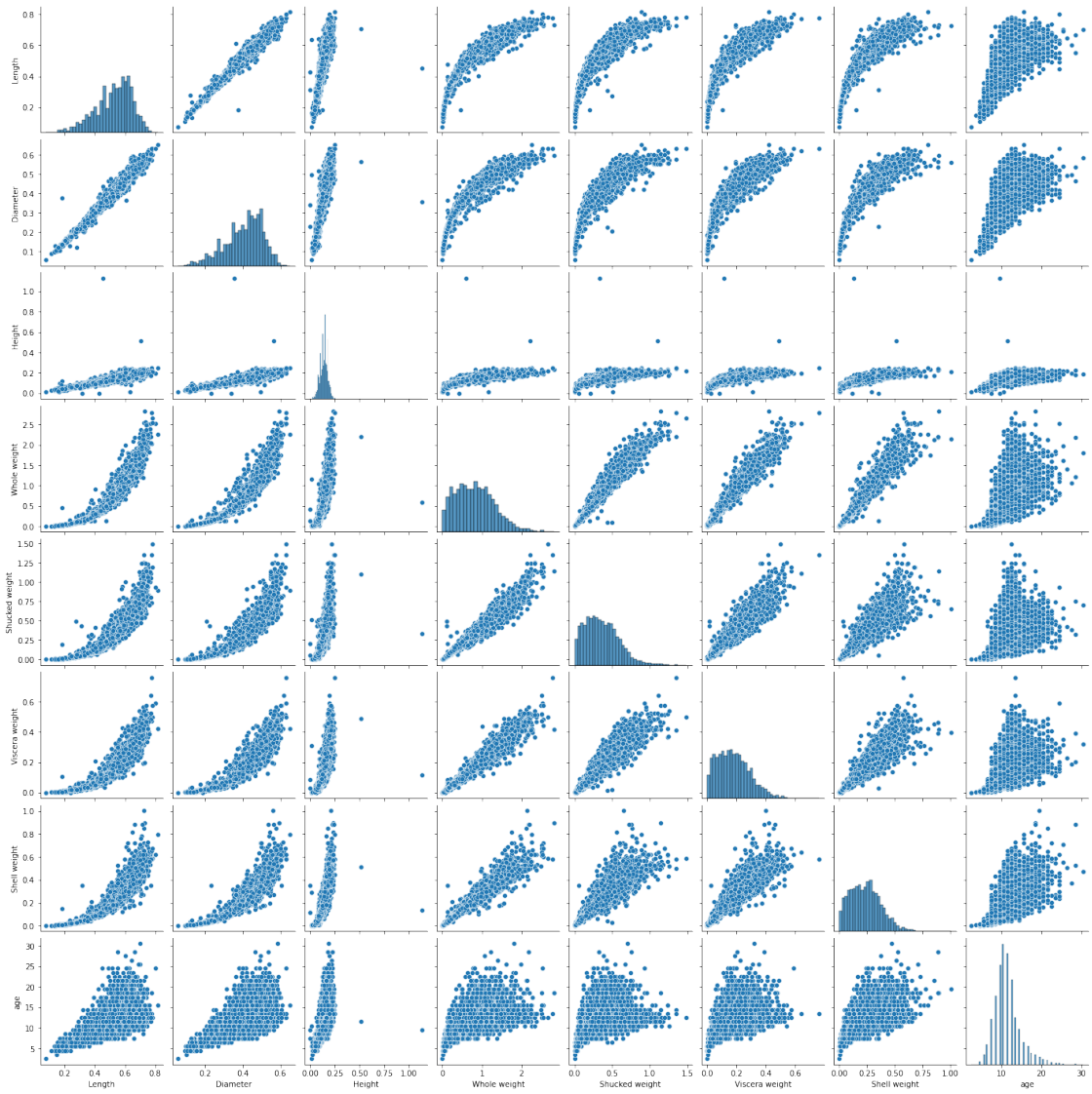



```
df.groupby('Sex')[['Length', 'Diameter', 'Height', 'Whole weight',
'Shucked weight',
       'Viscera weight', 'Shell weight',
'age']].mean().sort_values('age')
```

```
       Length  Diameter    Height  Whole weight  Shucked weight  \
Sex
I    0.427746  0.326494  0.107996      0.431363        0.191035
M    0.561391  0.439287  0.151381      0.991459        0.432946
F    0.579093  0.454732  0.158011      1.046532        0.446188

     Viscera weight  Shell weight        age
Sex
I          0.092010      0.128182   9.390462
M          0.215545      0.281969  12.205497
F          0.230689      0.302010  12.629304
```

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

```
<seaborn.axisgrid.PairGrid at 0x7fc787274bd0>
```




```
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
```

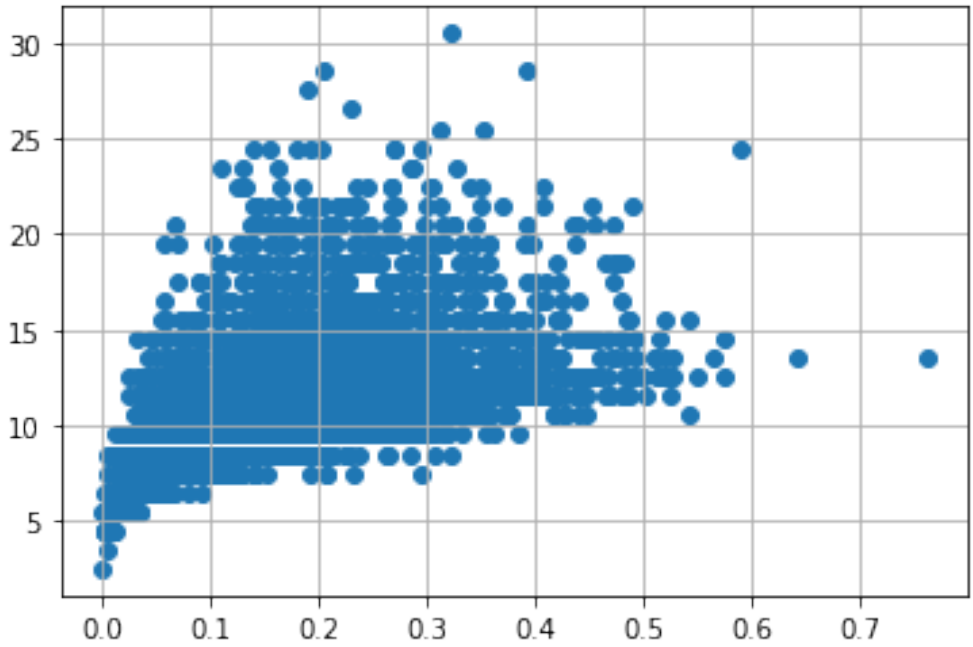
```
       Viscera weight  Shell weight          age
count     4177.000000   4177.000000  4177.000000
mean         0.180594      0.238831    11.433684
std          0.109614      0.139203     3.224169
min          0.000500      0.001500     2.500000
25%          0.093500      0.130000     9.500000
50%          0.171000      0.234000    10.500000
75%          0.253000      0.329000    12.500000
max          0.760000      1.005000    30.500000
```

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

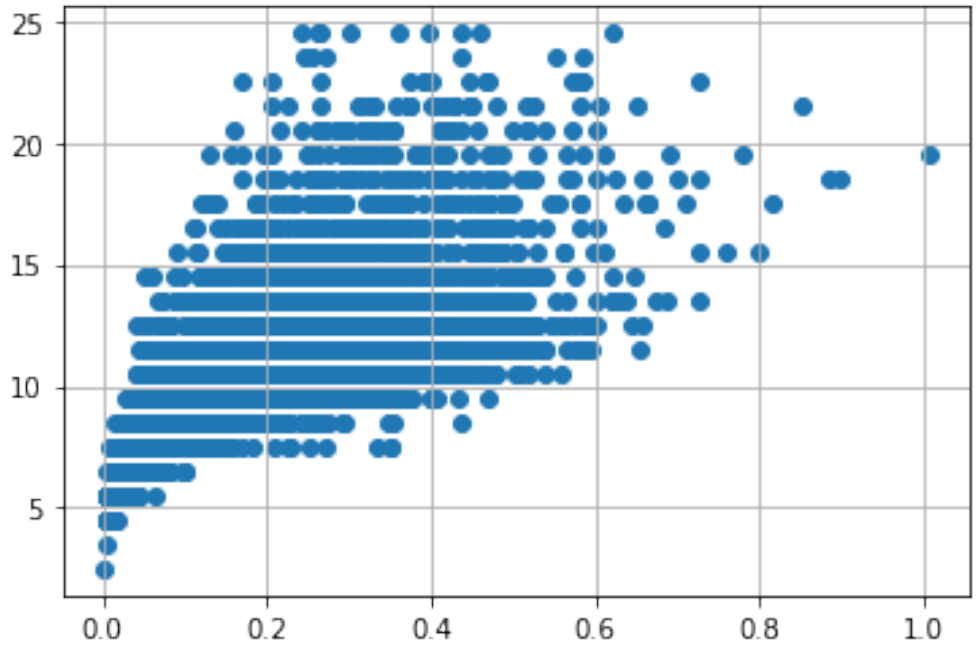
```
Sex               0
Length            0
Diameter          0
Height            0
Whole weight      0
Shucked weight    0
Viscera weight    0
Shell weight      0
age               0
dtype: int64
```

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

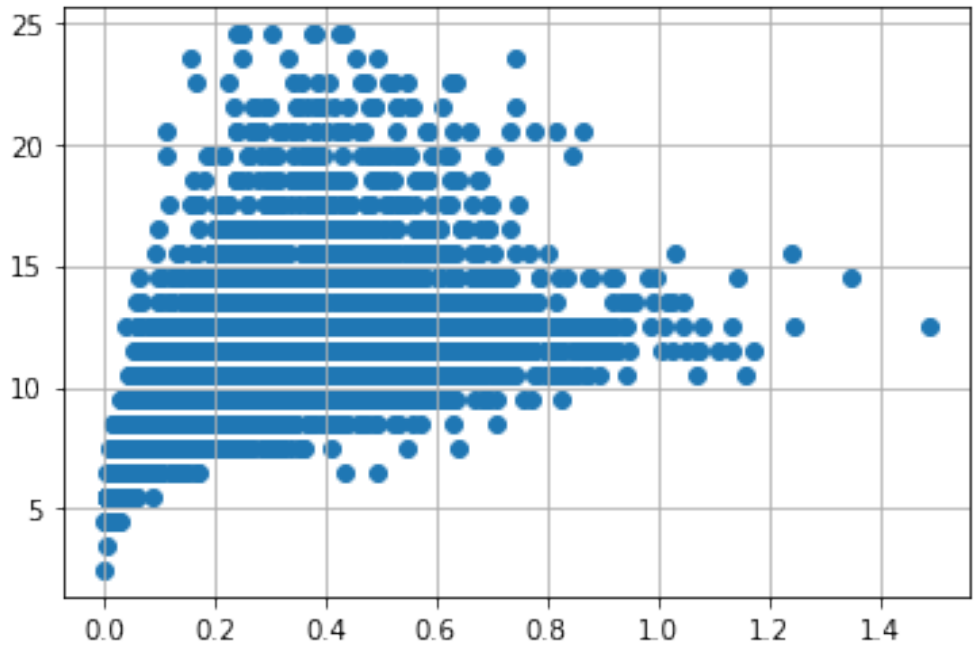
```
var = 'Viscera weight'
plt.scatter(x = df[var], y = df['age'],)
plt.grid(True)
```



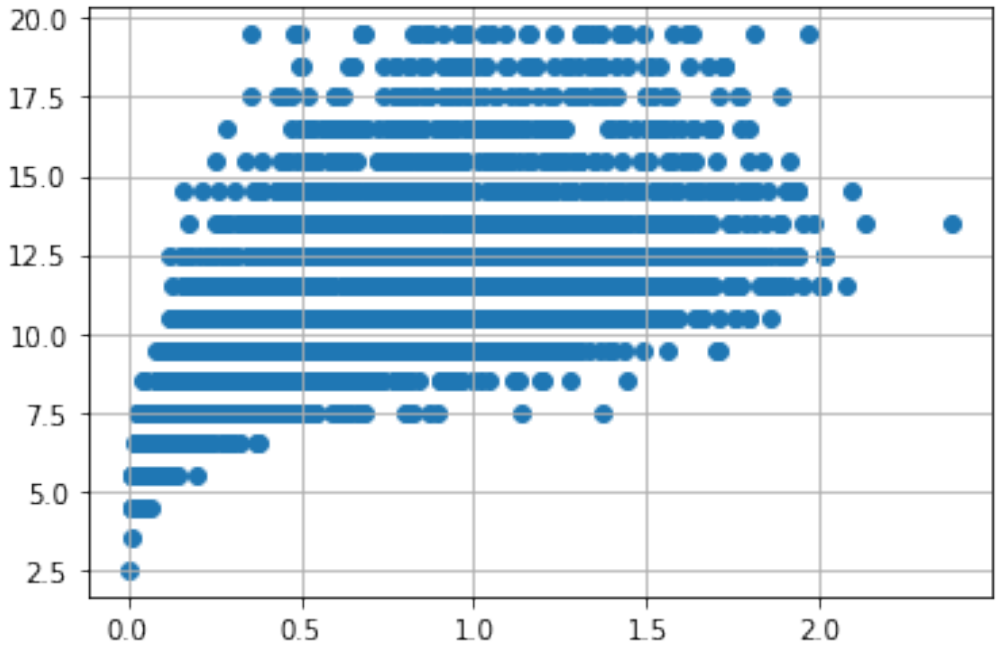

```
df.drop(df[(df['Viscera weight']> 0.5) & (df['age'] < 20)].index,
inplace=True)
df.drop(df[(df['Viscera weight']<0.5) & (df['age'] > 25)].index,
inplace=True)

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)
```



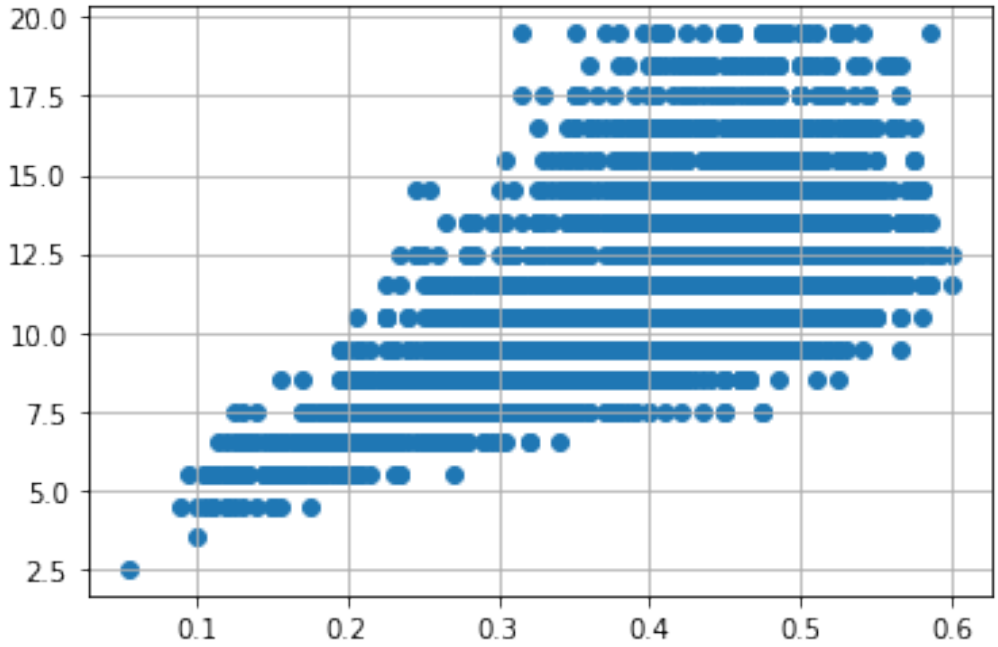

```
var = 'Shucked weight'
plt.scatter(x = df[var], y = df['age'],)
plt.grid(True)

#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)
```



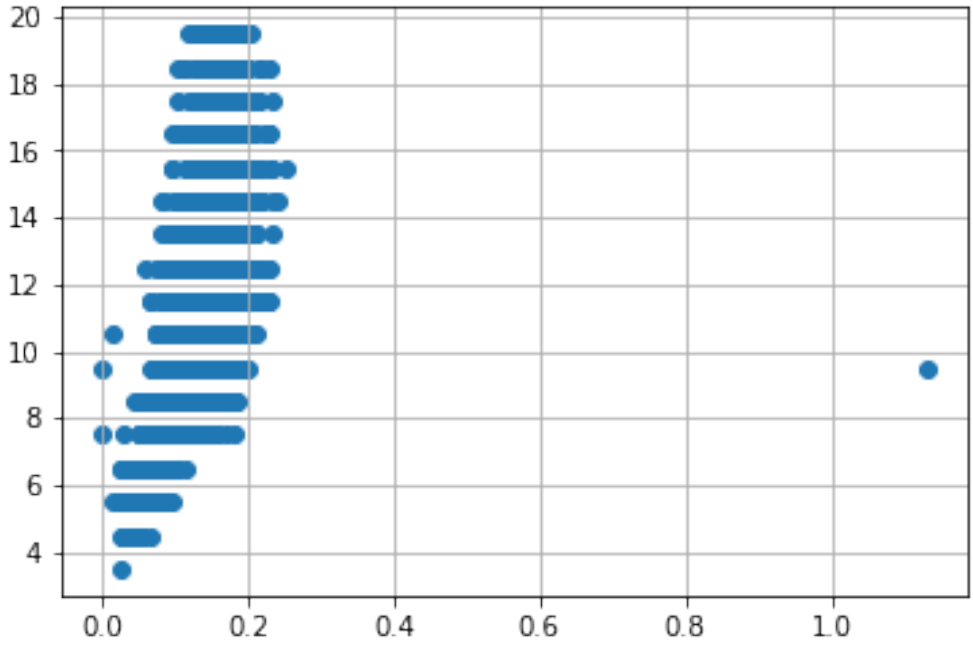
```
var = 'Whole weight'
plt.scatter(x = df[var], y = df['age'])
plt.grid(True)

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)
```




```
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)
```

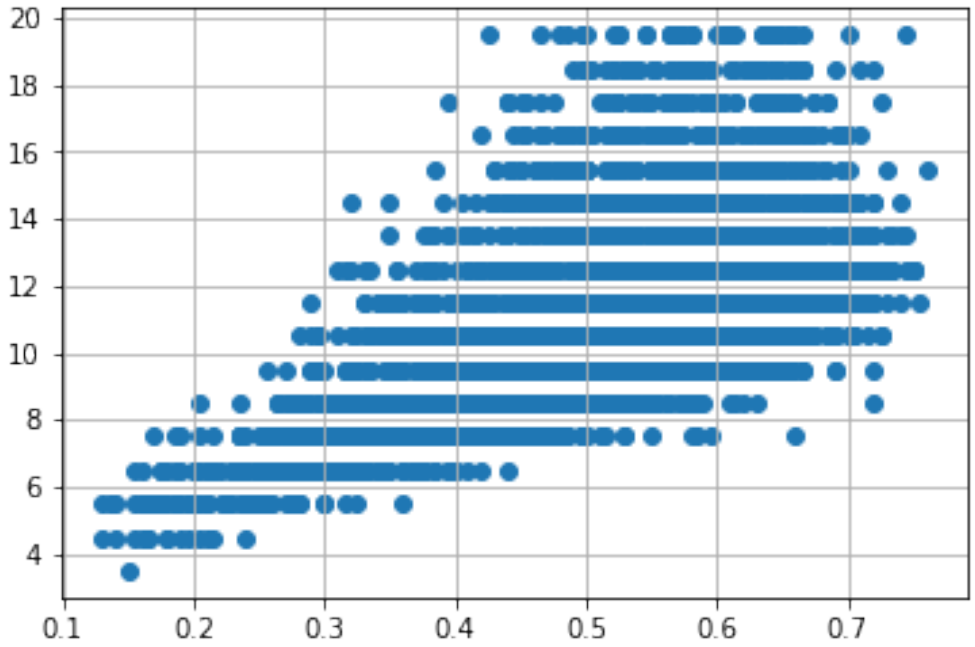



```
var = 'Height'
plt.scatter(x = df[var], y = df['age'])
plt.grid(True)
df.drop(df[(df['Height'] > 0.4) &
           (df['age'] < 15)].index, inplace = True)
df.drop(df[(df['Height']<0.4) & (
df['age'] > 25)].index, inplace = True)
```




```
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)
```




```
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.object` is a deprecated alias for the builtin
`object`. To silence this warning, use `object` by itself. Doing this
will not modify any behavior and is safe.
Deprecated in NumPy 1.20; for more details and guidance:
https://numpy.org/devdocs/release/1.20.0-notes.html#deprecations
```

```
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')
```

```
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
```

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

	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 x 5 columns]

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

	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 x 6 columns]
```

```
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)
```

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

```
LinearRegression()
```

```
x_test[0:5]
```

```
      Length  Diameter  Height  Whole weight  Shucked weight
3552   0.525     0.390   0.105        0.5670          0.2875
1510   0.650     0.545   0.175        1.5245          0.5900
3227   0.450     0.350   0.135        0.5600          0.2310
1836   0.415     0.310   0.090        0.2815          0.1245
3245   0.695     0.550   0.160        1.6365          0.6940
```

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

```
      Viscera weight  Shell weight   age  Sex_F  Sex_I  Sex_M
3552          0.1075         0.160   9.5      0      1      0
1510          0.3260         0.495  11.5      1      0      0
3227          0.1370         0.145  14.5      1      0      0
1836          0.0615         0.085   7.5      0      1      0
3245          0.3005         0.440  14.5      1      0      0
```

```
from sklearn.preprocessing import StandardScaler
ss=StandardScaler()
x_train=ss.fit_transform(x_train)
```

```
mlrpred=mlr.predict(x_test[0:9])
```

```
mlrpred
```

```
array([[ 0.12314288,  0.14882697,  9.50559719,  0.18763353,
0.54439026,
         0.26797621],
       [ 0.33411071,  0.45159809, 14.69490488,  0.61558621, -
0.16162173,
         0.54603552],
       [ 0.12416086,  0.17400362, 10.87336437,  0.25123873,
0.43361199,
         0.31514928],
       [ 0.06201807,  0.08539516,  9.15237015,  0.12736167,
0.67171807,
         0.20092025],
       [ 0.35857699,  0.44671913, 13.18812612,  0.54811905, -
0.08587925,
```

```
         0.53776019],
       [ 0.13878852,  0.18969716, 11.04119757,  0.27244889,
0.39558275,
         0.33196836],
       [ 0.09704726,  0.13899172, 10.28454348,  0.20876116,
0.49312395,
         0.29811489],
       [ 0.1673652 ,  0.20173416, 11.27270707,  0.23972914,
0.41169787,
         0.34857299],
       [ 0.08574993,  0.13356167, 10.31758419,  0.21821504,  0.5128507
,
         0.26893426]])
```

```
from sklearn.metrics import r2_score
r2_score(mlr.predict(x_test),y_test)
```

```
-3.4012749811462384
```