ASSIGNMENT-4

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
sns.set_style("darkgrid")


from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder

from sklearn.linear_model import LinearRegression

from sklearn import metrics

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

df.head(10)
```

```
  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
5   I   0.425     0.300   0.095        0.3515          0.1410
0.0775
6   F   0.530     0.415   0.150        0.7775          0.2370
0.1415
7   F   0.545     0.425   0.125        0.7680          0.2940
0.1495
8   M   0.475     0.370   0.125        0.5095          0.2165
0.1125
9   F   0.550     0.440   0.150        0.8945          0.3145
0.1510

   Shell weight  Rings
0         0.150     15
1         0.070      7
2         0.210      9
3         0.155     10
```

```
4          0.055      7
5          0.120      8
6          0.330     20
7          0.260     16
8          0.165      9
9          0.320     19
```

```
df.shape
```

```
(4177, 9)
```

```
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        Rings
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.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
```

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

```
df.head(10)
```

```
  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
5   I   0.425     0.300   0.095        0.3515          0.1410
0.0775
6   F   0.530     0.415   0.150        0.7775          0.2370
0.1415
7   F   0.545     0.425   0.125        0.7680          0.2940
0.1495
8   M   0.475     0.370   0.125        0.5095          0.2165
0.1125
9   F   0.550     0.440   0.150        0.8945          0.3145
0.1510

   Shell weight   age
0         0.150  16.5
1         0.070   8.5
2         0.210  10.5
3         0.155  11.5
4         0.055   8.5
5         0.120   9.5
6         0.330  21.5
7         0.260  17.5
8         0.165  10.5
9         0.320  20.5
```

```
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
```

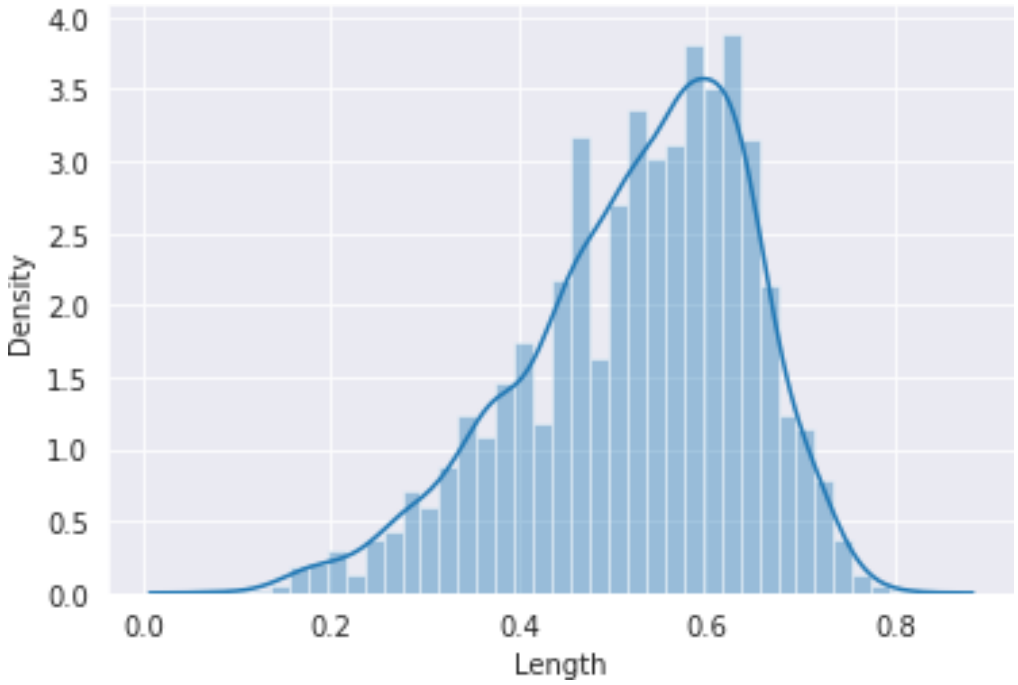
```
df.columns
```

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

```
sns.distplot(df['Length'])
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/distributions.py:2619:
FutureWarning: `distplot` is a deprecated function and will be removed
in a future version. Please adapt your code to use either `displot` (a
figure-level function with similar flexibility) or `histplot` (an
axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```

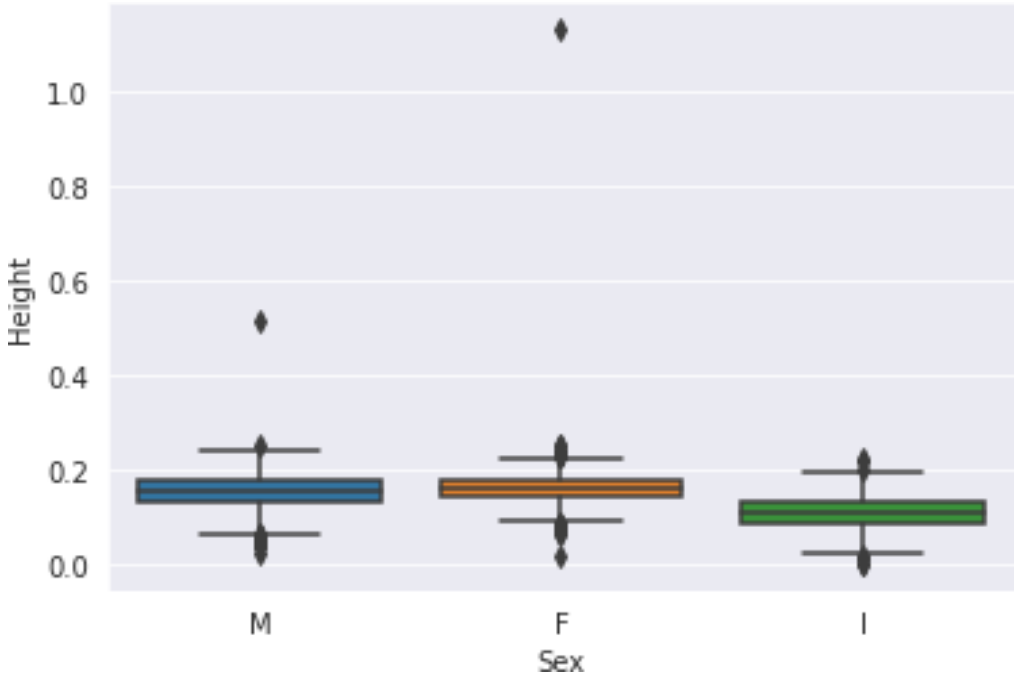
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f6285125e10>
```




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

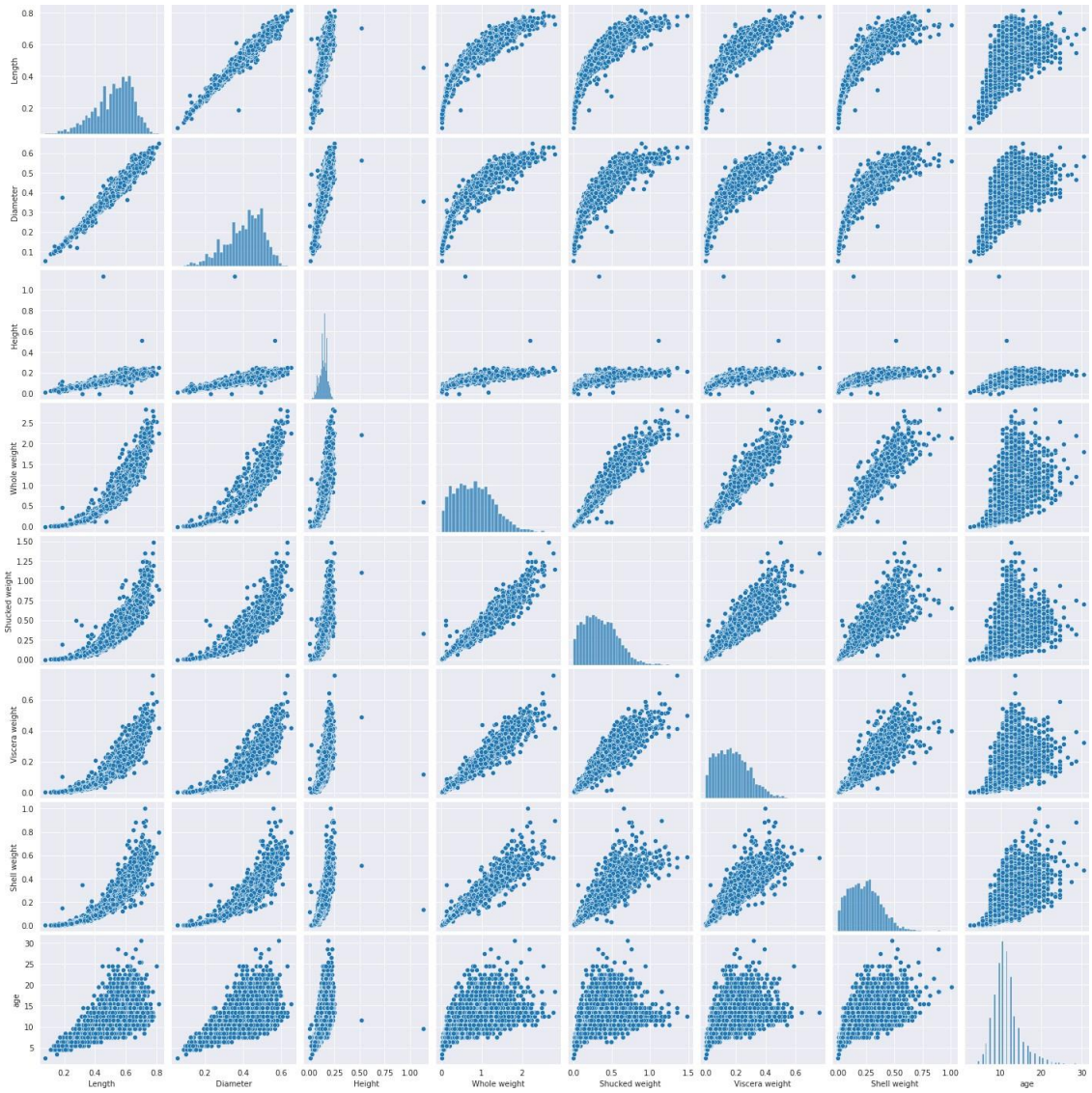
```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43:
FutureWarning: Pass the following variables as keyword args: x, y.
From version 0.12, the only valid positional argument will be `data`,
and passing other arguments without an explicit keyword will result in
an error or misinterpretation.
  FutureWarning
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f62847d9810>
```

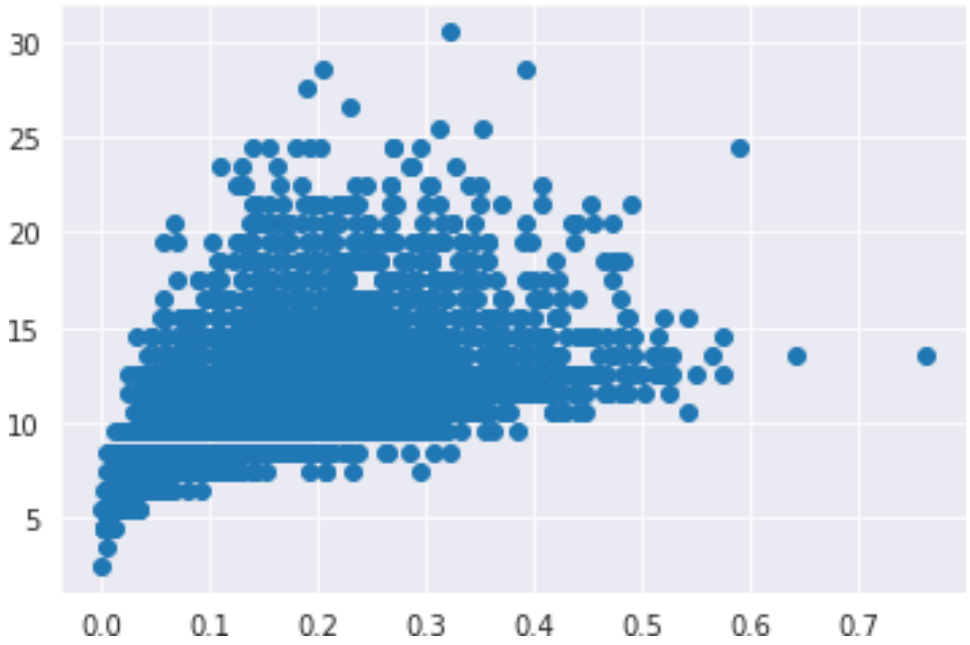



```
sns.pairplot(df)
```

```
<seaborn.axisgrid.PairGrid at 0x7f62842be8d0>
```

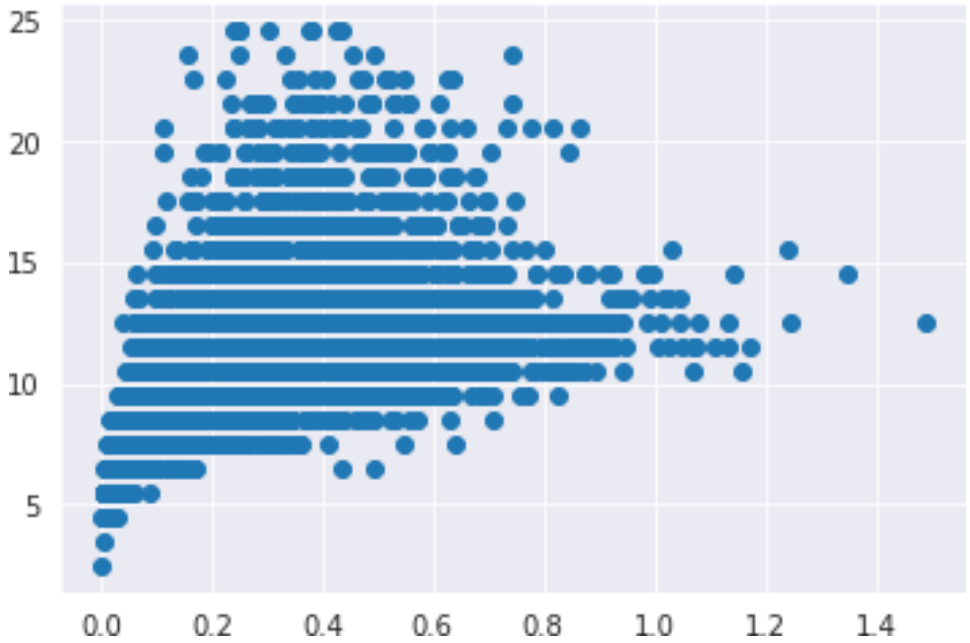



```
df = pd.get_dummies(df)
dummy_df = df

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

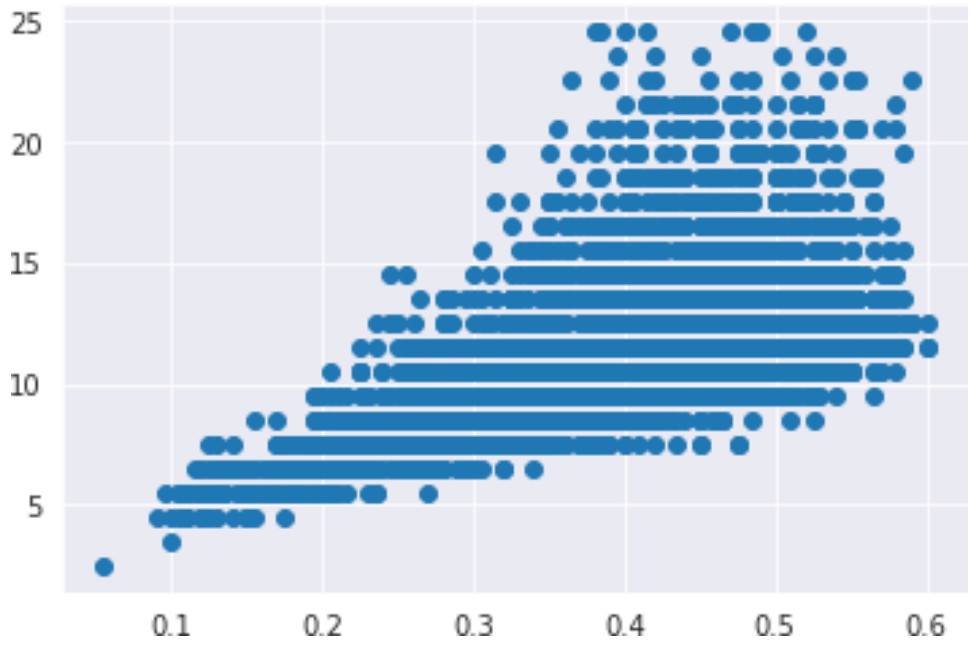



```
df.drop(df[(df['Viscera weight'] > 0.5) &
           (df['age'] < 20)].index, inplace = True)
df.drop(df[(df['Viscera weight']<0.5) & (
df['age'] > 25)].index, inplace = True)

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

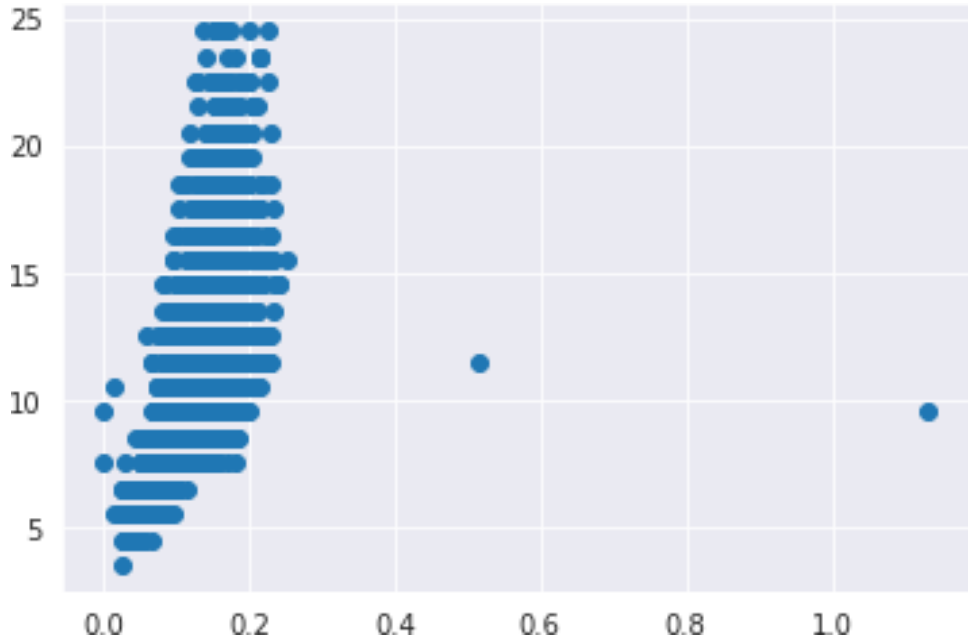
```
df.drop(df[(df['Shell weight'] > 0.6) & (df['age'] < 25)].index,
inplace = True)
df.drop(df[(df['Shell weight']<0.8) & (df['age'] > 25)].index, inplace
= True)
var = 'Shucked weight'
plt.scatter(x = df[var], y = df['age'])
plt.grid(True)
```



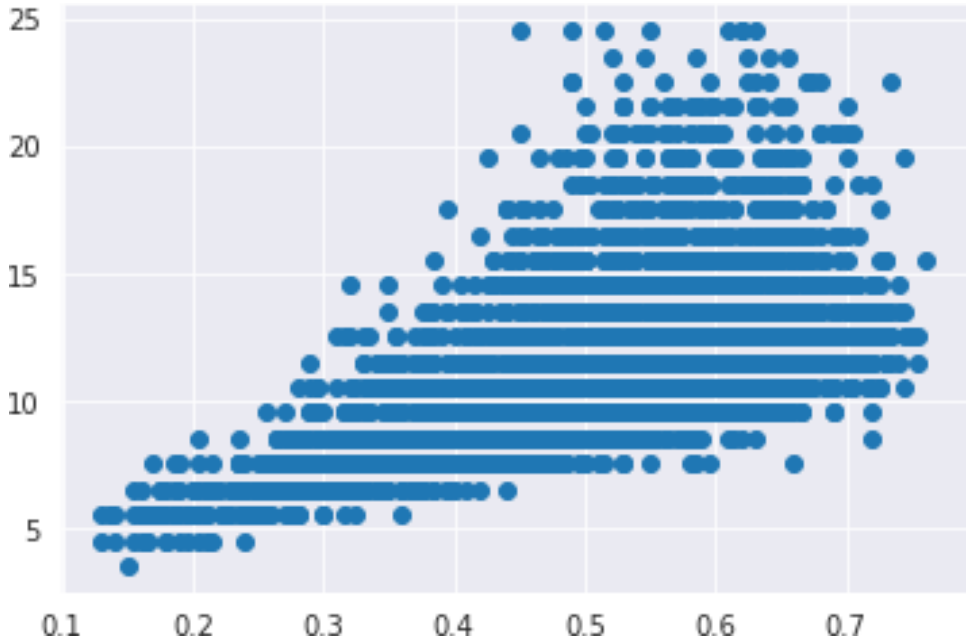

```
df.drop(df[(df['Whole weight'] >= 2.5) & (df['age'] < 25)].index,
inplace = True)
df.drop(df[(df['Whole weight']<2.5) & (df['age'] > 25)].index, inplace
= True)
var = 'Diameter'
plt.scatter(x = df[var], y = df['age'])
plt.grid(True)
```




```
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)
```




```
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
```

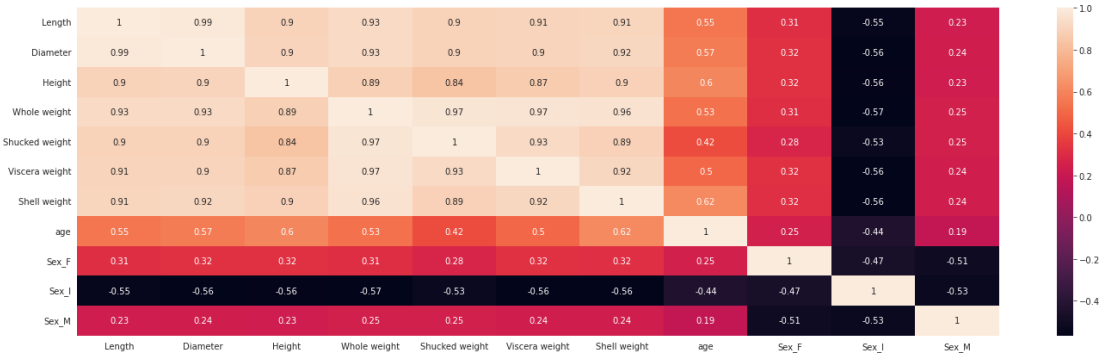
```
Index(['Length', 'Diameter', 'Height', 'Whole weight', 'Shucked
weight',
       'Viscera weight', 'Shell weight', 'age', 'Sex_F', 'Sex_I',
'Sex_M'],
      dtype='object')
```

```
categorical_features
```

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

```
plt.figure(figsize = (24,7))
sns.heatmap(df[numerical_features].corr(),annot = True)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f627e059c90>
```




```
df.columns
```

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

```
X = df.drop('age', axis = 1)
y = df['age']
```

LINEAR REGRESSION

```
from sklearn.feature_selection import SelectKBest
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import train_test_split, cross_val_score
standardScale = StandardScaler()
standardScale.fit_transform(X)

selectkBest = SelectKBest()
X_new = selectkBest.fit_transform(X, y)

X_train, X_test, y_train, y_test = train_test_split(X_new, y,
test_size = 0.25)


lr = LinearRegression()
lr.fit(X_train, y_train)

LinearRegression()

y_train_pred = lr.predict(X_train)
y_test_pred = lr.predict(X_test)

from sklearn.metrics import mean_absolute_error, mean_squared_error
s = mean_squared_error(y_train, y_train_pred)
```

```
print('Mean Squared Error of training set :%2f'%s)

p = mean_squared_error(y_test, y_test_pred)
print('Mean Squared Error of testing set :%2f'%p)
```

```
Mean Squared Error of training set :4.458678
Mean Squared Error of testing set :4.748683
```

```
from sklearn.metrics import r2_score
s = r2_score(y_train, y_train_pred)
print('R2 Score of training set:%.2f'%s)

p = r2_score(y_test, y_test_pred)
print('R2 Score of testing set:%.2f'%p)
```

```
R2 Score of training set:0.53
R2 Score of testing set:0.53
```