ASSIGNMENT III

SINDHUJA C

913119205043

importing Libraries

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
%matplotlib inline
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.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['age'] = df['Rings']+1.5
df = df.drop('Rings', axis = 1)
```

Exploratory Data Analysis

```
df.shape
```

```
(3995, 11)
```

```
df.duplicated()
```

```
0       False
1       False
2       False
3       False
4       False
        ...
4172    False
4173    False
4174    False
4175    False
4176    False
Length: 3995, dtype: bool
```

```
df.corr()
```

```
                  Length  Diameter    Height  Whole weight  Shucked
weight  \
Length          1.000000  0.986510  0.896952      0.933435
0.908398
Diameter        0.986510  1.000000  0.902958      0.933099
0.903410
Height          0.896952  0.902958  1.000000      0.888833
0.841209
```

```
Whole weight    0.933435  0.933099  0.888833      1.000000
0.971774
Shucked weight  0.908398  0.903410  0.841209      0.971774
1.000000
Viscera weight  0.906337  0.902777  0.866655      0.967086
0.930861
Shell weight    0.915167  0.922463  0.896162      0.960782
0.898596
age             0.585097  0.602680  0.625124      0.558879
0.455338
Sex_F           0.317044  0.326466  0.325117      0.322164
0.288956
Sex_I          -0.544717 -0.558042 -0.553108     -0.568049       -
0.533573
Sex_M           0.230649  0.234689  0.231134      0.248664
0.246704

                Viscera weight  Shell weight       age     Sex_F
Sex_I  \
Length                0.906337      0.915167  0.585097  0.317044 -
0.544717
Diameter              0.902777      0.922463  0.602680  0.326466 -
0.558042
Height                0.866655      0.896162  0.625124  0.325117 -
0.553108
Whole weight          0.967086      0.960782  0.558879  0.322164 -
0.568049
Shucked weight        0.930861      0.898596  0.455338  0.288956 -
0.533573
Viscera weight        1.000000      0.920124  0.535553  0.329030 -
0.564730
Shell weight          0.920124      1.000000  0.630952  0.326832 -
0.560320
age                   0.535553      0.630952  1.000000  0.262873 -
0.454388
Sex_F                 0.329030      0.326832  0.262873  1.000000 -
0.471406
Sex_I                -0.564730     -0.560320 -0.454388 -0.471406
1.000000
Sex_M                 0.238797      0.236575  0.193936 -0.498206 -
0.529816

                   Sex_M
Length          0.230649
Diameter        0.234689
Height          0.231134
Whole weight    0.248664
Shucked weight  0.246704
Viscera weight  0.238797
Shell weight    0.236575
```

```
age             0.193936
Sex_F          -0.498206
Sex_I          -0.529816
Sex_M           1.000000
```

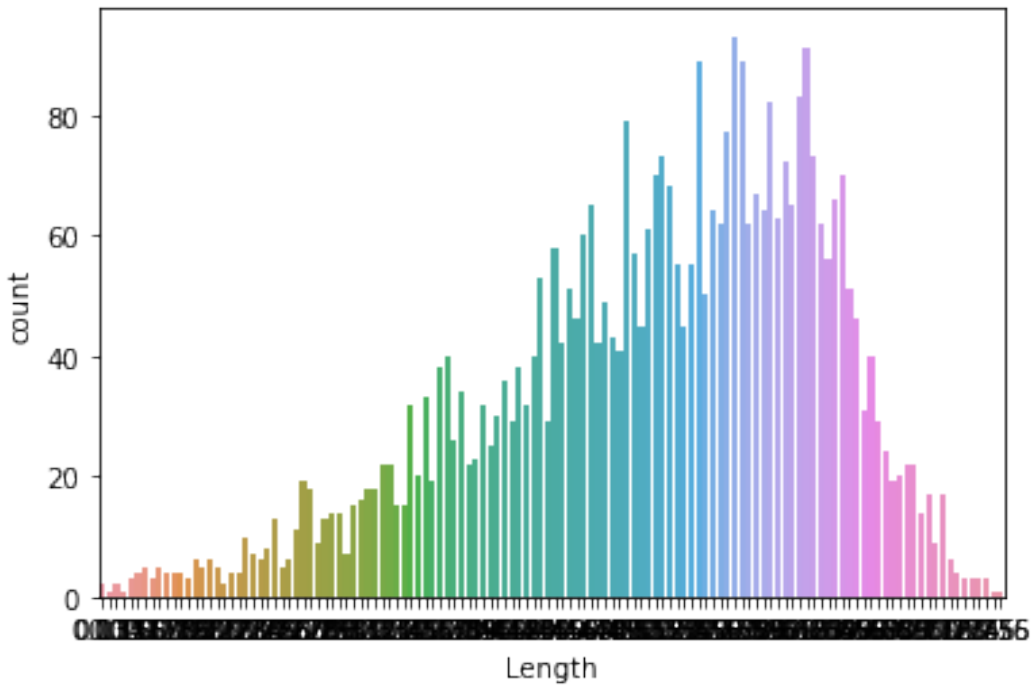
Univariate Analysis

```
#Categorical Data

#Countplot
sns.countplot(df['Length'])
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43:
FutureWarning: Pass the following variable as a keyword arg: x. 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
```

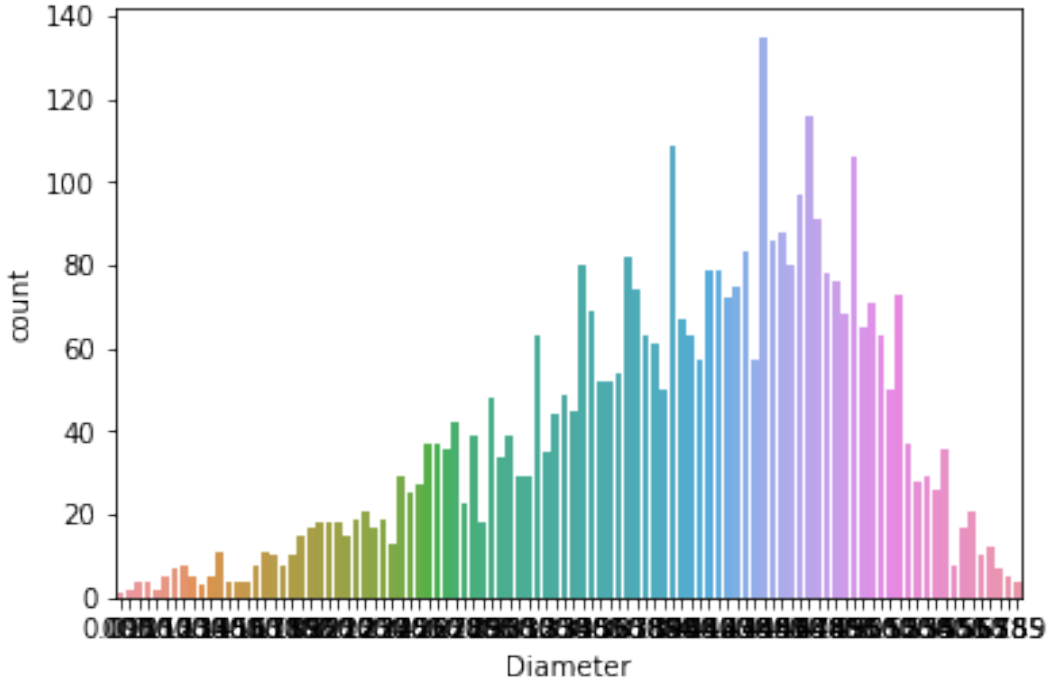
```
<matplotlib.axes._subplots.AxesSubplot at 0x7ffa76624790>
```




```
sns.countplot(df['Diameter'])
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43:
FutureWarning: Pass the following variable as a keyword arg: x. 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
```

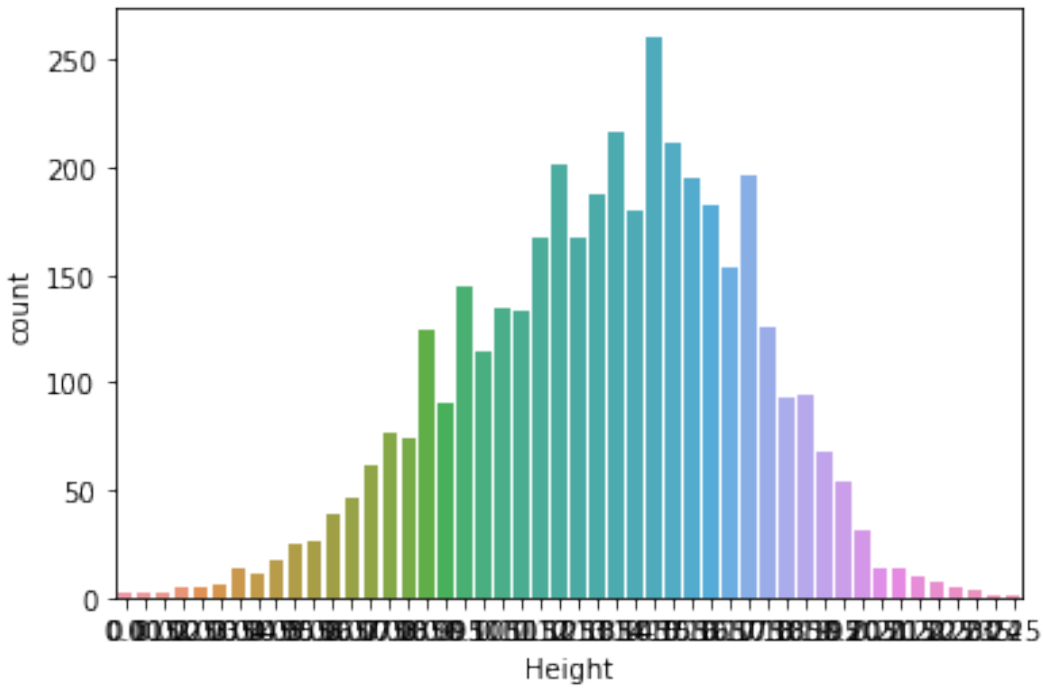
```
<matplotlib.axes._subplots.AxesSubplot at 0x7ffa7637c9d0>
```




```
sns.countplot(df['Height'])
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43:
FutureWarning: Pass the following variable as a keyword arg: x. 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
```

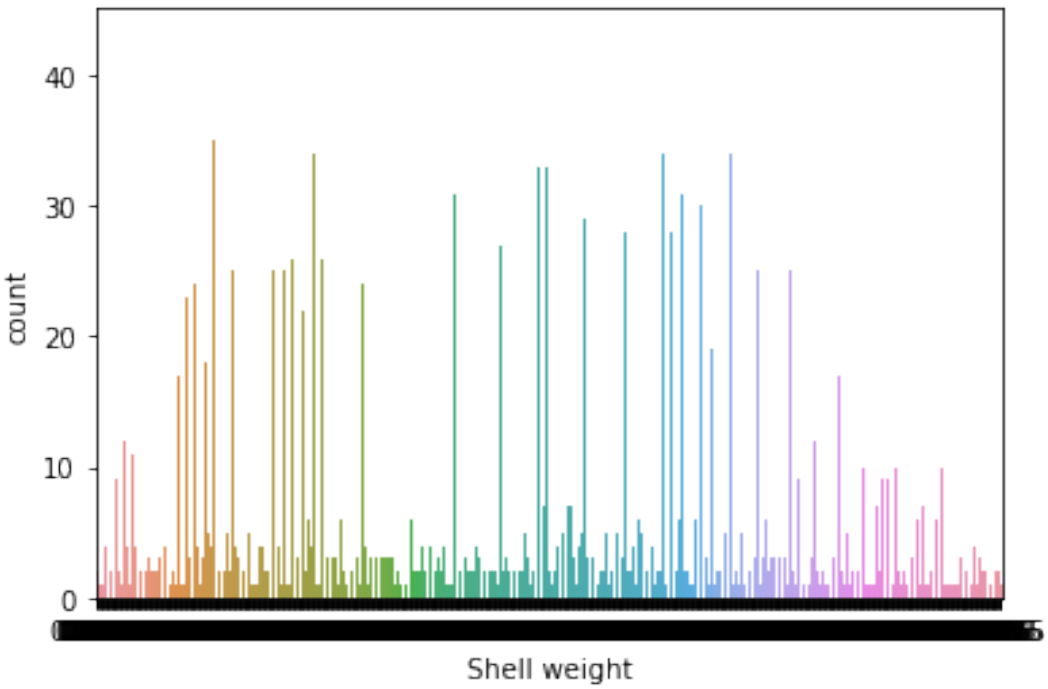
```
<matplotlib.axes._subplots.AxesSubplot at 0x7ffa7603a190>
```




```
sns.countplot(df['Shell weight'])
```

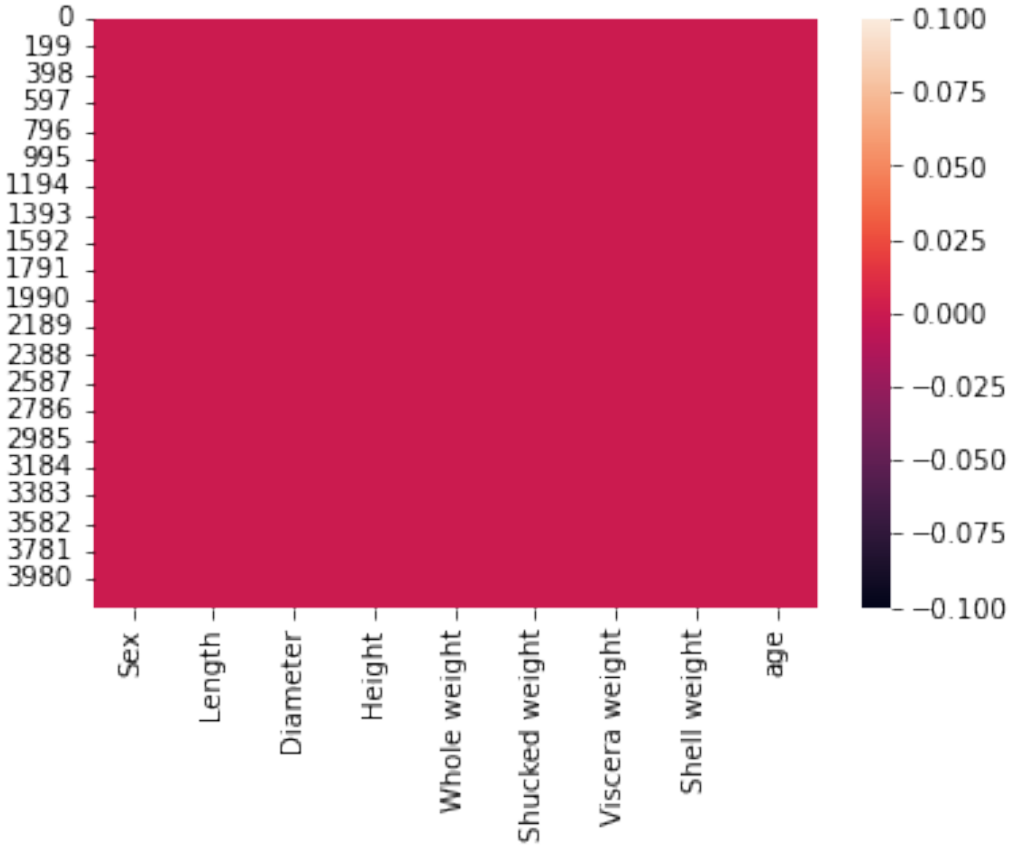
```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43:
FutureWarning: Pass the following variable as a keyword arg: x. From
version 0.12, the only valid positional argument will be `data`, and
passing other arguments without an explicit keyword will result in an
error or misinterpretation.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

```
<matplotlib.axes._subplots.AxesSubplot at 0x7ffa75eb1f90>
```



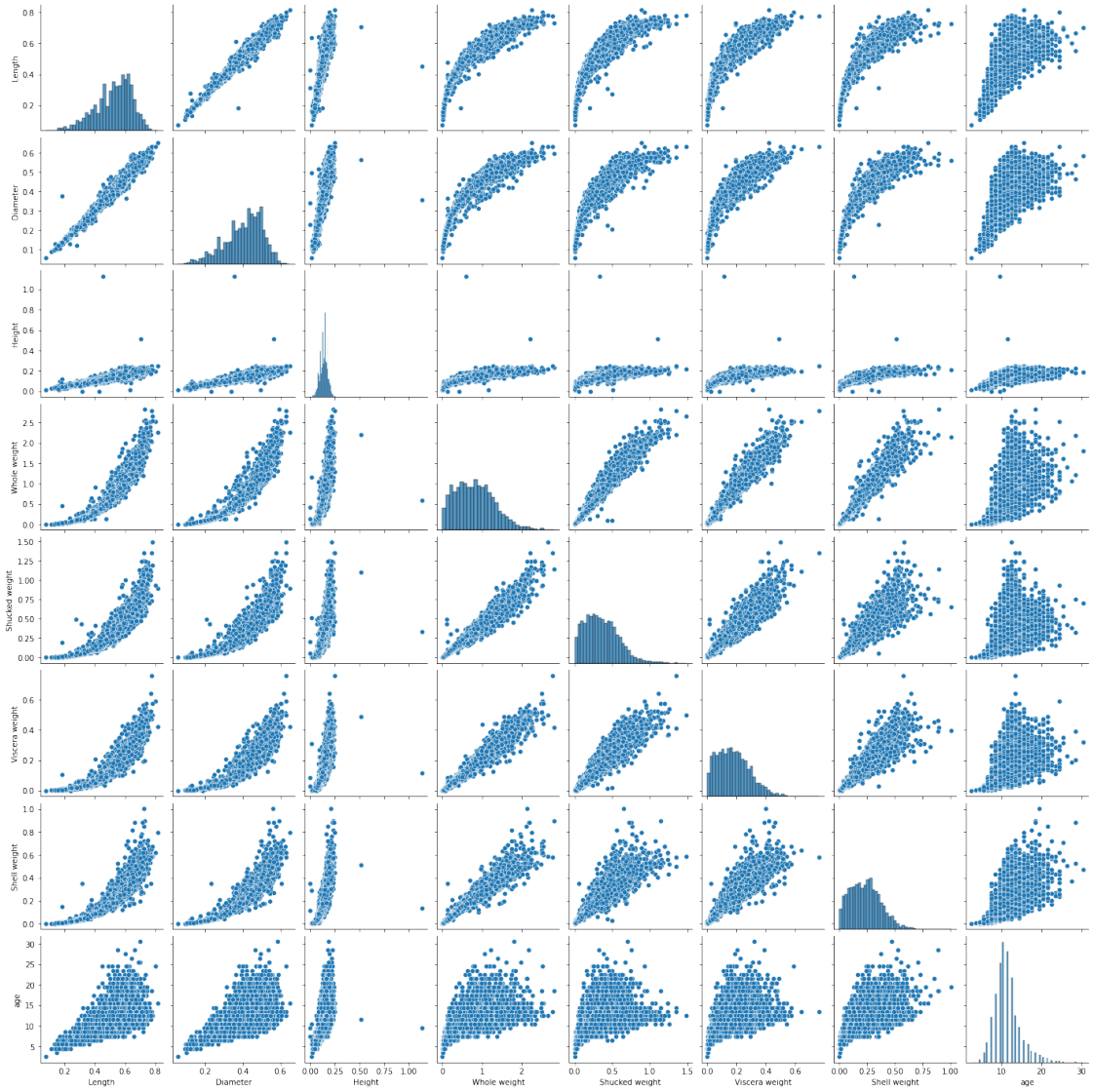

```
sns.heatmap(df.isnull())
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7ffa7e0bb250>
```



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

```
<seaborn.axisgrid.PairGrid at 0x7ffa7e0cf950>
```




```
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   float64
dtypes: float64(8), object(1)
memory usage: 293.8+ KB
```

```
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
```

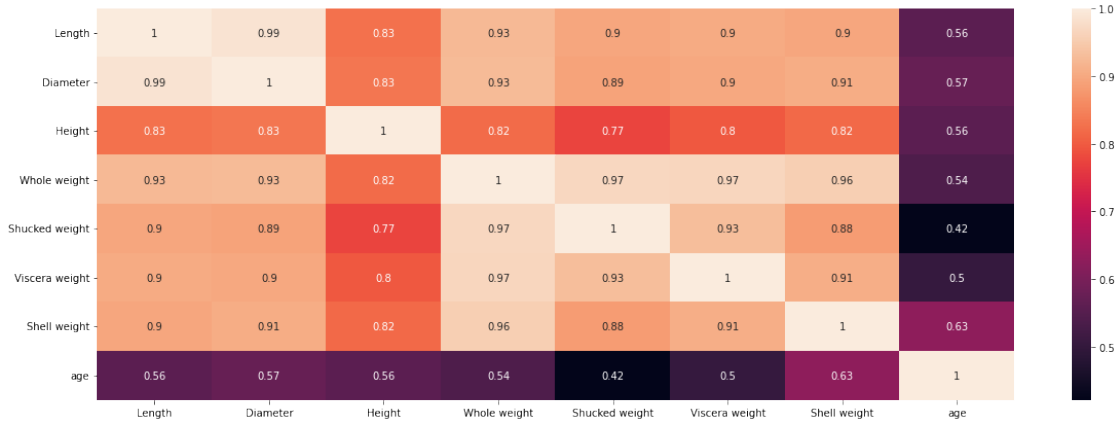
```
Index(['Length', 'Diameter', 'Height', 'Whole weight', 'Shucked
weight',
       'Viscera weight', 'Shell weight', 'age'],
      dtype='object')
```

```
categorical_features
```

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

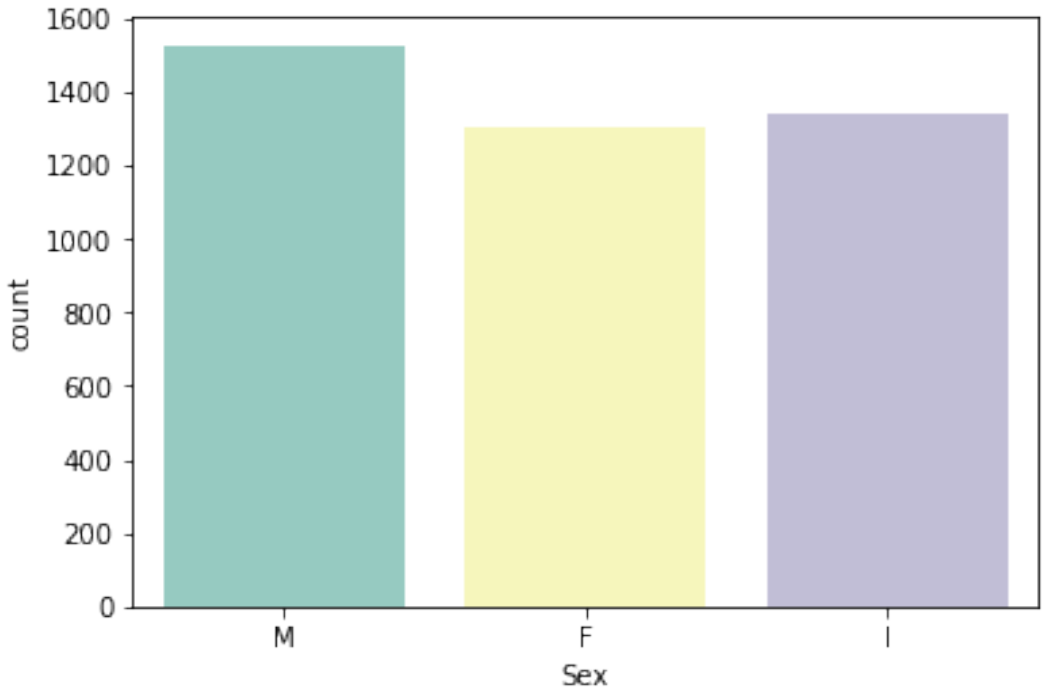
```
plt.figure(figsize = (20,7))
sns.heatmap(df[numerical_features].corr(),annot = True)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7ffa79a81c50>
```




```
sns.countplot(x = 'Sex', data = df, palette = 'Set3')
```

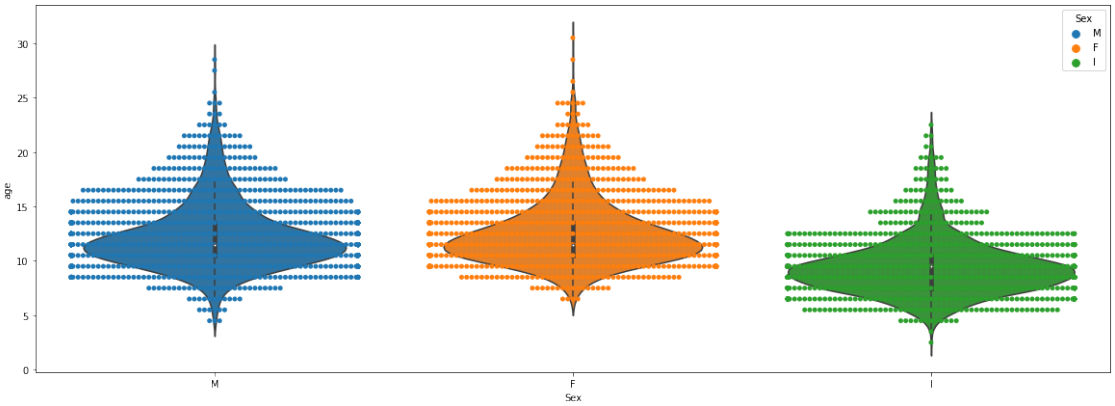
```
<matplotlib.axes._subplots.AxesSubplot at 0x7ffa77a1ca10>
```




```
plt.figure(figsize = (20,7))
sns.swarmplot(x = 'Sex', y = 'age', data = df, hue = 'Sex')
sns.violinplot(x = 'Sex', y = 'age',data = df)
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/categorical.py:1296:
UserWarning: 56.2% of the points cannot be placed; you may want to
decrease the size of the markers or use stripplot.
  warnings.warn(msg, UserWarning)
/usr/local/lib/python3.7/dist-packages/seaborn/categorical.py:1296:
UserWarning: 52.2% of the points cannot be placed; you may want to
decrease the size of the markers or use stripplot.
  warnings.warn(msg, UserWarning)
/usr/local/lib/python3.7/dist-packages/seaborn/categorical.py:1296:
UserWarning: 58.5% of the points cannot be placed; you may want to
decrease the size of the markers or use stripplot.
  warnings.warn(msg, UserWarning)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7ffa77abcf90>
```




```
#Descriptive Statistics
df.mean()

Length             0.518168
Diameter           0.402955
Height             0.136972
Whole weight       0.791814
Shucked weight     0.345022
Viscera weight     0.173504
Shell weight       0.227166
age               11.127284
Sex_F              0.307134
Sex_I              0.333917
Sex_M              0.358949
dtype: float64

df.median()

Length             0.5350
Diameter           0.4200
Height             0.1400
Whole weight       0.7745
Shucked weight     0.3265
Viscera weight     0.1650
Shell weight       0.2235
age               10.5000
Sex_F              0.0000
Sex_I              0.0000
Sex_M              0.0000
dtype: float64

df.mode()

   Length  Diameter  Height  Whole weight  Shucked weight  Viscera
weight  \
0   0.575      0.45    0.15        0.2225           0.175
0.1715
```

```
   Shell weight   age  Sex_F  Sex_I  Sex_M
0         0.275  10.5      0      0      0
```

```
#Handle Missing Value
df.isna()
```

```
      Length  Diameter  Height  Whole weight  Shucked weight  Viscera
weight  \
0      False     False   False         False           False
False
1      False     False   False         False           False
False
2      False     False   False         False           False
False
3      False     False   False         False           False
False
4      False     False   False         False           False
False
...      ...       ...     ...           ...             ...
...
4172   False     False   False         False           False
False
4173   False     False   False         False           False
False
4174   False     False   False         False           False
False
4175   False     False   False         False           False
False
4176   False     False   False         False           False
False

      Shell weight    age  Sex_F  Sex_I  Sex_M
0            False  False  False  False  False
1            False  False  False  False  False
2            False  False  False  False  False
3            False  False  False  False  False
4            False  False  False  False  False
...            ...    ...    ...    ...    ...
4172         False  False  False  False  False
4173         False  False  False  False  False
4174         False  False  False  False  False
4175         False  False  False  False  False
4176         False  False  False  False  False

[3995 rows x 11 columns]
```

```
df.isna().any()
```

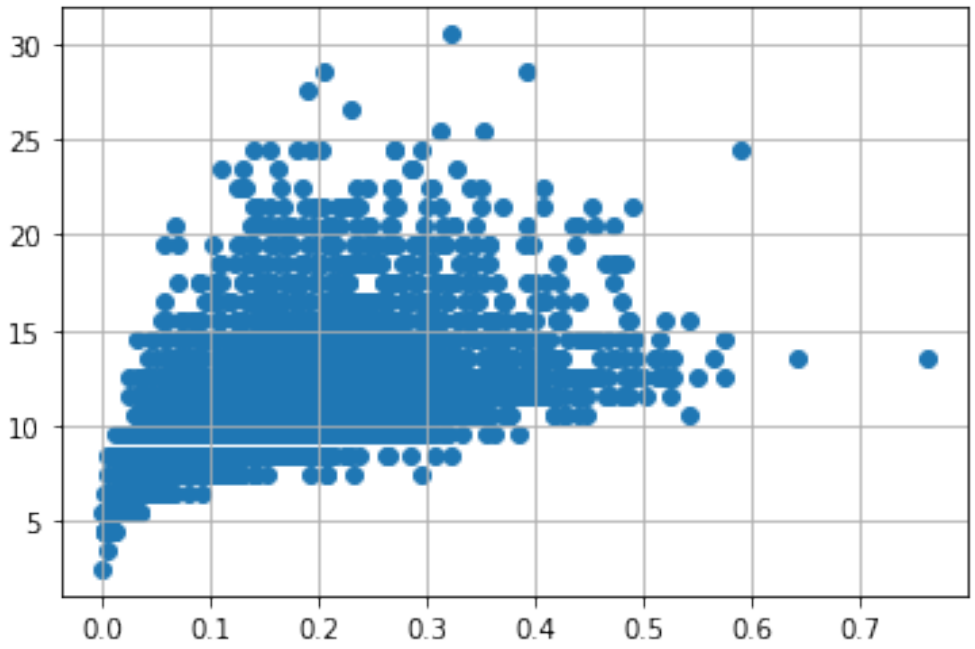
```
Length            False
Diameter          False
Height            False
```

```
Whole weight      False
Shucked weight    False
Viscera weight    False
Shell weight      False
age               False
Sex_F             False
Sex_I             False
Sex_M             False
dtype: bool
```

Outlier handling

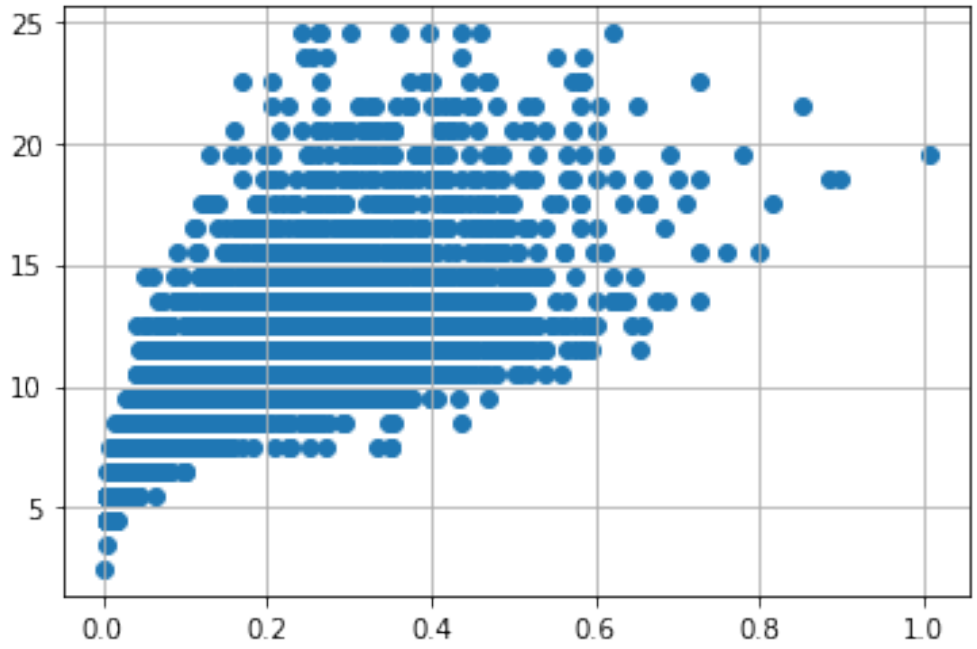
```
df = pd.get_dummies(df)
dummy_df = df

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



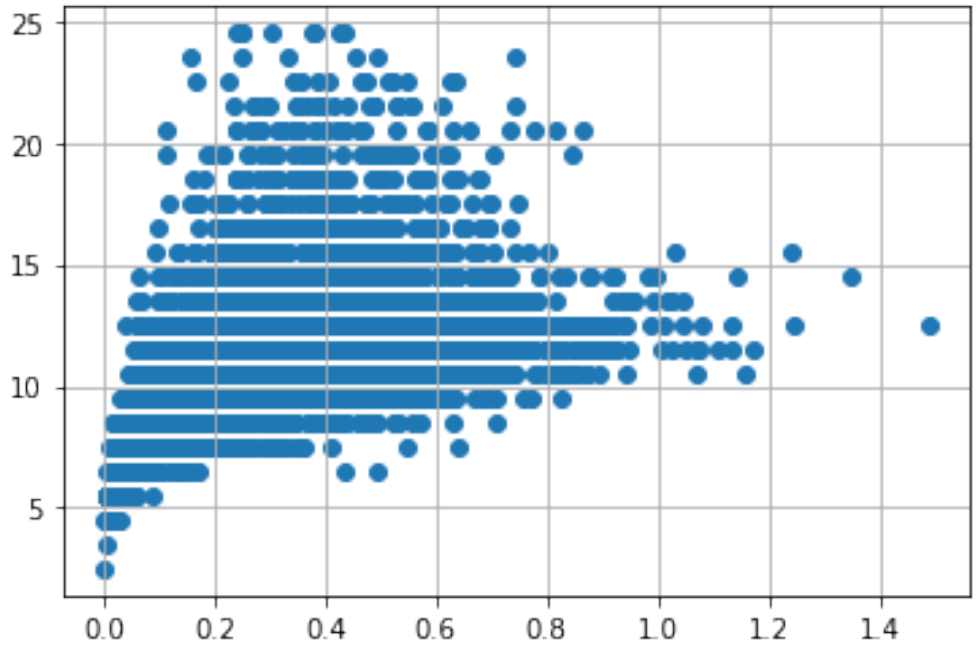

```
df.drop(df[(df['Viscera weight'] > 0.5) &(df['age'] < 20)].index,
inplace = True)
df.drop(df[(df['Viscera weight']<0.5) & (df['age'] > 25)].index,
inplace = True)

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



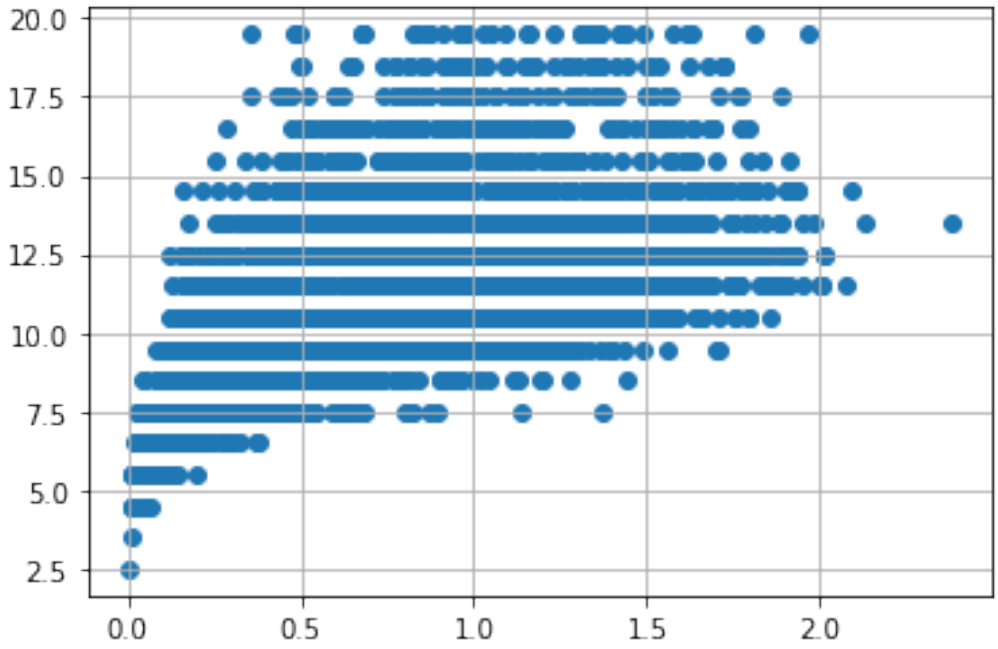

```
df.drop(df[(df['Shell weight'] > 0.6) &(df['age'] < 25)].index,
inplace = True)
df.drop(df[(df['Shell weight']<0.8) & (df['age'] > 25)].index, inplace
= True)

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



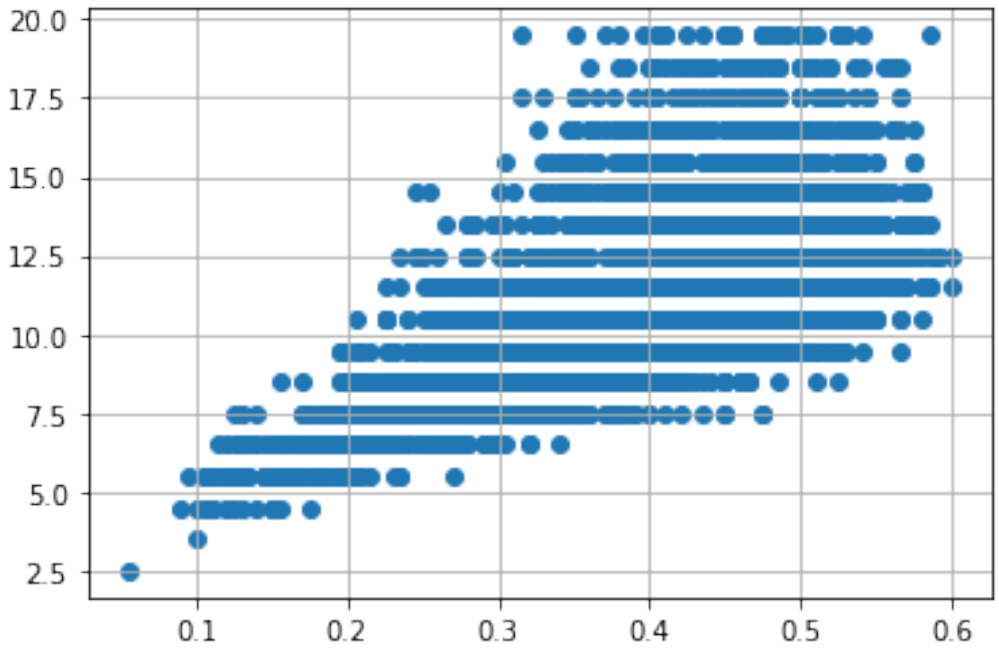
```
df.drop(df[(df['Shucked weight'] >= 1) &(df['age'] < 20)].index,
inplace = True)
df.drop(df[(df['Viscera weight']<1) & (df['age'] > 20)].index, inplace
= True)

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



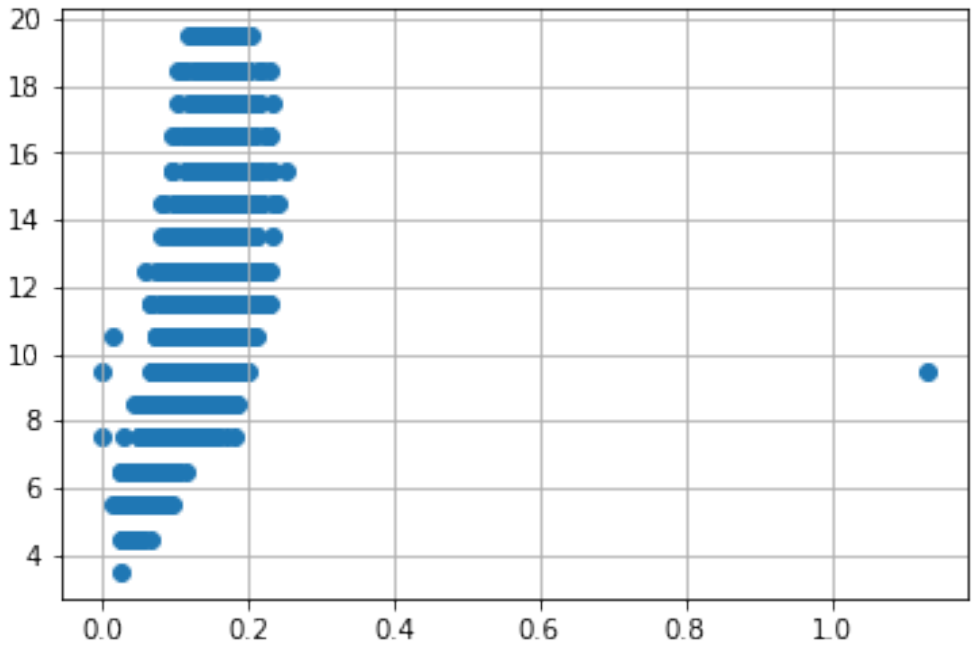

```
df.drop(df[(df['Whole weight'] >= 2.5) &(df['age'] < 25)].index,
inplace = True)
df.drop(df[(df['Whole weight']<2.5) & (df['age'] > 25)].index, inplace
= True)

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



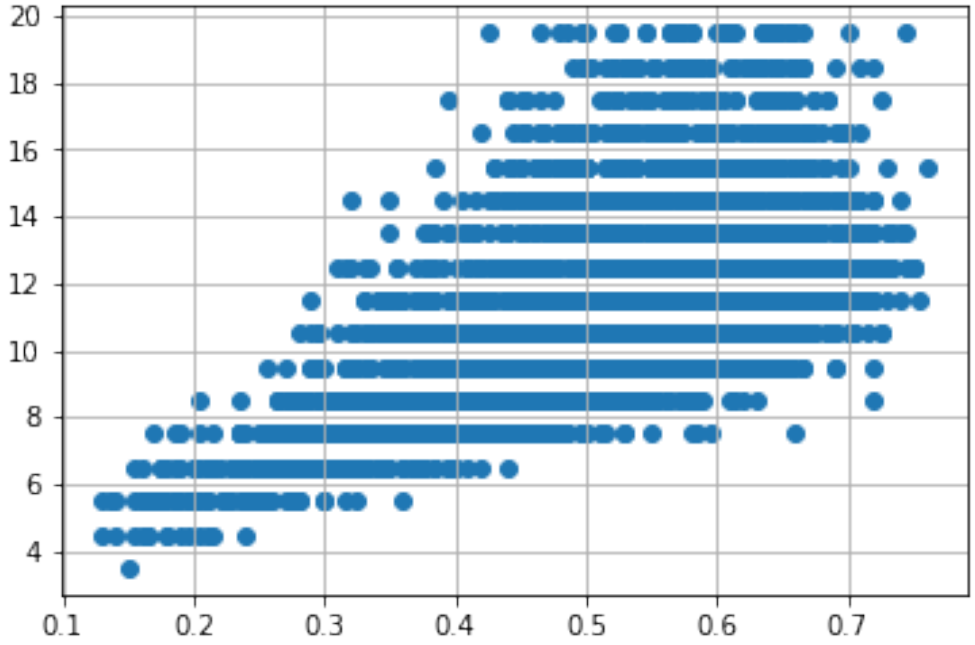

```
df.drop(df[(df['Diameter'] <0.1) &(df['age'] < 5)].index, inplace =
True)
df.drop(df[(df['Diameter']<0.6) & (df['age'] > 25)].index, inplace =
True)
df.drop(df[(df['Diameter']>=0.6) & (df['age'] < 25)].index, inplace =
True)

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)

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

KNeighbours Regression

```
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import train_test_split, cross_val_score
from sklearn.feature_selection import SelectKBest

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)

from sklearn.neighbors import KNeighborsRegressor

knn = KNeighborsRegressor(n_neighbors =4 )
knn.fit(X_train, y_train)
knn.fit(X_test, y_test)

KNeighborsRegressor(n_neighbors=4)

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

knn.score(X_train, y_train)

0.45267360390467304

knn.score(X_test, y_test)

0.6869454325742071
```

accuracy is 68%