```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from matplotlib import rcParams
```

1.Loading data

```
ds=pd.read_csv('abalone.csv')
ds
```

```
---------------------------------------------------------------------------
-----
FileNotFoundError                         Traceback (most recent call
last)
<ipython-input-5-819190645815> in <module>
----> 1 ds=pd.read_csv('abalone.csv')
      2 ds

/usr/local/lib/python3.7/dist-packages/pandas/util/_decorators.py in
wrapper(*args, **kwargs)
    309                     stacklevel=stacklevel,
    310                 )
--> 311             return func(*args, **kwargs)
    312 
    313         return wrapper

/usr/local/lib/python3.7/dist-packages/pandas/io/parsers/readers.py in
read_csv(filepath_or_buffer, sep, delimiter, header, names, index_col,
usecols, squeeze, prefix, mangle_dupe_cols, dtype, engine, converters,
true_values, false_values, skipinitialspace, skiprows, skipfooter,
nrows, na_values, keep_default_na, na_filter, verbose,
skip_blank_lines, parse_dates, infer_datetime_format, keep_date_col,
date_parser, dayfirst, cache_dates, iterator, chunksize, compression,
thousands, decimal, lineterminator, quotechar, quoting, doublequote,
escapechar, comment, encoding, encoding_errors, dialect,
error_bad_lines, warn_bad_lines, on_bad_lines, delim_whitespace,
low_memory, memory_map, float_precision, storage_options)
    584     kwds.update(kwds_defaults)
    585 
--> 586     return _read(filepath_or_buffer, kwds)
    587 
    588 

/usr/local/lib/python3.7/dist-packages/pandas/io/parsers/readers.py in
_read(filepath_or_buffer, kwds)
    480 
    481     # Create the parser.
--> 482     parser = TextFileReader(filepath_or_buffer, **kwds)
    483 
    484     if chunksize or iterator:
```

```
/usr/local/lib/python3.7/dist-packages/pandas/io/parsers/readers.py in
__init__(self, f, engine, **kwds)
    809             self.options["has_index_names"] =
kwds["has_index_names"]
    810 
--> 811         self._engine = self._make_engine(self.engine)
    812 
    813     def close(self):

/usr/local/lib/python3.7/dist-packages/pandas/io/parsers/readers.py in
_make_engine(self, engine)
   1038             )
   1039         # error: Too many arguments for "ParserBase"
-> 1040         return mapping[engine](self.f, **self.options)  #
type: ignore[call-arg]
   1041 
   1042     def _failover_to_python(self):

/usr/local/lib/python3.7/dist-packages/pandas/io/parsers/c_parser_wrap
per.py in __init__(self, src, **kwds)
     49 
     50         # open handles
---> 51         self._open_handles(src, kwds)
     52         assert self.handles is not None
     53 

/usr/local/lib/python3.7/dist-packages/pandas/io/parsers/base_parser.p
y in _open_handles(self, src, kwds)
    227             memory_map=kwds.get("memory_map", False),
    228             storage_options=kwds.get("storage_options", None),
--> 229             errors=kwds.get("encoding_errors", "strict"),
    230         )
    231 

/usr/local/lib/python3.7/dist-packages/pandas/io/common.py in
get_handle(path_or_buf, mode, encoding, compression, memory_map,
is_text, errors, storage_options)
    705                 encoding=ioargs.encoding,
    706                 errors=errors,
--> 707                 newline="",
    708             )
    709         else:

FileNotFoundError: [Errno 2] No such file or directory: 'abalone.csv'
```

```
ds.Rings=ds.Rings.add(1.5)
ds
```

```
     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   16.5  
1             0.0485        0.0700    8.5  
2             0.1415        0.2100   10.5  
3             0.1140        0.1550   11.5  
4             0.0395        0.0550    8.5  
...              ...           ...    ...  
4172          0.2390        0.2490   12.5  
4173          0.2145        0.2605   11.5  
4174          0.2875        0.3080   10.5  
4175          0.2610        0.2960   11.5  
4176          0.3765        0.4950   13.5  

[4177 rows x 9 columns]
```

```
ds=ds.rename(columns={'Rings':'Age'})
ds
```

```
     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   Age  
0             0.1010        0.1500  16.5  
1             0.0485        0.0700   8.5  
2             0.1415        0.2100  10.5  
3             0.1140        0.1550  11.5  
4             0.0395        0.0550   8.5  
```

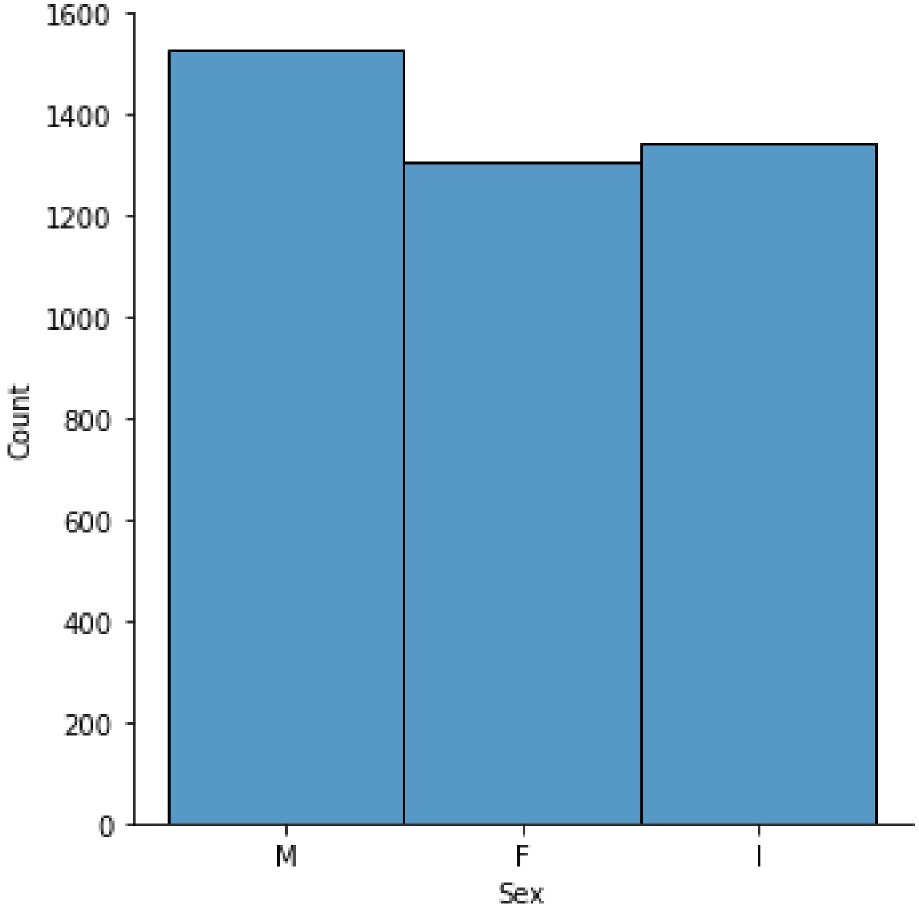
```
...             ...           ...   ...
4172         0.2390        0.2490  12.5
4173         0.2145        0.2605  11.5
4174         0.2875        0.3080  10.5
4175         0.2610        0.2960  11.5
4176         0.3765        0.4950  13.5

[4177 rows x 9 columns]
```

2.Visualisation

Uni-varient analysis

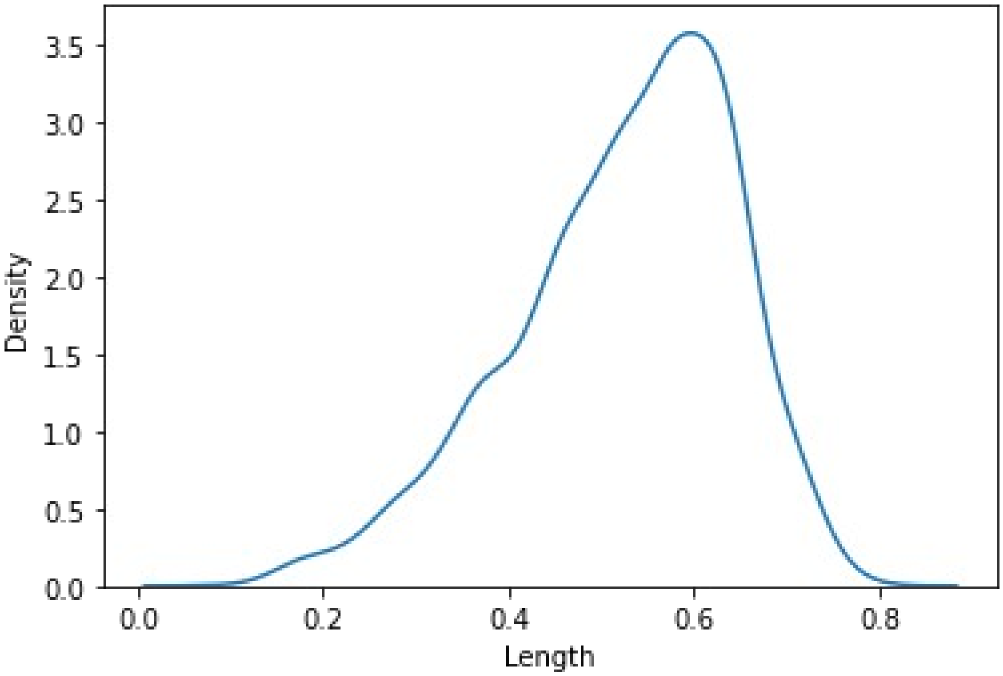
```
 sns.displot(ds.Sex)
```

```
<seaborn.axisgrid.FacetGrid at 0x20e7d309580>
```



```
sns.kdeplot(ds.Length)
```

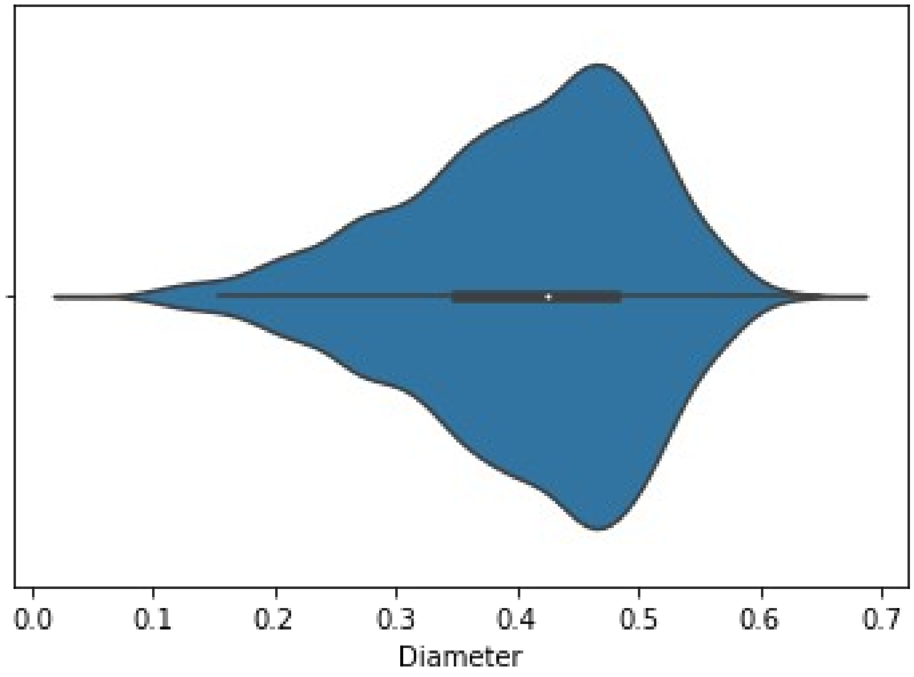
```
<AxesSubplot:xlabel='Length', ylabel='Density'>
```



```
sns.violinplot(ds.Diameter)
```

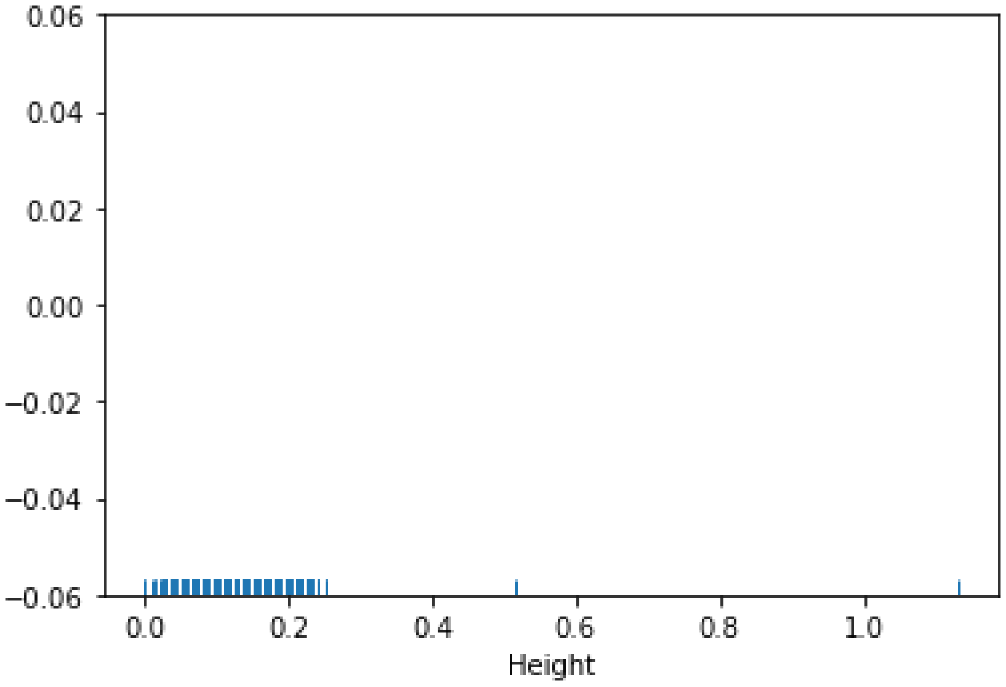
```
C:\Users\prave\anaconda3\lib\site-packages\seaborn\_decorators.py:36:
FutureWarning: Pass the following variable as a keyword arg: x. From
version 0.12, the only valid positional argument will be `data`, and
passing other arguments without an explicit keyword will result in an
error or misinterpretation.
  warnings.warn(
```

```
<AxesSubplot:xlabel='Diameter'>
```



```
sns.rugplot(ds.Height)
```

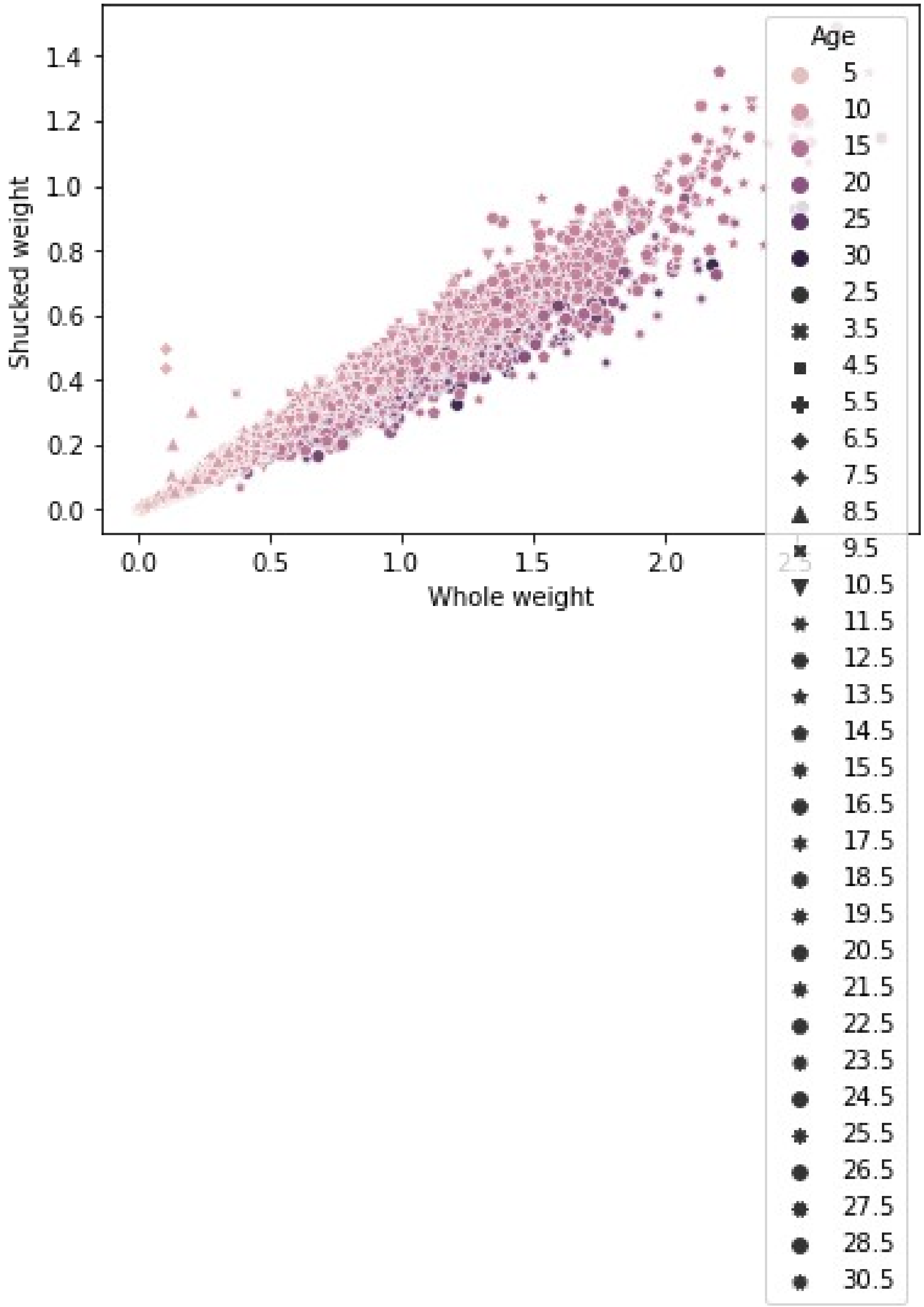
```
<AxesSubplot:xlabel='Height'>
```



Bi-varient analysis

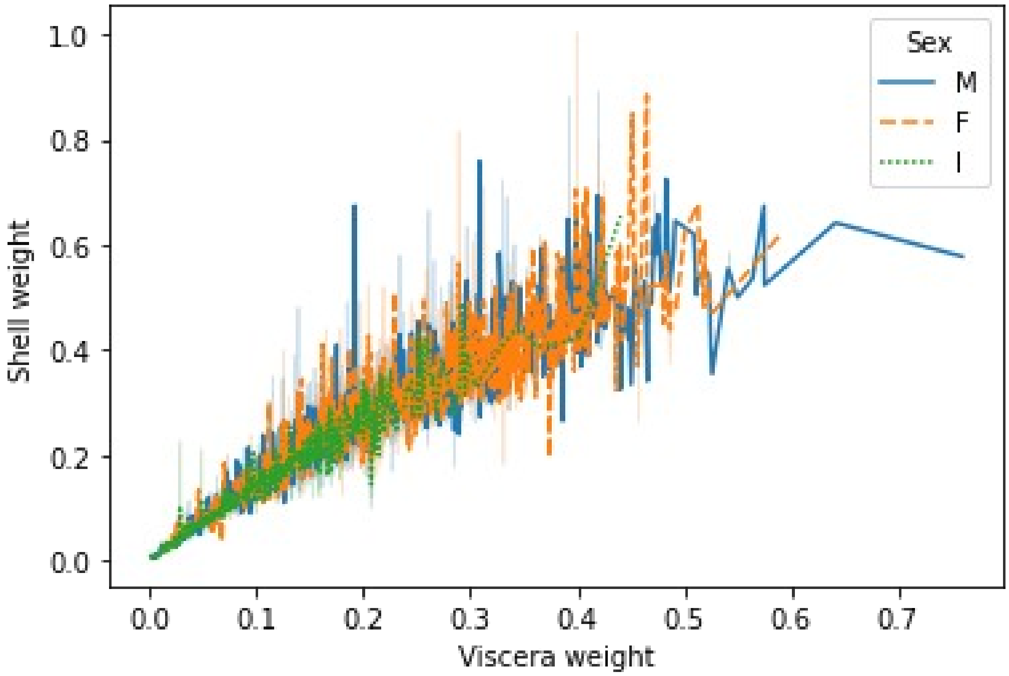
```
sns.scatterplot(x='Whole weight',y='Shucked
weight',data=ds,hue='Age',style='Age')
```

```
<AxesSubplot:xlabel='Whole weight', ylabel='Shucked weight'>
```



```
sns.lineplot(x='Viscera weight',y='Shell
weight',data=ds,hue='Sex',style='Sex')
```

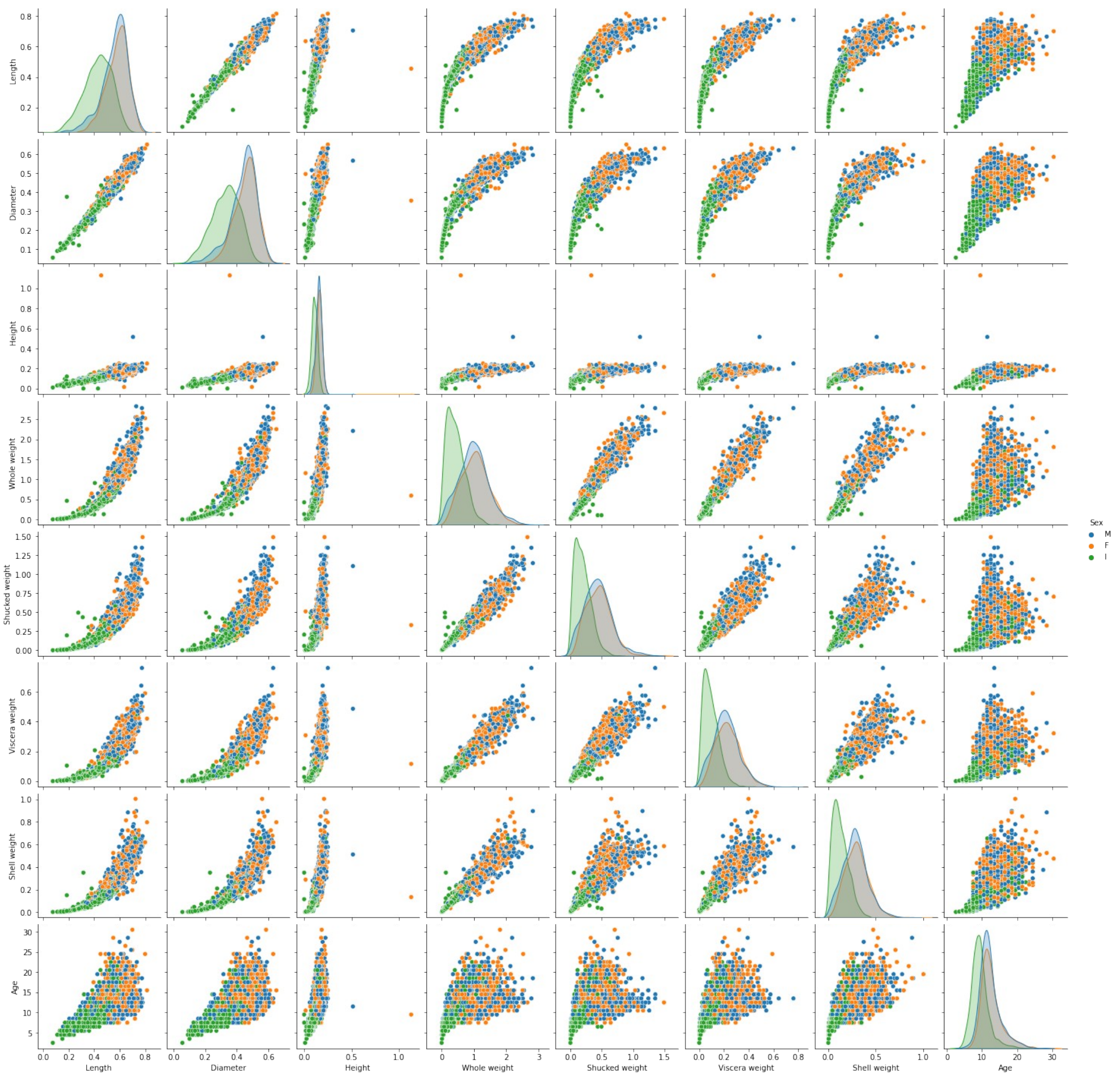
```
<AxesSubplot:xlabel='Viscera weight', ylabel='Shell weight'>
```



Multi-varient analysis

```
sns.pairplot(data=ds,hue='Sex')
```

```
<seaborn.axisgrid.PairGrid at 0x20e7e4e14f0>
```



3.Descriptive statistics

```
ds.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
```

4.Handling missing values

```
ds.isnull().any()
```

```
Sex                  False
Length               False
Diameter             False
Height               False
Whole weight         False
Shucked weight       False
Viscera weight       False
Shell weight         False
Age                  False
dtype: bool
```

5.Outlier checking and replace

```
sns.boxplot(ds.Length)
```

```
----------------------------------------------------------------------
-----
NameError                                 Traceback (most recent call
last)
<ipython-input-2-afbbcbd6920a> in <module>
----> 1 sns.boxplot(ds.Length)

NameError: name 'sns' is not defined
```

```
sns.boxplot(ds.Diameter)
```

```
----------------------------------------------------------------------
-----
NameError                                 Traceback (most recent call
last)
<ipython-input-3-5337434110d0> in <module>
----> 1 sns.boxplot(ds.Diameter)

NameError: name 'sns' is not defined
```

```
sns.boxplot(ds.Height)
```

```
sns.boxplot(ds.Age)
```

6.Categorical columns and perform encoding.

```
from sklearn.preprocessing import LabelEncoder
le=LabelEncoder()
ds.Sex=le.fit_transform(ds.Sex)
ds_main=ds
ds_main
```

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

      Viscera weight  Shell weight   Age  
0             0.1010        0.1500  16.5  
1             0.0485        0.0700   8.5  
2             0.1415        0.2100  10.5  
3             0.1140        0.1550  11.5  
4             0.0395        0.0550   8.5  
...              ...           ...   ...  
4172          0.2390        0.2490  12.5  
4173          0.2145        0.2605  11.5  
4174          0.2875        0.3080  10.5  
4175          0.2610        0.2960  11.5  
4176          0.3765        0.4950  13.5  

[4177 rows x 9 columns]
```

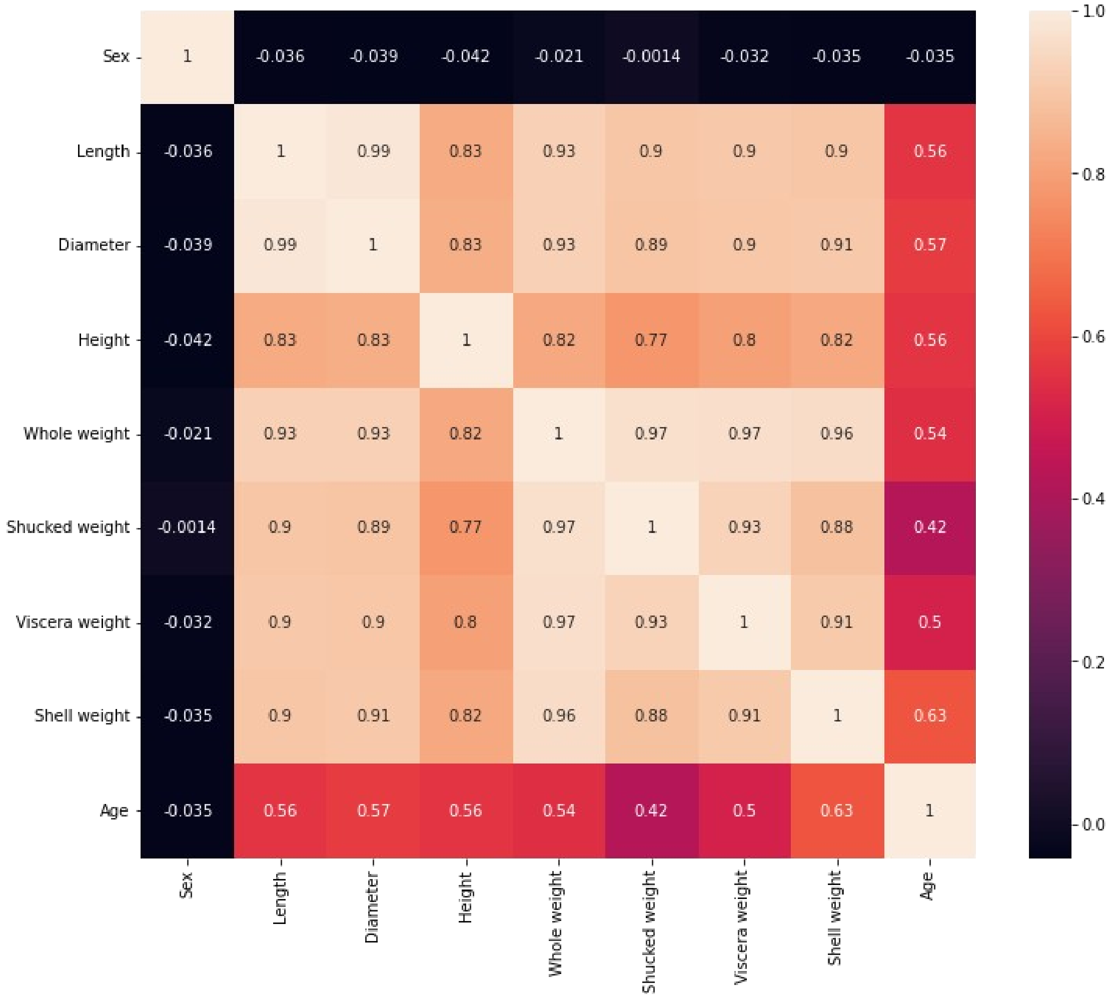
```
ds_main.corr()
```

```
                     Sex    Length  Diameter    Height  Whole 
weight  \
Sex             1.000000 -0.036066 -0.038874 -0.042077     -0.021391   

Length         -0.036066  1.000000  0.986812  0.827554       0.925261   

Diameter       -0.038874  0.986812  1.000000  0.833684       0.925452   

Height         -0.042077  0.827554  0.833684  1.000000       0.819221   
```

```
Whole weight   -0.021391  0.925261  0.925452  0.819221      1.000000
Shucked weight -0.001373  0.897914  0.893162  0.774972      0.969405
Viscera weight -0.032067  0.903018  0.899724  0.798319      0.966375
Shell weight   -0.034854  0.897706  0.905330  0.817338      0.955355
Age            -0.034627  0.556720  0.574660  0.557467      0.540390

                Shucked weight  Viscera weight  Shell weight       Age
Sex                  -0.001373       -0.032067     -0.034854 -0.034627
Length                0.897914        0.903018      0.897706  0.556720
Diameter              0.893162        0.899724      0.905330  0.574660
Height                0.774972        0.798319      0.817338  0.557467
Whole weight          0.969405        0.966375      0.955355  0.540390
Shucked weight        1.000000        0.931961      0.882617  0.420884
Viscera weight        0.931961        1.000000      0.907656  0.503819
Shell weight          0.882617        0.907656      1.000000  0.627574
Age                   0.420884        0.503819      0.627574  1.000000
```

```
plt.figure(figsize=(12,10))
sns.heatmap(ds_main.corr(),annot=True)
```

```
<AxesSubplot:>
```



```
ds_main.corr().Age.sort_values(ascending=False)
```

```
Age               1.000000
Shell weight      0.627574
Diameter          0.574660
Height            0.557467
Length            0.556720
Whole weight      0.540390
Viscera weight    0.503819
Shucked weight    0.420884
Sex              -0.034627
Name: Age, dtype: float64
```

7.Depended and independent value split

```
y=ds_main['Age']
y
```

```
0       16.5
1        8.5
2       10.5
```

```
3       11.5
4        8.5
        ...
4172    12.5
4173    11.5
4174    10.5
4175    11.5
4176    13.5
Name: Age, Length: 4177, dtype: float64
```

```
x=ds_main.drop(columns=['Age'],axis=1)
x.head()
```

```
   Sex  Length  Diameter  Height  Whole weight  Shucked weight  \
0    2   0.455     0.365   0.095        0.5140          0.2245   
1    2   0.350     0.265   0.090        0.2255          0.0995   
2    0   0.530     0.420   0.135        0.6770          0.2565   
3    2   0.440     0.365   0.125        0.5160          0.2155   
4    1   0.330     0.255   0.080        0.2050          0.0895   

   Viscera weight  Shell weight  
0          0.1010         0.150  
1          0.0485         0.070  
2          0.1415         0.210  
3          0.1140         0.155  
4          0.0395         0.055  
```

8.Scaling independent variable

```
from sklearn.preprocessing import scale
```

```
x_scaled=pd.DataFrame(scale(x),columns=x.columns)
x_scaled
```

```
           Sex    Length  Diameter    Height  Whole weight  Shucked 
weight  \
0     1.151980 -0.574558 -0.432149 -1.064424     -0.641898       -
0.607685   
1     1.151980 -1.448986 -1.439929 -1.183978     -1.230277       -
1.170910   
2    -1.280690  0.050033  0.122130 -0.107991     -0.309469       -
0.463500   
3     1.151980 -0.699476 -0.432149 -0.347099     -0.637819       -
0.648238   
4    -0.064355 -1.615544 -1.540707 -1.423087     -1.272086       -
1.215968   
...        ...       ...       ...       ...           ...            
...   
4172 -1.280690  0.341509  0.424464  0.609334      0.118813        
0.047908   
4173  1.151980  0.549706  0.323686 -0.107991      0.279929        
0.358808   
```

```
4174  1.151980  0.632985  0.676409  1.565767      0.708212
0.748559
4175 -1.280690  0.841182  0.777187  0.250672      0.541998
0.773341
4176  1.151980  1.549052  1.482634  1.326659      2.283681
2.640993

      Viscera weight  Shell weight
0          -0.726212     -0.638217
1          -1.205221     -1.212987
2          -0.356690     -0.207139
3          -0.607600     -0.602294
4          -1.287337     -1.320757
...              ...           ...
4172        0.532900      0.073062
4173        0.309362      0.155685
4174        0.975413      0.496955
4175        0.733627      0.410739
4176        1.787449      1.840481

[4177 rows x 8 columns]
```

9.Train and test split

```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x_scaled,y,test_size=0.
3,random_state=0)
```

```
x_train.shape
```

```
(2923, 8)
```

```
x_test.shape
```

```
(1254, 8)
```

```
y_train.shape
```

```
(2923,)
```

```
y_test.shape
```

```
(1254,)
```

10.Build ,Train and Test the model

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

```
LinearRegression()
```

```
y_predict=model.predict(x_test)
```

```
from sklearn.metrics import r2_score
test_score=r2_score(y_test,y_predict)
test_score
```

```
0.5140139913856603
```

```
train_score=model.score(x_train,y_train)
train_score
```

```
0.5327839192584529
```

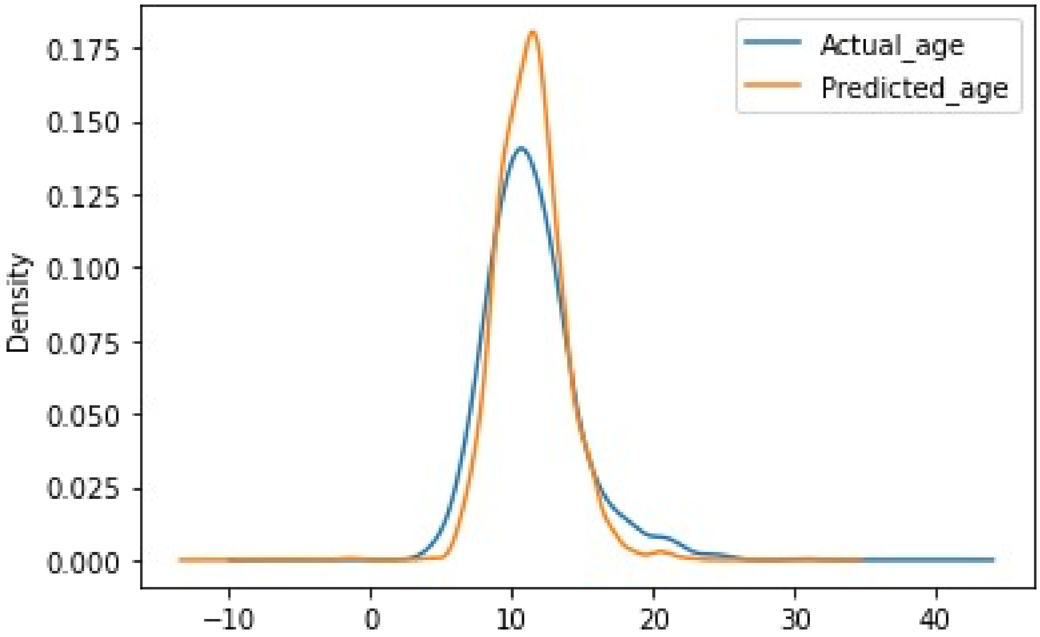
```
from sklearn import metrics

print('MAE:', metrics.mean_absolute_error(y_test,y_predict))
print('MSE:', metrics.mean_squared_error(y_test,y_predict))
print('RMSE:', np.sqrt(metrics.mean_squared_error(y_test,y_predict)))
```

```
MAE: 1.6138483794094411
MSE: 5.123755375518969
RMSE: 2.2635713762810683
```

```
pred_age=pd.DataFrame({'Actual_age':y_test,'Predicted_age':y_predict})
pred_age
```

```
      Actual_age  Predicted_age
668         14.5      14.616408
1580         9.5      11.156911
3784        12.5      11.853510
463          6.5       7.136487
2615        13.5      12.174365
...          ...            ...
1052        13.5      15.246148
3439         9.5      10.030551
1174        10.5      10.505741
2210        19.5      20.331258
2408        16.5      13.269233

[1254 rows x 2 columns]
```

```
pred_age=pre_age.plot.kde()
```


Actual_age
Predicted_age
Density
0.175
0.150
0.125
0.100
0.075
0.050
0.025
0.000
−10
0
10
20
30
40