

Import library

```
In [1]: import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

import warnings
warnings.filterwarnings("ignore")
```

Load the dataset

```
In [2]: data = pd.read_csv("abalone.csv")
```

Visualising 1st 5 rows

```
In [3]: data.head()
```

```
Out[3]:
```

	Sex	Length	Diameter	Height	Whole weight	Shucked weight	Viscera weight	Shell weight	Rings
0	M	0.455	0.365	0.095	0.5140	0.2245	0.1010	0.150	15
1	M	0.350	0.265	0.090	0.2255	0.0995	0.0485	0.070	7
2	F	0.530	0.420	0.135	0.6770	0.2565	0.1415	0.210	9
3	M	0.440	0.365	0.125	0.5160	0.2155	0.1140	0.155	10
4	I	0.330	0.255	0.080	0.2050	0.0895	0.0395	0.055	7

```
In [4]: data.tail()
```

```
Out[4]:
```

	Sex	Length	Diameter	Height	Whole weight	Shucked weight	Viscera weight	Shell weight	Rings
4172	F	0.565	0.450	0.165	0.8870	0.3700	0.2390	0.2490	11
4173	M	0.590	0.440	0.135	0.9660	0.4390	0.2145	0.2605	10
4174	M	0.600	0.475	0.205	1.1760	0.5255	0.2875	0.3080	9
4175	F	0.625	0.485	0.150	1.0945	0.5310	0.2610	0.2960	10
4176	M	0.710	0.555	0.195	1.9485	0.9455	0.3765	0.4950	12

```
In [5]: data.shape
```

```
Out[5]: (4177, 9)
```

```
In [6]: data.info
```

```
Out[6]:
```

```
In [7]: data.nunique()
```

```
Out[7]: Sex          3
Length        134
Diameter       111
Height         51
Whole weight   2429
Shucked weight 1515
Viscera weight 880
Shell weight   926
Rings          28
dtype: int64
```

Duplicate

```
In [8]: data.duplicated()

Out[8]: 0      False
        1      False
        2      False
        3      False
        4      False
        ...
        4172   False
        4173   False
        4174   False
        4175   False
        4176   False
        Length: 4177, dtype: bool

In [ ]: data.duplicated().sum()

Out[ ]: 0
```

Columns of the dataset

```
In [9]: data.columns

Out[9]: Index(['Sex', 'Length', 'Diameter', 'Height', 'Whole weight', 'Shucked weight',
              'Viscera weight', 'Shell weight', 'Rings'],
              dtype='object')
```

Missing values

```
In [10]: data.isna()

Out[10]:
```

	Sex	Length	Diameter	Height	Whole weight	Shucked weight	Viscera weight	Shell weight	Rings
0	False	False	False	False	False	False	False	False	False
1	False	False	False	False	False	False	False	False	False
2	False	False	False	False	False	False	False	False	False
3	False	False	False	False	False	False	False	False	False
4	False	False	False	False	False	False	False	False	False
...
4172	False	False	False	False	False	False	False	False	False
4173	False	False	False	False	False	False	False	False	False
4174	False	False	False	False	False	False	False	False	False
4175	False	False	False	False	False	False	False	False	False
4176	False	False	False	False	False	False	False	False	False

4177 rows × 9 columns

```
In [11]: data.isnull().any()
```

```
Out[11]: Sex          False
Length         False
Diameter       False
Height         False
Whole weight   False
Shucked weight False
Viscera weight False
Shell weight   False
Rings          False
dtype: bool
```

Descriptive statistics

```
In [12]: data.describe()
```

```
Out[12]:
```

	Length	Diameter	Height	Whole weight	Shucked weight	Viscera weight	Shell weight	Rings
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	9.933684
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	1.000000
25%	0.450000	0.350000	0.115000	0.441500	0.186000	0.093500	0.130000	8.000000
50%	0.545000	0.425000	0.140000	0.799500	0.336000	0.171000	0.234000	9.000000
75%	0.615000	0.480000	0.165000	1.153000	0.502000	0.253000	0.329000	11.000000
max	0.815000	0.650000	1.130000	2.825500	1.488000	0.760000	1.005000	29.000000

```
In [13]: data.head()
```

```
Out[13]:
```

	Sex	Length	Diameter	Height	Whole weight	Shucked weight	Viscera weight	Shell weight	Rings
0	M	0.455	0.365	0.095	0.5140	0.2245	0.1010	0.150	15
1	M	0.350	0.265	0.090	0.2255	0.0995	0.0485	0.070	7
2	F	0.530	0.420	0.135	0.6770	0.2565	0.1415	0.210	9
3	M	0.440	0.365	0.125	0.5160	0.2155	0.1140	0.155	10
4	I	0.330	0.255	0.080	0.2050	0.0895	0.0395	0.055	7

```
In [14]: data["Sex"].unique()
```

```
Out[14]: array(['M', 'F', 'I'], dtype=object)
```

Feature mapping

```
In [15]: data["Sex"].replace({"M":1,"F":0,"I":2},inplace=True)
```

```
In [16]: data.head()
```

Out[16]:

	Sex	Length	Diameter	Height	Whole weight	Shucked weight	Viscera weight	Shell weight	Rings
0	1	0.455	0.365	0.095	0.5140	0.2245	0.1010	0.150	15
1	1	0.350	0.265	0.090	0.2255	0.0995	0.0485	0.070	7
2	0	0.530	0.420	0.135	0.6770	0.2565	0.1415	0.210	9
3	1	0.440	0.365	0.125	0.5160	0.2155	0.1140	0.155	10
4	2	0.330	0.255	0.080	0.2050	0.0895	0.0395	0.055	7

In [17]:

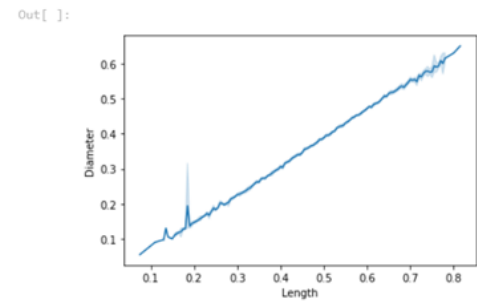
```
data.describe()
```

Out[17]:

	Sex	Length	Diameter	Height	Whole weight	Shucked weight	Viscera weight	Shell weight	Rings
count	4177.000000	4177.000000	4177.000000	4177.000000	4177.000000	4177.000000	4177.000000	4177.000000	4177.000000
mean	1.008379	0.523992	0.407881	0.139516	0.828742	0.359367	0.180594	0.238831	9.933684
std	0.796410	0.120093	0.099240	0.041827	0.490389	0.221963	0.109614	0.139203	3.224169
min	0.000000	0.075000	0.055000	0.000000	0.002000	0.001000	0.000500	0.001500	1.000000
25%	0.000000	0.450000	0.350000	0.115000	0.441500	0.186000	0.093500	0.130000	8.000000
50%	1.000000	0.545000	0.425000	0.140000	0.799500	0.336000	0.171000	0.234000	9.000000
75%	2.000000	0.615000	0.480000	0.165000	1.153000	0.502000	0.253000	0.329000	11.000000
max	2.000000	0.815000	0.650000	1.130000	2.825500	1.488000	0.760000	1.005000	29.000000

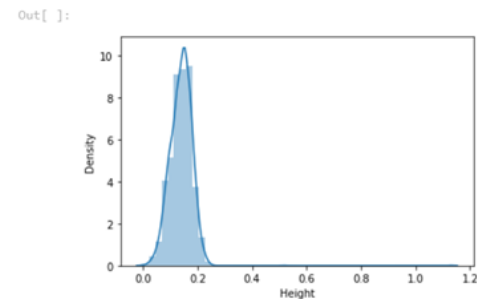
In []:

```
sns.lineplot(data["Length"],data["Diameter"])
```



In []:

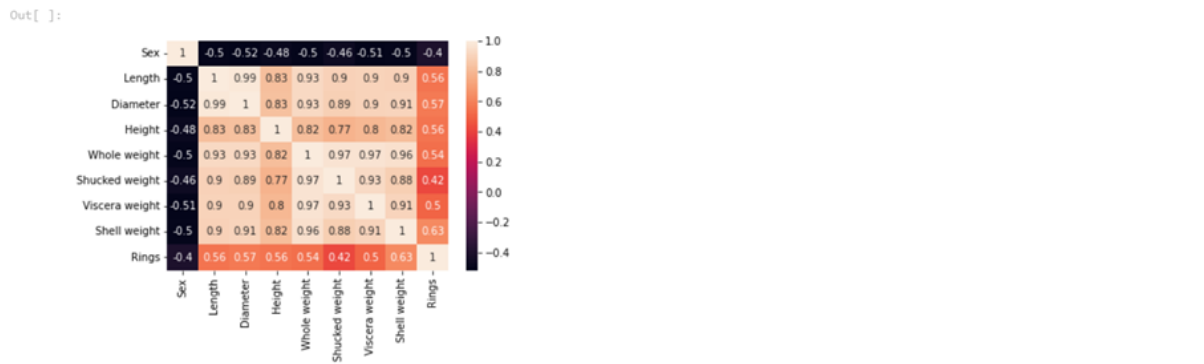
```
sns.distplot(data["Height"])
```



```
In [ ]: data.corr()
```

Out[]:	Sex	Length	Diameter	Height	Whole weight	Shucked weight	Viscera weight	Shell weight	Rings	
	Sex	1.000000	-0.503697	-0.516450	-0.477850	-0.501511	-0.459731	-0.505693	-0.499103	-0.401445
	Length	-0.503697	1.000000	0.986812	0.827554	0.925261	0.897914	0.903018	0.897706	0.556720
	Diameter	-0.516450	0.986812	1.000000	0.833684	0.925452	0.893162	0.899724	0.905330	0.574660
	Height	-0.477850	0.827554	0.833684	1.000000	0.819221	0.774972	0.798319	0.817338	0.557467
	Whole weight	-0.501511	0.925261	0.925452	0.819221	1.000000	0.969405	0.966375	0.955355	0.540390
	Shucked weight	-0.459731	0.897914	0.893162	0.774972	0.969405	1.000000	0.931961	0.882617	0.420884
	Viscera weight	-0.505693	0.903018	0.899724	0.798319	0.966375	0.931961	1.000000	0.907656	0.503819
	Shell weight	-0.499103	0.897706	0.905330	0.817338	0.955355	0.882617	0.907656	1.000000	0.627574
	Rings	-0.401445	0.556720	0.574660	0.557467	0.540390	0.420884	0.503819	0.627574	1.000000

```
In [ ]: sns.heatmap(data.corr(),annot = True)
```



```
In [18]: data.corr().Rings.sort_values()
```

```
Out[10]: Sex          -0.401445
Shucked weight      0.420884
Viscera weight      0.503819
Whole weight        0.540390
Length              0.556720
Height              0.557467
Diameter            0.574660
Shell weight        0.627574
Rings               1.000000
Name: Rings, dtype: float64
```

Split X & Y

```
In [19]: x = data.drop("Rings",axis = 1)
         y = data["Rings"]
```

```
In [20]: x
```

```
Out[20]:
```

	Sex	Length	Diameter	Height	Whole weight	Shucked weight	Viscera weight	Shell weight
0	1	0.455	0.365	0.095	0.5140	0.2245	0.1010	0.1500
1	1	0.350	0.265	0.090	0.2255	0.0995	0.0485	0.0700
2	0	0.530	0.420	0.135	0.6770	0.2565	0.1415	0.2100
3	1	0.440	0.365	0.125	0.5160	0.2155	0.1140	0.1550
4	2	0.330	0.255	0.080	0.2050	0.0895	0.0395	0.0550
...
4172	0	0.565	0.450	0.165	0.8870	0.3700	0.2390	0.2490
4173	1	0.590	0.440	0.135	0.9660	0.4390	0.2145	0.2605
4174	1	0.600	0.475	0.205	1.1760	0.5255	0.2875	0.3080
4175	0	0.625	0.485	0.150	1.0945	0.5310	0.2610	0.2960
4176	1	0.710	0.555	0.195	1.9485	0.9455	0.3765	0.4950

4177 rows × 8 columns

```
In [21]: y
```

```
Out[21]: 0      15
         1       7
         2       9
         3      10
         4       7
         ..
        4172    11
        4173    10
        4174     9
        4175    10
        4176    12
        Name: Rings, Length: 4177, dtype: int64
```

Train test split

```
In [22]: 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,random_state= 42)
```

Initialize logistic regression

```
In [23]: from sklearn.linear_model import LogisticRegression
```

```
In [24]: log_reg = LogisticRegression()
```

```
In [25]: log_reg.fit(x_train,y_train)
```

```
Out[25]: LogisticRegression()
```

Testing model

```
In [26]: pred = log_reg.predict(x_test)
```

Evaluation

```
In [27]: from sklearn.metrics import accuracy_score,precision_score,confusion_matrix
```

```
In [28]: print("Accuracy score :",accuracy_score(y_test,pred))
```

Accuracy score : 0.2703349282296651

