# TEAM ID PNT2022TMID21264

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

```
df=pd.read_csv("abalone.csv")
```

```
df.head()
```

| | 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 |

```
df.describe()
```

| | Length | Diameter | Height | Whole weight | Shucked weight | Viscera weight | |
|---|---|---|---|---|---|---|---|
| **count** | 4177.000000 | 4177.000000 | 4177.000000 | 4177.000000 | 4177.000000 | 4177.000000 | 4177. |
| **mean** | 0.523992 | 0.407881 | 0.139516 | 0.828742 | 0.359367 | 0.180594 | 0. |
| **std** | 0.120093 | 0.099240 | 0.041827 | 0.490389 | 0.221963 | 0.109614 | 0. |
| **min** | 0.075000 | 0.055000 | 0.000000 | 0.002000 | 0.001000 | 0.000500 | 0. |
| **25%** | 0.450000 | 0.350000 | 0.115000 | 0.441500 | 0.186000 | 0.093500 | 0. |
| **50%** | 0.545000 | 0.425000 | 0.140000 | 0.799500 | 0.336000 | 0.171000 | 0. |
| **75%** | 0.615000 | 0.480000 | 0.165000 | 1.153000 | 0.502000 | 0.253000 | 0. |

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

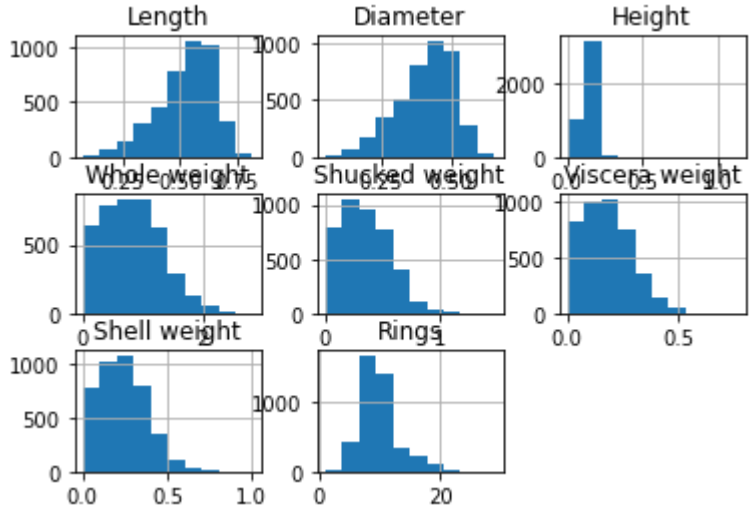
```
Sex               0
Length            0
Diameter          0
```

```
Height            0
Whole weight      0
Shucked weight    0
Viscera weight    0
Shell weight      0
Rings             0
dtype: int64
```

## Univariate analysis

```
df['Rings'].value_counts()
df.hist()
```

```
array([[<AxesSubplot:title={'center':'Length'}>,
        <AxesSubplot:title={'center':'Diameter'}>,
        <AxesSubplot:title={'center':'Height'}>],
       [<AxesSubplot:title={'center':'Whole weight'}>,
        <AxesSubplot:title={'center':'Shucked weight'}>,
        <AxesSubplot:title={'center':'Viscera weight'}>],
       [<AxesSubplot:title={'center':'Shell weight'}>,
        <AxesSubplot:title={'center':'Rings'}>, <AxesSubplot:>]],
      dtype=object)
```



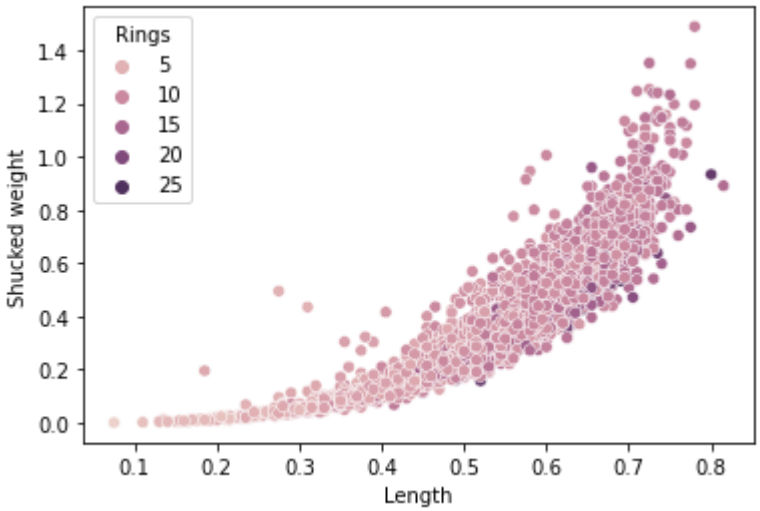

```
sns.pairplot(data=df,hue='Rings')
```

```
<seaborn.axisgrid.PairGrid at 0x276e0913850>
```

## Bi-variate analysis

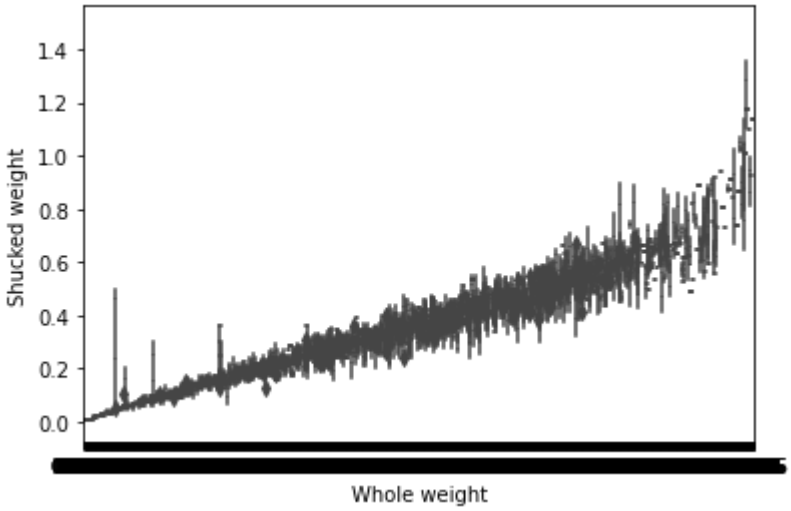
```
sns.scatterplot(data=df,x='Length',y='Shucked weight',hue='Rings')
```

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



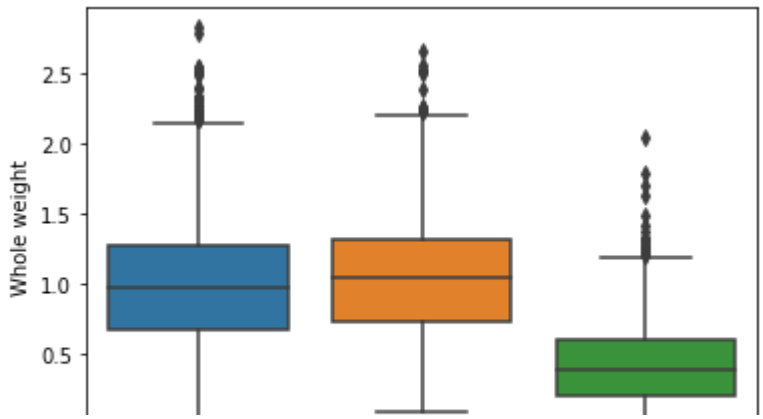

```
sns.boxplot(data=df,x='Whole weight',y='Shucked weight')
```

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



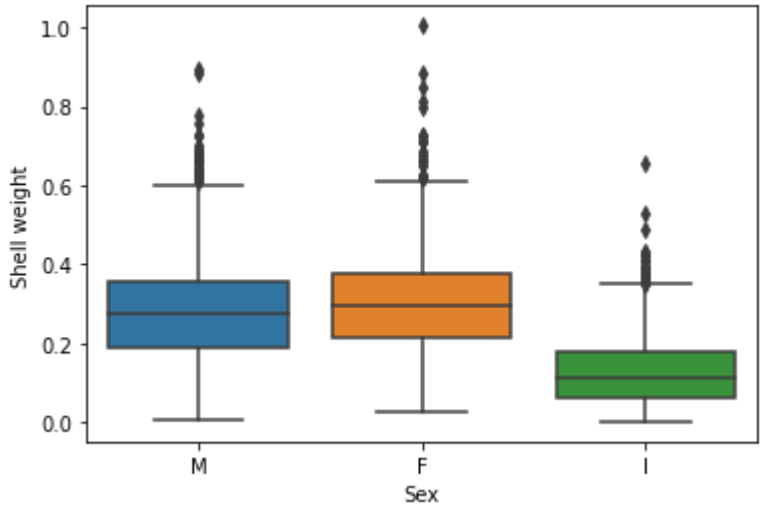

```
sns.boxplot(data=df,x='Sex',y='Whole weight')
```

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




```
sns.boxplot(data=df,x='Sex',y='Shell weight')
```

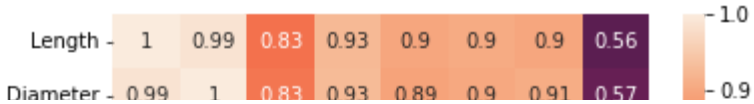
```
<AxesSubplot:xlabel='Sex', ylabel='Shell weight'>
```




## Multi-variate analysis

```
sns.heatmap(df.corr(),annot=True)
```

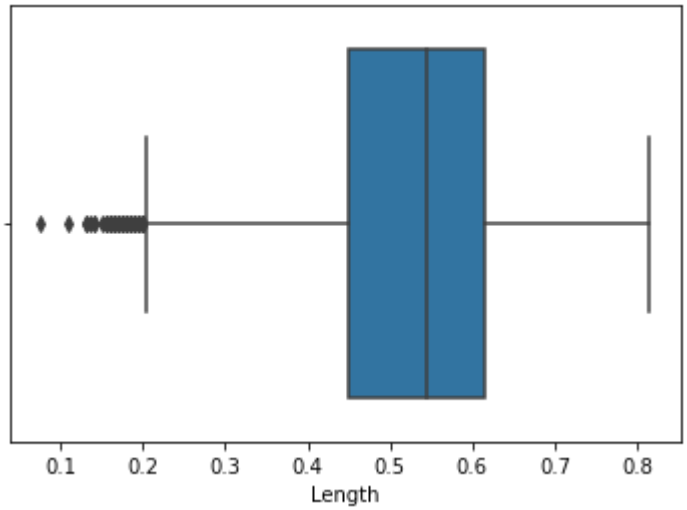
```
<AxesSubplot:>
```




## Outliers

```
sns.boxplot(x=df['Length'])
```

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




## Handling the outliers

```
[ ] ↳ 5 cells hidden
```

## Check for Categorical columns and perform encoding

```
from sklearn.preprocessing import LabelEncoder
le = LabelEncoder()
df["Sex"] = le.fit_transform(df["Sex"])
df["Sex"]
```

```
0       2
1       2
2       0
3       2
4       1
       ..
4172    0
4173    2
4174    2
4175    0
4176    2
Name: Sex, Length: 4177, dtype: int32
```

## Split the data into dependent and independent variables

```
x=df.iloc[:,0:8].values
y=df.iloc[:,8:9].values
```

```
x,y
```

```
(array([[2.    , 0.455 , 0.365 , ..., 0.2245, 0.101 , 0.15  ],
        [2.    , 0.355 , 0.265 , ..., 0.0995, 0.0485, 0.07  ],
        [0.    , 0.53  , 0.42  , ..., 0.2565, 0.1415, 0.21  ],
        ...,
        [2.    , 0.6   , 0.475 , ..., 0.5255, 0.2875, 0.308 ],
        [0.    , 0.625 , 0.485 , ..., 0.531 , 0.261 , 0.296 ],
        [2.    , 0.66  , 0.555 , ..., 0.647 , 0.3765, 0.495 ]]),
 array([[15],
        [ 7],
        [ 9],
        ...,
        [ 9],
        [10],
        [12]], dtype=int64))
```

## Split the data into training and testing

```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test = train_test_split(x,y,test_size=0.3,random_state=0)
```

```
x_train.shape
```

```
(2923, 8)
```

```
x_test.shape
```

```
(1254, 8)
```

## Build the Model

```
from sklearn.linear_model import LinearRegression
lr = LinearRegression()
```

## Train the Model

```
lr.fit(x_train, y_train)
```

```
LinearRegression()
```

## Test the Model

```
y_pred = lr.predict(x_test)
print((y_test)[0:6])
print((y_pred)[0:6])
```

```
[[13]
 [ 8]
 [11]
 [ 5]
 [12]
 [11]]
[[13.09114191]
 [ 9.88567356]
 [ 9.85523183]
 [ 5.27773184]
 [10.03324002]
 [11.98903078]]
```

## Measure the performance using Metrics

```
from sklearn.metrics import mean_squared_error, mean_absolute_error, r2_score
```

```
from sklearn.model_selection import cross_val_score
cv_scores = cross_val_score(lr, x, y, cv=5)
sco=cv_scores.round(4)
print(cv_scores.round(4))
print("Average",sco.sum()/5)
```

```
[0.3577 0.0399 0.4503 0.5076 0.3998]
Average 0.35106000000000004
```

```
print(r2_score( y_test,y_pred))
```

```
0.47569319658142517
```

```
print(mean_absolute_error( y_test, y_pred))
print(mean_squared_error(y_test, y_pred))
```

```
1.6823028820553498
5.527771900464179
```