ASSIGNMENT - 4

Assignment Date	07 OCTOBER 2022
Student Name	SUMITHRA S
Student Roll Number	111519205048

Description:- Predicting the age of abalone from physical measurements. The age of abalone is determined by cutting the shell through the cone, staining it, and counting the number of rings through a microscope -- a boring and time-consuming task. Other measurements, which are easier to obtain, are used to predict age. Further information, such as weather patterns and location (hence food availability) may be required to solve the problem.

Attribute Information:

Given is the attribute name, attribute type, measurement unit, and a brief description. The number of rings is the value to predict: either as a continuous value or as a classification problem.

Name / Data Type / Measurement Unit / Description

1- Sex / nominal / -- / M, F, and I (infant)

2- Length / continuous / mm / Longest shell measurement

3- Diameter / continuous / mm / perpendicular to length

4- Height / continuous / mm / with meat in shell

5- Whole weight / continuous / grams / whole abalone

6- Shucked weight / continuous / grams / weight of meat

7- Viscera weight / continuous / grams / gut weight (after bleeding)

8- Shell weight / continuous / grams / after being dried

9- Rings / integer / -- / +1.5 gives the age in years

Building a Regression Model

1. Download the dataset: Dataset
2. Load the dataset into the tool.
3. Perform Below Visualizations.

- Univariate Analysis
- Bi-Variate Analysis
- Multi-Variate Analysis

4. Perform descriptive statistics on the dataset.
5. Check for Missing values and deal with them.
6. Find the outliers and replace them outliers
7. Check for Categorical columns and perform encoding. 8. Split the data into dependent and independent variables. 9. Scale the independent variables
10. Split the data into training and testing
11. Build the Model
12. Train the Model
13. Test the Model

14. Measure the performance using Metrics.

```
#import libraries import numpy as np import pandas as pd
import matplotlib.pyplot as plt import seaborn as sb import
plotly.express as px
```

2. Load the dataset into the tool

```
from google.colab import drive
drive.mount('/content/drive')
```

```
Mounted at /content/drive
```

```
import os os.chdir("/content/drive/My
Drive") !ls
```

```
data = pd.read_csv('/content/drive/My Drive/Colab Notebooks/abalone.csv') data
```

	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	0.1500	15
1	M	0.350	0.265	0.090	0.2255	0.0995	0.0485	0.0700	7
2	F	0.530	0.420	0.135	0.6770	0.2565	0.1415	0.2100	9
3	M	0.440	0.365	0.125	0.5160	0.2155	0.1140	0.1550	10
4	I	0.330	0.255	0.080	0.2050	0.0895	0.0395	0.0550	7
...	...	...	...	...	...	...	...	...	...
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

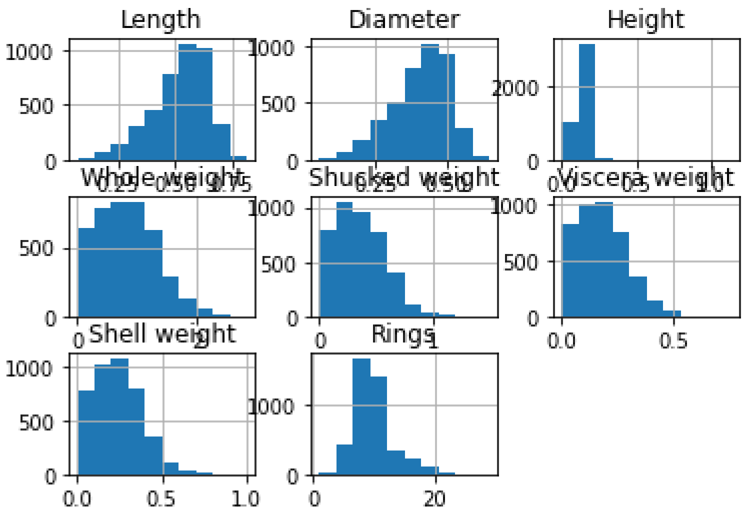
4177 rows × 9 columns

3. Perform Below Visualizations.

- Univariate Analysis

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

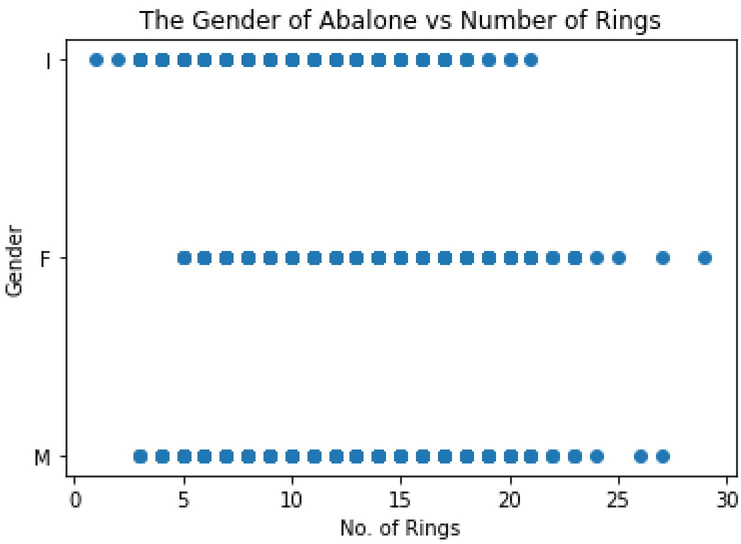
```
array([[<matplotlib.axes._subplots.AxesSubplot object at 0x7fa4a73e35d0>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7fa4a6255990>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7fa4a620df90>],
       [<matplotlib.axes._subplots.AxesSubplot object at 0x7fa4a61c3bd0>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7fa4a6188210>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7fa4a6140810>],
       [<matplotlib.axes._subplots.AxesSubplot object at 0x7fa4a6173e90>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7fa4a6138410>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x7fa4a6138450>]],
      dtype=object)
```



Bi-Variate Analysis

```
plt.scatter(data.Rings, data.Sex) plt.title('The
Gender of Abalone vs Number of Rings')
plt.xlabel('No. of Rings') plt.ylabel('Gender')
```

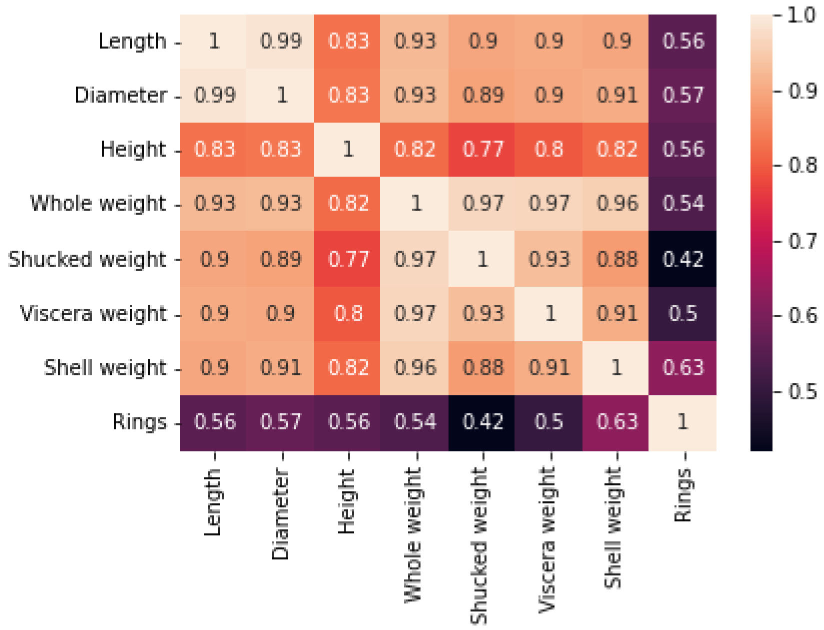
```
Text(0, 0.5, 'Gender')
```



· Multi-Variate Analysis

```
sb.heatmap(data.corr(),annot=True
)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fa4a59c4350>
```



4. Perform descriptive statistics on the dataset.

```
data.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
```

```
data.describe()
```

Whole Shucked Viscera Length

Diameter Height

weight **weight** **weight**

count 4177.000000 4177.000000 4177.000000 4177.000000 4177.000000 4177.000000 41

mean 0.523992 0.407881 0.139516 0.828742 0.359367 **std**

0.120093 0.099240 0.041827 0.490389 0.221963 **min** 0.180594

0.075000 0.055000 0.000000 0.002000 0.001000 **25%** 0.109614

0.450000 0.350000 0.115000 0.441500 0.186000 0.000500

50% 0.545000 0.425000 0.140000 0.799500 0.336000 0.093500

75% 0.615000 0.480000 0.165000 1.153000 0.502000 0.171000

0.253000

5. Check**max** for0.815000 Missing 0.650000 values 1.130000 and deal 0.760000

2.825500 with them.1.488000

There is no missing values

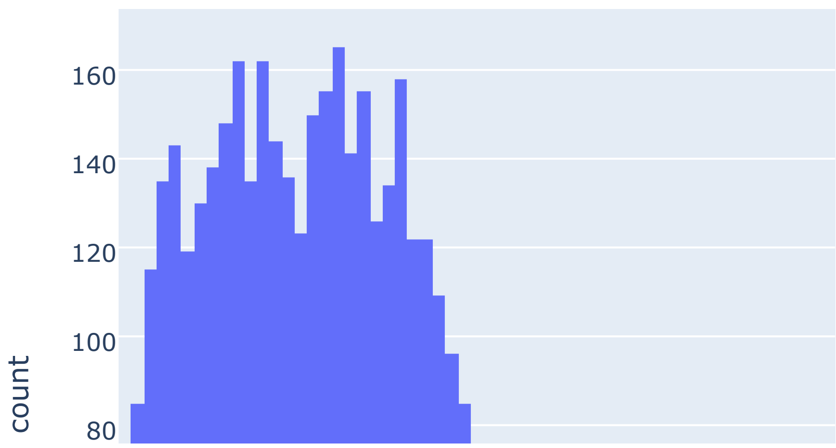
```
data.isnull().any()
```

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

6. Find the outl ers and replace them outliers

The dataset does not have a outliers

```
fig = px.histogram(data, x='Whole weight')
fig.show()
```



7. Check for Categorical columns and perform encoding.

There is one Categorical column SEX is replaced by an Integer

```
from sklearn.preprocessing import LabelEncoder
le = LabelEncoder()
data["Sex"] = le.fit_transform(data["Sex"])
data["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: int64
```

8. Split the data into dependent and independent variables.

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

```
x
```

```
array([[2.      , 0.455 , 0.365 , ..., 0.2245, 0.101 , 0.15 ],
       [2.      , 0.35 , 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. , 0.71 , 0.555 , ..., 0.9455, 0.3765, 0.495 ]])
```

```
y
array([[15], [ 7],
       [ 9], ...,
       [ 9],
       [10],
       [12]])
```

9. Scale the independent variables

```
x=data.iloc[:,0:8]
print(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
```

10. 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)
```

11. Build the Model

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

12. Train the Model

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

```
LinearRegression()
```

13. 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]]
[[13.11640829]
 [ 9.65691091]
 [10.35350972]
 [ 5.63648715]
 [10.67436485]
 [11.95341338]]
```

14. Measure the performance using Metrics.

```
# RMSE(Root Mean Square Error)

from sklearn.metrics import mean_squared_error mse =
mean_squared_error(y_test, y_pred) rmse = np.sqrt(mse)
print("RMSE value : {:.2f}".format(rmse))
```

```
RMSE value : 2.26
```

```
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)
```

```
[0.4113 0.1574 0.4807 0.5046 0.4362] -----  Average 0.39803999999999995
```