Regression Model

1.Downloading the Dataset

2.Load the dataset into the tool

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

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

```
data.shape
```

```
(4177, 9)
```

```
Age=1.5+data.Rings
data["Age"]=Age
data=data.rename(columns = {'Whole weight':'Whole_weight','Shucked weight': 'Shucked_weight',
                            'Shell weight': 'Shell_weight'})
data=data.drop(columns=["Rings"],axis=1)
data.head()
```

	Sex	Length	Diameter	Height	Whole_weight	Shucked_weight	Viscera_weight	Shell_w
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	

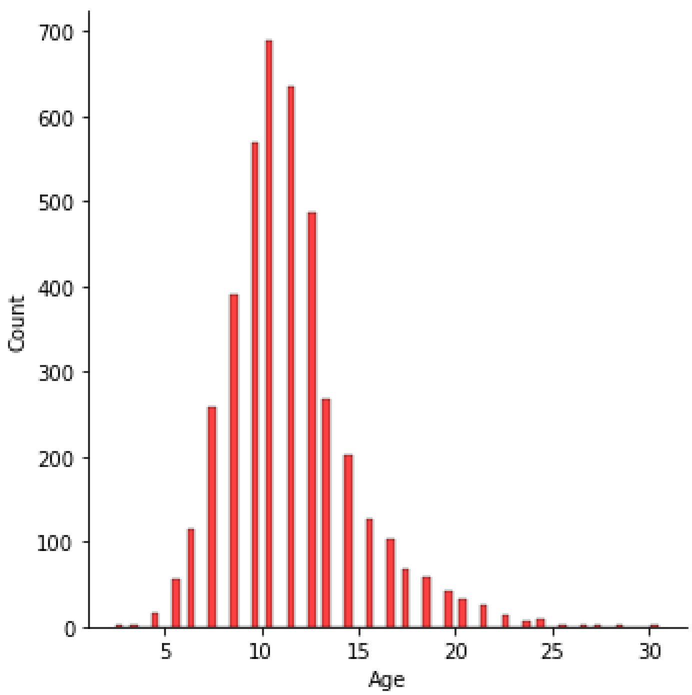
3.Perform Below Visualizations

(i) Univariate Analysis

Histogram

```
sns.displot(data["Age"], color='red')
```

<seaborn.axisgrid.FacetGrid at 0x2706bb8ebb0>

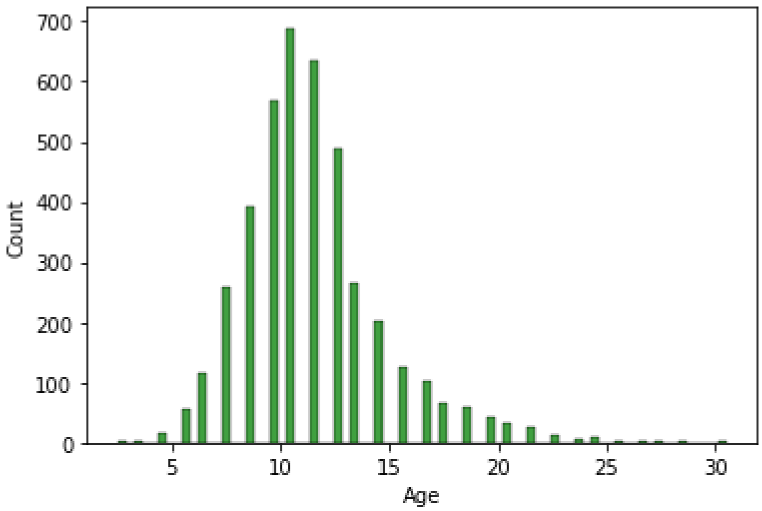



```
sns.histplot(y=data.Age,color='blue')
```

<AxesSubplot:xlabel='Count', ylabel='Age'>

```
sns.histplot(x=data.Age,color='green')
```

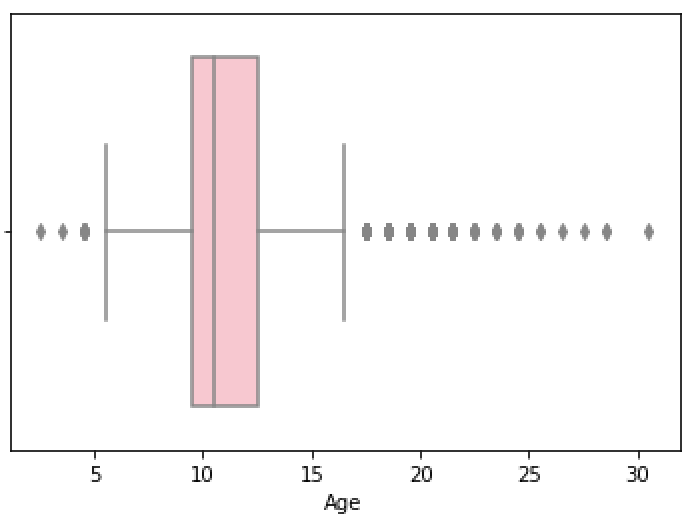
<AxesSubplot:xlabel='Age', ylabel='Count'>



Boxplot

```
sns.boxplot(x=data.Age,color='pink')
```

<AxesSubplot:xlabel='Age'>



Countplot

[ ] ↳ 1 cell hidden

(ii) Bi-Variate Analysis

Barplot

[ ] ↳ 1 cell hidden

Linearplot

[ ] ↳ 1 cell hidden

Scatterplot

[ ] ↳ 1 cell hidden

(iii) Multi-Variate Analysis

Pairplot

[ ] ↳ 1 cell hidden

4.Perform descriptive statistics on the dataset

```
data.describe(include='all')
```

	Sex	Length	Diameter	Height	Whole_weight	Shucked_weight	Viscer
count	4177	4177.000000	4177.000000	4177.000000	4177.000000	4177.000000	41
unique	3	NaN	NaN	NaN	NaN	NaN	
top	M	NaN	NaN	NaN	NaN	NaN	
freq	1528	NaN	NaN	NaN	NaN	NaN	

5.Check for Missing values and deal with them

```
data.isnull().sum()
```

```
Sex               0
Length            0
Diameter          0
Height            0
Whole_weight      0
Shucked_weight    0
Viscera_weight    0
Shell_weight      0
Age               0
dtype: int64
```

6.Find the outliers and replace them outliers

```
outliers=data.quantile(q=(0.25,0.75))
outliers
```

	Length	Diameter	Height	Whole_weight	Shucked_weight	Viscera_weight	Shell_weig
0.25	0.450	0.35	0.115	0.4415	0.186	0.0935	0.1
0.75	0.615	0.48	0.165	1.1530	0.502	0.2530	0.3

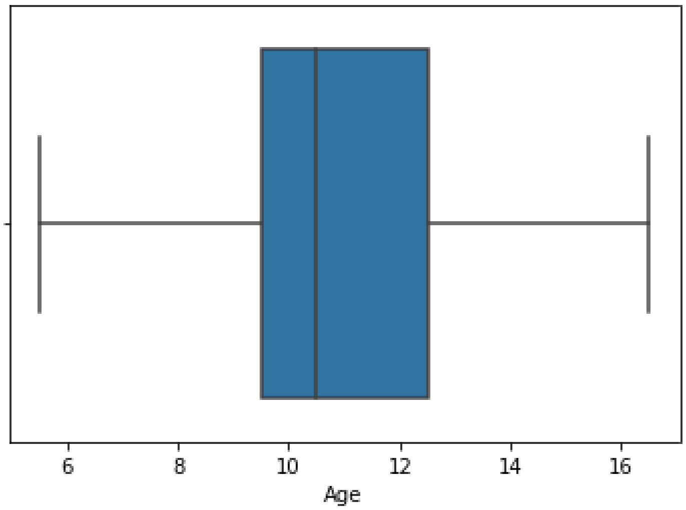
```
a = data.Age.quantile(0.25)
b = data.Age.quantile(0.75)
c = b - a
lower_limit = a - 1.5 * c
data.median(numeric_only=True)
```

```
Length            0.5450
Diameter          0.4250
Height            0.1400
Whole_weight      0.7995
Shucked_weight    0.3360
Viscera_weight    0.1710
```

```
Shell_weight       0.2340
Age               10.5000
dtype: float64
```

```
data['Age'] = np.where(data['Age'] < lower_limit, 7, data['Age'])
sns.boxplot(x=data.Age,showfliers = False)
```

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



7.Check for Categorical columns and perform encoding

```
data.head()
```

	Sex	Length	Diameter	Height	Whole_weight	Shucked_weight	Viscera_weight	Shell_w
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	

```
from sklearn.preprocessing import LabelEncoder

lab = LabelEncoder()
data.Sex = lab.fit_transform(data.Sex)

data.head()
```

	Sex	Length	Diameter	Height	Whole_weight	Shucked_weight	Viscera_weight	Shell_w
0	2	0.455	0.365	0.095	0.5140	0.2245	0.1010	
1	2	0.350	0.265	0.090	0.2255	0.0995	0.0485	
2	0	0.530	0.420	0.135	0.6770	0.2565	0.1415	
3	2	0.440	0.365	0.125	0.5160	0.2155	0.1140	

8.Split the data into dependent and independent variables

```
y = data["Sex"]
y.head()
```

```
0    2
1    2
2    0
3    2
4    1
Name: Sex, dtype: int32
```

```
x=data.drop(columns=["Sex"],axis=1)
x.head()
```

	Length	Diameter	Height	Whole_weight	Shucked_weight	Viscera_weight	Shell_weight
0	0.455	0.365	0.095	0.5140	0.2245	0.1010	0.150
1	0.350	0.265	0.090	0.2255	0.0995	0.0485	0.070
2	0.530	0.420	0.135	0.6770	0.2565	0.1415	0.210
3	0.440	0.365	0.125	0.5160	0.2155	0.1140	0.155
4	0.330	0.255	0.080	0.2050	0.0895	0.0395	0.055

9.Scale the independent variables

```
from sklearn.preprocessing import scale
X_Scaled = pd.DataFrame(scale(x), columns=x.columns)
X_Scaled.head()
```

	Length	Diameter	Height	Whole_weight	Shucked_weight	Viscera_weight	Shell_we
0	-0.574558	-0.432149	-1.064424	-0.641898	-0.607685	-0.726212	-0.63
1	-1.448986	-1.439929	-1.183978	-1.230277	-1.170910	-1.205221	-1.21
2	0.050033	0.122130	-0.107991	-0.309469	-0.463500	-0.356690	-0.20

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_Scaled, y, test_size=0.2, random_state=
```

```
X_Train.shape,X_Test.shape
```

```
((3341, 8), (836, 8))
```

```
Y_Train.shape,Y_Test.shape
```

```
((3341,), (836,))
```

```
X_Train.head()
```

	Length	Diameter	Height	Whole_weight	Shucked_weight	Viscera_weight	Shel
3141	-2.864726	-2.750043	-1.423087	-1.622870	-1.553902	-1.583867	-
3521	-2.573250	-2.598876	-2.020857	-1.606554	-1.551650	-1.565619	-
883	1.132658	1.230689	0.728888	1.145672	1.041436	0.286552	
3627	1.590691	1.180300	1.446213	2.164373	2.661269	2.330326	
2106	0.591345	0.474853	0.370226	0.432887	0.255175	0.272866	

```
X_Test.head()
```

	Length	Diameter	Height	Whole_weight	Shucked_weight	Viscera_weight	Shel
668	0.216591	0.172519	0.370226	0.181016	-0.368878	0.569396	
1580	-0.199803	-0.079426	-0.466653	-0.433875	-0.443224	-0.343004	-
3784	0.799543	0.726798	0.370226	0.870348	0.755318	1.764639	
463	-2.531611	-2.447709	-2.020857	-1.579022	-1.522362	-1.538247	-
2615	1.007740	0.928354	0.848442	1.390405	1.415417	1.778325	

```
Y_Train.head()
```

```
3141    1
3521    1
883     2
3627    2
2106    2
Name: Sex, dtype: int32
```

```
Y_Test.head()
```

```
668     2
1580    1
3784    2
463     1
2615    2
Name: Sex, dtype: int32
```

11.Build the Model

```
from sklearn.ensemble import RandomForestClassifier
model = RandomForestClassifier(n_estimators=10,criterion='entropy')
```

```
model.fit(X_Train,Y_Train)
```

```
RandomForestClassifier(criterion='entropy', n_estimators=10)
```

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

```
y_predict_train = model.predict(X_Train)
```

12.Train the Model

```
from sklearn.metrics import accuracy_score,confusion_matrix,classification_report
print('Training accuracy: ',accuracy_score(Y_Train,y_predict_train))
```

```
Training accuracy:  0.9841364860820113
```

13.Test the Model

```
print('Testing accuracy: ',accuracy_score(Y_Test,y_predict))
```

```
Testing accuracy:  0.5311004784688995
```

14.Measure the performance using Metrics

```
pd.crosstab(Y_Test,y_predict)
print(classification_report(Y_Test,y_predict))
```

```
              precision    recall  f1-score   support

           0       0.42      0.51      0.46       249
           1       0.74      0.73      0.74       291
           2       0.42      0.35      0.38       296

    accuracy                           0.53       836
   macro avg       0.53      0.53      0.53       836
weighted avg       0.53      0.53      0.53       836
```