

Name : EDE RENUKA MADHAV

REG NO: 21BCT0244

VIT VELLORE

SLOT : 6:00 PM TO 8:00PM

1. Download the dataset: Dataset # 2. Load the dataset

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

df=pd.read_csv("/content/penguins_size.csv")
df.head()
```

	species	island	culmen_length_mm	culmen_depth_mm
0	Adelie	Torgersen	39.1	18.7
1	Adelie	Torgersen	39.5	17.4
2	Adelie	Torgersen	40.3	18.0
3	Adelie	Torgersen	NaN	NaN
4	Adelie	Torgersen	36.7	19.3

	body_mass_g	sex
0	3750.0	MALE
1	3800.0	FEMALE
2	3250.0	FEMALE
3	NaN	NaN
4	3450.0	FEMALE

3. Perform Below Visualizations.

1. UNIVARIATE ANALYSIS

```
sns.distplot([df.culmen_length_mm])
```

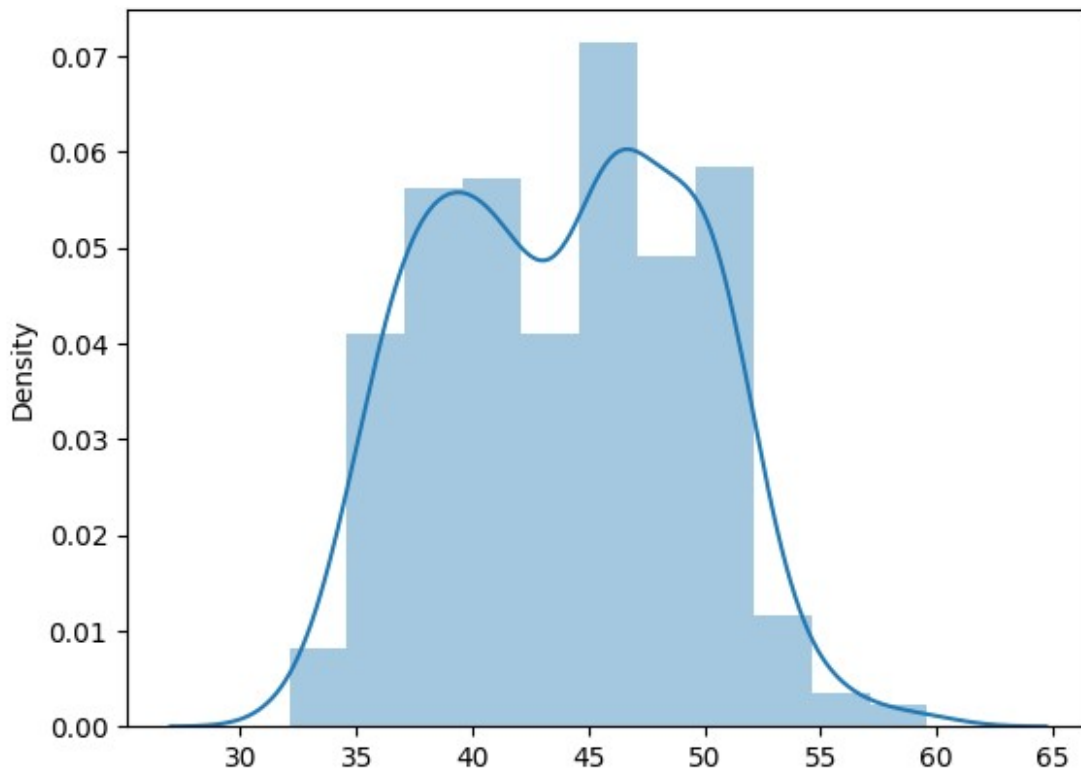
<ipython-input-5-632e7580f6db>:1: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with similar flexibility) or `histplot` (an axes-level function for histograms).

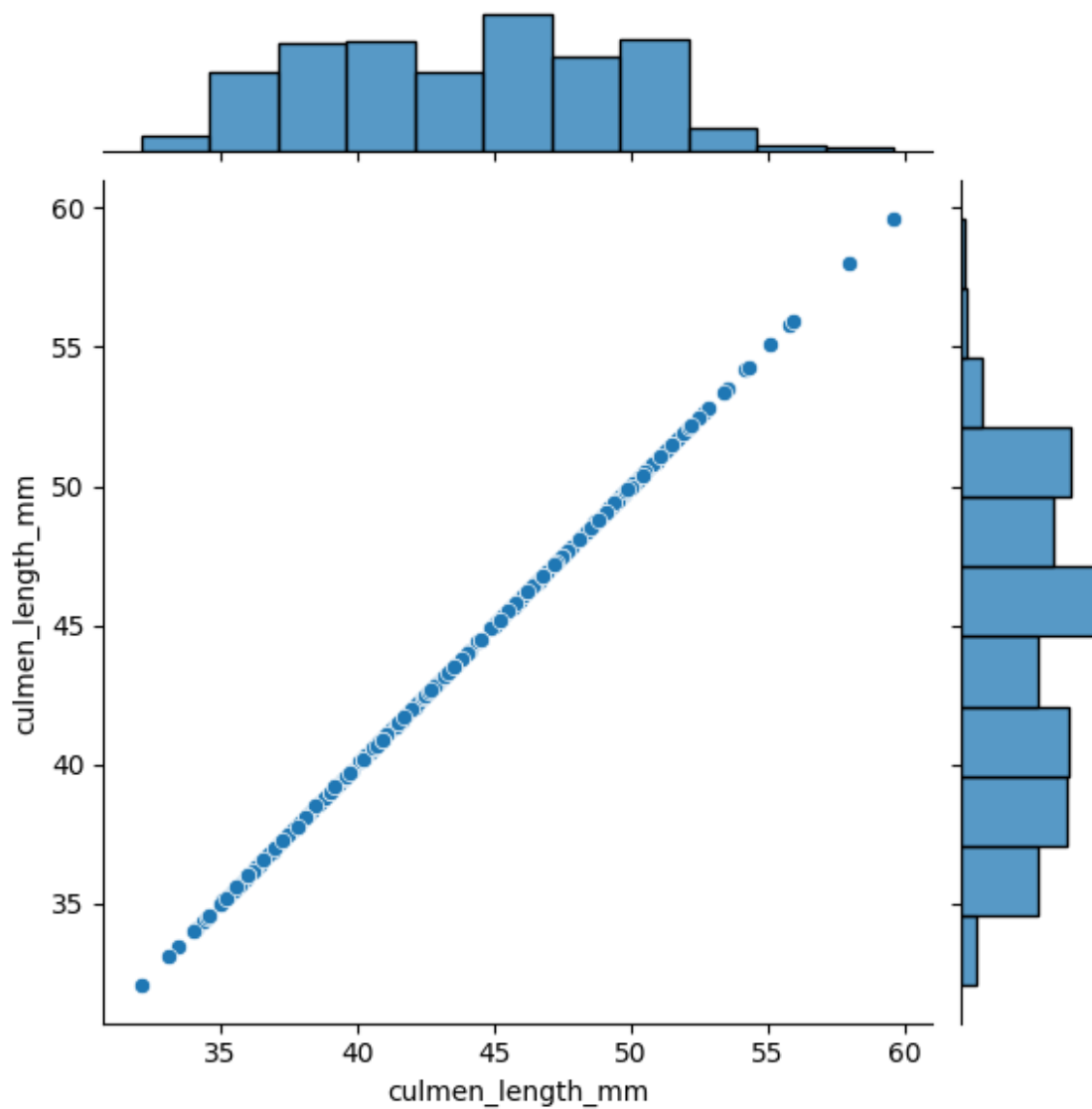
For a guide to updating your code to use the new functions, please see <https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751>

```
sns.distplot([df.culmen_length_mm])  
<Axes: ylabel='Density'>
```



2. BI-VARIATE ANALYSIS

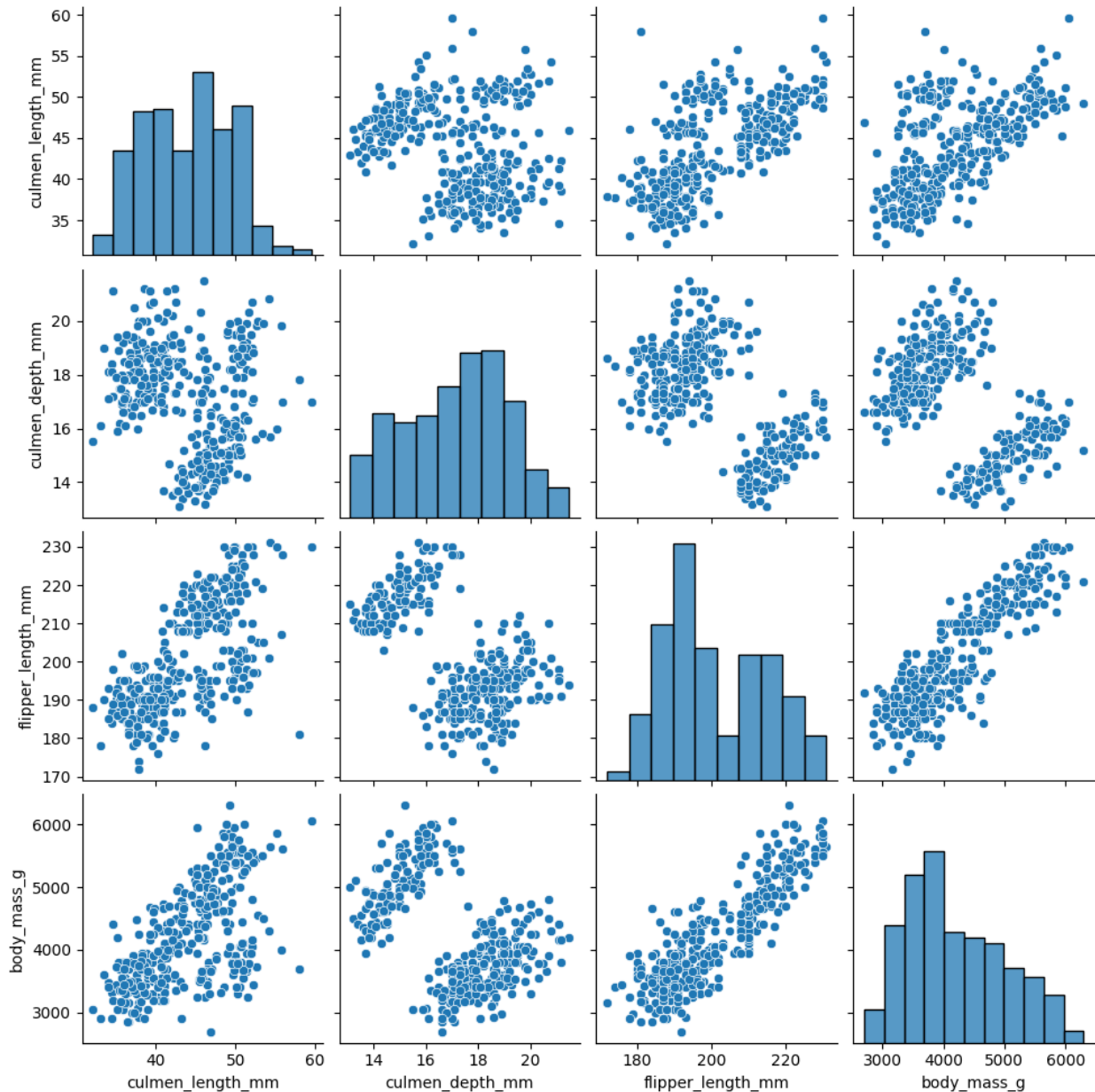
```
sns.jointplot(x='culmen_length_mm', y='culmen_length_mm', data=df)  
<seaborn.axisgrid.JointGrid at 0x7e73a9048970>
```



3. MULTIVARIATE ANALYSIS

```
sns.pairplot(df)
```

```
<seaborn.axisgrid.PairGrid at 0x7e73d7a939d0>
```



4. Perform descriptive statistics on the dataset.

```
df.describe()
```

	culmen_length_mm	culmen_depth_mm	flipper_length_mm
body_mass_g			
count	342.000000	342.000000	342.000000
mean	43.921930	17.151170	200.915205
std	5.459584	1.974793	14.061714

801.954536			
min	32.100000	13.100000	172.000000
2700.000000			
25%	39.225000	15.600000	190.000000
3550.000000			
50%	44.450000	17.300000	197.000000
4050.000000			
75%	48.500000	18.700000	213.000000
4750.000000			
max	59.600000	21.500000	231.000000
6300.000000			

5. Check for Missing values and deal with them.

```
df.isnull().any()
```

```
species          False
island           False
culmen_length_mm  True
culmen_depth_mm  True
flipper_length_mm True
body_mass_g      True
sex              True
dtype: bool
```

We have found that there are null values in the dataset

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 344 entries, 0 to 343
Data columns (total 7 columns):
#   Column                Non-Null Count  Dtype
---  -
0   species               344 non-null   object
1   island                344 non-null   object
2   culmen_length_mm      342 non-null   float64
3   culmen_depth_mm      342 non-null   float64
4   flipper_length_mm     342 non-null   float64
5   body_mass_g           342 non-null   float64
6   sex                   334 non-null   object
dtypes: float64(4), object(3)
memory usage: 18.9+ KB
```

Handling the null values of numerical parameters with median

```
df.median()
```

```
<ipython-input-11-6d467abf240d>:1: FutureWarning: The default value of
numeric_only in DataFrame.median is deprecated. In a future version,
it will default to False. In addition, specifying 'numeric_only=None'
is deprecated. Select only valid columns or specify the value of
numeric_only to silence this warning.
```

```
df.median()
```

```
culmen_length_mm      44.45
culmen_depth_mm       17.30
flipper_length_mm     197.00
body_mass_g          4050.00
dtype: float64
```

Replacing each column with the value of median

```
df['culmen_length_mm'].fillna(df['culmen_length_mm'].median(),inplace=
True)
```

```
df.head()
```

	species	island	culmen_length_mm	culmen_depth_mm
0	Adelie	Torgersen	39.10	18.7
1	Adelie	Torgersen	39.50	17.4
2	Adelie	Torgersen	40.30	18.0
3	Adelie	Torgersen	44.45	NaN
4	Adelie	Torgersen	36.70	19.3

	body_mass_g	sex
0	3750.0	MALE
1	3800.0	FEMALE
2	3250.0	FEMALE
3	NaN	NaN
4	3450.0	FEMALE

```
df['culmen_depth_mm'].fillna(df['culmen_depth_mm'].median(),inplace=Tr
ue)
```

```
df.head()
```

	species	island	culmen_length_mm	culmen_depth_mm
0	Adelie	Torgersen	39.10	18.7

```

181.0
1 Adelie Torgersen      39.50      17.4
186.0
2 Adelie Torgersen      40.30      18.0
195.0
3 Adelie Torgersen      44.45      17.3
NaN
4 Adelie Torgersen      36.70      19.3
193.0

```

```

      body_mass_g      sex
0      3750.0      MALE
1      3800.0     FEMALE
2      3250.0     FEMALE
3         NaN         NaN
4      3450.0     FEMALE

```

```

df['flipper_length_mm'].fillna(df['flipper_length_mm'].median(),inplace=True)
df.head()

```

```

      species      island  culmen_length_mm  culmen_depth_mm
flipper_length_mm \
0 Adelie Torgersen      39.10      18.7
181.0
1 Adelie Torgersen      39.50      17.4
186.0
2 Adelie Torgersen      40.30      18.0
195.0
3 Adelie Torgersen      44.45      17.3
197.0
4 Adelie Torgersen      36.70      19.3
193.0

```

```

      body_mass_g      sex
0      3750.0      MALE
1      3800.0     FEMALE
2      3250.0     FEMALE
3         NaN         NaN
4      3450.0     FEMALE

```

```

df['body_mass_g'].fillna(df['body_mass_g'].median(),inplace=True)
df.head()

```

```

      species      island  culmen_length_mm  culmen_depth_mm
flipper_length_mm \
0 Adelie Torgersen      39.10      18.7
181.0
1 Adelie Torgersen      39.50      17.4
186.0

```

2	Adelie	Torgersen	40.30	18.0
195.0				
3	Adelie	Torgersen	44.45	17.3
197.0				
4	Adelie	Torgersen	36.70	19.3
193.0				

	body_mass_g	sex
0	3750.0	MALE
1	3800.0	FEMALE
2	3250.0	FEMALE
3	4050.0	NaN
4	3450.0	FEMALE

Replacing the categorical parameters with Mode

```
df.mode()
```

	species	island	culmen_length_mm	culmen_depth_mm
	flipper_length_mm	\		
0	Adelie	Biscoe	41.1	17.0
190.0				

	body_mass_g	sex
0	3800.0	MALE

Replacing the categorical column "sex" with mode

```
df['sex'].fillna(df['sex'].mode().iloc[0],inplace=True)
df.head()
```

	species	island	culmen_length_mm	culmen_depth_mm
	flipper_length_mm	\		
0	Adelie	Torgersen	39.10	18.7
181.0				
1	Adelie	Torgersen	39.50	17.4
186.0				
2	Adelie	Torgersen	40.30	18.0
195.0				
3	Adelie	Torgersen	44.45	17.3
197.0				
4	Adelie	Torgersen	36.70	19.3
193.0				

	body_mass_g	sex
--	-------------	-----


```
0      3750.0    MALE
1      3800.0  FEMALE
2      3250.0  FEMALE
3      4050.0    MALE
4      3450.0  FEMALE
```

```
df.head()
```

```
   species  island  culmen_length_mm  culmen_depth_mm
flipper_length_mm \
0  Adelie  Torgersen             39.10             18.7
181.0
1  Adelie  Torgersen             39.50             17.4
186.0
2  Adelie  Torgersen             40.30             18.0
195.0
3  Adelie  Torgersen             44.45             17.3
197.0
4  Adelie  Torgersen             36.70             19.3
193.0
```

```
   body_mass_g  sex
0      3750.0    MALE
1      3800.0  FEMALE
2      3250.0  FEMALE
3      4050.0    MALE
4      3450.0  FEMALE
```

```
df.isnull().any()
```

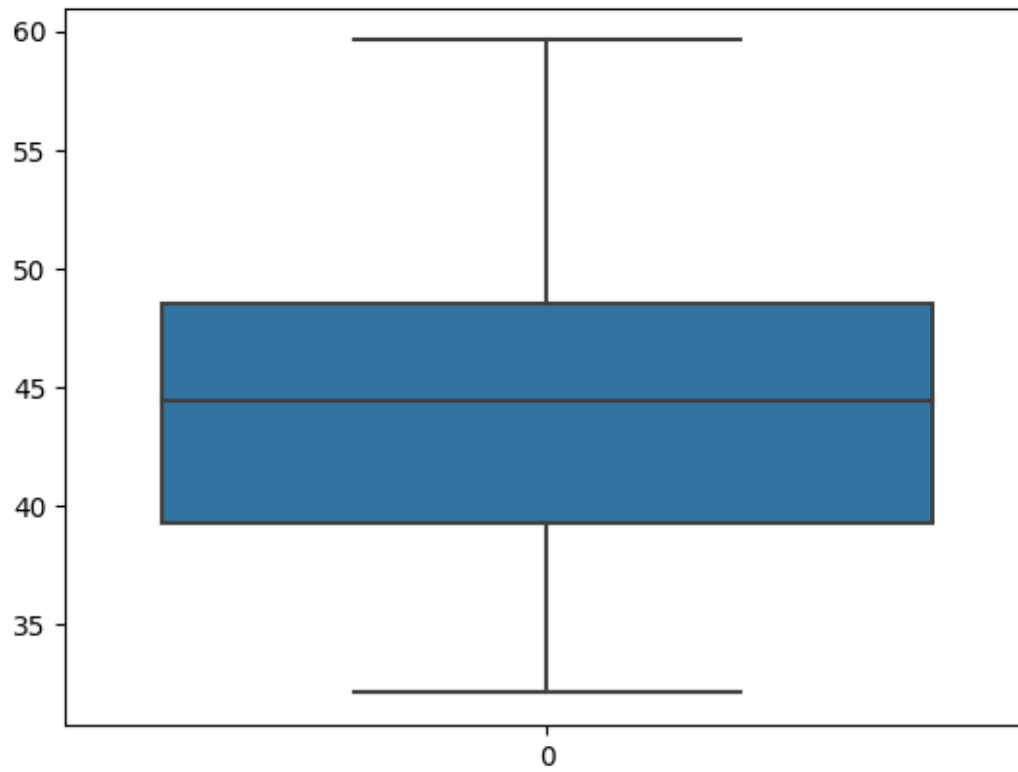
```
species      False
island       False
culmen_length_mm  False
culmen_depth_mm  False
flipper_length_mm  False
body_mass_g    False
sex            False
dtype: bool
```

All the missing values have been handled

6. Find the outliers and replace them outliers

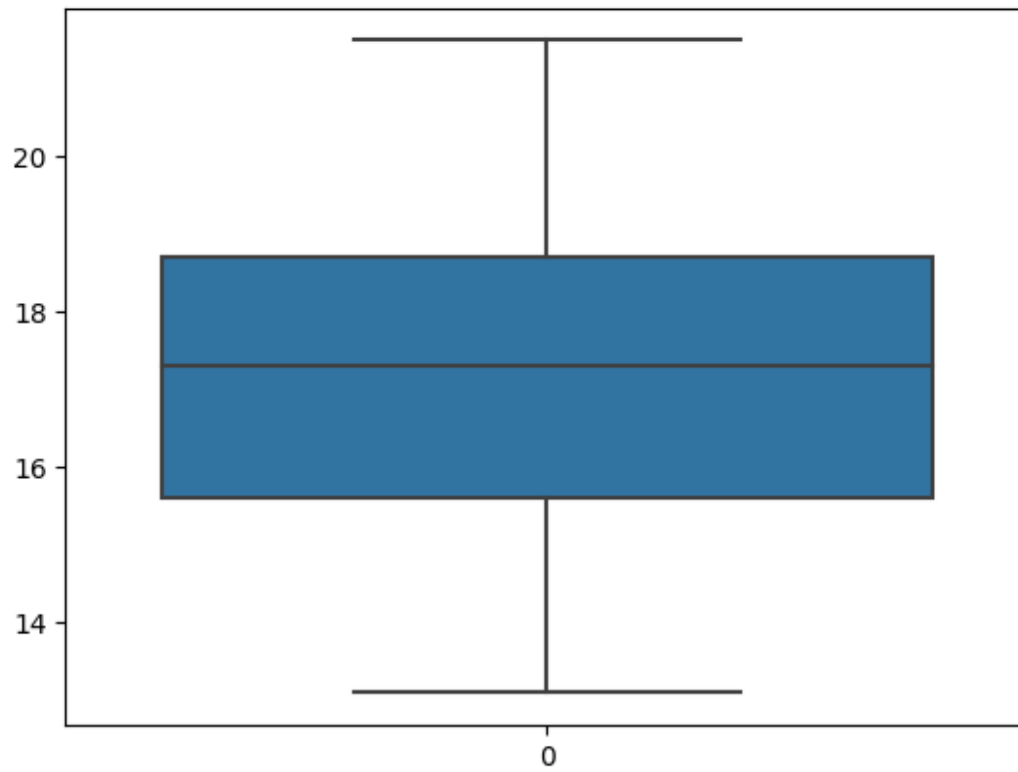
```
sns.boxplot(df.culmen_length_mm)
```

```
<Axes: >
```



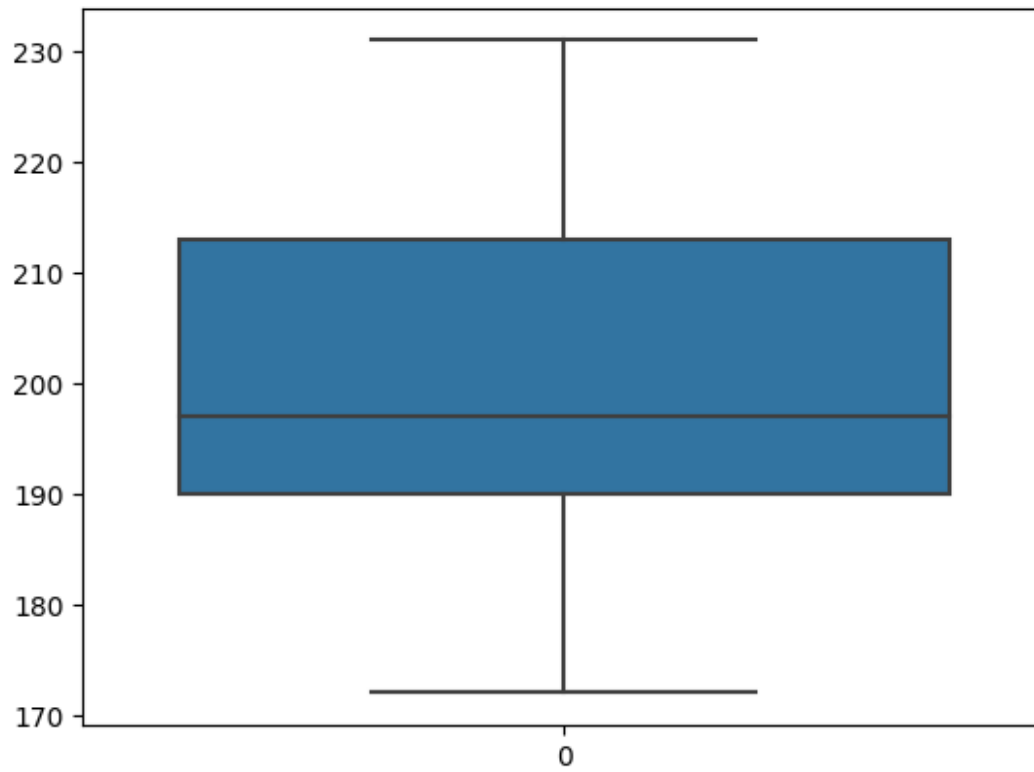
```
sns.boxplot(df.culmen_depth_mm)
```

```
<Axes: >
```



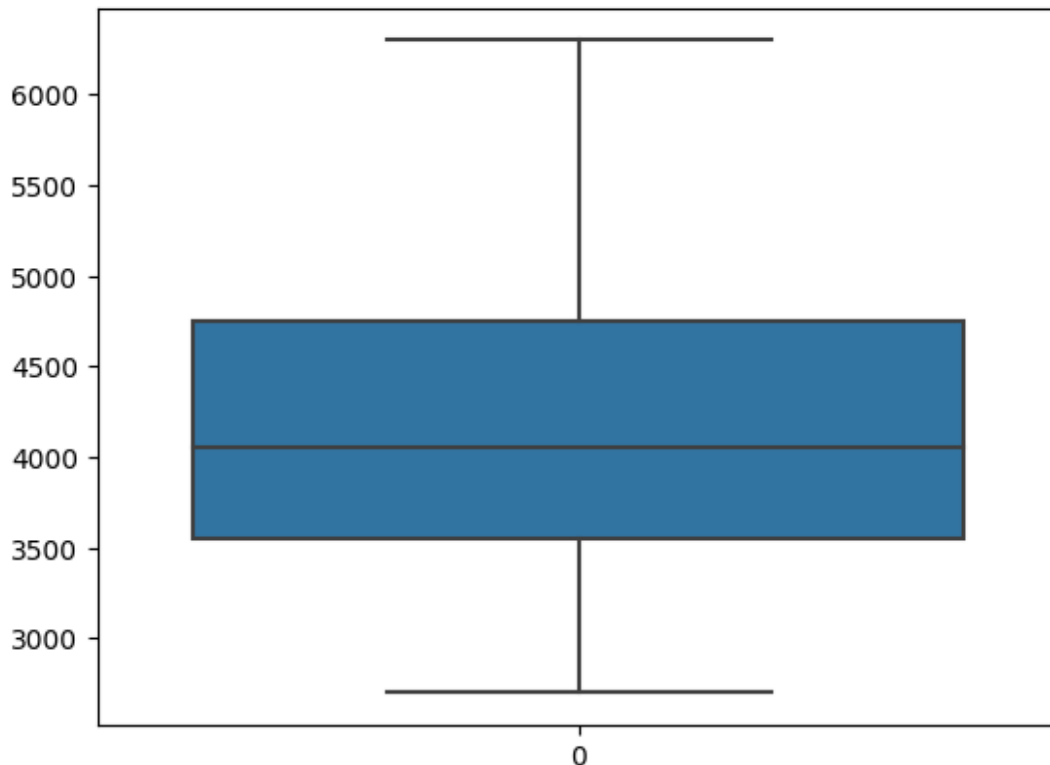
```
sns.boxplot(df.flipper_length_mm)
```

```
<Axes: >
```



```
sns.boxplot(df.body_mass_g)
```

```
<Axes: >
```



No outliers found

7. Check the correlation of independent variables with the target

```
df.corr()
```

<ipython-input-10-2f6f6606aa2c>:1: FutureWarning: The default value of numeric_only in DataFrame.corr is deprecated. In a future version, it will default to False. Select only valid columns or specify the value of numeric_only to silence this warning.

```
df.corr()
```

	culmen_length_mm	culmen_depth_mm
flipper_length_mm	0.656181	-0.583851
culmen_length_mm	1.000000	-0.235053
culmen_depth_mm	-0.235053	1.000000
flipper_length_mm	0.656181	-0.583851
	1.000000	

```
body_mass_g          0.595110      -0.471916
0.871202
```

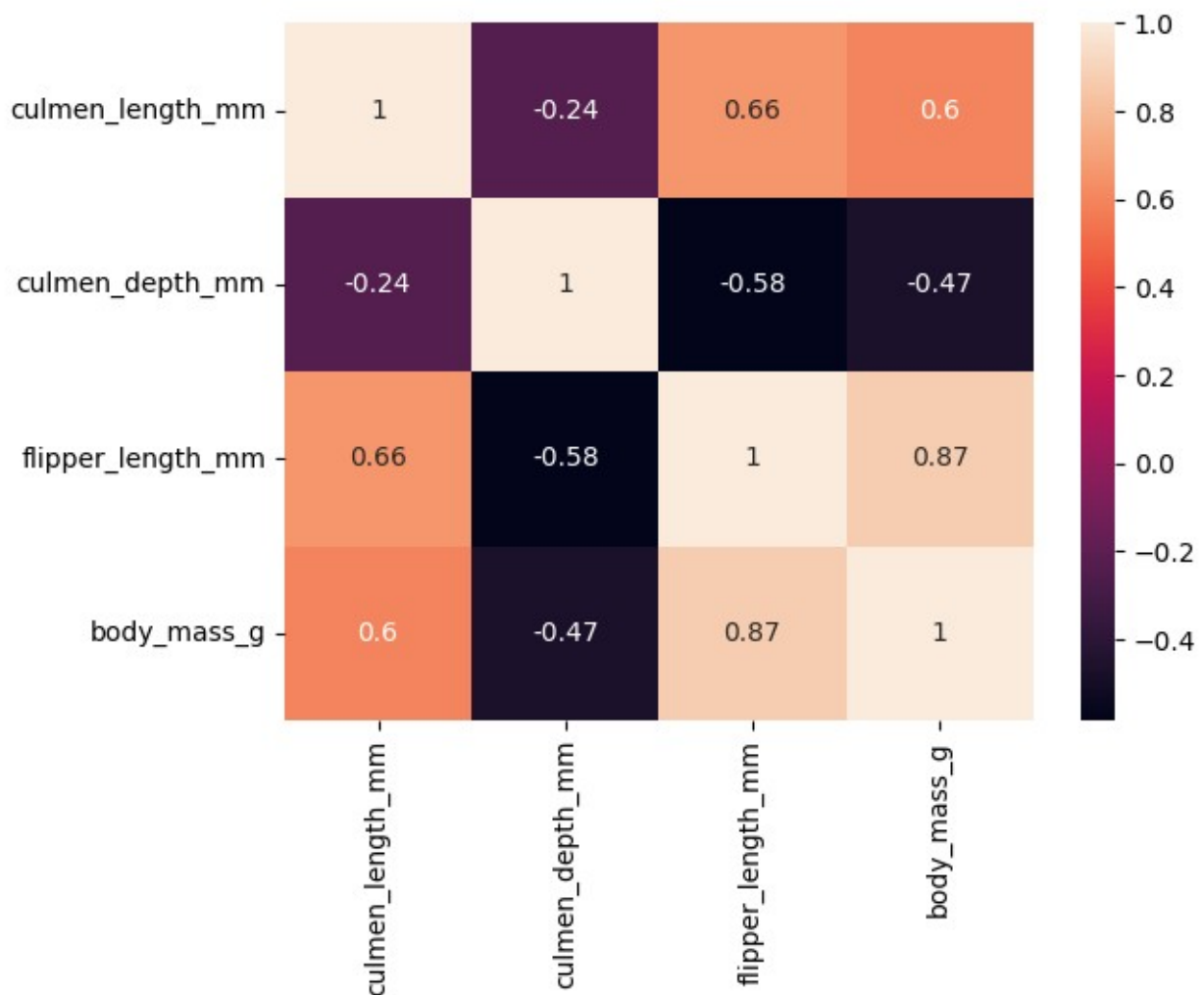
```
          body_mass_g
culmen_length_mm    0.595110
culmen_depth_mm     -0.471916
flipper_length_mm   0.871202
body_mass_g         1.000000
```

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

```
<ipython-input-11-8df7bcac526d>:1: FutureWarning: The default value of
numeric_only in DataFrame.corr is deprecated. In a future version, it
will default to False. Select only valid columns or specify the value
of numeric_only to silence this warning.
```

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

```
<Axes: >
```



"Species" is the target value

```
df.corr().species.sort_values(ascending=False)
```

```
species          1.000000
flipper_length_mm 0.854307
body_mass_g       0.750491
culmen_length_mm  0.728706
sex              -0.010379
island            -0.635659
culmen_depth_mm   -0.744076
Name: species, dtype: float64
```

8. Check for Categorical columns and perform encoding.

```
from sklearn.preprocessing import LabelEncoder
le = LabelEncoder()
```

"sex" column is categorical

```
df.sex = le.fit_transform(df.sex)
df.species = le.fit_transform(df.species)
df.island = le.fit_transform(df.island)
df.head()
```

	species	island	culmen_length_mm	culmen_depth_mm
0	0	2	39.10	18.7
1	0	2	39.50	17.4
2	0	2	40.30	18.0
3	0	2	44.45	NaN
4	0	2	36.70	19.3

	body_mass_g	sex
0	3750.0	2
1	3800.0	1
2	3250.0	1
3	NaN	3
4	3450.0	1

The "sex , species , island" column has been encoded

9.Split the data into dependent and independent variables.

```
y = df['species']
y.head()

0    0
1    0
2    0
3    0
4    0
Name: species, dtype: int64

X =df.drop(columns =['species'],axis =1)
X.head()
```

	island	culmen_length_mm	culmen_depth_mm	flipper_length_mm
0	2	39.10	18.7	181.0
1	2	39.50	17.4	186.0
2	2	40.30	18.0	195.0
3	2	44.45	17.3	197.0
4	2	36.70	19.3	193.0

	sex
0	2
1	1
2	1
3	2
4	1

10.Scaling the data

```
from sklearn.preprocessing import MinMaxScaler
scale =MinMaxScaler()
```



```
X_scaled = pd.DataFrame(scale.fit_transform(X),columns = X.columns)
X_scaled.head()
```

	island	culmen_length_mm	culmen_depth_mm	flipper_length_mm	body_mass_g \
0	1.0	0.254545	0.666667	0.152542	0.291667
1	1.0	0.269091	0.511905	0.237288	0.305556
2	1.0	0.298182	0.583333	0.389831	0.152778
3	1.0	0.449091	0.500000	0.423729	0.375000
4	1.0	0.167273	0.738095	0.355932	0.208333

	sex
0	1.0
1	0.5
2	0.5
3	1.0
4	0.5

11. 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=0.2,random_state=10)
```

12.check the training and testing data shape.

```
X_train.shape
```

```
(275, 6)
```

```
X_train.head()
```

	island	culmen_length_mm	culmen_depth_mm	flipper_length_mm	body_mass_g \	sex
315	0	50.8	15.7	226.0	5200.0	2
181	1	52.8	20.0	205.0	4550.0	2
25	0	35.3	18.9	187.0		
164	1	47.0	17.3	185.0		
247	0	47.8	15.0	215.0		

```
25      3800.0    1
164     3700.0    1
247     5650.0    2
```

```
y_train.shape
```

```
(275,)
```

```
y_train.head()
```

```
315    2
181    1
25     0
164    1
247    2
```

```
Name: species, dtype: int64
```

```
X_test.shape
```

```
(69, 6)
```

```
X_test.head()
```

	island	culmen_length_mm	culmen_depth_mm	flipper_length_mm	\
229	0	46.8	15.4	215.0	
80	2	34.6	17.2	189.0	
327	0	53.4	15.8	219.0	
6	2	38.9	17.8	181.0	
309	0	52.1	17.0	230.0	

	body_mass_g	sex
229	5150.0	2
80	3200.0	1
327	5500.0	2
6	3625.0	1
309	5550.0	2

```
y_test.shape
```

```
(69,)
```

```
y_test.head()
```

```
229    2
80     0
327    2
6      0
309    2
```

```
Name: species, dtype: int64
```