NAME: SHUVAM JENA
REG:21BAI1131

[1]:

#####1. penguins_size.csv is downloaded

#####2. Load the dataset into the tool.

[2]:

```
[2]:   species     island    culmen_length_mm   culmen_depth_mm   flipper_length_mm    \
   •   Adelie   Torgersen               39.1              18.7               181.0
   •   Adelie   Torgersen               39.5              17.4               186.0
   •   Adelie   Torgersen               40.3              18.0               195.0
   •   Adelie   Torgersen                NaN               NaN
       NaN
   •   Adelie   Torgersen               36.7              19.3               193.0

       body_mass_g     sex
   0        3750.0    MALE
   •        3800.0  FEMALE
   •        3250.0  FEMALE
   •           NaN     NaN
   •        3450.0  FEMALE
```

#####3.1. Perform Univariate Analysis

[3]:

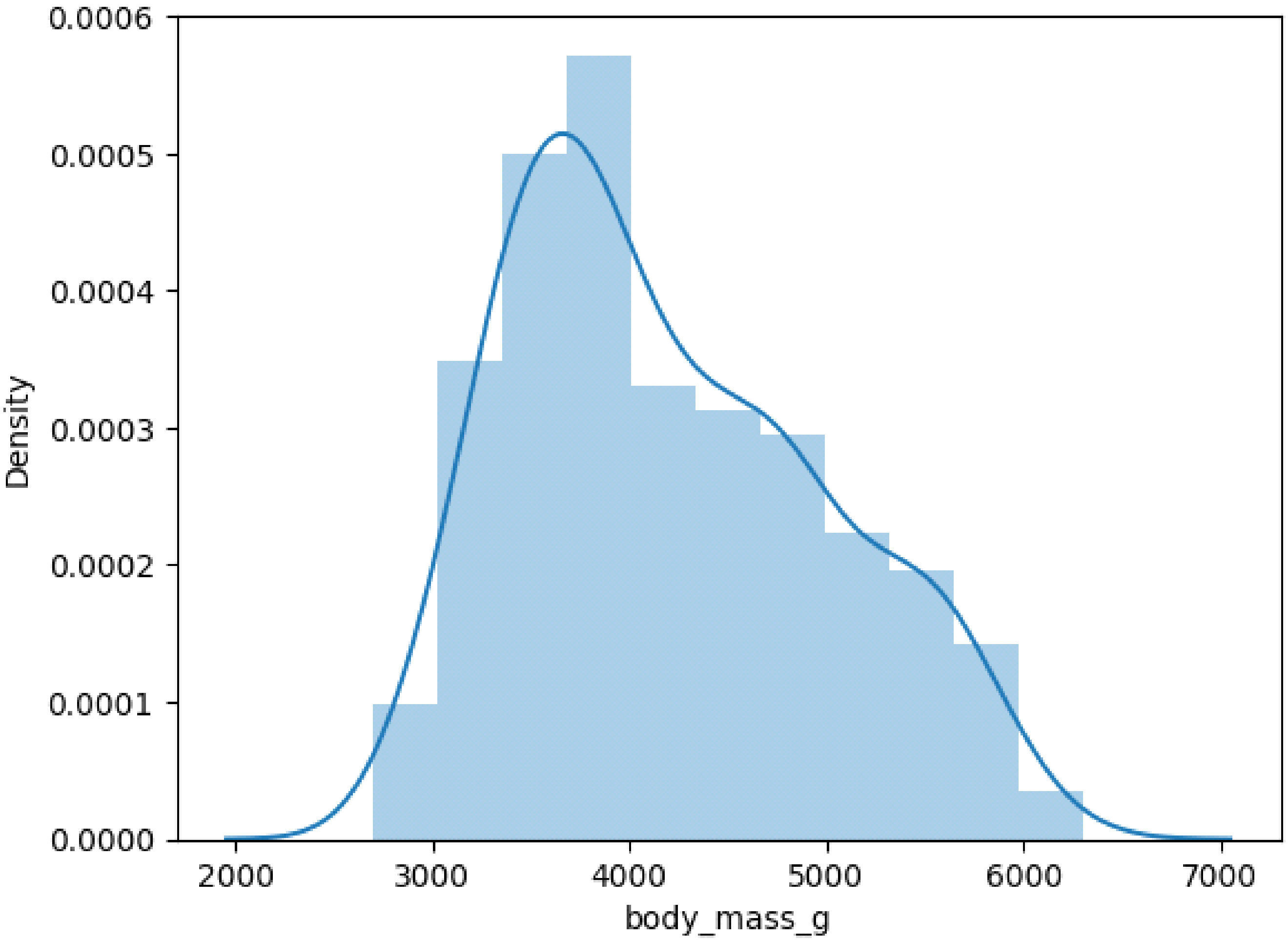
[4]:

```
<ipython-input-4-176964dae727>: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.body_mass_g)
```

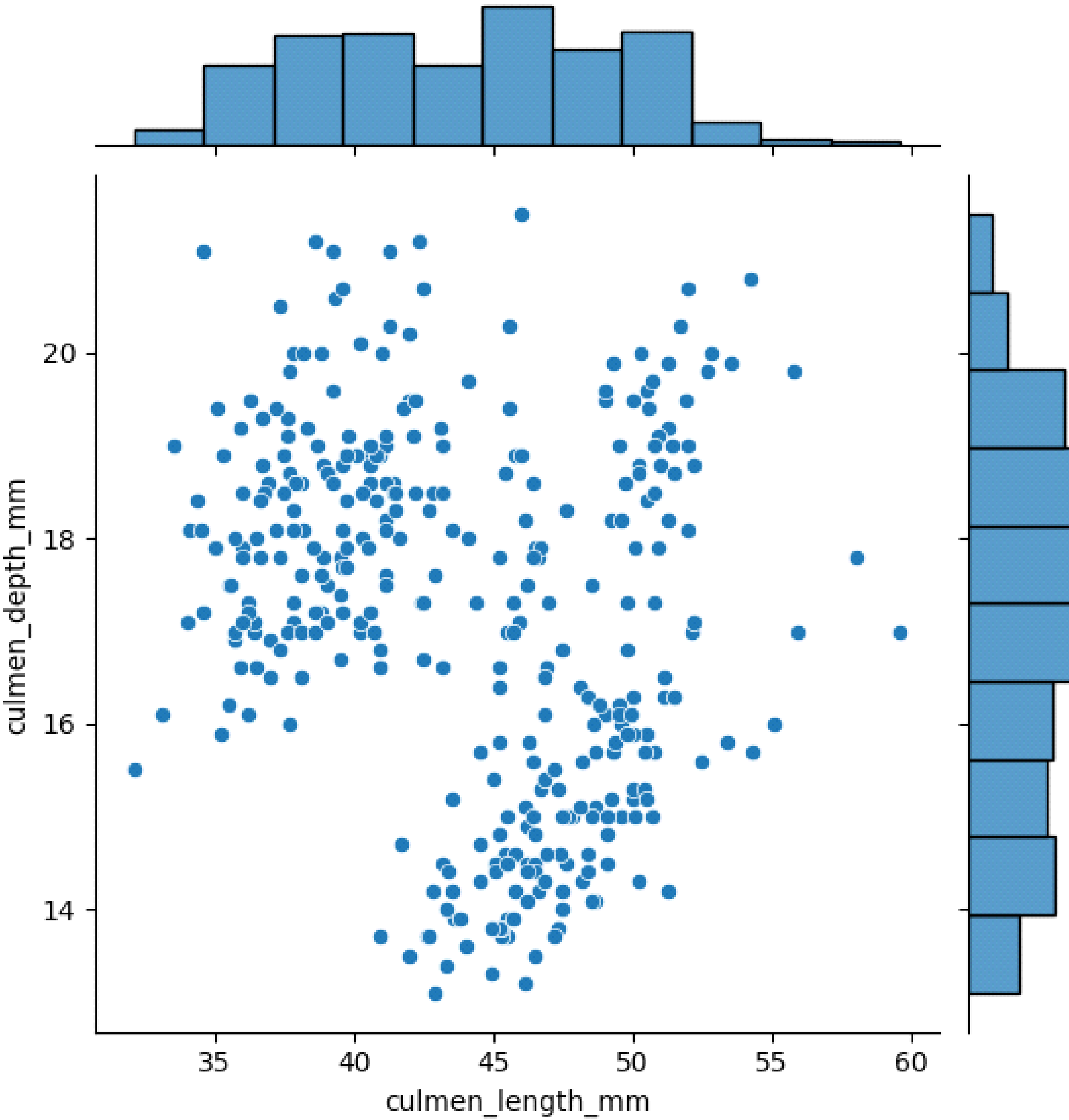
- : <Axes: xlabel='body_mass_g', ylabel='Density'>



#####3.2. Perform Bivariate Analysis

[5]:

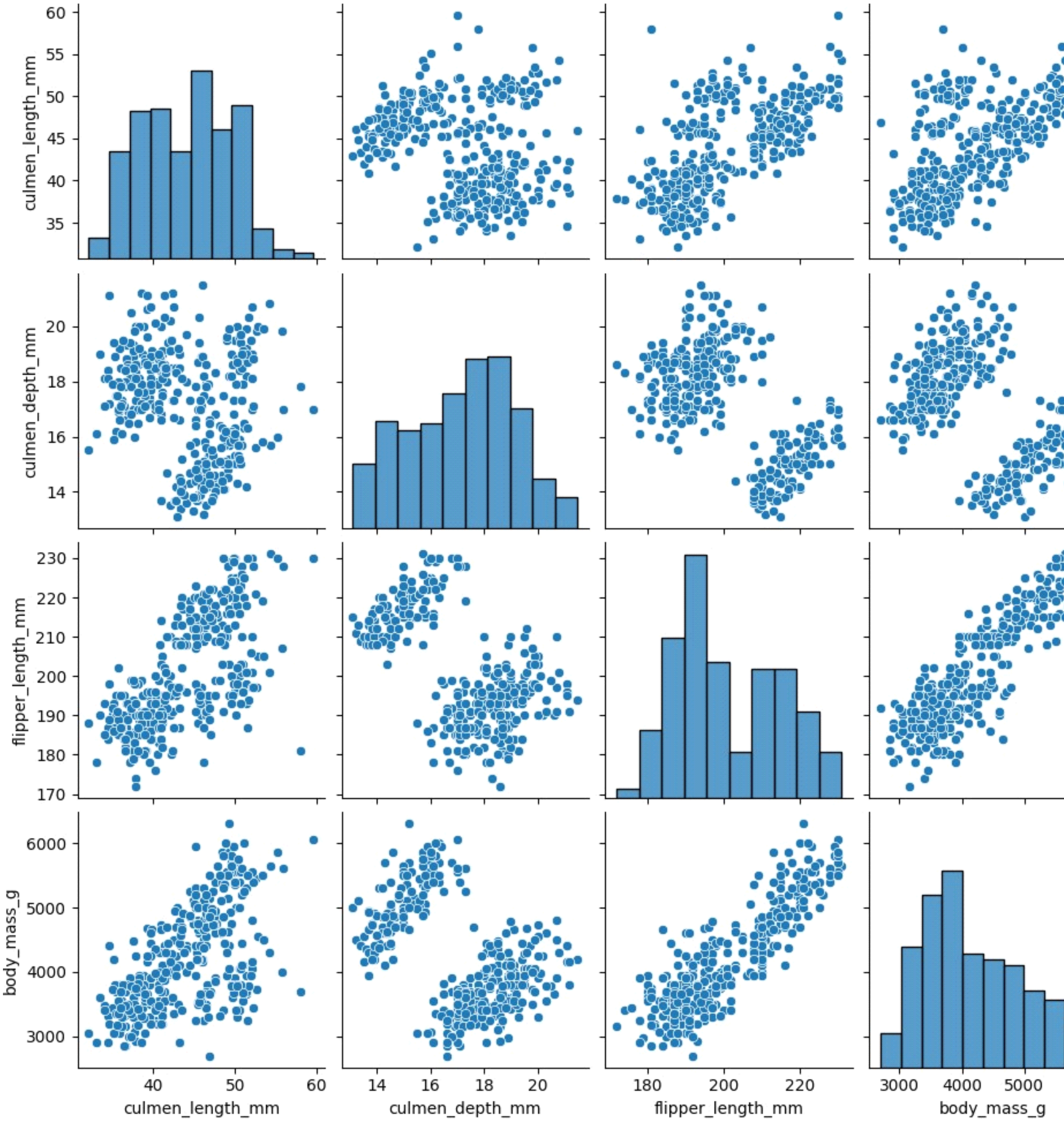
- : <seaborn.axisgrid.JointGrid at 0x7c313325c6a0>



#####3.3. Perform Multi-Variate Analysis

[6]:

- : <seaborn.axisgrid.PairGrid at 0x7c31298f71f0>



#####4. Perform descriptive statistics on the dataset.

[7]:

[7]:	culmen_length_mm	culmen_depth_mm	flipper_length_mm	body_mass_g
count	342.000000	342.000000	342.000000	342.000000

mean	43.921930	17.151170	200.915205	4201.754386
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.

[8]:

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

[9]:

```
[9]:  species              0
        island             0
        culmen_length_mm   2
        culmen_depth_mm    2
        flipper_length_mm  2
        body_mass_g        2
        sex               10
        dtype:  int64
```

```
# Code to replace null values in numerical columns with MEDIAN
df['culmen_length_mm'].fillna(df['culmen_length_mm'].median(),inplace=True)
df['culmen_depth_mm'].fillna(df['culmen_depth_mm'].median(),inplace=True)
df['flipper_length_mm'].fillna(df['flipper_length_mm'].median(),inplace=True)
df['body_mass_g'].fillna(df['body_mass_g'].median(),inplace=True)

# Code to replace null values in categorical column with MODE
df['sex'].fillna(df['sex'].mode().iloc[0],inplace=True)
```

[25]:

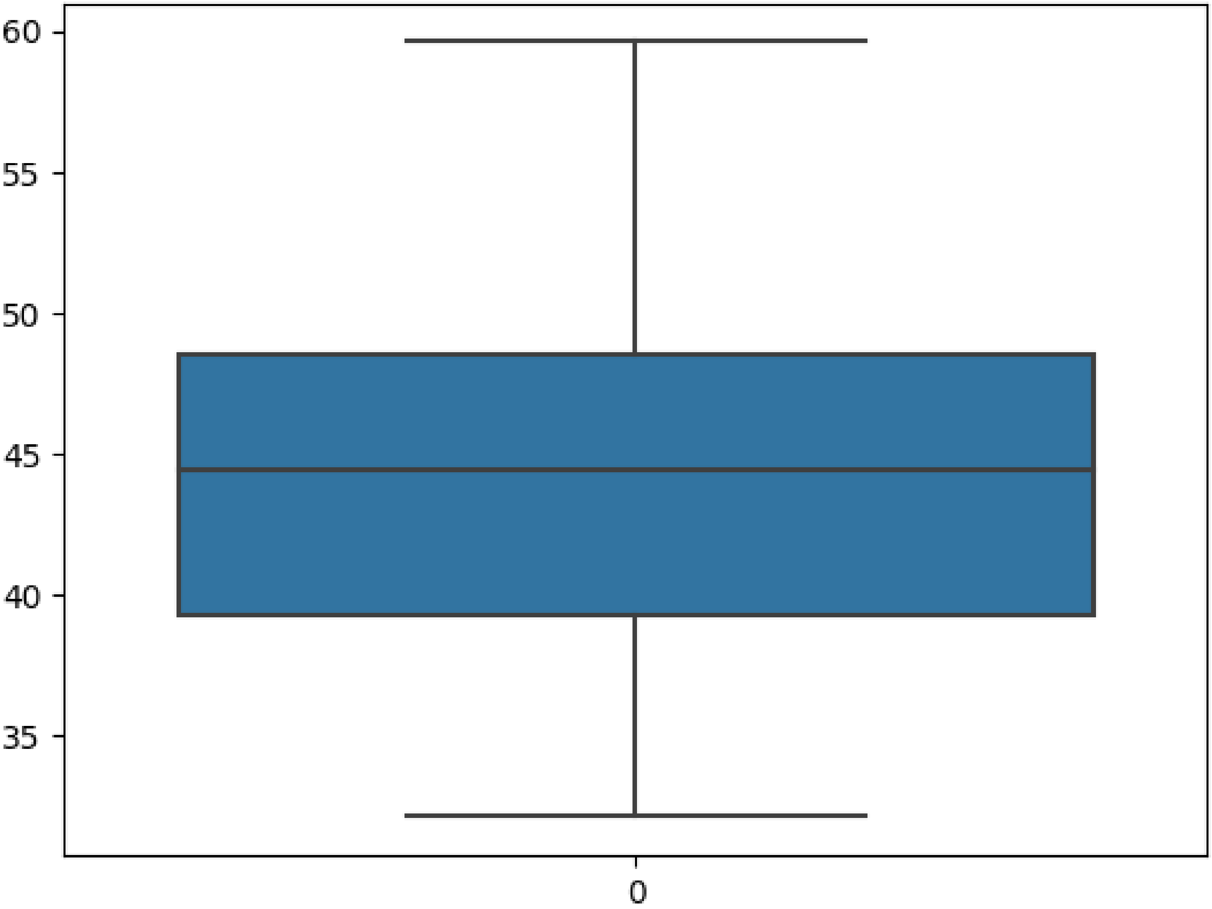
[26]:

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

#####6. Find the outliers and replace the outliers

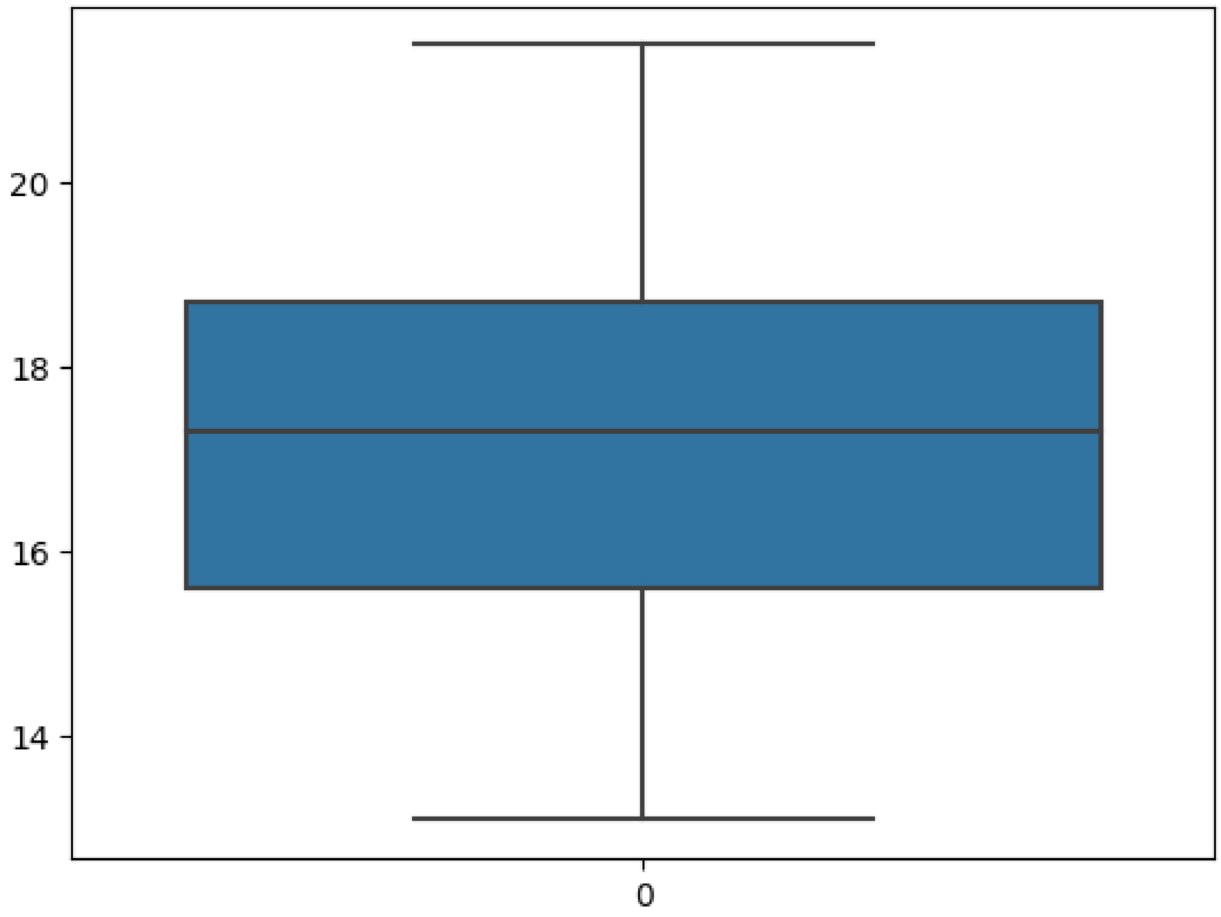
[29]:

- : <Axes: >



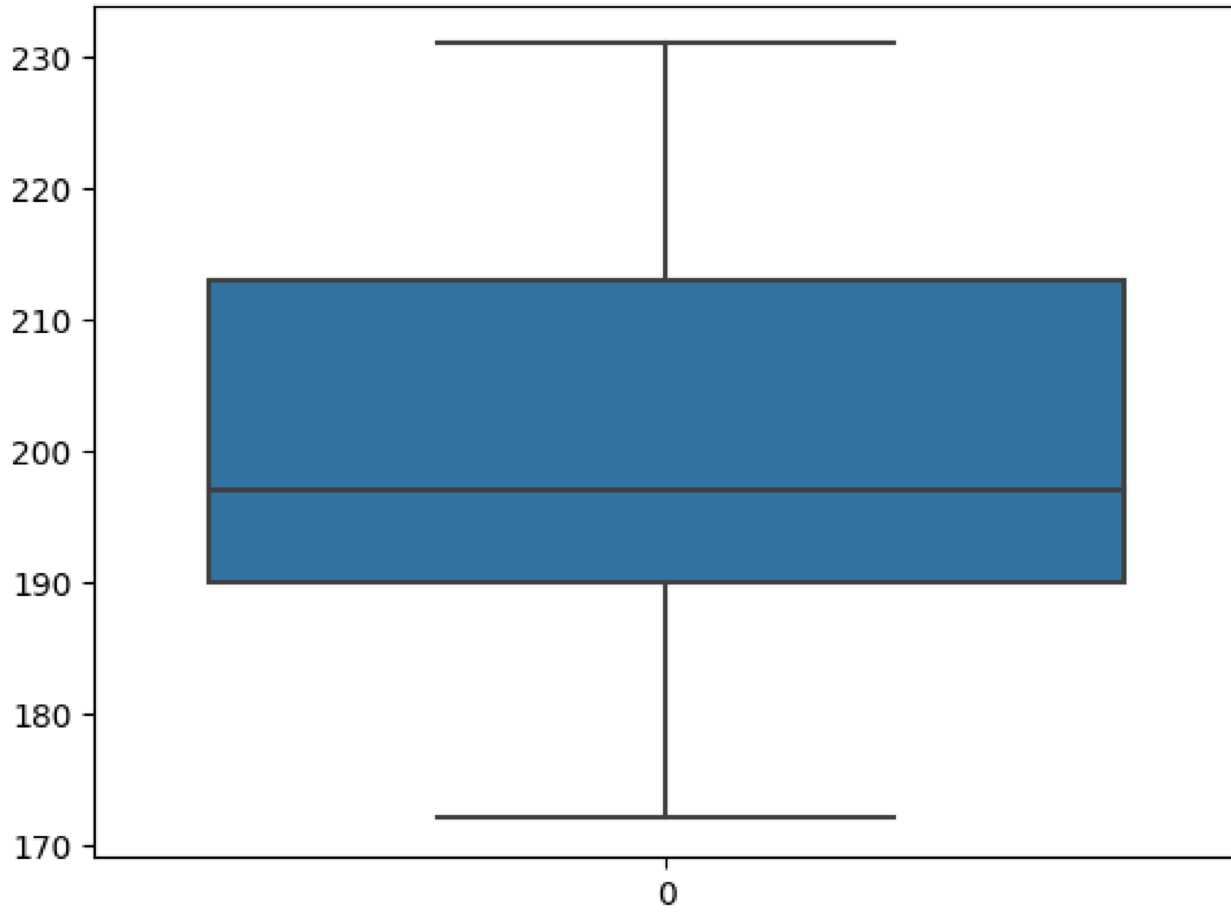
[30]:

- : <Axes: >



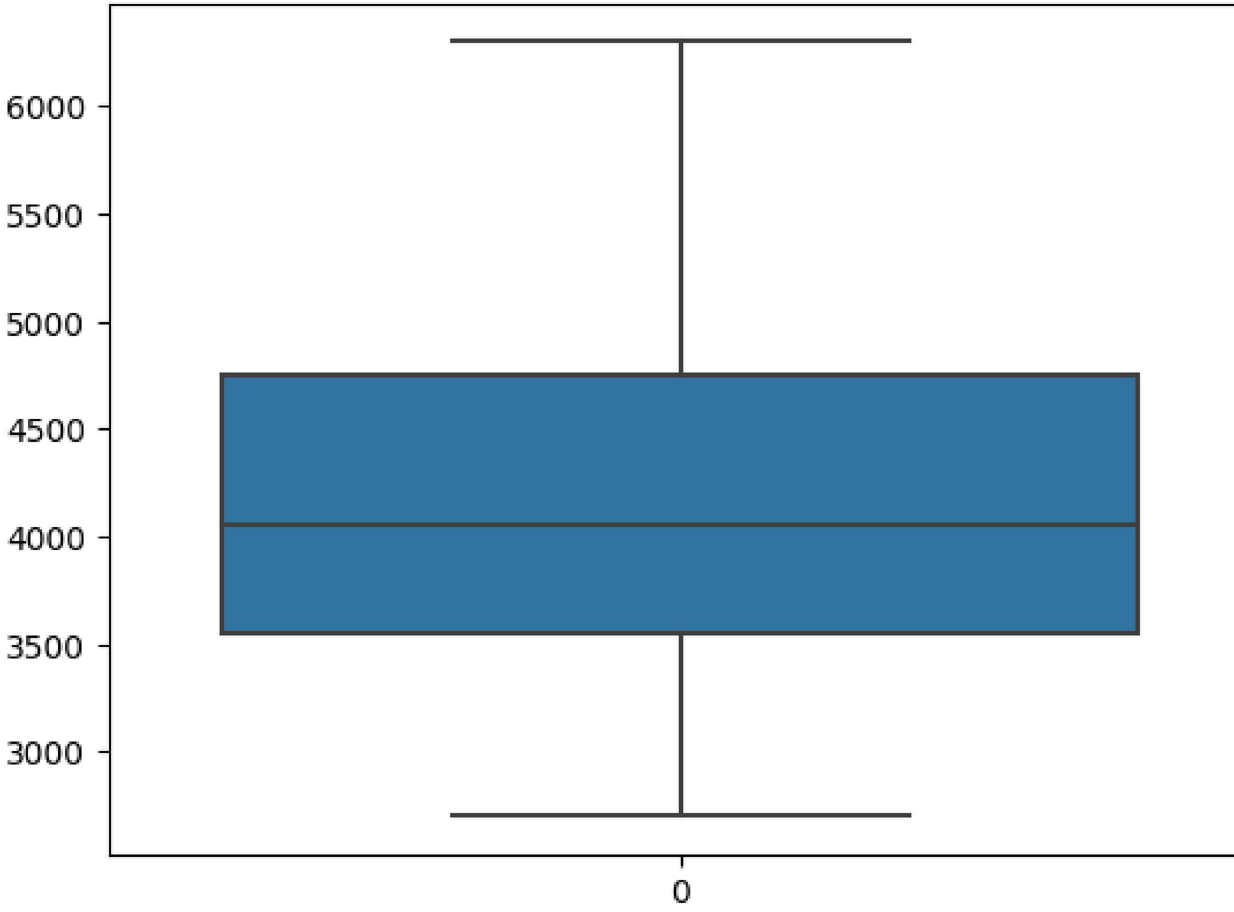
[31]:

- : <Axes: >



[32]:

- : <Axes: >



######Hence there are no outliers in the dataset.

#####7. Check for Categorical columns and perform encoding.

[37]:

[37]:

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

```
     body_mass_g    sex
0           3750.0        2
1           3800.0        1
2           3250.0        1
3           4050.0        2
```

[38]:

```
4            3450.0        1
```

#####8. Check the correlation of independent variables with the target (TARGET IS SPECIES and remaining are independent)

[38]: species	1.000000
flipper_length_mm	0.850819
body_mass_g	0.747547
culmen_length_mm	0.728706
sex	-0.003823
island	-0.635659
culmen_depth_mm	-0.741282

Name: species, dtype: float64

#####9. Split the data into dependent and independent variables

[40]:

```
[40]:        island      culmen_length_mm     culmen_depth_mm    flipper_length_mm
             body_mass_g                           \
```

0		2	39.10	18.7	181.0	3750.0
1		2	39.50	17.4	186.0	3800.0
2		2	40.30	18.0	195.0	3250.0
3		2	44.45	17.3	197.0	4050.0
4		2	36.70	19.3	193.0	3450.0

	sex
0	2
1	1
2	1
3	2

4	1	

[41]:

[41]:	0	0
	1	0
	2	0
	3	0
	4	0

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

#####10. Scaling the data

[42]:

[42]:

```
   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

[48]:

#####12. Check the training and testing data shape.

[49]:

[49]: (275, 6)

[50]:

```
[50]:  (69,  6)

[51]:

[51]:  (275,)

[52]:

[52]:  (69,)
```

#The End!!!