

Submission Date - 13.09.2023

ASSIGNMENT – 3

Artificial Intelligence & Machine Learning in
collaboration with Google (Applied Data Science)

Name: Rishikesh S

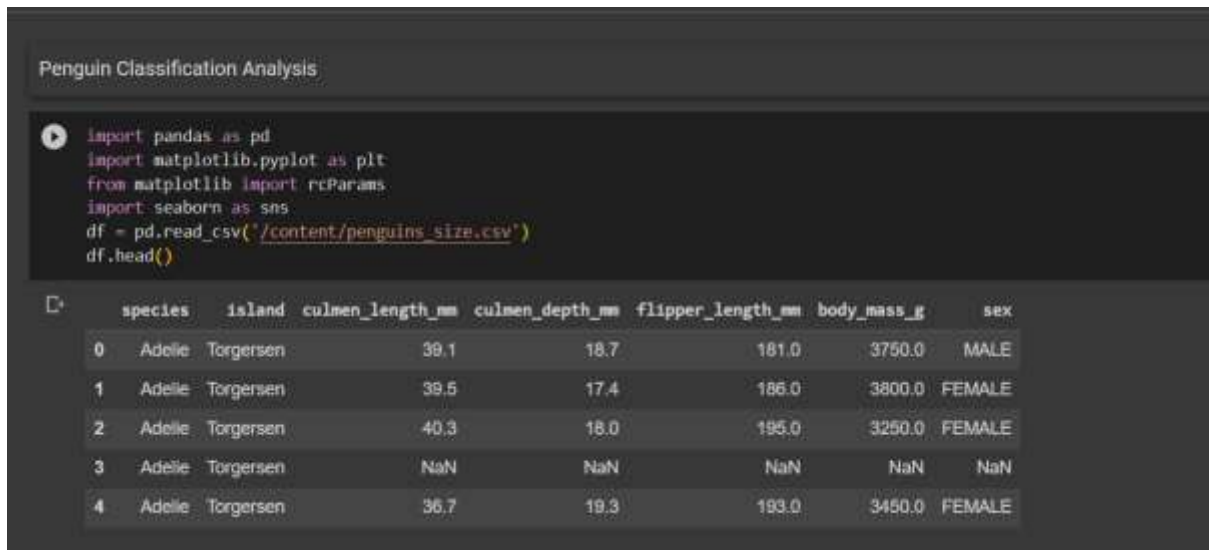
Regno: 21BME0159

Branch: Btech Mechanical Engineering

Campus: VIT Vellore

TASK

1. Download the dataset: Dataset
2. Load the dataset into the tool.



The screenshot shows a Jupyter Notebook interface with a dark theme. The title bar reads "Penguin Classification Analysis". The code cell contains the following Python code:

```
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()
```

Below the code cell, the output shows the first five rows of the DataFrame. The columns are: index, species, island, culmen_length_mm, culmen_depth_mm, flipper_length_mm, body_mass_g, and sex.

	species	island	culmen_length_mm	culmen_depth_mm	flipper_length_mm	body_mass_g	sex
0	Adelie	Torgersen	39.1	18.7	181.0	3750.0	MALE
1	Adelie	Torgersen	39.5	17.4	186.0	3800.0	FEMALE
2	Adelie	Torgersen	40.3	18.0	195.0	3250.0	FEMALE
3	Adelie	Torgersen	NaN	NaN	NaN	NaN	NaN
4	Adelie	Torgersen	36.7	19.3	193.0	3450.0	FEMALE

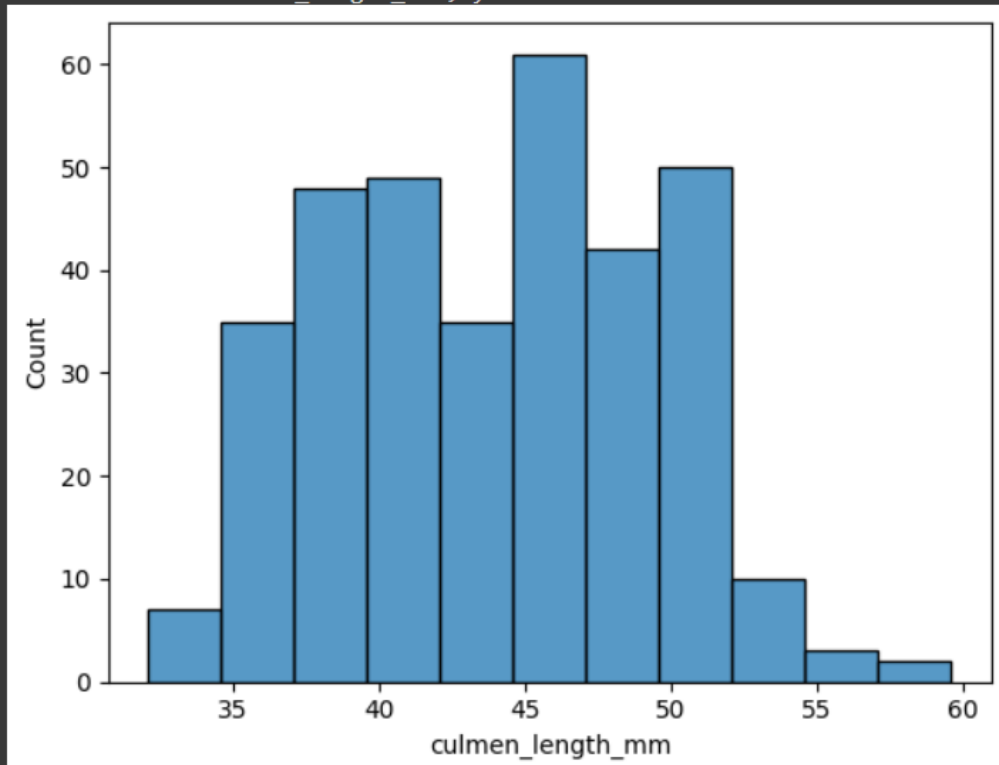
3. Perform Below Visualizations.
- Univariate Analysis

Univariate Analysis



```
sns.histplot(df['culmen_length_mm'])
```

<Axes: xlabel='culmen_length_mm', ylabel='Count'>





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



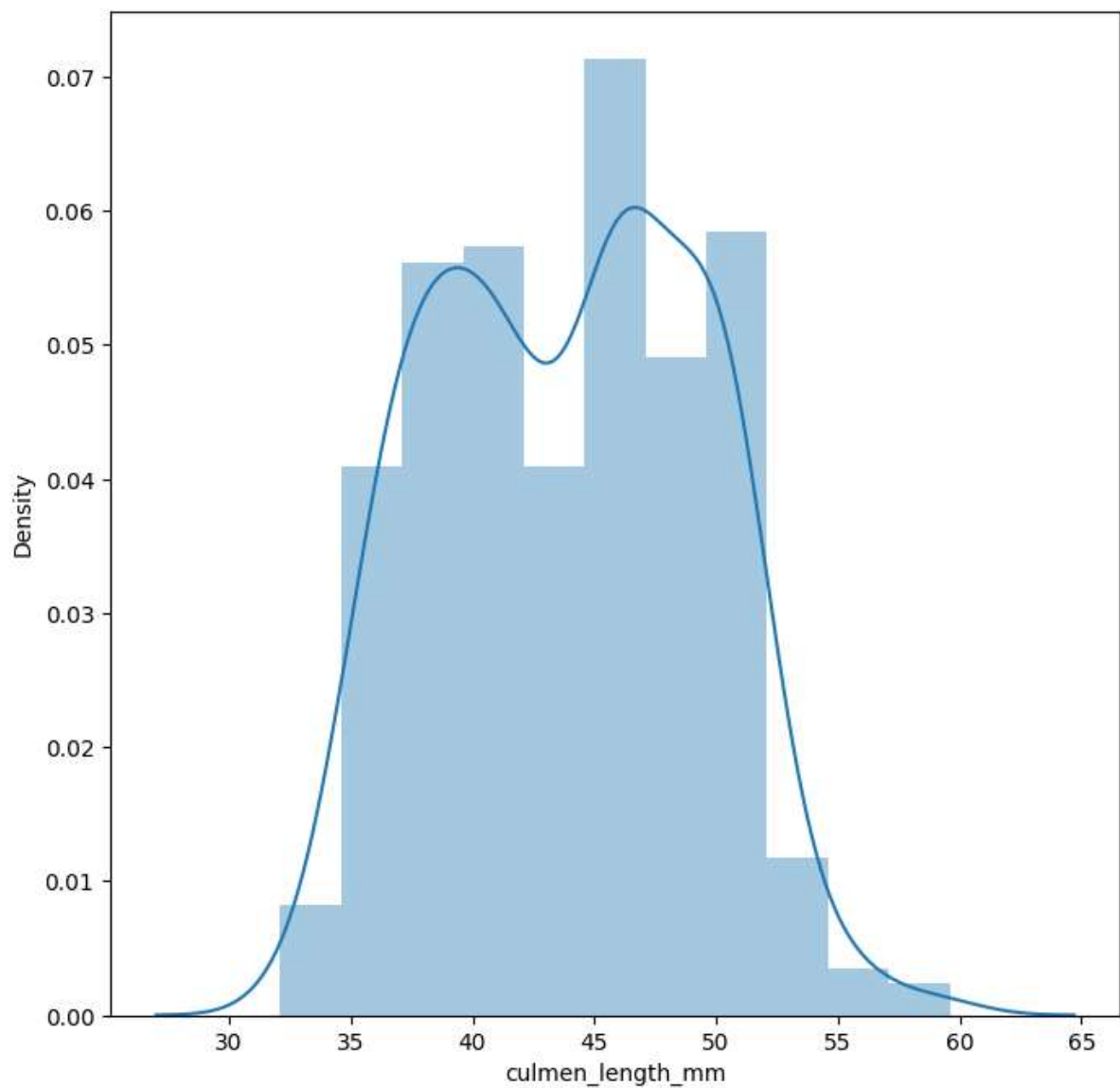
```
<ipython-input-15-24e9b5890c61>:1: UserWarning:
```

```
`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).
```

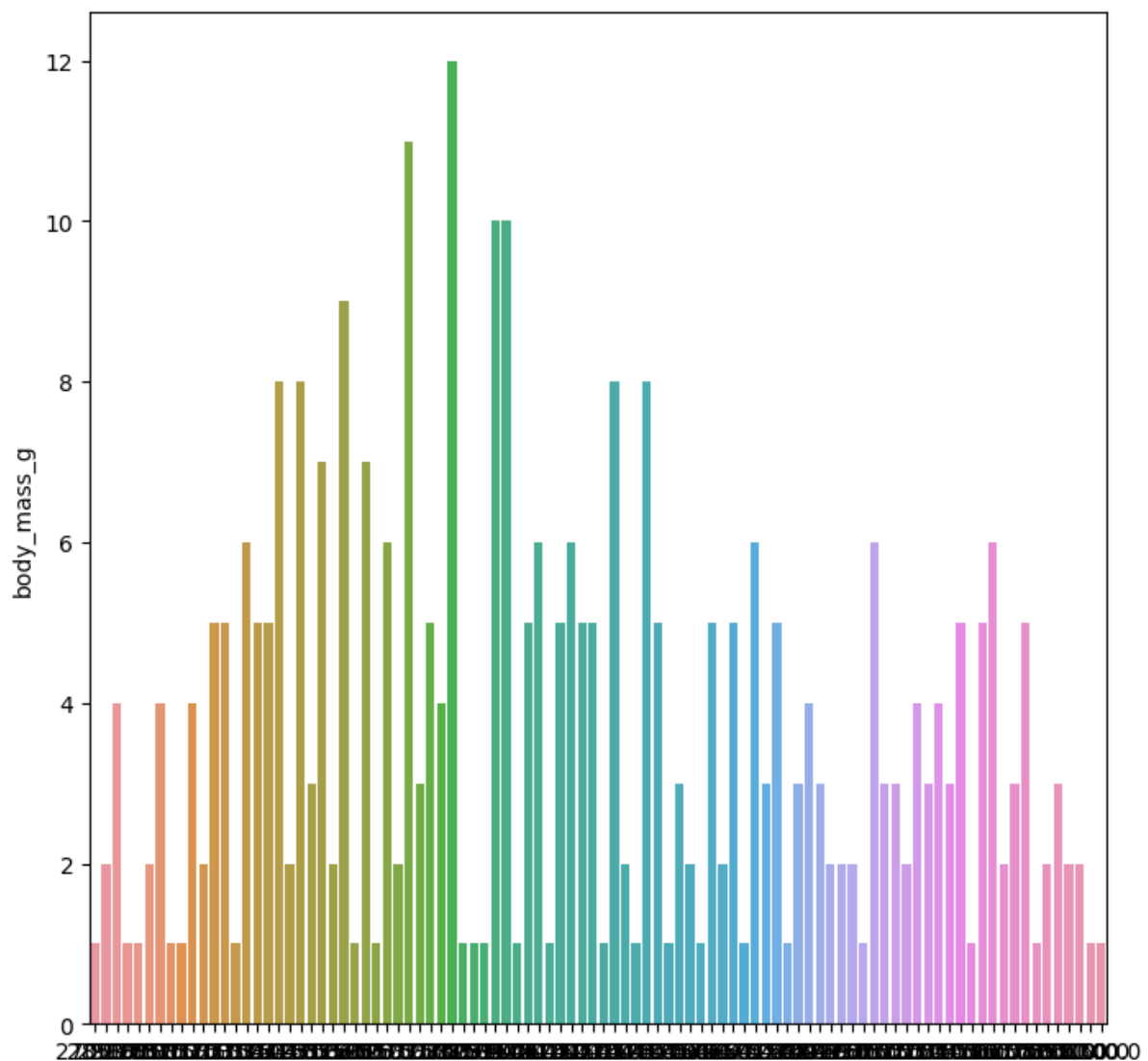
```
For a guide to updating your code to use the new functions, please see https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751
```

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



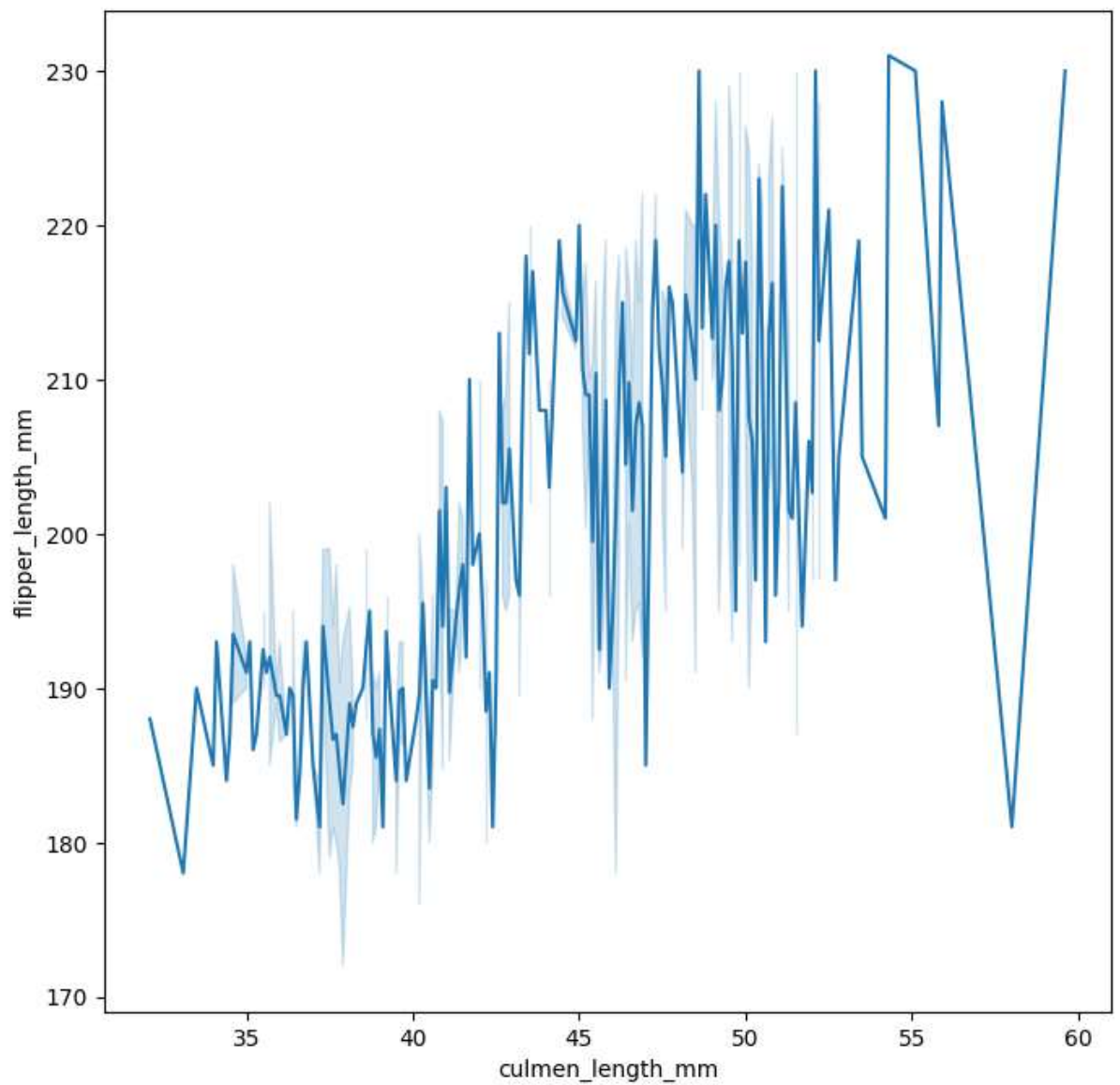
```
sns.barplot(x=df.body_mass_g.value_counts().index,y=df.body_mass_g.value_counts() )
```

<Axes: ylabel='body_mass_g'>

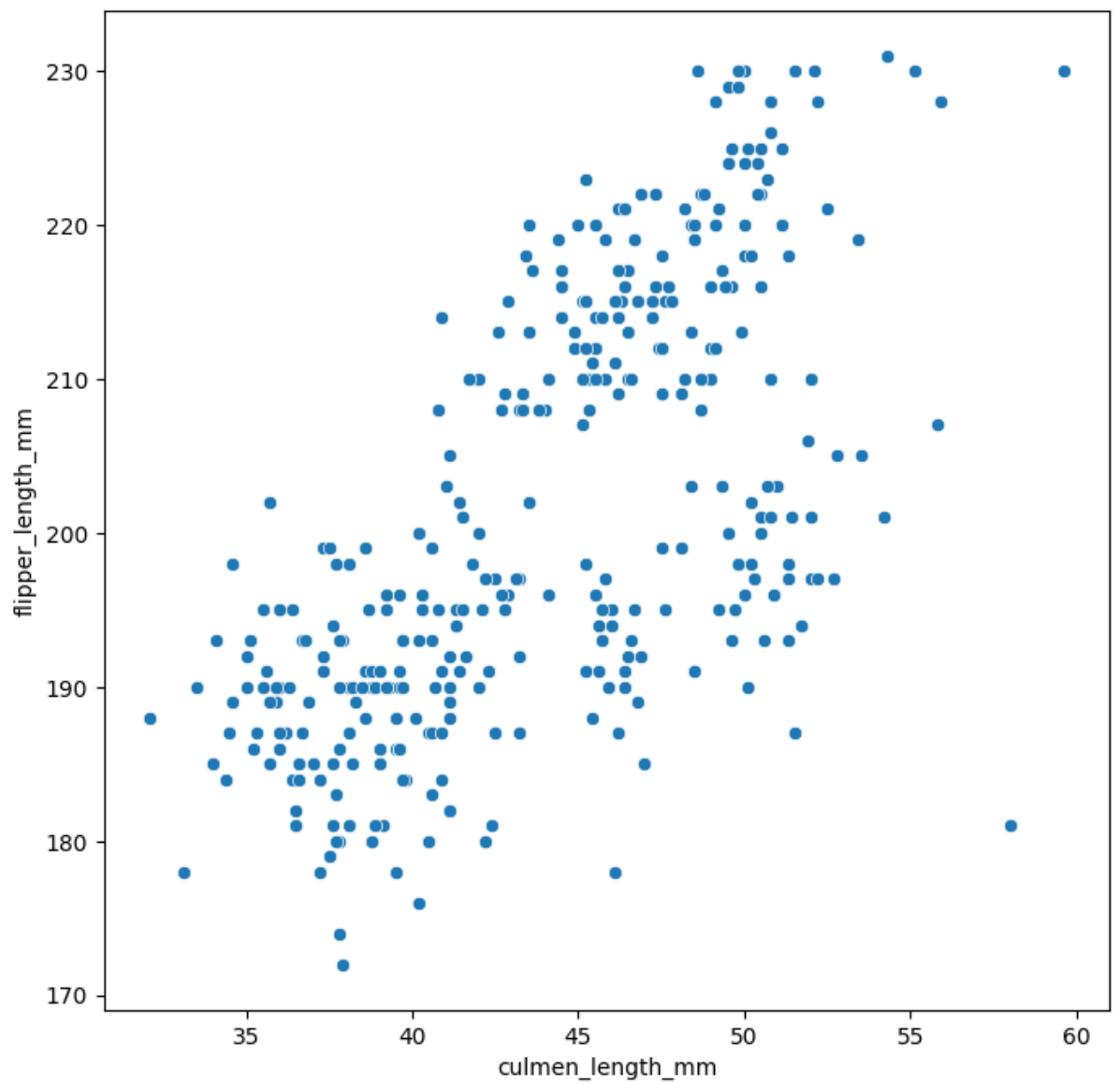


- Bi- Variate Analysis

```
sns.lineplot(x = df.culmen_length_mm,y=df.flipper_length_mm)
```

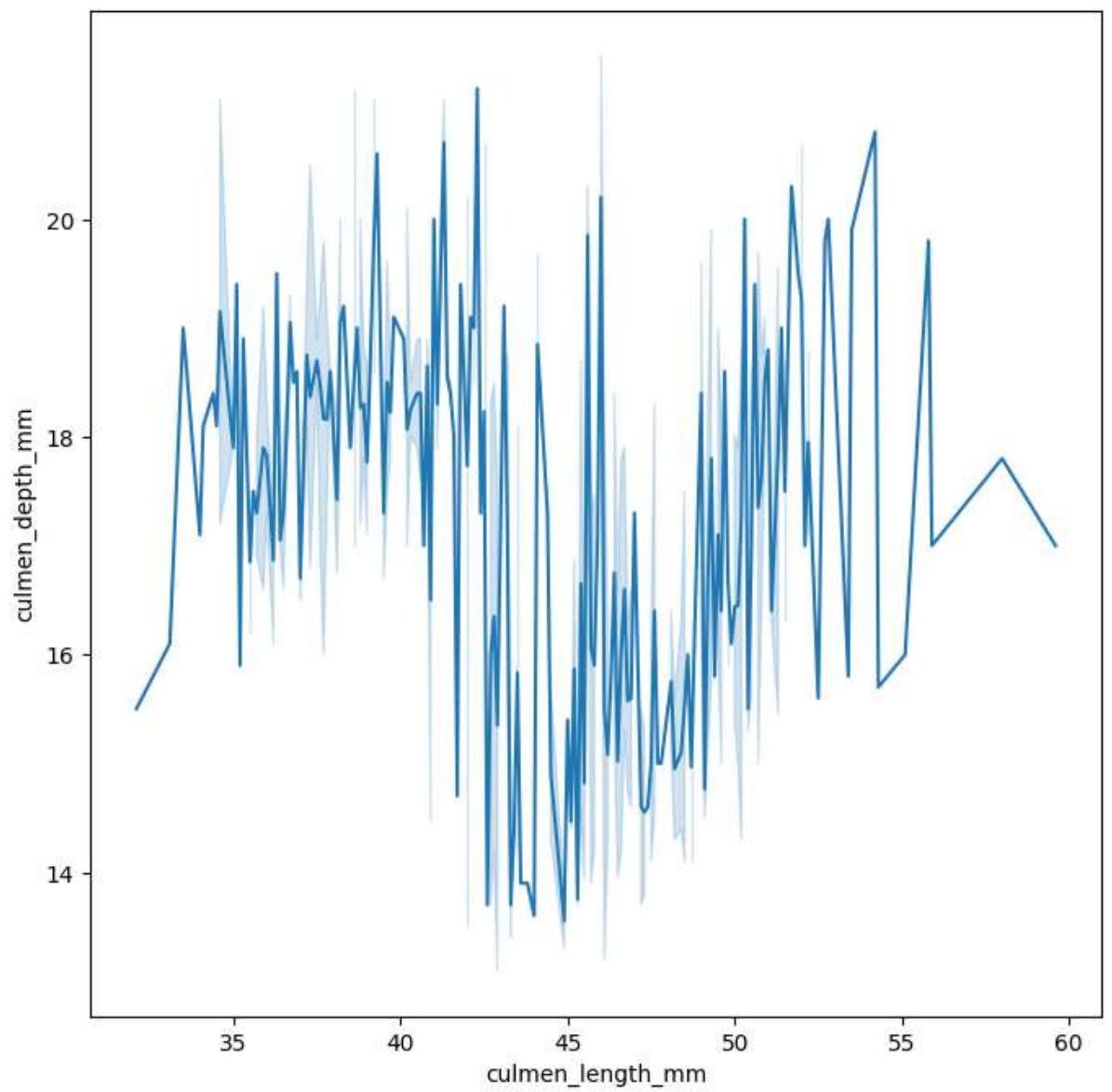


```
sns.scatterplot(x = df.culmen_length_mm,y=df.flipper_length_mm)  
<Axes: xlabel='culmen_length_mm', ylabel='flipper_length_mm'>
```



```
sns.lineplot(x = df.culmen_length_mm,y=df.culmen_depth_mm)
```

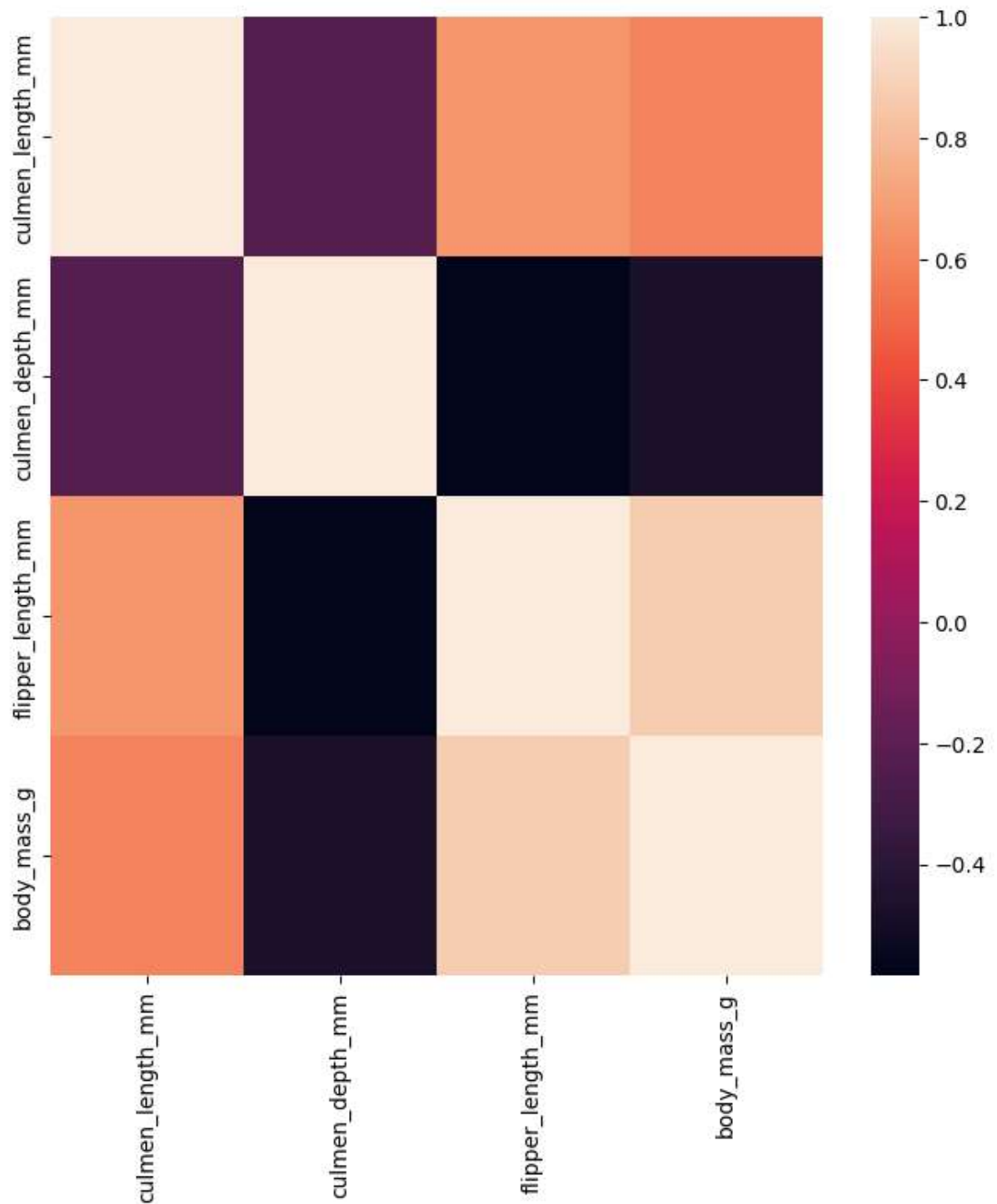
<Axes: xlabel='culmen_length_mm', ylabel='culmen_depth_mm'>



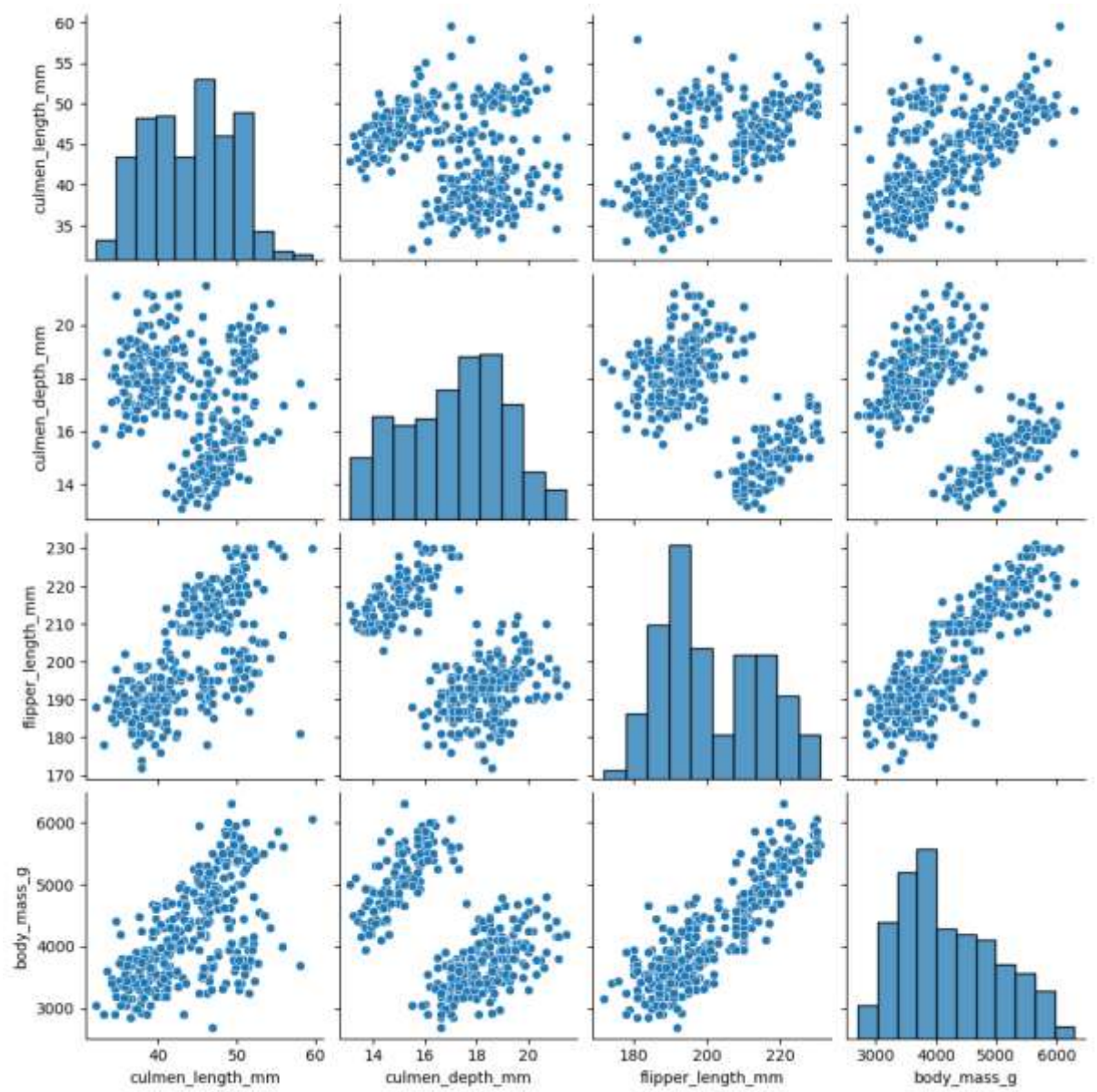
- Multi-Variate Analysis

```
sns.heatmap(df.corr())
```

`Clayton (aput: 0-aaafab0a2a501): 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: 0`



```
sns.pairplot(df)
<seaborn.axisgrid.PairGrid at 0x7c2f895ebb20>
```



4. Perform descriptive statistics on the dataset

4. Perform descriptive statistics on the dataset.

```
stats = df.describe(include = 'all')  
print(stats)
```

```
count    species    island    culmen_length_mm    culmen_depth_mm    flipper_length_mm  \  
count      344      344      342.000000      342.000000      342.000000  
unique        3        3          NaN          NaN          NaN  
top    Adelie    Biscoe          NaN          NaN          NaN  
freq      152      168          NaN          NaN          NaN  
mean        NaN      NaN      43.921930      17.151170      200.915205  
std         NaN      NaN       5.459584       1.974793      14.061714  
min         NaN      NaN      32.100000      13.100000      172.000000  
25%         NaN      NaN      39.225000      15.600000      190.000000  
50%         NaN      NaN      44.450000      17.300000      197.000000  
75%         NaN      NaN      48.500000      18.700000      213.000000  
max         NaN      NaN      59.600000      21.500000      231.000000  
  
count    body_mass_g    sex  
count    342.000000    334  
unique          NaN        3  
top          NaN    MALE  
freq          NaN    168  
mean    4201.754386    NaN  
std      801.954536    NaN  
min     2700.000000    NaN  
25%     3550.000000    NaN  
50%     4050.000000    NaN  
75%     4750.000000    NaN  
max     6300.000000    NaN
```

5. Check for Missing values and deal with them.

5. Check for Missing values and deal with them.

```
df.isnull()
```

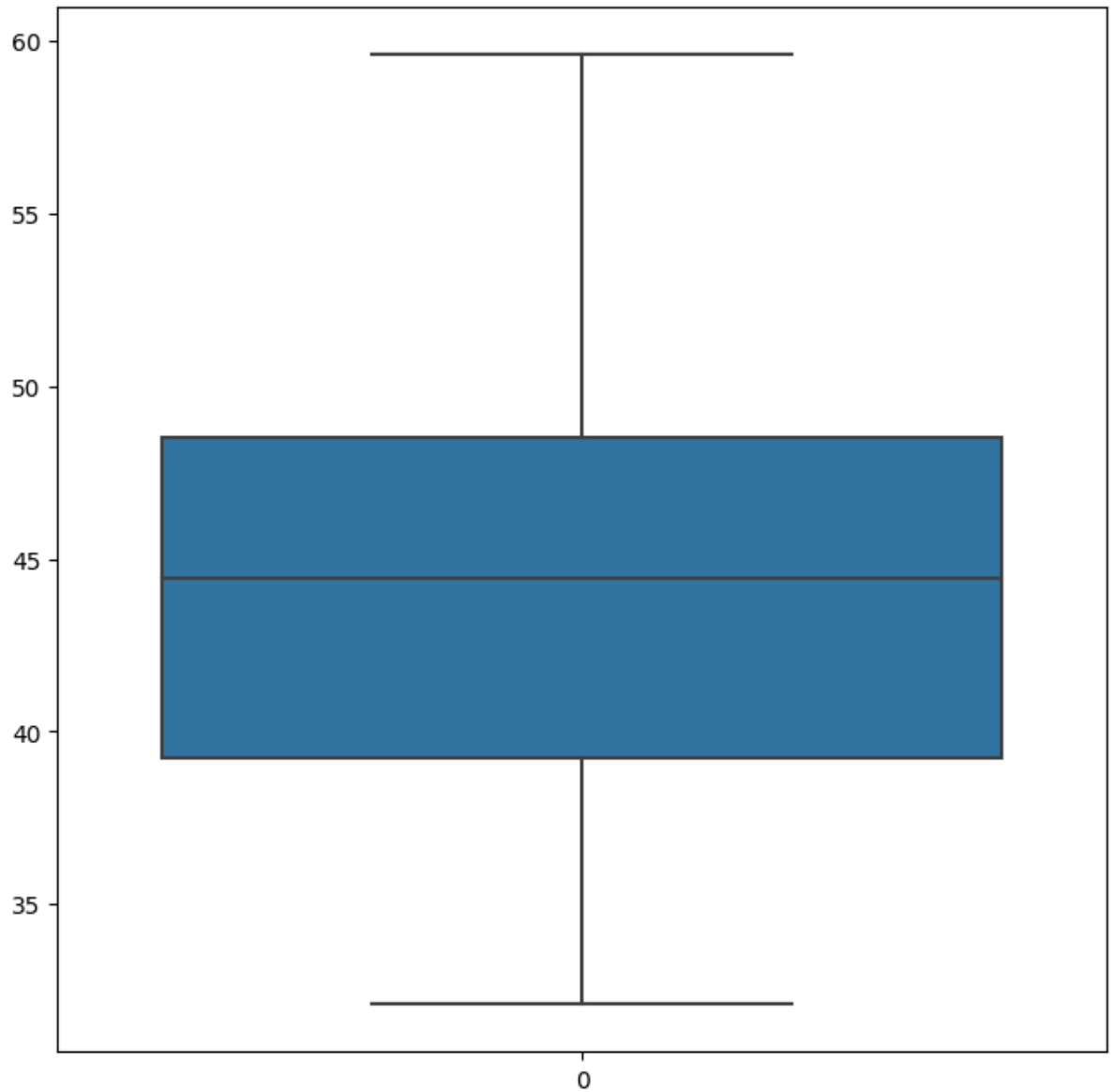
	species	island	culmen_length_mm	culmen_depth_mm	flipper_length_mm	body_mass_g	sex
0	False	False	False	False	False	False	False
1	False	False	False	False	False	False	False
2	False	False	False	False	False	False	False
3	False	False	True	True	True	True	True
4	False	False	False	False	False	False	False
...
339	False	False	True	True	True	True	True
340	False	False	False	False	False	False	False
341	False	False	False	False	False	False	False
342	False	False	False	False	False	False	False
343	False	False	False	False	False	False	False

344 rows × 7 columns

6. Find the outliers and replace them outliers.

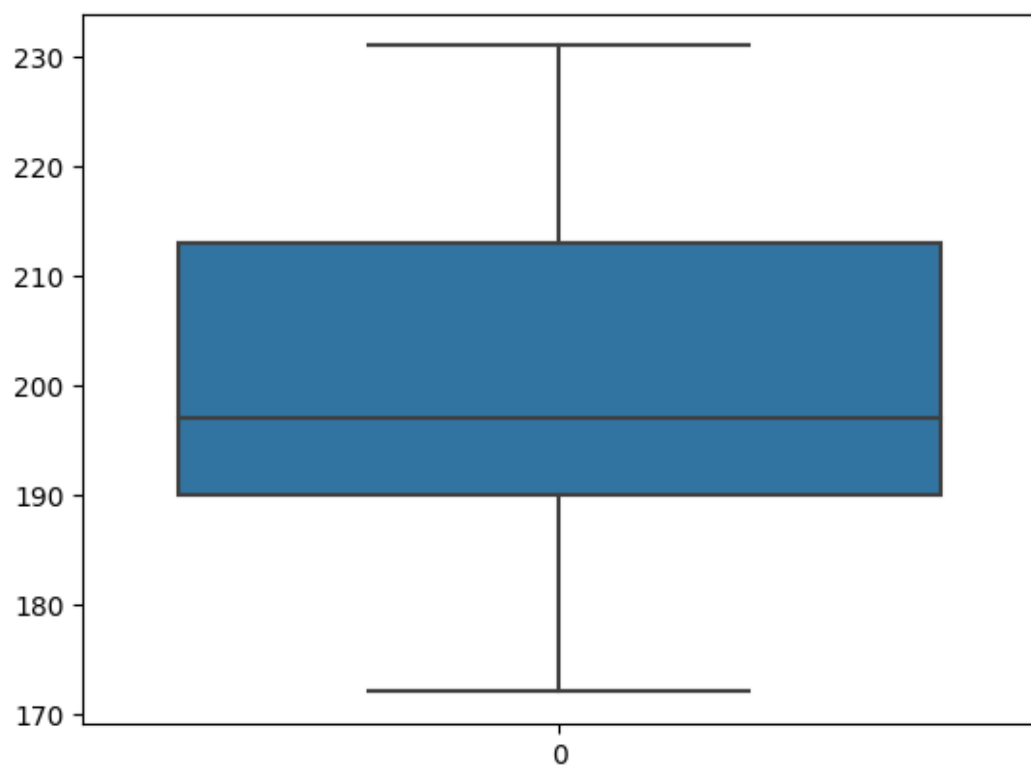
```
sns.boxplot(df.culmen_length_mm)
```

<Axes: >



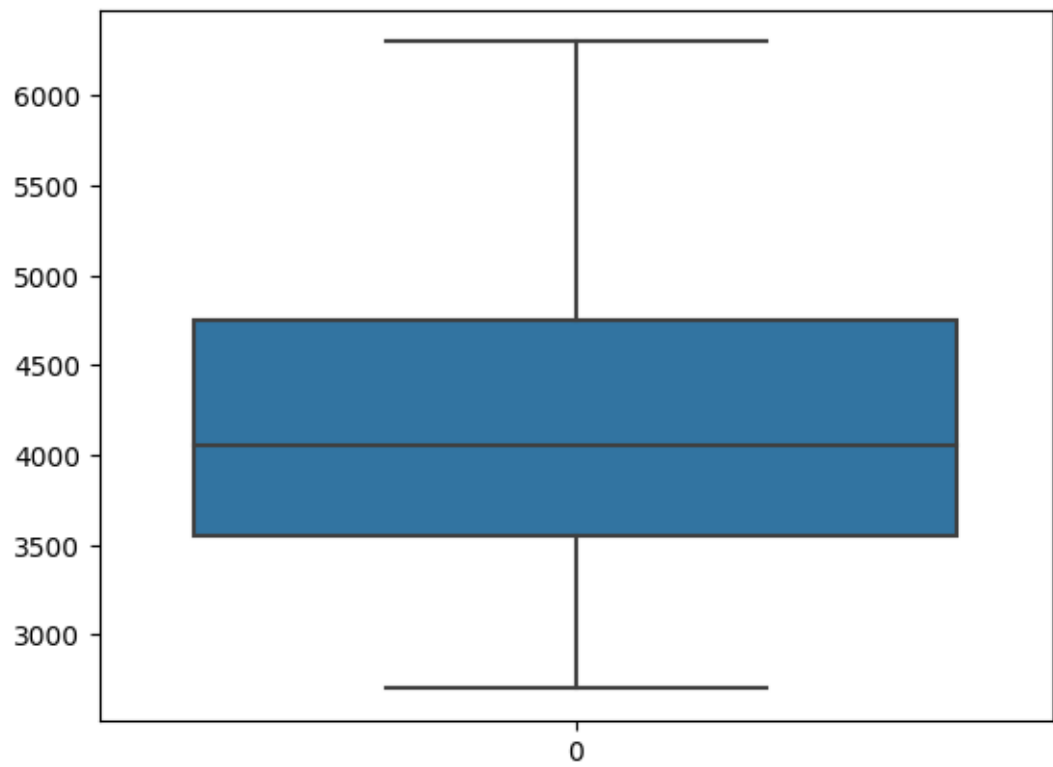


```
sns.boxplot(df.flipper_length_mm)
```





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



7. Check the correlation of independent variables with the target.

```
7. Check the correlation of independent variables with the target.

| | df[df.columns[1:]].corr()['column_length_ms'][1:]

<ipython-input-38-f42f6da421c1>: Out[38]: Out[38]: Warning: the default value of numeric_only in DataFrame.corr is deprecated. In a future version, it will default to False. Select only valid or
df[df.columns[1:]].corr()['column_length_ms'][1:]
column_length_ms      1.000000
column_width_ms       0.239933
flapper_length_ms     0.020181
body_mass_g           0.522130
Name: column_length_ms, dtype: float64
```


8. Check for Categorical columns and perform encoding.

```
8. Check for Categorical columns and perform encoding.
```

```
from sklearn.preprocessing import LabelEncoder
le = LabelEncoder()
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	flipper_length_mm	body_mass_g	sex
0	0	2	39.1	18.7	181.0	3750.0	2
1	0	2	39.5	17.4	186.0	3800.0	1
2	0	2	40.3	18.0	195.0	3250.0	1
3	0	2	NaN	NaN	NaN	NaN	3
4	0	2	36.7	19.3	193.0	3450.0	1

9. Split the data into dependent and independent variables.

9. Split the data into dependent and independent variables.

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

	island	culmen_length_mm	culmen_depth_mm	flipper_length_mm	body_mass_g	sex
0	2	39.1	18.7	181.0	3750.0	2
1	2	39.5	17.4	186.0	3800.0	1
2	2	40.3	18.0	195.0	3250.0	1
3	2	NaN	NaN	NaN	NaN	3
4	2	36.7	19.3	193.0	3450.0	1

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

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

10. Scaling the data

10. Scaling the data.

```
[22] 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	sex
0	1.0	0.254545	0.666667	0.152542	0.291667	0.666667
1	1.0	0.269091	0.511905	0.237288	0.305556	0.333333
2	1.0	0.298182	0.583333	0.389831	0.152778	0.333333
3	1.0	NaN	NaN	NaN	NaN	1.000000
4	1.0	0.167273	0.738095	0.355932	0.208333	0.333333

11. Split the data into training and testing.

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_scaled,Y,test_size=0.2,random_state=0)
```

12.check the training and testing data shape.

```
12. Check the training and testing data shape.  
  
[25] X_train.shape  
(275, 6)  
  
[26] X_test.shape  
(69, 6)  
  
[27] y_train.shape  
(275,)  
  
[28] y_test.shape  
(69,)
```