assignment-3

September 14, 2023

[225]:
```python
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
df=pd.read_csv('/content/penguins_size.csv')
df.head()
```

[225]:
```
  species     island  culmen_length_mm  culmen_depth_mm  flipper_length_mm  \
0  Adelie  Torgersen              39.1             18.7              181.0
1  Adelie  Torgersen              39.5             17.4              186.0
2  Adelie  Torgersen              40.3             18.0              195.0
3  Adelie  Torgersen               NaN              NaN                NaN
4  Adelie  Torgersen              36.7             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
```

[226]:
```python
df.shape
```

[226]:
```
(344, 7)
```

[227]:
```python
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
```

[228]:

```
df.describe()
```

[228]:

```
       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
```

[229]:

```
df.isnull().sum()
```

[229]:

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

[230]:

```
df.sex.value_counts()
```

[230]:

```
MALE      168
FEMALE    165
.           1
Name: sex, dtype: int64
```

[231]:

```
df['sex'] = df['sex'].replace(".","MALE")
```

[232]:

```
df.sex.value_counts()
```

[232]:

```
MALE      169
FEMALE    165
Name: sex, dtype: int64
```

[233]:

```
df['sex']=df['sex'].fillna("MALE")
```

[234]:

```
df.island.value_counts()
```

```
[234]: Biscoe       168
       Dream        124
       Torgersen     52
       Name: island, dtype: int64
```

```
[235]: df.species.value_counts()
```

```
[235]: Adelie       152
       Gentoo       124
       Chinstrap     68
       Name: species, dtype: int64
```

```
[236]: df.isnull().sum()
```

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

```
[237]: df.median()
```

```
<ipython-input-237-6d467abf240d>: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()
```

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

```
[238]: df=df.fillna(df.median())
```

```
<ipython-input-238-42d29455c84b>: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=df.fillna(df.median())
```

[239]:

```
df.isnull().sum()
```

[239]:

```
species              0
island               0
culmen_length_mm     0
culmen_depth_mm      0
flipper_length_mm    0
body_mass_g          0
sex                  0
dtype: int64
```

[240]:

```
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   344 non-null    float64
 3   culmen_depth_mm    344 non-null    float64
 4   flipper_length_mm  344 non-null    float64
 5   body_mass_g        344 non-null    float64
 6   sex                344 non-null    object 
dtypes: float64(4), object(3)
memory usage: 18.9+ KB
```

##Visualization

Univariate Analysis

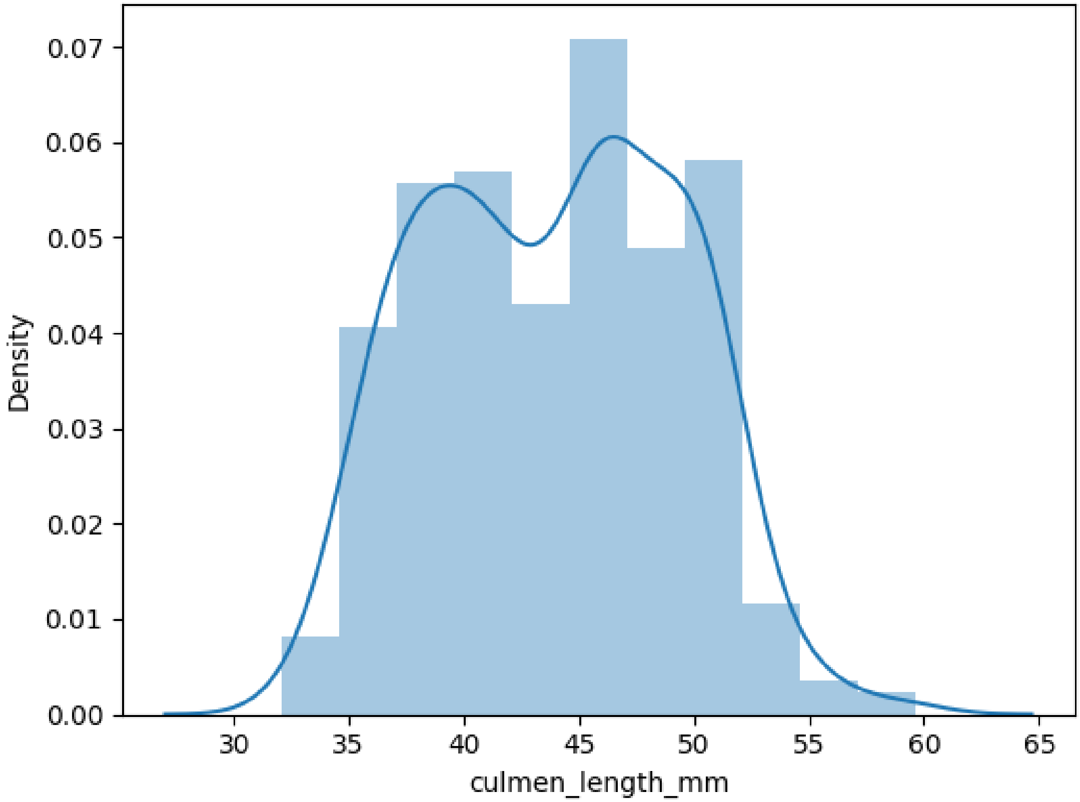
[241]:

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

```
<ipython-input-241-24e9b5890c61>: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)
```

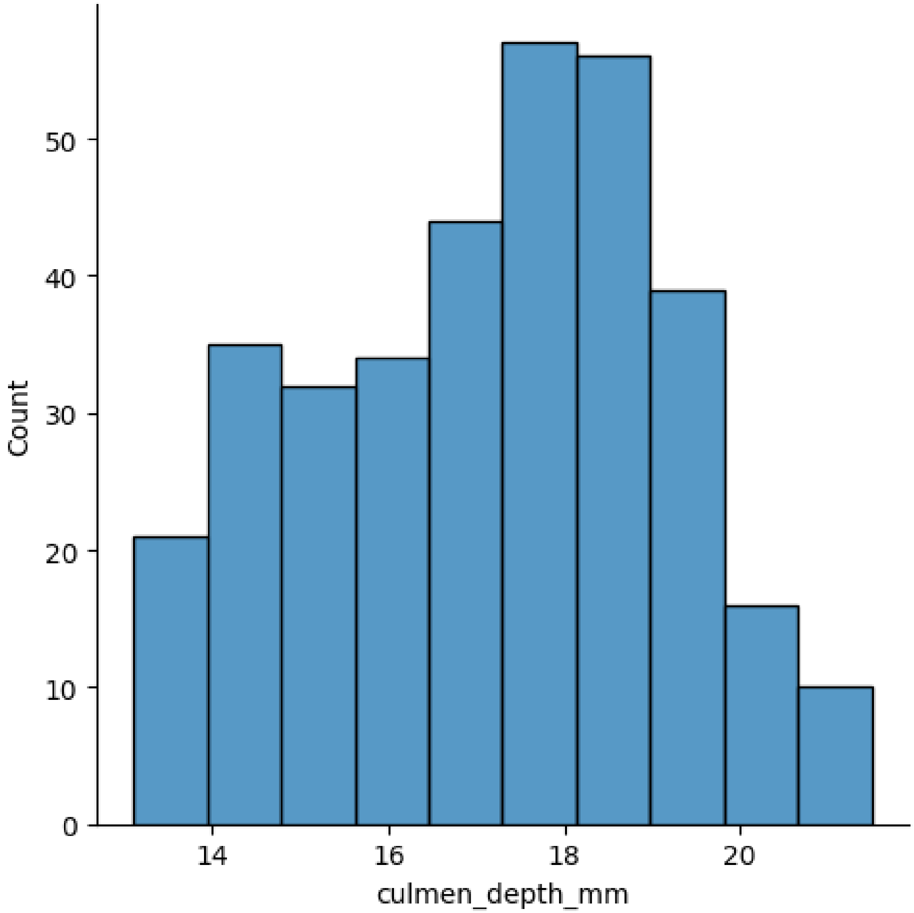
[241]:

```
<Axes: xlabel='culmen_length_mm', ylabel='Density'>
```



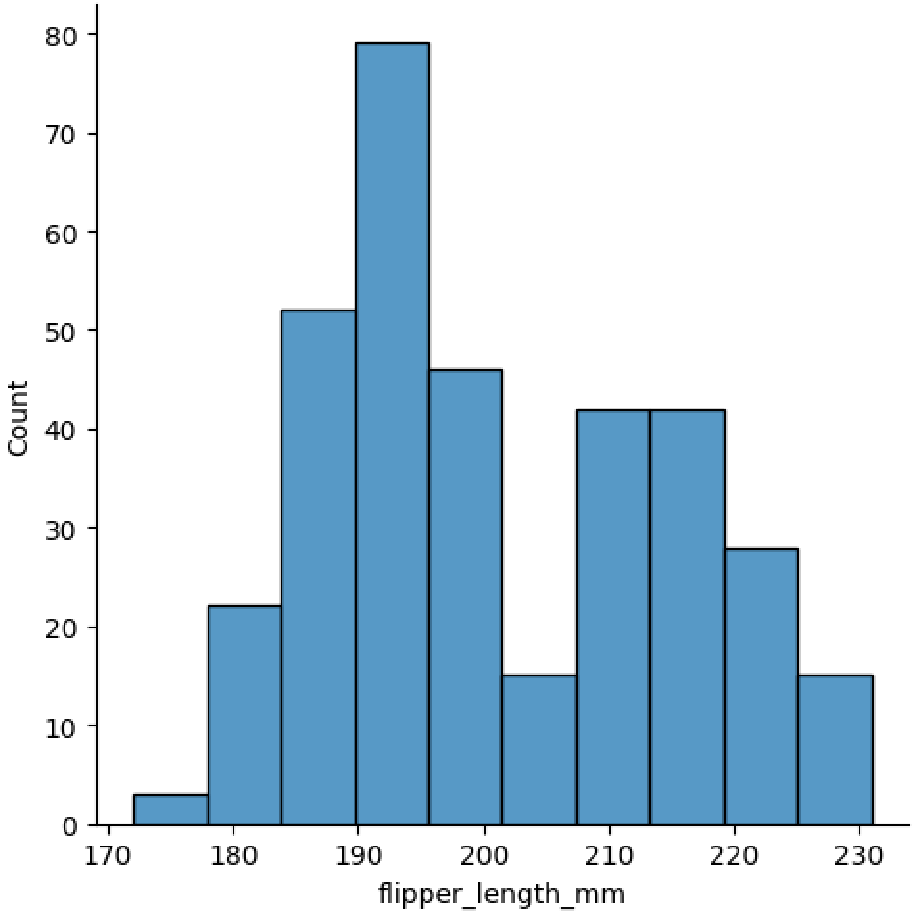
```
[242]: sns.displot(df.culmen_depth_mm)
```

```
[242]: <seaborn.axisgrid.FacetGrid at 0x7f78abc257e0>
```

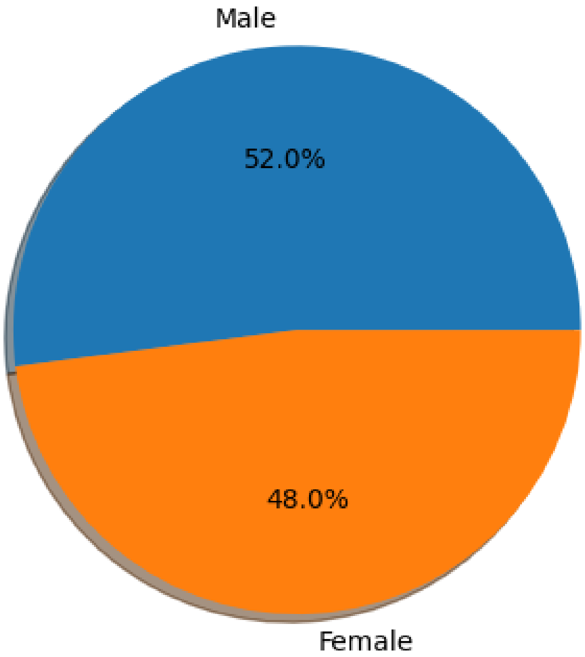


```
[243]: sns.displot(df.flipper_length_mm)
```

```
[243]: <seaborn.axisgrid.FacetGrid at 0x7f78ac10cc40>
```



[244]:
```
plt.pie(df.sex.value_counts(),[0.,0.],labels=['Male','Female'],autopct="%1.
 ↪1f%%",shadow=True)
plt.show()
```



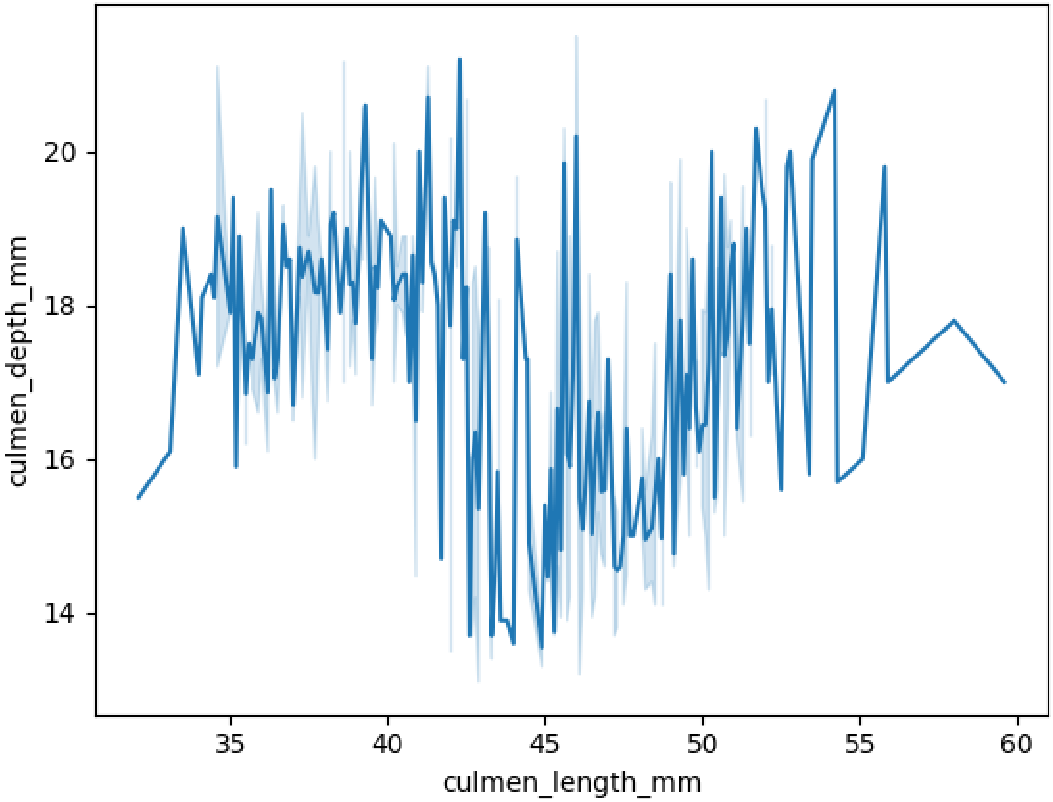
##Bivariate Analysis

[245]:

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

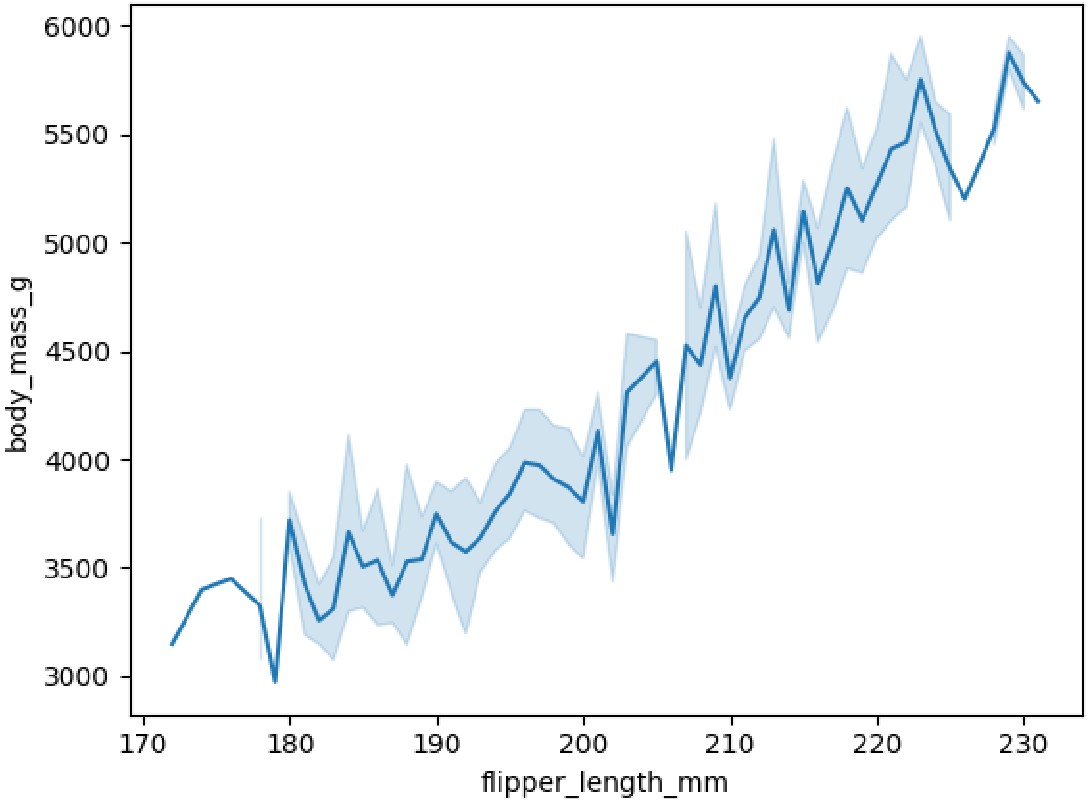
[245]:

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



```
[246]: sns.lineplot(x=df.flipper_length_mm,y=df.body_mass_g)
```

```
[246]: <Axes: xlabel='flipper_length_mm', ylabel='body_mass_g'>
```

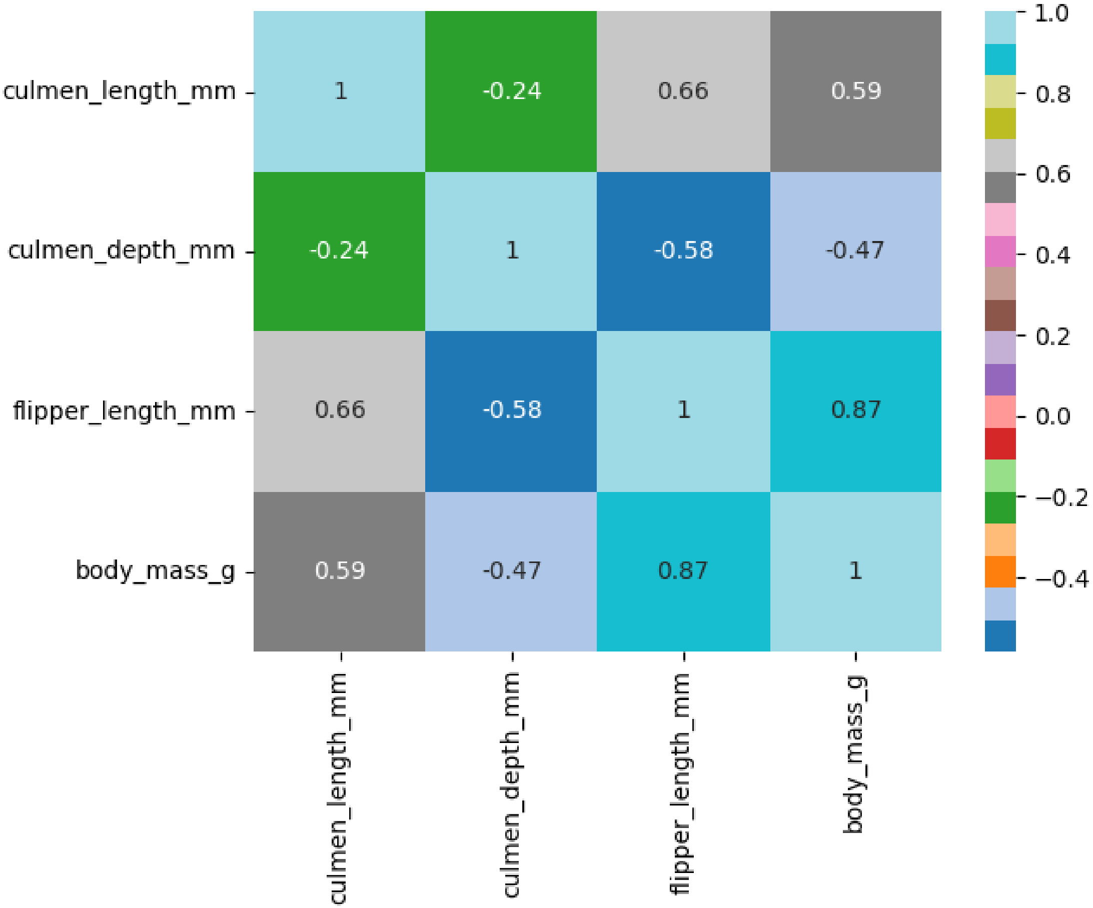


##Multivariate Analysis

[247]:
```
sns.heatmap(df.corr(),annot=True,cmap="tab20")
```

```
<ipython-input-247-dbf7c0edd73f>: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,cmap="tab20")
```

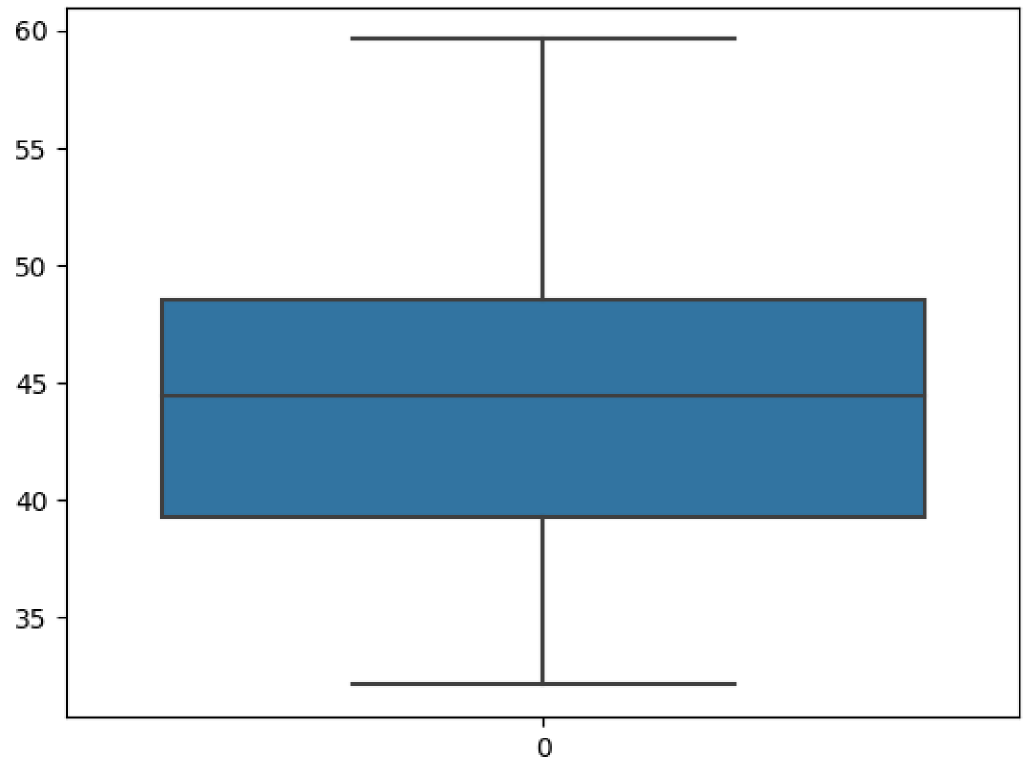
[247]:
```
<Axes: >
```



##Outlier Detection

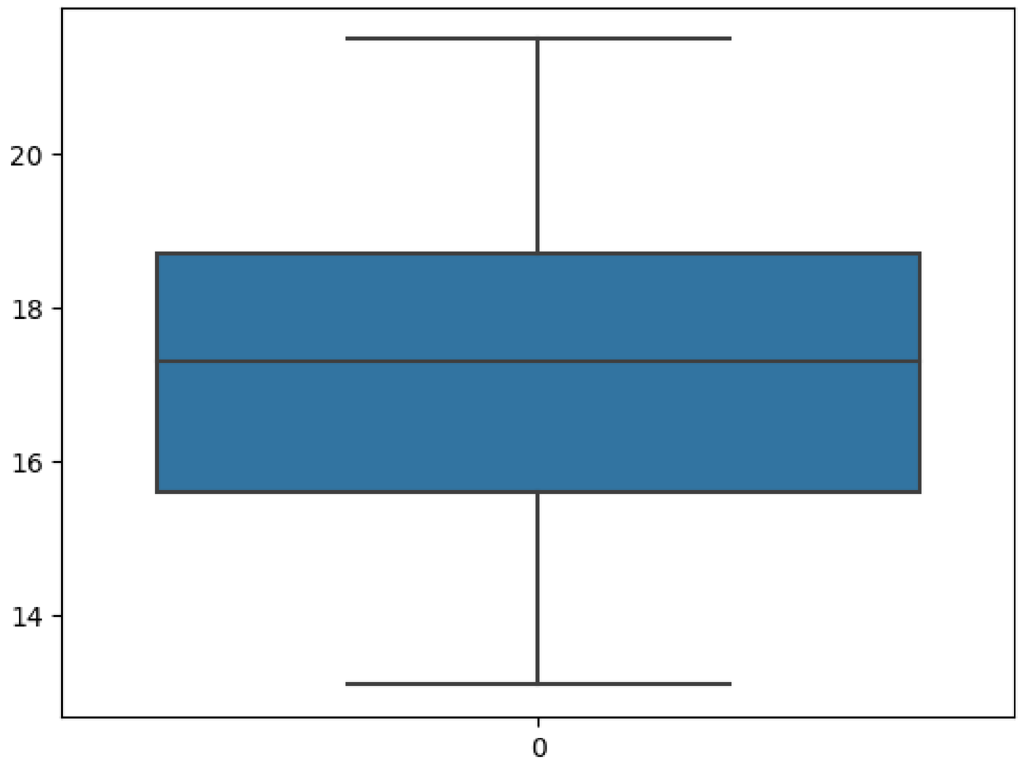
[248]:
```
sns.boxplot(df.culmen_length_mm)
```

[248]:
```
<Axes: >
```



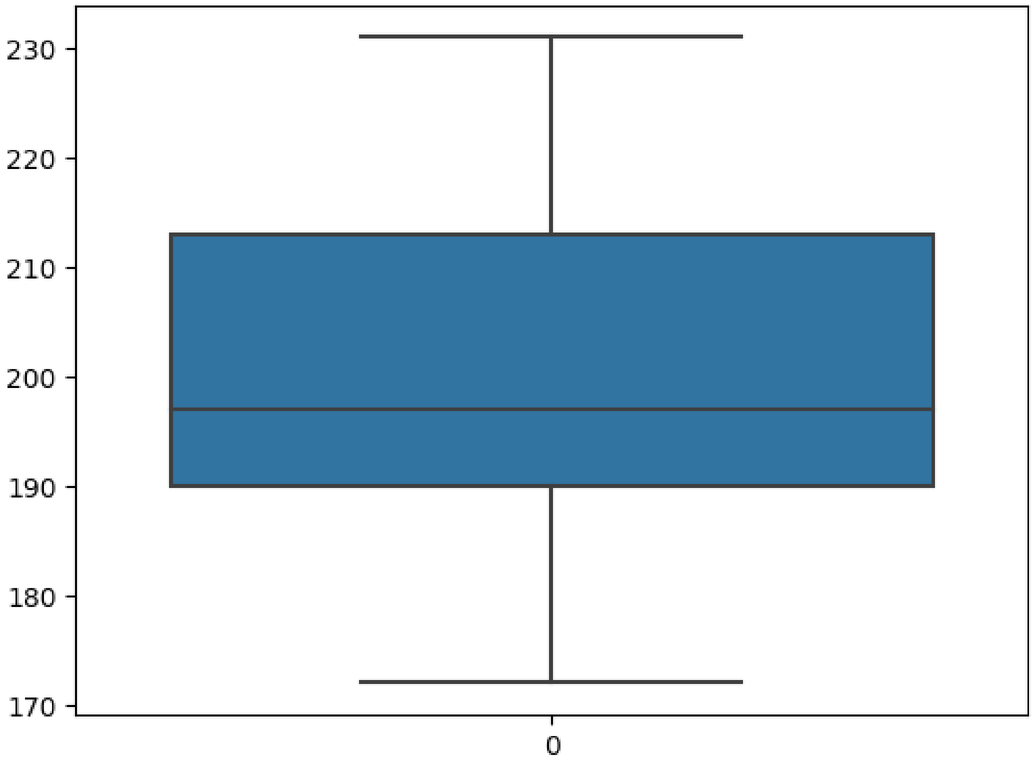
[249]:
```
sns.boxplot(df.culmen_depth_mm)
```

[249]:
```
<Axes: >
```



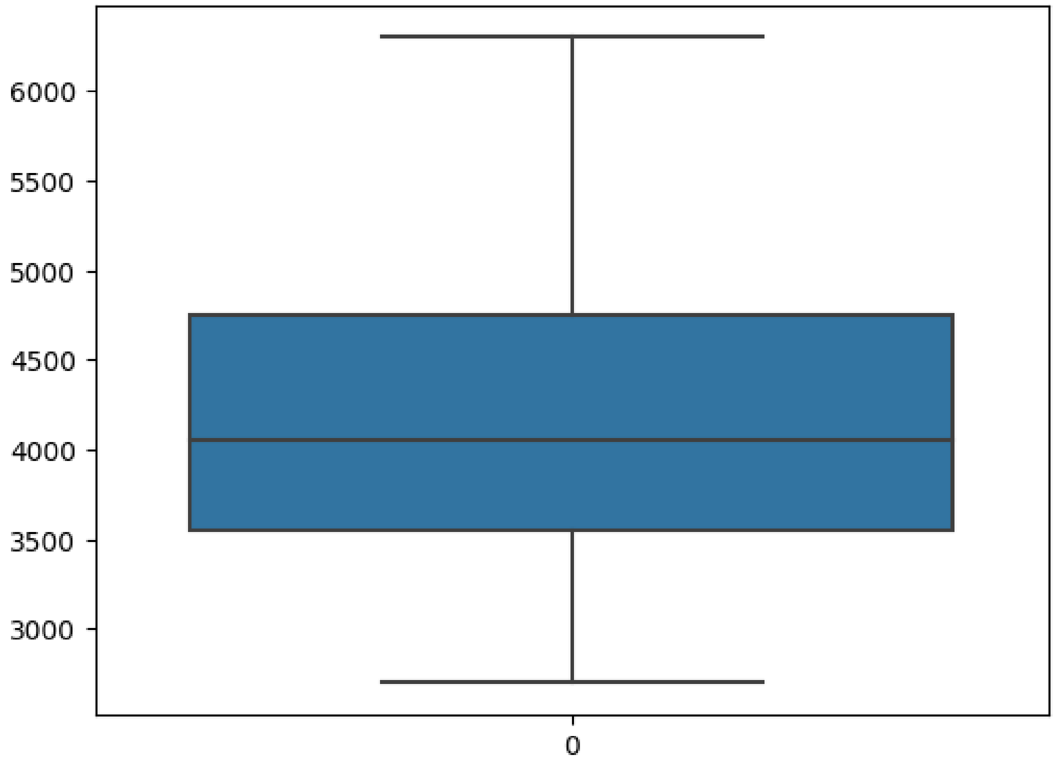
```
[250]: sns.boxplot(df.flipper_length_mm)
```

```
[250]: <Axes: >
```



```
[251]: sns.boxplot(df.body_mass_g)
```

```
[251]: <Axes: >
```



##Independent(x) and dependent(y) variable split

[252]:
```
x=df.iloc[:,1:]
x.head()
```

[252]:
```
      island  culmen_length_mm  culmen_depth_mm  flipper_length_mm  \
0  Torgersen             39.10             18.7              181.0
1  Torgersen             39.50             17.4              186.0
2  Torgersen             40.30             18.0              195.0
3  Torgersen             44.45             17.3              197.0
4  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
```

##Performing Label Encoding for categorical columns

[253]:
```
from sklearn.preprocessing import LabelEncoder
le = LabelEncoder()
```

```
[254]: x['island'] = le.fit_transform(df['island'])
       x['sex'] = le.fit_transform(df['sex'])
```

```
[255]: x.head()
```

```
[255]:    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    1
       1    0
       2    0
       3    1
       4    0
```

```
[256]: y = df.species
```

```
[257]: y.head()
```

```
[257]: 0    Adelie
       1    Adelie
       2    Adelie
       3    Adelie
       4    Adelie
       Name: species, dtype: object
```

##Scaling

```
[258]: from sklearn.preprocessing import MinMaxScaler
       scale = MinMaxScaler()
```

```
[259]: x_scaled=pd.DataFrame(scale.fit_transform(x),columns=x.columns)
       x_scaled.head()
```

```
[259]:    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  0.0
2  0.0
3  1.0
4  0.0
```

##Train Test Split

[260]:
```
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.
 ↪3,random_state=0)
```

##Train and Test data shape

[261]:
```
x_train.shape
```

[261]:
```
(240, 6)
```

[262]:
```
x_test.shape
```

[262]:
```
(104, 6)
```

[262]:
```

```