```
# Sukanth K
# 21BRS1617

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

#importing the dataset
dataset = pd.read_csv('penguins_size.csv')
dataset.head()
dataset.info()
dataset.describe()
```

```
<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
```

```
       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
```

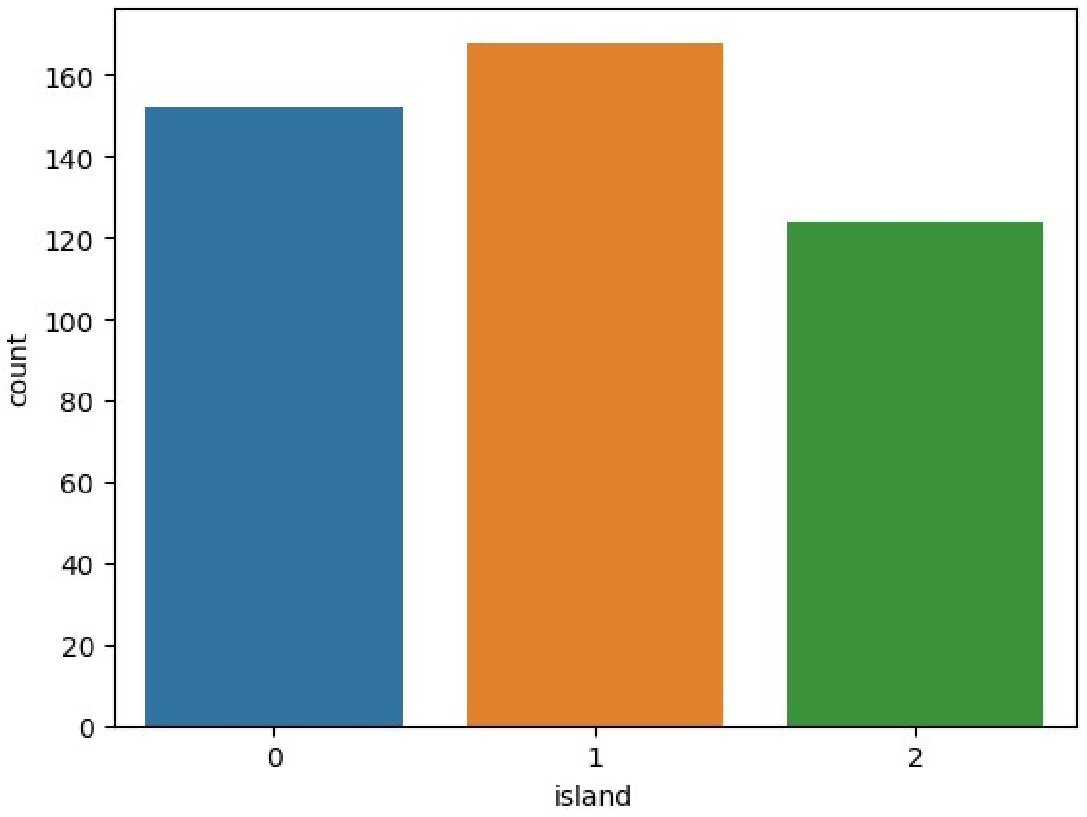
```
#check for null values
print(dataset.isnull().sum())
```

```python
#no columns have null values
#check for duplicate rows
print(dataset.duplicated().sum())

#check for duplicate columns
print(dataset.T.duplicated().sum())

#check for constant columns
print(dataset.columns[dataset.nunique()==1])

#check for constant rows
print(dataset[dataset.nunique(axis=1)==1])

#convert all categorical data to numerical data
dataset['species'] =
dataset['species'].map({'Adelie':0,'Chinstrap':1,'Gentoo':2})

#convert island to numerical data
dataset['island'] =
dataset['island'].map({'Torgersen':0,'Biscoe':1,'Dream':2})

#convert sex to numerical data
dataset['sex'] = dataset['sex'].map({'MALE':0 , 'FEMALE':1})
```

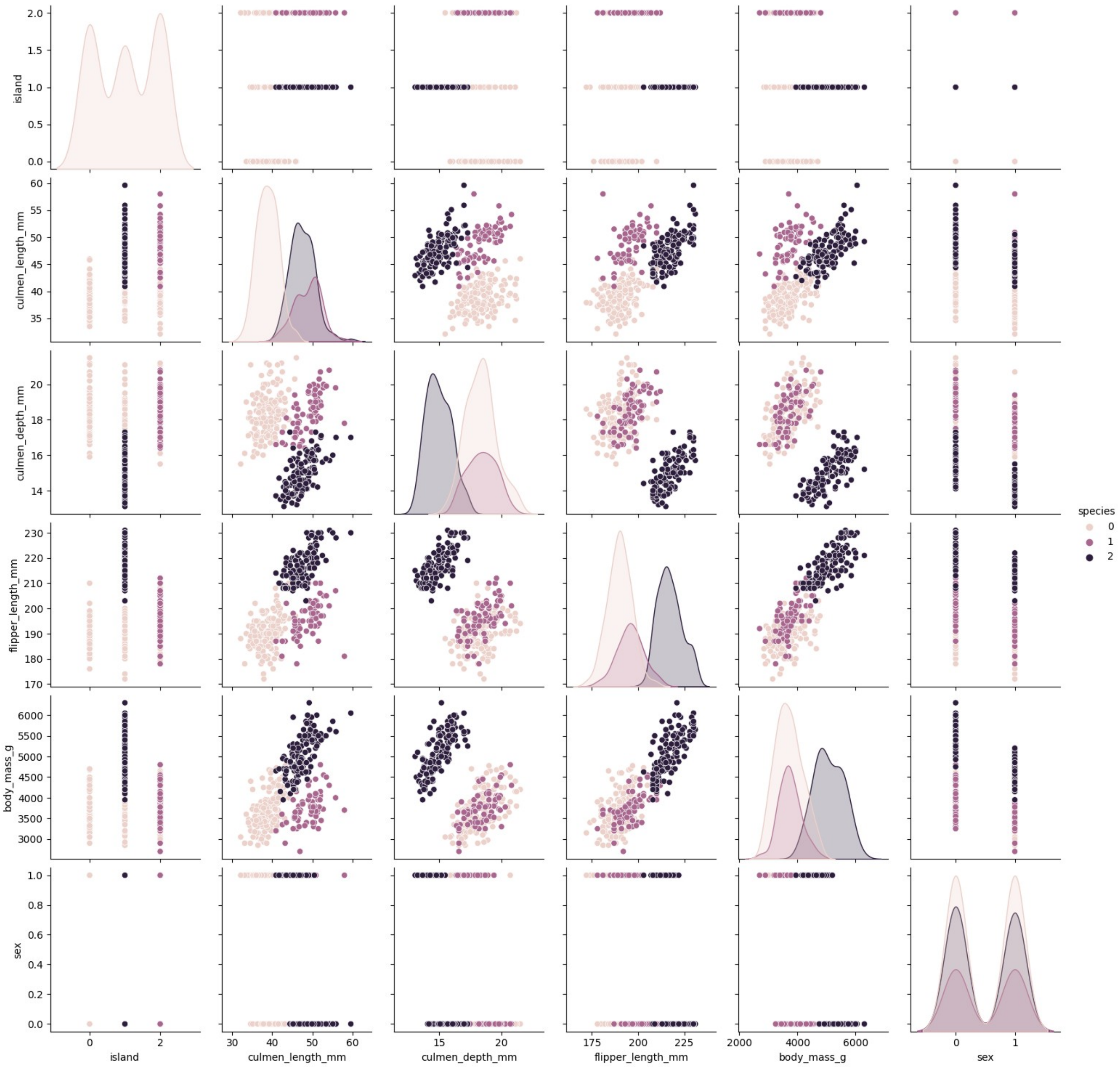
```
species               0
island                0
culmen_length_mm      2
culmen_depth_mm       2
flipper_length_mm     2
body_mass_g           2
sex                  10
dtype: int64
0
0
Index([], dtype='object')
Empty DataFrame
Columns: [species, island, culmen_length_mm, culmen_depth_mm,
flipper_length_mm, body_mass_g, sex]
Index: []
```

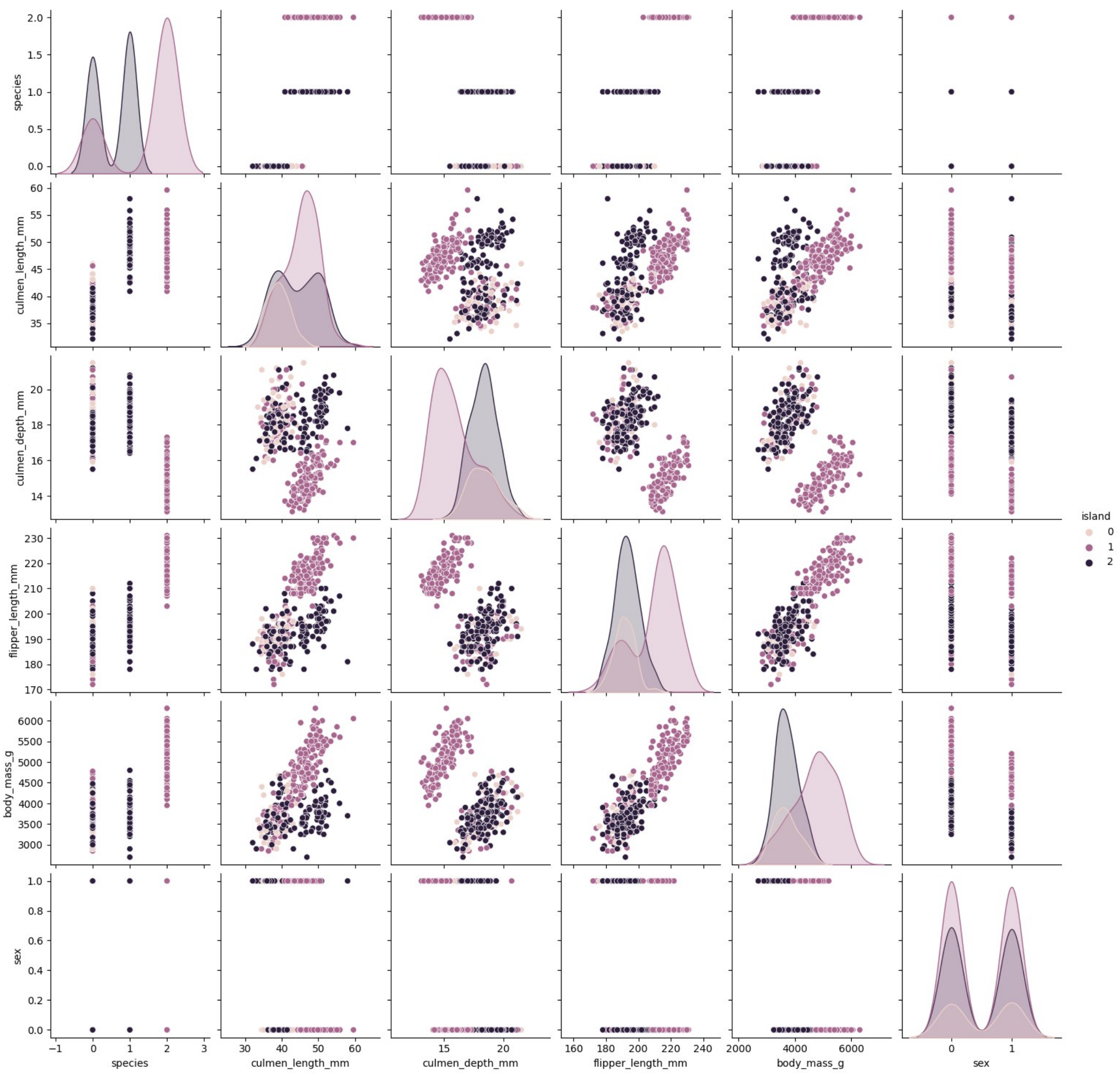
```python
#perform uni-variate analysis
sns.countplot(x='species',data=dataset)
sns.countplot(x='island',data=dataset)
```

```
<Axes: xlabel='island', ylabel='count'>
```

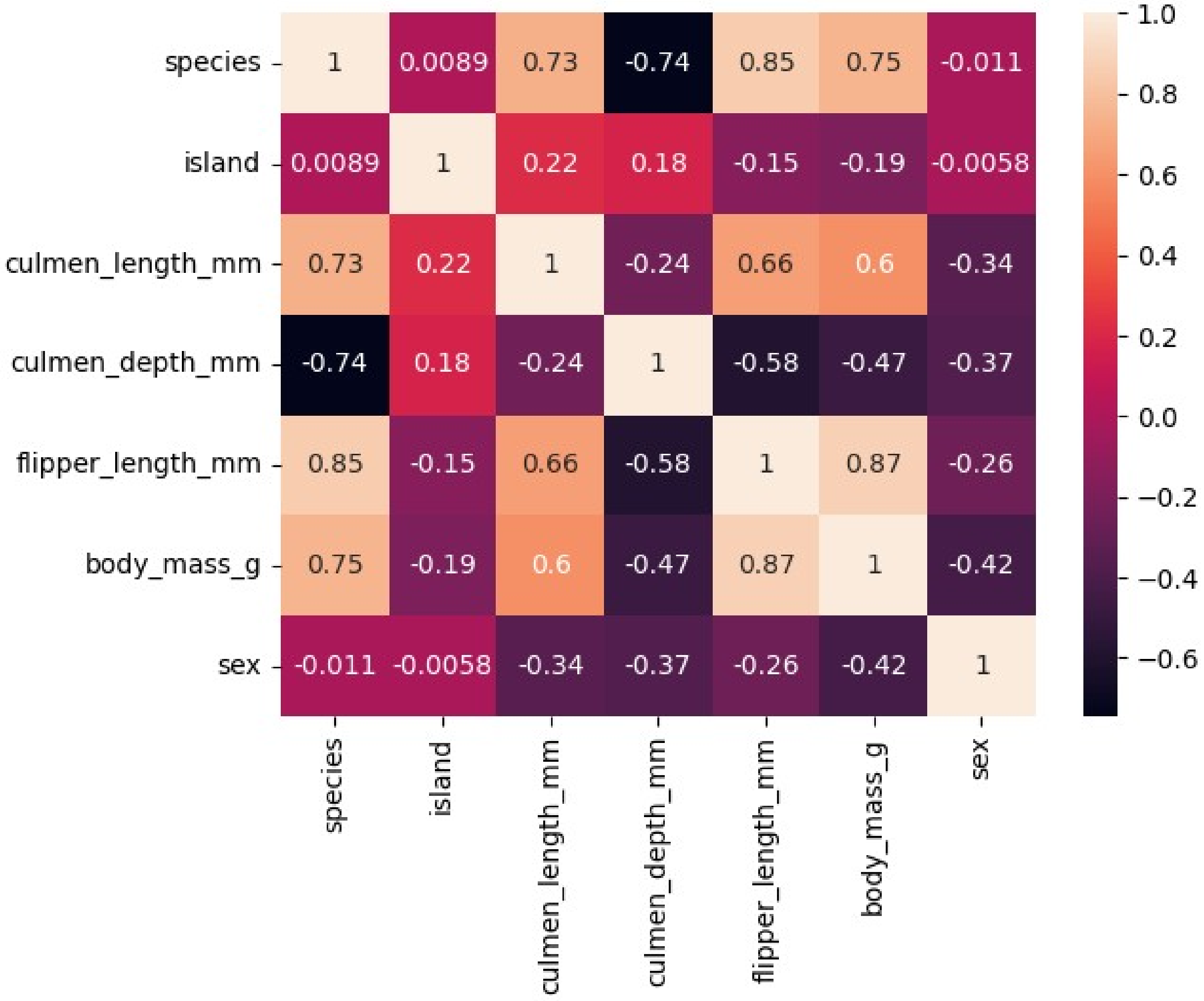


```
#perform bi-variate analysis
sns.pairplot(dataset,hue='species')
sns.pairplot(dataset,hue='island')

<seaborn.axisgrid.PairGrid at 0x7ae5b5751540>
```

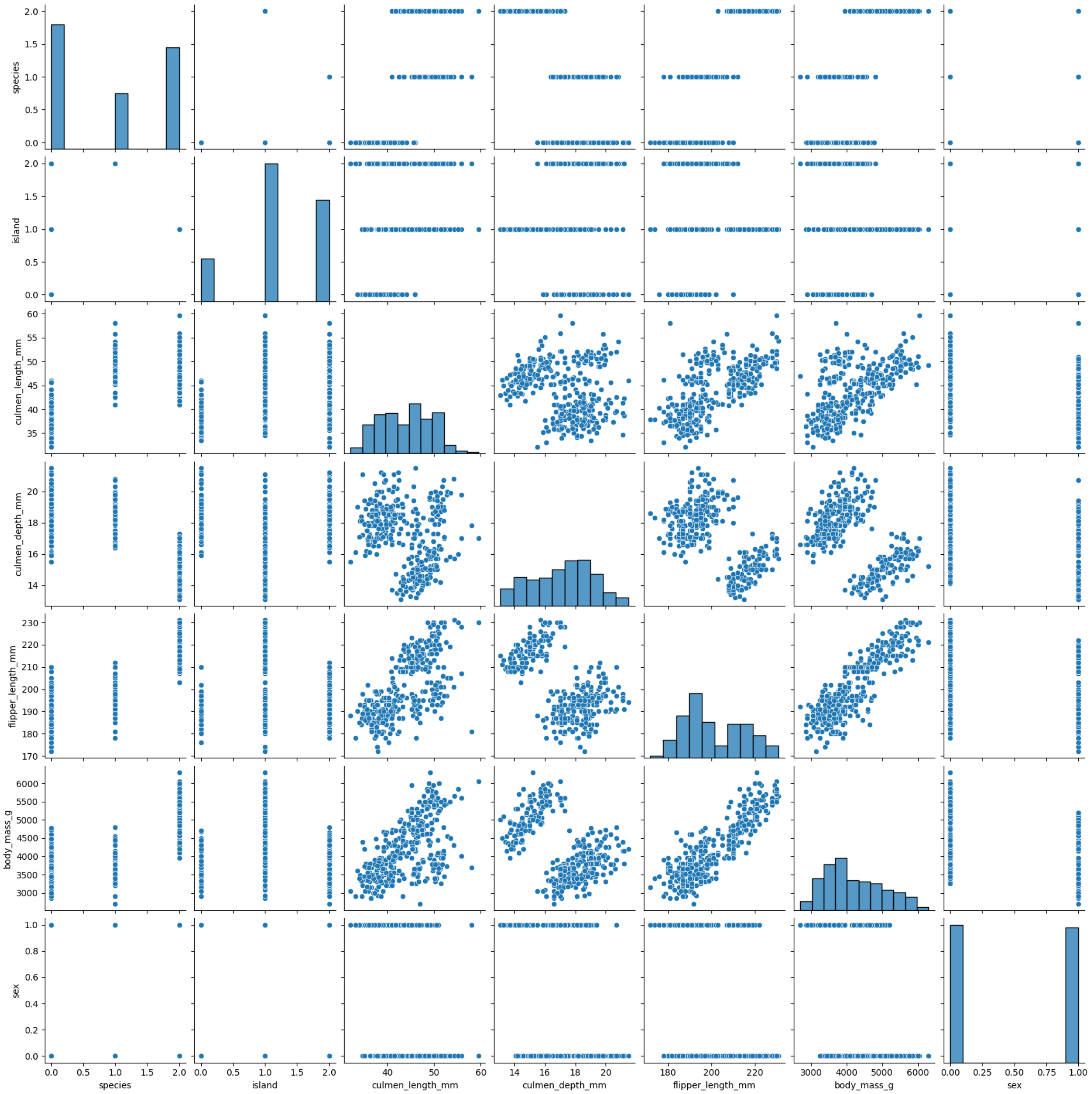

island
culmen_length_mm
culmen_depth_mm
flipper_length_mm
body_mass_g
sex
species
0
1
2
0.0
0.5
1.0
1.5
2.0
35
40
45
50
55
60
14
16
18
20
170
180
190
200
210
220
230
3000
3500
4000
4500
5000
5500
6000
0.0
0.2
0.4
0.6
0.8
1.0
0
1
2
30
40
50
60
15
20
175
200
225
2000
4000
6000
0
1



```
#perform multi-variate analysis
sns.heatmap(dataset.corr(),annot=True)
#perform another multi-variate analysis without heatmap
sns.pairplot(dataset)

<seaborn.axisgrid.PairGrid at 0x7ae5ae8d01c0>
```



	species	island	culmen_length_mm	culmen_depth_mm	flipper_length_mm	body_mass_g	sex
species	1	0.0089	0.73	-0.74	0.85	0.75	-0.011
island	0.0089	1	0.22	0.18	-0.15	-0.19	-0.0058
culmen_length_mm	0.73	0.22	1	-0.24	0.66	0.6	-0.34
culmen_depth_mm	-0.74	0.18	-0.24	1	-0.58	-0.47	-0.37
flipper_length_mm	0.85	-0.15	0.66	-0.58	1	0.87	-0.26
body_mass_g	0.75	-0.19	0.6	-0.47	0.87	1	-0.42
sex	-0.011	-0.0058	-0.34	-0.37	-0.26	-0.42	1



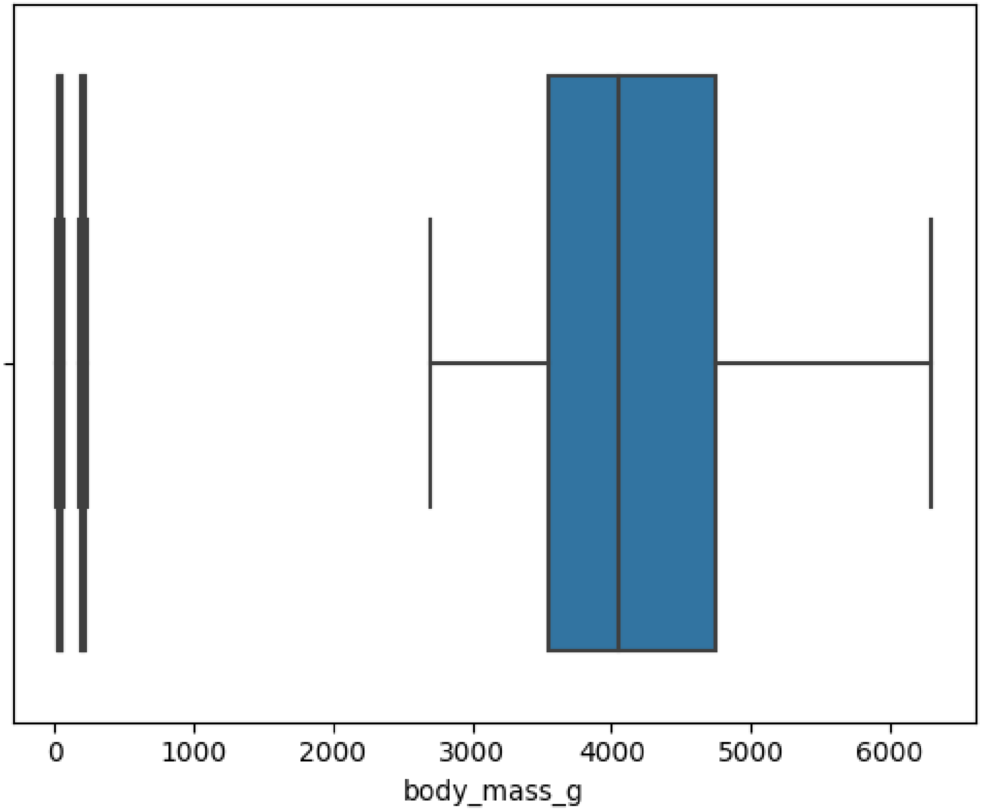
```
#dataset has null values in 5 columns
#columns culmen length (mm) and culmen depth (mm) have 2 null values
each
#we can replace them with mean values
dataset['culmen_length_mm'].fillna(dataset['culmen_length_mm'].mean(),
inplace=True)

#dataset['culmen_length_mm'].isnull().sum()
#flipper_length_mm and body_mass_g have 2 null values each
#we can replace them with mean values
dataset['flipper_length_mm'].fillna(dataset['flipper_length_mm'].mean(
),inplace=True)
```

```
#sex is categorical data and has 10 null values
#we can replace them with mode values
#describe the dataset
dataset.describe()
```

```
          species      island  culmen_length_mm  culmen_depth_mm  \
count  344.000000  344.000000        344.000000       342.000000   
mean     0.918605    1.209302         43.921930        17.151170   
std      0.893320    0.684970          5.443643         1.974793   
min      0.000000    0.000000         32.100000        13.100000   
25%      0.000000    1.000000         39.275000        15.600000   
50%      1.000000    1.000000         44.250000        17.300000   
75%      2.000000    2.000000         48.500000        18.700000   
max      2.000000    2.000000         59.600000        21.500000   

       flipper_length_mm  body_mass_g         sex  
count         344.000000   342.000000  333.000000  
mean          200.915205  4201.754386    0.495495  
std            14.020657   801.954536    0.500732  
min           172.000000  2700.000000    0.000000  
25%           190.000000  3550.000000    0.000000  
50%           197.000000  4050.000000    0.000000  
75%           213.000000  4750.000000    1.000000  
max           231.000000  6300.000000    1.000000  
```

```
#find outliers in the dataset
#boxplot for culmen_length_mm
sns.boxplot(x=dataset['culmen_length_mm'])

#boxplot for culmen_depth_mm
sns.boxplot(x=dataset['culmen_depth_mm'])

#boxplot for flipper_length_mm
sns.boxplot(x=dataset['flipper_length_mm'])

#boxplot for body_mass_g
sns.boxplot(x=dataset['body_mass_g'])
```

```
<Axes: xlabel='body_mass_g'>
```



```
#replace outliers with mean values
#culmen_length_mm
dataset['culmen_length_mm'] =
np.where(dataset['culmen_length_mm']>50,dataset['culmen_length_mm'].me
an(),dataset['culmen_length_mm'])

#culmen_depth_mm
dataset['culmen_depth_mm'] =
np.where(dataset['culmen_depth_mm']>25,dataset['culmen_depth_mm'].mean
(),dataset['culmen_depth_mm'])

#flipper_length_mm
dataset['flipper_length_mm'] =
np.where(dataset['flipper_length_mm']>230,dataset['flipper_length_mm']
.mean(),dataset['flipper_length_mm'])

#body_mass_g
dataset['body_mass_g'] =
np.where(dataset['body_mass_g']>6000,dataset['body_mass_g'].mean(),dat
aset['body_mass_g'])

#check correlation of independent variables with target variable

#here independent variable is species
dataset.corr()['species'].sort_values(ascending=False)
```

```
#check correlation of independent variables with each other
dataset.corr()
```

```
                    species    island  culmen_length_mm
culmen_depth_mm  \
species            1.000000  0.008864          0.772193        -
0.744076
island             0.008864  1.000000          0.135793
0.179753
culmen_length_mm   0.772193  0.135793          1.000000        -
0.398866
culmen_depth_mm   -0.744076  0.179753         -0.398866
1.000000
flipper_length_mm  0.849323 -0.143807          0.658667        -
0.583180
body_mass_g        0.746507 -0.187970          0.612879        -
0.472364
sex               -0.010964 -0.005834         -0.218821        -
0.372673

                   flipper_length_mm  body_mass_g       sex
species                     0.849323     0.746507 -0.010964
island                     -0.143807    -0.187970 -0.005834
culmen_length_mm            0.658667     0.612879 -0.218821
culmen_depth_mm            -0.583180    -0.472364 -0.372673
flipper_length_mm           1.000000     0.856008 -0.250515
body_mass_g                 0.856008     1.000000 -0.418046
sex                        -0.250515    -0.418046  1.000000
```

```
#already performed label encoding converting categorical data to
numerical data
dataset.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    int64
 1   island             344 non-null    int64
 2   culmen_length_mm   344 non-null    float64
 3   culmen_depth_mm    342 non-null    float64
 4   flipper_length_mm  344 non-null    float64
 5   body_mass_g        342 non-null    float64
 6   sex                333 non-null    float64
dtypes: float64(5), int64(2)
memory usage: 18.9 KB
```

```
#splitting the dataset into dependent and independent variables
X = dataset.iloc[:,1:7].values
y = dataset.iloc[:,0].values

#scale the data
from sklearn.preprocessing import StandardScaler
sc = StandardScaler()
X = sc.fit_transform(X)
Y = sc.fit_transform(y.reshape(-1,1))

#splitting the dataset into training and testing set
from sklearn.model_selection import train_test_split
X_train,X_test,y_train,y_test = train_test_split(X,y,test_size=0.25)

#check training and testing set shape
print(X_train.shape)
print(X_test.shape)
print(y_train.shape)
print(y_test.shape)

#visualise the training and testing set
print(X_train)
print(X_test)
print(y_train)
print(y_test)
```

```
(258, 6)
(86, 6)
(258,)
(86,)
[[ 1.15603468 -0.76023385  0.32903754 -0.92248098 -1.13216767
1.00904996]
 [ 1.15603468 -1.44534687  0.633312   -1.21013269 -0.90960418
1.00904996]
 [-0.30600918  1.69672317 -0.53307343  0.87534221  1.5385942  -
0.99103121]
 ...
 [-1.76805304 -0.76023385  0.1261879  -1.06630683 -0.49627199
1.00904996]
 [-1.76805304 -1.27997476  1.69827261 -0.13143877 -0.52806677 -
0.99103121]
 [-1.76805304 -0.38224046  0.73473682 -0.85056805 -1.10037289 -
0.99103121]]
[[ 1.15603468 -0.45311423  0.88687405  0.51577758  0.13962369 -
0.99103121]
 [-0.30600918 -0.12237001  1.19114851 -0.27526463  0.10782891 -
0.99103121]
 [ 1.15603468 -0.83110761  0.73473682 -0.77865512  0.07603413 -
0.99103121]
 [-0.30600918  1.1061085  -1.29375958  0.80342929  0.68013503
1.00904996]
 [ 1.15603468  1.60222483  0.93758646 -0.05952585 -0.49627199 -
0.99103121]
 [-0.30600918 -0.71298468  0.27832513 -0.56291634 -1.25934681
1.00904996]
 [-1.76805304  0.0430021   0.22761272 -0.34717756  0.64834024 -
0.99103121]
 [ 1.15603468 -1.51622063  1.19114851 -0.77865512 -0.49627199 -
0.99103121]
 [-1.76805304 -1.61071898 -0.27951138 -0.77865512 -1.45011551
1.00904996]
 [ 1.15603468  0.7517397  -0.02594933 -0.77865512 -0.78242505
1.00904996]
 [-0.30600918  1.08248392 -0.93877271  1.52255856  1.34782549 -
0.99103121]
 [ 1.15603468 -0.47673881 -0.07666174 -0.77865512 -0.59165634 -
0.99103121]
 [-0.30600918 -1.82334026  0.37974995 -0.63482927 -0.59165634
1.00904996]
 [-1.76805304 -1.91783861  2.00254707 -0.2033517   0.26680283 -
0.99103121]
 [-0.30600918  1.57860024 -0.68521066  1.091081    0.9344933  -
0.99103121]
 [-0.30600918 -1.09097807 -0.07666174 -1.42587147 -1.29114159
1.00904996]
 [-0.30600918  0.82261346 -1.14162235  1.45064564  1.41141506 -
0.99103121]
 [ 1.15603468 -0.90198137  0.83616164 -0.77865512 -0.75063026
1.00904996]
 [-0.30600918  0.01937751 -1.49660922  0.5876905   0.64834024
1.00904996]
 [-0.30600918  1.72034776 -0.43164861  2.09786198  1.92013161 -
0.99103121]
 [ 1.15603468  0.58636759  0.32903754 -0.2033517  -0.30550328
1.00904996]
 [-1.76805304  0.09025127  1.03901128 -0.27526463 -0.8778094  -
0.99103121]
 [ 1.15603468  0.68086594  1.1404361  -0.49100341 -0.84601462
1.00904996]
 [-0.30600918  0.2844288  -1.09090994  1.59447149  1.7293629  -
0.99103121]
 [-0.30600918  0.65724135 -1.09090994  1.37873271  1.02987765 -
0.99103121]
 [-0.30600918  0.60999218 -1.75017127  0.65960343  0.13962369
1.00904996]
 [-0.30600918 -1.65796815 -0.12737415 -1.13821976 -1.32293638
1.00904996]
 [ 1.15603468 -0.33499129  1.59684779 -0.49100341 -0.81421983 -
0.99103121]
```

```
 [ 1.15603468  1.53135107  0.53188718 -0.41909048  0.26680283 -
0.99103121]
 [-0.30600918  0.2844288  -0.73592307  0.0062892   1.85654204 -
0.99103121]
 [-0.30600918  1.41322813 -1.54732163  0.65960343  0.3303924
1.00904996]
 [-0.30600918  0.56274301 -1.3951844   0.65960343  0.26680283
1.00904996]
 [ 1.15603468 -1.23272559  0.88687405 -1.56969733 -1.54549987
nan]
 [-0.30600918  0.3973709   0.07547549  1.30681978  1.34782549 -
0.99103121]
 [-0.30600918 -0.73660927  0.27832513 -1.06630683 -0.8778094
1.00904996]
 [ 1.15603468 -1.35084852 -0.12737415 -1.13821976 -1.51370508
1.00904996]
 [ 1.15603468  0.11387586 -0.27951138 -0.9943939  -1.64088422
1.00904996]
 [-1.76805304 -0.99647972  0.37974995 -0.77865512 -1.10037289
1.00904996]
 [-0.30600918  0.2844288  -0.63449825  1.73829735  1.5385942  -
0.99103121]
 [-0.30600918  1.41322813 -1.04019753  1.52255856  1.47500463 -
0.99103121]
 [ 1.15603468 -0.12237001  0.68402441 -1.4977844  -0.81421983
1.00904996]
 [ 1.15603468 -0.28774212  0.68402441  0.01238708 -0.24191372 -
0.99103121]
 [-0.30600918  0.89348722 -1.3951844   1.16299392  0.90269851
1.00904996]
 [-0.30600918  0.16112503 -1.3951844   1.23490685  0.5211611
1.00904996]
 [-0.30600918  1.12973309 -1.59803404  0.80342929  0.87090373
1.00904996]
 [-0.30600918  0.56274301 -1.34447199  1.01916807  1.02987765
1.00904996]
 [-0.30600918  0.0430021  -2.05444573  1.01916807  1.02987765
1.00904996]
 [-0.30600918  0.79898887 -1.04019753  1.01916807  1.15705679 -
0.99103121]
 [-0.30600918  0.18474962 -0.98948512  0.87534221  0.58475067
1.00904996]
 [-0.30600918  0.2083742  -1.64874645  1.16299392  0.90269851
1.00904996]
 [-0.30600918  0.58636759 -0.3809362   1.59447149  2.23807945 -
0.99103121]
 [ 1.15603468  0.2844288   0.98829887 -0.34717756 -0.81421983 -
0.99103121]
 [ 1.15603468 -1.65796815  0.43046236  0.08430001 -0.81421983
1.00904996]
 [-0.30600918 -0.92560596  0.02476308 -1.4977844  -0.49627199 -
0.99103121]
 [ 1.15603468 -1.58709439 -0.02594933 -0.9943939  -0.62345113
1.00904996]
 [ 1.15603468 -0.42948964  0.88687405 -1.21013269 -0.36909285 -
0.99103121]
 [ 1.15603468  0.91711181  0.32903754 -0.56291634 -0.49627199
1.00904996]
 [ 1.15603468 -1.53984522  0.07547549 -0.9943939  -1.13216767
1.00904996]
 [-0.30600918 -1.209101    0.98829887 -0.49100341 -0.55986156 -
0.99103121]
 [-0.30600918  1.34235437 -0.43164861  1.37873271  1.5385942  -
0.99103121]
 [-1.76805304 -0.5003634   0.93758646 -0.13143877 -0.24191372 -
0.99103121]
 [-0.30600918  0.84623805 -0.68521066  1.01916807  1.09346722 -
0.99103121]
 [ 1.15603468  0.77536429  0.88687405 -0.41909048 -0.05114501
1.00904996]
 [ 1.15603468 -1.30359935  0.48117477 -1.64161025 -0.36909285 -
0.99103121]
 [-0.30600918  0.2844288  -0.43164861  2.09786198  1.66577333 -
0.99103121]
 [ 1.15603468  0.2844288   0.83616164  0.15621294 -0.11473458 -
0.99103121]
 [ 1.15603468 -0.57123716  0.68402441 -0.34717756  0.20321326 -
0.99103121]
 [-0.30600918  1.55497565 -0.73592307  1.16299392  2.11090031 -
0.99103121]
 [-0.30600918 -0.16961918 -1.85159609  0.65960343 -0.05114501
1.00904996]
 [ 1.15603468  1.48410189  1.19114851  0.65960343 -0.30550328 -
0.99103121]
 [ 1.15603468 -2.05958613 -0.02594933 -1.13821976 -1.00498854
1.00904996]
 [-1.76805304 -1.70521732  0.17690031 -0.77865512 -0.62345113
1.00904996]
 [-0.30600918  0.2844288  -0.78663548  1.45064564  1.60218377 -
0.99103121]
 [ 1.15603468 -0.05149625  0.07547549 -0.9943939  -1.0685781
1.00904996]
 [-0.30600918  0.2844288  -0.63449825  1.52255856  1.7293629  -
0.99103121]
 [ 1.15603468 -0.38224046  0.17690031 -0.77865512 -0.36909285 -
0.99103121]
 [ 1.15603468 -1.209101    1.08972369 -1.42587147 -1.13216767
1.00904996]
```

```
 [-0.30600918  0.42099549 -0.73592307  1.16299392  0.87090373
nan]
 [ 1.15603468 -0.73660927  0.48117477 -1.06630683  0.3303924  -
0.99103121]
 [-0.30600918 -1.09097807 -0.33022379 -0.2033517  -0.46447721
1.00904996]
 [ 1.15603468  0.86986264  0.73473682 -0.77865512 -0.94139897
1.00904996]
 [ 1.15603468 -1.3980977   0.68402441 -0.56291634 -0.8778094
1.00904996]
 [-0.30600918  0.11387586  0.93758646 -0.27526463  0.74372459 -
0.99103121]
 [-0.30600918  0.2844288  -0.02594933  1.95403613  1.5385942  -
0.99103121]
 [-0.30600918 -1.18547642  0.78544923 -1.4977844  -0.75063026 -
0.99103121]
 [-0.30600918 -1.82334026  0.37974995 -0.77865512 -0.94139897
1.00904996]]
[0 0 2 0 2 0 0 0 1 0 2 0 0 0 1 1 1 0 2 1 2 2 0 0 1 2 0 1 1 0 1 2 2 2 0
1 2
 2 2 0 0 2 0 0 0 2 1 0 2 0 0 2 2 1 0 2 2 0 0 0 0 2 0 0 0 0 0 0 2 2 1 0
0 2
 0 0 1 0 2 1 0 1 1 0 0 2 1 1 1 2 2 0 2 0 2 2 2 1 2 0 2 1 0 1 0 1 1 0 2
2 0
 2 0 2 2 1 1 1 1 0 0 2 2 0 2 1 0 2 0 2 2 2 0 0 0 2 0 2 2 0 0 2 2 0 0 0
0 2
 1 0 0 2 2 0 0 1 1 2 2 0 1 0 0 0 1 1 0 0 2 0 0 0 0 2 0 2 0 2 2 0 2 2 0
0 2
 1 2 1 2 0 2 2 2 0 2 2 0 1 2 1 0 1 2 1 1 2 0 0 2 0 0 0 2 0 1 0 0 0 0 0
1 2
 0 1 2 2 1 1 0 1 1 0 2 0 0 1 2 1 2 2 2 0 2 2 2 0 2 2 2 1 0 1 2 1 0 0 0
0]
[0 0 0 2 1 0 0 0 0 1 2 0 0 0 2 0 2 0 2 2 1 0 1 2 2 2 0 0 1 2 2 2 0 2 0
0 1
 0 2 2 0 0 2 2 2 2 2 2 2 2 2 1 0 0 0 0 1 0 0 2 0 2 1 0 2 1 0 2 2 1 0 0
2 1
 2 0 0 2 0 0 1 0 0 2 0 0]
```