Name: Paripalli Shobith

Registration Number: 21BAI1722

Campus: Chennai

G-mail: shobith.paripalli2021@vitstudent.ac.in

```
In [ ]: import numpy as np
        import seaborn as sns
        import pandas as pd
        import matplotlib.pyplot as plt
```

2. Load the dataset into the tool.

```
In [ ]: df = pd.read_csv(r"C:\Users\shobi\Downloads\penguins_size.csv")
        df.head()
```

Out[ ]:

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

```
In [ ]: df.dtypes
```

```
Out[ ]: species               object
        island                object
        culmen_length_mm     float64
        culmen_depth_mm      float64
        flipper_length_mm    float64
        body_mass_g          float64
        sex                   object
        dtype: object
```

```
In [ ]: for i in df.columns:
            if df[i].dtype=="object":
                print(df[i].value_counts())
```

```
Adelie       152
Gentoo       124
Chinstrap     68
Name: species, dtype: int64
Biscoe       168
Dream        124
Torgersen     52
Name: island, dtype: int64
MALE      168
FEFEMALE    165
.           1
Name: sex, dtype: int64
```

In [ ]:

```
df.sex.replace(".",df.sex.mode()[0],inplace=True)
```

3. Perform Below Visualizations.

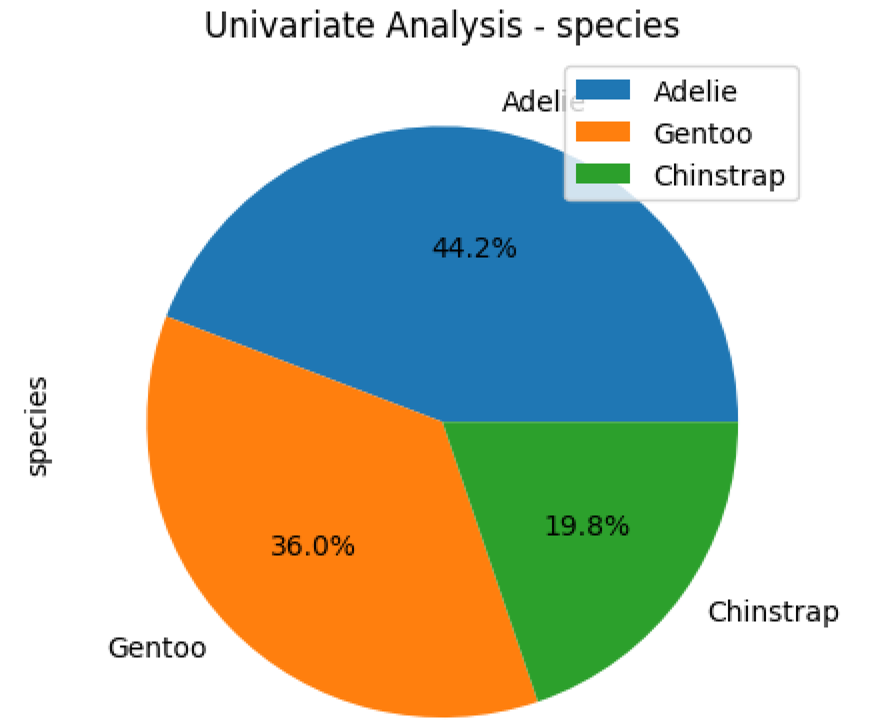
- Univariate Analysis

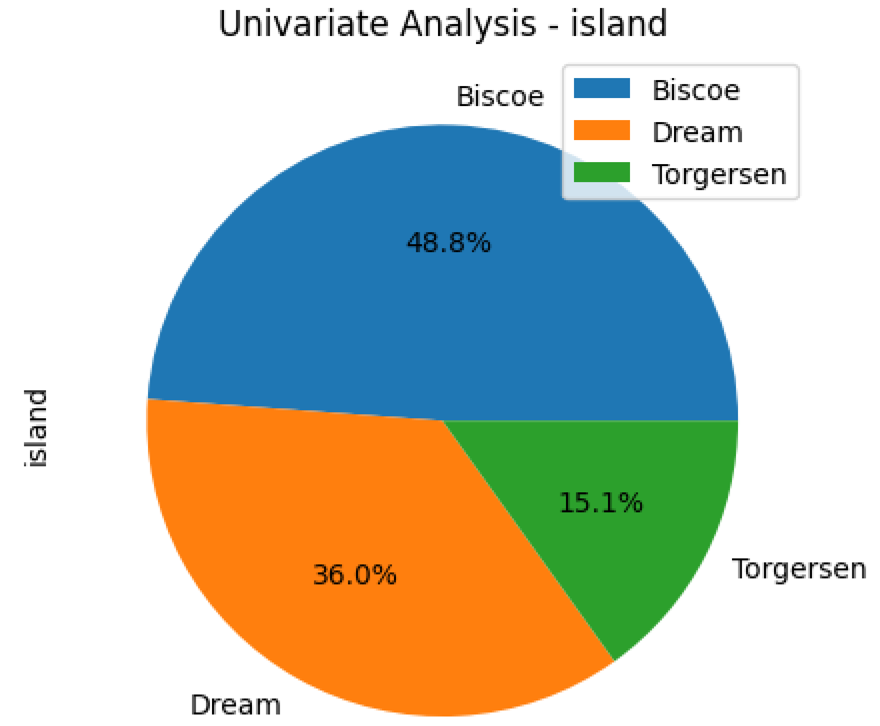
- Bi- Variate Analysis

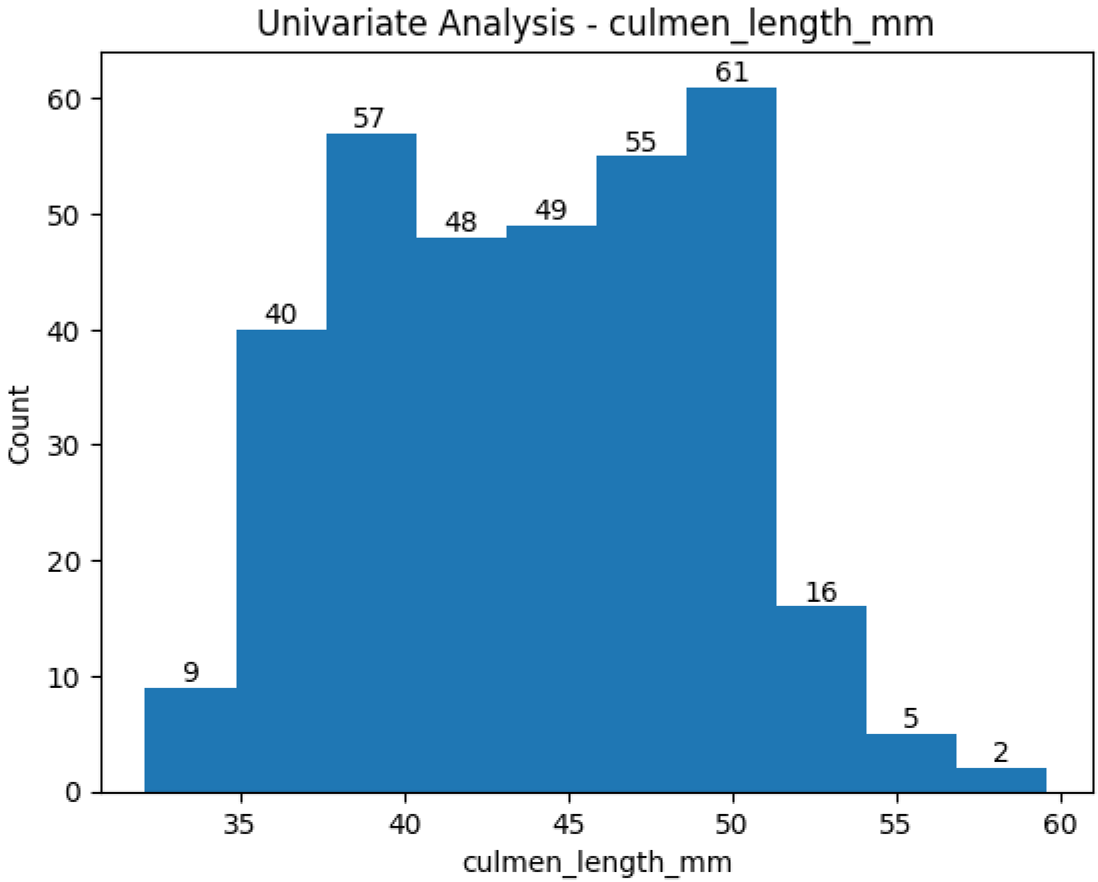
- Multi-Variate Analysis

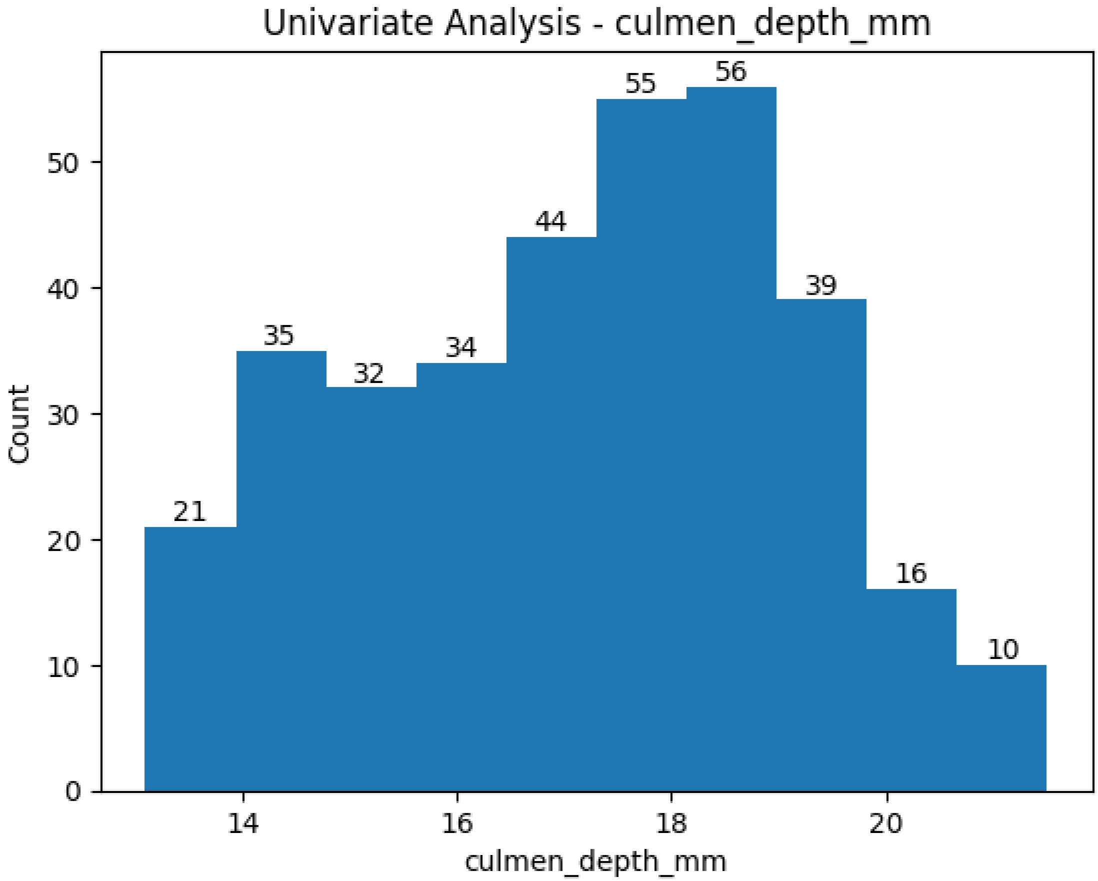
In [ ]:

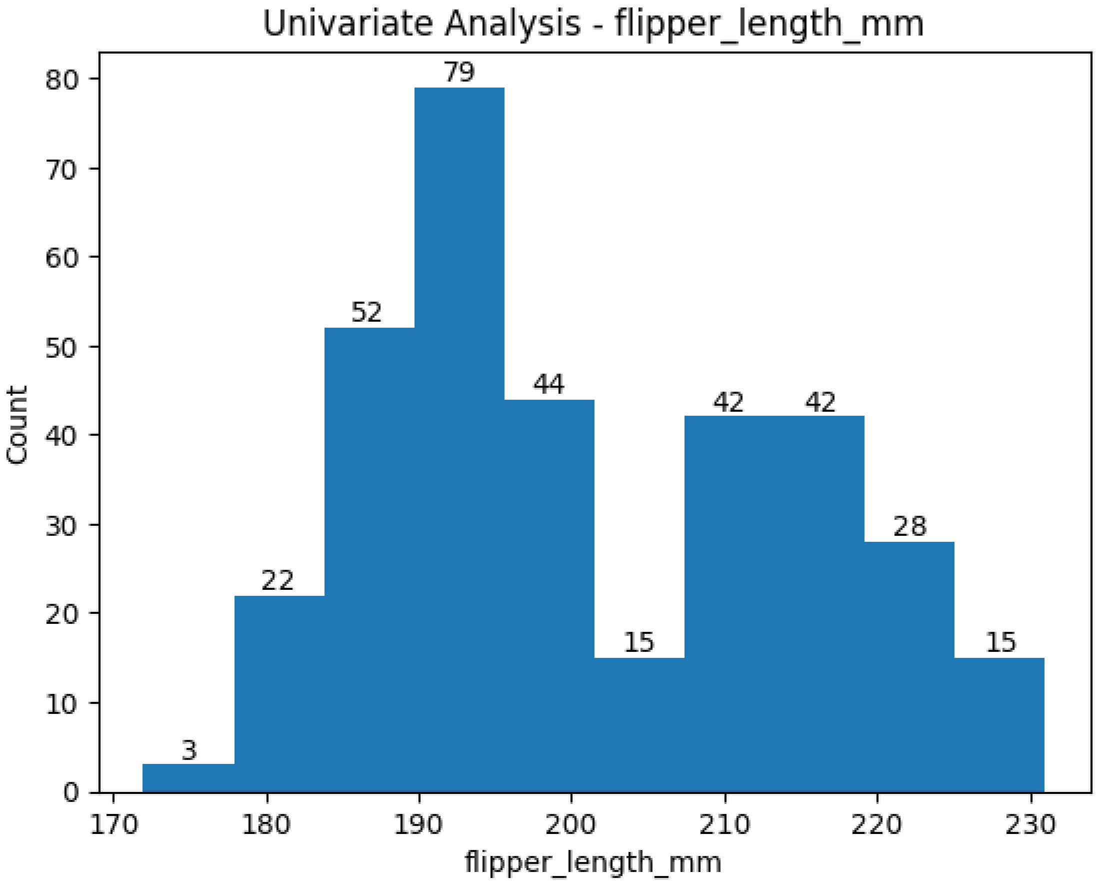
```
# Univariate Analysis

for i in df.columns:
    if df[i].dtype=="float64":
        counts, edges, bars = plt.hist(df[i])
        plt.title(f'Univariate Analysis - {i}')
        plt.xlabel(i)
        plt.ylabel("Count")
        plt.bar_label(bars)
        plt.show()
    else:
        df[i].value_counts().plot.pie(autopct='%1.1f%%')
        plt.title(f'Univariate Analysis - {i}')
        plt.legend()
        plt.show()
```

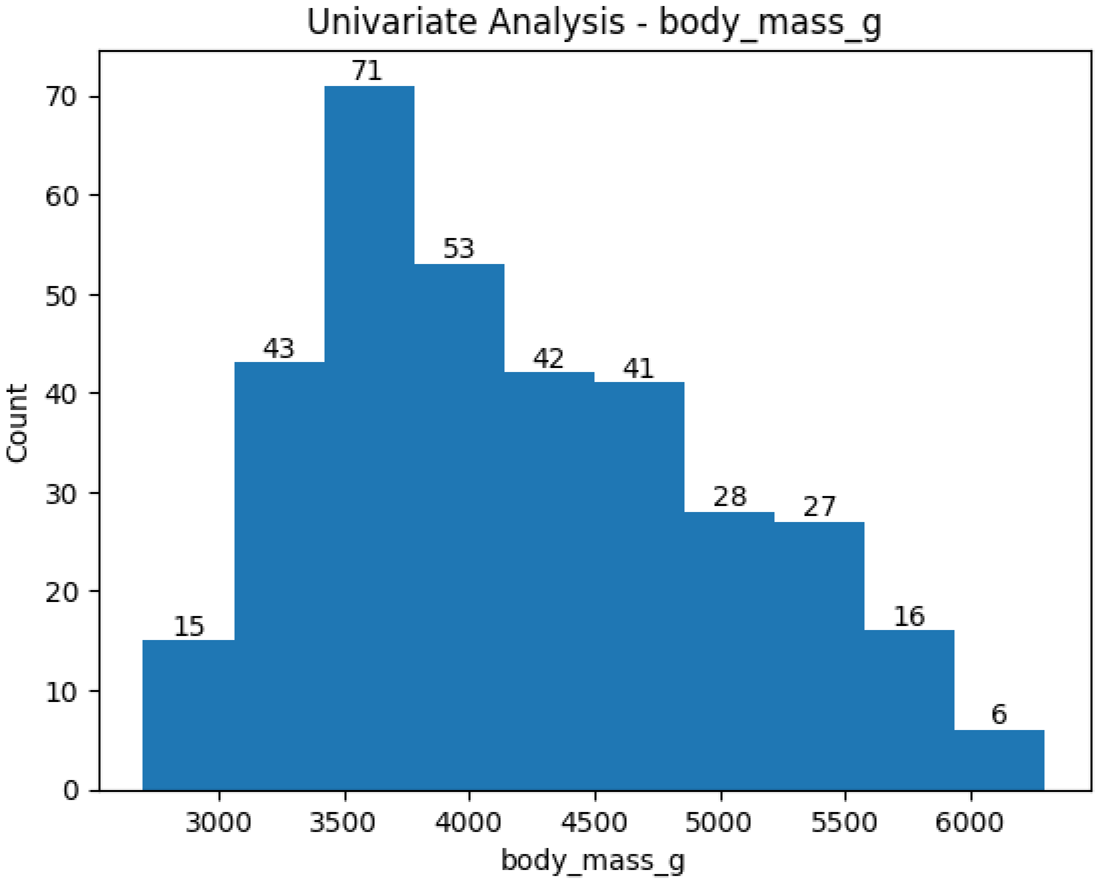

Univariate Analysis - species
Adelie
Gentoo
Chinstrap
species
44.2%
36.0%
19.8%
Adelie
Gentoo
Chinstrap

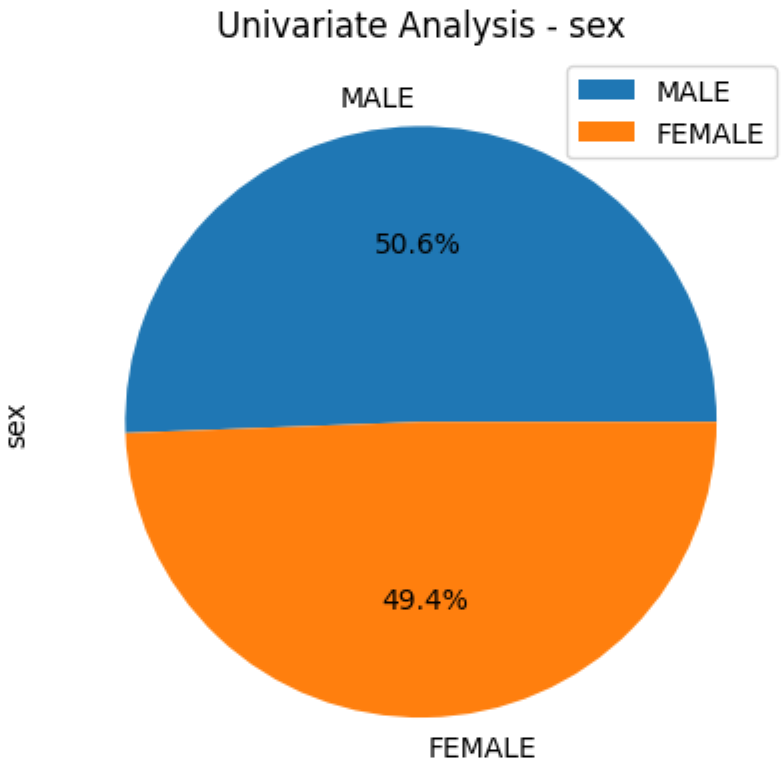

Univariate Analysis - island
Biscoe
Dream
Torgersen
island
48.8%
36.0%
15.1%
Biscoe
Dream
Torgersen


Univariate Analysis - culmen_length_mm
Count
culmen_length_mm
0
10
20
30
40
50
60
35
40
45
50
55
60
9
40
57
48
49
55
61
16
5
2


Univariate Analysis - culmen_depth_mm
Count
culmen_depth_mm
0
10
20
30
40
50
14
16
18
20
21
35
32
34
44
55
56
39
16
10

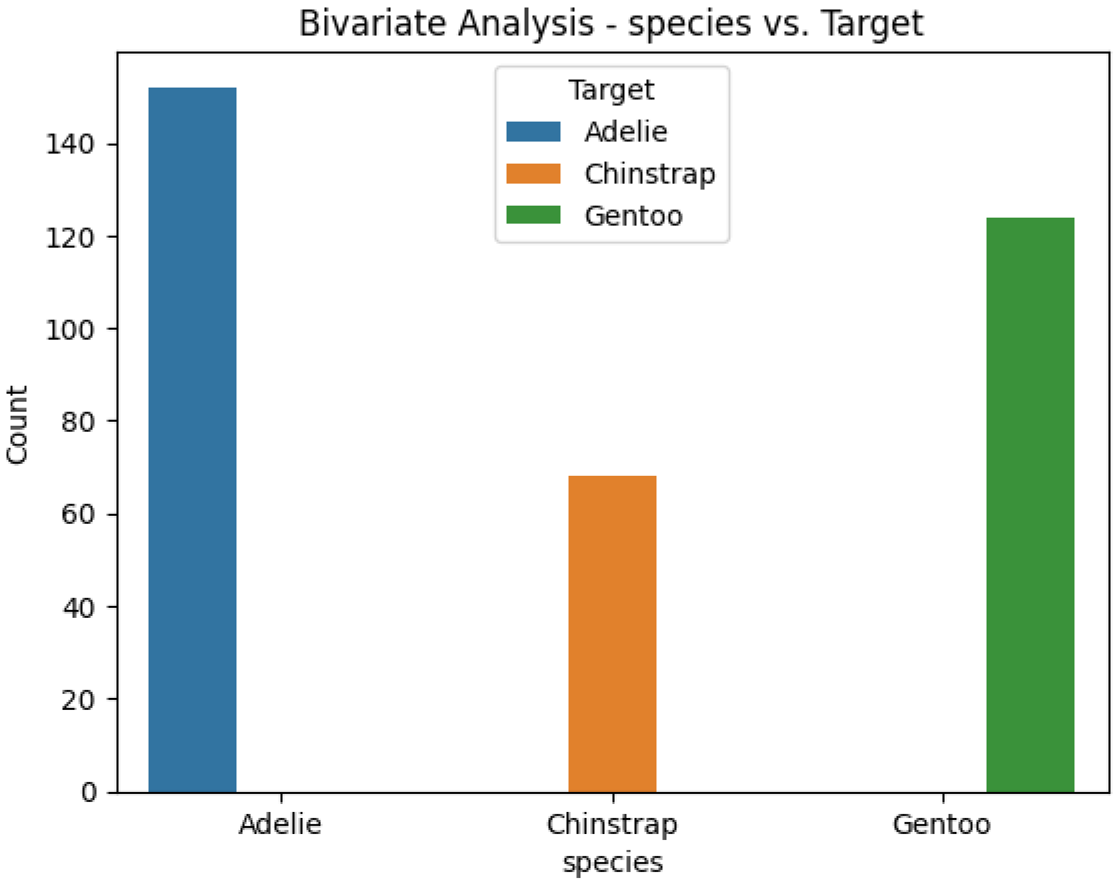

Univariate Analysis - flipper_length_mm
Count
0
10
20
30
40
50
60
70
80
3
22
52
79
44
15
42
42
28
15
170
180
190
200
210
220
230
flipper_length_mm

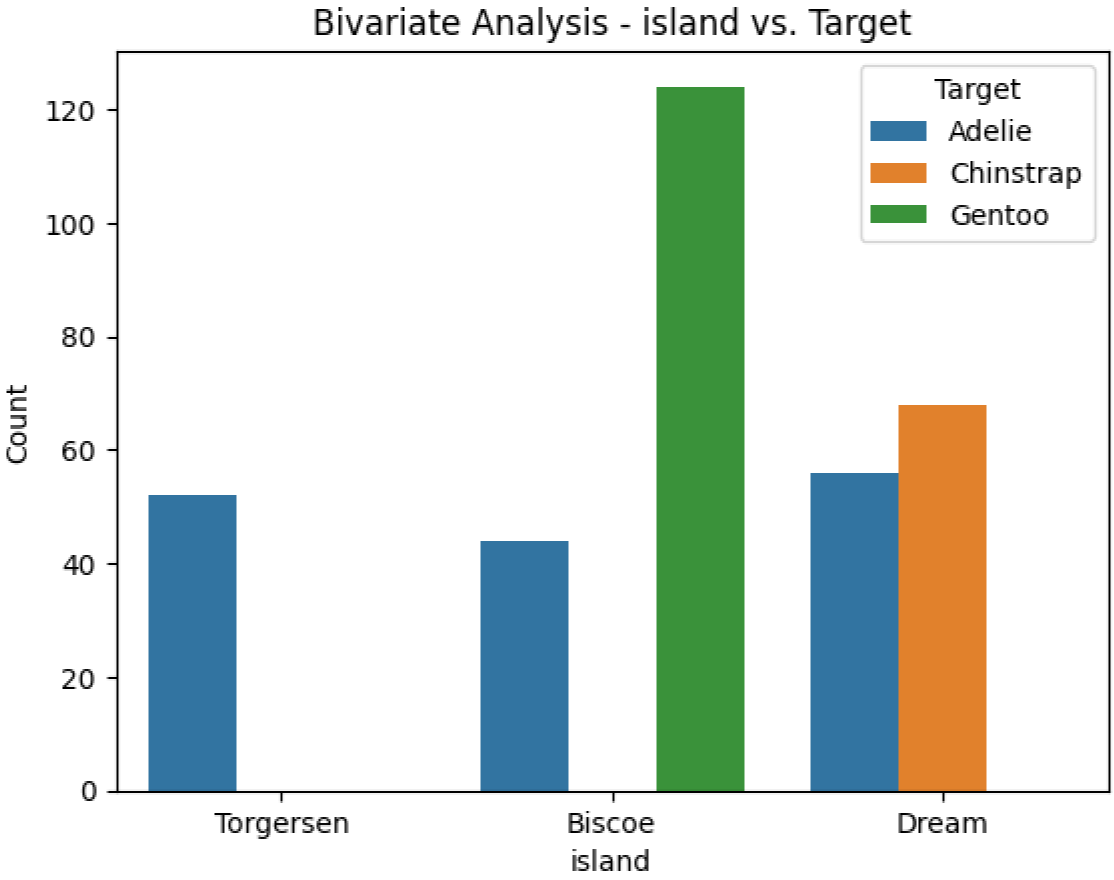

Univariate Analysis - body_mass_g
Count
0
10
20
30
40
50
60
70
15
43
71
53
42
41
28
27
16
6
3000
3500
4000
4500
5000
5500
6000
body_mass_g

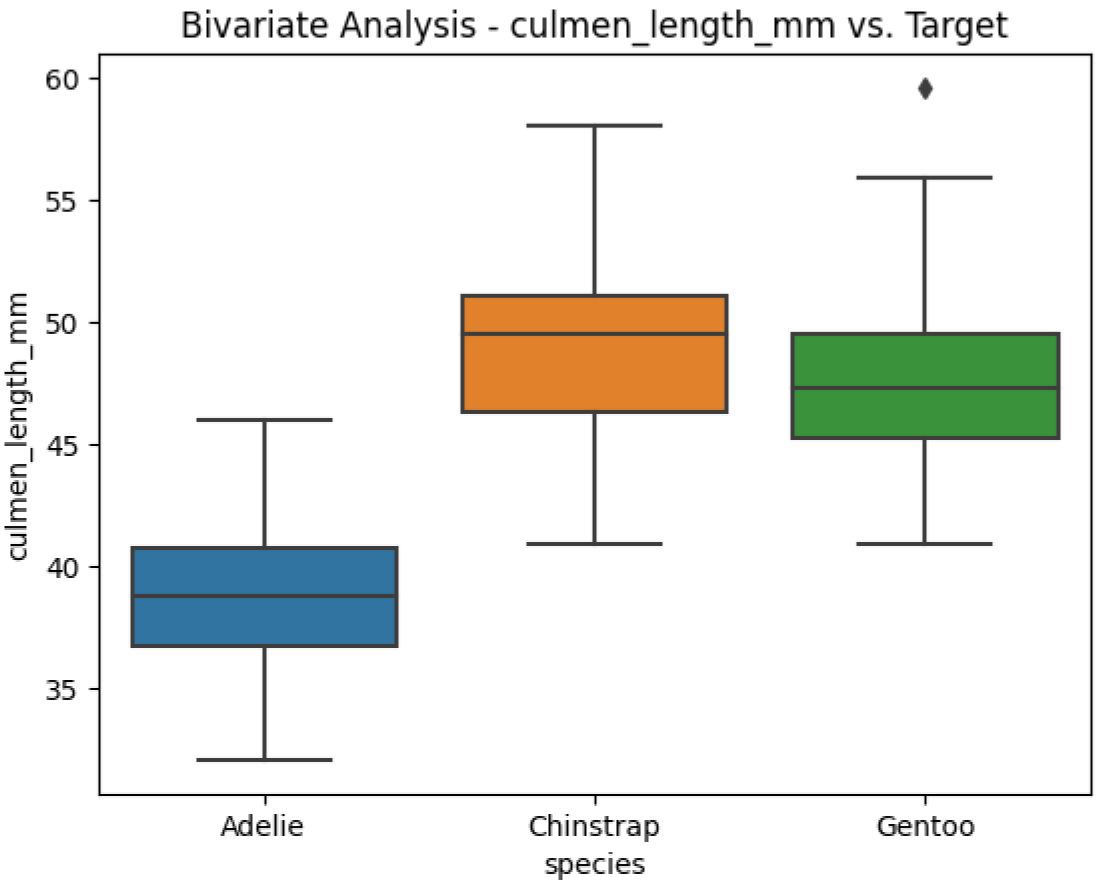


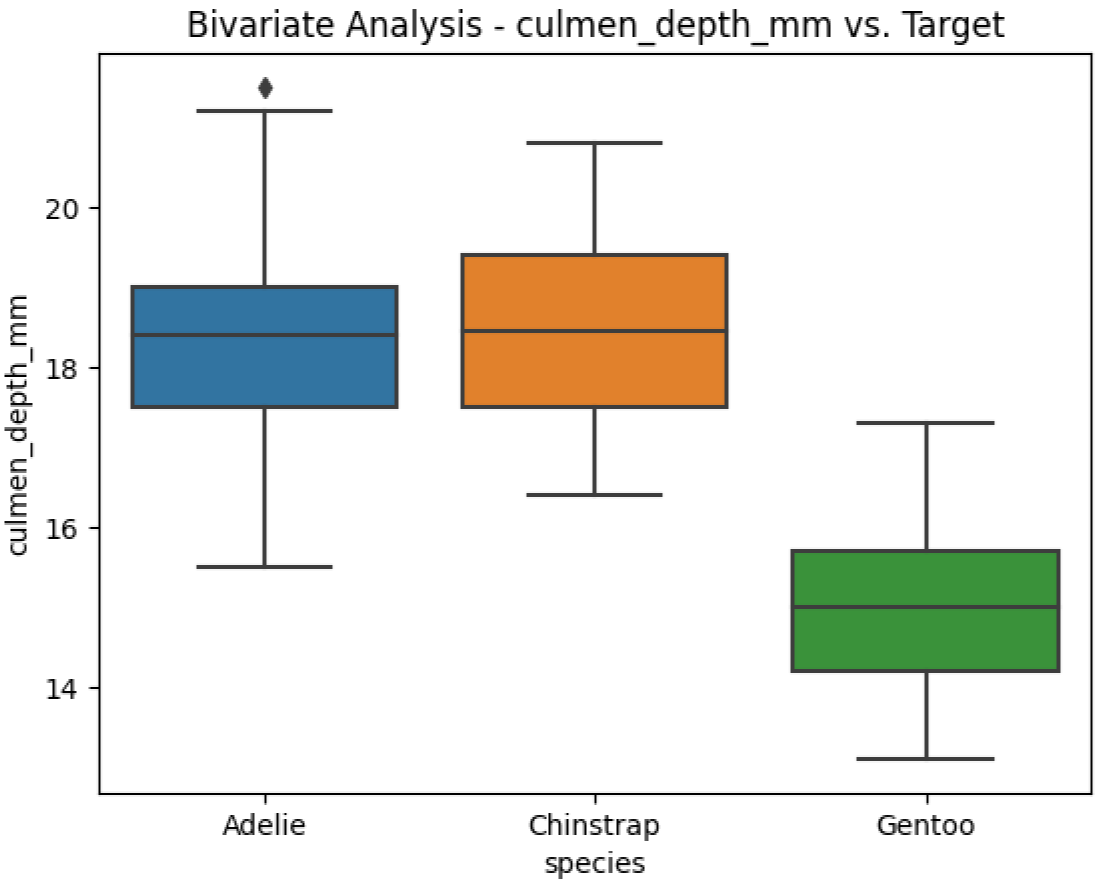
In [ ]:

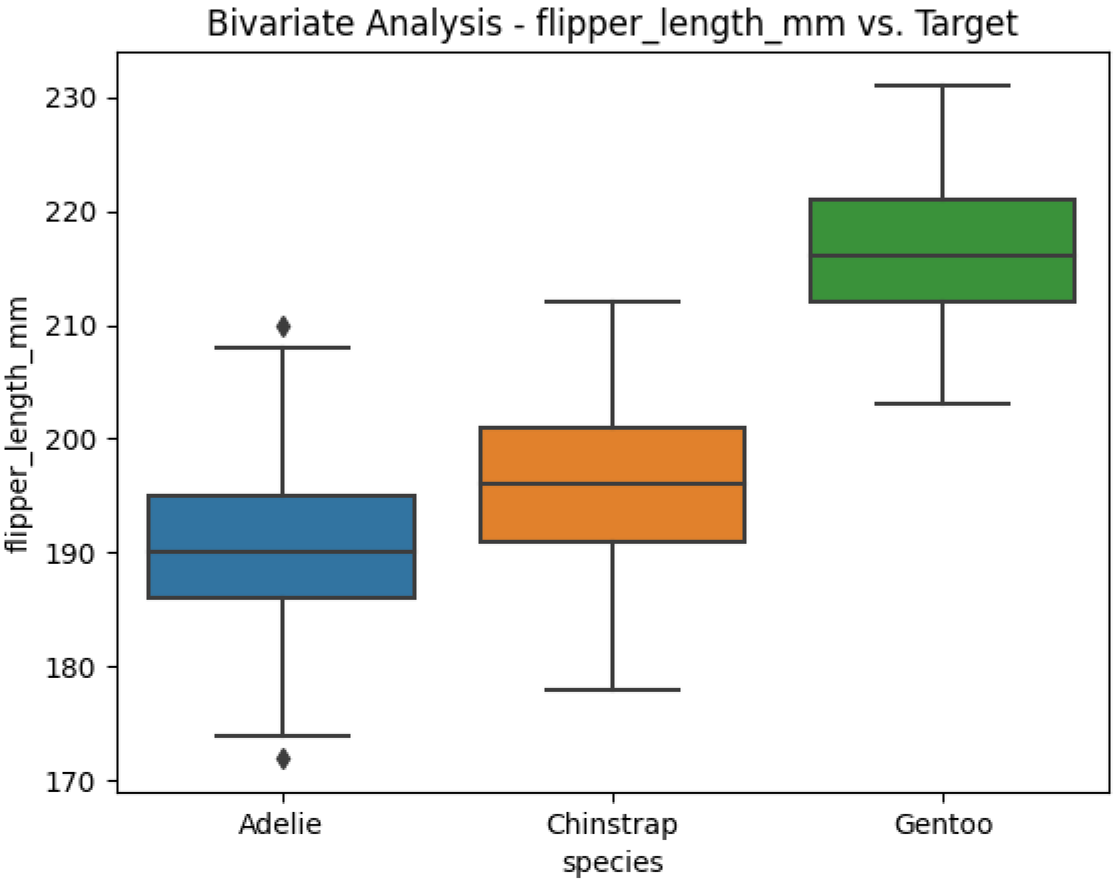
```
# Bivariate analysis

for col in df.columns:
    if df[col].dtype=='object':
        sns.countplot(data=df, x=col, hue='species')
        plt.xlabel(col)
        plt.ylabel('Count')
        plt.title(f'Bivariate Analysis - {col} vs. Target')
        plt.legend(title='Target')
        plt.show()
    else:
        sns.boxplot(data=df, x='species', y=col)
        plt.ylabel(col)
        plt.title(f'Bivariate Analysis - {col} vs. Target')
        plt.show()
```

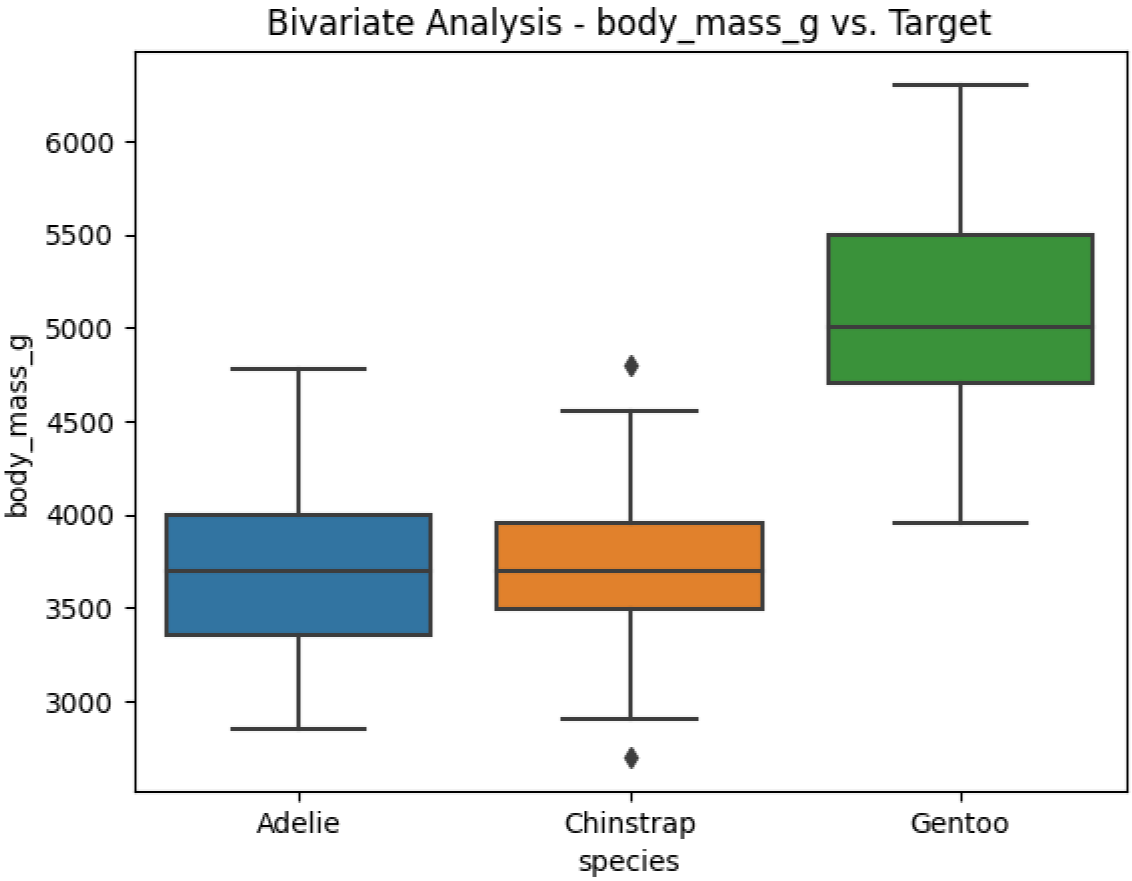

Bivariate Analysis - species vs. Target
Count
0
20
40
60
80
100
120
140
Adelie
Chinstrap
Gentoo
species
Target
Adelie
Chinstrap
Gentoo

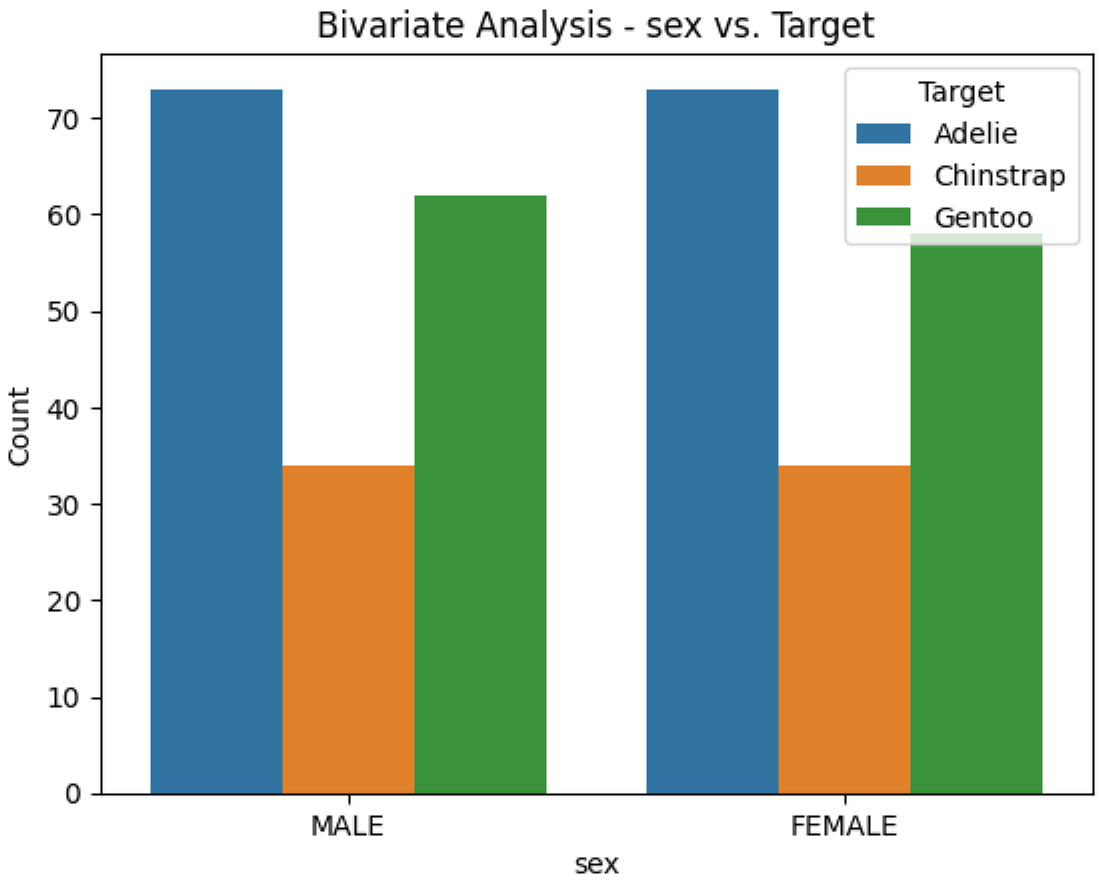

Bivariate Analysis - island vs. Target
Count
0
20
40
60
80
100
120
Torgersen
Biscoe
Dream
island
Target
Adelie
Chinstrap
Gentoo


Bivariate Analysis - culmen_length_mm vs. Target
culmen_length_mm
60
55
50
45
40
35
Adelie
Chinstrap
Gentoo
species


Bivariate Analysis - culmen_depth_mm vs. Target
culmen_depth_mm
20
18
16
14
Adelie
Chinstrap
Gentoo
species


Bivariate Analysis - flipper_length_mm vs. Target
flipper_length_mm
230
220
210
200
190
180
170
Adelie
Chinstrap
Gentoo
species


Bivariate Analysis - body_mass_g vs. Target
body_mass_g
6000
5500
5000
4500
4000
3500
3000
Adelie
Chinstrap
Gentoo
species

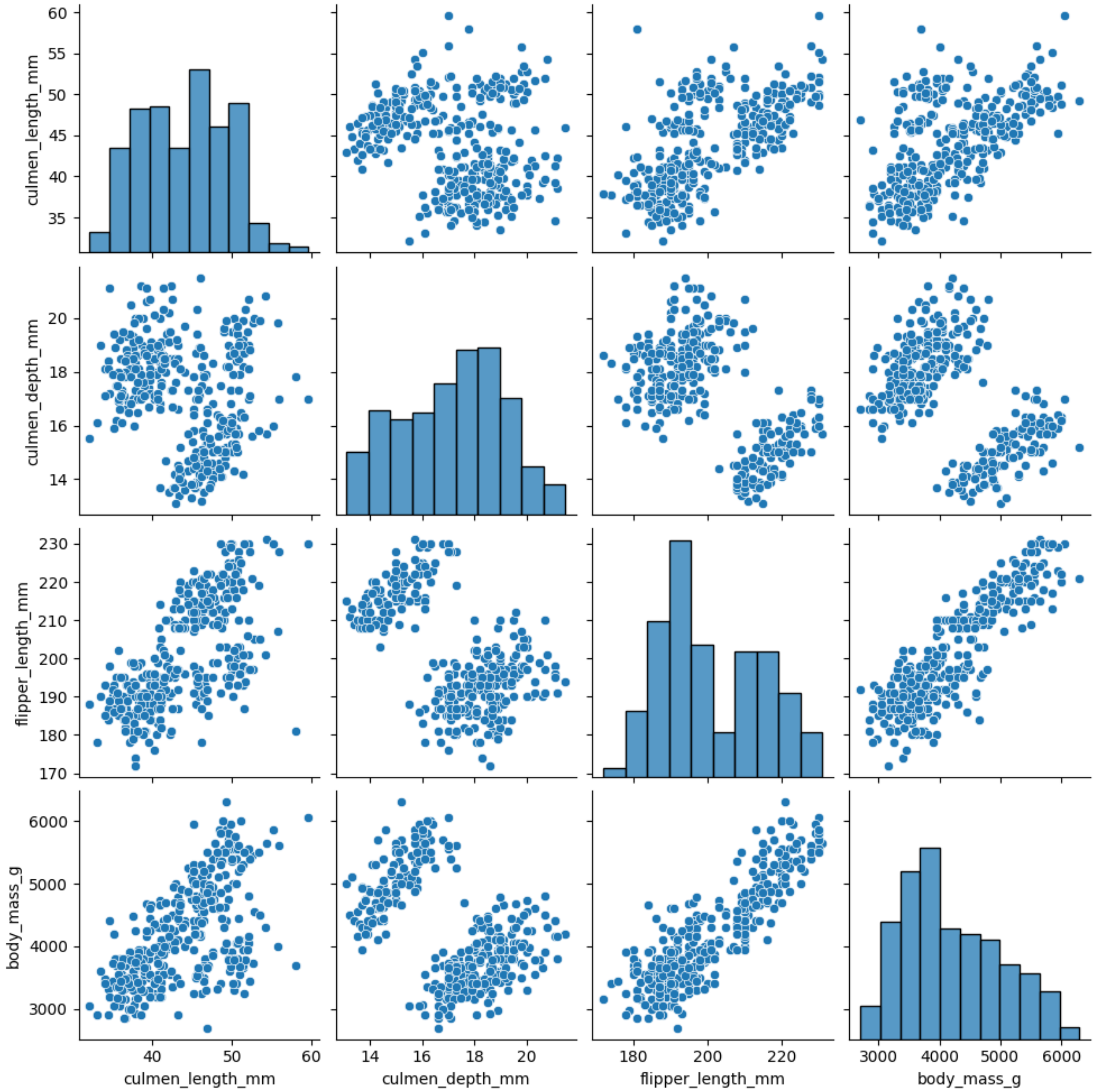


In [ ]:

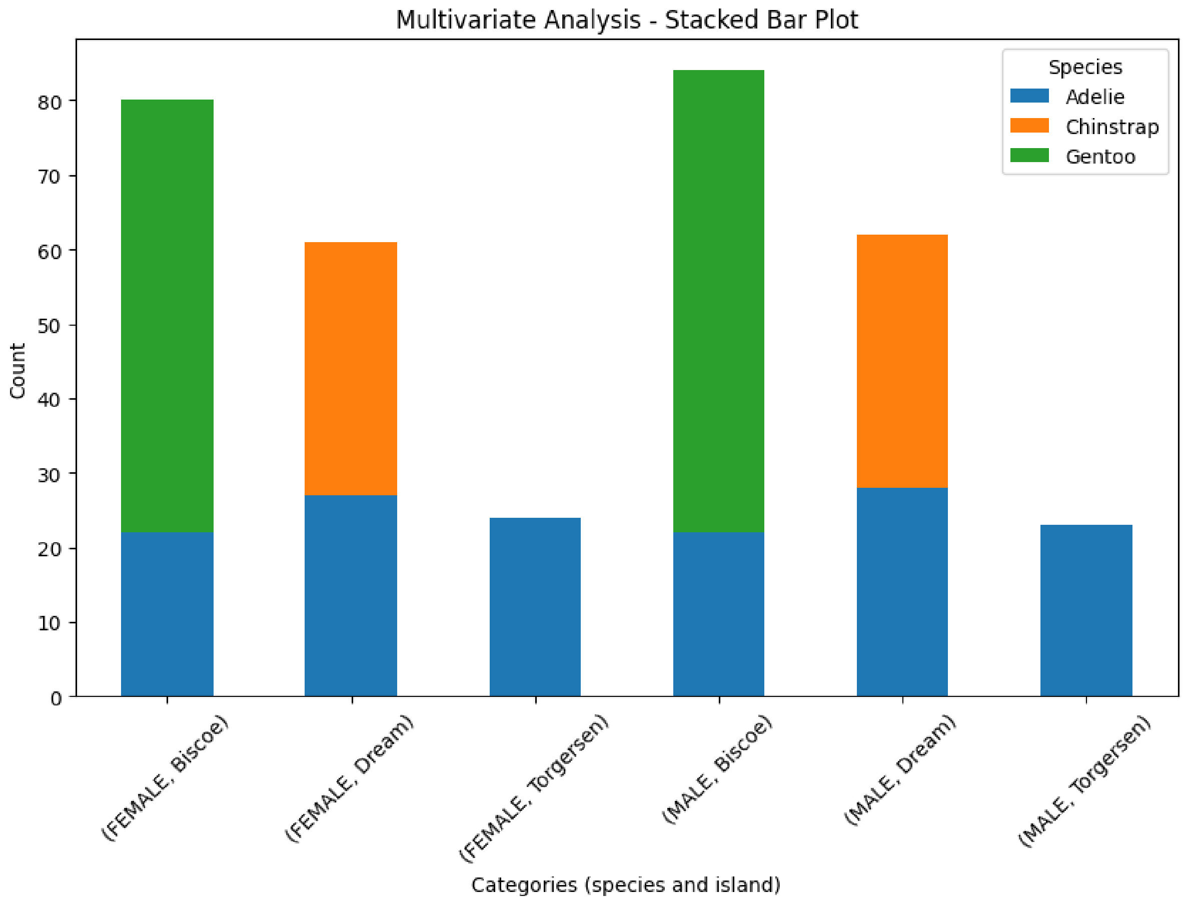
```
# Multivariate Analysis
sns.pairplot(df)
```

Out[ ]:

```
<seaborn.axisgrid.PairGrid at 0x1feefce5210>
```



In [ ]:
```
cross_tab = pd.crosstab([ df['sex'], df['island']],df['species'])

cross_tab.plot(kind='bar', stacked=True, figsize=(10, 6))
plt.xlabel('Categories (species and island)')
plt.ylabel('Count')
plt.title('Multivariate Analysis - Stacked Bar Plot')
plt.legend(title='Species')
plt.xticks(rotation=45)
plt.show()
```



4. Perform descriptive statistics on the dataset.

```
In [ ]: df.describe()
```

Out[ ]:

	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

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

Out[ ]:

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

In [ ]:

```python
df.culmen_depth_mm.fillna(df.culmen_depth_mm.mean(),inplace = True)
df.culmen_length_mm.fillna(df.culmen_length_mm.mean(),inplace = True)
df.flipper_length_mm.fillna(df.flipper_length_mm.mean(),inplace = True)
df.body_mass_g.fillna(df.body_mass_g.mean(),inplace = True)
df.sex.fillna(df.sex.mode()[0],inplace = True)
```

In [ ]:

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

Out[ ]:

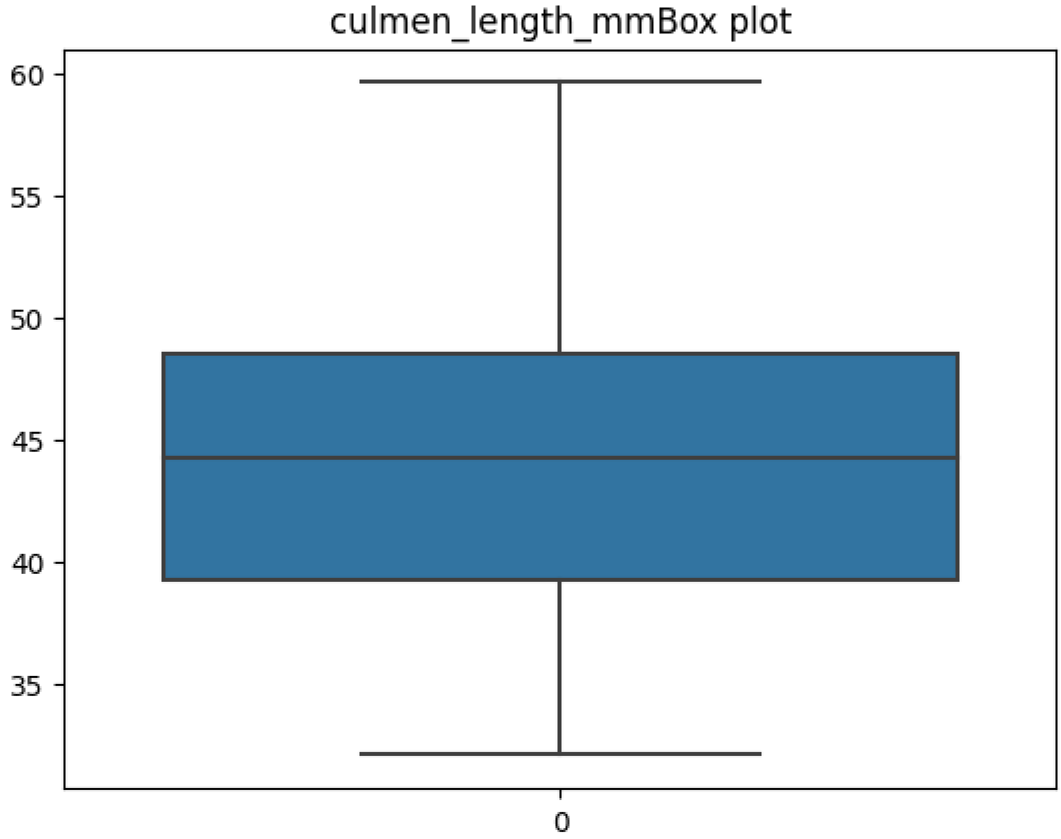
```
species              0
island               0
culmen_length_mm     0
culmen_depth_mm      0
flipper_length_mm    0
body_mass_g          0
sex                  0
dtype: int64
```

6. Find the outliers and replace them outliers

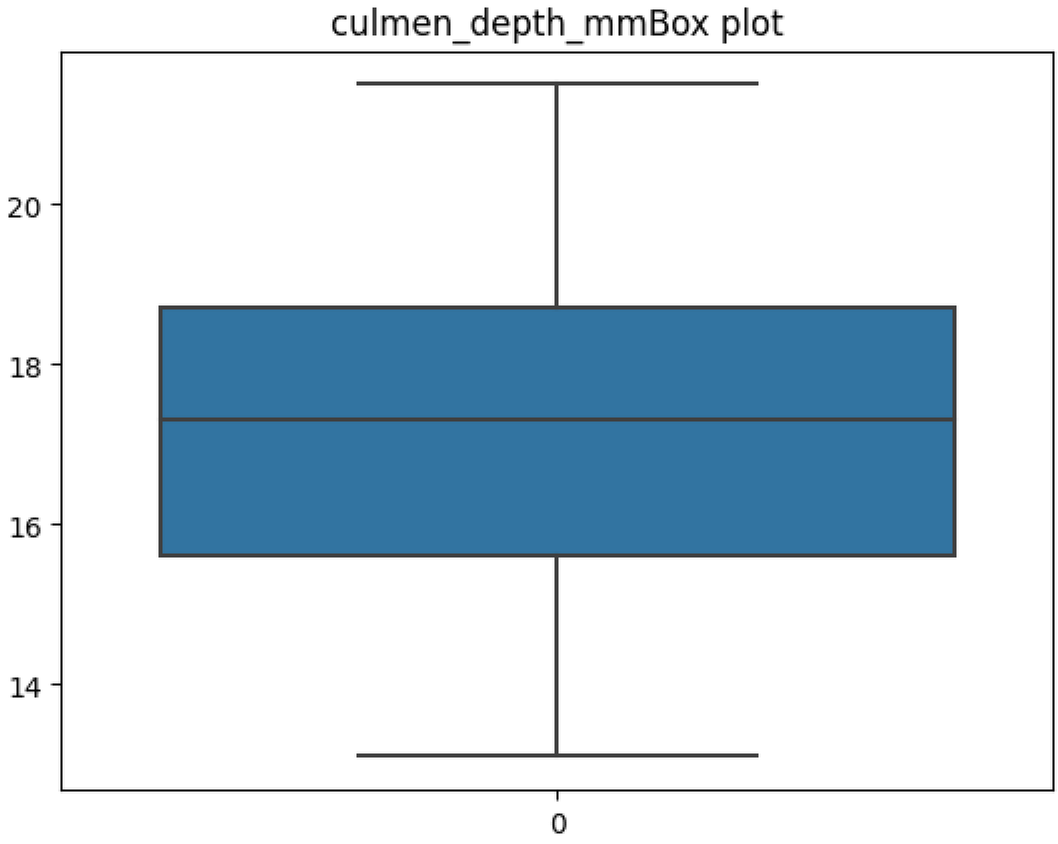
In [ ]:

```python
def replace_outliers(df, c):
    Q1 = df[c].quantile(0.25)
    Q3 = df[c].quantile(0.75)
    IQR = Q3 - Q1
    lower_bound = Q1 - 1.5 * IQR
    upper_bound = Q3 + 1.5 * IQR

    print("Column:",c)
    print("Lower Bound:",lower_bound)
    print("Upper Bound:",upper_bound)
    print("Inter Quartile Range:",IQR)

    sns.boxplot(df[c])
    plt.title(c+"Box plot")
    plt.show()

    outliers = (df[c] < lower_bound) | (df[c] > upper_bound)

    median = df[~outliers][c].median()
    df.loc[outliers, c] = median
    print("Outliers in column",c,"have been replace with the median\n")

columns = ['culmen_length_mm', 'culmen_depth_mm','flipper_length_mm','body_mass_g']

for column in columns:
    replace_outliers(df, column)
```

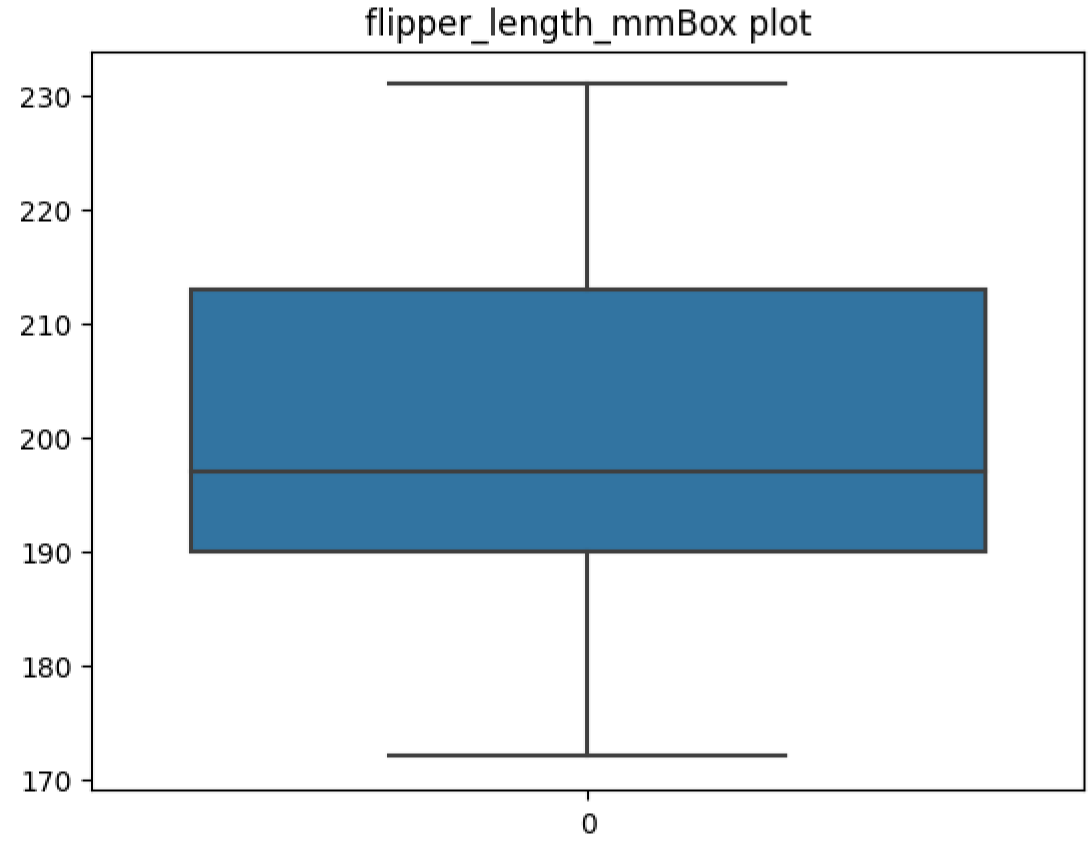
```
Column: culmen_length_mm
Lower Bound: 25.437499999999996
Upper Bound: 62.337500000000006
Inter Quartile Range: 9.225000000000001
```



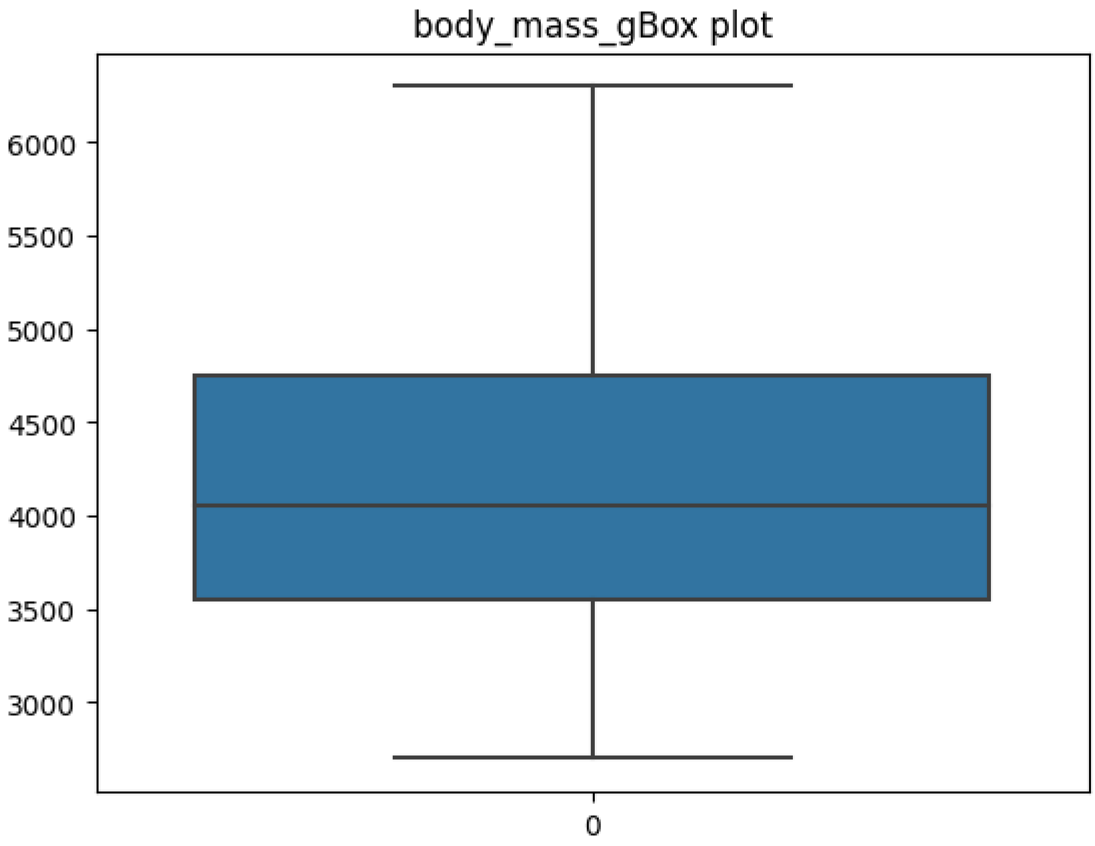
```
Outliers in column culmen_length_mm have been replace with the median

Column: culmen_depth_mm
Lower Bound: 10.95
Upper Bound: 23.349999999999998
Inter Quartile Range: 3.0999999999999996
```



```
Outliers in column culmen_depth_mm have been replace with the median

Column: flipper_length_mm
Lower Bound: 155.5
Upper Bound: 247.5
Inter Quartile Range: 23.0
```



```
Outliers in column flipper_length_mm have been replace with the median

Column: body_mass_g
Lower Bound: 1750.0
Upper Bound: 6550.0
Inter Quartile Range: 1200.0
```



```
Outliers in column body_mass_g have been replace with the median
```

8. Check for Categorical columns and perform encoding.

In [ ]:

```
from sklearn.preprocessing import LabelEncoder

label = LabelEncoder()
for i in df.columns:
    if df[i].dtype == 'object':
        df[i] = label.fit_transform(df[i])
```

7. Check the correlation of independent variables with the target

In [ ]:

```
df.corr()['species'][:]
```

Out[ ]:

```
species              1.000000
island              -0.635659
culmen_length_mm     0.728674
culmen_depth_mm     -0.741335
flipper_length_mm    0.851160
body_mass_g          0.747726
sex                  0.010240
Name: species, dtype: float64
```

9. Split the data into dependent and independent variables.

In [ ]:

```
X = df.drop(columns = ['species'])
y = df.species
```

10. Scaling the data

In [ ]:

```
from sklearn.preprocessing import MinMaxScaler

scaler = MinMaxScaler()
X = pd.DataFrame(scaler.fit_transform(X),columns = X.columns)
```

11. Split the data into training and testing

In [ ]:

```
from sklearn.model_selection import train_test_split

X_train, X_test,y_train, y_test = train_test_split(X,y ,random_state=104,test_size=
```

12. Check the training and testing data shape

In [ ]:

```
print("X_train",X_train.shape)
print("X_test",X_test.shape)
print("y_train",y_train.shape)
print("y_test", y_test.shape)
```

```
X_train (240, 6)
X_test (104, 6)
y_train (240,)
y_test (104,)
```