assignment-3

September 8, 2023

0.0.1 Pre-requisites

[377]:
```
import pandas as pd
from sklearn import preprocessing
```

[378]:
```
df = pd.read_csv('./Dataset.csv')
```

0.0.2 Data Pre-processing

[379]:
```
df.shape
```

[379]:
```
(344, 7)
```

[380]:
```
df.columns
```

[380]:
```
Index(['species', 'island', 'culmen_length_mm', 'culmen_depth_mm',
       'flipper_length_mm', 'body_mass_g', 'sex'],
      dtype='object')
```

[381]:
```
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
```

[382]:
```
df.head(5)
```

[382]:
```
  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
```

[383]:
```
df.isnull().values.any()
```

[383]:
```
True
```

[384]:
```
df.describe()
```

[384]:
```
       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
```

[385]:
```
df.culmen_length_mm.median()
```

[385]:
```
44.45
```

[386]:
```
df = df.fillna(df.median())
print(df)
```

```
    species     island  culmen_length_mm  culmen_depth_mm  flipper_length_mm  \
0    Adelie  Torgersen             39.10             18.7              181.0
1    Adelie  Torgersen             39.50             17.4              186.0
2    Adelie  Torgersen             40.30             18.0              195.0
3    Adelie  Torgersen             44.45             17.3              197.0
4    Adelie  Torgersen             36.70             19.3              193.0
..      ...        ...               ...              ...                ...
339  Gentoo     Biscoe             44.45             17.3              197.0
340  Gentoo     Biscoe             46.80             14.3              215.0
341  Gentoo     Biscoe             50.40             15.7              222.0
342  Gentoo     Biscoe             45.20             14.8              212.0
```

```
343  Gentoo     Biscoe            49.90             16.1              213.0

     body_mass_g     sex
0         3750.0    MALE
1         3800.0  FEMALE
2         3250.0  FEMALE
3         4050.0     NaN
4         3450.0  FEMALE
..           …      …
339       4050.0     NaN
340       4850.0  FEMALE
341       5750.0    MALE
342       5200.0  FEMALE
343       5400.0    MALE

[344 rows x 7 columns]
```

```
<ipython-input-386-276bd5f8c552>: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())
```

[387]:
```
dummies = pd.get_dummies(df['sex'])

# Concatenate the dummies to original dataframe
merged = pd.concat([df, dummies], axis='columns')
# drop the values
df = merged.drop(['sex'], axis='columns')
```

[388]:
```
label_encoder = preprocessing.LabelEncoder()

# Encode labels in column 'species'.
df['species'] = label_encoder.fit_transform(df['species'])

df['species'].unique()
```

[388]:
```
array([0, 1, 2])
```

[389]:
```
label_encoder = preprocessing.LabelEncoder()

# Encode labels in column 'island'.
df['island'] = label_encoder.fit_transform(df['island'])

df['island'].unique()
```

[389]:
```
array([2, 0, 1])
```

[390]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 344 entries, 0 to 343
Data columns (total 9 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    344 non-null    float64
 4   flipper_length_mm  344 non-null    float64
 5   body_mass_g        344 non-null    float64
 6   .                  344 non-null    uint8
 7   FEMALE             344 non-null    uint8
 8   MALE               344 non-null    uint8
dtypes: float64(4), int64(2), uint8(3)
memory usage: 17.3 KB
```

[391]:
```
df.drop('.', axis='columns')
```

[391]:

	species	island	culmen_length_mm	culmen_depth_mm	flipper_length_mm	body_mass_g	FEMALE	MALE
0	0	2	39.10	18.7	181.0	3750.0	0	1
1	0	2	39.50	17.4	186.0	3800.0	1	0
2	0	2	40.30	18.0	195.0	3250.0	1	0
3	0	2	44.45	17.3	197.0	4050.0	0	0
4	0	2	36.70	19.3	193.0	3450.0	1	0
..	…	…	…	…	…	…	…	…
339	2	0	44.45	17.3	197.0	4050.0	0	0
340	2	0	46.80	14.3	215.0	4850.0	1	0
341	2	0	50.40	15.7	222.0	5750.0	0	1
342	2	0	45.20	14.8	212.0	5200.0	1	0
343	2	0	49.90	16.1	213.0	5400.0	0	1

```
[344 rows x 8 columns]
```

[392]:
```
df.isnull().values.any()
```

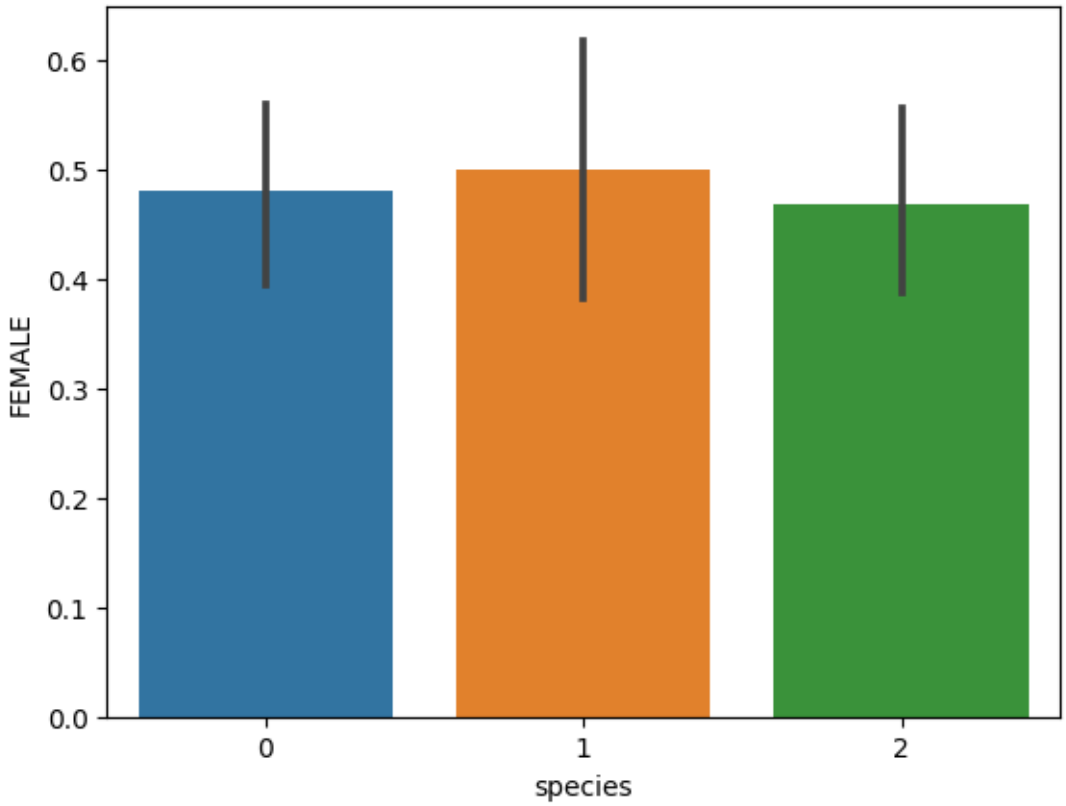
[392]:
```
False
```

0.0.3 Data Visualisation

[393]:
```
import seaborn as sns
import matplotlib.pyplot as plt
```

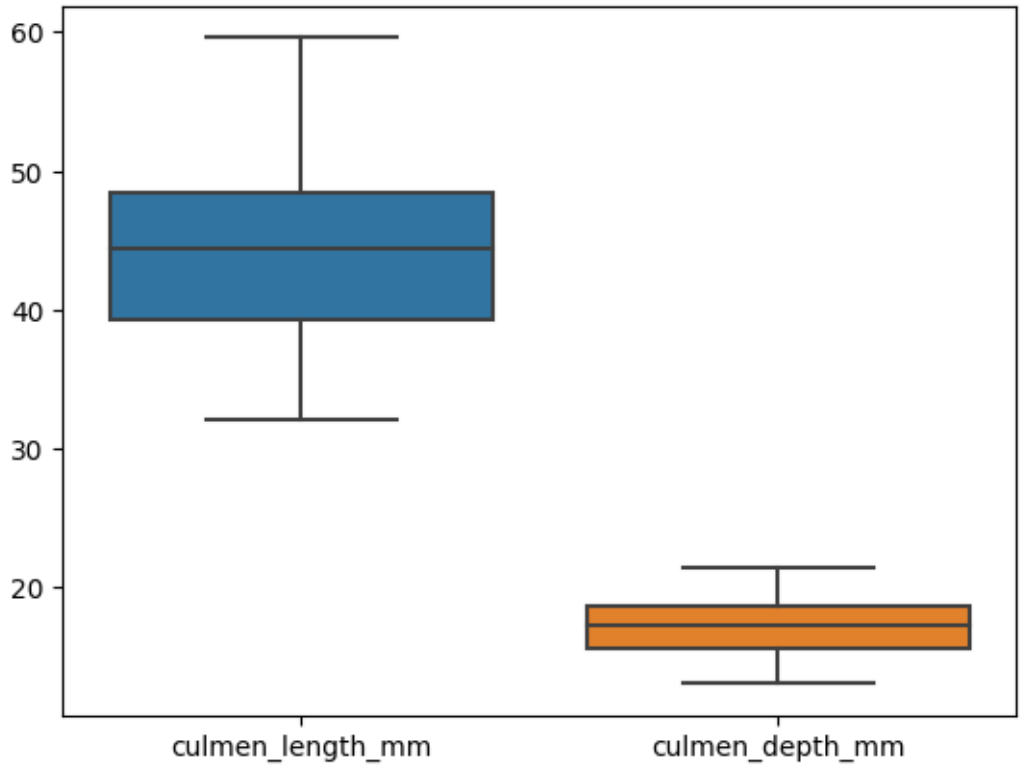
[394]:
```
sns.barplot(x=df['species'], y=df['FEMALE'])
```

[394]:
```
<Axes: xlabel='species', ylabel='FEMALE'>
```



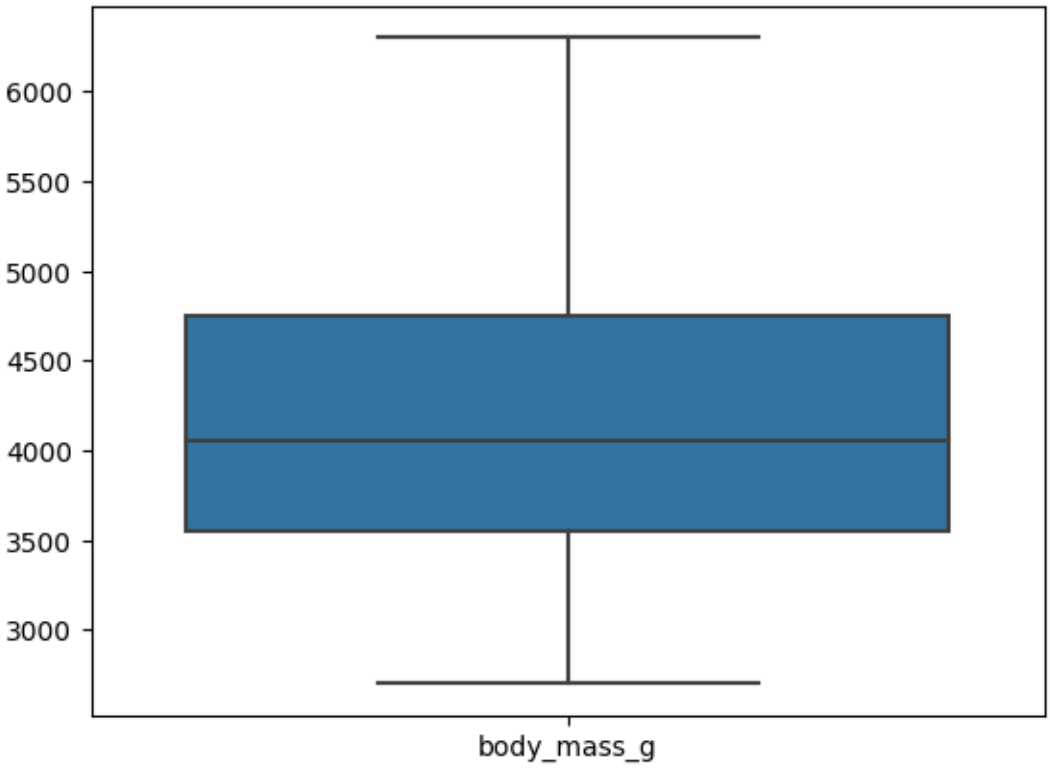
[395]:
```
sns.boxplot(data=df[['culmen_length_mm', 'culmen_depth_mm']])
```

[395]:
```
<Axes: >
```



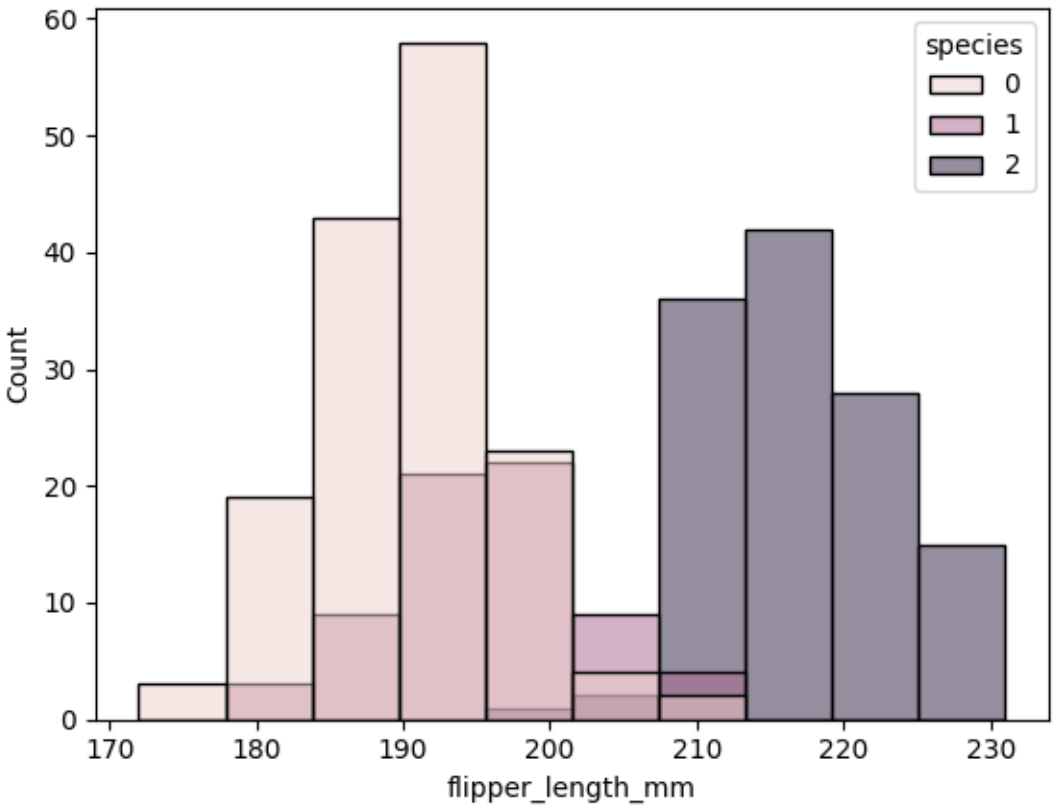
[396]:
```
sns.boxplot(data=df[['body_mass_g']])
```

[396]: <Axes: >



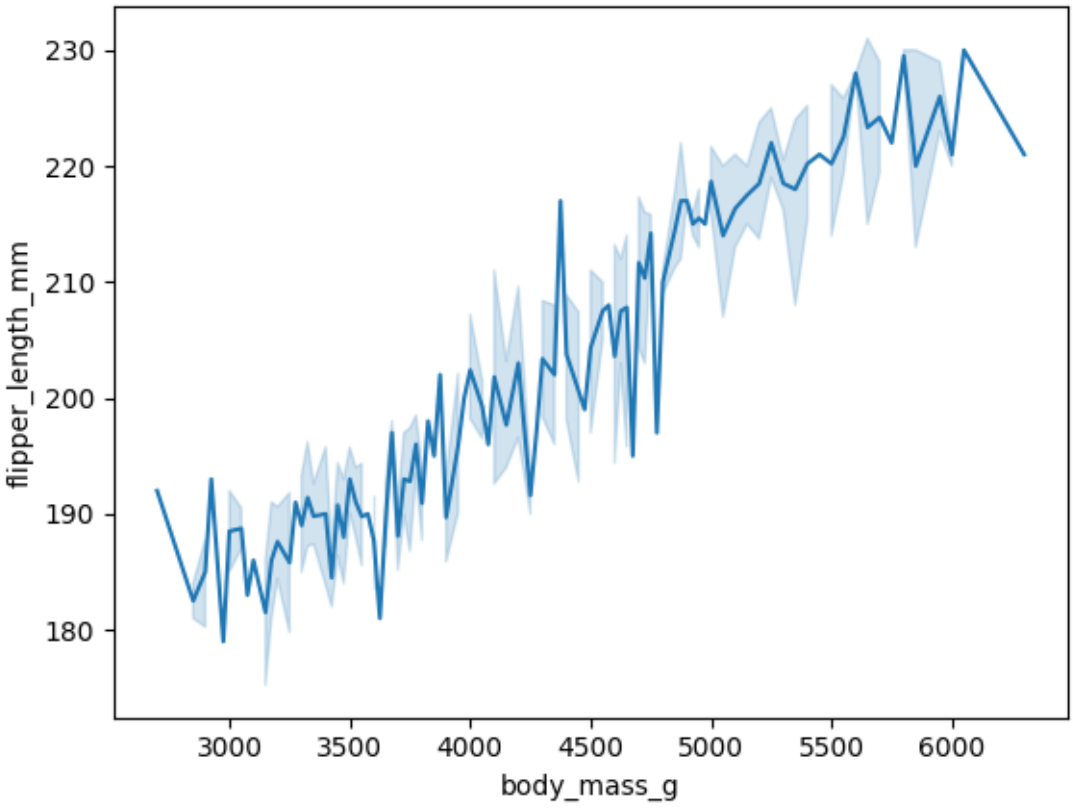
[397]:
```
sns.histplot(data=df, x="flipper_length_mm", hue='species')
```

[397]:
```
<Axes: xlabel='flipper_length_mm', ylabel='Count'>
```



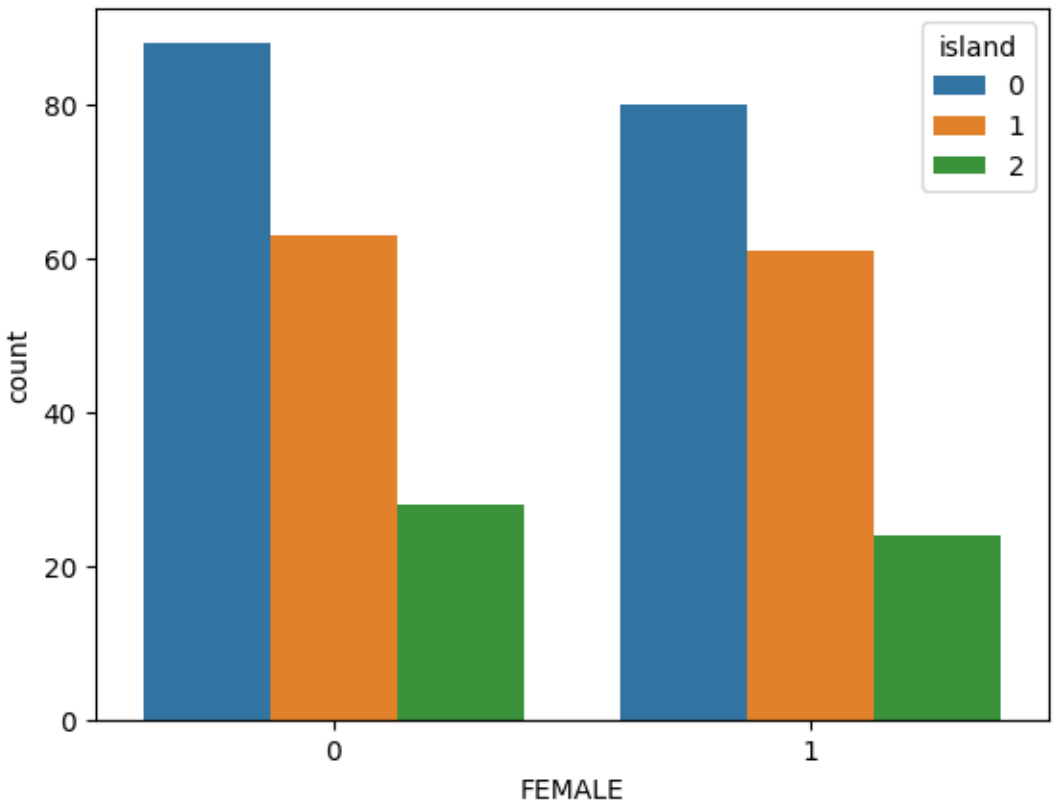
[398]:
```
sns.lineplot(data=df, y="flipper_length_mm", x="body_mass_g")
```

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



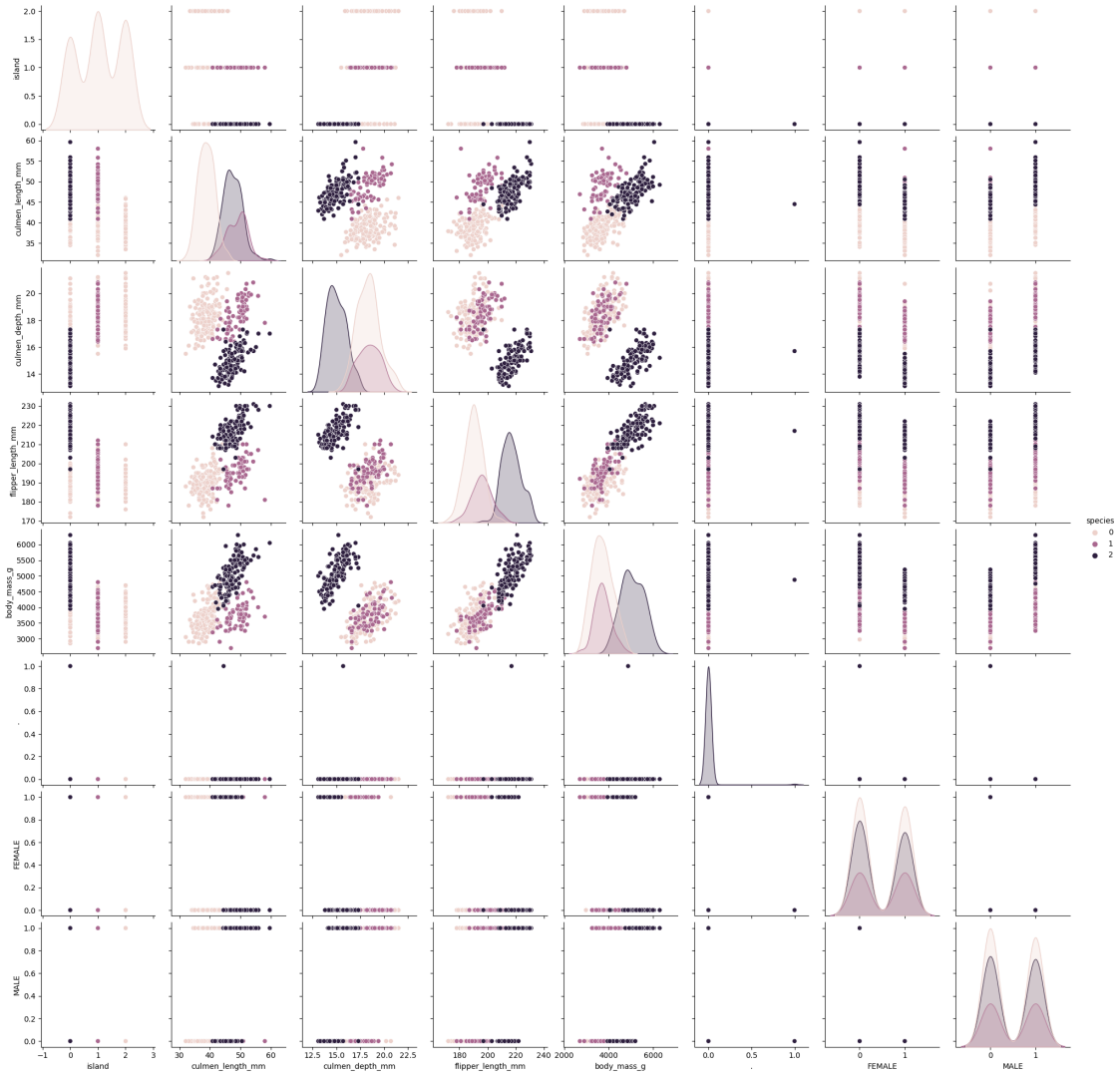
[399]:
```
sns.countplot(data=df, x="FEMALE", hue="island")
```

[399]:
```
<Axes: xlabel='FEMALE', ylabel='count'>
```



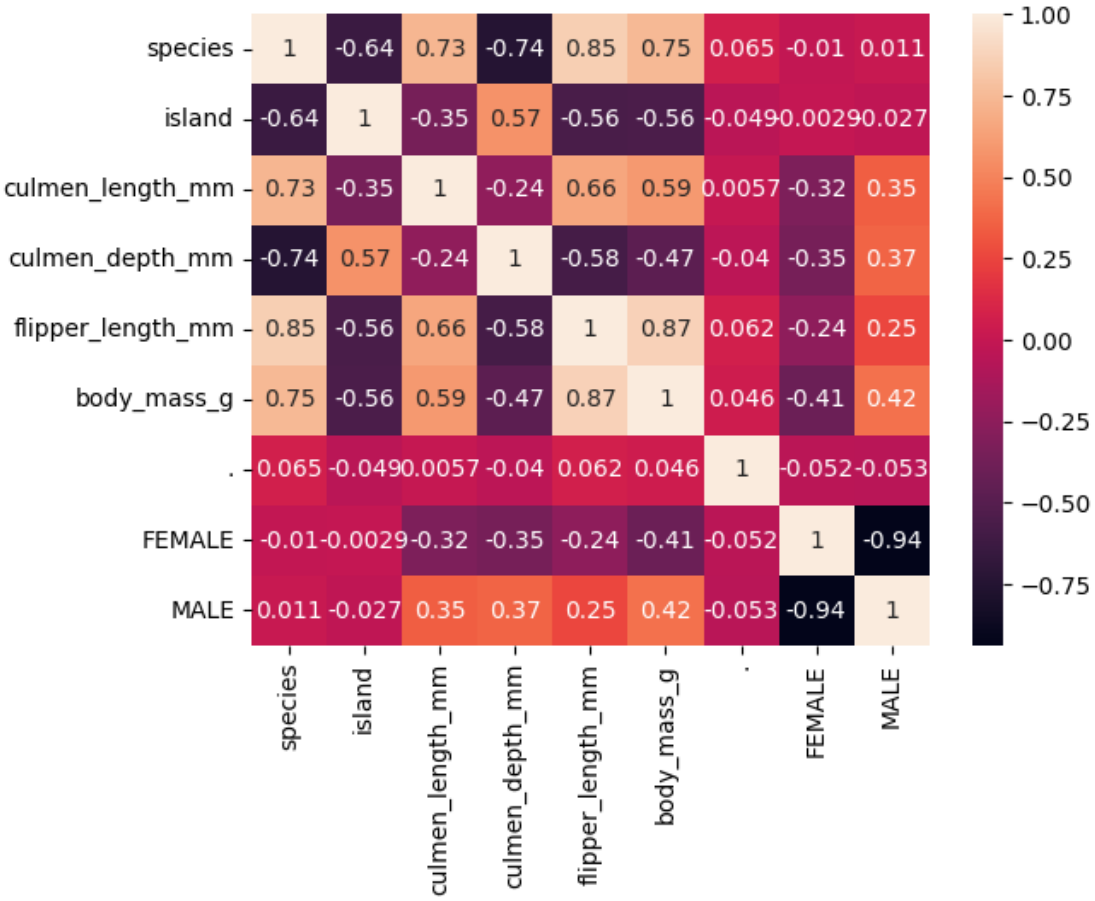
[400]:
```
sns.pairplot(df, hue="species")
```

[400]: <seaborn.axisgrid.PairGrid at 0x7e71a818ee30>



[401]:
```
sns.heatmap(df.corr(), annot=True)
```

[401]: <Axes: >



0.0.4 ML Model

[402]:
```
df
```

[402]:

	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	
..	…	…	…	…	…	
339	2	0	44.45	17.3	197.0	
340	2	0	46.80	14.3	215.0	
341	2	0	50.40	15.7	222.0	
342	2	0	45.20	14.8	212.0	
343	2	0	49.90	16.1	213.0	

	body_mass_g	.	FEMALE	MALE

```
0         3750.0  0       0     1
1         3800.0  0       1     0
2         3250.0  0       1     0
3         4050.0  0       0     0
4         3450.0  0       1     0
..           … ..     …    …
339       4050.0  0       0     0
340       4850.0  0       1     0
341       5750.0  0       0     1
342       5200.0  0       1     0
343       5400.0  0       0     1

[344 rows x 9 columns]
```

[403]:
```
# import pandas as pd
# from sklearn import preprocessing

# x = df.values #returns a numpy array
# min_max_scaler = preprocessing.MinMaxScaler()
# x_scaled = min_max_scaler.fit_transform(x)
# df = pd.DataFrame(x_scaled)
```

[404]:
```
df
```

[404]:
```
     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
..       …     …                 …                …                  …
339        2       0              44.45             17.3              197.0
340        2       0              46.80             14.3              215.0
341        2       0              50.40             15.7              222.0
342        2       0              45.20             14.8              212.0
343        2       0              49.90             16.1              213.0

     body_mass_g  .  FEMALE  MALE
0         3750.0  0       0     1
1         3800.0  0       1     0
2         3250.0  0       1     0
3         4050.0  0       0     0
4         3450.0  0       1     0
..           … ..     …    …
339       4050.0  0       0     0
340       4850.0  0       1     0
341       5750.0  0       0     1
```

```
342       5200.0  0       1     0
343       5400.0  0       0     1

[344 rows x 9 columns]
```

[405]:
```
from sklearn.model_selection import train_test_split as tts
```

[407]:
```
x = df.drop(['species'], axis=1)
y = df['species']
```

[408]:
```
x_train, x_test, y_train, y_test = tts(x, y, test_size=0.20, random_state=0)
```

[409]:
```
df.shape
```

[409]:
```
(344, 9)
```

[410]:
```
print(len(x_train))
print(len(x_test))
```

```
275
69
```

[411]:
```
print(len(y_train))
print(len(y_test))
```

```
275
69
```

[412]:
```
from sklearn.linear_model import LinearRegression
lr = LinearRegression()
```

[413]:
```
lr.fit(x_train, y_train)
```

[413]:
```
LinearRegression()
```

[414]:
```
lr.predict(x_test)
```

[414]:
```
array([ 0.22802448,  0.06538844,  0.19584304,  2.04855423, -0.08043875,
        1.85899576,  0.18207451,  1.2545099 ,  1.80143681,  0.1693787 ,
        0.33385215,  1.78313978,  0.07297072,  1.82155101,  1.87306968,
        0.27934894,  0.15919447,  1.76114572,  0.40536092, -0.0305428 ,
        1.94581613,  0.08661414,  0.91635598,  0.07394465,  1.98612676,
        2.20954868,  1.61113027,  0.07893267,  0.26874312,  1.61858796,
        1.88267763,  0.79962543,  0.00736498,  0.21136476, -0.14059915,
        0.99889605,  1.98479479,  0.85681744, -0.02795315,  0.42392954,
       -0.12092287,  0.37141319,  0.00847979,  1.89781321,  1.7987397 ,
       -0.07755989,  2.03356339,  1.22947219,  0.06138492,  2.03197749,
```

```
       -0.03412648,  0.14749097,  0.08891242,  0.4377562 ,  0.00352624,
        0.72234958,  0.10348793,  2.1792883 ,  0.89866743,  1.9883041 ,
        1.00425041,  1.05883933,  0.51236035,  1.63650671,  1.68163681,
        1.98817262, -0.1768879 ,  1.79384402,  1.68641973])
```

[415]:
```
y_test
```

[415]:
```
141    0
6      0
60     0
249    2
54     0
      ..
229    2
298    2
21     0
246    2
291    2
Name: species, Length: 69, dtype: int64
```

[416]:
```
lr.score(x_test, y_test)
```

[416]:
```
0.9501720710121413
```