lab-session-6-1

September 8, 2024

0.1 Loading the penguins dataset

[5]:
```
import seaborn as sns
import pandas as pd
```

[7]:
```
#Loading the dataset
penguins=sns.load_dataset('penguins')
```

[11]:
```
penguins.head(5)
```

[11]:
```
  species     island  bill_length_mm  bill_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
```

0.2 Performing descriptive statistics

0.2.1 Basic statistics

[12]:
```
#mean
penguins.mean(numeric_only=True)
```

[12]:
```
bill_length_mm         43.921930
bill_depth_mm          17.151170
flipper_length_mm     200.915205
body_mass_g          4201.754386
dtype: float64
```

[13]:
```
#median
penguins.median(numeric_only=True)
```

[13]:
```
bill_length_mm         44.45
bill_depth_mm          17.30
flipper_length_mm     197.00
body_mass_g          4050.00
dtype: float64
```

[16]:
```
#mode
penguins.mode(numeric_only=True).iloc[0]
```

[16]:
```
bill_length_mm         41.1
bill_depth_mm          17.0
flipper_length_mm     190.0
body_mass_g          3800.0
Name: 0, dtype: float64
```

[17]:
```
#standard_deviation
penguins.std(numeric_only=True)
```

[17]:
```
bill_length_mm         5.459584
bill_depth_mm          1.974793
flipper_length_mm     14.061714
body_mass_g          801.954536
dtype: float64
```

[19]:
```
#Variance
penguins.var(numeric_only=True)
```

[19]:
```
bill_length_mm           29.807054
bill_depth_mm             3.899808
flipper_length_mm       197.731792
body_mass_g          643131.077327
dtype: float64
```

0.2.2 Additional descriptive statistics

[21]:
```
#Range
penguins.max(numeric_only=True)-penguins.min(numeric_only=True)
```

[21]:
```
bill_length_mm         27.5
bill_depth_mm           8.4
flipper_length_mm      59.0
body_mass_g          3600.0
dtype: float64
```

[22]:
```
#Skewness
penguins.skew(numeric_only=True)
```

[22]:
```
bill_length_mm       0.053118
bill_depth_mm       -0.143465
flipper_length_mm    0.345682
body_mass_g          0.470329
dtype: float64
```

[23]:
```
#Kurtosis
penguins.kurt(numeric_only=True)
```

[23]:
```
bill_length_mm      -0.876027
bill_depth_mm       -0.906866
flipper_length_mm   -0.984273
body_mass_g         -0.719222
dtype: float64
```

0.3 Performing inferential statistics

[24]:
```
from scipy import stats
```

[25]:
```
# Example data: flipper_length_mm values
#Drop missing values
flipper_length_values = penguins['flipper_length_mm'].dropna()
```

[29]:
```
# Hypothetical population mean
population_mean = 200
```

[30]:
```
# Perform one-sample t-test
t_stat, p_value = stats.ttest_1samp(flipper_length_values, population_mean)
```

[31]:
```
t_stat
```

[31]:
```
1.2036300831163829
```

[32]:
```
p_value
```

[32]:
```
0.22956747651054768
```

0.4 Confidence intervals

[33]:
```
import numpy as np
from scipy import stats
```

[34]:
```
# Sample mean and standard error
sample_mean = np.mean(flipper_length_values)
standard_error = stats.sem(flipper_length_values)
```

[35]:
```
# Compute 95% confidence interval
confidence_interval = stats.norm.interval(0.95, loc=sample_mean,␣
 ↪scale=standard_error)
```

[36]:
```
confidence_interval
```

[36]:
```
(199.42490609474055, 202.40550326198462)
```

0.5 Regression analysis

[37]:
```
import statsmodels.api as sm
```

[45]:
```
# Drop rows with missing values for the selected columns
penguins_clean = penguins[['flipper_length_mm', 'body_mass_g']].dropna()
```

[38]:
```
# Define independent variable
X = sm.add_constant(penguins['flipper_length_mm'].dropna())
```

[39]:
```
# Define dependent variable
y = penguins['body_mass_g'].dropna()
```

[40]:
```
#To match the indices of X and y
X, y = X.loc[y.index], y
```

[41]:
```
# Fitting linear regression model
model = sm.OLS(y, X).fit()
```

[42]:
```
# Print model summary
print(model.summary())
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:            body_mass_g   R-squared:                       0.759
Model:                            OLS   Adj. R-squared:                  0.758
Method:                 Least Squares   F-statistic:                     1071.
Date:                Sun, 08 Sep 2024   Prob (F-statistic):          4.37e-107
Time:                        17:37:12   Log-Likelihood:                -2528.4
No. Observations:                 342   AIC:                             5061.
Df Residuals:                     340   BIC:                             5069.
Df Model:                           1
Covariance Type:            nonrobust
==============================================================================
=====
```

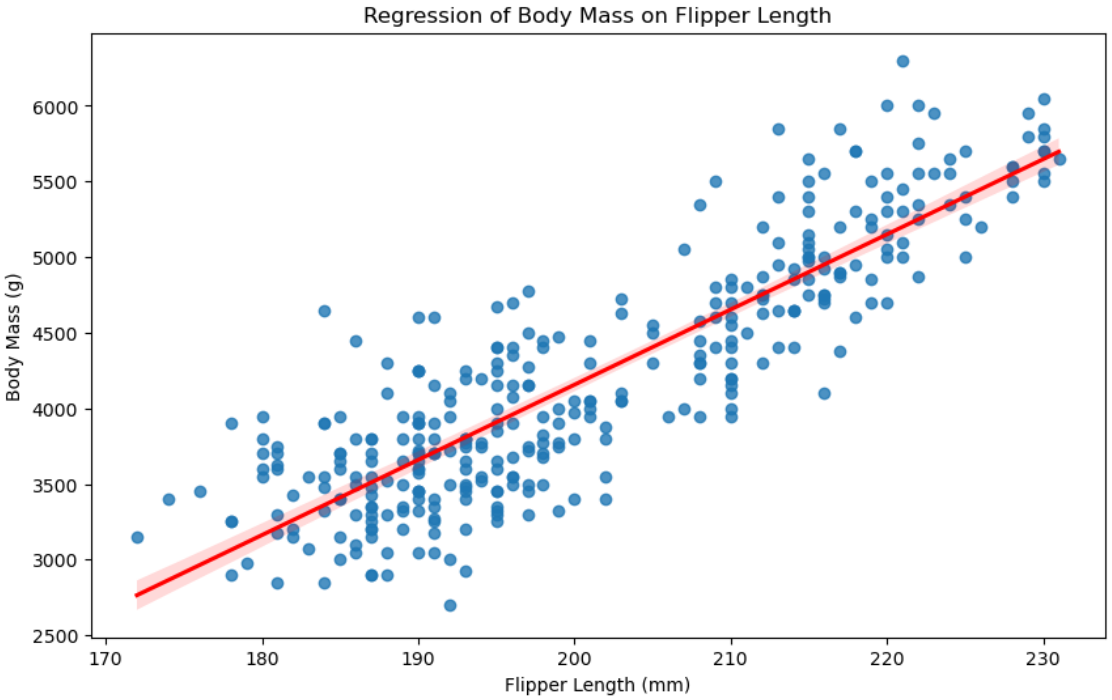
```
                        coef    std err          t      P>|t|      [0.025
0.975]
------------------------------------------------------------------------------
-----
const             -5780.8314    305.815    -18.903      0.000   -6382.358
-5179.305
flipper_length_mm    49.6856      1.518     32.722      0.000      46.699
52.672
==============================================================================
Omnibus:                        5.634   Durbin-Watson:                   2.176
Prob(Omnibus):                  0.060   Jarque-Bera (JB):                5.585
Skew:                           0.313   Prob(JB):                       0.0613
Kurtosis:                       3.019   Cond. No.                     2.89e+03
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly
specified.
[2] The condition number is large, 2.89e+03. This might indicate that there are
strong multicollinearity or other numerical problems.
```

0.6 Creating visualisation

[43]:
```
import matplotlib.pyplot as plt
import seaborn as sns
```

[49]:
```
# Create a scatter plot with regression line
plt.figure(figsize=(10, 6))
sns.regplot(x='flipper_length_mm', y='body_mass_g', data=penguins_clean,␣
 ↪line_kws={"color": "red"})
plt.title('Regression of Body Mass on Flipper Length')
plt.xlabel('Flipper Length (mm)')
plt.ylabel('Body Mass (g)')
plt.show()
```


Regression of Body Mass on Flipper Length
Body Mass (g)
Flipper Length (mm)
2500
3000
3500
4000
4500
5000
5500
6000
170
180
190
200
210
220
230