MTH 522: Advanced Mathematical Statistics

Calculating the Standard Error of Crab Molt Linear Model Coefficients

04/02/2023

Issues

This report will discuss the creation of a linear model that is designed to predict the pre-molt size of a crab based on its post-molt size. A linear model is a mathematical approach that takes input data (i.e., post-molt size) and produces an output (i.e. predicted pre-molt size). In this case, the model was constructed using two variables: presize (pre-molt size) and postsize (post-molt size), and the relationship between the two was expressed as presize ~ postsize.

Once the linear model was created, the first challenge that arose was determining the level of error associated with the coefficients used in the formula that generates the linear model. These coefficients are numerical values used in the formula that constructs the linear model, and the accuracy of the model is heavily dependent on their values.

Findings

After the evaluation of the data and the construction of the linear model, we proceeded to employ a method known as "Bootstrap". By utilizing this technique to generate "new data" from the original dataset through random sampling with replacement, we were able to compute the standard error for the model coefficients - $\beta 0$ and $\beta 1$. Upon analysis, we determined that the standard error for $\beta 0$ was approximately 0.000188, while the standard error for $\beta 1$ was calculated to be 0.275, specifically for this model.

Appendix A: Methods

The data has two main variable which is post molt and pre molt. To evaluate the variability of parameters, a resampling techniques known as bootstrap methods generate several samples from the dataset. The validation set is created by randomly dividing the data into two sets. The training set is used to estimate the coefficients and errors are calculated by iteratively estimating the coefficients and computing the standard deviation of the estimations for each split.

Appendix B: Results

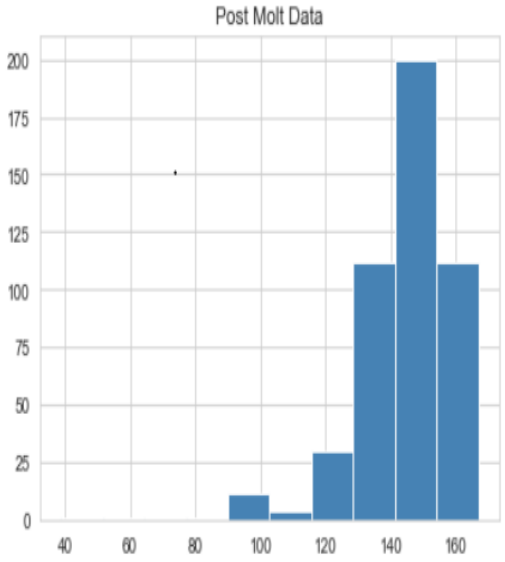
Summary statistics:

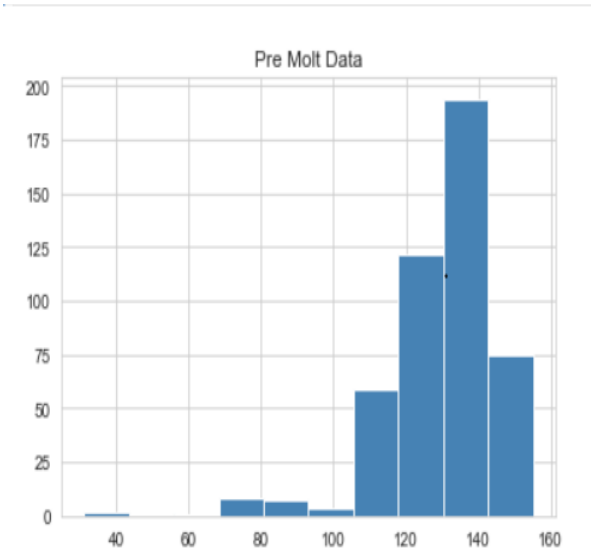
Postsize:

```
count    472.000000
mean     143.897669
std       14.640602
min       38.800000
25%      137.975000
50%      147.400000
75%      153.425000
max      166.800000
Name: postsize, dtype: float64
Kurtosis: 10.116042372071325
Skewness: -2.3469021583966594
```

Presize:

```
count    472.000000
mean     129.211864
std       15.864520
min       31.100000
25%      121.675000
50%      132.800000
75%      140.000000
max      155.100000
Name: presize, dtype: float64
Kurtosis: 6.766321650922
Skewness: -2.0034871763549766
```





Appendix C: Code

```
import numpy as np
import pandas as pd
import random
import matplotlib.pyplot as plt
from scipy.stats import kurtosis, skew
from sklearn.linear_model import LinearRegression
import scipy.stats as stats
from sklearn.utils import resample
```

```
crabMoltData = pd.read_excel("crab_molt.xlsx")
# Removing the na rows in the data
crabMoltData = crabMoltData.dropna()

def getCoefficients(data, indices):
    data_sample = data.iloc[indices]
    X = data_sample[['postsize']]
    y = data_sample['presize']
    model = LinearRegression().fit(X, y)
    return model.coef_[0], model.intercept_


print(crabMoltData['postsize'].describe())
print("Kurtosis:", kurtosis(crabMoltData['postsize']))
print("Skewness:", skew(crabMoltData['postsize']))

print(crabMoltData['presize'].describe())
print("Kurtosis:", kurtosis(crabMoltData['presize']))
print("Skewness:", skew(crabMoltData['presize']))

crabMoltData['postsize'].hist(color="steelblue")
plt.title("Post Molt Data")
plt.show()

crabMoltData['presize'].hist(color="steelblue")
plt.title("Pre Molt Data")
plt.show()

n_replicates = 10000
bootstrapped_results = []
for i in range(n_replicates):
    bootstrapped_sample = resample(crabMoltData, n_samples=len(crabMoltData))
    coefficients = getCoefficients(bootstrapped_sample,
np.arange(len(bootstrapped_sample)))
    bootstrapped_results.append(coefficients)

boot_summary = pd.DataFrame(bootstrapped_results, columns=['beta0', 'beta1'])
print('Summary:\n',boot_summary.describe())
print('Error:\n',boot_summary.sem(axis=0))
```