Week 3 - Homework

STAT 420, Summer 2019, Connor Segneri - segneri3

Exercise 1 (Using `lm` for Inference)

For this exercise we will use the `cats` dataset from the `MASS` package. You should use `?cats` to learn about the background of this dataset.

(a) Fit the following simple linear regression model in `R`. Use heart weight as the response and body weight as the predictor.

$$Y_i = \beta_0 + \beta_1 x_i + \epsilon_i$$

Store the results in a variable called `cat_model`. Use a t test to test the significance of the regression. Report the following:

- The null and alternative hypotheses
- The value of the test statistic
- The p-value of the test
- A statistical decision at $\alpha = 0.05$
- A conclusion in the context of the problem

When reporting these, you should explicitly state them in your document, not assume that a reader will find and interpret them from a large block of `R` output.

```r
library(MASS)

cat_model = lm(Hwt ~ Bwt, data = cats)

t.test(cats$Bwt, cats$Hwt)
```

```
##
##  Welch Two Sample t-test
##
## data:  cats$Bwt and cats$Hwt
## t = -38.22, df = 154.35, p-value < 2.2e-16
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -8.315622 -7.498267
## sample estimates:
## mean of x mean of y
##  2.723611 10.630556
```

```r
summary(cat_model)
```

```
##
## Call:
## lm(formula = Hwt ~ Bwt, data = cats)
```

```
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.5694 -0.9634 -0.0921  1.0426  5.1238
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -0.3567     0.6923  -0.515    0.607
## Bwt           4.0341     0.2503  16.119   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.452 on 142 degrees of freedom
## Multiple R-squared:  0.6466, Adjusted R-squared:  0.6441
## F-statistic: 259.8 on 1 and 142 DF,  p-value: < 2.2e-16
```

The null hypothesis is that there is no correlation, or that beta_1 is zero. The alternative hypothesis is that there is a correlation, or that beta_1 is **not** zero.

The value of the test statistic is `-38.22`.

The p-value is `< 2.2e-16`.

A statistical decision at $\alpha = 0.05$ is that we reject the null hypothesis in favor of the alternative hypothesis. The p-value is low enough to indicate a relationship between the variables.

A conclusion in the context of the problem would be that there seems to be a correlation between the the body weight of a cat and its heart weight.

(b) Calculate a 90% confidence interval for β_1. Give an interpretation of the interval in the context of the problem.

```r
confint(cat_model, level = .90)["Bwt",]
```

```
##      5 %     95 %
## 3.619716 4.448409
```

Based on the data, the model is 90% sure that β_1, the relationship between the heart and body weight of a cat, is within the two values shown above.

(c) Calculate a 99% confidence interval for β_0. Give an interpretation of the interval in the context of the problem.

```r
confint(cat_model, level = .99)["(Intercept)",]
```

```
##     0.5 %    99.5 %
## -2.164125  1.450800
```

Based on the data, the model is 99% sure that β_0, the heart weight of a cat when the body weight is zero, is within the two values shown above.

(d) Use a 99% confidence interval to estimate the mean heart weight for body weights of 2.1 and 2.8 kilograms. Which of the two intervals is wider? Why?

```r
new.bwts = data.frame(Bwt = c(2.1, 2.8))

new.pred = data.frame(predict(cat_model, newdata = new.bwts, interval = "confidence"))

new.width = data.frame(width = c(
  new.pred$upr[1] - new.pred$lwr[1],
  new.pred$upr[2] - new.pred$lwr[2]
))

cbind(new.bwts, new.pred, new.width)
```

```
##   Bwt       fit       lwr       upr     width
## 1 2.1  8.114869  7.724455  8.505284 0.7808286
## 2 2.8 10.938713 10.696491 11.180935 0.4844437
```

The first interval (2.1 kg) is wider. The model is trying to predict, within 99% certaintiy, what the heart weight is for cats that have a body weight of 2.1 and 2.8 kg. Based on the data, it's not as certain for 2.1 kg as it is for 2.8 kg.

(e) Use a 99% prediction interval to predict the heart weight for body weights of 2.8 and 4.2 kilograms.

```r
new.bwts = data.frame(Bwt = c(2.8, 4.2))

new.pred = data.frame(predict(cat_model, newdata = new.bwts, interval = "prediction"))

cbind(new.bwts, new.pred)
```

```
##   Bwt      fit       lwr      upr
## 1 2.8 10.93871  8.057446 13.81998
## 2 4.2 16.58640 13.614238 19.55856
```

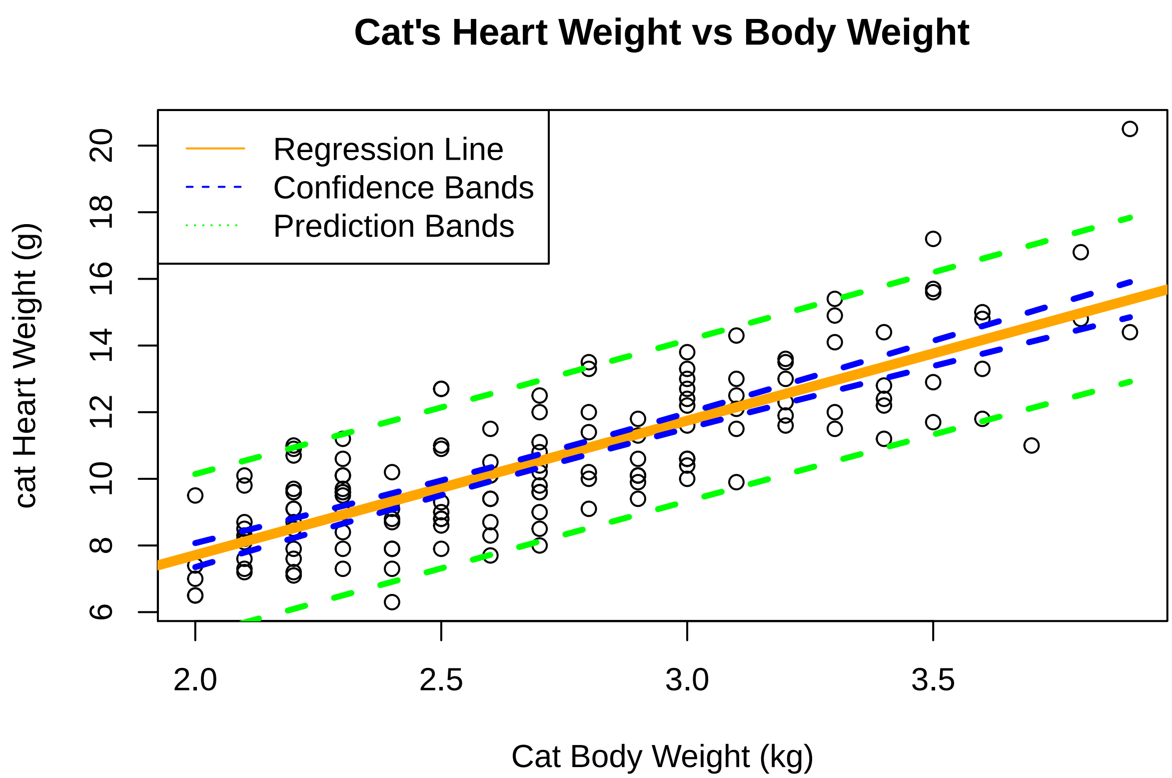
(f) Create a scatterplot of the data. Add the regression line, 90% confidence bands, and 90% prediction bands.

```r
bwt.grid = seq(min(cats$Bwt), max(cats$Bwt), by = 0.01)

bwt.ci.band = predict(cat_model, newdata = data.frame(Bwt = bwt.grid),
                      interval = "confidence", level = .90)

bwt.pi.band = predict(cat_model, newdata = data.frame(Bwt = bwt.grid),
                      interval = "prediction", level = .90)

plot(Hwt ~ Bwt, data = cats,
     xlab = "Cat Body Weight (kg)",
     ylab = "cat Heart Weight (g)",
     main = "Cat's Heart Weight vs Body Weight")
abline(cat_model, lwd = 5, col = 'orange')
lines(bwt.grid, bwt.ci.band[,"lwr"], col = "blue", lwd = 3, lty = 2)
lines(bwt.grid, bwt.ci.band[,"upr"], col = "blue", lwd = 3, lty = 2)
lines(bwt.grid, bwt.pi.band[,"lwr"], col = "green", lwd = 3, lty = 2)
lines(bwt.grid, bwt.pi.band[,"upr"], col = "green", lwd = 3, lty = 2)
legend("topleft", legend = c("Regression Line", "Confidence Bands", "Prediction Bands"),
       col = c("orange", "blue", "green"), lty = 1:3)
```



(g) Use a t test to test:

- $H_0 : \beta_1 = 4$
- $H_1 : \beta_1 \neq 4$

Report the following:

- The value of the test statistic
- The p-value of the test
- A statistical decision at $\alpha = 0.05$

When reporting these, you should explicitly state them in your document, not assume that a reader will find and interpret them from a large block of `R` output.

```
t.test(x = cats$Bwt, y = cats$Hwt,
       mu = 4)
```

```
##
##  Welch Two Sample t-test
##
## data:  cats$Bwt and cats$Hwt
## t = -57.555, df = 154.35, p-value < 2.2e-16
## alternative hypothesis: true difference in means is not equal to 4
## 95 percent confidence interval:
```

```
##  -8.315622 -7.498267
## sample estimates:
## mean of x mean of y
##  2.723611 10.630556
```

The value of the test statistic is `-57.555`. The p-value of the test is `2.2e-16`. A statistical decision at $\alpha = 0.05$ is that the p-value is below `0.05`, so we reject the null hypothesis. The true difference in means is not equal to 4.

Exercise 2 (More `lm` for Inference)

For this exercise we will use the `Ozone` dataset from the `mlbench` package. You should use `?Ozone` to learn about the background of this dataset. You may need to install the `mlbench` package. If you do so, do not include code to install the package in your `R` Markdown document.

For simplicity, we will re-perform the data cleaning done in the previous homework.

```r
data(Ozone, package = "mlbench")
Ozone = Ozone[, c(4, 6, 7, 8)]
colnames(Ozone) = c("ozone", "wind", "humidity", "temp")
Ozone = Ozone[complete.cases(Ozone), ]
```

(a) Fit the following simple linear regression model in `R`. Use the ozone measurement as the response and wind speed as the predictor.

$$Y_i = \beta_0 + \beta_1 x_i + \epsilon_i$$

Store the results in a variable called `ozone_wind_model`. Use a t test to test the significance of the regression. Report the following:

- The null and alternative hypotheses
- The value of the test statistic
- The p-value of the test
- A statistical decision at $\alpha = 0.01$
- A conclusion in the context of the problem

When reporting these, you should explicitly state them in your document, not assume that a reader will find and interpret them from a large block of `R` output.

```r
ozone_wind_model = lm(ozone ~ wind, data = Ozone)

summary(ozone_wind_model)
```

```
##
## Call:
## lm(formula = ozone ~ wind, data = Ozone)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
```

```
## -10.730  -6.652  -1.752   4.687  26.359
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  11.8636     1.0856  10.928   <2e-16 ***
## wind         -0.0445     0.2032  -0.219    0.827
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 7.985 on 342 degrees of freedom
## Multiple R-squared:  0.0001402,  Adjusted R-squared:  -0.002783
## F-statistic: 0.04795 on 1 and 342 DF,  p-value: 0.8268
```

The null hypothesis is that beta_1 is zero, or that there is no correlation between wind speed and the ozone measurement. The alternate hypothesis is that beta_1 is not zero, or that there is a correlation.

The test statistic is `-0.219`.

The p-value is `0.8268`.

A statistical decision at $\alpha = 0.01$, is that the p-value is greater than 0.01, so we fail to reject the null hypothesis.

A conclusion in the context of the problem is that the model fails to establish a correlation between wind speed and the ozone measurement.

(b) Fit the following simple linear regression model in `R`. Use the ozone measurement as the response and temperature as the predictor.

$$Y_i = \beta_0 + \beta_1 x_i + \epsilon_i$$

Store the results in a variable called `ozone_temp_model`. Use a t test to test the significance of the regression. Report the following:

- The null and alternative hypotheses
- The value of the test statistic
- The p-value of the test
- A statistical decision at $\alpha = 0.01$
- A conclusion in the context of the problem

When reporting these, you should explicitly state them in your document, not assume that a reader will find and interpret them from a large block of `R` output.

```r
ozone_temp_model = lm(ozone ~ temp, data = Ozone)

summary(ozone_temp_model)
```

```
##
## Call:
## lm(formula = ozone ~ temp, data = Ozone)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -11.7630  -3.7495  -0.1849   3.1099  15.1118
##
```

```
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)    
## (Intercept) -14.88480    1.19229  -12.48   <2e-16 ***
## temp          0.42968    0.01881   22.85   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## 
## Residual standard error: 5.024 on 342 degrees of freedom
## Multiple R-squared:  0.6042, Adjusted R-squared:  0.603 
## F-statistic: 522.1 on 1 and 342 DF,  p-value: < 2.2e-16
```

The null hypothesis is that beta_1 is zero, or that there is no correlation between temperature and the ozone measurement. The alternate hypothesis is that beta_1 is not zero, or that there is a correlation.

The test statistic is `22.85`.

The p-value is `< 2.2e-16`.

A statistical decision at $\alpha = 0.01$, is that the p-value is less than 0.01, so we reject the null hypothesis.

A conclusion in the context of the problem is that there is a correlation between the temperature and the ozone measurement.

Exercise 3 (Simulating Sampling Distributions)

For this exercise we will simulate data from the following model:

$$Y_i = \beta_0 + \beta_1 x_i + \epsilon_i$$

Where $\epsilon_i \sim N(0, \sigma^2)$. Also, the parameters are known to be:

- $\beta_0 = -5$
- $\beta_1 = 3.25$
- $\sigma^2 = 16$

We will use samples of size $n = 50$.

(a) Simulate this model 2000 times. Each time use `lm()` to fit a simple linear regression model, then store the value of $\hat{\beta}_0$ and $\hat{\beta}_1$. Set a seed using **your** birthday before performing the simulation. Note, we are simulating the x values once, and then they remain fixed for the remainder of the exercise.

```r
birthday = 19951015
set.seed(birthday)
n = 50
x = seq(0, 10, length = n)

sxx = sum((x - mean(x)) ^ 2)

beta_0 = -5
beta_1 = 3.25
sigma = 4
```

```r
var_beta_1_hat = sigma ^ 2 / sxx
var_beta_0_hat = sigma ^ 2 * (1 / n + mean(x) ^ 2 / sxx)

num_samples = 2000
beta_0_hats = rep(0, num_samples)
beta_1_hats = rep(0, num_samples)

for (i in 1:num_samples){
  eps = rnorm(n, mean = 0, sd = sigma)
  y = beta_0 + beta_1*x + eps

  sim_model = lm(y ~ x)

  beta_0_hats[i] = coef(sim_model)[1]
  beta_1_hats[i] = coef(sim_model)[2]
}
```

(b) Create a table that summarizes the results of the simulations. The table should have two columns, one for $\hat{\beta}_0$ and one for $\hat{\beta}_1$. The table should have four rows:

- A row for the true expected value given the known values of x
- A row for the mean of the simulated values
- A row for the true standard deviation given the known values of x
- A row for the standard deviation of the simulated values

```r
library(knitr)
library(kableExtra)

sum.table = data.frame(
  beta_0 = c(
    beta_0,
    mean(beta_0_hats),
    sigma,
    sd(beta_0_hats)
  ),
  beta_1 = c(
    beta_1,
    mean(beta_1_hats),
    sigma,
    sd(beta_1_hats)
  )
)

sum.table %>% kable() %>% kable_styling()
```

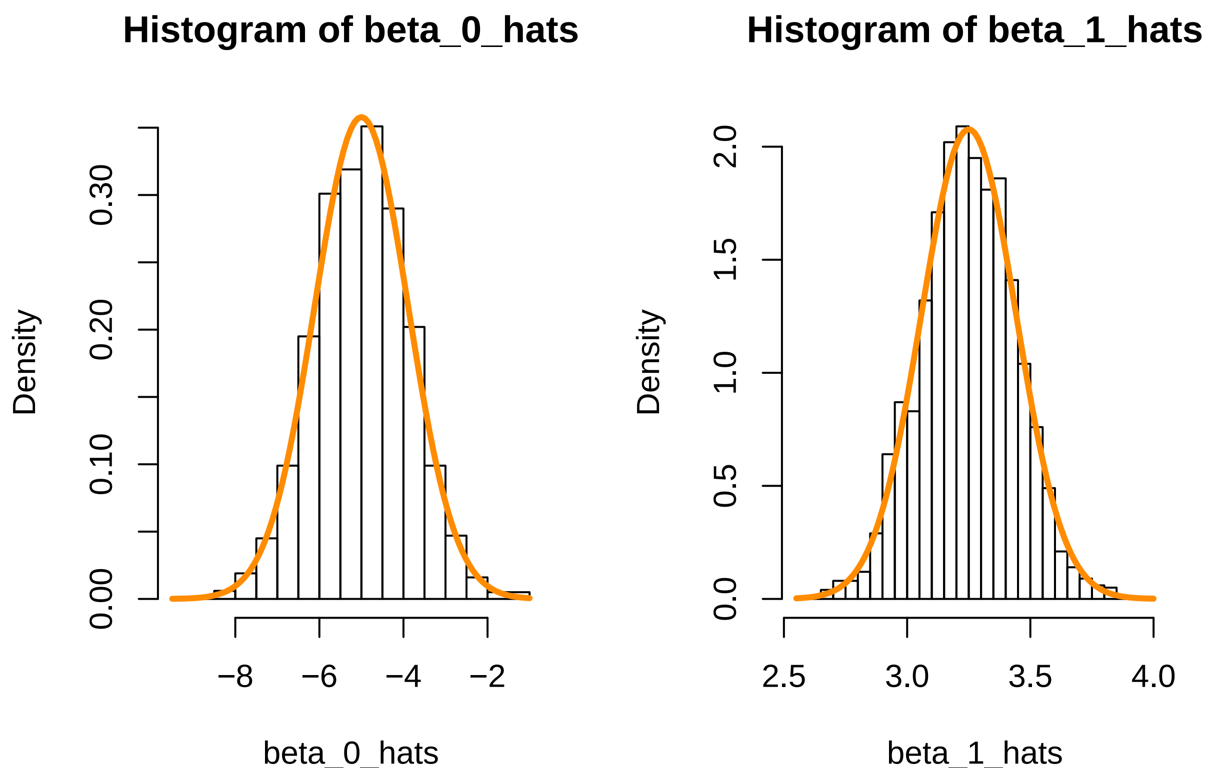
beta_0	beta_1
-5.000000	3.2500000
-4.996491	3.2487071
4.000000	4.0000000
1.119096	0.1920827

(c) Plot two histograms side-by-side:

- A histogram of your simulated values for $\hat{\beta}_0$. Add the normal curve for the true sampling distribution of $\hat{\beta}_0$.
- A histogram of your simulated values for $\hat{\beta}_1$. Add the normal curve for the true sampling distribution of $\hat{\beta}_1$.

```
par(mfrow=c(1,2))

hist(beta_0_hats, prob = TRUE, breaks = 20)
curve(dnorm(x, mean = beta_0, sd = sqrt(var_beta_0_hat)), col = 'darkorange', add = TRUE, lwd = 3)
hist(beta_1_hats, prob = TRUE, breaks = 20)
curve(dnorm(x, mean = beta_1, sd = sqrt(var_beta_1_hat)), col = 'darkorange', add = TRUE, lwd = 3)
```



Exercise 4 (Simulating Confidence Intervals)

For this exercise we will simulate data from the following model:

$$Y_i = \beta_0 + \beta_1 x_i + \epsilon_i$$

Where $\epsilon_i \sim N(0, \sigma^2)$. Also, the parameters are known to be:

- $\beta_0 = 5$

- $\beta_1 = 2$
- $\sigma^2 = 9$

We will use samples of size $n = 25$.

Our goal here is to use simulation to verify that the confidence intervals really do have their stated confidence level. Do **not** use the `confint()` function for this entire exercise.

(a) Simulate this model 2500 times. Each time use `lm()` to fit a simple linear regression model, then store the value of $\hat{\beta}_1$ and s_e. Set a seed using **your** birthday before performing the simulation. Note, we are simulating the x values once, and then they remain fixed for the remainder of the exercise.

```
birthday = 19951015
set.seed(birthday)
n = 25
x = seq(0, 2.5, length = n)

sxx = sum((x - mean(x)) ^ 2)

beta_0 = 5
beta_1 = 2
sigma = 3

var_beta_1_hat = sigma ^ 2 / sxx

num_samples = 2500
beta_1_hats = rep(0, num_samples)
s_e = rep(0, num_samples)

for (i in 1:num_samples){
  eps = rnorm(n, mean = 0, sd = sigma)
  y = beta_0 + beta_1*x + eps

  sim_model = lm(y ~ x)

  beta_1_hats[i] = coef(sim_model)[2]
  s_e[i] = summary(sim_model)$coefficients[2,2]
}
```

(b) For each of the $\hat{\beta}_1$ that you simulated, calculate a 95% confidence interval. Store the lower limits in a vector `lower_95` and the upper limits in a vector `upper_95`. Some hints:

- You will need to use `qt()` to calculate the critical value, which will be the same for each interval.
- Remember that `x` is fixed, so S_{xx} will be the same for each interval.
- You could, but do not need to write a `for` loop. Remember vectorized operations.

```
crit = qt(1 - (1 - 0.95) / 2, df = n - 2)

lower_95 = beta_1_hats - (crit * s_e)
upper_95 = beta_1_hats + (crit * s_e)
```

(c) What proportion of these intervals contains the true value of β_1?

```r
in_interval = lower_95 <= beta_1 & upper_95 >= beta_1
prop = length(in_interval[in_interval]) / length(in_interval)
prop
```

```
## [1] 0.9548
```

`95.48%` of intervals contain the true value of β_1.

(d) Based on these intervals, what proportion of the simulations would reject the test $H_0 : \beta_1 = 0$ vs $H_1 : \beta_1 \neq 0$ at $\alpha = 0.05$?

```r
in_interval = lower_95 > 0

prop = length(in_interval[in_interval]) / length(in_interval)
prop
```

```
## [1] 0.6768
```

`67.68%` of these simulations would reject the null hypothesis.

(e) For each of the $\hat{\beta}_1$ that you simulated, calculate a 99% confidence interval. Store the lower limits in a vector `lower_99` and the upper limits in a vector `upper_99`.

```r
crit = qt(1 - (1 - 0.99) / 2, df = n - 2)

lower_99 = beta_1_hats - (crit * s_e)
upper_99 = beta_1_hats + (crit * s_e)
```

(f) What proportion of these intervals contains the true value of β_1?

```r
in_interval = lower_99 <= beta_1 & upper_99 >= beta_1
prop = length(in_interval[in_interval]) / length(in_interval)
prop
```

```
## [1] 0.9888
```

`98.88%` of intervals contain the true value of β_1.

(g) Based on these intervals, what proportion of the simulations would reject the test $H_0 : \beta_1 = 0$ vs $H_1 : \beta_1 \neq 0$ at $\alpha = 0.01$?

```r
in_interval = lower_99 > 0

prop = length(in_interval[in_interval]) / length(in_interval)
prop
```

```
## [1] 0.3992
```

`39.92%` of these simulations would reject the nully hypothesis.

Exercise 5 (Prediction Intervals "without" `predict`)

Write a function named `calc_pred_int` that performs calculates prediction intervals:

$$\hat{y}(x) \pm t_{\alpha/2,n-2} \cdot s_e \sqrt{1 + \frac{1}{n} + \frac{(x - \bar{x})^2}{S_{xx}}}.$$

for the linear model

$$Y_i = \beta_0 + \beta_1 x_i + \epsilon_i.$$

(a) Write this function. You may use the `predict()` function, but you may **not** supply a value for the `level` argument of `predict()`. (You can certainly use `predict()` any way you would like in order to check your work.)

The function should take three inputs:

- `model`, a model object that is the result of fitting the SLR model with `lm()`
- `newdata`, a data frame with a single observation (row)
  - This data frame will need to have a variable (column) with the same name as the data used to fit `model`.
- `level`, the level (0.90, 0.95, etc) for the interval with a default value of `0.95`

The function should return a named vector with three elements:

- `estimate`, the midpoint of the interval
- `lower`, the lower bound of the interval
- `upper`, the upper bound of the interval

```
calc_pred_int = function(model, newdata, level = 0.95){

  n = nrow(model.frame(model))
  x_bar = mean(model.frame(model)$Bwt)
  est = newdata$Bwt
  crit = qt(1 - (1 - level) / 2, df = n - 2)
  se = sqrt(1 + 1/n + ((est - x_bar)) ^ 2)

  lower = est - crit * se
  upper = est + crit * se

  prediction = c(
    estimate = est,
    lower = lower,
    upper = upper
  )
  prediction
}
```

(b) After writing the function, run this code:

```r
newcat_1 = data.frame(Bwt = 4.0)
calc_pred_int(cat_model, newcat_1)
```

```
##  estimate     lower     upper
## 4.0000000 0.7904277 7.2095723
```

(c) After writing the function, run this code:

```r
newcat_2 = data.frame(Bwt = 3.3)
calc_pred_int(cat_model, newcat_2, level = 0.99)
```

```
##  estimate     lower     upper
## 3.3000000 0.2786079 6.3213921
```