Week 3 - Homework

STAT 420, Summer 2022, Ilya Andreev, iandre3@illinois.edu

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.

```
library(MASS)
y = cats$Hwt
x = cats$Bwt
cat_model = lm(y ~ x)
```

- The null hypothesis: $H_0 : \beta_1 = 0$. The alternative hypothesis: $H_1 : \beta_1 \neq 0$.
- Test statistic value: 16.1193908.
- P-value of the test: $6.9690446 \times 10^{-34}$.
- At $\alpha = 0.05$, the statistical decision is to reject H_0.
- We conclude that there is enough statistical evidence to reject the null hypothesis in favor of the hypothesis that β_1 is not equal to zero.

(b) Calculate a 95% confidence interval for β_1. Give an interpretation of the interval in the context of the problem.

```
c = confint(cat_model, "x", 0.95)
```

We are 95% confident that the value of β_1 lies between 3.539343 and 4.5287824. Since 0 is not in that range, we can conclude that there is a linear relationship between the predictor and the response.

(c) Calculate a 90% confidence interval for β_0. Give an interpretation of the interval in the context of the problem.

```
c = confint(cat_model, "(Intercept)", 0.90)
```

Mathematically, we are 90% confident that the value of β_0 lies between -1.5028345 and 0.7895096. But pragmatically, we know that this is not the whole truth, since we know that a zero bodyweight does not result in a negative weight of a heart.

(d) Use a 90% 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?

```
prediction = predict.lm(cat_model, newdata=data.frame(x=c(2.1, 2.8)), interval="confidence", level=0.9)
prediction[,3] - prediction[,2]
```

```
##         1         2
## 0.6539740 0.4057402
```

Evidently the first confidence interval is wider. This is because the second data point is much closer to the mean of x in the data set, which is 2.7236111. We are generally more confident in our predictions around the mean predictor value.

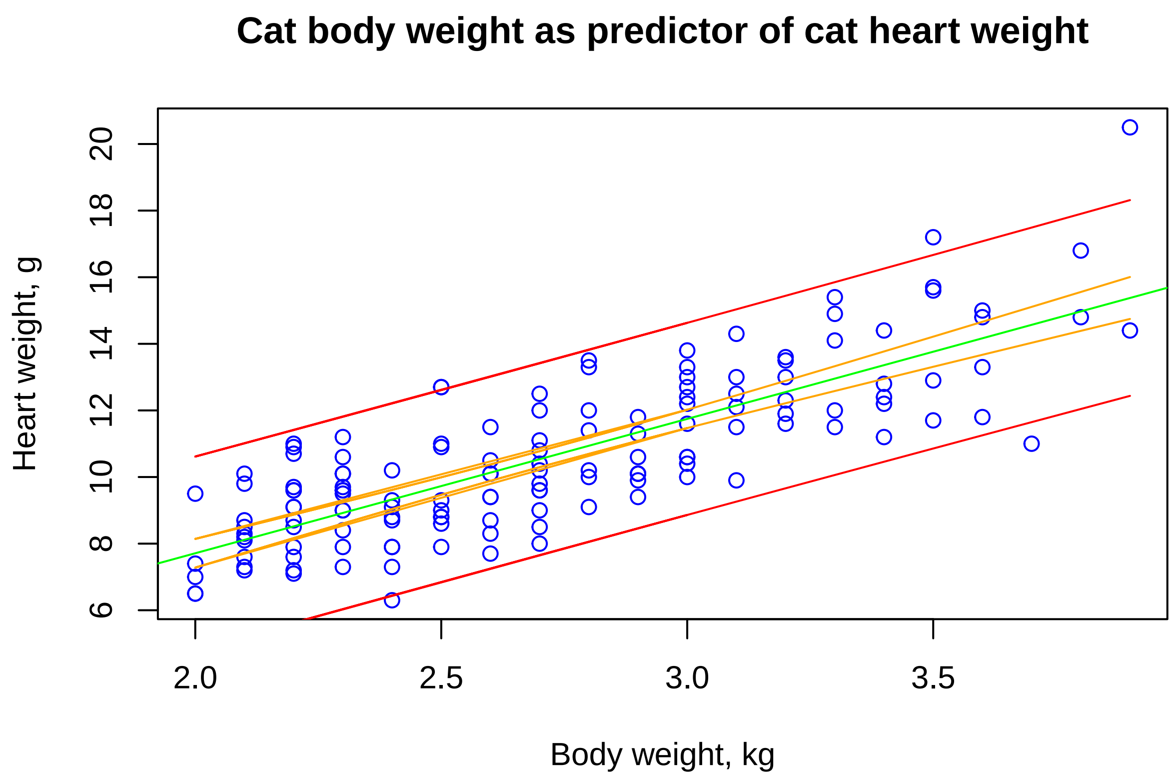
(e) Use a 90% prediction interval to predict the heart weight for body weights of 2.8 and 4.2 kilograms.

```
predict.lm(cat_model, newdata=data.frame(x=c(2.8, 4.2)), interval="prediction", level=0.9)
```

```
##        fit       lwr      upr
## 1 10.93871  8.525541 13.35189
## 2 16.58640 14.097100 19.07570
```

(f) Create a scatterplot of the data. Add the regression line, 95% confidence bands, and 95% prediction bands.

```
plot(x, y, main="Cat body weight as predictor of cat heart weight", xlab="Body weight, kg", ylab="Heart
abline(cat_model, col="green")
confidence_interval = predict(cat_model, interval="confidence", level=0.95)
prediction_interval = suppressWarnings(predict(cat_model, interval="prediction", level=0.95))
lines(x, confidence_interval[,2], col="orange")
lines(x, confidence_interval[,3], col="orange")
lines(x, prediction_interval[,2], col="red")
lines(x, prediction_interval[,3], col="red")
```



(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.

```
s = summary(lm(y ~ x, offset=4*x))
s$coefficients
```

```
##               Estimate Std. Error    t value  Pr(>|t|)
## (Intercept) -0.3566624  0.6922770 -0.5152019 0.6072131
## x            0.0340627  0.2502615  0.1361084 0.8919283
```

The value of the test statistic is 0.1361084. The p-value is 0.8919283. The statistical decision at 0.05 is to fail to reject the null hypothesis.

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.

```
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.

```
x = Ozone$wind
y = Ozone$ozone
ozone_wind_model = lm(y ~ x)
```

- The null hypothesis: $H_0 : \beta_1 = 0$. The alternative hypothesis: $H_1 : \beta_1 \neq 0$.
- Test statistic value: -0.2189811.
- P-value of the test: 0.8267954.
- At $\alpha = 0.01$, the statistical decision is to fail to reject H_0.
- We conclude that there is not enough statistical evidence to claim that β_1 is not equal to 0.

(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.

```
x = Ozone$temp
y = Ozone$ozone
ozone_temp_model = lm(y ~ x)
```

- The null hypothesis: $H_0 : \beta_1 = 0$. The alternative hypothesis: $H_1 : \beta_1 \neq 0$.
- Test statistic value: 22.848962.
- P-value of the test: $8.1537636 \times 10^{-71}$.
- At $\alpha = 0.01$, the statistical decision is to reject H_0.

- We conclude that there is enough statistical evidence to claim that β_1 is not equal to 0.

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.

```
birthday = 24111999
set.seed(birthday)
n = 50
x = seq(0, 10, length = n)
y_func = function(x) {
  -5 + 3.25 * x + rnorm(length(x), 0, 4)
}

b_0s = numeric(2000)
b_1s = numeric(2000)

for (i in 1:2000) {
  y = y_func(x)
  m = lm(y ~ x)
  b_0s[i] = m$coefficients[1]
  b_1s[i] = m$coefficients[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

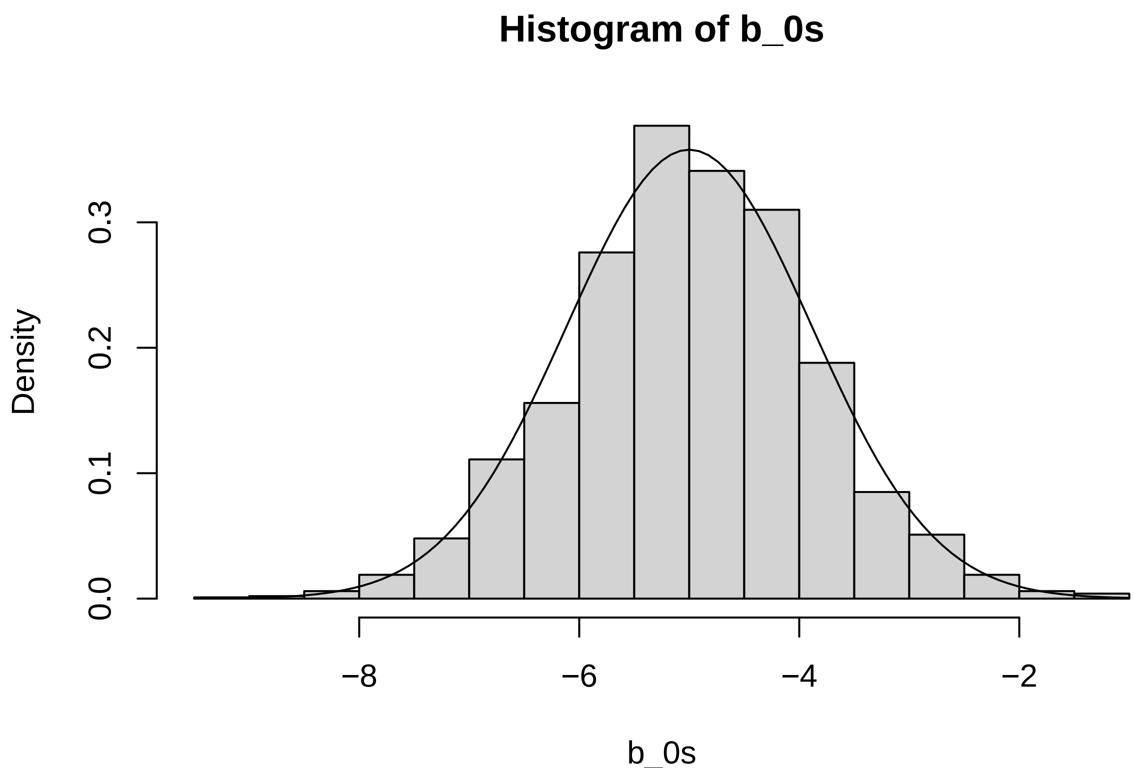
```
sd_b0 = 4 * sqrt(1/length(x) + (mean(x) ** 2) / sum((x - mean(x)) ** 2))
sd_b1 = 4 / sqrt(sum((x - mean(x)) ** 2))
knitr::kable(matrix(c(-5, 3.25, mean(b_0s), mean(b_1s), sd_b0, sd_b1, sd(b_0s), sd(b_1s)), nrow=4, ncol
```

	beta_0	beta_1
True	-5.000000	3.2500000
Simulated mean	-4.984623	3.2455909
True standard deviation	1.114609	0.1920784
Simulated standard deviation	1.114917	0.1936617

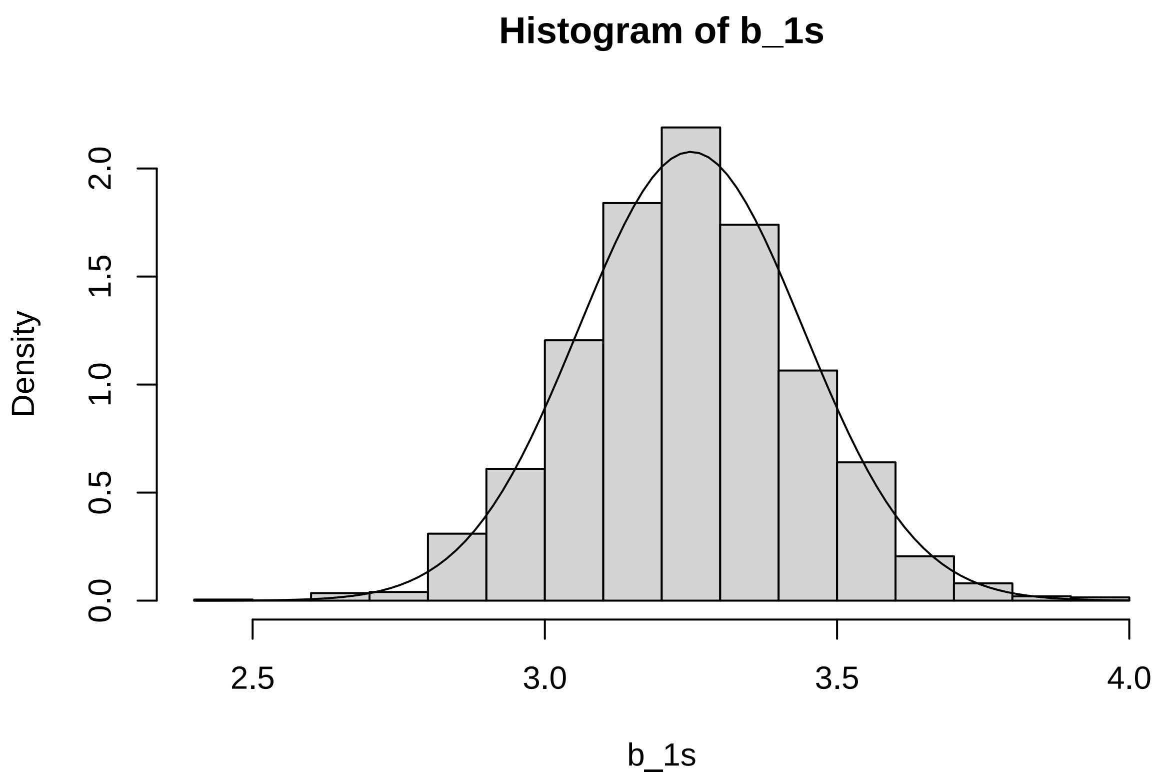
(c) Plot two histograms side-by-side:

- 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 $\hat{\beta}_1$.

```
hist(b_0s, freq=FALSE)
x = seq(0, 1, 0.01)
curve(dnorm(x, -5, sd_b0), add=TRUE)
```




```
hist(b_1s, freq=FALSE)
x = seq(0, 1, 0.01)
curve(dnorm(x, 3.25, sd_b1), add=TRUE)
```



***

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 = 24111999
set.seed(birthday)
n = 25
x = seq(0, 2.5, length = n)

y_func = function(x) {
  5 + 2 * x + rnorm(length(x), 0, 3)
}

b_0s = numeric(2500)
```

```
b_1s = numeric(2500)
se = numeric(2500)

for (i in 1:2500) {
  y = y_func(x)
  m = lm(y ~ x)
  se[i] = summary(m)$sigma
  b_0s[i] = m$coefficients[1]
  b_1s[i] = m$coefficients[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.

```
t = qt(0.975, df=(length(x)-2))
#lower_t = qt(0.025, df=(length(x)-2))
Sxx = sum((x - mean(x)) ** 2)

lower_95 = b_1s - t * se / sqrt(Sxx)
upper_95 = b_1s + t * se / sqrt(Sxx)
```

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

```
above_lower = lower_95 < 2
below_upper = 2 < upper_95

prop = sum(above_lower * below_upper) / length(above_lower)
```

0.9532 is the proportion of confidence 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 $\alpha = 0.05$?

```
prop = (sum(upper_95 < 0) + sum(lower_95 > 0)) / length(upper_95)
```

0.67 is the proportion of simulations that 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`.

```
t = qt(0.995, df=(length(x)-2))
Sxx = sum((x - mean(x)) ** 2)

lower_99 = b_1s - t * se / sqrt(Sxx)
upper_99 = b_1s + t * se / sqrt(Sxx)
```

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

```
above_lower = lower_99 < 2
below_upper = 2 < upper_99

prop = sum(above_lower * below_upper) / length(above_lower)
```

0.9876 is the proportion of confidence 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 $\alpha = 0.01$?

```
prop = (sum(upper_99 < 0) + sum(lower_99 > 0)) / length(upper_99)
```

0.394 is the proportion of simulations that would reject the null 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) {
  x = model$model[,2]
  t = qt(level + (1 - level) / 2, length(x)-2)
  y_hat = predict(model, newdata=newdata)
  s_e = sigma(model)
  m = mean(x)
  S_xx = sum((newdata[,1] - m) ** 2)
  sqroot = sqrt(1 + 1 / length(x) + ((newdata[,1] - m) ** 2) / S_xx)
  lower = y_hat - t * s_e * sqroot
  upper = y_hat + t * s_e * sqroot
  ret = c(y_hat, lower, upper)
  names(ret) = c("estimate", "lower", "upper")

  ret
}
```

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

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

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

```
newcat_2 = data.frame(x = 3.3)
calc_pred_int(cat_model, newcat_2, level = 0.90)
```