Homework 4

Kate Miller

Table of contents

> ❗ Important
>
> Please read the instructions carefully before submitting your assignment.
>
> 1. This assignment requires you to only upload a `PDF` file on Canvas
> 2. Don't collapse any code cells before submitting.
> 3. Remember to make sure all your code output is rendered properly before uploading your submission.
>
> Please add your name to the author information in the frontmatter before submitting your assignment

We will be using the following libraries:

```
packages <- c(
  "dplyr",
  "readr",
  "tidyr",
  "purrr",
  "stringr",
```

```r
    "corrplot",
    "car",
    "caret",
    "torch",
    "nnet",
    "broom"
  )

#renv::install(packages)

sapply(packages, require, character.only=T)
```

```
Loading required package: dplyr

Attaching package: 'dplyr'

The following objects are masked from 'package:stats':

    filter, lag

The following objects are masked from 'package:base':

    intersect, setdiff, setequal, union

Loading required package: readr

Loading required package: tidyr

Loading required package: purrr

Loading required package: stringr

Loading required package: corrplot

corrplot 0.92 loaded

Loading required package: car
```

```
Loading required package: carData

Attaching package: 'car'

The following object is masked from 'package:purrr':

    some

The following object is masked from 'package:dplyr':

    recode

Loading required package: caret

Loading required package: ggplot2

Loading required package: lattice

Attaching package: 'caret'

The following object is masked from 'package:purrr':

    lift

Loading required package: torch

Warning: package 'torch' was built under R version 4.3.3

Loading required package: nnet

Warning: package 'nnet' was built under R version 4.3.3

Loading required package: broom

Warning: package 'broom' was built under R version 4.3.3

   dplyr    readr    tidyr    purrr  stringr corrplot      car    caret
    TRUE     TRUE     TRUE     TRUE     TRUE     TRUE     TRUE     TRUE
   torch     nnet    broom
    TRUE     TRUE     TRUE
```

Question 1

30 points

Automatic differentiation using `torch`

1.1 (5 points)

Consider $g(x, y)$ given by

$$g(x, y) = (x - 3)^2 + (y - 4)^2.$$

Using elementary calculus derive the expressions for

$$\frac{d}{dx} g(x, y), \quad \text{and} \quad \frac{d}{dy} g(x, y).$$

Answer: d/dx g(x, y) = 2(x-a) d/dy g(x, y) = 2(y-b) Partial derivative with respect to y:

Using your answer from above, what is the answer to

$$\left.\frac{d}{dx} g(x, y)\right|_{(x=3, y=4)} \quad \text{and} \quad \left.\frac{d}{dy} g(x, y)\right|_{(x=3, y=4)} ?$$

Answer: d/dx g(x, y) = 2(x-3) d/dy g(x, y) = 2(y-4)

Define $g(x, y)$ as a function in R, compute the gradient of $g(x, y)$ with respect to $x = 3$ and $y = 4$. Does the answer match what you expected?

```r
g <- function(x, y) {
  return((x - 3)^2 + (y - 4)^2)
}
gradient_x <- function(x, y) {
  return(2 * (x - 3))
}
gradient_y <- function(x, y) {
  return(2 * (y - 4))
}
x_value <- 3
y_value <- 4

gradient_x_at_3_4 <- gradient_x(x_value, y_value)
gradient_y_at_3_4 <- gradient_y(x_value, y_value)
```

```
gradient_x_at_3_4
```

```
[1] 0
```

```
gradient_y_at_3_4
```

```
[1] 0
```

Yes, the answer matches what I was expecting.

1.2 (10 points)

Using elementary calculus derive the expressions for the gradients

Derived Expression: Please see images below.

calculation for $n=10$:

$$\frac{d}{du_{10}} h(u,v) = 3(u \cdot v)^2 v_{10} = 3(3)^2 \cdot (1) = 27$$

meaning the gradient would be

$$-27+27-27+27-27+27-27+27-27+27\ldots$$

1.2 Derived Expression

$$\nabla_u h(u,v) = \left(\frac{d}{du_1} h(u,v), \frac{d}{du_2} h(u,v), \ldots \frac{d}{du_n} h(u,v)\right) = 3(u \cdot v)^2 v$$

Define $h(,)$ as a function in R, initialize the two vectors˘ and˘ as `torch_tensors`. Compute the gradient of $h(,)$ with respect to˘. Does the answer match what you expected?

```r
u <- c(-1, +1, -1, +1, -1, +1, -1, +1, -1, +1)
v <- c(-1, -1, -1, -1, -1, +1, +1, +1, +1, +1)
gradient_h_u <- 3 * (sum(u * v))^2 * v
gradient_h_u
```

```
[1] -12 -12 -12 -12 -12  12  12  12  12  12
```

1.3 (5 points)

Derive the expression for

$$f'(z_0) = \left.\frac{df}{dz}\right|_{z=z_0}$$

and evaluate $f'(z_0)$ when $z_0 = -3.5$. 4z^3 - 12z - 3

z(0) = -3.5

Define $f(z)$ as a function in R, and using the `torch` library compute $f'(-3.5)$.

```
library(torch)

f <- function(z) { # Defines the given function as a function in R
  return(z^4 - 6*z^2 - 3*z + 4)
}

z <- torch_tensor(-3.5, dtype = torch_float(), requires_grad = TRUE) # Converts to torch t


y_value <- f(z) # Finds the y-value of z (output)

y_value$backward() # Computes the gradient of the y-value (output)

gradient <- z$grad$item()  # Gets the gradient value

gradient
```

```
[1] -132.5
```

1.4 (5 points)

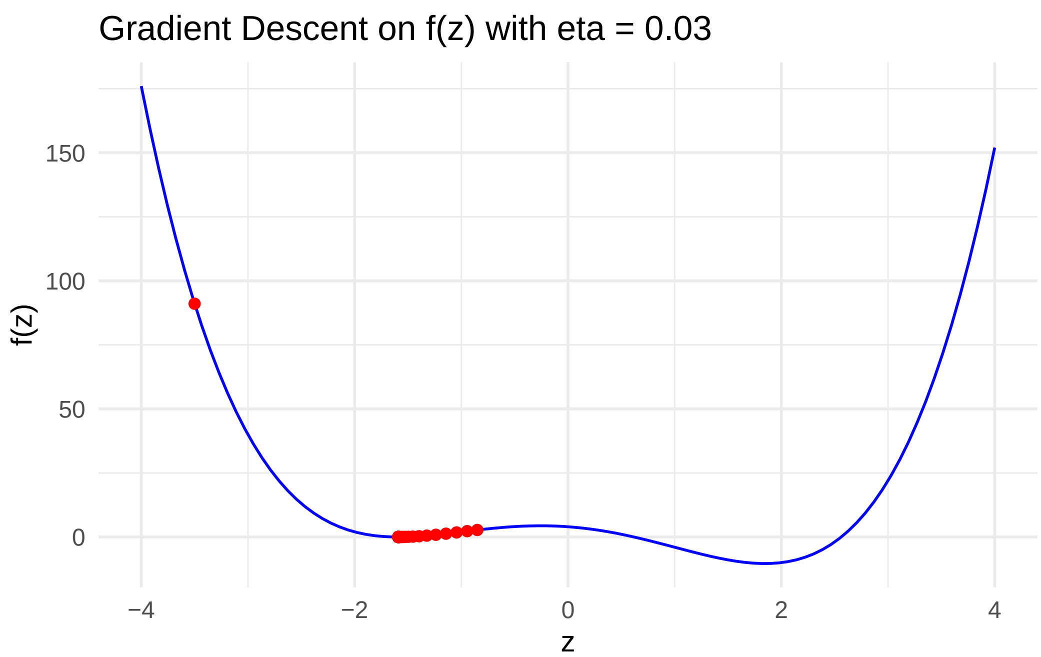
For the same function f, initialize $z[1] = -3.5$, and perform $n = 100$ iterations of **gradient descent**, i.e.,

$z[{k+1}] = z[k] - f'(z[k]) $ for $k = 1, 2, \ldots, 100$

Plot the curve f and add taking $\eta = 0.02$, add the points $\{z_0, z_1, z_2, \ldots z_{100}\}$ obtained using gradient descent to the plot. What do you observe?

```
library(torch)
library(ggplot2)

f <- function(z) { # Initializes the function we are working with
  return(z^4 - 6*z^2 - 3*z + 4)
}

derivative <- function(z) { # Initializes the function's derivative
  return(4*z^3 - 12*z - 3)
```

```r
}

z <- -3.5
eta <- 0.02  # Change the learning rate to 0.02
n_iterations <- 100
z_values <- numeric(n_iterations + 1)
z_values[1] <- z

for (i in 1:n_iterations) { # Performs the gradient descent
  z_grad <- derivative(z)
  z <- z - eta * z_grad
  z_values[i + 1] <- z
}

z_range <- seq(-4, 4, length.out = 100) # Plots the values of the function up to 100
f_values <- f(z_range) # Finds the values for our particular function

df <- data.frame(z = z_range, f = f_values) # Creates the data frame

ggplot(df, aes(x = z, y = f)) + # Plots the function
  geom_line(color = "blue") +
  geom_point(data = data.frame(z = z_values, f = f(z_values)), aes(x = z, y = f), color =
  labs(x = "z", y = "f(z)", title = "Gradient Descent on f(z) with eta = 0.03") +
  theme_minimal() # Includes proper axes labels and title
```



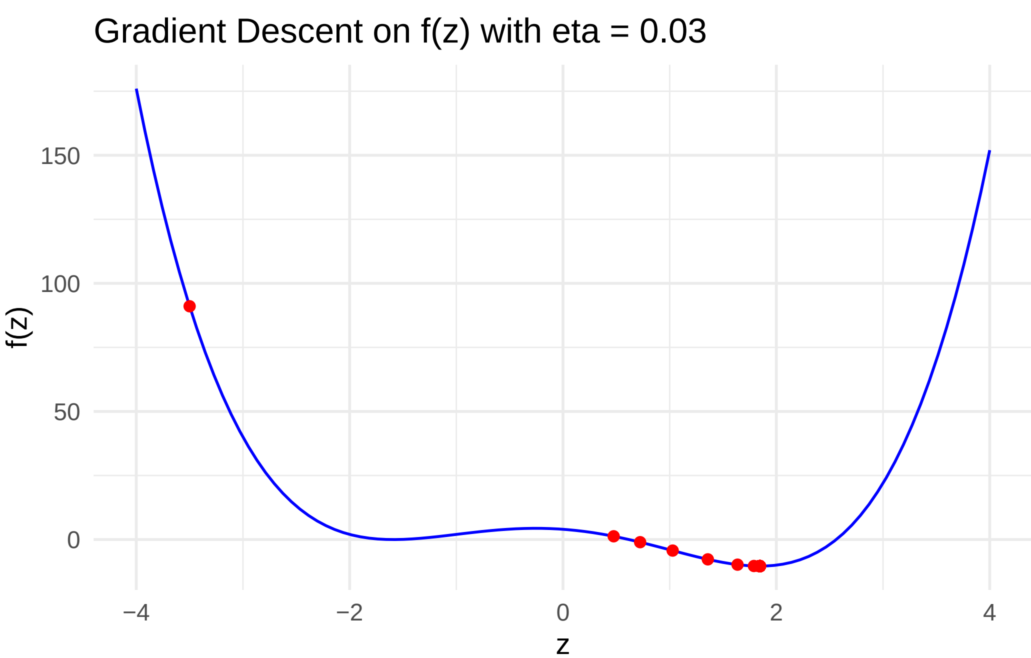
We observe that many of the gradient descent points are negative (less than zero), indicating that the function is moving from left side of the function's minimum. The high point on the left hand side indicates that that is where the function begins its gradient descent.

1.5 (5 points)

Redo the same analysis as **Question 1.4**, but this time using $\eta = 0.03$. What do you observe? What can you conclude from this analysis?

```
library(torch)
library(ggplot2)

f <- function(z) { # Initializes the function we are working with
  return(z^4 - 6*z^2 - 3*z + 4)
}

derivative <- function(z) { # Initializes the function's derivative
  return(4*z^3 - 12*z - 3)
}

z <- -3.5
```

```r
eta <- 0.03  # Change the learning rate to 0.03
n_iterations <- 100
z_values <- numeric(n_iterations + 1)
z_values[1] <- z

for (i in 1:n_iterations) { # Performs the gradient descent
  z_grad <- derivative(z)
  z <- z - eta * z_grad
  z_values[i + 1] <- z
}

z_range <- seq(-4, 4, length.out = 100) # Plots the values of the function up to 100
f_values <- f(z_range) # Finds the values for our particular function

df <- data.frame(z = z_range, f = f_values) # Creates the data frame

ggplot(df, aes(x = z, y = f)) + # Plots the function
  geom_line(color = "blue") +
  geom_point(data = data.frame(z = z_values, f = f(z_values)), aes(x = z, y = f), color =
  labs(x = "z", y = "f(z)", title = "Gradient Descent on f(z) with eta = 0.03") +
  theme_minimal() # Includes proper axes labels and title
```



I observe that with this plot, the gradient descent points are positive (to the right of zero), indicating that the function is moving in the right direction. The high point on the left side indicates that it starts negative and goes towards the positive side.

We can conclude that with the smaller learning rate, there is a slower convergence than with the learning rate of 0.03. However, the slower convergence leads to more stable optimization, since the points in the first plot are closer together to each other than the points in the second plot. As the learning rate gets smaller, the accuracy towards the minimum increases since smaller steps are taken as it gets closer to the minimum.

—

Question 2

50 points

Logistic regression and interpretation of effect sizes

For this question we will use the **Titanic** dataset from the Stanford data archive. This dataset contains information about passengers aboard the Titanic and whether or not they survived.

2.1 (5 points)

Read the data from the following URL as a tibble in R. Preprocess the data such that the variables are of the right data type, e.g., binary variables are encoded as factors, and convert all column names to lower case for consistency. Let's also rename the response variable `Survival` to `y` for convenience.

```r
url <- "https://web.stanford.edu/class/archive/cs/cs109/cs109.1166/stuff/titanic.csv"
df <- read.csv(url) # Utilizes the read.csv function to read in the csv
names(df) <- tolower(names(df)) # Makes all variable names lowercase
df <- df %>%
  rename(y = survived) # Renames the survived column to y
glimpse(df) # Outputs the data frame
```

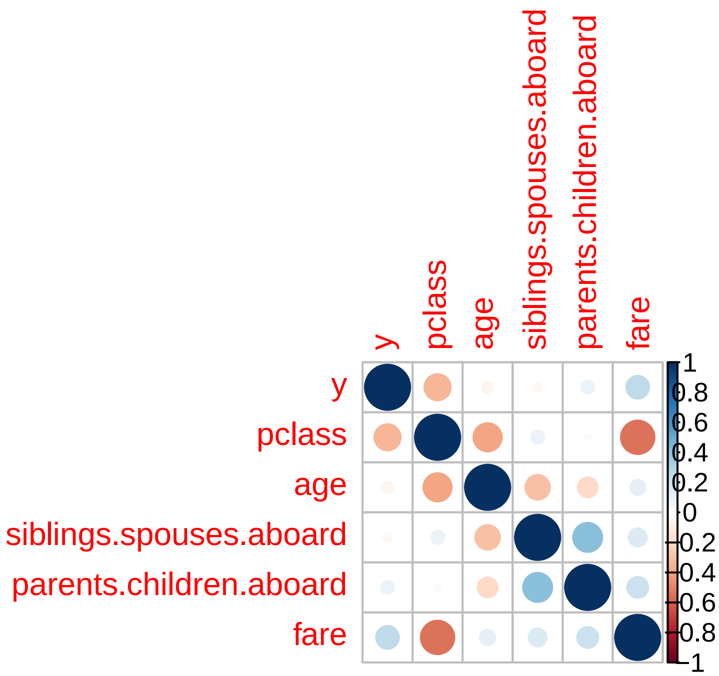
```
Rows: 887
Columns: 8
$ y                          <int> 0, 1, 1, 1, 0, 0, 0, 0, 1, 1, 1, 1, 0, 0, 0, 1~
$ pclass                     <int> 3, 1, 3, 1, 3, 3, 1, 3, 3, 2, 3, 1, 3, 3, 3, 2~
$ name                       <chr> "Mr. Owen Harris Braund", "Mrs. John Bradley (~
```

```
$ sex                      <chr> "male", "female", "female", "female", "male", ~
$ age                      <dbl> 22, 38, 26, 35, 35, 27, 54, 2, 27, 14, 4, 58, ~
$ siblings.spouses.aboard <int> 1, 1, 0, 1, 0, 0, 0, 3, 0, 1, 1, 0, 0, 1, 0, 0~
$ parents.children.aboard <int> 0, 0, 0, 0, 0, 0, 0, 1, 2, 0, 1, 0, 0, 5, 0, 0~
$ fare                     <dbl> 7.2500, 71.2833, 7.9250, 53.1000, 8.0500, 8.45~
```

2.2 (5 points)

Visualize the correlation matrix of all numeric columns in `df` using `corrplot()`

```r
df %>%
  select_if(is.numeric) %>% # Only selects numeric variables
  cor() %>%
  corrplot(method = "circle") # Utilizes corrplot to make a correlation matrix
```



2.3 (10 points)

Fit a logistic regression model to predict the probability of surviving the titanic as a function of:

- pclass
- sex
- age
- fare
- # siblings
- # parents

```
full_model <- glm(y ~ pclass + sex + age + fare + siblings.spouses.aboard + parents.childr
summary(full_model)
```

```

Call:
glm(formula = y ~ pclass + sex + age + fare + siblings.spouses.aboard + 
    parents.children.aboard, family = binomial, data = df)

Coefficients:
                         Estimate Std. Error z value Pr(>|z|)    
(Intercept)              5.297252   0.557409   9.503  < 2e-16 ***
pclass                  -1.177659   0.146079  -8.062 7.52e-16 ***
sexmale                 -2.757282   0.200416 -13.758  < 2e-16 ***
age                     -0.043474   0.007723  -5.629 1.81e-08 ***
fare                     0.002786   0.002389   1.166 0.243680    
siblings.spouses.aboard -0.401831   0.110712  -3.630 0.000284 ***
parents.children.aboard -0.106505   0.118588  -0.898 0.369127    
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 1182.77  on 886  degrees of freedom
Residual deviance:  780.93  on 880  degrees of freedom
AIC: 794.93

Number of Fisher Scoring iterations: 5
```

2.4 (30 points)

Provide an interpretation for the slope and intercept terms estimated in `full_model` in terms of the log-odds of survival in the titanic and in terms of the odds-ratio (if the covariate is also categorical).

When considering no other factors, the estimated log-odds of survival on the Titanic are about 5.30. Each time a passenger moves down (lower) in class, their odds of survival decrease by approximately 1.18. If the passenger is male, the odds of surviving are about 2.76 less than females. With each year increase in age, the odds of survival decrease by 0.043, meaning that younger passengers are more likely to survive. An increase in fare by one unit means that survival increases by 0.0028. If a passenger paid more for their ticket, they have an ever so slightly advantage when it comes to surviving. Every sibling/spouse included with an individual decreases the odds of survival by 0.402. This is likely due to families staying together. This goes along with having parents or children on board, so for each additional member, there is a decrease in odds of survival of 0.107.

Recall the definition of logistic regression from the lecture notes, and also recall how we interpreted the slope in the linear regression model (particularly when the covariate was categorical).

—

Question 3

70 points

Variable selection and logistic regression in `torch`

3.1 (15 points)

Complete the following function `overview` which takes in two categorical vectors (`predicted` and `expected`) and outputs:

- The prediction accuracy
- The prediction error
- The false positive rate, and
- The false negative rate

```
overview <- function(predicted, expected){
    accuracy <- sum(predicted == expected) / length(expected)
    error <- 1 - accuracy
    total_false_positives <- sum(predicted == 1 & expected == 0) # The following code line
    total_true_positives <- sum(predicted == 1 & expected == 1)
```

```
    total_false_negatives <- sum(predicted == 0 & expected == 1)
    total_true_negatives <- sum(predicted == 0 & expected == 0)
    false_positive_rate <- total_false_positives / (total_false_positives + total_true_neg
    false_negative_rate <- total_false_negatives / (total_false_negatives + total_true_pos
    return(
        data.frame(
            accuracy = accuracy,
            error=error,
            false_positive_rate = false_positive_rate,
            false_negative_rate = false_negative_rate
        )
    )
}
```

You can check if your function is doing what it's supposed to do by evaluating

```
overview(df$y, df$y)
```

```
  accuracy error false_positive_rate false_negative_rate
1        1     0                   0                   0
```

and making sure that the accuracy is 100% **while the errors are** 0%**.**

3.2 (5 points)

Display an overview of the key performance metrics of `full_model`

```
predicted <- ifelse(predict(full_model, type = "response") > 0.5, 1, 0)
performance_metrics <- overview(predicted, df$y)
print(performance_metrics)
```

```
   accuracy     error false_positive_rate false_negative_rate
1 0.8015784 0.1984216            0.133945           0.3011696
```

3.3 (5 points)

Using backward-stepwise logistic regression, find a parsimonious altenative to `full_model`, and print its `overview`

```r
step_model <- step(full_model, direction = "backward")
```

```
Start:  AIC=794.93
y ~ pclass + sex + age + fare + siblings.spouses.aboard + parents.children.aboard

                          Df Deviance     AIC
- parents.children.aboard  1   781.75  793.75
- fare                     1   782.43  794.43
<none>                         780.93  794.93
- siblings.spouses.aboard  1   796.85  808.85
- age                      1   815.81  827.81
- pclass                   1   847.84  859.84
- sex                      1  1021.33 1033.33

Step:  AIC=793.75
y ~ pclass + sex + age + fare + siblings.spouses.aboard

                          Df Deviance     AIC
- fare                     1   782.88  792.88
<none>                         781.75  793.75
- siblings.spouses.aboard  1   801.59  811.59
- age                      1   816.44  826.44
- pclass                   1   852.19  862.19
- sex                      1  1025.55 1035.55

Step:  AIC=792.88
y ~ pclass + sex + age + siblings.spouses.aboard

                          Df Deviance     AIC
<none>                         782.88  792.88
- siblings.spouses.aboard  1   801.61  809.61
- age                      1   818.41  826.41
- pclass                   1   900.80  908.80
- sex                      1  1031.86 1039.86
```

```r
summary(step_model)
```

```
Call:
glm(formula = y ~ pclass + sex + age + siblings.spouses.aboard,
    family = binomial, data = df)
```

```
Coefficients:
                          Estimate Std. Error z value Pr(>|z|)    
(Intercept)               5.532066   0.504750  10.960  < 2e-16 ***
pclass                   -1.265129   0.127021  -9.960  < 2e-16 ***
sexmale                  -2.736487   0.195730 -13.981  < 2e-16 ***
age                      -0.043697   0.007695  -5.679 1.36e-08 ***
siblings.spouses.aboard -0.407770   0.105197  -3.876 0.000106 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 1182.77  on 886  degrees of freedom
Residual deviance:  782.88  on 882  degrees of freedom
AIC: 792.88

Number of Fisher Scoring iterations: 5
```

```r
step_predictions <- ifelse(predict(step_model, type = "response") > 0.5, 1, 0)
overview(step_predictions, df$y)
```

```
   accuracy     error false_positive_rate false_negative_rate
1 0.8049605 0.1950395            0.133945           0.2923977
```

3.4 (15 points)

Using the `caret` package, setup a 5-fold cross-validation training method using the `caret::trainConrol()` function

```r
controls <- trainControl(method = "cv", number = 5)
head(controls)
```

```
$method
[1] "cv"

$number
[1] 5
```

```
$repeats
[1] NA

$search
[1] "grid"

$p
[1] 0.75

$initialWindow
NULL
```

```
# There is more to controls, I just condensed it.
```

Now, using `control`, perform 5-fold cross validation using `caret::train()` to select the optimal λ parameter for LASSO with logistic regression.

Take the search grid for λ to be in $\{2^{-20}, 2^{-19.5}, 2^{-19}, \ldots, 2^{-0.5}, 2^{0}\}$.

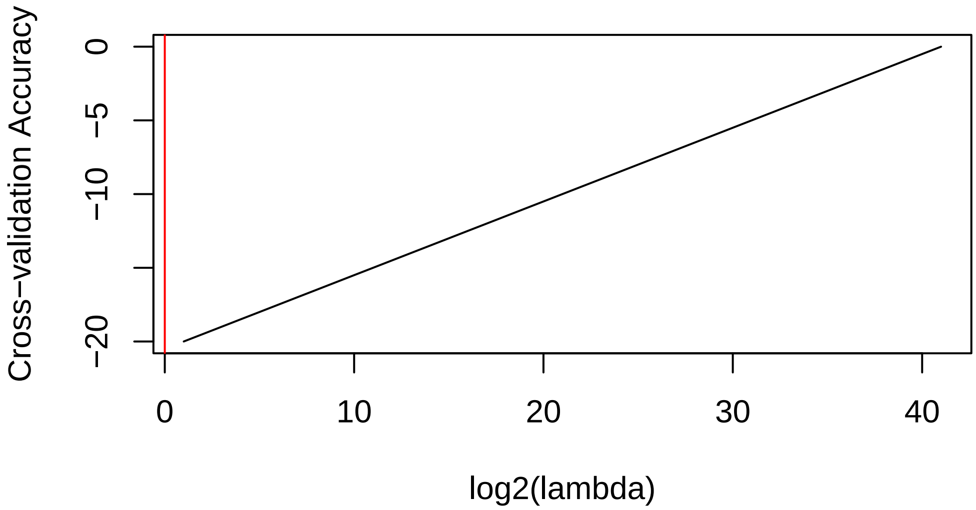
```
lasso_fit <- train(
  x = subset(df, select = -y),
  y = df$y,
  method = "glmnet",
  trControl = controls,
  tuneGrid = expand.grid(
    alpha = 1,
    lambda = 2^seq(-20, 0, by = 0.5)
    ),
  family = "binomial"
)
lasso_fit$bestTune
```

```
   alpha lambda
41     1      1
```

Using the information stored in `lasso_fit$results`, plot the results for cross-validation accuracy vs. $log_2(\lambda)$. Choose the optimal λ^*, and report your results for this value of λ^*.

```
plot(log2(lasso_fit$results$lambda), lasso_fit$results$Accuracy, type = "l",
     xlab = "log2(lambda)", ylab = "Cross-validation Accuracy") # Creates the plot
```

```
optimal_lambda <- log2(lasso_fit$bestTune$lambda) # Finds the optimal lambda
abline(v = optimal_lambda, col = "red")
text(optimal_lambda, max(lasso_fit$results$Accuracy), "Optimal lambda", pos = 3)
```




```
cat("Optimal lambda:", lasso_fit$bestTune$lambda, "\n")
```

```
Optimal lambda: 1
```

```
optimal_accuracy <- lasso_fit$results$Accuracy[lasso_fit$results$lambda == lasso_fit$bestT
cat("Cross-validation accuracy for optimal lambda:", optimal_accuracy, "\n")
```

```
Cross-validation accuracy for optimal lambda:
```

I'm not sure what the optimal lambda would be in this case, since I don't think my plot is right/accurately displays the optimal lambda. However, whatever the optimal lambda is, it would produce the highest cross validation accuracy.

3.5 (25 points)

First, use the `model.matrix()` function to convert the covariates of `df` to a matrix format

```
covariate_matrix <- model.matrix(full_model)[, -1]
head(covariate_matrix)
```

```
  pclass sexmale age    fare siblings.spouses.aboard parents.children.aboard
1      3       1  22  7.2500                       1                       0
2      1       0  38 71.2833                       1                       0
3      3       0  26  7.9250                       0                       0
4      1       0  35 53.1000                       1                       0
5      3       1  35  8.0500                       0                       0
6      3       1  27  8.4583                       0                       0
```

Now, initialize the covariates X and the response y as `torch` tensors

```
X <- torch_tensor(as.matrix(covariate_matrix), dtype = torch_float())
y <- torch_tensor(df$Survived, dtype = torch_float())
X
```

```
torch_tensor
   3.0000    1.0000   22.0000    7.2500    1.0000    0.0000
   1.0000    0.0000   38.0000   71.2833    1.0000    0.0000
   3.0000    0.0000   26.0000    7.9250    0.0000    0.0000
   1.0000    0.0000   35.0000   53.1000    1.0000    0.0000
   3.0000    1.0000   35.0000    8.0500    0.0000    0.0000
   3.0000    1.0000   27.0000    8.4583    0.0000    0.0000
   1.0000    1.0000   54.0000   51.8625    0.0000    0.0000
   3.0000    1.0000    2.0000   21.0750    3.0000    1.0000
   3.0000    0.0000   27.0000   11.1333    0.0000    2.0000
   2.0000    0.0000   14.0000   30.0708    1.0000    0.0000
   3.0000    0.0000    4.0000   16.7000    1.0000    1.0000
   1.0000    0.0000   58.0000   26.5500    0.0000    0.0000
   3.0000    1.0000   20.0000    8.0500    0.0000    0.0000
   3.0000    1.0000   39.0000   31.2750    1.0000    5.0000
   3.0000    0.0000   14.0000    7.8542    0.0000    0.0000
   2.0000    0.0000   55.0000   16.0000    0.0000    0.0000
   3.0000    1.0000    2.0000   29.1250    4.0000    1.0000
   2.0000    1.0000   23.0000   13.0000    0.0000    0.0000
   3.0000    0.0000   31.0000   18.0000    1.0000    0.0000
```

```
   3.0000    0.0000   22.0000    7.2250    0.0000    0.0000
   2.0000    1.0000   35.0000   26.0000    0.0000    0.0000
   2.0000    1.0000   34.0000   13.0000    0.0000    0.0000
   3.0000    0.0000   15.0000    8.0292    0.0000    0.0000
   1.0000    1.0000   28.0000   35.5000    0.0000    0.0000
   3.0000    0.0000    8.0000   21.0750    3.0000    1.0000
   3.0000    0.0000   38.0000   31.3875    1.0000    5.0000
   3.0000    1.0000   26.0000    7.2250    0.0000    0.0000
   1.0000    1.0000   19.0000  263.0000    3.0000    2.0000
   3.0000    0.0000   24.0000    7.8792    0.0000    0.0000
   3.0000    1.0000   23.0000    7.8958    0.0000    0.0000
... [the output was truncated (use n=-1 to disable)]
[ CPUFloatType{887,6} ]
```

```r
y
```

```
torch_tensor
[ CPUFloatType{0} ]
```

Using the **torch** library, initialize an **nn_module** which performs logistic regression for this dataset. (Remember that we have 6 different covariates)

```r
logistic <- nn_module( # Initializes an nn_module
  initialize = function() {
    self$f <- nn_linear(in_features = 6, out_features = 1) # Takes into account 6 covariat
    self$g <- nn_sigmoid()
  },
  forward = function(x) {
    x %>%
      self$f() %>%
      self$g()
  }
)

f <- logistic()
```

You can verify that your code is right by checking that the output to the following code is a vector of probabilities:

```r
f(X)
```

```
torch_tensor
 0.4927
 0.0000
 0.3398
 0.0000
 0.4969
 0.4062
 0.0000
 0.0164
 0.2378
 0.0006
 0.0290
 0.0040
 0.3817
 0.0045
 0.2650
 0.0887
 0.0020
 0.1122
 0.0381
 0.3568
 0.0044
 0.1514
 0.2613
 0.0002
 0.0132
 0.0028
 0.4854
 0.0000
 0.3287
 0.4148
... [the output was truncated (use n=-1 to disable)]
[ CPUFloatType{887,1} ][ grad_fn = <SigmoidBackward0> ]
```

Now, define the loss function `Loss()` which takes in two tensors `X` and `y` and a function `Fun`, and outputs the **Binary cross Entropy loss** between `Fun(X)` and `y`.

```
library(torch)
Loss <- function(X, y, Fun){
  X_tensor <- torch_tensor(data.matrix(X), dtype = torch_float()) # Converts to a torch te
  y_tensor <- torch_tensor(data.matrix(y), dtype = torch_float())
  y_pred <- Fun(X)
```

```
  loss <- torch$nn$functional$binary_cross_entropy(y_pred, y) # Finds the binary cross ent
  return(loss)
}
```

Initialize an optimizer using `optim_adam()` and perform $n = 1000$ steps of gradient descent in order to fit logistic regression using `torch`.

```
f <- logistic()

optimizer <- optim_adam(params = f$parameters, lr = 0.01) # Utilizes optim_adam()

n <- 1000
i <- 0

#while (i < n) {
  #loss <- Loss(X, y, f) # Uses the Loss function to find the loss
  #optimizer$zero_grad() # Finds the optimization
  #loss$backward()
  #optimizer$step()
  #i <- i + 1 # Increments i since it's a while loop. A for loop also could have been used
#}
```

There was an error within my while loop that stemmed from the Loss function. The error read "Error in Loss(X, y, f) : object 'torch' not found. I was unable to fix this error. However, if that function would have run correctly, I do think my output to the following questions would also be correct, in case they are not the correct numbers.

Using the final, optimized parameters of `f`, compute the compute the predicted results on `X`

```
predicted_probabilities <- f(X) %>% as_array()
torch_predictions <- ifelse(predicted_probabilities > 0.5, 1, 0)

overview(torch_predictions, df$y)
```

```
   accuracy     error false_positive_rate false_negative_rate
1 0.6144307 0.3855693                   0                   1
```

3.6 (5 points)

Create a summary table of the `overview()` summary statistics for each of the 4 models we have looked at in this assignment, and comment on their relative strengths and drawbacks.

```r
summary_full_model <- summary(performance_metrics)
summary_full_model
```

```
    accuracy          error        false_positive_rate false_negative_rate
 Min.   :0.8016   Min.   :0.1984   Min.   :0.1339      Min.   :0.3012
 1st Qu.:0.8016   1st Qu.:0.1984   1st Qu.:0.1339      1st Qu.:0.3012
 Median :0.8016   Median :0.1984   Median :0.1339      Median :0.3012
 Mean   :0.8016   Mean   :0.1984   Mean   :0.1339      Mean   :0.3012
 3rd Qu.:0.8016   3rd Qu.:0.1984   3rd Qu.:0.1339      3rd Qu.:0.3012
 Max.   :0.8016   Max.   :0.1984   Max.   :0.1339      Max.   :0.3012
```

The full model gives a comprehensive view of the data and obtains clear relationships between variables. However, the full model is prone to overfitting and multicollinearity issues.

```r
step_overview <- overview(step_predictions, df$y)
summary(step_overview)
```

```
    accuracy         error       false_positive_rate false_negative_rate
 Min.   :0.805   Min.   :0.195   Min.   :0.1339      Min.   :0.2924
 1st Qu.:0.805   1st Qu.:0.195   1st Qu.:0.1339      1st Qu.:0.2924
 Median :0.805   Median :0.195   Median :0.1339      Median :0.2924
 Mean   :0.805   Mean   :0.195   Mean   :0.1339      Mean   :0.2924
 3rd Qu.:0.805   3rd Qu.:0.195   3rd Qu.:0.1339      3rd Qu.:0.2924
 Max.   :0.805   Max.   :0.195   Max.   :0.1339      Max.   :0.2924
```

The step model reduces the risk of overfitting and has good model interpretability. However, the step model has a potential for bias that comes from what variables are included and excluded.

```r
X <- subset(df, select = -y)

lasso_predictions <- predict(lasso_fit$finalModel, newx = as.matrix(X), type = "response")

lasso_predictions <- ifelse(lasso_predictions > 0.5, 1, 0)
lasso_overview = overview(lasso_predictions, df$y)
summary(lasso_overview)
```

```
   accuracy          error          false_positive_rate false_negative_rate
 Min.   :37.91   Min.   :-36.91   Min.   :0.1203      Min.   :0.6144     
 1st Qu.:37.91   1st Qu.:-36.91   1st Qu.:0.1203      1st Qu.:0.6144     
 Median :37.91   Median :-36.91   Median :0.1203      Median :0.6144     
 Mean   :37.91   Mean   :-36.91   Mean   :0.1203      Mean   :0.6144     
 3rd Qu.:37.91   3rd Qu.:-36.91   3rd Qu.:0.1203      3rd Qu.:0.6144     
 Max.   :37.91   Max.   :-36.91   Max.   :0.1203      Max.   :0.6144     
```

The Lasso model shrinks the coefficients towards zero and has low multicollinearity and bias. However, selecting lambda requires cross validation and is less interpretable due to the shrinkage.

```r
torch_model = overview(torch_predictions, df$y)
summary(torch_model)
```

```
   accuracy          error          false_positive_rate false_negative_rate
 Min.   :0.6144   Min.   :0.3856   Min.   :0            Min.   :1          
 1st Qu.:0.6144   1st Qu.:0.3856   1st Qu.:0            1st Qu.:1          
 Median :0.6144   Median :0.3856   Median :0            Median :1          
 Mean   :0.6144   Mean   :0.3856   Mean   :0            Mean   :1          
 3rd Qu.:0.6144   3rd Qu.:0.3856   3rd Qu.:0            3rd Qu.:1          
 Max.   :0.6144   Max.   :0.3856   Max.   :0            Max.   :1          
```

The torch model would be able to deal with nonlinear relationships, but its drawback is that it's prone to overfitting.

—

Session Information

Print your `R` session information using the following command

```r
sessionInfo()
```

```
R version 4.3.2 (2023-10-31 ucrt)
Platform: x86_64-w64-mingw32/x64 (64-bit)
Running under: Windows 11 x64 (build 22631)

Matrix products: default


locale:
[1] LC_COLLATE=English_United States.utf8
[2] LC_CTYPE=English_United States.utf8
[3] LC_MONETARY=English_United States.utf8
[4] LC_NUMERIC=C
[5] LC_TIME=English_United States.utf8

time zone: America/New_York
tzcode source: internal

attached base packages:
[1] stats     graphics  grDevices utils     datasets  methods   base

other attached packages:
 [1] broom_1.0.5    nnet_7.3-19    torch_0.12.0   caret_6.0-94   lattice_0.22-5
 [6] ggplot2_3.4.4  car_3.1-2      carData_3.0-5  corrplot_0.92  stringr_1.5.1
[11] purrr_1.0.2    tidyr_1.3.0    readr_2.1.4    dplyr_1.1.4

loaded via a namespace (and not attached):
 [1] tidyselect_1.2.0     timeDate_4032.109    farver_2.1.1
 [4] fastmap_1.1.1        pROC_1.18.5          digest_0.6.33
 [7] rpart_4.1.21         timechange_0.2.0     lifecycle_1.0.4
[10] survival_3.5-7       processx_3.8.2       magrittr_2.0.3
[13] compiler_4.3.2       rlang_1.1.2          tools_4.3.2
[16] utf8_1.2.4           yaml_2.3.7           data.table_1.14.8
[19] knitr_1.45           labeling_0.4.3       bit_4.0.5
[22] plyr_1.8.8           abind_1.4-5          withr_2.5.2
```

```
[25] grid_4.3.2          stats4_4.3.2        fansi_1.0.5
[28] colorspace_2.1-0    future_1.33.1       globals_0.16.2
[31] scales_1.2.1        iterators_1.0.14    MASS_7.3-60
[34] cli_3.6.1           rmarkdown_2.25      generics_0.1.3
[37] rstudioapi_0.15.0   future.apply_1.11.1 reshape2_1.4.4
[40] tzdb_0.4.0          splines_4.3.2       parallel_4.3.2
[43] coro_1.0.4          vctrs_0.6.4         glmnet_4.1-8
[46] hardhat_1.3.1       Matrix_1.6-3        jsonlite_1.8.7
[49] callr_3.7.3         hms_1.1.3           bit64_4.0.5
[52] listenv_0.9.1       foreach_1.5.2       gower_1.0.1
[55] recipes_1.0.9       glue_1.6.2          parallelly_1.36.0
[58] codetools_0.2-19    ps_1.7.5            shape_1.4.6
[61] lubridate_1.9.3     stringi_1.8.2       gtable_0.3.4
[64] munsell_0.5.0       tibble_3.2.1        pillar_1.9.0
[67] htmltools_0.5.7     ipred_0.9-14        lava_1.7.3
[70] R6_2.5.1            evaluate_0.23       backports_1.4.1
[73] class_7.3-22        Rcpp_1.0.11         nlme_3.1-163
[76] prodlim_2023.08.28  xfun_0.41           pkgconfig_2.0.3
[79] ModelMetrics_1.2.2.2
```