Homework 4

Nicholas Allen

Table of contents

Question 1 4
Question 2 9
Question 3 12

> **❗ 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",
  "corrplot",
  "car",
  "caret",
```

```r
  "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

Loading required package: nnet

Loading required package: broom

   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

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

1. $$\frac{d}{dx}g(x, y) = (2x - 6) + (y - 4)^2$$

2. $$\frac{d}{dy}g(x, y) = (x - 3)^2 + (2y - 8)$$

Using your answer from above, what is the answer to

$$\frac{d}{dx}g(x, y)\bigg|_{(x=3,y=4)} 0 \quad \text{and} \quad \frac{d}{dy}g(x, y)\bigg|_{(x=3,y=4)} 0$$

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
q <- torch_tensor(3, requires_grad=T)
y <- torch_tensor(4, requires_grad=T)
g <- ((2*q) - 6) + (y - 4)^2
g$backward()

q$grad
```

```
torch_tensor
 2
[ CPUFloatType{1} ]
```

The answer matched what I expected

1.2 (10 points)

Consider $h(u, v)$ given by

$$h(u,) = (u \cdot)^3,$$

where $\check{\cdot}$ denotes the dot product of two vectors, i.e., $\cdot = \sum_{i=1}^{n} u_i v_i$.

Using elementary calculus derive the expressions for the gradients

$$\nabla_u h(u, v) = 3u^2 v^3$$

Using your answer from above, what is the answer to $\nabla_u h(u, v)$ when $n = 10$ and

$$u = (-1, +1, -1, +1, -1, +1, -1, +1, -1, +1)$$
$$v = (-1, -1, -1, -1, -1, +1, +1, +1, +1, +1)$$

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

```
h <- function(u,v){
  (u*v)^3

}
u <- torch_tensor(c(-1, 1, -1, 1, -1, 1, -1, 1, -1, 1), requires_grad = T)
v <- torch_tensor(c(-1, -1, -1, -1, -1, 1, 1, 1, 1, 1), requires_grad=T)
```

1.3 (5 points)

Consider the following function

$$f(z) = z^4 - 6z^2 - 3z + 4$$

Derive the expression for

$$f'(z_0) = \left. \frac{df}{dz} \right|_{z=z_0} = 4z^3 - 12z - 3$$

and evaluate $f'(z_0)$ when $z_0 = -3.5$.

```r
4*(-3.5)^3 - 12*(-3.5) - 3
```

```
[1] -132.5
```

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

```r
f <- function(z){
  z^4 - 6*(z)^2 - 3*(z) + 4
}



z <- torch_tensor(-3.5, requires_grad=T)

z0 <- f(z)

z0$backward()

z$grad
```

```
torch_tensor
-132.5000
[ CPUFloatType{1} ]
```

1.4 (5 points)

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

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

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

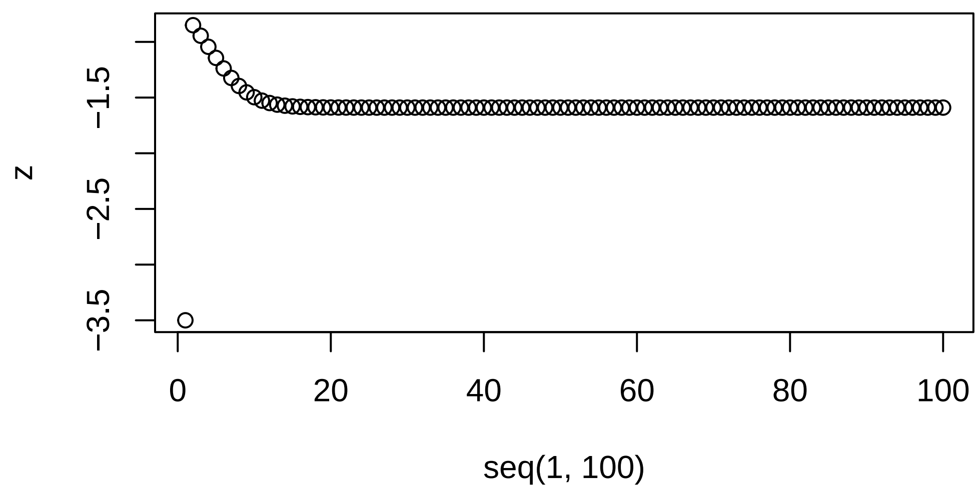
The gradients get closer to 0

```
f_der <- function(z)
  return(4*(z^3)-12*(z)-3)

steps <- 100
z <- rep(NA, steps)
z[1] <- -3.5

for (i in 1:(steps-1)){
  z[i+1] <- z[i]-(0.02*f_der(z[i]))
}

plot(seq(1,100),z)
```



It seems to level out at around -2

1.5 (5 points)

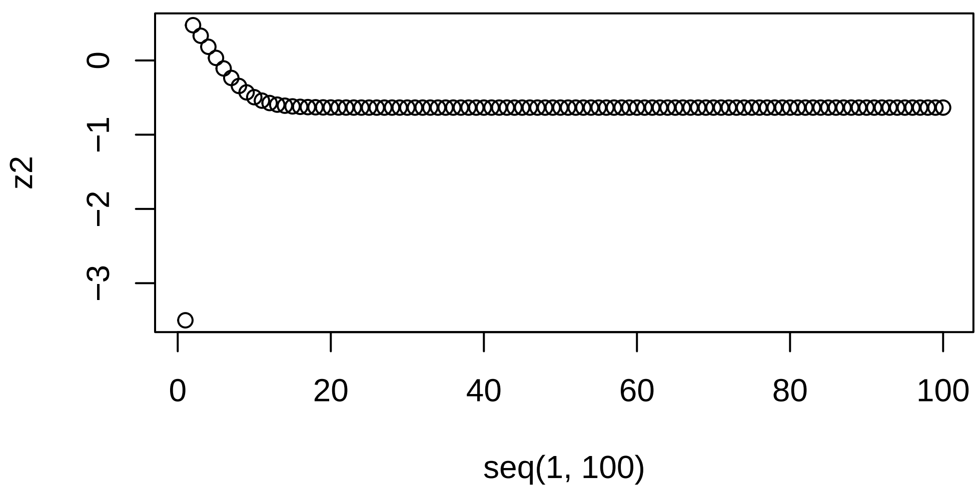
Redo the same analysis as **Question 1.4**, but this time using $\eta = 0.03$. What do you observe? What can you conclude from this analysis

```
f_der2 <- function(z)
  return(4*(z^3)-12*(z)-3)

steps <- 100
z2 <- rep(NA, steps)
z2[1] <- -3.5

for (i in 1:(steps-1)){
  z2[i+1] <- z2[i]-(0.03*f_der2(z[i]))
}

plot(seq(1,100),z2)
```



This time it seems to level out at 2. It also looks as though it is increasing while the other one was decreasing

---

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.

```
url <- "https://web.stanford.edu/class/archive/cs/cs109/cs109.1166/stuff/titanic.csv"

df <- read_csv(url) %>% as_tibble()
```

```
Rows: 887 Columns: 8
-- Column specification --------------------------------------------------------
Delimiter: ","
chr (2): Name, Sex
dbl (6): Survived, Pclass, Age, Siblings/Spouses Aboard, Parents/Children Ab...

i Use `spec()` to retrieve the full column specification for this data.
i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
df$Survived <- as.factor(df$Survived)
df$Pclass <- as.factor(df$Pclass)
df$Sex <- as.factor(df$Sex)
df <- df %>% rename('y' = 'Survived')
names(df)[2:8] <- tolower(names(df)[2:8])
df
```

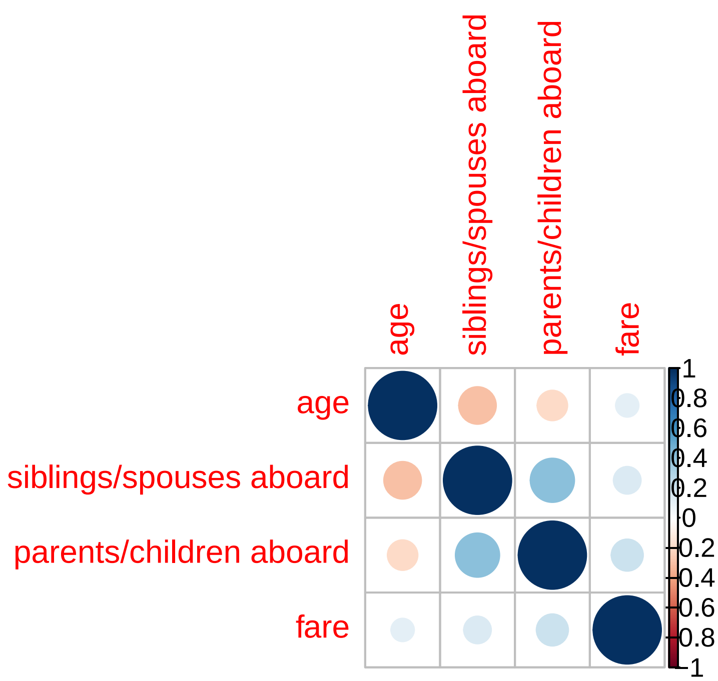
```
# A tibble: 887 x 8
   y     pclass name  sex     age siblings/spouses abo~1 parents/children abo~2
   <fct> <fct>  <chr> <fct> <dbl>                  <dbl>                  <dbl>
```

```
 1 0      3       Mr. O~ male     22                       1                       0
 2 1      1       Mrs. ~ fema~    38                       1                       0
 3 1      3       Miss.~ fema~    26                       0                       0
 4 1      1       Mrs. ~ fema~    35                       1                       0
 5 0      3       Mr. W~ male     35                       0                       0
 6 0      3       Mr. J~ male     27                       0                       0
 7 0      1       Mr. T~ male     54                       0                       0
 8 0      3       Maste~ male      2                       3                       1
 9 1      3       Mrs. ~ fema~    27                       0                       2
10 1      2       Mrs. ~ fema~    14                       1                       0
# i 877 more rows
# i abbreviated names: 1: `siblings/spouses aboard`,
#   2: `parents/children aboard`
# i 1 more variable: fare <dbl>
```

2.2 (5 points)

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

```
df %>% select(!c(y, pclass, name, sex)) %>% cor() %>% corrplot()
```



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

```r
df <- df %>% select(!name)
full_model <- glm(y ~ ., data = df, family = binomial)
summary(full_model)
```

```

Call:
glm(formula = y ~ ., family = binomial, data = df)

Coefficients:
                           Estimate Std. Error z value Pr(>|z|)
(Intercept)                4.109777   0.463602   8.865  < 2e-16 ***
pclass2                   -1.161491   0.300960  -3.859 0.000114 ***
pclass3                   -2.350022   0.304666  -7.713 1.22e-14 ***
sexmale                   -2.756710   0.200642 -13.739  < 2e-16 ***
age                       -0.043410   0.007790  -5.573 2.51e-08 ***
`siblings/spouses aboard` -0.401572   0.110795  -3.624 0.000290 ***
`parents/children aboard` -0.106884   0.118767  -0.900 0.368151
fare                       0.002823   0.002468   1.144 0.252771
---
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 879  degrees of freedom
AIC: 796.93

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 all numerical values are set to 0 and the person is in pclass one and they are female the log odds of survival are 4.1.

If fare increases by one, log odds of survival increase by 0.002823.

If age increases by one, log odds of survival decrease 0.043410.

If # parents/children increases by one, log odds of survival decrease by 0.107.

If # siblings/spouses increase by one, log odds of survival decrease by 0.402.

If a passenger is 2nd class their log odds of survival decrease by 1.16

If a passenger is 3rd class their log odds of survival decrease by 2.35

If a passenger is male their log odds of survival decrease by 2.75

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){
    x <- table(expected, predicted)
    accuracy <- ((x[1] + x[4]) / length(expected))*100
    error <- 100 - accuracy
    total_false_positives <- x[3]
    total_true_positives <- x[4]
    total_false_negatives <- x[2]
    total_true_negatives <- x[1]
    false_positive_rate <- total_false_positives / (total_false_negatives + total_true_negat
    false_negative_rate <- total_false_negatives / (total_false_negatives + total_true_posit
    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      100     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

```
yhat <- predict(full_model, type = 'response')
yhat2 <- ifelse(yhat <0.5, 0, 1)
overview(yhat2, df$y)
```

```
  accuracy    error false_positive_rate false_negative_rate
1 80.27057 19.72943               0.125           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=796.93
y ~ pclass + sex + age + `siblings/spouses aboard` + `parents/children aboard` +
    fare

                            Df Deviance     AIC
- `parents/children aboard`  1   781.75  795.75
- fare                       1   782.37  796.37
<none>                           780.93  796.93
- `siblings/spouses aboard`  1   796.79  810.79
- age                        1   815.20  829.20
- pclass                     2   847.84  859.84
- sex                        1  1020.26 1034.26

Step:  AIC=795.75
y ~ pclass + sex + age + `siblings/spouses aboard` + fare

                            Df Deviance     AIC
- fare                       1   782.82  794.82
<none>                           781.75  795.75
- `siblings/spouses aboard`  1   801.56  813.56
- age                        1   815.88  827.88
- pclass                     2   852.19  862.19
- sex                        1  1024.08 1036.08

Step:  AIC=794.82
y ~ pclass + sex + age + `siblings/spouses aboard`

                            Df Deviance     AIC
<none>                           782.82  794.82
- `siblings/spouses aboard`  1   801.59  811.59
- age                        1   818.25  828.25
- pclass                     2   900.80  908.80
- sex                        1  1031.69 1041.69
```

```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)                4.294169   0.417879  10.276  < 2e-16 ***
pclass2                   -1.321703   0.268452  -4.923  8.5e-07 ***
pclass3                   -2.541237   0.258324  -9.837  < 2e-16 ***
sexmale                   -2.738024   0.195796 -13.984  < 2e-16 ***
age                       -0.043918   0.007757  -5.662  1.5e-08 ***
`siblings/spouses aboard` -0.409624   0.105495  -3.883 0.000103 ***
---
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:  782.82  on 881  degrees of freedom
AIC: 794.82

Number of Fisher Scoring iterations: 5
```

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

```
  accuracy    error false_positive_rate false_negative_rate
1 80.49605 19.50395           0.1276224           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='repeatedcv', number=5)
```

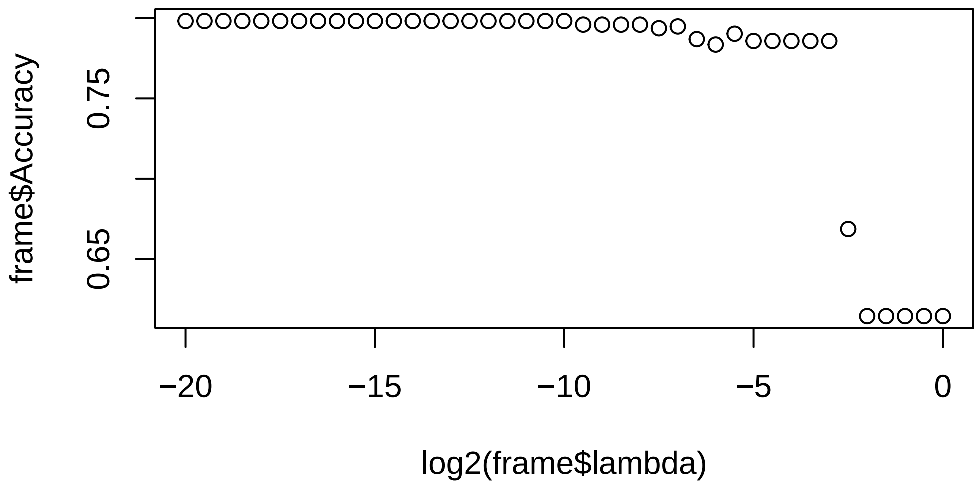
Now, using `control`, perform 5-fold cross validation using `caret::train()` to select the optimal λ parameter for LASSO with logistic regression.

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

```r
# Insert your code in the ... region
lasso_fit <- train(
  x = df %>% select(!y) %>% data.matrix(),
  y = df$y,
  method = 'glmnet',
  trControl = controls,
  tuneGrid = expand.grid(
    alpha = 1,
    lambda = 2^seq(-20, 0, by = 0.5)
    ),
  family = 'binomial'
  )
```

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^*.

```r
frame <- lasso_fit$results
plot(log2(frame$lambda), frame$Accuracy)
```



The optimal value of lambda is 0.125 because this is the value where any larger causes the accuracy to spike downwards.

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]
```

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

```
X <- torch_tensor(covariate_matrix)
y <- torch_tensor(as.numeric(df$y)-1)
```

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(
  initialize = function() {
    self$f <- nn_linear(7,1)
    self$g <- nn_sigmoid()
  },
  forward = function(x) {
    x %>% self$f() %>% self$g()
  }
)

f1 <- logistic()
```

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

```r
f1(X)
```

```
torch_tensor
 0.7402
 1.0000
 0.8025
 1.0000
 0.7304
 0.7570
 0.9999
 0.9807
 0.8361
 0.9978
 0.9607
 0.9923
 0.7518
 0.9905
 0.8141
 0.9498
 0.9964
 0.9028
 0.9691
 0.7849
 0.9908
 0.8954
 0.8182
```

```
 0.9985
 0.9859
 0.9935
 0.7114
 1.0000
 0.8034
 0.7419
... [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`.

```
Loss <- function(x, y, Fun){
  nnf_binary_cross_entropy(Fun(x), y)

}
```

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

```
f2 <- logistic()
f2$parameters
```

```
$f.weight
torch_tensor
-0.0577 -0.1009  0.0700  0.2092 -0.1678  0.2205 -0.1845
[ CPUFloatType{1,7} ][ requires_grad = TRUE ]

$f.bias
torch_tensor
 0.2226
[ CPUFloatType{1} ][ requires_grad = TRUE ]
```

```
optimizer <- optim_adam(f2$parameters, lr=0.01)


n <- 1000
for (i in 1:n){
    loss <- Loss(X, y, f2)
```

```
    optimizer$zero_grad()
    loss$backward()
    optimizer$step()
}
f2$parameters
```

```
$f.weight
torch_tensor
-0.1984 -1.1642 -2.3969 -0.0215 -0.3243 -0.1077  0.0090
[ CPUFloatType{1,7} ][ requires_grad = TRUE ]

$f.bias
torch_tensor
 2.1540
[ CPUFloatType{1} ][ requires_grad = TRUE ]
```

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

```
predicted_probabilities <- f2(X) %>% as_array()
torch_predictions <- ifelse(predicted_probabilities < 0.5, 0, 1)

overview(torch_predictions, df$y)
```

```
  accuracy    error false_positive_rate false_negative_rate
1 80.15784 19.84216           0.1120543           0.3216374
```

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.

```
lasso_pred <- predict(lasso_fit)

summary_table <- rbind(
overview(lasso_pred, df$y),
overview(yhat2, df$y),
overview(step_predictions, df$y),
overview(torch_predictions, df$y)
```

```r
)
summary_table$name <- c('lasso', 'full', 'step', 'nn')
summary_table <- summary_table[,c(5,1,2,3,4)]

summary_table
```

```
   name accuracy    error false_positive_rate false_negative_rate
1 lasso 80.04510 19.95490           0.1267361           0.3040936
2  full 80.27057 19.72943           0.1250000           0.3011696
3  step 80.49605 19.50395           0.1276224           0.2923977
4    nn 80.15784 19.84216           0.1120543           0.3216374
```

All of the models have about the same accuracy. The neural network logistic regression has the lowest false positive rate by far and a slightly higher false negative rate.

Session Information

Print your R session information using the following command

```
sessionInfo()
```

```
R version 4.3.3 (2024-02-29 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_3.4.3  car_3.1-2      carData_3.0-5  corrplot_0.92  stringr_1.5.1
[11] purrr_1.0.2    tidyr_1.3.1    readr_2.1.5    dplyr_1.1.4

loaded via a namespace (and not attached):
 [1] tidyselect_1.2.1     timeDate_4032.109    fastmap_1.1.0
 [4] pROC_1.18.5          digest_0.6.31        rpart_4.1.23
 [7] timechange_0.2.0     lifecycle_1.0.4      survival_3.5-8
[10] processx_3.8.2       magrittr_2.0.3       compiler_4.3.3
[13] rlang_1.1.3          tools_4.3.3          utf8_1.2.4
[16] yaml_2.3.6           data.table_1.14.6    knitr_1.41
[19] curl_5.0.0           bit_4.0.5            plyr_1.8.8
[22] abind_1.4-5          withr_3.0.0          grid_4.3.3
```

```
[25] stats4_4.3.3        fansi_1.0.6         e1071_1.7-14
[28] colorspace_2.0-3    future_1.33.1       globals_0.16.2
[31] scales_1.2.1        iterators_1.0.14    MASS_7.3-60.0.1
[34] cli_3.6.2           crayon_1.5.2        rmarkdown_2.19
[37] generics_0.1.3      rstudioapi_0.14     future.apply_1.11.1
[40] reshape2_1.4.4      tzdb_0.4.0          proxy_0.4-27
[43] splines_4.3.3       parallel_4.3.3      coro_1.0.4
[46] vctrs_0.6.5         glmnet_4.1-6        hardhat_1.3.1
[49] Matrix_1.6-5        jsonlite_1.8.4      callr_3.7.3
[52] hms_1.1.3           bit64_4.0.5         listenv_0.9.1
[55] foreach_1.5.2       gower_1.0.1         recipes_1.0.10
[58] glue_1.7.0          parallelly_1.37.0   codetools_0.2-19
[61] ps_1.7.3            shape_1.4.6         lubridate_1.9.3
[64] stringi_1.7.12      gtable_0.3.1        munsell_0.5.0
[67] tibble_3.2.1        pillar_1.9.0        htmltools_0.5.4
[70] ipred_0.9-14        lava_1.7.3          R6_2.5.1
[73] vroom_1.6.5         evaluate_0.19       backports_1.4.1
[76] class_7.3-22        Rcpp_1.0.9          nlme_3.1-164
[79] prodlim_2023.08.28  xfun_0.36           pkgconfig_2.0.3
[82] ModelMetrics_1.2.2.2
```