Homework 4

Marc Hughes

Table of contents

Link to the Github repository

❗ Due: Sun, Apr 2, 2023 @ 11:59pm

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",
```

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

  renv::install(packages)
```

```
Installing dplyr [1.1.1] ...
    OK [linked cache in 5.5 milliseconds]
Installing readr [2.1.4] ...
    OK [linked cache in 4.4 milliseconds]
Installing purrr [1.0.1] ...
    OK [linked cache in 5.3 milliseconds]
Installing stringr [1.5.0] ...
    OK [linked cache in 5.7 milliseconds]
Installing tidyr [1.3.0] ...
    OK [linked cache in 4.9 milliseconds]
Installing corrplot [0.92] ...
    OK [linked cache in 4.7 milliseconds]
Installing nnet [7.3-18] ...
    OK [linked cache in 4.9 milliseconds]
Installing broom [1.0.4] ...
    OK [linked cache in 7.3 milliseconds]
Installing car [3.1-2] ...
    OK [linked cache in 5.1 milliseconds]
Installing caret [6.0-94] ...
    OK [linked cache in 6.3 milliseconds]
Installing torch [0.9.1] ...
    OK [linked cache in 4.6 milliseconds]
```

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

```
Loading required package: dplyr

Warning: package 'dplyr' was built under R version 4.2.3

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

Warning: package 'tidyr' was built under R version 4.2.2

Loading required package: purrr

Warning: package 'purrr' was built under R version 4.2.2

Loading required package: stringr

Warning: package 'stringr' was built under R version 4.2.2

Loading required package: corrplot

Warning: package 'corrplot' was built under R version 4.2.2

corrplot 0.92 loaded

Loading required package: car

Loading required package: carData

Warning: package 'carData' was built under R version 4.2.2

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

Warning: package 'caret' was built under R version 4.2.3

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 4.2.2

Loading required package: nnet

Warning: package 'nnet' was built under R version 4.2.3

Loading required package: broom

Warning: package 'broom' was built under R version 4.2.2

   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

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

$$\frac{d}{dx} g(x, y) = 2x - 6, \quad \frac{d}{dy} g(x, y) = 2y - 8$$

Using your answer from above, what is the answer to

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

The answer to both is 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
library(numDeriv)


g <- function(x) {
  (x[1]-3)^2 + (x[2]-4)^2
}


gradient <- grad(g, c(3, 4))
gradient
```

```
[1] 0 0
```

Yes, the answer matches exactly what was expected.

1.2 (10 points)

```
# command not working as intended so I put it into a comment
# $$\newcommand{\u}{\boldsymbol{u}}\newcommand{\v}{\boldsymbol{v}}$$
```

Consider $h(,)$ given by

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

where ˇ· 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

Had to comment out because I was getting an error when rendering…

The answer is below:

$$= \left(3(u \cdot v)^2 \times v_1, 3(u \cdot v)^2 \times v_2, \ldots, 3(u \cdot v)^2 \times v_n\right)$$

Using your answer from above, what is the answer to change in $h(,)$ when $n = 10$ and

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

The answer is (-12, -12, -12, -12, -12, 12, 12, 12, 12, 12).

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

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

}

u <- torch_tensor(c(-1, 1, -1, 1, -1, 1, -1, 1, -1, 1), requires_grad=TRUE)
v <- torch_tensor(c(-1, -1, -1, -1, -1, 1, 1, 1, 1, 1))

y <- h(u, v)
y$backward()

u$grad
```

```
torch_tensor
-12
-12
-12
-12
-12
 12
 12
 12
 12
 12
[ CPUFloatType{10} ]
```

Yes, the answer does match what was expected

1.3 (5 points)

Consider the following function

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

Derive the expression for

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

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

$$ f'(z_0) = 4z^3 - 12z - 3 \

f'(-3.5) = -132.5 $$

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

```r
library(torch)

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

z <- torch_tensor(-3.5, requires_grad = TRUE)
```

```
y <- f(z)

y$backward()
z$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.,

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

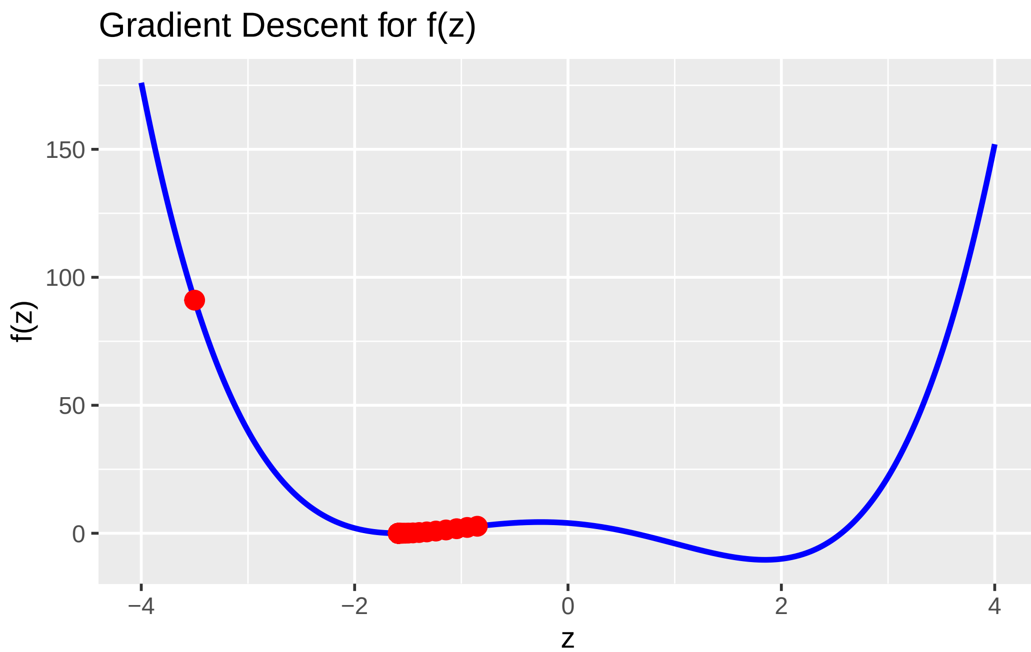
```
n <- 100
z <- -3.5
lr <- 0.02
zvals <- c(z)

for (i in 1:n) {
  df <- 4*z^3 - 12*z - 3
  z <- z - lr * df

  zvals <- c(zvals, z)
}
```

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?

```
xvals <- seq(-4, 4, by = 0.01)
yvals <- f(xvals)
df_f <- data.frame(x = xvals, y = yvals)
df_z <- data.frame(x = zvals, y = f(zvals))

ggplot() +
  geom_line(data = df_f, aes(x=x, y=y), color = "blue", size = 1) +
  geom_point(data = df_z, aes(x=x, y=y), color = "red", size = 3) +
  ggtitle("Gradient Descent for f(z)") +
```

```r
  xlab("z") +
  ylab("f(z)")
```

```
Warning: Using `size` aesthetic for lines was deprecated in ggplot2 3.4.0.
i Please use `linewidth` instead.
```



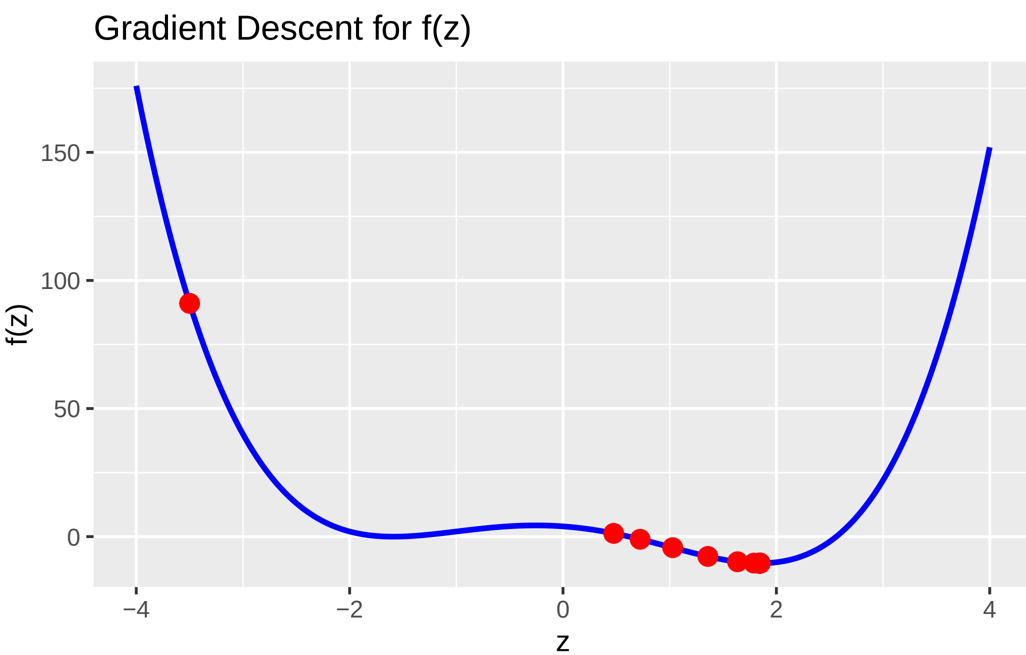
I can observe that gradient descent is not properly converging at the global minimum. This is most likely stemming from the learning rate.

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

```r
n <- 100
z <- -3.5
lr <- 0.03
zvals <- c(z)
```

```r
for (i in 1:n) {
  df <- 4*z^3 - 12*z - 3
  z <- z - lr * df

  zvals <- c(zvals, z)
}

xvals <- seq(-4, 4, by = 0.01)
yvals <- f(xvals)
df_f <- data.frame(x = xvals, y = yvals)
df_z <- data.frame(x = zvals, y = f(zvals))

ggplot() +
  geom_line(data = df_f, aes(x=x, y=y), color = "blue", size = 1) +
  geom_point(data = df_z, aes(x=x, y=y), color = "red", size = 3) +
  ggtitle("Gradient Descent for f(z)") +
  xlab("z") +
  ylab("f(z)")
```



I can observe that the gradient descent converges at the global minimum instead of a suboptimal local minimum. I can conclude that one most use the optimal learning rate in order for gradient descent to properly converge at the global minimum of a non-convex function.

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)
```

```
`curl` package not installed, falling back to using `url()`
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 <-
  df %>%
  mutate(Survived = as.factor(Survived),
         Sex = as.factor(Sex),
         # converting Pclass to factor because it is the passenger class
         # therefore it is categorical and must be converted
```

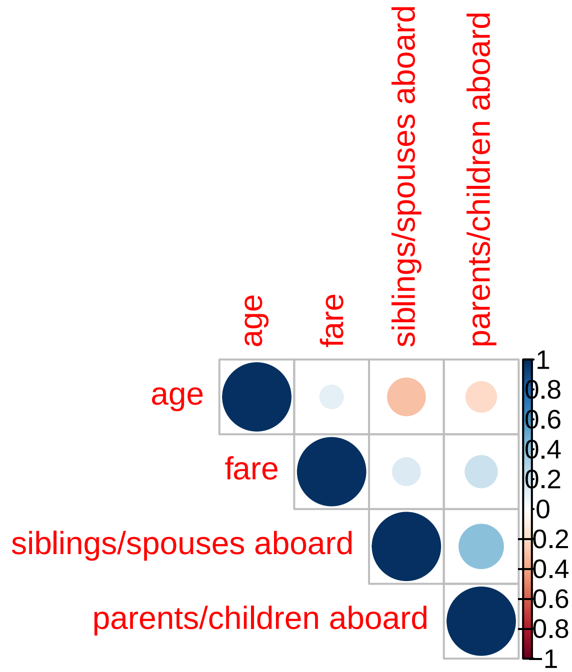
```r
         Pclass = as.factor(Pclass)) %>%
  rename("y" = Survived)

names(df) <- tolower(names(df))


head(df)
```

```
# A tibble: 6 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. Ow~ male     22                      1                      0
2 1     1      Mrs. J~ fema~    38                      1                      0
3 1     3      Miss. ~ fema~    26                      0                      0
4 1     1      Mrs. J~ fema~    35                      1                      0
5 0     3      Mr. Wi~ male     35                      0                      0
6 0     3      Mr. Ja~ male     27                      0                      0
# 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()`

```r
df %>%
  keep(is.numeric) %>%
  cor() %>%
  corrplot(type = "upper", order = "hclust")
```



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`

```
df <-
  df %>%
  select(!name)

full_model <- glm(y ~ ., df, family = binomial())
summary(full_model)
```

```
Call:
```

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

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.7773  -0.5991  -0.3984   0.6131   2.4412

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).

The intercept term represents the log-odds of survival when all other variables are set to 0. The slope represents the change in log-odds with a 1 unit change in the predictor variable. The odds-ratio (if the covariate is also categorical) represents the ratio of the odds of the outcome occurring in one group compared to the odds of the outcome occurring in a different group.

> 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){
    total_false_positives <- sum(predicted != expected & predicted == 1)
    total_true_positives <- sum(predicted == expected & expected == 1)
    total_false_negatives <- sum(predicted != expected & predicted == 0)
    total_true_negatives <- sum(predicted == expected & expected == 0)
    false_positive_rate <- total_false_positives / (total_false_positives +
                                                     total_true_negatives)
    false_negative_rate <- total_false_negatives / (total_false_negatives +
                                                     total_true_positives)
    accuracy <- (total_true_positives + total_true_negatives) /
      length(predicted)
    error <- 1- accuracy
    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

```r
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`

```r
# predicting the full_model
full_predictions = predict(full_model, type = "response")

full_predictions <- ifelse(full_predictions >= 0.5, 1, 0)

# setting the expected variables with the true values
expected <- df$y

full_overview <- overview(full_predictions, expected)
full_overview
```

```
   accuracy     error false_positive_rate false_negative_rate
1 0.8027057 0.1972943           0.1321101           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", scope=formula(full_model))
```

```
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)

Deviance Residuals:
```

```
    Min       1Q   Median       3Q      Max
-2.7637  -0.5883  -0.3930   0.6136   2.4543

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
# creating the prediction variables
step_predictions <- predict(step_model, type = "response")

step_predictions <- ifelse(step_predictions >= 0.5, 1, 0)

# setting the expected variables
expected <- df$y

step_overview <- overview(step_predictions, expected)
step_overview
```

```
   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)
```

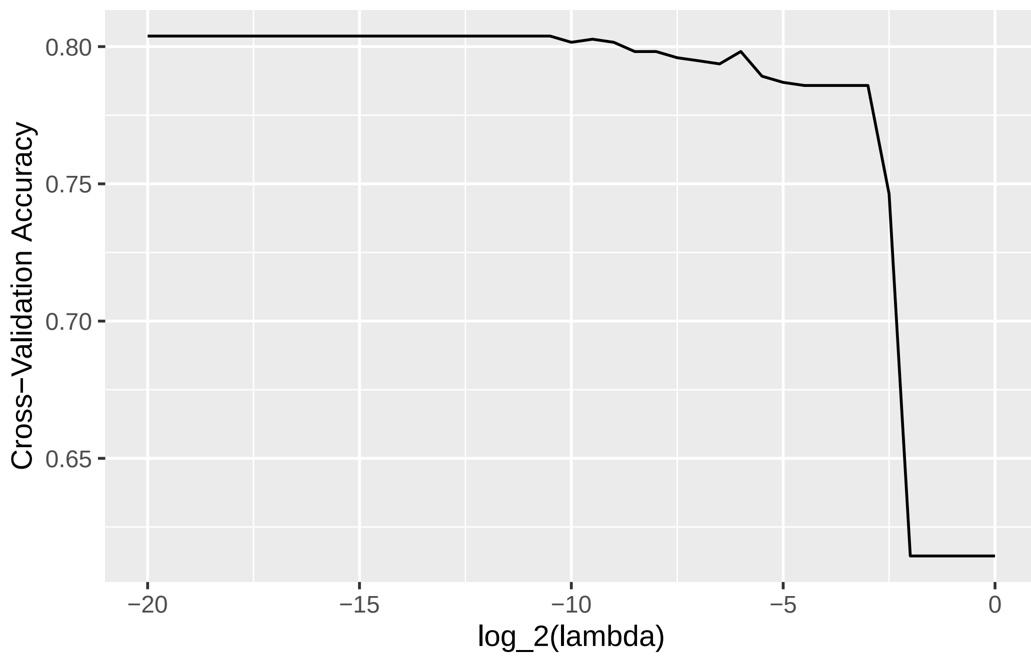
Now, using `control`, perform 5-fold cross validation using `caret::train()` to select the optimal λ parameter for LASSO with logistic regression.

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

```r
# Insert your code in the ... region
lasso_fit <- train(
  y ~ .,
  data = df,
  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
ggplot(data = lasso_fit$results, aes(x = log2(lambda), y = Accuracy)) +
  geom_line() +
  xlab("log_2(lambda)") +
  ylab("Cross-Validation Accuracy")
```




```
# creating optmal lambda variable
optimal_lambda <- lasso_fit$results$lambda[which.max(lasso_fit$results$Accuracy)]
optimal_accuracy <- max(lasso_fit$results$Accuracy)

paste0("The optimal lambda is ", optimal_lambda)
```

```
[1] "The optimal lambda is 9.5367431640625e-07"
```

```
paste0("The optimal accuracy is ", optimal_accuracy)
```

```
[1] "The optimal accuracy is 0.803827842315749"
```

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

```r
X <- torch_tensor(covariate_matrix, dtype = torch_float())
y <- torch_tensor(df$y, dtype = torch_float())
```

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()
  }
)

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
 0.9989
 1.0000
 0.9997
 1.0000
 1.0000
 0.9997
 1.0000
 0.9687
 0.9998
 0.9996
 0.9640
 1.0000
 0.9980
 1.0000
 0.9915
 1.0000
```

```
 0.9900
 0.9996
 1.0000
 0.9990
 1.0000
 1.0000
 0.9937
 1.0000
 0.9948
 1.0000
 0.9996
 1.0000
 0.9994
 0.9991
... [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){
  nn_bce_loss()(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`.

```
f <- logistic()
optimizer <- optim_adam(f$parameters, lr = 0.0001)

n <- 1000

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

  optimizer$zero_grad()
  loss$backward()
  optimizer$step()

  if(i %% 100 == 0){
    cat(sprintf("Step %d, Loss = %.4f\n", i, loss))
  }
```

```
}
```

```
Step 100, Loss = 4.2683
Step 200, Loss = 3.6286
Step 300, Loss = 3.0962
Step 400, Loss = 2.5810
Step 500, Loss = 2.0810
Step 600, Loss = 1.4116
Step 700, Loss = 0.6015
Step 800, Loss = 0.0777
Step 900, Loss = -0.5172
Step 1000, Loss = -1.2798
```

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)

torch_overview <- overview(torch_predictions, df$y)
torch_overview
```

```
   accuracy     error false_positive_rate false_negative_rate
1 0.3461105 0.6538895           0.8972477           0.2660819
```

```
# creating the lasso regression overview
lasso_prediction <- predict(lasso_fit)

lasso_overview <- overview(lasso_prediction, df$y)
lasso_overview
```

```
  accuracy    error false_positive_rate false_negative_rate
1 0.800451 0.199549            0.133945           0.3040936
```

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
name <- c("full_overview", "step_overview", "torch_overview", "lasso_overview")

all_overviews <-
  rbind(full_overview, step_overview, torch_overview, lasso_overview) %>%
  cbind(name) %>%
  select(name, accuracy, error, false_positive_rate, false_negative_rate)
all_overviews
```

```
            name  accuracy     error false_positive_rate false_negative_rate
1  full_overview 0.8027057 0.1972943           0.1321101           0.3011696
2  step_overview 0.8049605 0.1950395           0.1339450           0.2923977
3 torch_overview 0.3461105 0.6538895           0.8972477           0.2660819
4 lasso_overview 0.8004510 0.1995490           0.1339450           0.3040936
```

It seems that the backwards-stepwise logistic regression had the highest accuracy by a slight margin. Although it had the highest accuracy stepwise regression is not the wisest choice to use on massive datasets due to the shear computation intensity required to slowly reduce AIC through its method of feature selection. The full overview and lasso overview had similar accuracies and errors although using LASSO regression is much more reliable for very large datasets. The torch overview had the lowest accuracy due to its dependancy on an effective learning rate which can make it at times unreliable.

Session Information

Print your R session information using the following command

```r
sessionInfo()
```

```
R version 4.2.1 (2022-06-23 ucrt)
Platform: x86_64-w64-mingw32/x64 (64-bit)
Running under: Windows 10 x64 (build 22000)

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

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

other attached packages:
 [1] numDeriv_2016.8-1.1 broom_1.0.4         nnet_7.3-18
 [4] torch_0.9.1         caret_6.0-94        lattice_0.20-45
 [7] ggplot2_3.4.1       car_3.1-2           carData_3.0-5
[10] corrplot_0.92       stringr_1.5.0       purrr_1.0.1
[13] tidyr_1.3.0         readr_2.1.4         dplyr_1.1.1

loaded via a namespace (and not attached):
 [1] nlme_3.1-157         lubridate_1.9.2      bit64_4.0.5
 [4] tools_4.2.1          backports_1.4.1      utf8_1.2.3
 [7] R6_2.5.1             rpart_4.1.16         colorspace_2.1-0
[10] withr_2.5.0          tidyselect_1.2.0     processx_3.8.0
[13] bit_4.0.5            compiler_4.2.1       glmnet_4.1-7
[16] cli_3.6.1            labeling_0.4.2       scales_1.2.1
[19] proxy_0.4-27         callr_3.7.3          digest_0.6.31
[22] rmarkdown_2.21       coro_1.0.3           pkgconfig_2.0.3
[25] htmltools_0.5.5      parallelly_1.35.0    fastmap_1.1.1
[28] rlang_1.1.0          shape_1.4.6          generics_0.1.3
```

```
[31] farver_2.1.1         jsonlite_1.8.4       vroom_1.6.1
[34] ModelMetrics_1.2.2.2 magrittr_2.0.3       Matrix_1.4-1
[37] Rcpp_1.0.10          munsell_0.5.0        fansi_1.0.4
[40] abind_1.4-5          lifecycle_1.0.3      stringi_1.7.12
[43] pROC_1.18.0          yaml_2.3.7           MASS_7.3-57
[46] plyr_1.8.8           recipes_1.0.5        grid_4.2.1
[49] parallel_4.2.1       listenv_0.9.0        crayon_1.5.2
[52] splines_4.2.1        hms_1.1.3            knitr_1.42
[55] ps_1.7.3             pillar_1.9.0         future.apply_1.10.0
[58] reshape2_1.4.4       codetools_0.2-18     stats4_4.2.1
[61] glue_1.6.2           evaluate_0.20        data.table_1.14.8
[64] renv_0.16.0-53       vctrs_0.6.1          tzdb_0.3.0
[67] foreach_1.5.2        gtable_0.3.3         future_1.32.0
[70] xfun_0.38            gower_1.0.1          prodlim_2019.11.13
[73] e1071_1.7-13         class_7.3-20         survival_3.3-1
[76] timeDate_4022.108    tibble_3.2.1         iterators_1.0.14
[79] hardhat_1.2.0        lava_1.7.2.1         timechange_0.2.0
[82] globals_0.16.2       ipred_0.9-14
```