Homework 4

AUTHOR
Alvaro Tapia

[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:

```
library(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
```

```
library(readr)
library(tidyr)
library(purrr)
library(stringr)
library(corrplot)
```

```
corrplot 0.92 loaded
```

```
library(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
```

```
library(caret)
```

```
Loading required package: ggplot2

Loading required package: lattice

Warning in system("timedatectl", intern = TRUE): running command 'timedatectl'
had status 1


Attaching package: 'caret'

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

    lift
```

```
library(torch)
library(nnet)
library(broom)

packages <- c(
  "dplyr",
  "readr",
  "tidyr",
  "purrr",
  "stringr",
  "corrplot",
  "car",
  "caret",
  "torch",
  "nnet",
  "broom"
)

# renv::install(packages)
sapply(packages, require, character.only=T)
```

```
   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

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

Using your answer from above, what is the answer to

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

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?

```
library(numDeriv)
x <- c(3, 4)
g <- \(x) {
  (x[1] - 3)^2 + (x[2] - 4)^2
}
grad(g, x)
```

```
[1] 0 0
```

Yes, for this case the answer does match with the expected answer which is 0,0.

1.2 (10 points)

Consider $h(\boldsymbol{u}, \boldsymbol{v})$ given by

$$h(\boldsymbol{u}, \boldsymbol{v}) = (\boldsymbol{u} \cdot \boldsymbol{v})^3,$$

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

Using elementary calculus derive the expressions for the gradients

$$\nabla_{\boldsymbol{u}} h(\boldsymbol{u}, \boldsymbol{v}) = \left(\frac{d}{du_1} h(\boldsymbol{u}, \boldsymbol{v}), \frac{d}{du_2} h(\boldsymbol{u}, \boldsymbol{v}), \ldots, \frac{d}{du_n} h(\boldsymbol{u}, \boldsymbol{v}) \right)$$

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

$$\boldsymbol{u} = (-1, +1, -1, +1, -1, +1, -1, +1, -1, +1)$$
$$\boldsymbol{v} = (-1, -1, -1, -1, -1, +1, +1, +1, +1, +1)$$

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

```r
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), requires_grad = TRUE)

h <- function(u,v){
  (torch_dot(u,v))^3
}

ans <- h(u, v)
ans$backward()
u$grad
```

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

Yes, the answer matches as expected: torch_tensor -12 -12 -12 -12 -12 12 12 12 12 12

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 $z_0 = -3.5$.

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

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

torched_z <- torch_tensor(-3.5, requires_grad = TRUE)
ans <- f(torched_z)
ans$backward()
torched_z$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.,

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

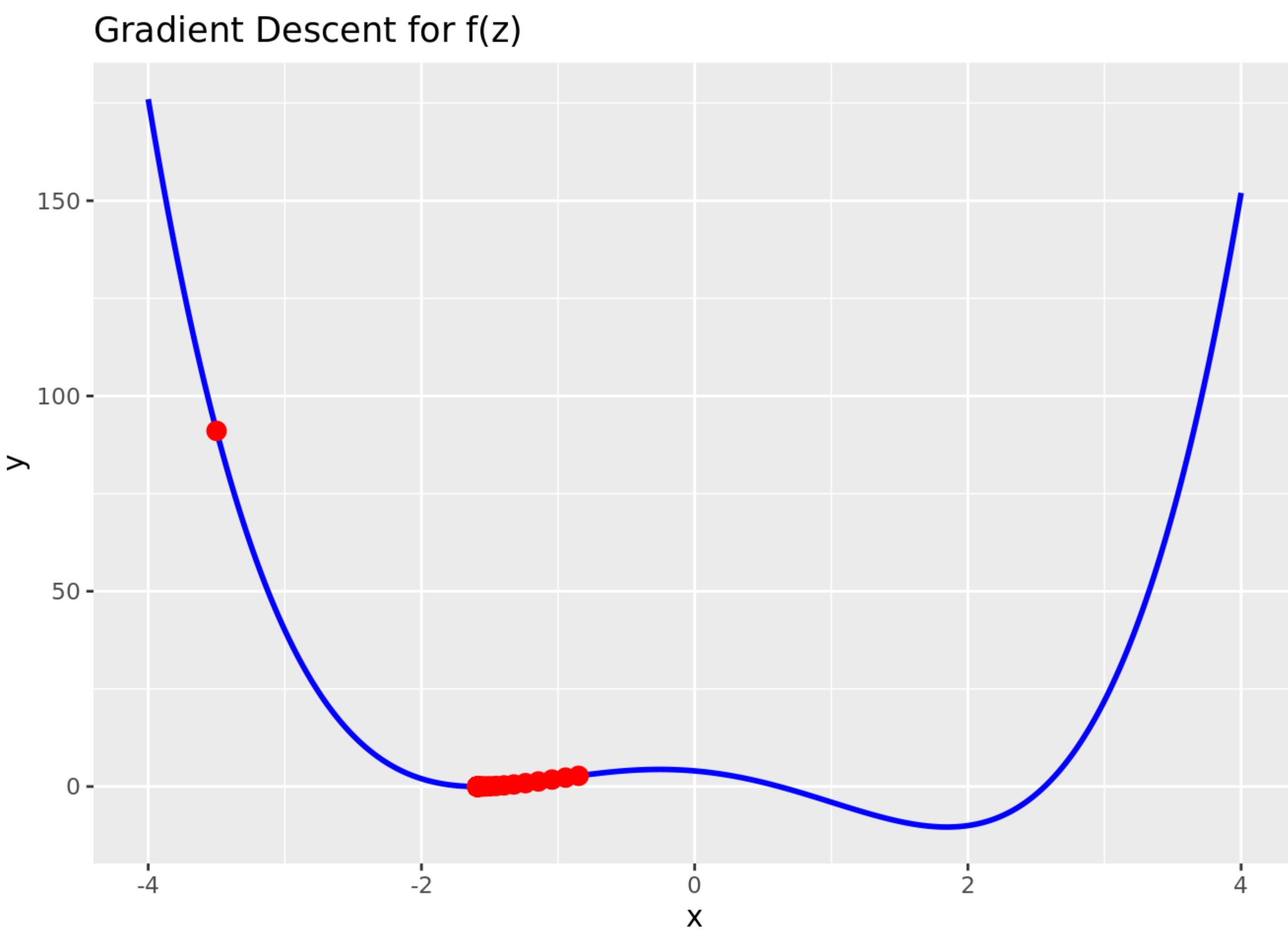
```
z <- -3.5
n <- 100
eta <- 0.02
z_values <- c(z)
for(i in 1:n){
  df <- 4*z^3 - 12*z - 3
  z <- z - eta * df
  z_values <- c(z_values, z)
}
```

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?

```
x_values <- seq(-4, 4, by = 0.01)
y_values <- f(x_values)
df_f <- data.frame(x = x_values, y = y_values)
df_z <- data.frame(x = z_values, y = f(z_values))
ggplot() +
  geom_line(data = df_f, aes(x, y), color = "blue", size = 1) +
```

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

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



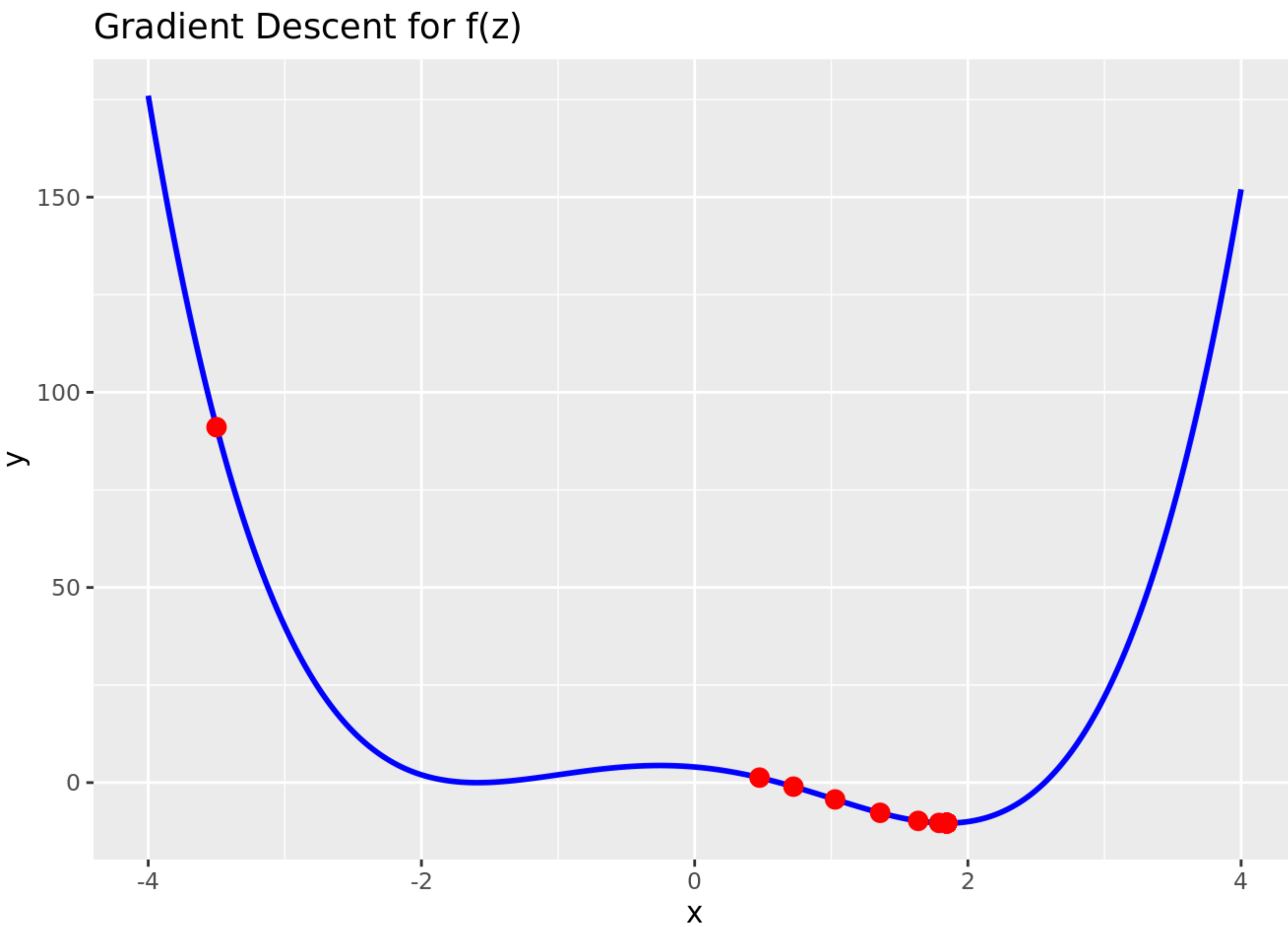
Here it is possible to identify a graph showing the gradient descent for f(z) thus showing the curve function with red dots that simbolize the values of z on each iteration. Here it is not totally converging to the global minimum but it is getting there. Right now they are around the local minimum.

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

```
z <- -3.5
n <- 100
eta <- 0.03
z_values <- c(z)
for(i in 1:n){
  df <- 4*z^3 - 12*z - 3
  z <- z - eta * df
  z_values <- c(z_values, z)
}
```

```
x_values <- seq(-4, 4, by = 0.01)
y_values <- f(x_values)
df_f <- data.frame(x = x_values, y = y_values)
df_z <- data.frame(x = z_values, y = f(z_values))
ggplot() +
  geom_line(data = df_f, aes(x, y), color = "blue", size = 1) +
  geom_point(data = df_z, aes(x, y), color = "red", size = 3) +
  ggtitle("Gradient Descent for f(z)")
```



Here it is possible to visualize that now the gradient descent does converge to the global minimum, as we can see, the red dots which are categorized as the values of iteration of z went from the local minimum to the global minimum. This changed happened due to a variation in the learning rate from 0.02 to 0.03 which we can say that it is a good choice of learning rate for the performance of the gradient descent.

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)

df$Survived = as.factor(df$Survived)
df$Sex = as.factor(df$Sex)

colnames(df)[colnames(df) == 'Survived'] <- 'y'
colnames(df) <- tolower(colnames(df))
head(df)
```

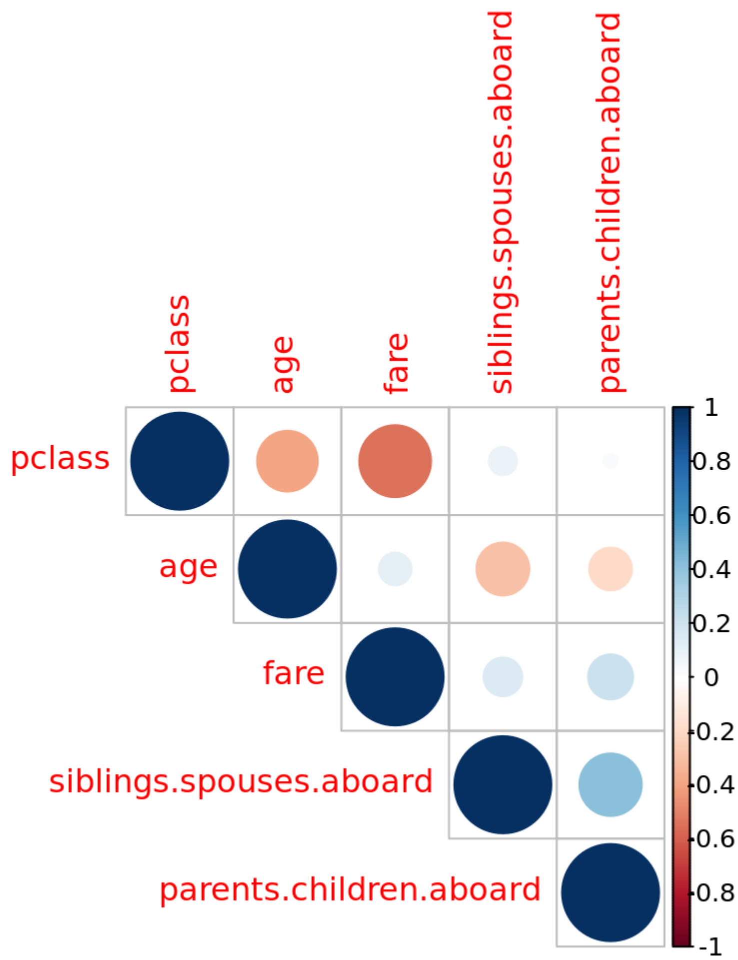
```
  y pclass                                                name    sex age
1 0      3                              Mr. Owen Harris Braund   male  22
2 1      1 Mrs. John Bradley (Florence Briggs Thayer) Cumings female  38
3 1      3                               Miss. Laina Heikkinen female  26
4 1      1         Mrs. Jacques Heath (Lily May Peel) Futrelle female  35
5 0      3                             Mr. William Henry Allen   male  35
6 0      3                                     Mr. James Moran   male  27
  siblings.spouses.aboard parents.children.aboard    fare
1                       1                       0  7.2500
2                       1                       0 71.2833
3                       0                       0  7.9250
4                       1                       0 53.1000
5                       0                       0  8.0500
6                       0                       0  8.4583
```

2.2 (5 points)

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

```
df %>%
  select(where(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`

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

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

```
Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.7789  -0.5976  -0.3987   0.6156   2.4409

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).

> 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).

We learned that in logistic regression the intercept term corresponds to the log-odds of survival when all other covariates are set to 0. In this sense, since in this model we can see that the intercept term of 5.297252, what this shows is that when all other covariates are held constant, the log-odds of survival is 5.297252. In regard to the slope, this shows how the log-odds of survival for a one-unit increase in the corresponding covariate, holding all other covariates constant so for example, a one-unit increase in pclass decreases the log-odds of survival by 1.177659, and a one-unit increase in age decreases the log-odds of survival by 0.043474. For the categorical values of the covariates such as sex, we have that the coefficient estimate of -2.757282 implies that being male (compared to female) decreases the log-odds of survival by 2.757282 when all other covariates are held constant. In regard to the odds-ratios, it is possible to identify that the odds of survival for a female is exp(-2.757282) = 0.0636896 times the odds of survival for a male, holding all other covariates constant.

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 == "positive") & (expected == "negative"))
    total_true_positives <- sum((predicted == "positive") & (expected == "negative"))
    total_false_negatives <- sum((predicted == "negative") & (expected == "negative"))
    total_true_negatives <- sum((predicted == "negative") & (expected == "negative"))
    false_positive_rate <- total_false_positives / (total_false_positives + total_true_negatives)
    false_negative_rate <- total_false_negatives / (total_false_negatives + total_true_positives)
    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                 NaN                 NaN
```

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

As it was expected, the accuracy is 100% and errors are 0%.

3.2 (5 points)

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

```
fmodel_prob <- predict(full_model, type = "response")
fmodel_pred <- ifelse(fmodel_prob >= 0.5, 1, 0)
full_model_overview <- overview(fmodel_pred, df$y) #Using true values of expected variables
full_model_overview
```

```
   accuracy     error false_positive_rate false_negative_rate
1 0.8015784 0.1984216                 NaN                 NaN
```

3.3 (5 points)

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

```
model_step <- 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(model_step)
```

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

Deviance Residuals: 
    Min       1Q   Median       3Q      Max  
-2.7548  -0.5987  -0.3917   0.6143   2.4562  

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
model_step <- 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
```

```
summary(model_step)
```

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

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.7548  -0.5987  -0.3917   0.6143   2.4562

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_pred <- predict(model_step, type = "response")
step_pred <- ifelse(step_pred >= 0.5, 1, 0)
step_overview <- overview(step_pred, df$y)
step_overview
```

```
   accuracy     error false_positive_rate false_negative_rate
1 0.8049605 0.1950395                 NaN                 NaN
```

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)
```

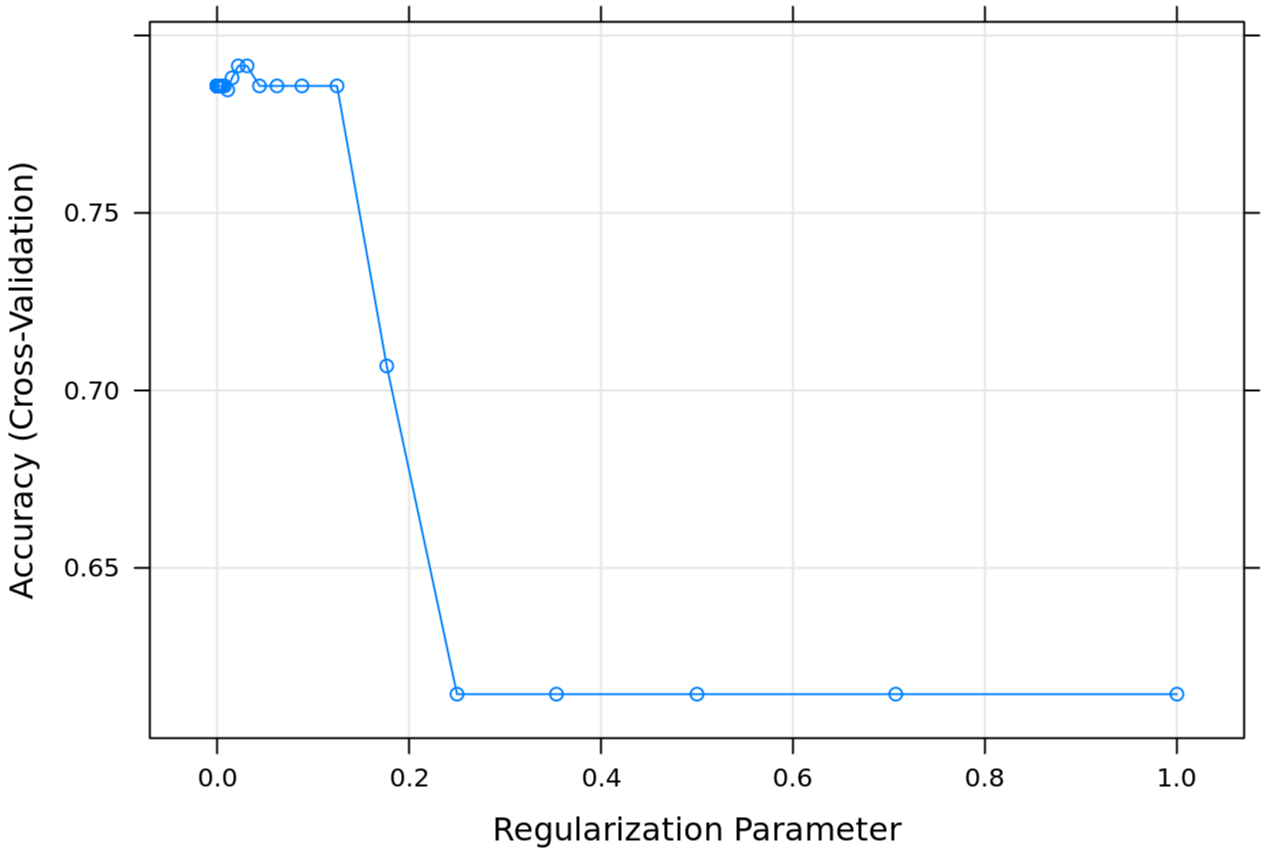
Now, using `control`, perform 5-fold cross validation using `caret::train()` to select the optimal λ parameter for LASSO with logistic regression.

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

```r
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
plot(lasso_fit) #Plotting
```




```
optimal_lambda <- lasso_fit$results$lambda[which.max(lasso_fit$results$Accuracy)]
paste0("Optimal lambda: ", optimal_lambda)
```

```
[1] "Optimal lambda: 0.03125"
```

```
optimal_accuracy <- max(lasso_fit$results$Accuracy)
paste0("Optimal accuracy: ", optimal_accuracy)
```

```
[1] "Optimal accuracy: 0.791436551767917"
```

3.5 (25 points)

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

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

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

```
X <- torch_tensor(covariated, 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)

```
logistic <- nn_module(
  initialize = function() {
    self$f <- nn_linear(6, 1) #6 covariates
    self$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:

```
f(X)
```

```
torch_tensor
 0.8474
 1.0000
 0.8082
 1.0000
 0.8473
 0.8504
 1.0000
 0.9979
 0.9383
 0.9996
 0.9842
 0.9994
 0.8253
 0.9999
 0.7842
 0.9852
 0.9998
 0.9606
 0.9897
 0.7697
 0.9989
 0.9648
 0.7939
 0.9999
 0.9977
 0.9999
 0.8013
```

```
 1.0000
 0.8029
 0.8240
... [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 = 0.01)
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 = -29.9743
Step 200, Loss = -35.1276
Step 300, Loss = -36.8022
Step 400, Loss = -37.3010
Step 500, Loss = -37.6891
Step 600, Loss = -37.9778
Step 700, Loss = -38.0757
Step 800, Loss = -38.0782
Step 900, Loss = -38.1749
Step 1000, Loss = -38.2708
```

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 0.3855693 0.6144307                 NaN                 NaN
```

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.

```
#Creating lasso overview for this
lasso_predictions <- predict(lasso_fit)
lasso_overview<- overview(lasso_predictions, df$y)
lasso_overview
```

```
   accuracy     error false_positive_rate false_negative_rate
1 0.7857948 0.2142052                 NaN                 NaN
```

```
library(knitr)
all_overviews <- bind_rows(
  full_model_overview,
  step_overview,
  lasso_overview,
  torch_overview
) %>%
  mutate(Model = c("Full_model Overview", "Step Overview", "Lasso Overview", "Torch Overview")) %
  select(Model, accuracy, error, false_positive_rate, false_negative_rate)

kable(all_overviews, caption = "Summary table of Overviews", align = "c")
```

Summary table of Overviews

Model	accuracy	error	false_positive_rate	false_negative_rate
Full_model Overview	0.8015784	0.1984216	NaN	NaN
Step Overview	0.8049605	0.1950395	NaN	NaN
Lasso Overview	0.7857948	0.2142052	NaN	NaN
Torch Overview	0.3855693	0.6144307	NaN	NaN

From this table we can conclude many different things. First, that the Full model overview and the stepwise overview have the highest accuracy over all the other models except for the lasso overview which is kind of close to the accurarcy of the previously mentioned models. However, even though they have the highest accuracy rate, it is important to know that the stepwise model could act poorly if it is used a very much larger dataset due to its computation that slows AIC. Then, in regard to the torch overview, it is possible to say that it has the lowes accuracy rate from all of them, this because it is highly dependend of the effectiveness of the learning rate that can increase its error rate.

Session Information