Linear Models in R (M1–MIDO)

Lab Session 5 — Solutions

Henri PANJO

Table of contents

Setup

To keep numbers readable and reproducible, we set display options:

```
options(scipen = 999, digits = 5)
```

We also load the packages used during this session.

> ⚠ Warning
>
> Don’t worry if you don’t know them all — we’ll introduce functions as we need them. Some provide regression tools, others are for data visualization or diagnostics.

```
library(broom)
library(performance)
library(parameters)
library(datawizard)
library(see)
library(effectsize)
library(insight)
library(correlation)
library(modelbased)
library(glue)
library(scales)
library(GGally)
library(ggpubr)
library(car)
library(lmtest)
library(multcomp)
library(rstatix)
library(matrixTests)
library(ggfortify)
library(qqplotr)
library(patchwork)
library(ggrepel)
library(gtsummary)
library(kableExtra)
library(openxlsx)
library(janitor)
library(marginaleffects)
library(pROC)
library(caret)
library(collapse)
library(tidyverse)
```

```
source("helper_functions5.R")
```

Data import

The file `data05.csv` contains data on binge drinking among 2951 adolescents aged 13 to 19 (`age`), as well as the following variables/predictors: an indicator showing whether the teenager has a friend who drinks alcohol (`friendalc`), sensation seeking on a scale from 5 to 20 (`sensation`), watching movies with alcohol on a scale from 0 to 10 (`filmalc`) and parental supervision in category (`parentsurv`).

The variable `binge` is the dependant variable that indicates if the adolescent does binge drinking.

- We import the data

```r
base <- read_csv("data05.csv", show_col_types = FALSE) |>
  mutate(friendalc = factor(friendalc, levels = c("No", "Yes"))) |>
  mutate(binge = factor(binge, levels = c("No", "Yes"))) |>
  mutate(binge_bin = 1 * (binge == "Yes"), .after = binge) |>
  mutate(parentsurv = factor(parentsurv, levels = c("Low", "Medium", "High"))) |>
  relabel(
    friendalc = "Has a drinking friend", parentsurv = "Parental supervision",
    binge = "Binge drinking", age = "Age (years)", sensation = "Sensation seeking score",
    filmalc = "Movies with alcohol score"
  )
```

```r
base
```

```
# A tibble: 2,951 x 8
      id binge binge_bin friendalc sensation   age filmalc parentsurv
   <dbl> <fct>     <dbl> <fct>         <dbl> <dbl>   <dbl> <fct>
 1     1 No            0 Yes              15    15   1.56  Low
 2     2 No            0 Yes              10    15   0.941 High
 3     3 No            0 No               15    14   0.808 High
 4     4 Yes           1 Yes              18    16   0.892 Low
 5     5 Yes           1 Yes              13    18   1.94  Low
 6     6 Yes           1 Yes              12    18   6.89  High
 7     7 No            0 Yes              16    14   1.31  High
 8     8 Yes           1 Yes              11    17   2.24  Low
 9     9 No            0 Yes              11    15   2.41  High
10    10 No            0 No                5    14   0.290 High
# i 2,941 more rows
```

- Distribution of categorical variables with `tab_freq1()` from `helper_functions5.R`

```
tab_freq1(base, c("binge", "friendalc", "parentsurv"), digits = 1) |>
  kable(align = "l", padding = 2) |>
  row_spec(c(1, 4, 7), bold = TRUE)
```

Variable	Count (n)	Percent (%)
Binge drinking		
No	2173	73.6%
Yes	778	26.4%
Has a drinking friend		
No	699	23.7%
Yes	2252	76.3%
Parental supervision		
Low	843	28.6%
Medium	975	33.0%
High	1133	38.4%

- Summary of continuous variables with `tbl_summary()` from `{gtsummary}`

```
tab_summary <- select(base, age, sensation, filmalc) |>
  tbl_summary(
    type = list(age ~ "continuous2", all_continuous() ~ "continuous2"),
    statistic = all_continuous() ~ c(
      "{mean} ({sd})", "{median} ({p25}, {p75})", "{min}, {max}"
    ),
    digits = ~ 1
  ) |>
  bold_labels() |>
  as_kable_extra(booktabs = TRUE, linesep = "") |>
  kable_styling(position = "center", latex_options = "HOLD_position")
```

```
tab_summary
```

Characteristic	**N = 2,951**
Age (years)	
Mean (SD)	15.7 (1.4)
Median (Q1, Q3)	16.0 (15.0, 17.0)
Min, Max	13.0, 19.0
Sensation seeking score	
Mean (SD)	12.0 (2.9)
Median (Q1, Q3)	12.0 (10.0, 14.0)
Min, Max	5.0, 20.0
Movies with alcohol score	
Mean (SD)	2.8 (1.6)
Median (Q1, Q3)	2.7 (1.6, 3.9)
Min, Max	0.0, 10.0

Question 1: Exploratory analysis of binge drinking

1. Compute the proportion of binge drinking with respect to the variable `friendalc` and `parentsurv`
2. Compute the mean of the continous variables with respect to the binge drinking status

Solution

- We use the variable `binge_bin` inside `percent_by_group()` (`helper_functions5.R`) to get the percentage of binge drinking for each group

```r
c("friendalc", "parentsurv") |>
  map_dfr(\(by) percent_by_group(base, "binge_bin", by, digits = 1)) |>
  rename("Binge drinking (%)" = Percentage) |>
  kable(align = "l", padding = 2) |>
  row_spec(c(1, 4), bold = TRUE)
```

Variable	N	Binge drinking (%)
Has a drinking friend		
No	699	2.3%
Yes	2252	33.8%
Parental supervision		
Low	843	40.6%
Medium	975	25.2%
High	1133	16.8%

- To compute the mean of the continous variables with respect to the binge drinking status, we use `tbl_summary()`

```r
tab_summary2 <- select(base, binge, age, sensation, filmalc) |>
  tbl_summary(
    by = binge,
    type = list(age ~ "continuous", all_continuous() ~ "continuous"),
    statistic = all_continuous() ~ c("{mean} ({sd})"),
    digits = ~1
  ) |>
  bold_labels() |>
  remove_footnote_header() |>
  as_kable_extra(booktabs = TRUE, linesep = "") |>
  kable_styling(position = "center", latex_options = "HOLD_position")
```

```r
tab_summary2
```

Characteristic	No	Yes
	N = 2,173	N = 778
Age (years)	15.5 (1.4)	16.5 (1.2)
Sensation seeking score	11.4 (2.8)	13.7 (2.6)
Movies with alcohol score	2.6 (1.6)	3.5 (1.6)

Question 2: Spliting the data

Create a training database `data_train` (approximately 70% of `base`) and a test database `data_test` (approximately 30% of `base`).

Solution

- We create a binary variable `tag` that take the value 1 around 70% of the time and 0 around 30% of the time

```
set.seed(123) # for reproductibility

base <- base |>
  mutate(tag = rbinom(n(), size = 1, prob = 0.70))
```

- We check the distribution of `tag`

```
tabyl(base, tag)
```

```
 tag    n percent
   0  884 0.29956
   1 2067 0.70044
```

- We create `data_train` and `data_test`

```
data_train <- filter(base, tag == 1)
nrow(data_train)
```

```
[1] 2067
```

```
data_test <- filter(base, tag == 0)
nrow(data_test)
```

```
[1] 884
```

Generalized linear model refresher

Model components

1. Random part :
   distribution of the response y, derived from the exponential family

2. Fixed (systematic), linear predictor:
   linear dependence of explanatory variables, $\mathbf{x}^t\boldsymbol{\beta} = \sum_{j=0}^{p} \beta_j x_j$

3. The link function, generally *canonical*:
   function $g(.)$ that links $E(y|\mathbf{x}) = \mu$ to $\mathbf{x}^t\boldsymbol{\beta}$: $g\left(\mu\right) = \mathbf{x}^t\boldsymbol{\beta}$

How do I return to the data scale ?

$$g\left(\mu\right) = \beta_0 + \beta_1 x_1 + \cdots + \beta_p x_p = \mathbf{x}^t\boldsymbol{\beta}$$

$$\mu = E(y|\mathbf{x}) = g^{-1}\left(\mathbf{x}^t\boldsymbol{\beta}\right)$$

$$g^{-1}(.) = \begin{cases} \dfrac{\exp(.)}{1+\exp(.)} & \textbf{if logistic model} \\ \exp(.) & \textbf{is Poisson model} \end{cases}$$

Question 3: Logistic model

- $y =$ `binge_bin` la réponse pour un étudiant (`binge_bin = 0 ou 1`)
- $\mathbb{E}(y \mid \mathbf{x}) = \mathbb{P}(y = 1 \mid \mathbf{x}) = p(\mathbf{x})$

1. Fit the folowing logistic model (`mod1`), interpret the output test the significance of each variables

$$\log\left[\frac{p(\mathbf{x})}{1-p(\mathbf{x})}\right] = \beta_0 + \beta_1\texttt{age} + \beta_2\texttt{sensation} + \beta_3\texttt{filmalc} + \\ \beta_4\texttt{parentsurv}_{\texttt{Medium}} + \beta_5\texttt{parentsurv}_{\texttt{High}} + \beta_6\texttt{friendalc}_{\texttt{Yes}}$$

Solution

- We fit the model with `glm()` using the variable `binge_bin`

```r
mod1 <- glm(
  binge_bin ~ age + sensation + filmalc + parentsurv + friendalc,
  data = data_train, family = binomial(link = "logit")
)

summary(mod1)
```

```
Call:
glm(formula = binge_bin ~ age + sensation + filmalc + parentsurv + 
    friendalc, family = binomial(link = "logit"), data = data_train)

Coefficients:
                 Estimate Std. Error z value             Pr(>|z|)    
(Intercept)      -13.9476     0.8968  -15.55 < 0.0000000000000002 ***
age                0.4833     0.0470   10.28 < 0.0000000000000002 ***
sensation          0.2662     0.0232   11.47 < 0.0000000000000002 ***
filmalc            0.2135     0.0367    5.82          0.000000006 ***
parentsurvMedium  -0.4265     0.1398   -3.05               0.0023 ** 
parentsurvHigh    -0.6620     0.1481   -4.47          0.000007858 ***
friendalcYes       1.6746     0.2994    5.59          0.000000022 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 2366.0  on 2066  degrees of freedom
Residual deviance: 1762.8  on 2060  degrees of freedom
AIC: 1777

Number of Fisher Scoring iterations: 6
```

- Output with `model_parameters()` in Log-Odds scale

```
model_parameters(mod1, ci_method = "wald", digits = 2)
```

```
Parameter            | Log-Odds |   SE |           95% CI |      z |      p
--------------------------------------------------------------------------
(Intercept)          |   -13.95 | 0.90 | [-15.71, -12.19] | -15.55 | < .001
age                  |     0.48 | 0.05 | [  0.39,   0.58] |  10.28 | < .001
sensation            |     0.27 | 0.02 | [  0.22,   0.31] |  11.47 | < .001
filmalc              |     0.21 | 0.04 | [  0.14,   0.29] |   5.82 | < .001
parentsurv [Medium] |    -0.43 | 0.14 | [ -0.70,  -0.15] |  -3.05 | 0.002
parentsurv [High]    |    -0.66 | 0.15 | [ -0.95,  -0.37] |  -4.47 | < .001
friendalc [Yes]      |     1.67 | 0.30 | [  1.09,   2.26] |   5.59 | < .001
```

Interpretation

- The effect of `age` is statistically significant and positive (beta = 0.48, 95% CI [0.39, 0.58], p < .001)
- The effect of `sensation` is statistically significant and positive (beta = 0.27, 95% CI [0.22, 0.31], p < .001)
- The effect of `filmalc` is statistically significant and positive (beta = 0.21, 95% CI [0.14, 0.29], p < .001)
- The effect of `parentsurv` [`Medium` vs `Low`] is statistically significant and negative (beta = -0.43, 95% CI [-0.70, -0.15], p = 0.002)
- The effect of `parentsurv` [`High` vs `Low`] is statistically significant and negative (beta = -0.66, 95% CI [-0.95, -0.37], p < .001)
- The effect of `friendalc` [`Yes` vs `No`] is statistically significant and positive (beta = 1.67, 95% CI [1.13, 2.31], p < .001)

- Output with `model_parameters()` in Odds Ratio scale

```
model_parameters(mod1, ci_method = "wald", digits = 2, exponentiate = TRUE)
```

```
Parameter            | Odds Ratio |       SE |       95% CI |      z |      p
----------------------------------------------------------------------------
(Intercept)          |   8.76e-07 | 7.86e-07 | [0.00, 0.00] | -15.55 | < .001
age                  |       1.62 |     0.08 | [1.48, 1.78] |  10.28 | < .001
sensation            |       1.31 |     0.03 | [1.25, 1.37] |  11.47 | < .001
filmalc              |       1.24 |     0.05 | [1.15, 1.33] |   5.82 | < .001
parentsurv [Medium] |       0.65 |     0.09 | [0.50, 0.86] |  -3.05 | 0.002
parentsurv [High]    |       0.52 |     0.08 | [0.39, 0.69] |  -4.47 | < .001
friendalc [Yes]      |       5.34 |     1.60 | [2.97, 9.60] |   5.59 | < .001
```

- `Anova()` to test all variables. This will perform the likelihood ratio test

```
Anova(mod1, type = 3)
```

```
Analysis of Deviance Table (Type III tests)

Response: binge_bin
           LR Chisq Df            Pr(>Chisq)    
age           116.2  1 < 0.0000000000000002 ***
sensation     148.8  1 < 0.0000000000000002 ***
filmalc        34.6  1         0.000000004137 ***
parentsurv     21.2  2         0.000024643763 ***
friendalc      45.7  1         0.000000000014 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

- `lrtest()` from `{lmtest}`. This will perform the likelihood ratio test on one variable

```
lrtest(mod1, "parentsurv")
```

```
Likelihood ratio test

Model 1: binge_bin ~ age + sensation + filmalc + parentsurv + friendalc
Model 2: binge_bin ~ age + sensation + filmalc + friendalc
  #Df LogLik Df Chisq Pr(>Chisq)    
1   7   -881                        
2   5   -892 -2  21.2   0.000025 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

- `waldtest()` from `{lmtest}`. This will perform the Wald test on one variable

```
waldtest(mod1, "parentsurv")
```

```
Wald test

Model 1: binge_bin ~ age + sensation + filmalc + parentsurv + friendalc
Model 2: binge_bin ~ age + sensation + filmalc + friendalc
  Res.Df Df    F   Pr(>F)    
1   2060                     
2   2062 -2 10.6 0.000026 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Question 4: Prediction grid and model-based probabilities

1. Using the fitted model, construct a new dataset (`grid1`) that contains **all combinations** of :

- `age` taking values from **13 to 19**,
- all observed levels of `parentsurv`,
- all observed levels of `friendalc`,
- `sensation` and `filmalc` at their means

2. Using `grid1`, compute the **expected probability** of `binge_bin = 1` implied by the fitted logistic model, together with 95% **confidence intervals**.

3. Produce a figure displaying the model-based predictions with the following characteristics:

- x-axis: `age`,
- y-axis: predicted probability of `binge_bin = 1`,
- color: `friendalc`,
- separate panels (facets) for `parentsurv`,
- include both points and connecting lines.

Solution

- Creation of `grid1` with **`get_datagrid()`** from **`{insight}`**

```r
grid1 <- select(data_train, find_predictors(mod1, flatten = TRUE)) |>
  get_datagrid(
    by = list(
      age = seq(13, 19, 0.5),
      parentsurv = levels(data_train$parentsurv),
      friendalc = levels(data_train$friendalc)
    ), numerics = "mean"
  )

print(as_tibble(grid1), n = 10)
```

```
# A tibble: 78 x 5
     age parentsurv friendalc sensation filmalc
   <dbl> <fct>      <fct>         <dbl>   <dbl>
 1  13   Low        No             12.0    2.81
 2  13.5 Low        No             12.0    2.81
 3  14   Low        No             12.0    2.81
 4  14.5 Low        No             12.0    2.81
 5  15   Low        No             12.0    2.81
 6  15.5 Low        No             12.0    2.81
 7  16   Low        No             12.0    2.81
 8  16.5 Low        No             12.0    2.81
 9  17   Low        No             12.0    2.81
10  17.5 Low        No             12.0    2.81
# i 68 more rows
```

- **Expected probability** of `binge_bin = 1`

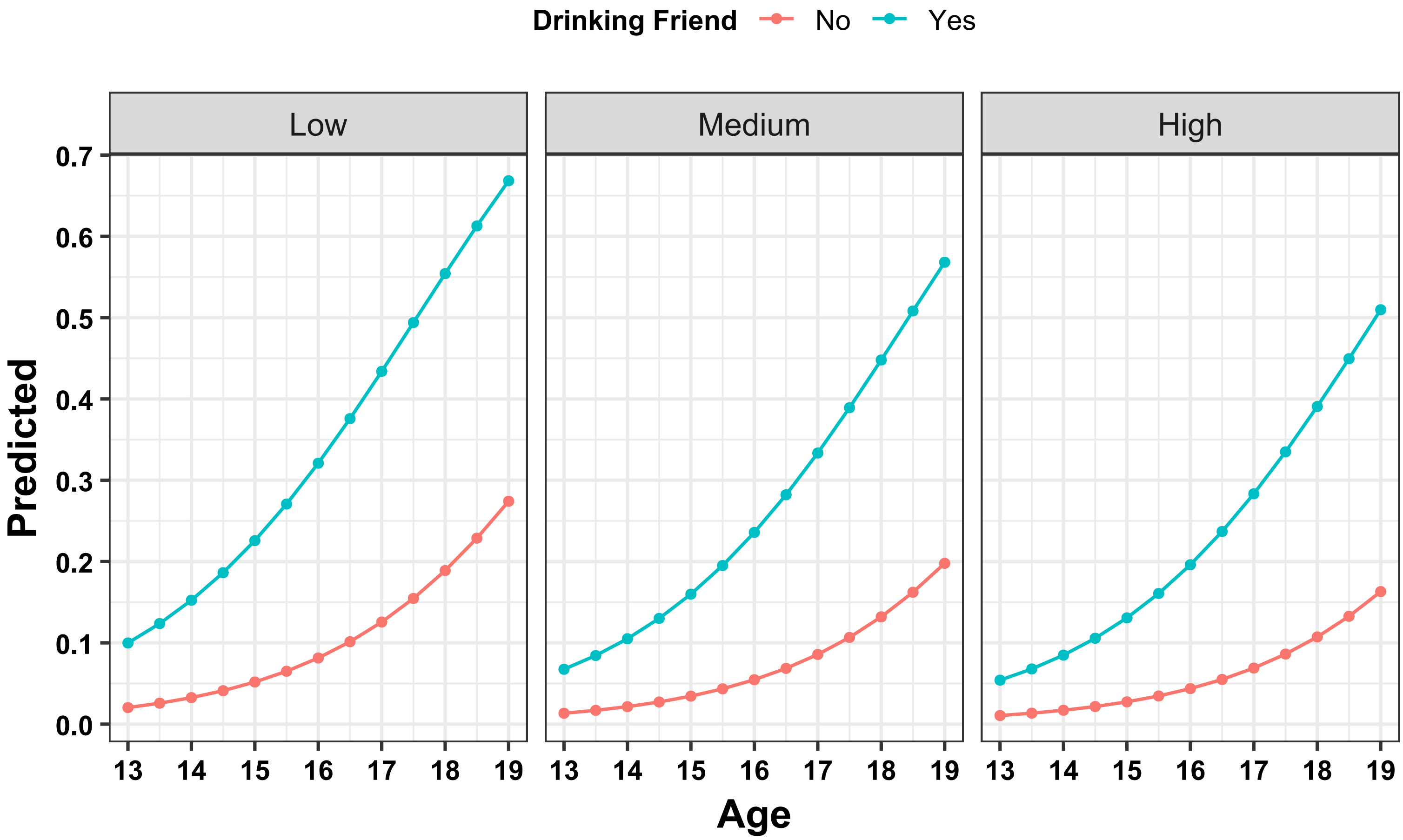
```
expect_grid1 <- estimate_expectation(mod1, data = grid1, ci = 0.95) |>
  as_tibble()

print(expect_grid1, n = 30)
```

```
# A tibble: 78 x 9
     age parentsurv friendalc sensation filmalc Predicted      SE  CI_low CI_high
   <dbl> <fct>      <fct>         <dbl>   <dbl>     <dbl>   <dbl>   <dbl>   <dbl>
 1  13   Low        No             12.0    2.81    0.0204 0.00656 0.0108   0.0381
 2  13.5 Low        No             12.0    2.81    0.0258 0.00805 0.0139   0.0473
 3  14   Low        No             12.0    2.81    0.0326 0.00991 0.0179   0.0587
 4  14.5 Low        No             12.0    2.81    0.0411 0.0122  0.0229   0.0729
 5  15   Low        No             12.0    2.81    0.0518 0.0150  0.0291   0.0905
 6  15.5 Low        No             12.0    2.81    0.0650 0.0185  0.0369   0.112
 7  16   Low        No             12.0    2.81    0.0814 0.0228  0.0464   0.139
 8  16.5 Low        No             12.0    2.81    0.101  0.0280  0.0582   0.171
 9  17   Low        No             12.0    2.81    0.126  0.0342  0.0724   0.209
10  17.5 Low        No             12.0    2.81    0.155  0.0415  0.0894   0.254
11  18   Low        No             12.0    2.81    0.189  0.0497  0.110    0.306
12  18.5 Low        No             12.0    2.81    0.229  0.0588  0.134    0.363
13  19   Low        No             12.0    2.81    0.274  0.0684  0.162    0.425
14  13   Medium     No             12.0    2.81    0.0134 0.00426 0.00716  0.0249
15  13.5 Medium     No             12.0    2.81    0.0170 0.00525 0.00924  0.0310
16  14   Medium     No             12.0    2.81    0.0215 0.00650 0.0119   0.0387
17  14.5 Medium     No             12.0    2.81    0.0272 0.00806 0.0152   0.0484
18  15   Medium     No             12.0    2.81    0.0344 0.0100  0.0194   0.0605
19  15.5 Medium     No             12.0    2.81    0.0434 0.0125  0.0246   0.0756
20  16   Medium     No             12.0    2.81    0.0547 0.0156  0.0310   0.0945
21  16.5 Medium     No             12.0    2.81    0.0686 0.0194  0.0390   0.118
22  17   Medium     No             12.0    2.81    0.0857 0.0242  0.0487   0.147
23  17.5 Medium     No             12.0    2.81    0.107  0.0300  0.0605   0.181
24  18   Medium     No             12.0    2.81    0.132  0.0370  0.0747   0.223
25  18.5 Medium     No             12.0    2.81    0.162  0.0451  0.0917   0.271
26  19   Medium     No             12.0    2.81    0.198  0.0544  0.112    0.326
27  13   High       No             12.0    2.81    0.0106 0.00337 0.00568  0.0197
28  13.5 High       No             12.0    2.81    0.0135 0.00416 0.00733  0.0246
29  14   High       No             12.0    2.81    0.0171 0.00515 0.00944  0.0307
30  14.5 High       No             12.0    2.81    0.0217 0.00640 0.0121   0.0385
# i 48 more rows
```

- Visualization of predictions

```r
expect_grid1 |>
  ggplot(aes(x = age, y = Predicted, color = friendalc)) +
  facet_wrap(vars(parentsurv)) +
  geom_line() +
  geom_point(size = 1.25) +
  scale_y_continuous(breaks = pretty_breaks(n = 10)) +
  scale_x_continuous(breaks = pretty_breaks(n = 5)) +
  labs(x = "Age", color = "Drinking Friend") +
  theme_bw(base_size = 14) +
  labs_pubr(16) +
  theme(legend.position = "top")
```



Question 5: Predicted values on the test data

1. Use `augment()` from `{broom}` to obtain the values predicted by the `mod1` model based on the test data `data_test`. The database obtained with `augment()` will be named `pred_test`
2. Using `ggplot()`, plot the histogram of the predicted values

Solution

- `augment()` on `mod1`

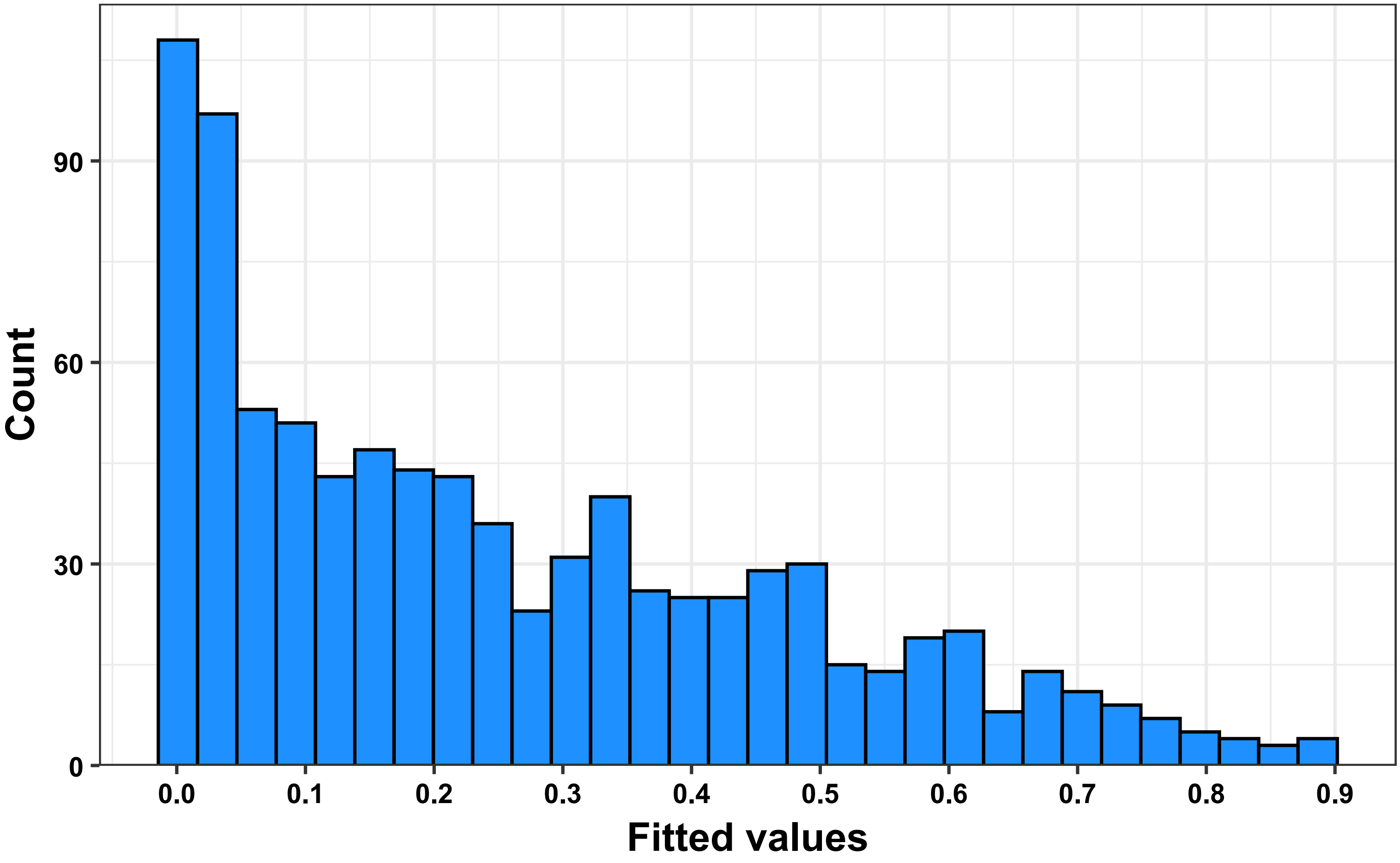
```
pred_test <- augment(mod1, newdata = data_test, type.predict = "response")

print(pred_test, n = 20)
```

```
# A tibble: 884 x 10
      id binge binge_bin friendalc sensation   age filmalc parentsurv   tag .fitted
   <dbl> <fct>     <dbl> <fct>         <dbl> <dbl>   <dbl> <fct>      <int>   <dbl>
 1     2 No            0 Yes              10    15   0.941 High           0 0.0562
 2     4 Yes           1 Yes              18    16   0.892 Low            0 0.609
 3     5 Yes           1 Yes              13    18   1.94  Low            0 0.575
 4     8 Yes           1 Yes              11    17   2.24  Low            0 0.343
 5    11 No            0 No                5    15   1.20  High           0 0.00310
 6    16 No            0 Yes              13    17   3.87  High           0 0.394
 7    20 Yes           1 Yes              16    17   1.80  Low            0 0.643
 8    21 No            0 No               14    14   2.97  Low            0 0.0563
 9    24 No            0 No                7    14   3.37  High           0 0.00517
10    26 No            0 Yes              16    16   1.72  Low            0 0.522
11    31 No            0 Yes              12    16   4.12  High           0 0.245
12    32 No            0 Yes              12    15   3.23  Medium         0 0.173
13    34 No            0 No                6    15   1.20  High           0 0.00405
14    37 No            0 No               12    16   0.835 Low            0 0.0551
15    50 Yes           1 Yes              10    18   1.45  High           0 0.220
16    53 No            0 Yes              12    17   4.15  High           0 0.346
17    58 No            0 Yes              14    16   1.89  High           0 0.255
18    59 Yes           1 Yes              12    17   2.58  Low            0 0.423
19    65 No            0 No               13    14   3.55  Low            0 0.0491
20    67 Yes           1 Yes              11    15   7.86  Low            0 0.397
# i 864 more rows
```

- Histogram of the predicted values

```
ggplot(pred_test, aes(x = .fitted)) +
  geom_histogram(fill = "dodgerblue", color = "black") +
  labs(y = "Count", x = "Fitted values") +
  scale_y_continuous(expand = expansion(c(0, 0.05))) +
  scale_x_continuous(breaks = pretty_breaks(n = 10)) +
  theme_bw(base_size = 14) +
  labs_pubr(16)
```



Question 6: Confusion matrix

Recall the *confusion matrix*

Threshold s	$y = 0$	$y = 1$
$\hat{y} = 0$	TN	FN
$\hat{y} = 1$	FP	TP

- True Positive (TP): Number of observations that are correctly predicted positives.
- True Negative (TN): Number of observations that are correctly predicted negatives.
- False Positive (FP): Number of observations that are incorrectly predicted positives.
- False Negative (FP): Number of observations that are incorrectly predicted negatives.

The counts in the matrix depend on the threshold s used to classify probabilities estimated by the model

Determine the confusion matrix with $s = 0.5$ and $s = 0.4$. Save the matrices in `confusion50`, `confusion40`
Hint: you can use `confusion_matrix()` from `helper_functions5.R`

Solution

- We generate `confusion50` and `confusion40`

```r
confusion50 <- confusion_matrix(pred_test[[".fitted"]], pred_test[["binge_bin"]], 0.5)
confusion50
```

```
       true
predict 0        1
      0 TN = 592 FN = 151
      1 FP = 50  TP = 91
```

```r
confusion40 <- confusion_matrix(pred_test[[".fitted"]], pred_test[["binge_bin"]], 0.4)
confusion40
```

```
       true
predict 0        1
      0 TN = 557 FN = 98
      1 FP = 85  TP = 144
```

Effect of the threshold as the threshold decreases from $s = 0.5$ to $s = 0.4$

- FN and TN decrease, FP and TP increase.
- A lower threshold captures more positive cases but may also increase false positives.

Question 7: Accuracy

Accuracy (`acc`) measures the proportion of correct predictions out of the total predictions, providing a general sense of model effectiveness

Threshold s	$y = 0$	$y = 1$
$\hat{y} = 0$	TN	FN
$\hat{y} = 1$	FP	TP

$$\texttt{acc} = \frac{TP + TN}{TP + TN + FP + FN}$$

1. Calculate the accuracy for values of s between 0 and 1 in increments of 0.01. Use
2. Plot the accuracy as a function of s

Solution

- We test the function with $s = 0.5$

```
acc(pred_test[[".fitted"]], pred_test[["binge"]], s = 0.5)
```

```
[1] 0.77262
```

- We compute all `acc`

```
acc_data <- tibble(s = seq(0, 1, 0.01)) |>
  group_by(s) |>
  mutate(acc = acc(pred_test[[".fitted"]], pred_test[["binge"]], s = s)) |>
  ungroup()

print(acc_data, n = 5)
```

```
# A tibble: 101 x 2
      s   acc
  <dbl> <dbl>
1  0    0.274
2  0.01 0.343
3  0.02 0.411
4  0.03 0.449
5  0.04 0.488
# i 96 more rows
```

- Graph of *accuracy* as a function of threshold s

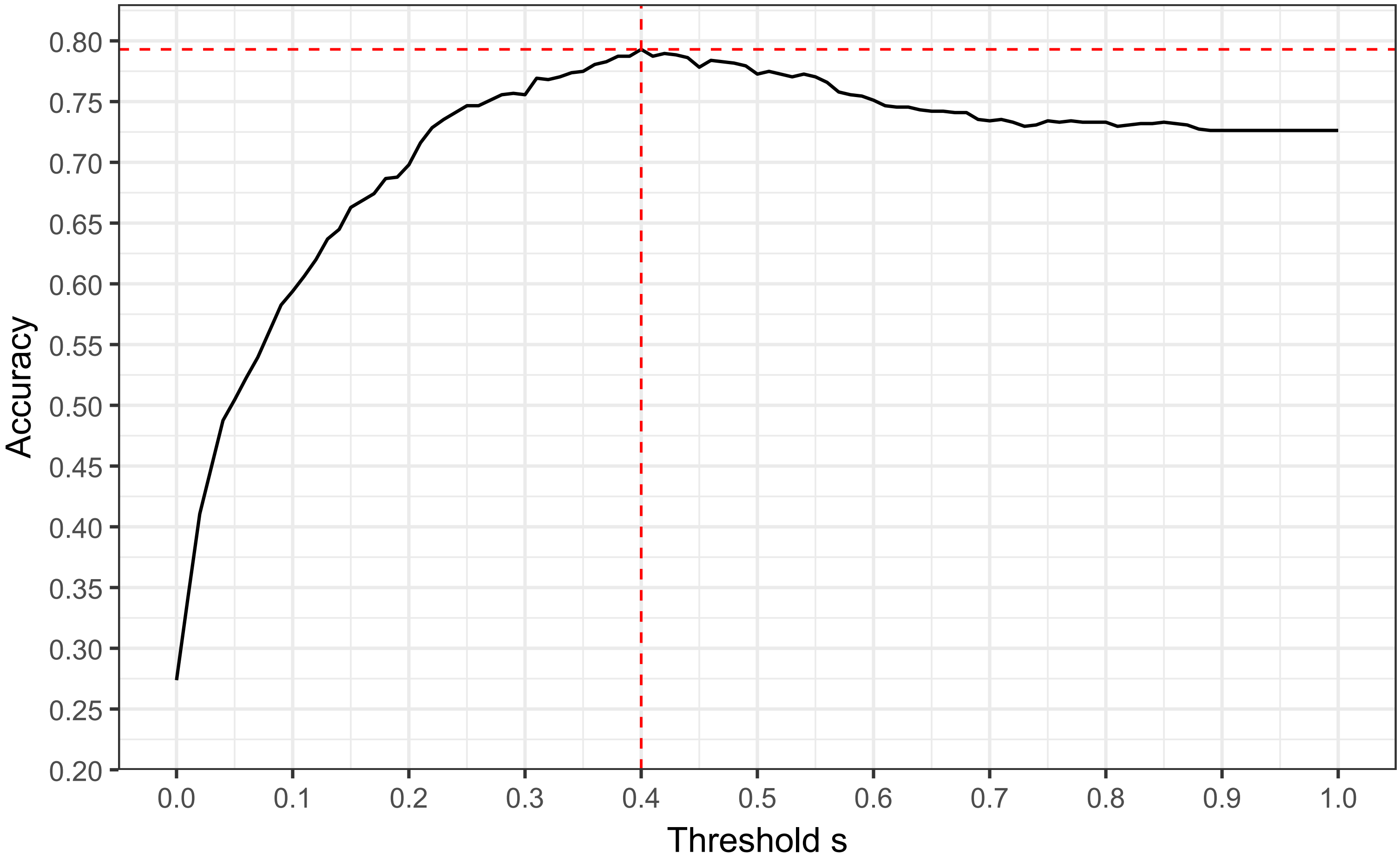
```
acc_max <- max(acc_data$acc)
acc_max
```

```
[1] 0.79299
```

```
s_max <- acc_data[acc_data$acc == acc_max, ][["s"]][1]
s_max
```

```
[1] 0.4
```

```
ggplot(acc_data, aes(x = s, y = acc)) +
  geom_line(color = "black") +
  geom_hline(yintercept = acc_max, linetype = 2, linewidth = 0.5, color = "red") +
  geom_vline(xintercept = s_max, linetype = 2, linewidth = 0.5, color = "red") +
  scale_y_continuous(breaks = pretty_breaks(n = 10), expand = expansion(c(0, 0.05))) +
  scale_x_continuous(breaks = seq(0, 1, 0.1)) +
  coord_cartesian(ylim = c(0.2, 0.8)) +
  labs(y = "Accuracy", x = "Threshold s") +
  theme_bw(base_size = 14)
```



- `error rate` $= 1 -$ `accuracy`

Question 8: Precision, Positive Predictive Value

Precision (`prec`), also known as *Positive Predictive Value* (`ppv`), measures the accuracy of positive predictions made by a classification model. It quantifies how often the model correctly identifies instances of the positive class. Specifically, precision is defined as the ratio of true positives to the sum of true positives and false positives

Threshold s	$y = 0$	$y = 1$
$\hat{y} = 0$	TN	FN
$\hat{y} = 1$	FP	TP

$$\texttt{prec} = \texttt{ppv} = \frac{TP}{TP + FP} = \mathbb{P}(y = 1 \mid \hat{y} = 1)$$

Negative Predictive Value (`npv`), which assesses the model's performance concerning the negative class

$$\texttt{npv} = \frac{TN}{TN + FN} = \mathbb{P}(y = 0 \mid \hat{y} = 0)$$

1. Calculate the *Precision* (`prec`) and `npv` for values of s between 0 and 1 in increments of 0.01.

Solution

- We test the function with $s = 0.5$

```r
prec(pred_test[[".fitted"]], pred_test[["binge"]], s = 0.5)
```

```
[1] 0.64539
```

```r
npv(pred_test[[".fitted"]], pred_test[["binge"]], s = 0.5)
```

```
[1] 0.79677
```

- We compute all `prec` and `npv`

```r
prec_npv_data <- tibble(s = seq(0, 1, 0.01)) |>
  group_by(s) |>
  mutate(prec = prec(pred_test[[".fitted"]], pred_test[["binge"]], s = s)) |>
  mutate(npv = npv(pred_test[[".fitted"]], pred_test[["binge"]], s = s)) |>
  ungroup()

prec_npv_data
```

```
# A tibble: 101 x 3
       s  prec     npv
   <dbl> <dbl>   <dbl>
 1  0     0.274 NaN
 2  0.01 0.294   0.984
 3  0.02 0.317   0.992
 4  0.03 0.331   0.987
 5  0.04 0.347   0.990
 6  0.05 0.355   0.990
 7  0.06 0.364   0.991
 8  0.07 0.371   0.984
 9  0.08 0.382   0.981
10  0.09 0.394   0.982
# i 91 more rows
```

Question 9: Recall (Sensitivity, True positive rate), Specificity (True Negative Rate)

Recall (`rec`), also known as *Sensitivity* (`sens`) or *True positive rate* (`tpr`), quantifies the ability of a classification model to correctly identify positive instances from the total actual positives in the dataset. It is defined as the ratio of true positives to the total number of actual positives:

$$\texttt{rec} = \texttt{sens} = \texttt{tpr} = \frac{TP}{TP + FN} = \mathbb{P}(\hat{y} = 1 | y = 1)$$

Specificity (`spec`), *True Negative Rate* (`tnr`) measures the proportion of actual negatives that are correctly identified by the model. It assesses the model's ability to avoid false positives

$$\texttt{spec} = \texttt{tnr} = \frac{TN}{TN + FP} = \mathbb{P}(\hat{y} = 0 | y = 0)$$

1. Calculate the `rec` and `spec` for values of s between 0 and 1 in increments of 0.01.
2. On the same graph, plot *Recall* and *Precision* as a function of s.

Solution

- We test the function with $s = 0.5$

```r
rec(pred_test[[".fitted"]], pred_test[["binge"]], s = 0.5)
```

```
[1] 0.37603
```

```r
spec(pred_test[[".fitted"]], pred_test[["binge"]], s = 0.5)
```

```
[1] 0.92212
```

- We compute all `rec` and `spec`

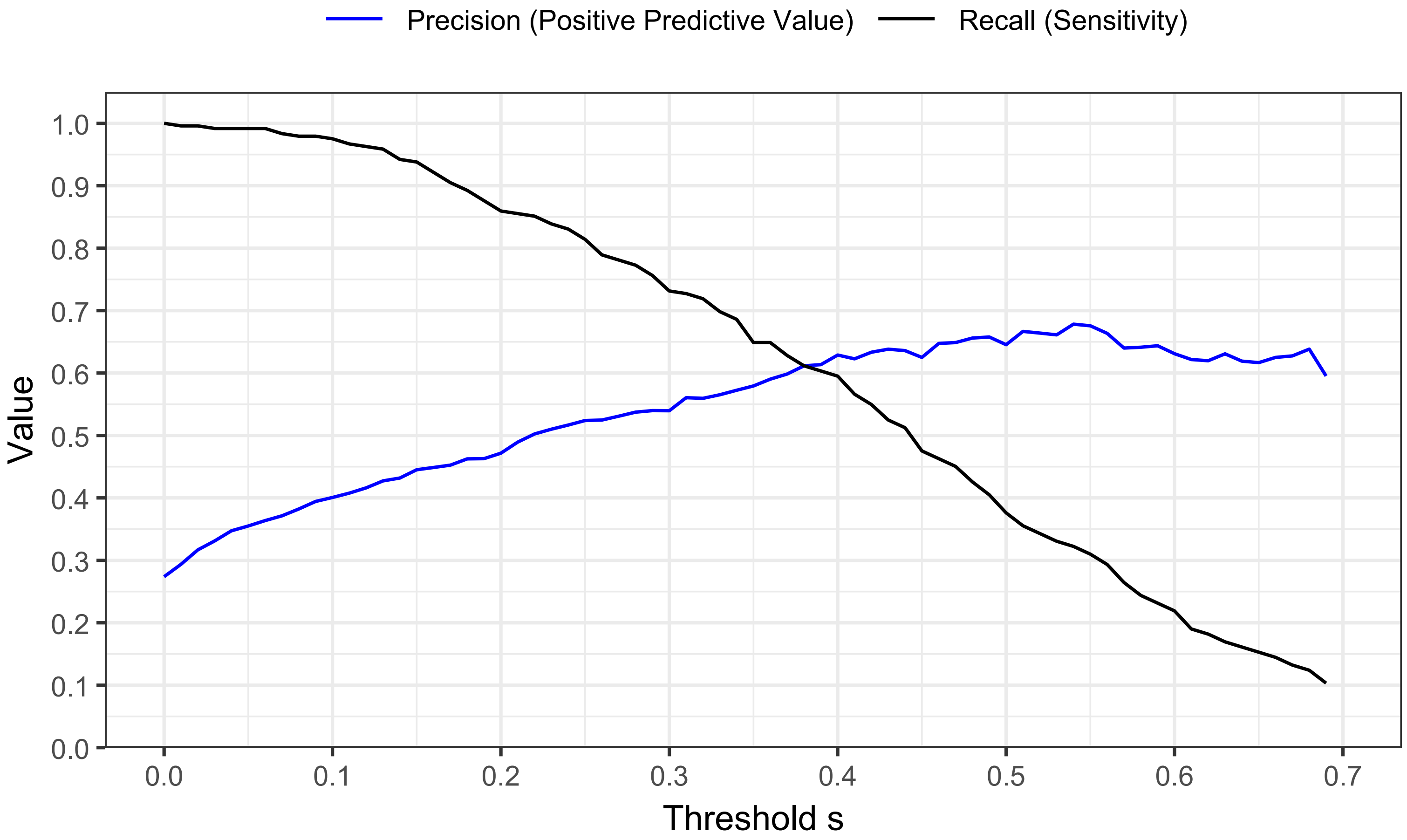
```r
rec_spec_data <- tibble(s = seq(0, 1, 0.01)) |>
  group_by(s) |>
  mutate(rec = rec(pred_test[[".fitted"]], pred_test[["binge"]], s = s)) |>
  mutate(spec = spec(pred_test[[".fitted"]], pred_test[["binge"]], s = s)) |>
  ungroup()

print(rec_spec_data, n = 20)
```

```
# A tibble: 101 x 3
       s   rec   spec
   <dbl> <dbl>  <dbl>
 1  0     1     0
 2  0.01 0.996 0.0966
 3  0.02 0.996 0.190
 4  0.03 0.992 0.245
 5  0.04 0.992 0.298
 6  0.05 0.992 0.321
 7  0.06 0.992 0.346
 8  0.07 0.983 0.372
 9  0.08 0.979 0.403
10  0.09 0.979 0.433
11  0.1  0.975 0.450
12  0.11 0.967 0.470
13  0.12 0.963 0.491
14  0.13 0.959 0.516
15  0.14 0.942 0.533
16  0.15 0.938 0.559
17  0.16 0.921 0.573
18  0.17 0.905 0.587
19  0.18 0.893 0.609
20  0.19 0.876 0.617
# i 81 more rows
```

- Graph of *recall* and *precision* as a function of s

```r
full_join(rec_spec_data, prec_npv_data) |>
  select(-spec, -npv) |>
  filter(s < 0.7) |>
  pivot_longer(-s) |>
  ggplot(aes(x = s, y = value, color = name)) +
  geom_line() +
  scale_y_continuous(breaks = pretty_breaks(n = 10), expand = expansion(c(0, 0.05))) +
  scale_x_continuous(breaks = seq(0, 1, 0.1), expand = expansion(c(0.05, 0.05))) +
  scale_color_manual(
    name = NULL, values = c("prec" = "blue", "rec" = "black"),
    labels = c("Precision (Positive Predictive Value)", "Recall (Sensitivity)")
  ) +
  coord_cartesian(ylim = c(0, 1), xlim = c(0, 0.7)) +
  labs(y = "Value", x = "Threshold s") +
  theme_bw(base_size = 14) +
  theme(legend.position = "top", legend.key.width = unit(1, "cm"))
```



Question 10: Receiver Operating Characteristic Curve (ROC curve)

1. Plot the ROC curve, which is the graph of `fpr` $= 1 -$ `spec` (False Positive Rate) as a function of *recall* (*sensitivity*, *tpr*)

$$\texttt{fpr} = 1 - \texttt{spec} = 1 - \mathbb{P}(\hat{y} = 0 \mid y = 0) = \mathbb{P}(\hat{y} = 1 \mid y = 0) = \frac{FP}{TN + FP}$$

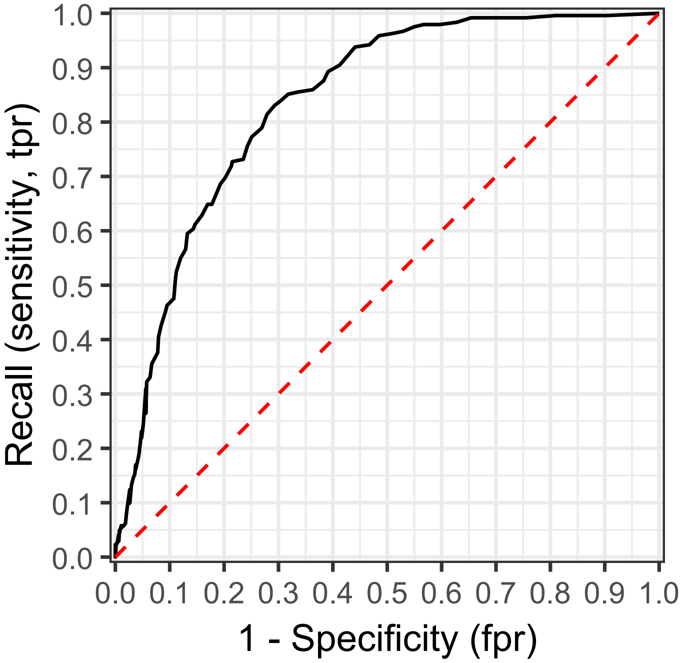
2. Compute the **Area Under the Curve** (AUC) with `performance_roc()` from `{performance}`

Reminder:
The **Area Under the Curve** (AUC) serves as a robust metric for summarizing the performance of a classification model across all possible thresholds. By comparing the AUC values of different models, we can assess their relative strengths and weaknesses.
AUC quantifies the model's overall ability to discriminate between positive and negative classes. AUC values range from 0 to 1, where 0 indicates that all predictions are incorrect, and 1 indicates that all predictions are correct.

Solution

- We plot the ROC curve using `rec_spec_data`.

```r
ggplot(rec_spec_data, aes(x = 1 - spec, y = rec)) +
  geom_line() + geom_line(aes(x = rec, y = rec), linetype = 2, color = "red") +
  scale_y_continuous(breaks = pretty_breaks(n = 10), expand = expansion(0.01)) +
  scale_x_continuous(breaks = pretty_breaks(n = 10), expand = expansion(0.01)) +
  labs(y = "Recall (sensitivity, tpr)", x = "1 - Specificity (fpr)") +
  theme_bw(base_size = 14) + theme(aspect.ratio = 1)
```



- We calculate the area under the curve (AUC) with `roc()` and `auc()` from `{pROC}`.

```
roc(pred_test, "binge", ".fitted") |>
  auc()
```

```
Area under the curve: 0.836
```

- With `performance_roc()` from `{performance}`

```
performance_roc(mod1, new_data = data_test)
```

```
AUC: 83.56%
```