432 Class 04

Thomas E. Love, Ph.D.

2026-01-22

Today's Agenda

- The HELP study
- Using tools from `rms` to fit:
  - linear models with `ols()`
  - logistic models with `lrm()`

Today’s R Setup

```r
knitr::opts_chunk$set(comment = NA)

library(janitor)
library(naniar)
library(broom); library(gt); library(patchwork)

library(haven)              ## for zapping labels
library(mosaic)             ## auto-loads mosaicData - data source
library(GGally)             ## for scatterplot matrix
library(rsample)
library(yardstick)

library(rms)                ## auto-loads Hmisc
library(easystats)
library(tidyverse)

theme_set(theme_bw())
```

Data from the HELP study

New Data (The HELP study)

Today's main data set comes from the Health Evaluation and Linkage to Primary Care trial, and is stored as `HELPrct` in the `mosaicData` package.

HELP was a clinical trial of adult inpatients recruited from a detoxification unit. Patients with no primary care physician were randomized to receive a multidisciplinary assessment and a brief motivational intervention or usual care, with the goal of linking them to primary medical care.

Key Variables for Today

Variable	Description
`id`	subject identifier (note: n = 453 subjects)
`cesd`	Center for Epidemiologic Studies Depression measure (scale is 0-60; higher scores indicate more depressive symptoms)
`age`	subject age (in years)
`sex`	female (n = 107) or male (n = 346)
`subst`	primary substance of abuse (alcohol, cocaine or heroin)
`mcs`	SF-36 Mental Component Score (lower = worse status)
`pcs`	SF-36 Physical Component Score (lower = worse status)
`pss_fr`	perceived social support by friends (higher = more support)

- All measures from baseline during the subjects' detoxification stay.
- More data and details at https://nhorton.people.amherst.edu/help/.

help_rct data load

```r
help_rct <- tibble(mosaicData::HELPrct) |>
  select(id, cesd, age, sex, subst = substance, mcs, pcs, pss_fr) |>
  mutate(across(where(is.character), as_factor)) |>
  mutate(id = as.character(id))

help_rct
```

```
# A tibble: 453 × 8
   id     cesd   age sex    subst     mcs   pcs pss_fr
   <chr> <int> <int> <fct>  <fct>   <dbl> <dbl>  <int>
 1 1        49    37 male   cocaine 25.1   58.4      0
 2 2        30    37 male   alcohol 26.7   36.0      1
 3 3        39    26 male   heroin   6.76  74.8     13
 4 4        15    39 female heroin  44.0   61.9     11
 5 5        39    32 male   cocaine 21.7   37.3     10
 6 6         6    47 female cocaine 55.5   46.5      5
 7 7        52    49 female cocaine 21.8   24.5      1
 8 8        32    28 male   alcohol  9.16  65.1      4
 9 9        50    50 female alcohol 22.0   38.3      5
10 10       46    39 male   heroin  36.1   22.6      0
# ℹ 443 more rows
```

What the data look like in help_rct

Note the labels.

```r
str(help_rct)
```

```
tibble [453 × 8] (S3: tbl_df/tbl/data.frame)
 $ id    : chr [1:453] "1" "2" "3" "4" ...
 $ cesd  : int [1:453] 49 30 39 15 39 6 52 32 50 46 ...
  ..- attr(*, "label")= chr "CESD at baseline"
 $ age   : int [1:453] 37 37 26 39 32 47 49 28 50 39 ...
  ..- attr(*, "label")= chr "age (years)"
 $ sex   : Factor w/ 2 levels "female","male": 2 2 2 1 2 1 1 2 1 2 ...
  ..- attr(*, "label")= chr "sex"
 $ subst : Factor w/ 3 levels "alcohol","cocaine",..: 2 1 3 3 2 2 2 1 1 3 ...
  ..- attr(*, "label")= chr "primary substance of abuse"
 $ mcs   : num [1:453] 25.11 26.67 6.76 43.97 21.68 ...
  ..- attr(*, "label")= chr "SF-36 Mental Component Score"
 $ pcs   : num [1:453] 58.4 36 74.8 61.9 37.3 ...
  ..- attr(*, "label")= chr "SF-36 Physical Component Score"
```

Getting rid of the labels

Suppose I don’t want the labels for some reason…

```r
help1 <- help_rct |> zap_label()
data_codebook(help1 |> select(-id))
```

```
select(help1, -id) (453 rows and 7 variables, 7 shown)

ID | Name    | Type        | Missings |         Values |           N
---+---------+-------------+----------+----------------+------------
1  | cesd    | integer     | 0 (0.0%) |        [1, 60] |         453
---+---------+-------------+----------+----------------+------------
2  | age     | integer     | 0 (0.0%) |       [19, 60] |         453
---+---------+-------------+----------+----------------+------------
3  | sex     | categorical | 0 (0.0%) |         female | 107 (23.6%)
   |         |             |          |           male | 346 (76.4%)
---+---------+-------------+----------+----------------+------------
4  | subst   | categorical | 0 (0.0%) |        alcohol | 177 (39.1%)
   |         |             |          |        cocaine | 152 (33.6%)
   |         |             |          |         heroin | 124 (27.4%)
```

Quantitative Summaries

```r
df_stats(~ cesd + age + mcs + pcs + pss_fr, data = help1) |>
  gt() |>
  fmt_number(min:max, decimals = 1) |>
  fmt_number(mean:sd, decimals = 2) |>
  tab_options(table.font.size = 24) |>
  opt_stylize(style = 1, color = "blue")
```

response	min	Q1	median	Q3	max	mean	sd	n	missing
cesd	1.0	25.0	34.0	41.0	60.0	32.85	12.51	453	0
age	19.0	30.0	35.0	40.0	60.0	35.65	7.71	453	0
mcs	6.8	21.7	28.6	40.9	62.2	31.68	12.84	453	0
pcs	14.1	40.4	48.9	57.0	74.8	48.05	10.78	453	0
pss_fr	0.0	3.0	7.0	10.0	14.0	6.71	4.00	453	0

`help1` categorical variables

```r
help1 |> tabyl(sex, subst) |>
    adorn_totals(where = c("row", "col")) |>
    adorn_percentages(denominator = "row") |>
    adorn_pct_formatting() |>
    adorn_ns(position = "front") |>
    adorn_title(placement = "combined") |>
  gt() |> tab_options(table.font.size = 24) |>
  opt_stylize(style = 2, color = "green")
```

sex/subst	alcohol	cocaine	heroin	Total
female	36 (33.6%)	41 (38.3%)	30 (28.0%)	107 (100.0%)
male	141 (40.8%)	111 (32.1%)	94 (27.2%)	346 (100.0%)
Total	177 (39.1%)	152 (33.6%)	124 (27.4%)	453 (100.0%)

Our quantitative outcome

- The CES-D is a 20-item measure that asks people to rate how often over the past week they experienced symptoms associated with depression, such as restless sleep, poor appetite, and feeling lonely.
  - Each item is rated on a 0-3 scale, and then summed, so possible scores range from 0 to 60.
  - Higher scores indicate more symptoms (or more frequent symptoms.)
- A version of the CES-D scale is available here as a PDF.

A cutoff for CES-D: Our binary outcome

- Scores of 16 or higher on the CES-D scale are sometimes taken to indicate that a person is at risk for clinical depression.

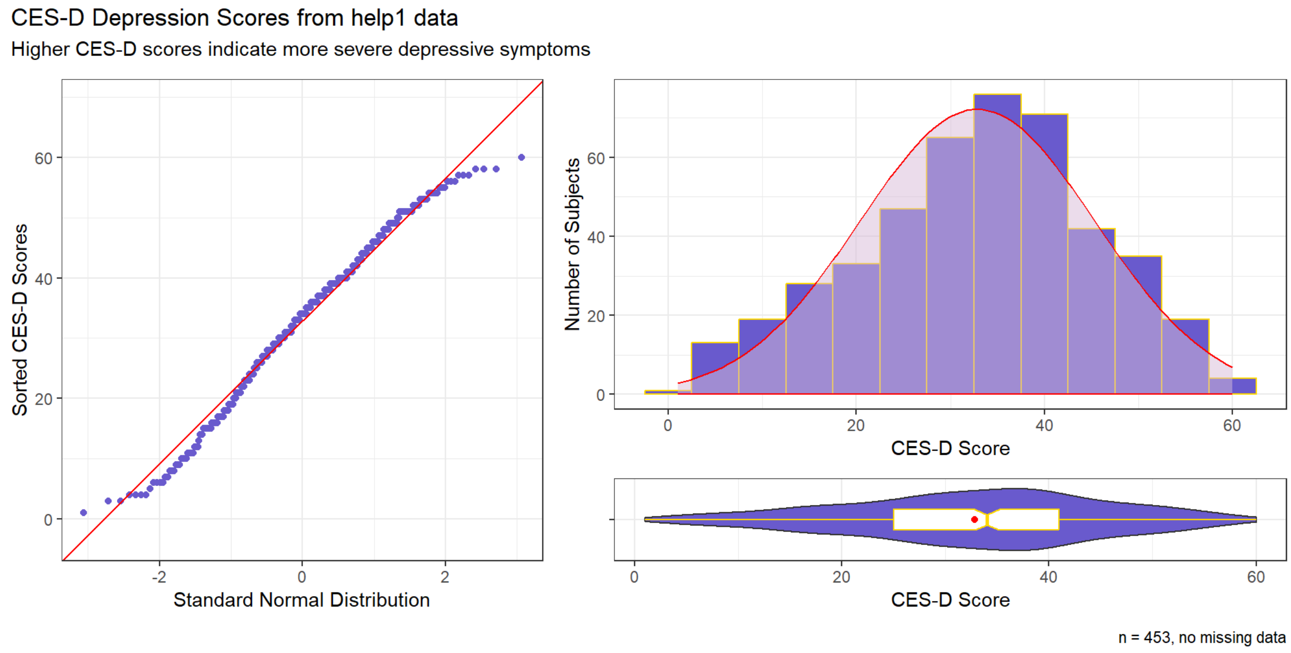
```r
help1 <- help1 |> mutate(cesd_hi = factor(as.numeric(cesd >= 16)))

help1 |> tabyl(cesd_hi) |> adorn_pct_formatting()
```

```
 cesd_hi   n percent
       0  46   10.2%
       1 407   89.8%
```

Quantitative Outcome (CES-D)

```r
p1 <- ggplot(help1, aes(sample = cesd)) +
  geom_qq(col = "slateblue") + geom_qq_line(col = "red") +
  theme(aspect.ratio = 1) +
    labs(y = "Sorted CES-D Scores",
         x = "Standard Normal Distribution")

bw = 5 # I tried a couple of things - this worked best for me with these data

p2 <- ggplot(help1, aes(x = cesd)) +
  geom_histogram(binwidth = bw, fill = "slateblue", col = "gold") +
  stat_function(fun = function(x)
    dnorm(x, mean = mean(help1$cesd), sd = sd(help1$cesd)) *
      length(help1$cesd) * bw,
    geom = "area", alpha = 0.5, fill = "thistle", col = "red") +
  labs(y = "Number of Subjects", x = "CES-D Score")

p3 <- ggplot(help1, aes(x = cesd, y = "")) +
  geom_violin(fill = "slateblue") +
```

Quantitative Outcome (CES-D)



Describing CES-D (1/2)

```r
describe(help1$cesd)   ## describe comes from the Hmisc package
```

```
help1$cesd
       n  missing distinct     Info     Mean  pMedian      Gmd      .05 
     453        0       58    0.999    32.85       33    14.23     10.0 
     .10      .25      .50      .75      .90      .95 
    15.2     25.0     34.0     41.0     49.0     52.4 

lowest :  1  3  4  5  6, highest: 55 56 57 58 60
```

- `Info` = variable’s information, between 0 and 1: the higher the `Info`, the more continuous the variable is (the fewer ties there are.)
- `pMedian` = Hodges-Lehman one-sample estimator of the pseudo-median. Median of all possible pairs of values.

Describing our outcome (2/2)

```r
describe(help1$cesd)
```

```
help1$cesd 
       n  missing distinct     Info     Mean  pMedian      Gmd      .05 
     453        0       58    0.999    32.85       33    14.23     10.0 
     .10      .25      .50      .75      .90      .95 
    15.2     25.0     34.0     41.0     49.0     52.4 

lowest :  1  3  4  5  6, highest: 55 56 57 58 60
```

- `Gmd` = Gini's mean difference, a robust measure of variation. If you select two subjects at random many times, the mean `cesd` difference will be 14.23 points.

More on the `Hmisc` package and `describe()` at Frank Harrell's website

The `easystats` approach

```r
describe_distribution(help1$cesd, iqr = FALSE, range = FALSE, ci = 0.90)
```

```
 Mean |  90% CI (Mean) |    SD | Skewness | Kurtosis |   n | n_Missing
----------------------------------------------------------------------
32.85 | [32.11, 33.78] | 12.51 |    -0.26 |    -0.44 | 453 |         0
```

```r
describe_distribution(help1$cesd,
                      centrality = "median", iqr = TRUE, quartiles = FALSE)
```

```
Median |   MAD |   IQR |         Range | Skewness | Kurtosis |   n | n_Missing
------------------------------------------------------------------------------
    34 | 11.86 | 16.50 | [1.00, 60.00] |    -0.26 |    -0.44 | 453 |         0
```

See this link at the datawizard package for more

Scatterplot Matrix (code)

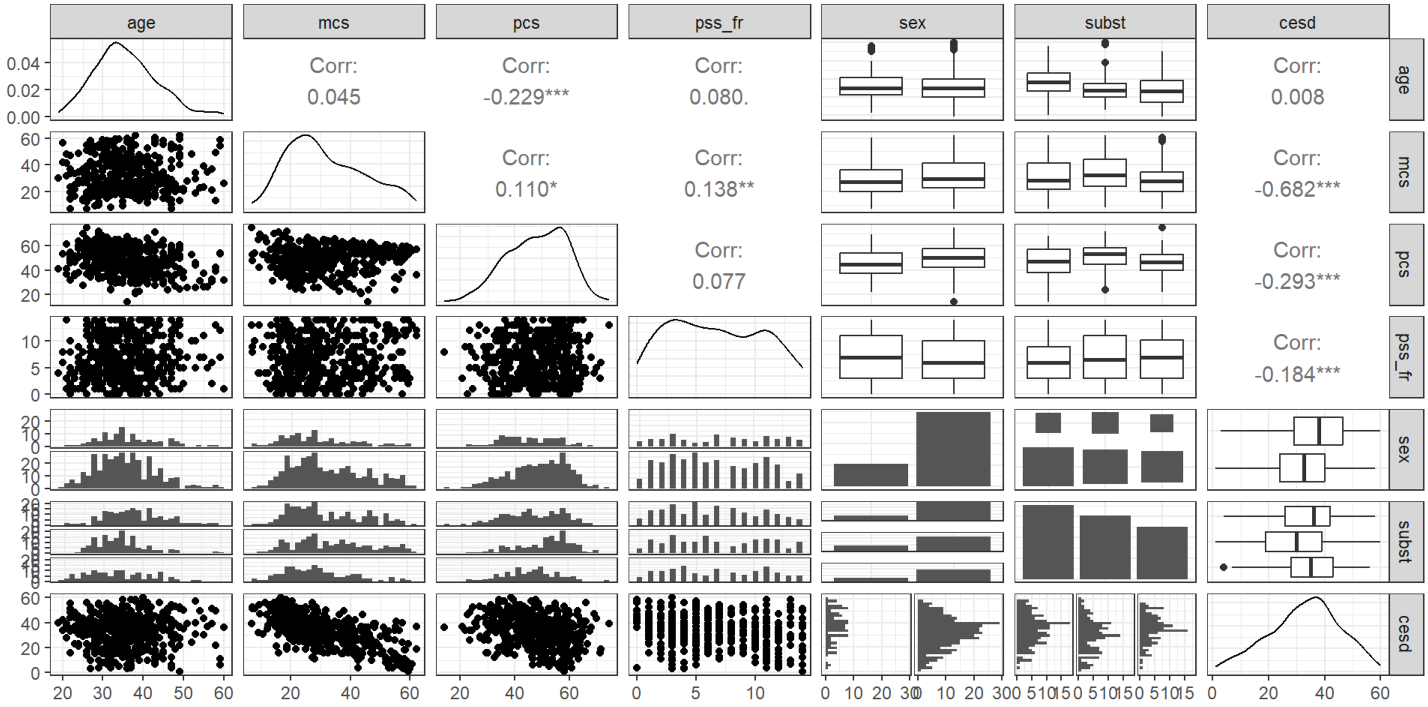
```r
temp <- help1 |>
    select(age, mcs, pcs, pss_fr, sex, subst, cesd)

ggpairs(temp)  ## ggpairs from the GGally package
```

We place the outcome (`cesd`) last (result on next slide.)

Saving the Data Set

```r
write_rds(help1, "c04/data/help1.Rds")
```

Scatterplot Matrix (result)



Using `ols()` to fit a linear regression model

Fitting using `ols()`

The `ols` function stands for ordinary least squares and comes from the `rms` package, by Frank Harrell and colleagues. Any model fit with `lm` can also be fit with `ols`.

- To predict `var_y` using `var_x` from the `my_tibble` data, we would use the following syntax:

```r
dd <- datadist(my_tibble)
options(datadist = "dd")

model_name <- ols(var_y ~ var_x, data = my_tibble,
                  x = TRUE, y = TRUE)
```

This leaves a few questions…

What’s the `datadist` stuff doing?

Before fitting an `ols` model to data from `my_tibble`, use:

```r
dd <- datadist(my_tibble)
options(datadist = "dd")
```

> Run (the datadist code above) once before any models are fitted, storing the distribution summaries for all potential variables. Adjustment values are 0 for binary variables, the most frequent category (or optionally the first category level) for categorical (factor) variables, the middle level for ordered factor variables, and medians for continuous variables. (excerpt from `datadist` documentation)

Why use `x = TRUE, y = TRUE`?

Once we’ve set up the summaries with `datadist`, we fit a model:

```r
model_name <- ols(var_y ~ var_x, data = my_tibble,
                  x = TRUE, y = TRUE)
```

- `ols` stores additional information beyond what `lm` does
- `x = TRUE` and `y = TRUE` save even more expanded information for building plots and summarizing fit.
- The defaults are `x = FALSE, y = FALSE`, but in 432, we’ll want them saved.

Using ols to fit a model

Let’s try to predict our outcome (cesd) using mcs and subst

- Start with setting up the datadist
- Then fit the model, including x = TRUE, y = TRUE

```r
dd <- datadist(help1)
options(datadist = "dd")

mod1 <- ols(cesd ~ mcs + subst, data = help1,
                x = TRUE, y = TRUE)
```

Contents of mod1?

```r
mod1
```

```
Linear Regression Model

ols(formula = cesd ~ mcs + subst, data = help1, x = TRUE, y = TRUE)

                Model Likelihood    Discrimination
                      Ratio Test           Indexes
Obs      453    LR chi2    295.10    R2       0.479
sigma9.0657    d.f.            3    R2 adj   0.475
d.f.     449    Pr(> chi2) 0.0000    g        9.827

Residuals

      Min        1Q    Median        3Q       Max
-25.43696  -6.74592   0.09334   6.16212  24.24842
```

New elements in `ols`

For our `mod1`,

- Model Likelihood Ratio test output includes `LR chi2 = 295.10, d.f. = 3, Pr(> chi2) = 0.0000`

The log of the likelihood ratio, multiplied by -2, yields a test against a χ^2 distribution. Interpret this as a goodness-of-fit test that compares `mod1` to a null model with only an intercept term. In `ols` this is similar to a global (ANOVA) F test.

New elements in `ols`

Under the R^2 values, we have `g = 9.827`.

- This is the g-index, based on Gini's mean difference. If you randomly selected two of the subjects in the model, the average difference in predicted `cesd` will be 9.827.
- This can be compared to the Gini's mean difference for the original `cesd` values, from `describe`, which was `Gmd = 14.23`.

Validate summaries from an `ols` fit

- The data used to fit the model provide an overly optimistic view of the quality of fit.
- We’re interested here in assessing how well the model might work in new data, using a resampling approach.

Validation Results

```r
set.seed(432)
validate(mod1)
```

```
          index.orig training    test optimism index.corrected   Lower   Upper
R-square      0.4787   0.4874  0.4737   0.0137          0.4650  0.3904  0.5302
MSE          81.4606  79.7851 82.2361  -2.4510         83.9116 75.1632 93.5270
g             9.8272   9.9133  9.8038   0.1095          9.7177  8.3704 10.7752
Intercept     0.0000   0.0000  0.2793  -0.2793          0.2793 -3.9317  5.1316
Slope         1.0000   1.0000  0.9894   0.0106          0.9894  0.8637  1.1075
           n
R-square  40
MSE       40
g         40
Intercept 40
Slope     40
```

- `index.orig` for R^2 is 0.4787. That’s what we get from the data used to fit `mod1`.

Resampling Validation for R^2

–	index.orig	training	test	optimism	index.corrected	n
R^2	0.4787	0.4874	0.4737	0.0137	0.4650	40

- With `validate` we create 40 (by default) bootstrapped resamples of the data and then split each of those into training and test samples.
  - For each of the 40 splits, R refits the model (same predictors) in the `training` sample to obtain R^2: mean across 40 splits is 0.4874
  - Check each model in its `test` sample: average R^2 was 0.4737
- `optimism` = `training` result - `test` result = 0.0137
- `index.corrected` = `index.orig` - `optimism` = 0.4650

While our *nominal* R^2 is 0.4787; correcting for optimism yields *validated* R^2 of 0.4650, so we conclude that R^2 = 0.4650 better estimates how `mod1` will perform in new data.

Resampling Validation for MS(Error)

–	index.orig	training	test	optimism	index.corrected	n
MSE	81.4606	79.7851	82.2361	-2.4510	83.9116	40

- `index.orig` for MSE = 81.4606. That's what we get from the data used to fit `mod1`.
- For each of the 40 splits, R refits the model (same predictors) in the `training` sample to obtain MSE: mean across 40 splits is 79.7851
- Check each model in its `test` sample: average MSE was 82.2361
- `optimism` = `training` result - `test` result = -2.4510
- `index.corrected` = `index.orig` - `optimism` = 83.9116

While our *nominal* MSE is 81.4606 (so RMSE = $\sqrt{81.4606} = 9.03$); correcting for optimism yields *validated* MSE of 83.9116 and validated RMSE = $\sqrt{83.9116} = 9.16$.

ANOVA for `mod1` fit by `ols`

```r
anova(mod1)
```

```
                Analysis of Variance          Response: cesd 

 Factor     d.f. Partial SS MS          F      P     
 mcs          1  31182.7237 31182.72373 379.42 <.0001
 subst        2    968.7563   484.37816   5.89 0.003 
 REGRESSION   3  33886.8359 11295.61195 137.44 <.0001
 ERROR      449  36901.6542    82.18631              
```

- This adds a line for the complete regression model (both terms) which can be helpful, but is otherwise the same as `anova()` after a fit using `lm()`.
- As with `lm`, this is a sequential ANOVA table, so if we had included `subst` in the model first, we'd get a different SS, MS, F and p for `mcs` and `subst`, but the same `REGRESSION` and `ERROR` results.

summary for `mod1` fit by `ols`

```r
summary(mod1, conf.int = 0.90)
```

```
             Effects              Response : cesd 

 Factor                  Low    High   Diff.  Effect   S.E.    Lower 0.9
 mcs                     21.676 40.941 19.266 -12.6580 0.64984 -13.7290 
 subst - cocaine:alcohol  1.000  2.000     NA  -3.4440 1.00550  -5.1013 
 subst - heroin:alcohol   1.000  3.000     NA  -1.7791 1.06810  -3.5396 
 Upper 0.9
 -11.587000
  -1.786700
  -0.018654
```

- How do we interpret the `subst` effects estimated by this model?
  - Effect of `subst` being `cocaine` instead of `alcohol` on `ces_d` is `-3.44` assuming no change in `mcs`, with 90% CI (-5.10, -1.79).
  - Effect of `subst` being `heroin` instead of `alcohol` on `ces_d` is `-1.78` assuming no change in `mcs`, with 90% CI (-3.54, -0.02).

But what about the `mcs` effect?

summary for `mod1` fit by `ols`

```r
summary(mod1, conf.int = 0.90)
```

```
             Effects              Response : cesd 

 Factor                     Low    High   Diff.  Effect   S.E.    Lower 0.9
 mcs                        21.676 40.941 19.266 -12.6580 0.64984 -13.7290 
 subst - cocaine:alcohol     1.000  2.000     NA  -3.4440 1.00550  -5.1013 
 subst - heroin:alcohol      1.000  3.000     NA  -1.7791 1.06810  -3.5396 
 Upper 0.9 
 -11.587000
  -1.786700
  -0.018654
```

- Effect of `mcs`: `-12.66` is the estimated change in `cesd` associated with a move from `mcs` = 21.68 (see `Low` value) to `mcs` = 40.94 (the `High` value) assuming no change in `subst`.
- `ols` chooses the `Low` and `High` values from the interquartile range.

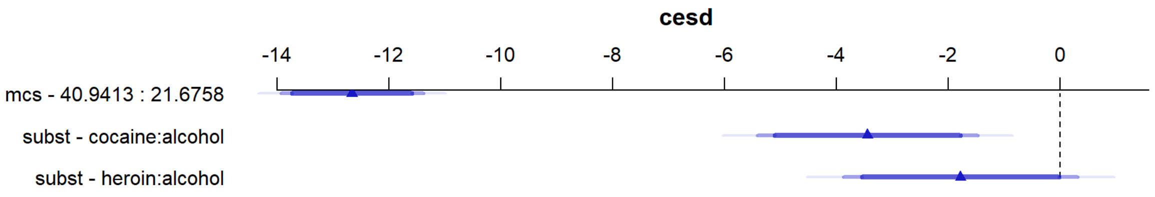
```r
quantile(help1$mcs, c(0.25, 0.75))
```

```
     25%      75% 
21.67575 40.94134 
```

Plot the summary to see effect sizes

- Goal: plot effect sizes for similar moves within predictor distributions.

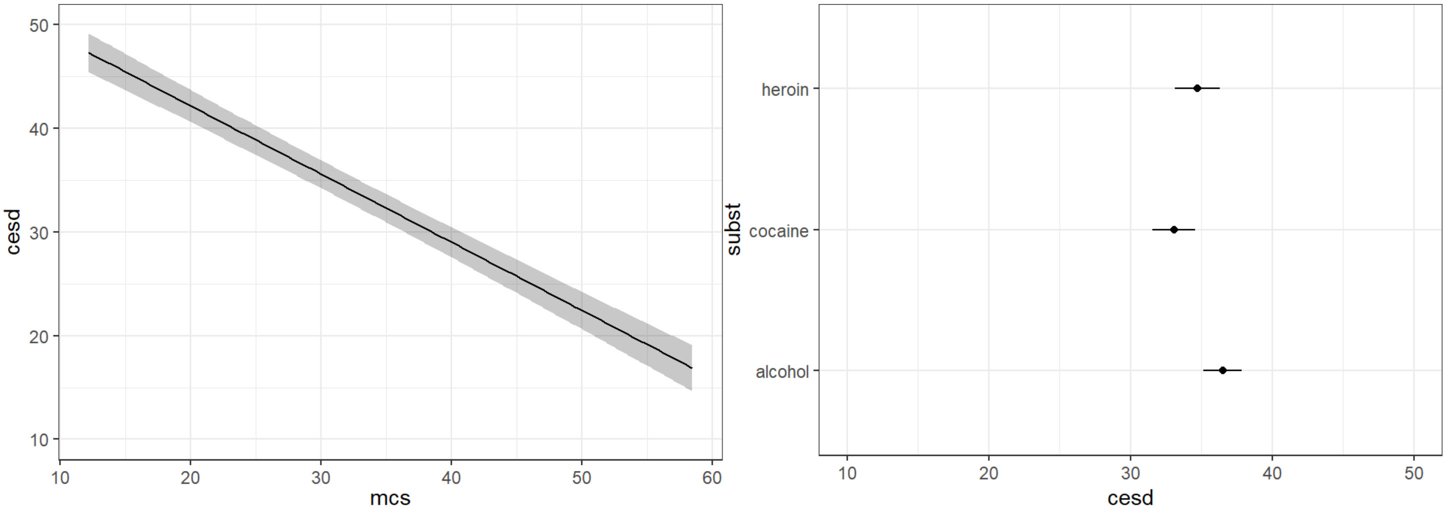
```r
plot(summary(mod1))
```



- The triangles indicate the point estimate, augmented with confidence interval bars.
  - The 90% confidence intervals are plotted with the thickest bars.
  - The 95% CIs are then shown with thinner, more transparent bars.
  - Finally, the 99% CIs are shown as the longest, thinnest bars.

Plot the individual effects?

```r
ggplot(Predict(mod1, conf.int = 0.95), layout = c(1,2))
```

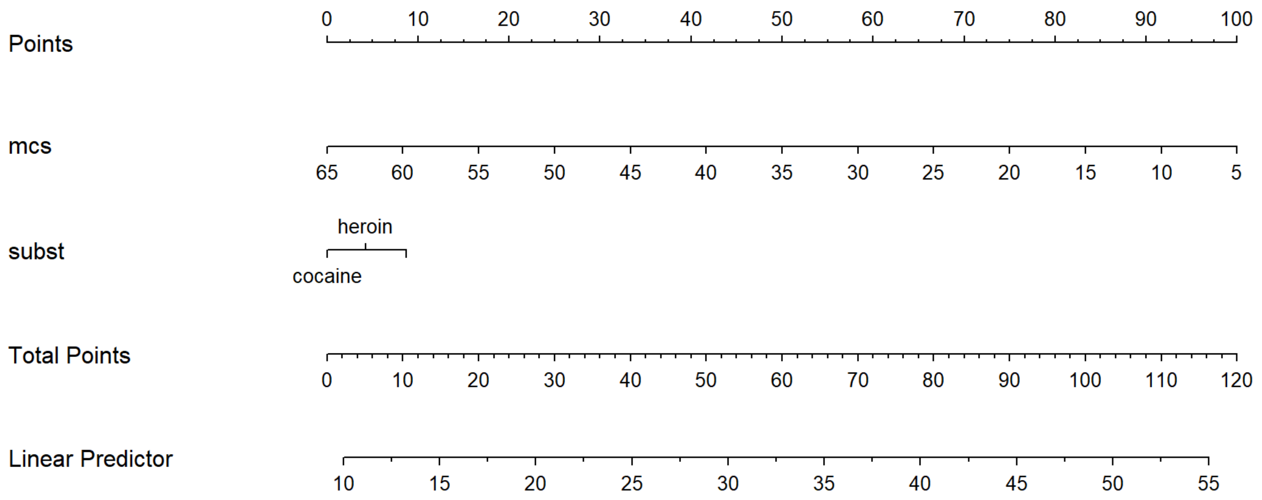


- At left, impact of changing `mcs` on `cesd` holding `subst` at its baseline (alcohol).
- At right, impact of changing `subst` on `cesd` holding `mcs` at its median (28.602417).
- Defaults: add 95% CI bands and layout tries for a square.

Standing Break

Build a nomogram for the `ols` fit

```r
plot(nomogram(mod1))
```



Nomograms

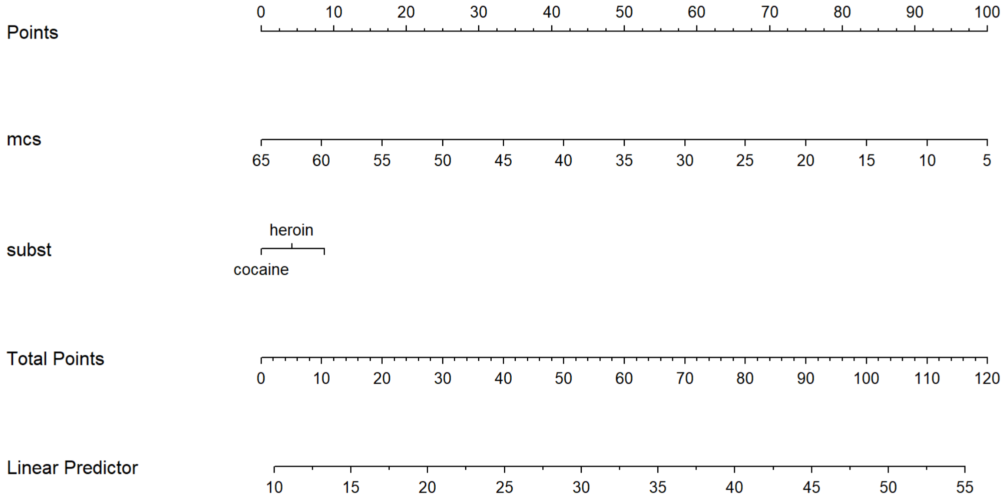
For complex models (this model isn't actually very complex) it can be helpful to have a tool that will help you see the modeled effects in terms of their impact on the predicted outcome.

A *nomogram* is an established graphical tool for doing this.

- Find the value of each predictor on its provided line, and identify the "points" for that predictor by drawing a vertical line up to the "Points".
- Then sum up the points over all predictors to obtain "Total Points".
- Draw a vertical line down from the "Total Points" to the "Linear Predictor" to get the predicted `cesd` for this subject.

Using the nomogram for `mod1`

Predicted `cesd` if `mcs` = 35 and `subst` = heroin?



Actual Prediction for this subject…

- The `predict` function for our `ols` fit provides fitted values.

```r
predict(mod1, newdata = tibble(mcs = 35, subst = "heroin"))
```

```
       1 
30.52766 
```

Using `lrm()` to fit a logistic regression model

Fitting using `lrm()`

The `lrm()` function stands for logistic regression model and also comes from the `rms` package. Let’s predict our binary outcome (`cesd_hi`) using `mcs` and `subst`.

- Start with setting up the `datadist` Then fit model, including `x = TRUE, y = TRUE`

```r
dd <- datadist(help1)
options(datadist = "dd")

mod2 <- lrm(cesd_hi ~ mcs + subst, data = help1, x = TRUE, y = TRUE)
```

Contents of mod2?

```
mod2
```

```
Logistic Regression Model

lrm(formula = cesd_hi ~ mcs + subst, data = help1, x = TRUE, 
    y = TRUE)

                       Model Likelihood    Discrimination    Rank Discrim.    
                             Ratio Test           Indexes          Indexes    
Obs           453    LR chi2     134.24    R2       0.533    C       0.938    
 0             46    d.f.             3    R2(3,453)0.252    Dxy     0.875    
 1            407    Pr(> chi2) <0.0001    R2(3,124)0.653    gamma   0.875    
max |deriv| 6e-06                          Brier    0.056    tau-a   0.160    

              Coef    S.E.   Wald Z Pr(>|Z|)
Intercept     10.5778 1.2429  8.51  <0.0001 
```

New elements in `lrm`

For our `mod2`,

- Model Likelihood Ratio test output includes `LR chi2 = 134.24, d.f. = 3, Pr(> chi2) <0.0001`

Again, the log of the likelihood ratio, multiplied by -2, yields a test against a χ^2 distribution. Interpret this as a goodness-of-fit test that compares `mod2` to a null model with only an intercept term.

Discrimination Indexes in `lrm()`

```
R2 = 0.533, R2(3,453) = 0.252, R2(3,124) = 0.653,
Brier = 0.056
```

The `R2` value is the *Nagelkerke* R^2, which is another pseudo-R^2 measure that provides a correction to the Cox-Snell R^2 so that the maximum value is 1.

- Other R^2s are detailed here

Discrimination Indexes in `lrm()`

```
R2 = 0.533, R2(3,453) = 0.252, R2(3,124) = 0.653,
Brier = 0.056
```

The `Brier` score is the mean squared error between predictions and actual (1/0) observations. The lower the score (closer to 0), the better the model's predictions are calibrated. It's not really useful on its own, but helps when comparing models.

Rank Discrimination Indexes in `lrm()`

`C = 0.938, Dxy = 0.875, gamma = 0.875, tau-a = 0.160`

- C is the C statistic, the area under the ROC curve
- Dxy is Somers' d, and note that C = 0.5 + (Dxy/2)
- gamma is the Goodman-Kruskal γ statistic
- tau-a is the Kendall τ statistic (version a)

Validate summaries from an `lrm` fit

- Can we validate summary statistics by resampling?

```r
set.seed(432432)
validate(mod2)
```

```
          index.orig training   test optimism index.corrected   Lower  Upper  n
Dxy           0.8751   0.8825 0.8707   0.0118          0.8634  0.8139 0.9271 40
R2            0.5326   0.5421 0.5247   0.0174          0.5152  0.4313 0.6253 40
Intercept     0.0000   0.0000 0.0069  -0.0069          0.0069 -0.5537 0.6074 40
Slope         1.0000   1.0000 0.9619   0.0381          0.9619  0.6776 1.2723 40
Emax          0.0000   0.0000 0.0582  -0.0582          0.0582 -0.0216 0.1823 40
D             0.2941   0.2988 0.2891   0.0097          0.2844  0.2113 0.3697 40
U            -0.0044  -0.0044 0.0001  -0.0045          0.0001 -0.0079 0.0163 40
Q             0.2985   0.3032 0.2890   0.0142          0.2843  0.2048 0.3666 40
B             0.0560   0.0548 0.0571  -0.0022          0.0583  0.0419 0.0736 40
g             2.7444   2.8543 2.7041   0.1502          2.5942  1.9223 3.2205 40
gp            0.1577   0.1573 0.1569   0.0004          0.1574  0.1197 0.1921 40
```

Resampling Validation after `lrm()`

–	index.orig	training	test	optimism	index.corrected	n
Dxy	0.8751	0.8825	0.8707	0.0118	0.8634	40
R2	0.5326	0.5421	0.5247	0.0174	0.5152	40

- Dxy = Somers' d, and the area under the ROC curve is C = 0.5 + (Dxy/2)
- Our original Dxy = 0.8751, implying C = 0.9376
- Our validated Dxy = 0.8634, so validated C = 0.5 + (0.8634/2) = 0.9317
- While our *nominal* R^2 is 0.5326; correcting for optimism yields *validated* R^2 of 0.5152.

ANOVA for `mod2` fit by `lrm`

```
anova(mod2)
```

```
                Wald Statistics          Response: cesd_hi 

 Factor     Chi-Square d.f. P     
 mcs        58.43      1    <.0001
 subst      10.04      2    0.0066
 TOTAL      62.30      3    <.0001
```

- Again, this is a sequential ANOVA table, so if we had included `subst` in the model first, we'd get a different Chi-Square, and p for `mcs` and `subst`, but the same `TOTAL` result.

summary for `mod2` fit by `lrm`

```r
summary(mod2, conf.int = 0.90)
```

```
             Effects              Response : cesd_hi 

 Factor                    Low    High   Diff.  Effect    S.E.    Lower 0.9
 mcs                       21.676 40.941 19.266 -3.460400 0.45270 -4.20500
  Odds Ratio               21.676 40.941 19.266  0.031417      NA  0.01492
 subst - cocaine:alcohol    1.000  2.000     NA -1.502500 0.48114 -2.29390
  Odds Ratio                1.000  2.000     NA  0.222580      NA  0.10087
 subst - heroin:alcohol     1.000  3.000     NA -1.269500 0.59788 -2.25290
  Odds Ratio                1.000  3.000     NA  0.280980      NA  0.10509
 Upper 0.9
 -2.715800
  0.066152
 -0.711070
  0.491120
```

summary for `mod2` fit by `lrm`

```
Factor             Low    High   Diff.   Effect   S.E.  Lower 0.9 Upper 0.9
 mcs            21.676 40.941 19.266 -3.46040 0.4527 -4.2050   -2.71580
  Odds Ratio    21.676 40.941 19.266  0.03142     NA  0.0149    0.06615
```

- Odds of `cesd_hi` are 0.03 times as high for a subject with `mcs` = 40.94 (`High`) as compared to a subject with `mcs` = 21.68 (`Low`) assuming no change in `subst`.
- `ols` chooses the `Low` and `High` values from the interquartile range.

summary for mod2 fit by lrm

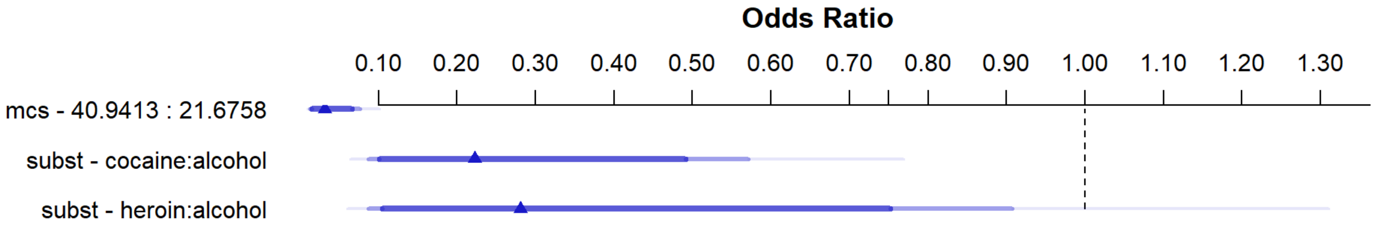
```
Factor                      Low High  Diff Effect   S.E. Lower 0.9 Upper 0.9
 subst - cocaine:alcohol  1    2    NA   -1.5025 0.4811 -2.2939   -0.71107
  Odds Ratio              1    2    NA    0.2226     NA  0.1009    0.49112
 subst - heroin:alcohol   1    3    NA   -1.2695 0.5979 -2.2529   -0.28607
  Odds Ratio              1    3    NA    0.2810     NA  0.1051    0.75121
```

- Effect of `subst` being `cocaine` instead of `alcohol` on `cesd_hi` is an Odds Ratio of 0.22 (0.10, 0.49), assuming no change in `mcs`.
- Effect of `subst` being `heroin` instead of `alcohol` on `cesd_hi` is an Odds Ratio of 0.28 (0.11, 0.75), assuming no change in `mcs`.

Plot the summary to see effect sizes

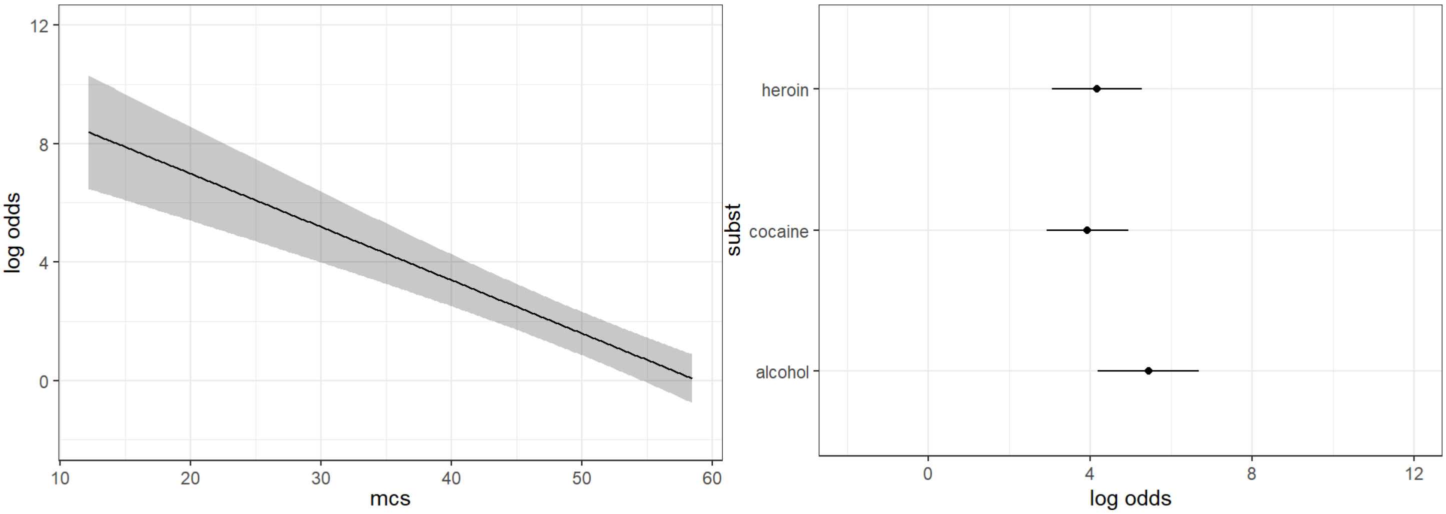
- Goal: plot effect sizes for similar moves within predictor distributions.

```
plot(summary(mod2))
```



Plot the individual effects?

```r
ggplot(Predict(mod2, conf.int = 0.95), layout = c(1,2))
```

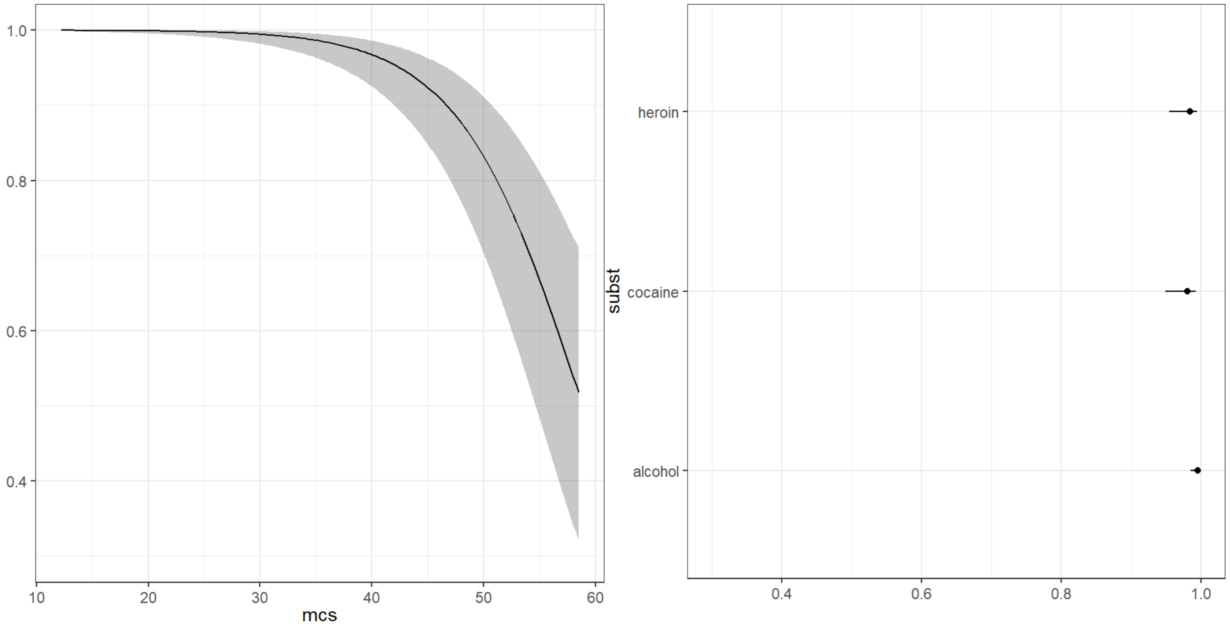


- At left, impact of changing `mcs` on `cesd` holding `subst` at its baseline (alcohol).
- At right, impact of changing `subst` on `cesd` holding `mcs` at its median (28.602417).
- Defaults: add 95% CI bands and layout tries for a square.

Plot on probability scale?

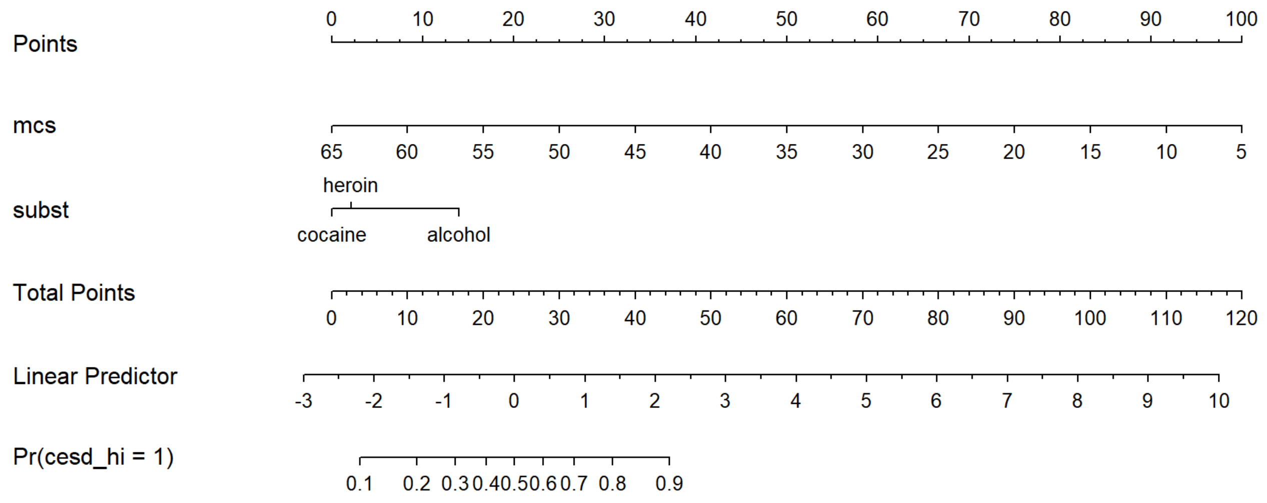
Add `fun = plogis`.

```r
ggplot(Predict(mod2, conf.int = 0.95, fun = plogis), layout = c(1,2))
```



Build a nomogram for the `ols` fit

```r
plot(nomogram(mod2, fun = plogis, funlabel = 'Pr(cesd_hi = 1)'))
```



Making a Prediction…

- The `predict` function for our `lrm()` fit provides fitted values, either on the log odds scale…

```r
predict(mod2, newdata = tibble(mcs = 35, subst = "heroin"), type = "lp")
```

```
       1 
3.021763 
```

- or on the probability scale …

```r
predict(mod2, newdata = tibble(mcs = 35, subst = "heroin"), type = "fitted")
```

```
        1 
0.9535477 
```

Getting more good stuff

- Anything you can fit with `ols()` can also be fit with `lm()`, so you have access to everything in `lm()` as well, like `check_model()`, etc.
- Same goes for `glm(..., family = binomial(link = "logit"))` and `lrm()`.

Coming Soon

- Fitting more complex linear and logistic regression models
  - Adding non-linearity in the predictors through interactions, polynomials and splines
  - Spending degrees of freedom and the Spearman ρ^2 plot