432 Class 06

https://thomaselove.github.io/432-2025/

2025-01-30

## Today’s Agenda

* The HELP trial, again
* Incorporating Non-Linearity into our models
  + Polynomial terms
  + Restricted Cubic Splines

## Today’s R Setup

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(rms) ## auto-loads Hmisc  
library(easystats)  
library(tidyverse)  
  
theme\_set(theme\_bw())

## Reminders: The HELP Study

Health Evaluation and Linkage to Primary Care (HELP) was a clinical trial of adult inpatients recruited from a detoxification unit.

* We have baseline data for each subject on several variables, including two outcomes:

| Variable | Description |
| --- | --- |
| cesd | Center for Epidemiologic Studies-Depression |
| cesd\_hi | cesd above 15 (indicates high risk) |

## Potential Predictors in help1

| Variable | Description |
| --- | --- |
| age | subject age (in years) |
| sex | female (n = 107) or male (n = 346) |
| subst | substance abused (alcohol, cocaine, heroin) |
| mcs | SF-36 Mental Component Score |
| pcs | SF-36 Physical Component Score |
| pss\_fr | perceived social support by friends |

* See <https://nhorton.people.amherst.edu/help/> for more.

## help1 data load

help1 <- tibble(mosaicData::HELPrct) |>  
 select(id, cesd, age, sex, subst = substance, mcs, pcs, pss\_fr) |>  
 zap\_label() |>  
 mutate(across(where(is.character), as\_factor),   
 id = as.character(id),   
 cesd\_hi = factor(as.numeric(cesd >= 16)))  
  
dim(help1); n\_miss(help1)

[1] 453 9

[1] 0

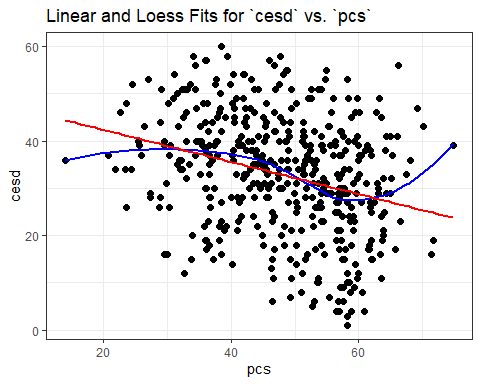
head(help1, 5)

# A tibble: 5 × 9  
 id cesd age sex subst mcs pcs pss\_fr cesd\_hi  
 <chr> <int> <int> <fct> <fct> <dbl> <dbl> <int> <fct>   
1 1 49 37 male cocaine 25.1 58.4 0 1   
2 2 30 37 male alcohol 26.7 36.0 1 1   
3 3 39 26 male heroin 6.76 74.8 13 1   
4 4 15 39 female heroin 44.0 61.9 11 0   
5 5 39 32 male cocaine 21.7 37.3 10 1

## Can we use pcs to predict cesd?

Does the loess smooth match up well with the linear fit?

ggplot(help1, aes(x = pcs, y = cesd)) +   
 geom\_point(size = 2) +  
 geom\_smooth(method = "loess", formula = y ~ x, se = FALSE, col = "blue") +  
 geom\_smooth(method = "lm", formula = y ~ x, se = FALSE, col = "red") +   
 labs(title = "Linear and Loess Fits for `cesd` vs. `pcs`")



## A simple linear regression: fitA

dd <- datadist(help1); options(datadist = "dd")  
  
fitA <- ols(cesd ~ pcs, data = help1, x = TRUE, y = TRUE)  
  
fitA$coefficients

Intercept pcs   
49.1673458 -0.3396495

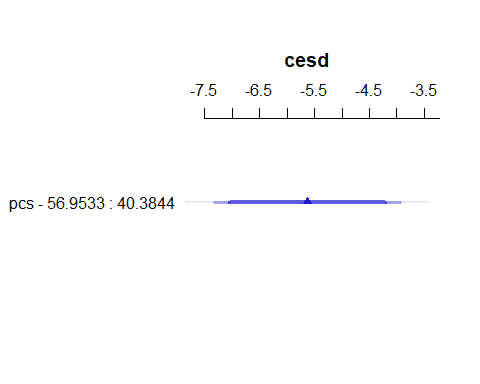
## Our simple linear regression

fitA

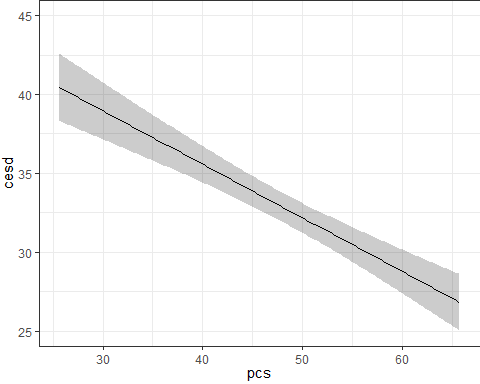
Linear Regression Model  
  
ols(formula = cesd ~ pcs, data = help1, x = TRUE, y = TRUE)  
  
 Model Likelihood Discrimination   
 Ratio Test Indexes   
Obs 453 LR chi2 40.57 R2 0.086   
sigma11.9796 d.f. 1 R2 adj 0.084   
d.f. 451 Pr(> chi2) 0.0000 g 4.177   
  
Residuals  
  
 Min 1Q Median 3Q Max   
-28.4116 -7.8036 0.6846 8.7917 29.3281   
  
 Coef S.E. t Pr(>|t|)  
Intercept 49.1673 2.5728 19.11 <0.0001   
pcs -0.3396 0.0522 -6.50 <0.0001

## Effect Sizes in fitA

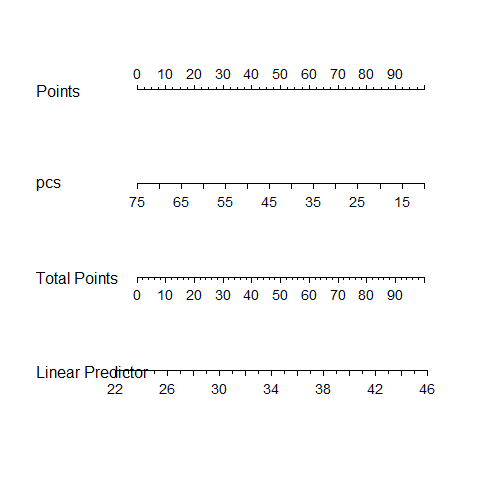
plot(summary(fitA))



ggplot(Predict(fitA, conf.int = 0.90))



plot(nomogram(fitA))



## Using ols to fit a larger model

dd <- datadist(help1)  
options(datadist = "dd")  
  
fitB <- ols(cesd ~ pcs + subst + pss\_fr + sex,   
 data = help1, x = TRUE, y = TRUE)  
  
fitB$coefficients

Intercept pcs subst=cocaine subst=heroin pss\_fr   
 53.7511151 -0.2574023 -3.8664109 0.2322071 -0.5370221   
 sex=male   
 -4.8446977

* Can use model\_parameters() and model\_performance() with fitB or other ols() fits.
* We could also fit this model, naturally, using lm() instead.

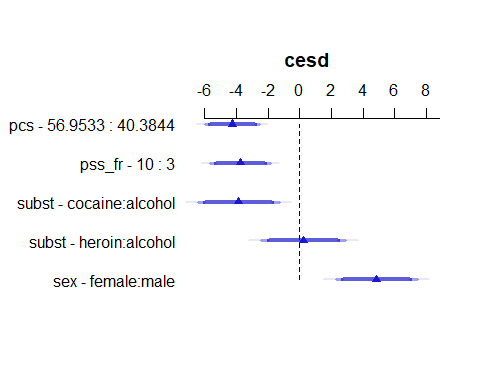
## Contents of fitB?

fitB

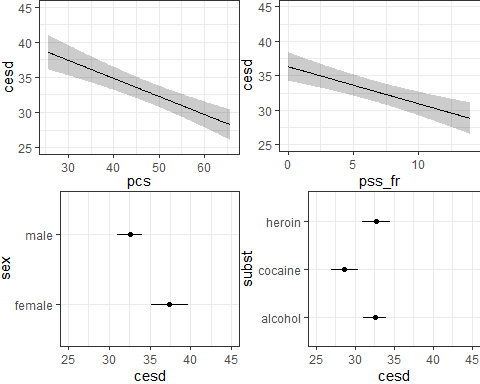
Linear Regression Model  
  
ols(formula = cesd ~ pcs + subst + pss\_fr + sex, data = help1,   
 x = TRUE, y = TRUE)  
  
 Model Likelihood Discrimination   
 Ratio Test Indexes   
Obs 453 LR chi2 76.43 R2 0.155   
sigma11.5662 d.f. 5 R2 adj 0.146   
d.f. 447 Pr(> chi2) 0.0000 g 5.625   
  
Residuals  
  
 Min 1Q Median 3Q Max   
-30.2603 -8.2273 0.5828 7.8033 27.7604   
  
 Coef S.E. t Pr(>|t|)  
Intercept 53.7511 2.7505 19.54 <0.0001   
pcs -0.2574 0.0526 -4.89 <0.0001   
subst=cocaine -3.8664 1.3061 -2.96 0.0032   
subst=heroin 0.2322 1.3568 0.17 0.8642   
pss\_fr -0.5370 0.1371 -3.92 0.0001   
sex=male -4.8447 1.3054 -3.71 0.0002

## Effect Sizes in fitB

plot(summary(fitB))

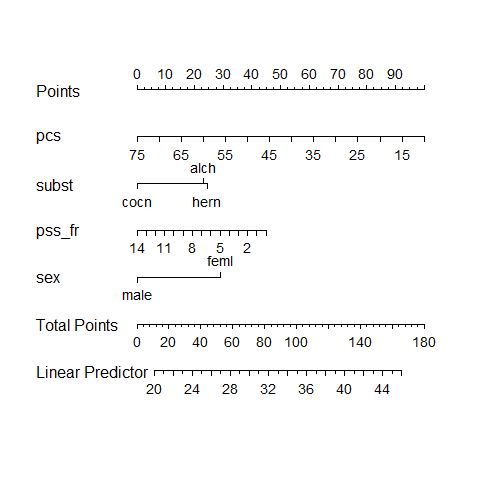


ggplot(Predict(fitB, conf.int = 0.90))



## A Nomogram for fitB

plot(nomogram(fitB, abbrev = TRUE))



## Non-Linear Terms

In building a linear regression model, we’re most often going to be thinking about:

* for quantitative predictors, some curvature…
  + perhaps polynomial terms
  + but more often restricted cubic splines
* for any predictors, possible interactions
  + between categorical predictors
  + between categorical and quantitative predictors
  + between quantitative predictors

# Non-Linear Terms: Polynomials

## Polynomial Regression

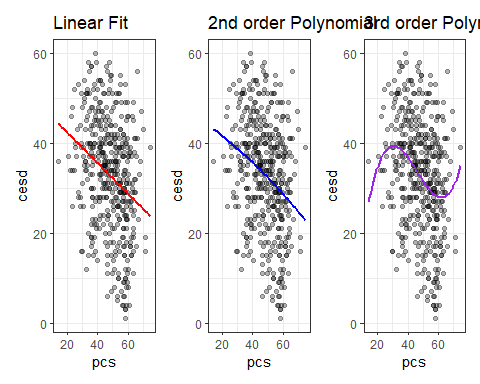
A polynomial in the variable x of degree D is a linear combination of the powers of x up to D. For example:

* Linear:
* Quadratic:
* Cubic:
* Quartic:

Fitting such a model creates a **polynomial regression**.

## Plotting the Polynomials

p1 <- ggplot(help1, aes(x = pcs, y = cesd)) +  
 geom\_point(alpha = 0.3) +   
 geom\_smooth(formula = y ~ x, method = "lm",   
 col = "red", se = FALSE) +   
 labs(title = "Linear Fit")  
  
p2 <- ggplot(help1, aes(x = pcs, y = cesd)) +  
 geom\_point(alpha = 0.3) +   
 geom\_smooth(formula = y ~ poly(x, 2), method = "lm",  
 col = "blue", se = FALSE) +  
 labs(title = "2nd order Polynomial")  
  
p3 <- ggplot(help1, aes(x = pcs, y = cesd)) +  
 geom\_point(alpha = 0.3) +   
 geom\_smooth(formula = y ~ poly(x, 3), method = "lm",  
 col = "purple", se = FALSE) +  
 labs(title = "3rd order Polynomial")  
  
p1 + p2 + p3



## Adding a polynomial in pcs

Can we predict cesd with a polynomial in pcs?

Yes, with ols() and pol(), as follows:

fitA <- ols(cesd ~ pcs, data = help1, x = TRUE, y = TRUE)  
fitA\_2 <- ols(cesd ~ pol(pcs,2), data = help1, x = TRUE, y = TRUE)  
fitA\_3 <- ols(cesd ~ pol(pcs,3), data = help1, x = TRUE, y = TRUE)

With lm(), we use poly() instead of pol()…

lmfitA <- lm(cesd ~ pcs, data = help1)  
lmfitA\_2 <- lm(cesd ~ poly(pcs,2), data = help1)  
lmfitA\_3 <- lm(cesd ~ poly(pcs,3), data = help1)

## Raw vs. Orthogonal Polynomials

Predict cesd using pcs with a “raw polynomial of degree 2.”

(temp1 <- lm(cesd ~ pcs + I(pcs^2), data = help1))

Call:  
lm(formula = cesd ~ pcs + I(pcs^2), data = help1)  
  
Coefficients:  
(Intercept) pcs I(pcs^2)   
 46.400713 -0.213627 -0.001356

Predicted cesd for pcs = 40 is

cesd = 46.400713 - 0.213627 (40) - 0.001356 (40^2)  
 = 46.400713 - 8.545080 - 2.169600  
 = 35.686

## Does the raw polynomial match our expectations?

temp1 <- lm(cesd ~ pcs + I(pcs^2), data = help1)  
  
augment(temp1, newdata = tibble(pcs = 40)) |>   
 gt() |> tab\_options(table.font.size = 24)

| pcs | .fitted |
| --- | --- |
| 40 | 35.6856 |

This matches our “by hand” calculation.

* But it turns out most regression models use *orthogonal* rather than raw polynomials…

## Fitting an Orthogonal Polynomial

Predict cesd using pcs with an *orthogonal* polynomial of degree 2.

(temp2 <- lm(cesd ~ poly(pcs,2), data = help1))

Call:  
lm(formula = cesd ~ poly(pcs, 2), data = help1)  
  
Coefficients:  
 (Intercept) poly(pcs, 2)1 poly(pcs, 2)2   
 32.848 -77.876 -3.944

This looks very different from our previous version of the model. What happens when we make a prediction, though?

## Orthogonal Polynomial Model Prediction

Remember that in our raw polynomial model, our “by hand” and “using R” calculations each predicted cesd for a subject with pcs = 40 to be 35.686.

What happens with the orthogonal polynomial model temp2?

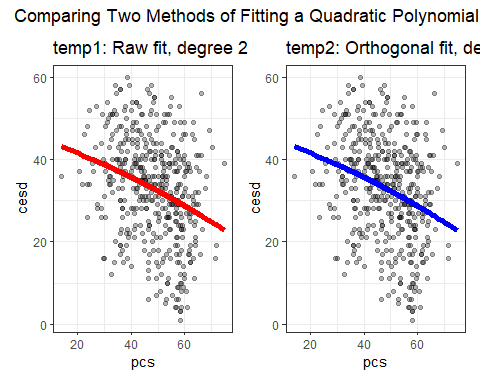
augment(temp2, newdata = data.frame(pcs = 40)) |>   
 gt() |> tab\_options(table.font.size = 24)

| pcs | .fitted |
| --- | --- |
| 40 | 35.6856 |

* No change in the prediction.

## Fits of raw vs orthogonal polynomials

temp1\_aug <- augment(temp1, help1)  
temp2\_aug <- augment(temp2, help1)  
  
p1 <- ggplot(temp1\_aug, aes(x = pcs, y = cesd)) +  
 geom\_point(alpha = 0.3) +  
 geom\_line(aes(x = pcs, y = .fitted), col = "red", linewidth = 2) +  
 labs(title = "temp1: Raw fit, degree 2")  
  
p2 <- ggplot(temp2\_aug, aes(x = pcs, y = cesd)) +  
 geom\_point(alpha = 0.3) +  
 geom\_line(aes(x = pcs, y = .fitted), col = "blue", linewidth = 2) +  
 labs(title = "temp2: Orthogonal fit, degree 2")  
  
p1 + p2 +   
 plot\_annotation(title = "Comparing Two Methods of Fitting a Quadratic Polynomial")



* The two models are, in fact, identical.

## Why use orthogonal polynomials?

* The main reason is to avoid having to include powers of our predictor that are highly collinear.
* Variance Inflation Factor assesses collinearity…

rms::vif(temp1) ## from rms package

pcs I(pcs^2)   
54.66793 54.66793

* Orthogonal polynomial terms are uncorrelated…

rms::vif(temp2)

poly(pcs, 2)1 poly(pcs, 2)2   
 1 1

## Why orthogonal polynomials?

An **orthogonal polynomial** sets up a model design matrix and then scales those columns so that each column is uncorrelated with the others. The tradeoff is that the raw polynomial is a lot easier to explain in terms of a single equation in the simplest case.

Actually, we’ll often use splines instead of polynomials, which are more flexible and require less maintenance, but at the cost of pretty much requiring you to focus on visualizing their predictions rather than their equations.

## fitA with a cubic polynomial

dd <- datadist(help1); options(datadist = "dd")  
  
fitA\_3 <- ols(cesd ~ pol(pcs,3), data = help1, x = TRUE, y = TRUE)  
  
fitA\_3$coefficients

Intercept pcs pcs^2 pcs^3   
-1.340758e+01 4.132348e+00 -1.009667e-01 7.268386e-04

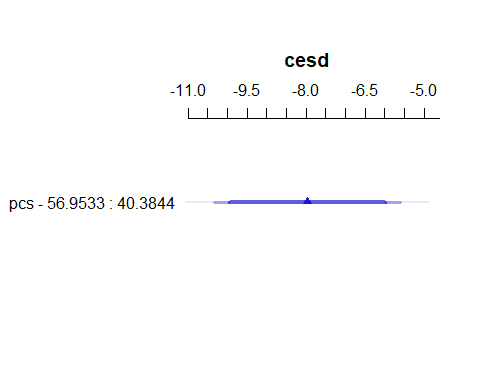
## Our model fitA\_3

fitA\_3

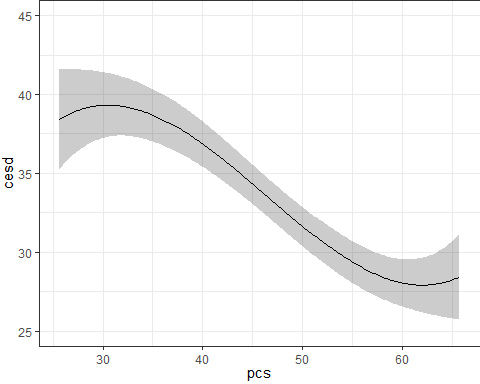
Linear Regression Model  
  
ols(formula = cesd ~ pol(pcs, 3), data = help1, x = TRUE, y = TRUE)  
  
 Model Likelihood Discrimination   
 Ratio Test Indexes   
Obs 453 LR chi2 48.70 R2 0.102   
sigma11.8991 d.f. 3 R2 adj 0.096   
d.f. 449 Pr(> chi2) 0.0000 g 4.556   
  
Residuals  
  
 Min 1Q Median 3Q Max   
-27.5245 -8.2651 0.7988 8.9004 27.4480   
  
 Coef S.E. t Pr(>|t|)  
Intercept -13.4076 22.8605 -0.59 0.5578   
pcs 4.1323 1.5825 2.61 0.0093   
pcs^2 -0.1010 0.0354 -2.85 0.0046   
pcs^3 0.0007 0.0003 2.83 0.0049

## Effect Sizes in fitA\_3

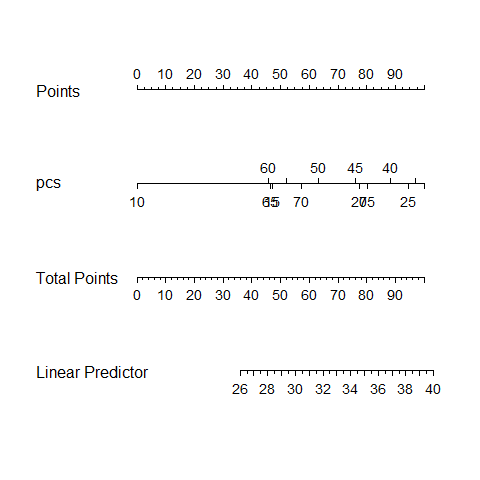
plot(summary(fitA\_3))



ggplot(Predict(fitA\_3, conf.int = 0.90))



plot(nomogram(fitA\_3))



## Fitting fitB including a polynomial

dd <- datadist(help1)  
options(datadist = "dd")  
  
fitB\_3 <- ols(cesd ~ pol(pcs,3) + subst + pss\_fr + sex,   
 data = help1, x = TRUE, y = TRUE)  
  
fitB\_3$coefficients

Intercept pcs pcs^2 pcs^3 subst=cocaine   
 5.2983256376 3.2271532761 -0.0794837993 0.0005770243 -3.8581390102   
 subst=heroin pss\_fr sex=male   
 0.0455051022 -0.5127744954 -4.5981834492

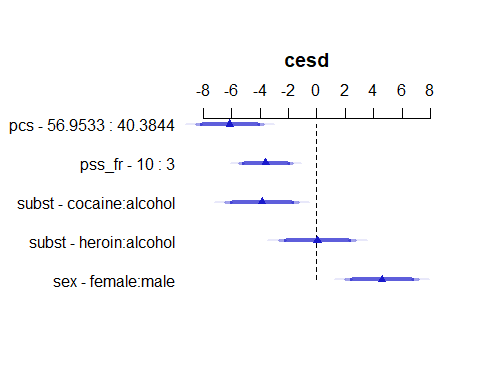
## Contents of fitB\_3?

fitB\_3

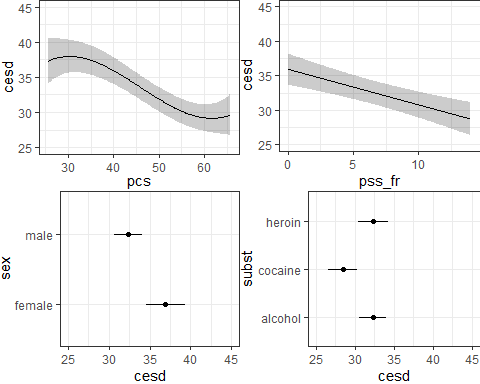
Linear Regression Model  
  
ols(formula = cesd ~ pol(pcs, 3) + subst + pss\_fr + sex, data = help1,   
 x = TRUE, y = TRUE)  
  
 Model Likelihood Discrimination   
 Ratio Test Indexes   
Obs 453 LR chi2 81.80 R2 0.165   
sigma11.5236 d.f. 7 R2 adj 0.152   
d.f. 445 Pr(> chi2) 0.0000 g 5.808   
  
Residuals  
  
 Min 1Q Median 3Q Max   
-29.2871 -7.9098 0.9439 7.8319 27.6486   
  
 Coef S.E. t Pr(>|t|)  
Intercept 5.2983 22.4423 0.24 0.8135   
pcs 3.2272 1.5457 2.09 0.0374   
pcs^2 -0.0795 0.0346 -2.30 0.0221   
pcs^3 0.0006 0.0003 2.30 0.0217   
subst=cocaine -3.8581 1.3015 -2.96 0.0032   
subst=heroin 0.0455 1.3589 0.03 0.9733   
pss\_fr -0.5128 0.1371 -3.74 0.0002   
sex=male -4.5982 1.3073 -3.52 0.0005

## Effect Sizes in fitB\_3

plot(summary(fitB\_3))

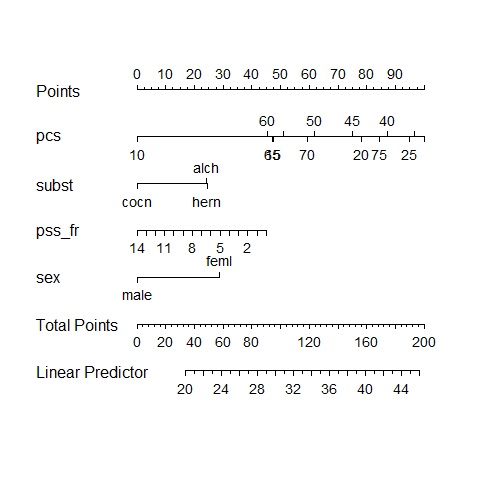


ggplot(Predict(fitB\_3, conf.int = 0.90))



## A Nomogram for fitB\_3

plot(nomogram(fitB\_3, abbrev = TRUE))



# Non-Linear Terms: Splines

## Types of Splines

* A **linear spline** is a continuous function formed by connecting points (called **knots** of the spline) by line segments.
* A **restricted cubic spline** is a way to build highly complicated curves into a regression equation in a fairly easily structured way.
* A restricted cubic spline is a series of polynomial functions joined together at the knots.
  + Such a spline gives us a way to flexibly account for non-linearity without over-fitting the model.

## How complex should our spline be?

Restricted cubic splines can fit many different types of non-linearities. Specifying the number of knots is all you need to do in R to get a reasonable result from a restricted cubic spline.

The most common choices are 3, 4, or 5 knots.

* 3 Knots, 2 degrees of freedom, allows the curve to “bend” once.
* 4 Knots, 3 degrees of freedom, lets the curve “bend” twice.
* 5 Knots, 4 degrees of freedom, lets the curve “bend” three times.

## Restricted Cubic Splines with ols

Let’s consider a restricted cubic spline model for cesd based on pcs with:

* 3 knots in fitC3, 4 knots in fitC4, and 5 knots in fitC5

dd <- datadist(help1)  
options(datadist = "dd")  
  
fitC3 <- ols(cesd ~ rcs(pcs, 3),   
 data = help1, x = TRUE, y = TRUE)  
fitC4 <- ols(cesd ~ rcs(pcs, 4),   
 data = help1, x = TRUE, y = TRUE)  
fitC5 <- ols(cesd ~ rcs(pcs, 5),  
 data = help1, x = TRUE, y = TRUE)

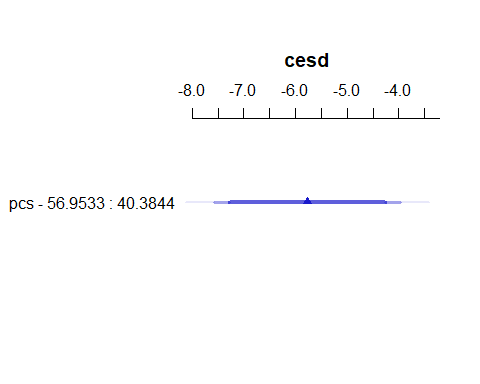
## Model fitC3 (3-knot spline in pcs)

fitC3

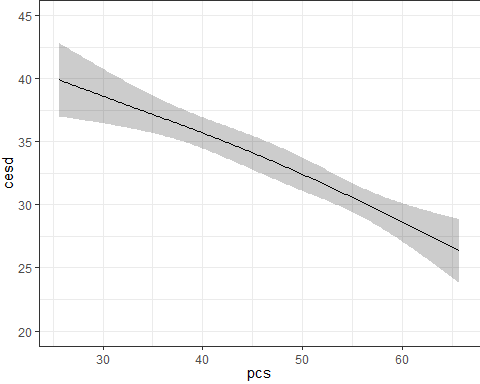
Linear Regression Model  
  
ols(formula = cesd ~ rcs(pcs, 3), data = help1, x = TRUE, y = TRUE)  
  
 Model Likelihood Discrimination   
 Ratio Test Indexes   
Obs 453 LR chi2 40.79 R2 0.086   
sigma11.9901 d.f. 2 R2 adj 0.082   
d.f. 450 Pr(> chi2) 0.0000 g 4.206   
  
Residuals  
  
 Min 1Q Median 3Q Max   
-28.3462 -7.7005 0.5098 8.6376 29.8454   
  
 Coef S.E. t Pr(>|t|)  
Intercept 47.3631 4.7053 10.07 <0.0001   
pcs -0.2908 0.1187 -2.45 0.0146   
pcs' -0.0624 0.1363 -0.46 0.6471

## Effect Sizes in fitC3

plot(summary(fitC3))

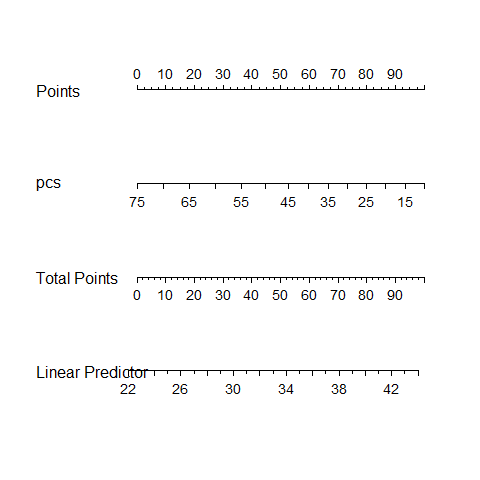


ggplot(Predict(fitC3, conf.int = 0.90))



## A Nomogram for fitC3

plot(nomogram(fitC3, abbrev = TRUE))



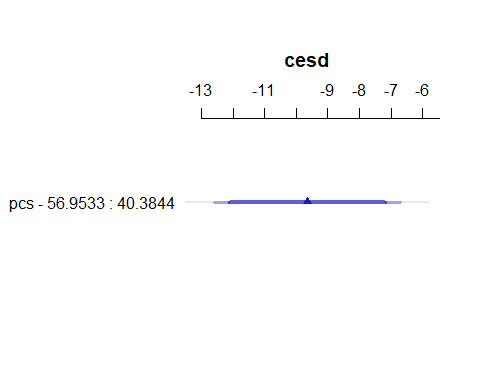
## Model fitC4 (4-knot spline in pcs)

fitC4

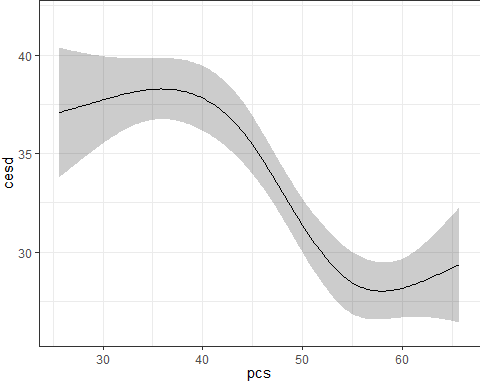
Linear Regression Model  
  
ols(formula = cesd ~ rcs(pcs, 4), data = help1, x = TRUE, y = TRUE)  
  
 Model Likelihood Discrimination   
 Ratio Test Indexes   
Obs 453 LR chi2 51.31 R2 0.107   
sigma11.8648 d.f. 3 R2 adj 0.101   
d.f. 449 Pr(> chi2) 0.0000 g 4.590   
  
Residuals  
  
 Min 1Q Median 3Q Max   
-28.3147 -8.2830 0.8559 8.8866 26.5458   
  
 Coef S.E. t Pr(>|t|)  
Intercept 33.3298 6.5742 5.07 <0.0001   
pcs 0.1464 0.1856 0.79 0.4308   
pcs' -1.4383 0.4497 -3.20 0.0015   
pcs'' 6.2561 1.9076 3.28 0.0011

## Effect Sizes in fitC4

plot(summary(fitC4))

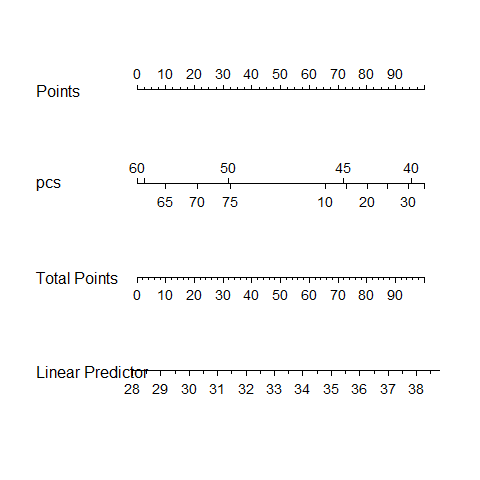


ggplot(Predict(fitC4, conf.int = 0.90))



## A Nomogram for fitC4

plot(nomogram(fitC4, abbrev = TRUE))



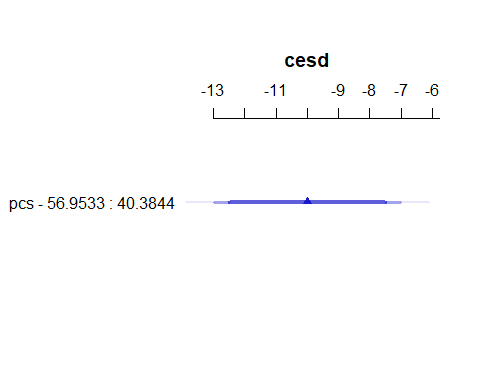
## Model fitC5 (5-knot spline in pcs)

fitC5

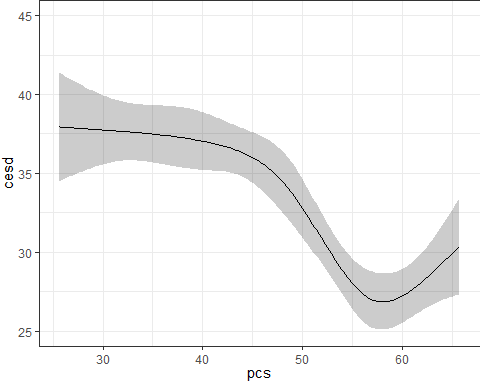
Linear Regression Model  
  
ols(formula = cesd ~ rcs(pcs, 5), data = help1, x = TRUE, y = TRUE)  
  
 Model Likelihood Discrimination   
 Ratio Test Indexes   
Obs 453 LR chi2 54.64 R2 0.114   
sigma11.8345 d.f. 4 R2 adj 0.106   
d.f. 448 Pr(> chi2) 0.0000 g 4.744   
  
Residuals  
  
 Min 1Q Median 3Q Max   
-29.396 -7.928 1.016 8.762 26.974   
  
 Coef S.E. t Pr(>|t|)  
Intercept 39.0631 7.8282 4.99 <0.0001   
pcs -0.0436 0.2332 -0.19 0.8517   
pcs' -0.2952 1.0079 -0.29 0.7697   
pcs'' -3.1835 4.8079 -0.66 0.5082   
pcs''' 14.4216 8.3721 1.72 0.0857

## Effect Sizes in fitC5

plot(summary(fitC5))

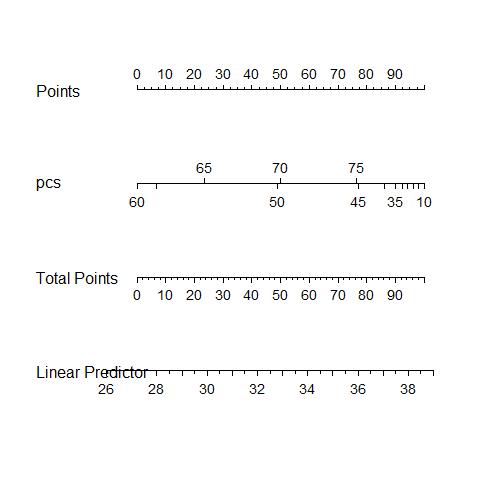


ggplot(Predict(fitC5, conf.int = 0.90))



## A Nomogram for fitC5

plot(nomogram(fitC5, abbrev = TRUE))



## Fitting fitB including a 5-knot RCS

dd <- datadist(help1)  
options(datadist = "dd")  
  
fitB5 <- ols(cesd ~ rcs(pcs,5) + subst + pss\_fr + sex,   
 data = help1, x = TRUE, y = TRUE)  
  
fitB5$coefficients

Intercept pcs pcs' pcs'' pcs'''   
 48.4263870 -0.1151961 0.2519557 -4.9683530 15.6487400   
subst=cocaine subst=heroin pss\_fr sex=male   
 -3.7580871 -0.1528491 -0.4985050 -4.7527659

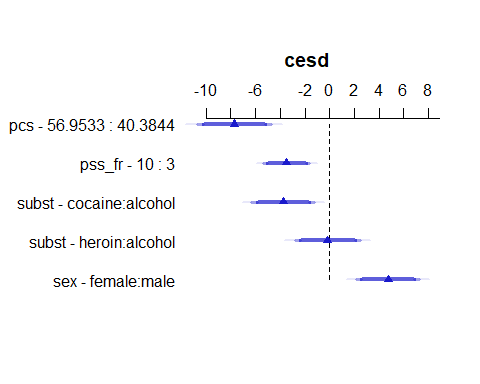
## Contents of fitB5?

fitB5

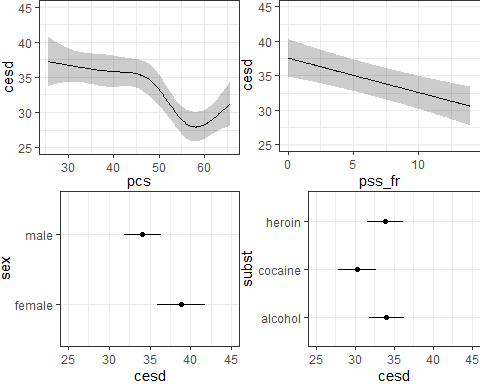
Linear Regression Model  
  
ols(formula = cesd ~ rcs(pcs, 5) + subst + pss\_fr + sex, data = help1,   
 x = TRUE, y = TRUE)  
  
 Model Likelihood Discrimination   
 Ratio Test Indexes   
Obs 453 LR chi2 86.99 R2 0.175   
sigma11.4707 d.f. 8 R2 adj 0.160   
d.f. 444 Pr(> chi2) 0.0000 g 5.991   
  
Residuals  
  
 Min 1Q Median 3Q Max   
-31.189 -8.141 1.309 7.755 28.766   
  
 Coef S.E. t Pr(>|t|)  
Intercept 48.4264 7.8123 6.20 <0.0001   
pcs -0.1152 0.2270 -0.51 0.6122   
pcs' 0.2520 0.9852 0.26 0.7983   
pcs'' -4.9684 4.6941 -1.06 0.2904   
pcs''' 15.6487 8.1700 1.92 0.0561   
subst=cocaine -3.7581 1.2962 -2.90 0.0039   
subst=heroin -0.1528 1.3561 -0.11 0.9103   
pss\_fr -0.4985 0.1369 -3.64 0.0003   
sex=male -4.7528 1.3063 -3.64 0.0003

## Effect Sizes in fitB5

plot(summary(fitB5))

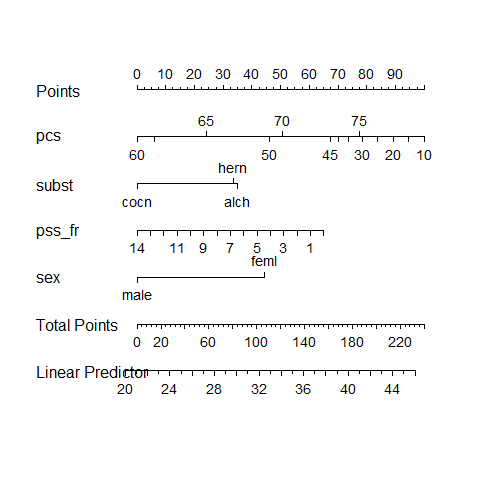


ggplot(Predict(fitB5, conf.int = 0.90))



## A Nomogram for fitB5

plot(nomogram(fitB5, abbrev = TRUE))



# What if you’re doing a logistic regression?

## Predicting Pr(CESD>15) with a spline

dd <- datadist(help1)  
options(datadist = "dd")  
  
fitD5 <- lrm(cesd\_hi ~ rcs(pcs,5) + subst + pss\_fr + sex,   
 data = help1, x = TRUE, y = TRUE)  
  
fitD5$coefficients

Intercept pcs pcs' pcs'' pcs'''   
 6.62332347 -0.05399131 0.03371332 -0.87238631 3.36265431   
subst=cocaine subst=heroin pss\_fr sex=male   
 -1.10386515 -0.40683747 -0.08278496 -0.22546869

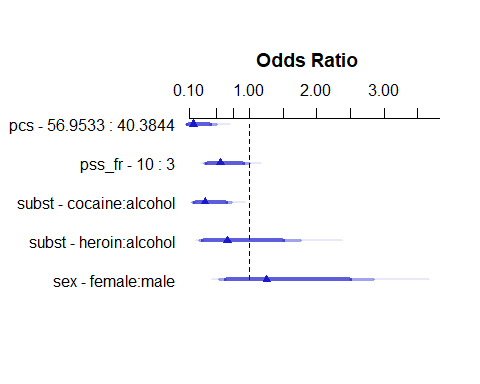
## Contents of fitD5?

fitD5

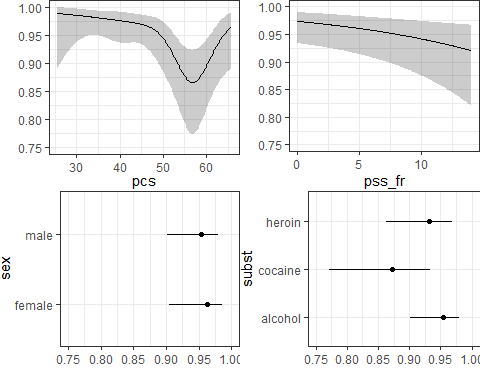
Logistic Regression Model  
  
lrm(formula = cesd\_hi ~ rcs(pcs, 5) + subst + pss\_fr + sex, data = help1,   
 x = TRUE, y = TRUE)  
  
 Model Likelihood Discrimination Rank Discrim.   
 Ratio Test Indexes Indexes   
Obs 453 LR chi2 42.01 R2 0.184 C 0.778   
 0 46 d.f. 8 R2(8,453)0.072 Dxy 0.555   
 1 407 Pr(> chi2) <0.0001 R2(8,124)0.240 gamma 0.555   
max |deriv| 3e-05 Brier 0.083 tau-a 0.102   
  
 Coef S.E. Wald Z Pr(>|Z|)  
Intercept 6.6233 4.8970 1.35 0.1762   
pcs -0.0540 0.1393 -0.39 0.6984   
pcs' 0.0337 0.4794 0.07 0.9439   
pcs'' -0.8724 1.9976 -0.44 0.6623   
pcs''' 3.3627 3.0965 1.09 0.2775   
subst=cocaine -1.1039 0.4039 -2.73 0.0063   
subst=heroin -0.4068 0.4956 -0.82 0.4117   
pss\_fr -0.0828 0.0412 -2.01 0.0447   
sex=male -0.2255 0.4187 -0.54 0.5902

## Effect Sizes in fitD5

plot(summary(fitD5))



ggplot(Predict(fitD5, conf.int = 0.90, fun = plogis))



## A Nomogram for fitD5

plot(nomogram(fitD5, abbrev = TRUE, fun = plogis, funlabel = "Pr(CESD > 15)"))

