

432 Class 01 Slides

thomaselove.github.io/432

2022-01-11

Today's Agenda

- ① Mechanics of the course
- ② Why I write dates the way I do
- ③ Data organization in spreadsheets
- ④ Naming Things and Getting Organized
- ⑤ Building a Table 1 (see Course Notes, Chapter 1)

There's a (pre-recorded) "live code" demo, too.

Welcome to 432.

Everything is at <https://thomaselove.github.io/432/>

- Syllabus
- Calendar
 - with all deadlines, and links to class READMEs
- Course Notes
- Details on Assignments to come (+ next slide)
- R and Data
 - Updating / Installing R, RStudio, necessary R Packages
 - Review / Learn some R Basics (also see 431 web site)
- Sources
 - Books, Articles, YouTube series, etc.
- Links to Canvas, Piazza and Contact Us

Assignments

Every deliverable for the entire semester is listed in the Calendar, except for the Welcome to 432 Survey, which at least 30 of you've done, and if you haven't, visit <https://bit.ly/432-2022-welcome-survey>.

- Two projects
 - Project 1 (use publicly available data for linear & logistic models)
 - ① Proposal due 2022-01-31 (data selection, cleaning, exploration)
 - ② Final Materials due 2022-03-04 (analyses, discussion)
 - Project 2 (use almost any data you like and analyze it well)
- Two Quizzes (Quiz 1 due 2022-02-21, Quiz 2 due 2022-04-18)
 - Multiple choice and short answer, mostly, taken via a Google Form
- Six labs
 - Labs will be posted before our next class. Lab 01 is due Monday 2022-01-24 at 9 PM.
- Ten minute papers
 - First is due 2022-01-19. These actually take about 5 minutes each.

Syllabus and Instructions will provide more information on grading/feedback.

The Spring 2022 Teaching Assistants for 432 are:

- Stephanie Merlino Barr, PhD student in Clinical Translational Science
- Wyatt Bensken, PhD student in Epidemiology & Biostatistics
- Ali Elsharkawi, MS student in Clinical Research
- Shiying Liu, PhD student in Epidemiology & Biostatistics
- Marie Michenkova, MS student in Biomedical Health Informatics
- Julia Yang Payne, PhD student in Clinical & Translational Science
- Monika Strah, MS student in Epidemiology & Biostatistics
- Yanning Wu, PhD student in Epidemiology & Biostatistics

All return from working with students in 431 this past Fall, and I couldn't be more grateful for their energy and effort. Learn more about the TAs in Section 6 of the Syllabus.

Getting Help

- Piazza is the location for discussion about the class. I follow it closely.
- We have 8 teaching assistants volunteering their time to help you.
- TAs will hold Office Hours beginning next Monday 2021-01-17 via Zoom, and the details will be available on Canvas (see Announcements) and our shared Google Drive.
- Dr. Love is available before and (especially) after class.
- Email Dr. Love directly only if you have a matter you need to discuss with him specifically. He's at Thomas dot Love at case dot edu.

We WELCOME your questions/comments/corrections/thoughts!

Tools You Will Definitely Use in this Class

- **Course Website** (see the bottom of this slide) especially the Calendar
 - Each class has a README (announcements, reminders, etc.) plus slides
- **R, RStudio and R Markdown** for, well, everything
- **Canvas** for access to Zoom meetings *and 432 recordings*, submission of most assignments
- **Google Drive via CWRU** for *recordings from 431*, forms (Minute Papers/Surveys/Quizzes) and receiving feedback on labs, projects, and Minute Papers
- **Piazza** is our discussion board. It's a moderated place to ask questions, answer questions of your colleagues, and get help fast. You don't have to pay to use it.
- **Zoom** for class sessions and for TA office hours

Some source materials are **password-protected**. What is the password?



An approximate answer to the right
problem is worth a good deal more
than an exact answer to an
approximate problem.

— *John Tukey* —

AZ QUOTES

How To Write Dates (<https://xkcd.com/1179/>)

PUBLIC SERVICE ANNOUNCEMENT:

OUR DIFFERENT WAYS OF WRITING DATES AS NUMBERS CAN LEAD TO ONLINE CONFUSION. THAT'S WHY IN 1988 ISO SET A GLOBAL STANDARD NUMERIC DATE FORMAT.

THIS IS **THE** CORRECT WAY TO WRITE NUMERIC DATES:

2013-02-27

THE FOLLOWING FORMATS ARE THEREFORE DISCOURAGED:

02/27/2013 02/27/13 27/02/2013 27/02/13

20130227 2013.02.27 27.02.13 27-02-13

27.2.13 2013. II. 27. $2\frac{1}{2}$ -13 2013.158904109

MMXIII-II-XXVII MMXIII $\frac{LVII}{CCCLXV}$ 1330300800

$((3+3)\times(111+1)-1)\times3/3-1/3^3$ 2013 Mississ

10/11011/1101 02/27/20/13 

$\begin{matrix} 2 & 3 & 1 & 4 \\ \hline 5 & 6 & 7 & 8 \end{matrix}$

Tidy Data (Wickham)

"A huge amount of effort is spent cleaning data to get it ready for analysis, but there has been little research on how to make data cleaning as easy and effective as possible. . . .

Tidy datasets are easy to manipulate, model and visualize, and have a specific structure: each variable is a column, each observation is a row, and each type of observational unit is a table.

This framework makes it easy to tidy messy datasets because only a small set of tools are needed to deal with a wide range of un-tidy datasets. This structure also makes it easier to develop tidy tools for data analysis, tools that both input and output tidy datasets. The advantages of a consistent data structure and matching tools are demonstrated with a case study free from mundane data manipulation chores."

<https://www.jstatsoft.org/article/view/v059i10>

“Data Tidying” presentation in *R for Data Science*

- Defines tidy data
- Demonstrates methods for tidying messy data in R

Read Sections

- 5 (Data transformation),
- 10 (Tibbles), 11 (Data import) and, especially, 12 (Tidy data)

<https://r4ds.had.co.nz/>

Data Organization in Spreadsheets (Broman & Woo)

- Create a data dictionary.
 - Jeff Leek has good thoughts on this in “How to Share Data with a Statistician” at <https://github.com/jtleek/datasharing>
 - Shannon Ellis and Jeff Leek’s preprint “How to Share data for Collaboration” touches on many of the same points at <https://peerj.com/preprints/3139v5.pdf>

We want:

- ① The raw data.
- ② A tidy data set.
- ③ A codebook describing each variable and its values in the tidy data set.
- ④ An explicit and exact recipe describing how you went from 1 to 2 and 3.

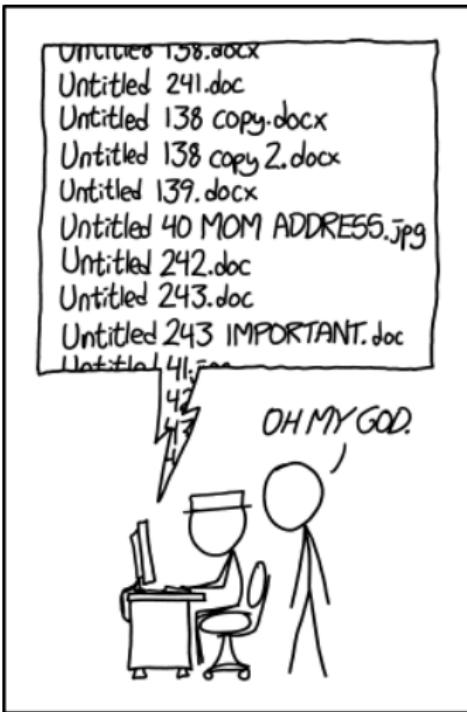
Data Organization in Spreadsheets: Be Consistent

- Consistent codes for categorical variables.
 - Either “M” or “Male” but not both at the same time.
 - Make it clear enough to reduce dependence on a codebook.
 - No spaces or special characters other than `_` in category names.
- Consistent fixed codes for missing values.
 - `NA` is the most convenient R choice.
- Consistent variable names
 - In R, I'll use `clean_names` from the `janitor` package to turn everything into `snake_case`.
 - In R, start your variable names with letters. No spaces, no special characters other than `_`.
- Consistent subject / record identifiers
 - And if you're building a `.csv` in Excel, don't use `ID` as the name of that identifier.
- Consistent data layouts across multiple files.

What Goes in a Cell?

- Make your data a rectangle.
 - Each row represents a record (sometimes a subject).
 - Each column represents a variable.
 - First column is a unique identifier for each record.
- No empty cells.
- One Thing in each cell.
- No calculations in the raw data
- No font colors
- No highlighting

Naming Files is Hard (<https://xkcd.com/1459/>)



PROTIP: NEVER LOOK IN SOMEONE
ELSE'S DOCUMENTS FOLDER.

How To Name Files

NO

myabstract.docx

Joe's Filenames Use Spaces and Punctuation.xlsx

figure 1.png

fig 2.png

JW7d^(2sl@deletethisandyourcareerisoverWx2*.txt

YES

2014-06-08_abstract-for-sla.docx

joes-filenames-are-getting-better.xlsx

fig01_scatterplot-talk-length-vs-interest.png

fig02_histogram-talk-attendance.png

1986-01-28_raw-data-from-challenger-o-rings.txt

Data Organization in Spreadsheets: Use consistent, strong file names.

Jenny Bryan's advice on "Naming Things" hold up well. There's a full presentation at SpeakerDeck.

Good file names:

- are machine readable (easy to search, easy to extract info from names)
- are human readable (name contains content information, so it's easy to figure out what something is based on its name)
- play well with default ordering (something numeric first, left padded with zeros as needed, use ISO 8601 standard for dates)

Avoid: spaces, punctuation, accented characters, case sensitivity

left pad other numbers with zeros

```
01_marshall-data.r  
02_pre-dea-filtering.r  
03_dea-with-limma-voom.r  
04_explore-dea-results.r  
90_limma-model-term-name-fiasco.r  
helper01_load-counts.r  
helper02_load-exp-des.r  
helper03_load-focus-statinf.r  
helper04_extract-and-tidy.r
```

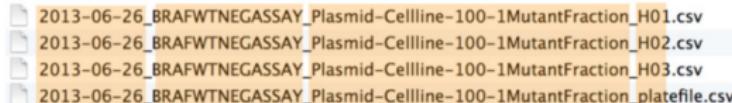
if you don't left pad, you get this:

```
10_final-figs-for-publication.R  
1_data-cleaning.R  
2_fit-model.R
```

which is just sad

Jenny Bryan: Deliberate Use of Delimiters

Deliberately use delimiters to make things easy to compute on and make it easy to recover meta-data from the filenames.



```
> flist <- list.files(pattern = "Plasmid") %>% head  
  
> stringr::str_split_fixed(flist, "[_\\.]", 5)  
[1,] [1] [2] [3] [4] [5]  
[1,] "2013-06-26" "BRAFWTNEGASSAY" "Plasmid-Cellline-100-1MutantFraction" "A01" "csv"  
[2,] "2013-06-26" "BRAFWTNEGASSAY" "Plasmid-Cellline-100-1MutantFraction" "A02" "csv"  
[3,] "2013-06-26" "BRAFWTNEGASSAY" "Plasmid-Cellline-100-1MutantFraction" "A03" "csv"  
[4,] "2013-06-26" "BRAFWTNEGASSAY" "Plasmid-Cellline-100-1MutantFraction" "B01" "csv"  
[5,] "2013-06-26" "BRAFWTNEGASSAY" "Plasmid-Cellline-100-1MutantFraction" "B02" "csv"  
[6,] "2013-06-26" "BRAFWTNEGASSAY" "Plasmid-Cellline-100-1MutantFraction" "B03" "csv"
```

“_” underscore used to delimit units of meta-data I want later

“-” hyphen used to delimit words so my eyes don't bleed

Don't get too cute.



Jenny Bryan

@JennyBryan

Following



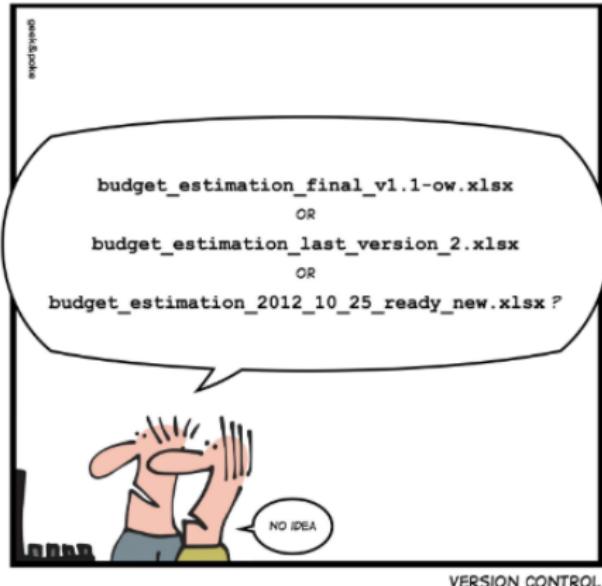
The Golden Rule of Naming Files and Other Things:

Thou shalt get only as creative with names as thy own skill with regular expressions.

11:31 PM - 10 Dec 2016

Goal: Avoid this...

SIMPLY EXPLAINED



Idea from Jen Simmons and John Albin Wilkins during episode #40 of "Web Ahead" about Git:
<http://SbyS.tv/webahead/40>

Be organized

do this as you go, not "tomorrow"

but also don't fret over past mistakes
raise the bar for *new* work

Don't spend a lot of time bemoaning or cleaning up past ills. Strive to improve this sort of thing going forward.

“Good Enough Practices in Scientific Computing”

- ① Save the raw data.
- ② Ensure that raw data is backed up in more than one location.
- ③ Create the data you wish to see in the world (the data you wish you had received.)
- ④ Create analysis-friendly, tidy data.
- ⑤ Record all of the steps used to process data.
- ⑥ Anticipate the need for multiple tables, and use a unique identifier for every record.

<http://bit.ly/good-enuff>

Lots of great advice here on software, collaboration and project organization.

Something Practical: Building Table 1

An Original Clinical Investigation

Original Investigation | Cardiology



January 18, 2019

Incidence, Risk Factors, and Outcomes Associated With In-Hospital Acute Myocardial Infarction

Steven M. Bradley, MD, MPH^{1,2}; Joleen A. Borgerding, MS³; G. Blake Wood, MS³; [et al](#)

[» Author Affiliations](#) | [Article Information](#)

JAMA Netw Open. 2019;2(1):e187348. doi:10.1001/jamanetworkopen.2018.7348

Key Points

Question What are the incidence, risk factors, and outcomes associated with in-hospital acute myocardial infarction (AMI)?

Findings This cohort study of 1.3 million patients hospitalized in US Veterans Health Administration facilities found an incidence of in-hospital AMI of 4.27 per 1000 admissions, and risk factors associated with in-hospital AMI included history of coronary artery disease, elevated heart rate, low hemoglobin level, and elevated white blood cell count. Compared with a matched control group, mortality was significantly higher for in-hospital AMI.

Meaning In-hospital AMI is common and is associated with prior cardiovascular disease, physiological disturbances, and poor survival.

Part of Bradley et al.'s Table 1

Table 1. Patient Characteristics on Admission and In-Hospital Variables Prior to Event for Matched In-Hospital Acute Myocardial Infarction Cases and Controls

Characteristic	No. (%)			
	Total (N = 1374)	Cases (n = 687)	Controls (n = 687)	P Value
Age, mean (SD), y	73.3 (10.2)	73.3 (10.1)	73.4 (10.3)	.80
Male	1343 (97.7)	677 (98.5)	666 (96.9)	.05
White race/ethnicity	1073 (78.1)	546 (79.5)	527 (76.7)	.22
Married	666 (48.5)	356 (51.8)	310 (45.1)	.01
Location				
Intensive care unit	251 (18.3)	186 (27.1)	65 (9.5)	
Medical bed	1026 (74.7)	446 (64.9)	580 (84.4)	<.001
Other	97 (7.1)	55 (8.0)	42 (6.1)	

Table Creation Instructions, JAMA: linked here

Creating a Table

Use the table editor of the word processing software to build a table. Do not embed tables as images in the manuscript file or upload tables in image formats. Regardless of which program is used, each piece of data needs to be contained in its own cell in the table. Tables should be single-spaced.

Avoid creating tables using spaces or tabs. For accepted manuscripts, tables created with spaces, tabs, and/or hard returns must be retyped during the editing process, creating delays and opportunities for error. Do not try to align cells with hard returns or extra spaces. Similarly, no cell should contain a hard return or tab. Although individual empty cells are acceptable in a table, be sure there are no empty columns.

Place each row of data in a separate row of cells:

Table 1. Title

Treatment	Group A	Group B
Medical	500	510
Surgical	500	490

Note that numbers and percentages are presented in the same cell, and measures of variability are in the same cell as their corresponding statistic:

Table 2. Title

Characteristics	Group A (n = 50)	Group B (n = 50)	Relative Risk (95% CI)
Women, No. (%)	25 (50)	20 (40)	1.25 (1.11-1.57)
Age, mean (SD), y	35 (8)	37 (7)	0.98 (0.92-1.05)

the rows. In Table 3, the final column lists the P value for the overall age comparison:

Table 3. Title

Age, y	Blood Pressure, mm Hg	P Value
18-34	120/75	
35-50	110/80	.08
51-80	125/82	

The table should be constructed such that comparisons between groups read horizontally (see Tables 1 and 2).

Do not draw lines or rules—the table grid feature will display the outlines of each cell.

Data Presentation

When presenting percentages, include numbers (numerator, and denominator if necessary). Include variability where applicable (eg, mean [SD] or median [interquartile range]).

All P values should be reported as exact numbers to 2 digits past the decimal point, regardless of significance, unless they are lower than .01, in which case they should be presented to 3 digits. Express any P values lower than .001 as $P < .001$. P values can never equal 0 or 1.

Footnotes

Be sure to explain empty cells. Also, if necessary add a footnote to explain why numbers may not sum to group totals or percentages do not total 100. List abbreviations for the table in a footnote and use superscript letters to mark each footnote (a,b,c, etc).

Questions

For questions on table construction or formatting, contact Stacy Christian, stacy.christian@jama.jamapublications.org.

A Data Set

The `bradley.csv` data set on our web site is simulated, but consists of 1,374 observations (687 Cases and 687 Controls) containing:

- a subject identification code, in `subject`
- status (case or control)
- age (in years)
- sex (Male or Female)
- race/ethnicity (white or non-white)
- married (1 = yes or 0 = no)
- location (ICU, bed, other)

The `bradley.csv` data closely match the summary statistics provided in Table 1 of the Bradley et al. article. Our job is to recreate that part of Table 1, as best as we can.

The bradley.csv data (first 5 rows)

- The bradley_sim.md file on our web site shows you how I simulated the data.

	A	B	C	D	E	F	G
1	subject	status	age	sex	race_eth	married	location
2	1	Control	64	Male	white	1	Bed
3	2	Case	70	Male	white	1	ICU
4	3	Control	68	Male	white	0	Bed
5	4	Control	76	Male	white	1	Bed
6	5	Control	70	Male	white	1	Bed

To “Live” Coding

On our web site (Data and Code + Class 01 materials)

- In the data folder:
 - bradley.csv data file
- bradley_table1.Rmd R Markdown script
- bradley_table1.md Results of running R Markdown
- bradley_table1_result.csv is the table generated by that R Markdown script

To The “Live Code”

Opening bradley_table1_result.csv in Excel

	A	B	C	D	E
1		Case	Control	p	test
2	n	687	687		
3	age (mean (SD))	73.78 (10.24)	72.60 (10.50)	0.035	
4	sex = Male (%)	677 (98.5)	666 (96.9)	0.069	
5	race_eth = white (%)	546 (79.5)	527 (76.7)	0.24	
6	marital = yes (%)	356 (51.8)	310 (45.1)	0.015	
7	loc (%)				<0.001
8	ICU	186 (27.1)	65 (9.5)		
9	Bed	446 (64.9)	580 (84.4)		
10	Other	55 (8.0)	42 (6.1)		
11					

Learning More About Table 1

Chapter 1 of the Course Notes covers two larger examples, and more details, like...

- specifying factors, and re-ordering them when necessary
- using non-normal summaries or exact categorical tests
- dealing with warning messages and with missing data
- producing Table 1 in R so you can cut and paste it into Excel or Word

FYI: Lab 01 requires you to build a Table 1 from data.

Wrapping Up

Today we discussed

- ① Why I write dates the way I do
- ② Mechanics of the course
- ③ Data organization in spreadsheets
- ④ Naming Things and Getting Organized
- ⑤ Building a Table 1 (review Course Notes, Chapter 1)

Next Steps?

- Complete the Welcome to 432 survey by tomorrow noon.
- Look over the syllabus and the website, get R up and running.
- Look at the suggestions in the Class 01 README.
- Get started reading Leek's short book.
- You **can** do this.

432 Class 02 Slides

thomaselove.github.io/432

2022-01-13

Today's Agenda

- Linear Regression with Categorical Predictors
- Building Two-Way ANOVA models with interaction
- Building Analysis of Covariance Models

We'll use some BRFSS/SMART data about Ohio residents to address the following questions.

- ① What is the association of diabetes diagnosis and smoking status on BMI?
- ② Does adjusting for subject's age affect our assessments?
- ③ How does the subject's level of physical activity fit into this?

Chapter 2 of the Course Notes builds the BRFSS/SMART data.

Setup

```
knitr::opts_chunk$set(comment = NA)
options(width = 60)      ## for the slides

library(here)            ## project management
library(knitr)           ## mostly for kable
library(mosaic)          ## mostly for favstats
library(patchwork)        ## combine plots
library(janitor)          ## mostly for tabyl
library(naniar)           ## missing data tools
library(simputation)       ## for single imputation
library(broom)             ## for tidying model output
library(tidyverse)         ## as always (dplyr, ggplot2, etc.)

theme_set(theme_bw())     ## my personal preference
```

- I used message = FALSE in the code chunk setup.

Codebook of variables we'll select from smart_ohio

```
smart_ohio <- read_csv(here("data/smart_ohio.csv"))
```

```
dim(smart_ohio)
```

```
[1] 7412    99
```

We'll sample 432 observations from smart_ohio on these six variables...

Variable	Type	Description
SEQNO	ID code	we'll represent as a character
bmi	Quantity	body-mass index (in kg/m ²)
smoker	4 levels	smoking status (we'll collapse to 3 levels)
dm_status	4 levels	diabetes status (we'll collapse to 2)
activity	4 levels	physical activity level (we'll re-level)
age_imp	Quantity	age (imputed from groups, see Chapter 2)

Create our working data set (day2)

```
set.seed(43202)                                ## note 1

day2 <- smart_ohio %>%
  select(SEQNO, bmi, smoker, dm_status,
         activity, age_imp) %>%
  slice_sample(n = 432) %>%
  type.convert(as.is = FALSE) %>%
  mutate(SEQNO = as.character(SEQNO))           ## note 6
```

- ① set seed for random sampling
- ② modify smart_ohio data and place in new day2 tibble
- ③ select our six variables
- ④ take a random sample of 432 rows from the data
- ⑤ convert all character variables into factors
- ⑥ set ID code as a character variable

The day2 tibble, printed

day2

```
# A tibble: 432 x 6
  SEQNO      bmi smoker dm_status    activity age_imp
  <chr>     <dbl> <fct>   <fct>      <fct>      <int>
1 2017000934   NA  Never  No-Diabetes  Inactive      71
2 2017000411  27.3 Never  No-Diabetes  <NA>          69
3 2017001084  19.2 Former No-Diabetes  Inactive      NA
4 2017000065  20.5 Never  No-Diabetes Highly_Acti~      22
5 2017000240  25.1 Never  No-Diabetes Insufficien~      61
6 2017000300  24.8 Never  Pregnancy-I~ Inactive      40
7 2017001444  31.2 Former No-Diabetes Highly_Acti~      57
8 2017000258  29.4 Never  No-Diabetes  Active        27
9 2017000769  26.7 Former No-Diabetes Highly_Acti~      71
10 2017000247 27.4 Former No-Diabetes Inactive      91
# ... with 422 more rows
```

Initial Data Checking (Quantities)

```
favstats(~ bmi, data = day2) %>% kable(dig = 1)
```

min	Q1	median	Q3	max	mean	sd	n	missing
16.9	23.9	27.4	31.2	56.6	28.4	6.3	400	32

```
favstats(~ age_imp, data = day2) %>% kable(dig = 1)
```

min	Q1	median	Q3	max	mean	sd	n	missing
18	43	58.5	70	96	56.9	19	424	8

- ① Do the observed ranges of values look plausible in context?
- ② Are there missing values we need to deal with?

Checking the Categorical Data (activity)

```
day2 %>% count(activity)
```

```
# A tibble: 5 x 2
  activity             n
  <fct>              <int>
1 Active                60
2 Highly_Active         114
3 Inactive               124
4 Insufficiently_Active    87
5 <NA>                  47
```

- What does <NA> mean? What should we do with this variable?

Checking the Categorical Data (activity)

```
day2 %>% count(activity)
```

```
# A tibble: 5 x 2
  activity             n
  <fct>              <int>
1 Active                60
2 Highly_Active         114
3 Inactive               124
4 Insufficiently_Active    87
5 <NA>                  47
```

- What does <NA> mean? What should we do with this variable?
- Shortly, we'll **reorder** these levels in a more sensible way (suggestions?) and then we'll have to deal with the missing values, somehow.

Checking the Categorical Data (smoker)

```
day2 %>% count(smoker)
```

```
# A tibble: 5 x 2
  smoker              n
  <fct>            <int>
1 Current_daily      58
2 Current_not_daily  19
3 Former             124
4 Never              216
5 <NA>               15
```

- OK. Some missing values to deal with. What else might we do here?

Checking the Categorical Data (smoker)

```
day2 %>% count(smoker)
```

```
# A tibble: 5 x 2
  smoker              n
  <fct>            <int>
1 Current_daily      58
2 Current_not_daily  19
3 Former             124
4 Never              216
5 <NA>               15
```

- OK. Some missing values to deal with. What else might we do here?
- Shortly, we'll **collapse** this from 4 to 3 levels (how?)

Checking the Categorical Data (smoker)

```
day2 %>% count(smoker)
```

```
# A tibble: 5 x 2
  smoker              n
  <fct>            <int>
1 Current_daily      58
2 Current_not_daily  19
3 Former             124
4 Never              216
5 <NA>               15
```

- OK. Some missing values to deal with. What else might we do here?
- Shortly, we'll **collapse** this from 4 to 3 levels (how?)
- I think we'll go with Current, Former and Never

Checking the Categorical Data (dm_status)

```
day2 %>% count(dm_status)
```

```
# A tibble: 5 x 2
  dm_status      n
  <fct>        <int>
1 Diabetes       67
2 No-Diabetes   351
3 Pre-Diabetes    9
4 Pregnancy-Induced  4
5 <NA>           1
```

- Next Steps?

Checking the Categorical Data (dm_status)

```
day2 %>% count(dm_status)
```

```
# A tibble: 5 x 2
  dm_status      n
  <fct>        <int>
1 Diabetes       67
2 No-Diabetes   351
3 Pre-Diabetes    9
4 Pregnancy-Induced  4
5 <NA>           1
```

- Next Steps?
- Shortly, we'll collapse this to two levels (how might we do that?) and then we'll deal with the missing information.

Re-ordering and collapsing in day2

```
day2 <- day2 %>%
  mutate(activity =
    fct_relevel(activity, "Highly_Active",
                "Active", "Insufficiently_Active",
                "Inactive")) %>%
  mutate(smoker =
    fct_collapse(smoker,
                Current = c("Current_not_daily",
                            "Current_daily"))) %>%
  mutate(dm_status =
    fct_collapse(dm_status,
                No = c("No-Diabetes",
                      "Pre-Diabetes",
                      "Pregnancy-Induced"),
                Yes = "Diabetes")))
```

Sanity Checks

```
day2 %>% tabyl(activity) %>% adorn_pct_formatting()
```

	activity	n	percent	valid_percent
	Highly_Active	114	26.4%	29.6%
	Active	60	13.9%	15.6%
	Insufficiently_Active	87	20.1%	22.6%
	Inactive	124	28.7%	32.2%
	<NA>	47	10.9%	-

- Still need to deal with the missing values, but now the order makes sense.

Sanity Checks

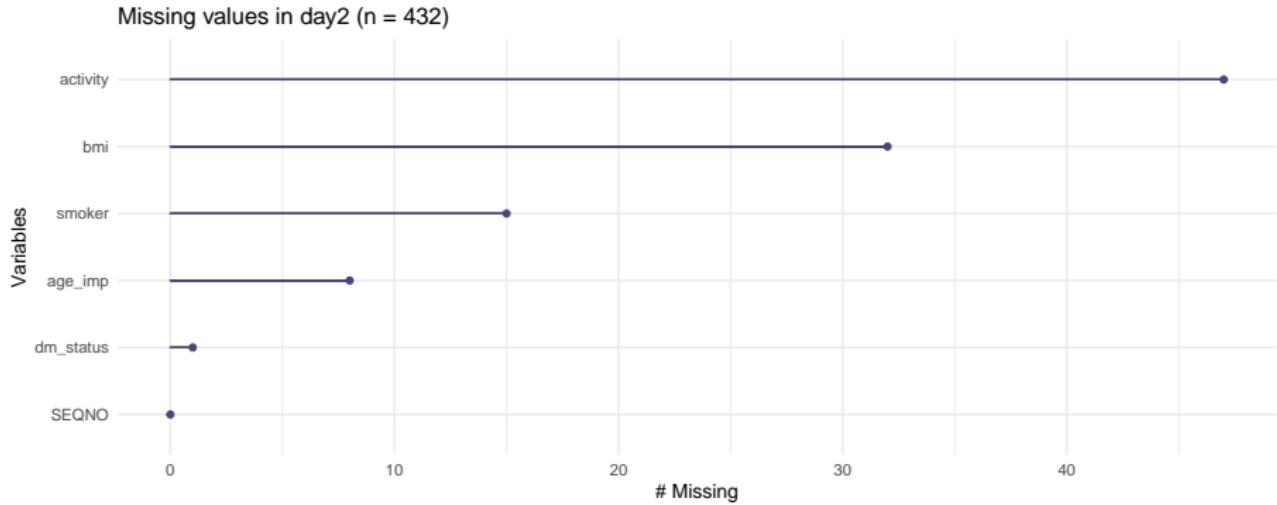
```
day2 %>% tabyl(dm_status, smoker)
```

dm_status	Current	Former	Never	NA_
Yes	10	23	33	1
No	67	100	183	14
<NA>	0	1	0	0

- OK, now we have two `dm_status` levels and three `smoker` levels, although we don't have a lot of currently smoking people with diabetes.
- Once we deal with the missing values, we should be all set.

How many missing values in each variable?

```
gg_miss_var(day2) +  
  labs(title = "Missing values in day2 (n = 432)")
```



- I used `warning = FALSE` in the code chunk setup.

Table of missing values in each variable?

We can also just get a count of missing values for each variable with `miss_var_summary()`, which is also part of `naniar`.

```
miss_var_summary(day2)
```

```
# A tibble: 6 x 3
  variable n_miss pct_miss
  <chr>     <int>    <dbl>
1 activity      47     10.9
2 bmi          32      7.41
3 smoker        15      3.47
4 age_imp       8      1.85
5 dm_status     1      0.231
6 SEQNO         0      0
```

How many missing values per row (subject)?

```
miss_case_table(day2) ## also from naniar package
```

```
# A tibble: 5 x 3
  n_miss_in_case n_cases pct_cases
      <int>     <int>     <dbl>
1             0     362     83.8
2             1      48     11.1
3             2      14      3.24
4             3       5      1.16
5             4       3      0.694
```

- How many observations would we lose in a complete case analysis?
- Can we make the necessary assumption for a complete case analysis?

What do we lose in a complete case analysis?

```
day2_cc <- day2 %>%  
  filter(complete.cases(.))
```

```
dim(day2_cc)
```

```
[1] 362    6
```

This seems clean in some ways (and is the default approach in software), but actually it hides a very important assumption, that the data are **missing completely at random**.

```
prop_miss_case(day2); prop_miss_case(day2_cc)
```

```
[1] 0.162037
```

```
[1] 0
```

Missing Data Mechanisms (Notes, Chapter 3)

- Missing Completely at Random (MCAR)
 - The probability of missing data is the same for every subject, so that throwing out cases with missing data does not bias inferences.
- Missing at Random (MAR)
 - Here, the probability that a variable is missing depends only on available information in your data (the other variables we have). If this is so, then **imputation** is the most appropriate option.
- Missing Not at Random (MNAR)
 - Whether data are missing is dependent on either unobserved predictors, or on the actual true (but unavailable) value of the observation itself or both. Even imputation cannot solve the problem.

What should we assume in our day2 scenario?

Formulating a single imputation plan

```
miss_var_summary(day2) %>% kable()
```

variable	n_miss	pct_miss
activity	47	10.8796296
bmi	32	7.4074074
smoker	15	3.4722222
age_imp	8	1.8518519
dm_status	1	0.2314815
SEQNO	0	0.0000000

Today, we'll use a *naive* approach to generating a single imputation.

- Impute dm_status with a random draw from its distribution.
- Use CART to impute smoker and activity from dm_status.
- Impute quantities via robust linear models using the factors.

Single imputation in day2 to yield day2_im

```
set.seed(432021)
day2_im <- day2 %>%
  data.frame() %>%
  impute_rhd(., dm_status ~ 1) %>%
  impute_cart(., smoker + activity ~ dm_status) %>%
  impute_rlm(., age_imp + bmi ~
              dm_status + smoker + activity) %>%
  tibble()
```

- impute_rhd (random hot deck) for dm_status
- impute_cart (classification and regression trees) for other factors
- impute_rlm (robust linear model) for age_imp and bmi

```
prop_miss_case(day2_im)
```

[1] 0

Draw the outcome?

- We're interested in diabetes and smoking's association with BMI
 - What do the BMI data look like? (plots shown on next slide)

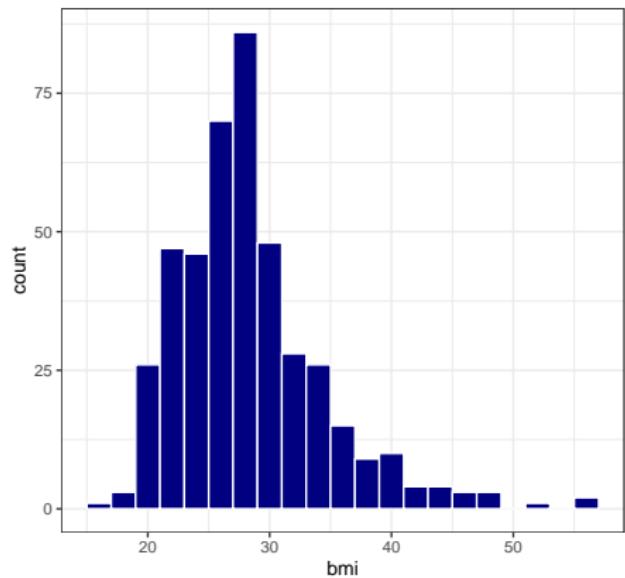
```
p1 <- ggplot(day2_im, aes(x = bmi)) +  
  geom_histogram(fill = "navy", col = "white",  
                 binwidth = 2) +  
  labs(title = "Histogram of BMI")
```

```
p2 <- ggplot(day2_im, aes(sample = bmi)) +  
  geom_qq(col = "navy") + geom_qq_line(col = "red") +  
  labs(title = "Normal Q-Q plot of BMI")
```

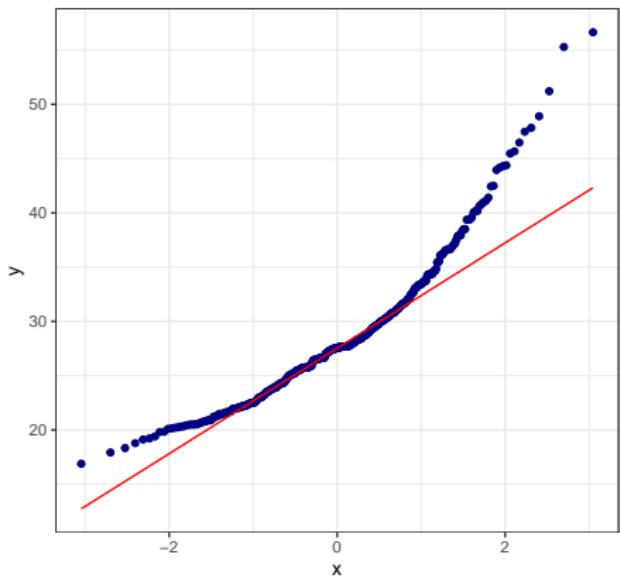
```
p1 + p2
```

BMI data in day2_im

Histogram of BMI



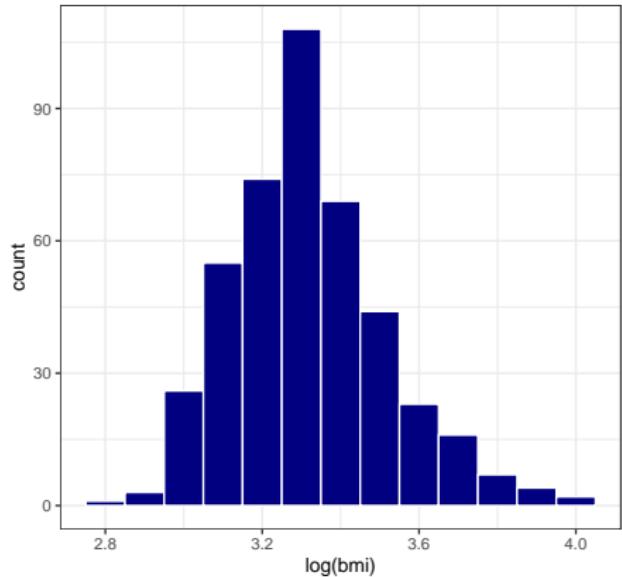
Normal Q–Q plot of BMI



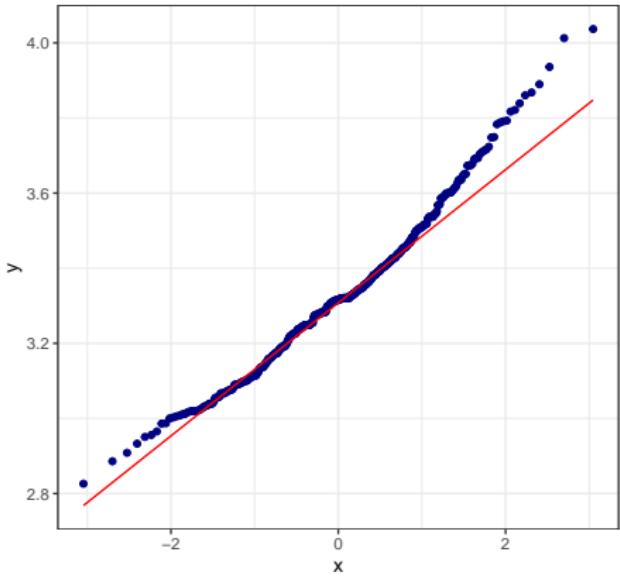
- These data are a little right-skewed. Transform?

Consider a logarithmic transformation of BMI?

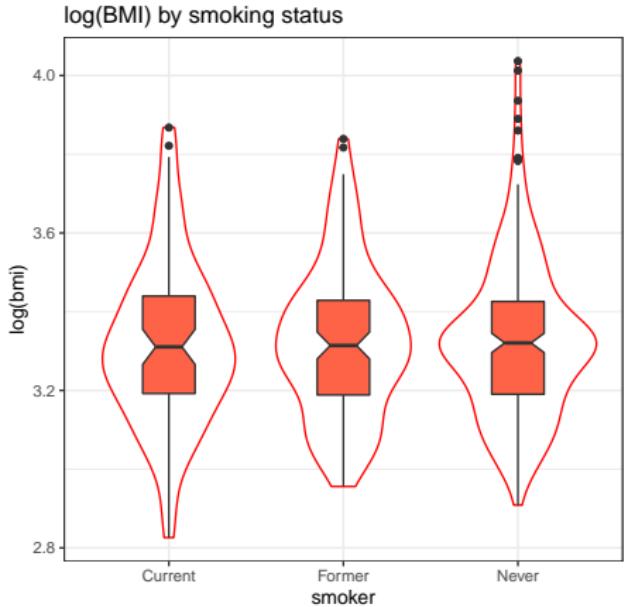
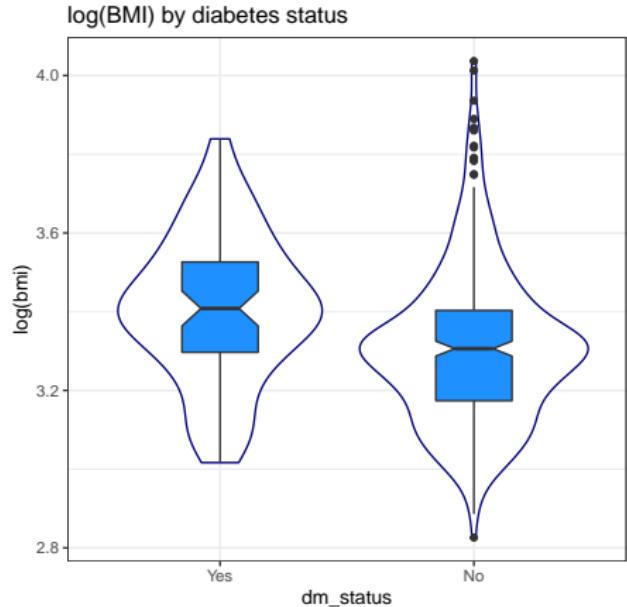
Histogram of log(BMI)



Normal Q–Q plot of log(BMI)



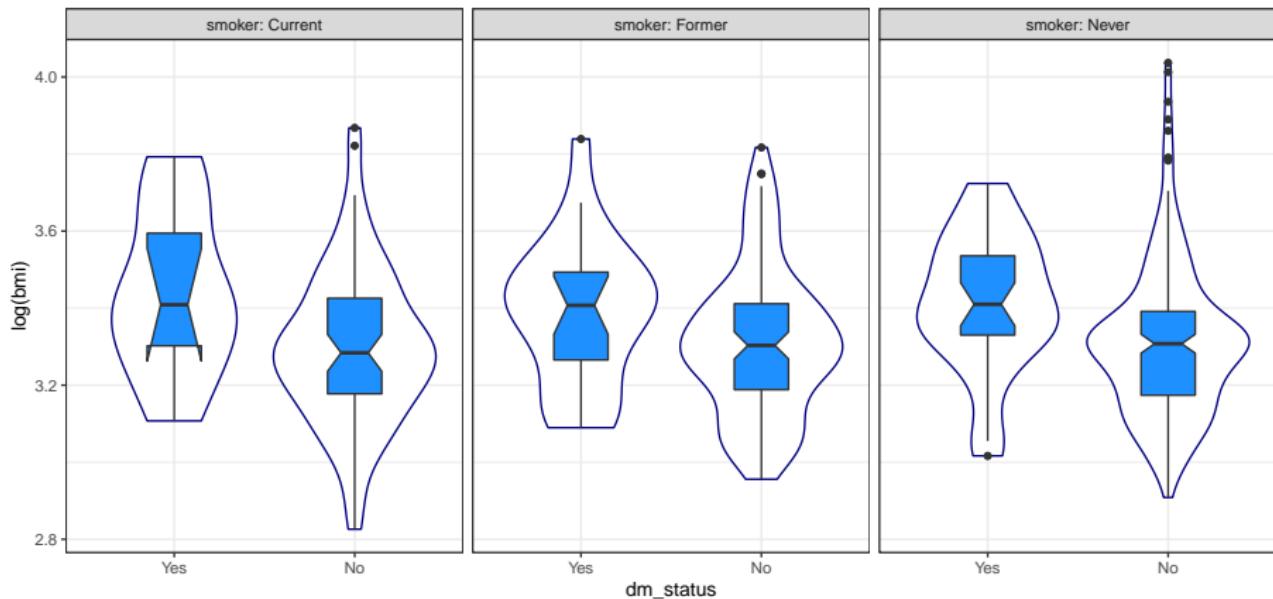
Compare $\log(\text{BMI})$ by diabetes and by smoking



$\log(\text{BMI})$ by diabetes and smoking together

notch went outside hinges. Try setting notch=FALSE.

$\log(\text{BMI})$ by diabetes and smoking status



Finding the Means of Each Group

We'll plot the mean of $\log(\text{bmi})$ in six combinations:

- two levels of `dm_status` combined with
- three levels of `smoker`

```
summaries_1 <- day2_im %>%
  group_by(dm_status, smoker) %>%
  summarise(n = n(), mean = mean(log(bmi)),
            stdev = sd(log(bmi)))
```

``summarise()`` has grouped output by '`dm_status`'. You can overr

We can suppress this message with `message = FALSE` in the code chunk label.

Here are the means of log(BMI) in each group

```
summaries_1 %>% kable(digits = 2)
```

dm_status	smoker	n	mean	stdev
Yes	Current	10	3.44	0.22
Yes	Former	23	3.39	0.19
Yes	Never	34	3.41	0.18
No	Current	67	3.30	0.20
No	Former	101	3.31	0.19
No	Never	197	3.31	0.20

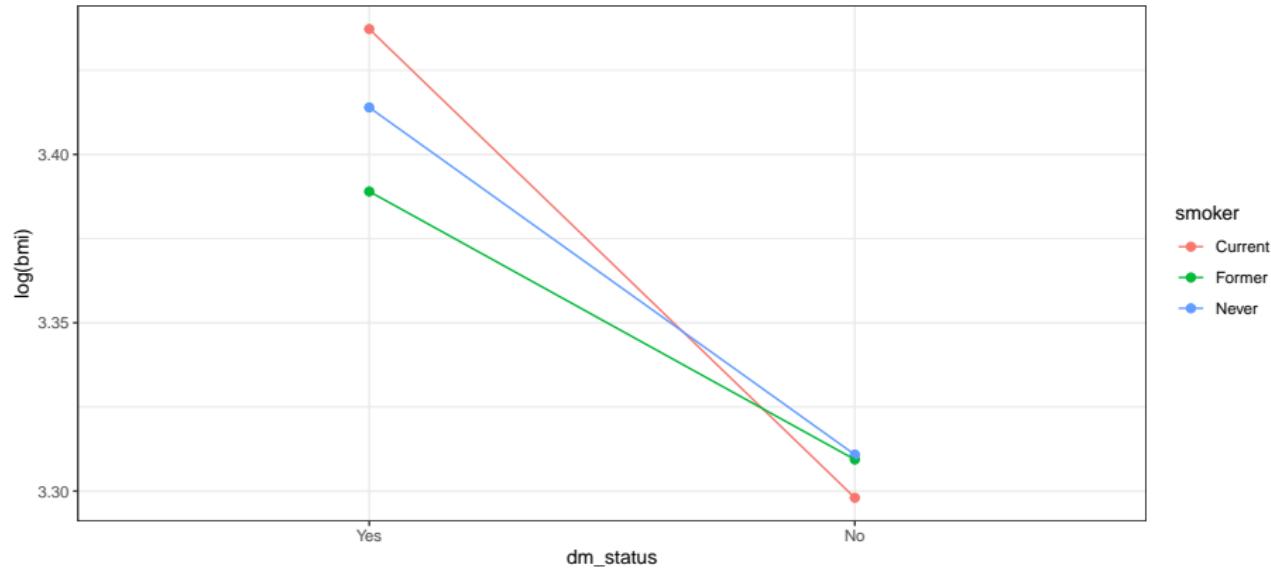
- Can we plot this information?

Plotting the Means (code)

```
ggplot(summaries_1, aes(x = dm_status, y = mean,
                        col = smoker)) +
  geom_point(size = 2) +
  geom_line(aes(group = smoker)) +
  labs(y = "log(bmi)",
       title = "Observed Means of log(BMI)",
       subtitle = "by Diabetes and Smoking Status")
```

Plotting the Means (results)

Observed Means of log(BMI)
by Diabetes and Smoking Status

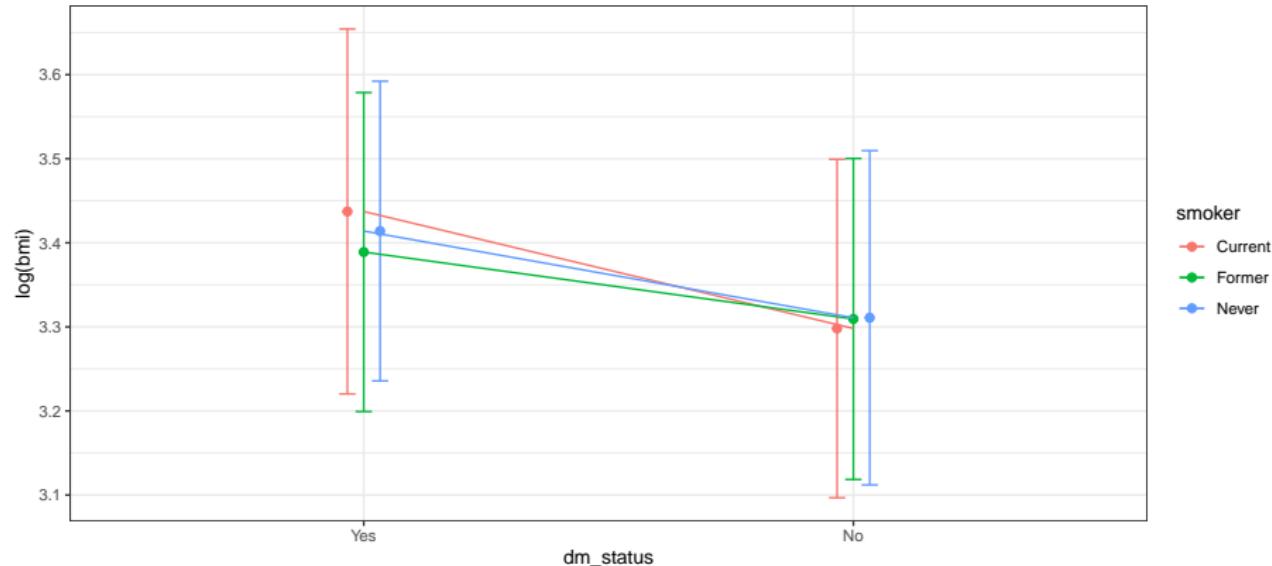


Adding in standard deviations (code)

```
pd <- position_dodge(0.1)
ggplot(summaries_1, aes(x = dm_status, y = mean,
                        col = smoker)) +
  geom_errorbar(aes(ymin = mean - stdev,
                     ymax = mean + stdev),
                width = 0.1, position = pd) +
  geom_point(size = 2, position = pd) +
  geom_line(aes(group = smoker)) +
  labs(y = "log(bmi)",
       title = "Means (+/- SD) of log(BMI)",
       subtitle = "by Diabetes and Smoking Status")
```

Adding in standard deviations (code)

Means (+/- SD) of log(BMI)
by Diabetes and Smoking Status



Review: One-Factor Analysis of Variance

```
m1 <- lm(log(bmi) ~ dm_status, data = day2_im)  
  
anova(m1)
```

Analysis of Variance Table

Response: log(bmi)

	Df	Sum Sq	Mean Sq	F value	Pr(>F)
dm_status	1	0.5748	0.57482	15.113	0.0001172 ***
Residuals	430	16.3546	0.03803		

Signif. codes:

0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Tidied m1 output

```
tidy(m1, conf.int = TRUE, conf.level = 0.90) %>%
  select(term, estimate,
         low90 = conf.low, high90 = conf.high,
         se = std.error, t = statistic, p = p.value) %>%
  kable(digits = c(0,2,2,2,2,2,5))
```

term	estimate	low90	high90	se	t	p
(Intercept)	3.41	3.37	3.45	0.02	143.07	0.000000
dm_statusNo	-0.10	-0.14	-0.06	0.03	-3.89	0.00012

Revising the order?

```
day2_im <- day2_im %>%
  mutate(dm_status = fct_relevel(dm_status, "No", "Yes"))

m1 <- lm(log(bmi) ~ dm_status, data = day2_im)

tidy(m1, conf.int = TRUE, conf.level = 0.90) %>%
  select(term, estimate,
         low90 = conf.low, high90 = conf.high,
         se = std.error, t = statistic, p = p.value) %>%
  kable(digits = c(0,2,2,2,2,2,5))
```

term	estimate	low90	high90	se	t	p
(Intercept)	3.31	3.29	3.32	0.01	324.07	0.00000
dm_statusYes	0.10	0.06	0.14	0.03	3.89	0.00012

Glancing at m1

```
glance(m1) %>%
  select(r.squared, adj.r.squared, sigma, AIC, BIC) %>%
  kable(digits = c(3, 3, 2, 1, 1))
```

r.squared	adj.r.squared	sigma	AIC	BIC
0.034	0.032	0.2	-182.4	-170.2

Developing a Two-Factor Model

We want to describe the mean of $\log(\text{BMI})$ as a function of **both**

- the two-level factor `dm_status`, and
- the three-level factor `smoker`

One decision is whether we'll consider an **interaction** term between these two factors.

- A model with the interaction will fit the data a bit better, by some measures.
- A model with the interaction is most appropriate if we believe the `dm_status` relationship with $\log(\text{BMI})$ changes depending on the level of `smoker`.
 - or at least if we are unwilling to assume the `smoker` effect is the same regardless of `dm_status`

What is an interaction term (a product term)?

When we build our two-way model with interaction, we'll include a product term

```
m2 <- lm(log(bmi) ~ dm_status*smoker, data = day2_im)
```

as compared to a model without interaction, which we'd fit with:

```
m2_no <- lm(log(bmi) ~ dm_status + smoker, data = day2_im)
```

Our main tool in thinking about these will be a means plot.

Two-Way ANOVA with Interaction

```
m2 <- lm(log(bmi) ~ dm_status*smoker, data = day2_im)  
  
anova(m2)
```

Analysis of Variance Table

Response: log(bmi)

	Df	Sum Sq	Mean Sq	F value	Pr(>F)	
dm_status	1	0.5748	0.57482	14.9970	0.0001246	***
smoker	2	0.0051	0.00253	0.0659	0.9362544	
dm_status:smoker	2	0.0214	0.01070	0.2791	0.7566000	
Residuals	426	16.3282	0.03833			

Signif. codes:

0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Tidied m2 coefficients

```
tidy(m2, conf.int = TRUE, conf.level = 0.90) %>%
  select(term, estimate,
         low90 = conf.low, high90 = conf.high,
         se = std.error, p = p.value) %>%
  kable(digits = c(0,3,2,2,2,3))
```

term	estimate	low90	high90	se	p
(Intercept)	3.298	3.26	3.34	0.02	0.000
dm_statusYes	0.139	0.03	0.25	0.07	0.037
smokerFormer	0.011	-0.04	0.06	0.03	0.713
smokerNever	0.013	-0.03	0.06	0.03	0.644
dm_statusYes:smokerFormer	-0.060	-0.19	0.07	0.08	0.459
dm_statusYes:smokerNever	-0.036	-0.16	0.09	0.08	0.634

Interpreting m2 (the interaction model)

m2 estimates derived from the indicator (1/0) variables

$$\begin{aligned}\log(\text{BMI}) = & 3.298 + 0.139 (\text{dm_status} = \text{Yes}) \\ & + 0.011 (\text{smoker} = \text{Former}) + 0.013 (\text{smoker} = \text{Never}) \\ & - 0.060 (\text{dm} = \text{Yes})(\text{smoker} = \text{Former}) \\ & - 0.036 (\text{dm} = \text{Yes})(\text{smoker} = \text{Never})\end{aligned}$$

- Estimated mean for a current smoker with no diabetes diagnosis?

Interpreting m2 (the interaction model)

m2 estimates derived from the indicator (1/0) variables

$$\begin{aligned}\log(\text{BMI}) = & 3.298 + 0.139 (\text{dm_status} = \text{Yes}) \\ & + 0.011 (\text{smoker} = \text{Former}) + 0.013 (\text{smoker} = \text{Never}) \\ & - 0.060 (\text{dm} = \text{Yes})(\text{smoker} = \text{Former}) \\ & - 0.036 (\text{dm} = \text{Yes})(\text{smoker} = \text{Never})\end{aligned}$$

- Estimated mean for a current smoker with no diabetes diagnosis?
- $\log(\text{BMI}) = 3.298$, so estimated BMI = $\exp(3.298) = 27.06$

Interpreting m2 (the interaction model)

m2 estimates derived from the indicator (1/0) variables

$$\begin{aligned}\log(\text{BMI}) = & 3.298 + 0.139 (\text{dm_status} = \text{Yes}) \\ & + 0.011 (\text{smoker} = \text{Former}) + 0.013 (\text{smoker} = \text{Never}) \\ & - 0.060 (\text{dm} = \text{Yes})(\text{smoker} = \text{Former}) \\ & - 0.036 (\text{dm} = \text{Yes})(\text{smoker} = \text{Never})\end{aligned}$$

- Estimated mean for a current smoker with no diabetes diagnosis?
- $\log(\text{BMI}) = 3.298$, so estimated BMI = $\exp(3.298) = 27.06$
- Estimated mean for a never smoker with no diabetes diagnosis?

Interpreting m2 (the interaction model)

m2 estimates derived from the indicator (1/0) variables

$$\begin{aligned}\log(\text{BMI}) = & 3.298 + 0.139 (\text{dm_status} = \text{Yes}) \\ & + 0.011 (\text{smoker} = \text{Former}) + 0.013 (\text{smoker} = \text{Never}) \\ & - 0.060 (\text{dm} = \text{Yes})(\text{smoker} = \text{Former}) \\ & - 0.036 (\text{dm} = \text{Yes})(\text{smoker} = \text{Never})\end{aligned}$$

- Estimated mean for a current smoker with no diabetes diagnosis?
- $\log(\text{BMI}) = 3.298$, so estimated BMI = $\exp(3.298) = 27.06$
- Estimated mean for a never smoker with no diabetes diagnosis?
- $\log(\text{BMI}) = 3.298 + 0.013 = 3.311$, so BMI = $\exp(3.311) = 27.41$

Interpreting m2 (the interaction model)

m2 estimates derived from the indicator (1/0) variables

$$\begin{aligned}\log(\text{BMI}) = & 3.298 + 0.139 (\text{dm_status} = \text{Yes}) \\ & + 0.011 (\text{smoker} = \text{Former}) + 0.013 (\text{smoker} = \text{Never}) \\ & - 0.060 (\text{dm} = \text{Yes})(\text{smoker} = \text{Former}) \\ & - 0.036 (\text{dm} = \text{Yes})(\text{smoker} = \text{Never})\end{aligned}$$

- Estimated mean for a current smoker with no diabetes diagnosis?
- $\log(\text{BMI}) = 3.298$, so estimated BMI = $\exp(3.298) = 27.06$
- Estimated mean for a never smoker with no diabetes diagnosis?
- $\log(\text{BMI}) = 3.298 + 0.013 = 3.311$, so BMI = $\exp(3.311) = 27.41$
- Estimated mean for a never smoker with a diabetes diagnosis?

Interpreting m2 (the interaction model)

m2 estimates derived from the indicator (1/0) variables

$$\begin{aligned}\log(\text{BMI}) = & 3.298 + 0.139 (\text{dm_status} = \text{Yes}) \\ & + 0.011 (\text{smoker} = \text{Former}) + 0.013 (\text{smoker} = \text{Never}) \\ & - 0.060 (\text{dm} = \text{Yes})(\text{smoker} = \text{Former}) \\ & - 0.036 (\text{dm} = \text{Yes})(\text{smoker} = \text{Never})\end{aligned}$$

- Estimated mean for a current smoker with no diabetes diagnosis?
- $\log(\text{BMI}) = 3.298$, so estimated BMI = $\exp(3.298) = 27.06$
- Estimated mean for a never smoker with no diabetes diagnosis?
- $\log(\text{BMI}) = 3.298 + 0.013 = 3.311$, so BMI = $\exp(3.311) = 27.41$
- Estimated mean for a never smoker with a diabetes diagnosis?
- $\log(\text{BMI}) = 3.298 + 0.139 + 0.013 - 0.036 = 3.414$; BMI = 30.39

What if we assume there's no interaction?

```
m2_no <- lm(log(bmi) ~ dm_status + smoker, data = day2_im)  
  
anova(m2_no)
```

Analysis of Variance Table

Response: log(bmi)

	Df	Sum Sq	Mean Sq	F value	Pr(>F)	
dm_status	1	0.5748	0.57482	15.0477	0.0001213	***
smoker	2	0.0051	0.00253	0.0661	0.9360460	
Residuals	428	16.3496	0.03820			

Signif. codes:

0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Tidied m2_no coefficients

```
tidy(m2_no, conf.int = TRUE, conf.level = 0.90) %>%
  select(term, estimate,
         low90 = conf.low, high90 = conf.high,
         se = std.error, p = p.value) %>%
kable(digits = c(0,3,2,2,2,3))
```

term	estimate	low90	high90	se	p
(Intercept)	3.303	3.27	3.34	0.02	0.000
dm_statusYes	0.101	0.06	0.14	0.03	0.000
smokerFormer	0.002	-0.04	0.05	0.03	0.932
smokerNever	0.008	-0.03	0.05	0.03	0.751

Interpreting m2_no (no interaction model)

m2 estimates derived from the indicator (1/0) variables

$$\begin{aligned}\log(\text{BMI}) = & 3.303 + 0.101 \ (\text{dm_status} = \text{Yes}) \\ & + 0.002 \ (\text{smoker} = \text{Former}) \\ & + 0.008 \ (\text{smoker} = \text{Never})\end{aligned}$$

- Estimated mean for a current smoker with no diabetes diagnosis?
 - $\log(\text{BMI}) = 3.303$, so estimated BMI = $\exp(3.303) = 27.19$
- Estimated mean for a never smoker with no diabetes diagnosis?
 - $\log(\text{BMI}) = 3.303 + 0.008 = 3.311$, so BMI = $\exp(3.311) = 27.41$
- Estimated mean for a never smoker with a diabetes diagnosis?
 - $\log(\text{BMI}) = 3.303 + 0.008 + 0.101 = 3.412$ so BMI = 30.33

What did we see in the data?

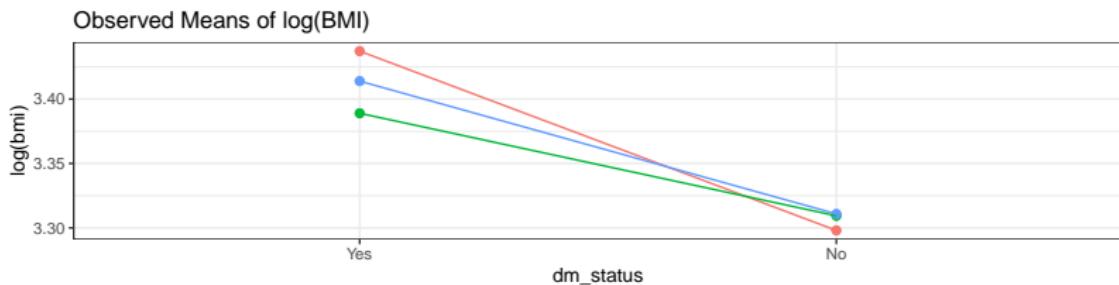
Estimates of $\text{mean}(\log(\text{BMI}))$ from the two models, vs. the actual data.

dm_status	smoker	n	actual	m2 est.	m2_no est.
No	Current	67	3.298	3.298	3.303
No	Never	197	3.311	3.311	3.311
Yes	Never	34	3.414	3.414	3.412
No	Former	101	3.309	3.309	3.305
Yes	Current	10	3.437	3.437	3.404
Yes	Former	23	3.389	3.389	3.406

- The two-way ANOVA model with interaction simply reproduces the observed means.
- Not clear we want to assume the interaction effect is actually zero.

Interaction Plot shows non-parallel lines?

Two versions of the Means Plot



How about % of variation explained measures?

```
tidy(anova(m2)) %>% select(term, df, sumsq) %>%
  kable(dig = c(0, 0, 4))
```

term	df	sumsq
dm_status	1	0.5748
smoker	2	0.0051
dm_status:smoker	2	0.0214
Residuals	426	16.3282

- R^2 associated with the interaction term?
 - SS(interaction) is 0.0214
 - $SS(m2) = 0.5748 + 0.0051 + 0.0214 = 0.6013$
 - Interaction accounts for 3.6% of the variation explained by m2
 - Interaction accounts for $0.0214 / (0.6013 + 16.3282) = 0.0013$, or about 0.13% of the variation in log(BMI)

Comparison of Fit across the models?

```
comp_table <- bind_rows( glance(m2), glance(m2_no) ) %>%
  mutate(mod = c("m2", "m2_no"))

comp_table %>%
  select(mod, r.squared, adj.r.squared, sigma, AIC, BIC) %>%
  kable(dig = c(0, 3, 3, 3, 1, 1))
```

mod	r.squared	adj.r.squared	sigma	AIC	BIC
m2	0.036	0.024	0.196	-175.1	-146.6
m2_no	0.034	0.027	0.195	-178.5	-158.2

- Is there much to choose from in comparing the in-sample performance?

How else can we assess the fit of these models?

We're not keen on making model decisions based on significance tests.
Model selection doesn't actually work well, in practice.

```
anova(m2, m2_no)
```

Analysis of Variance Table

Model 1: $\log(\text{bmi}) \sim \text{dm_status} * \text{smoker}$

Model 2: $\log(\text{bmi}) \sim \text{dm_status} + \text{smoker}$

	Res.Df	RSS	Df	Sum of Sq	F	Pr(>F)
1	426	16.328				
2	428	16.350	-2	-0.021396	0.2791	0.7566

- We'd rather think about how the two models reflect the data we have *and* predict on new data.

OK, what if we add age as a covariate?

ANCOVA model `m3` takes our model `m2` (with interaction) and adds in `age_imp` (centered by subtracting off its mean) as a predictor.

```
day2_im <- day2_im %>%
  mutate(age_c = age_imp - mean(age_imp))

day2_im %>% select(age_imp, age_c) %>% summary()
```

	age_imp	age_c
Min.	:18.00	Min. : -38.942
1st Qu.	:44.00	1st Qu.: -12.942
Median	:59.00	Median : 2.058
Mean	:56.94	Mean : 0.000
3rd Qu.	:70.00	3rd Qu.: 13.058
Max.	:96.00	Max. : 39.058

OK, what if we add age as a covariate?

Here's an analysis of **covariance** model m3 with a factor-factor interaction, plus a centered quantitative covariate.

```
m3 <- lm(log(bmi) ~ dm_status * smoker + age_c,  
         data = day2_im)
```

Does this change the nature of the relationship we see between dm_status, smoker and bmi?

Model m3 coefficients

```
tidy(m3, conf.int = TRUE, conf.level = 0.90) %>%
  select(term, estimate, low90 = conf.low,
         high90 = conf.high) %>% kable(dig = 3)
```

term	estimate	low90	high90
(Intercept)	3.288	3.249	3.328
dm_statusYes	0.148	0.039	0.257
smokerFormer	0.026	-0.025	0.077
smokerNever	0.019	-0.026	0.064
age_c	-0.001	-0.002	-0.001
dm_statusYes:smokerFormer	-0.056	-0.188	0.075
dm_statusYes:smokerNever	-0.025	-0.149	0.099

ANOVA results for m2 and m3

```
tidy(anova(m3)) %>% select(term, df, sumsq) %>% kable(dig = 3)
```

term	df	sumsq
dm_status	1	0.575
smoker	2	0.005
age_c	1	0.307
dm_status:smoker	2	0.021
Residuals	425	16.021

```
tidy(anova(m2)) %>% select(term, df, sumsq) %>% kable(dig = 3)
```

term	df	sumsq
dm_status	1	0.575
smoker	2	0.005
dm_status:smoker	2	0.021
Residuals	426	16.328

Does age_imp improve quality of fit?

```
comp_table <- bind_rows( glance(m2), glance(m3) ) %>%
  mutate(mod = c("m2", "m3"))

comp_table %>%
  select(mod, r.squared, adj.r.squared, sigma, AIC, BIC) %>%
  kable(dig = c(0, 3, 3, 3, 1, 1))
```

mod	r.squared	adj.r.squared	sigma	AIC	BIC
m2	0.036	0.024	0.196	-175.1	-146.6
m3	0.054	0.040	0.194	-181.3	-148.7

How about if we add a third factor (activity)?

```
m4 <- lm(log(bmi) ~ dm_status * smoker + age_c + activity,  
         data = day2_im)  
tidy(anova(m4)) %>% select(term, df, sumsq) %>%  
  kable(dig = 3)
```

term	df	sumsq
dm_status	1	0.575
smoker	2	0.005
age_c	1	0.307
activity	3	0.269
dm_status:smoker	2	0.036
Residuals	422	15.737

Does activity improve quality of fit?

```
comp_table <- bind_rows( glance(m1), glance(m2), glance(m3),
                        glance(m4)) %>%
  mutate(mod = c("m1", "m2", "m3", "m4"))

comp_table %>%
  select(mod, r.squared, adj.r.squared, sigma, AIC, BIC) %>%
  kable(dig = c(0, 3, 3, 3, 1, 1))
```

mod	r.squared	adj.r.squared	sigma	AIC	BIC
m1	0.034	0.032	0.195	-182.4	-170.2
m2	0.036	0.024	0.196	-175.1	-146.6
m3	0.054	0.040	0.194	-181.3	-148.7
m4	0.070	0.051	0.193	-183.0	-138.3

What's next?

- Is it feasible to look at the assumptions of these models?
- Could we consider additional interaction terms?
 - factor-factor interactions?
 - factor-covariate interactions?
- Interaction is just one type of non-linearity. Can we include other types?
- Should we think more about back-transformation in this setting?
- Could we split the sample and consider how well we'd predict in new data?
- Is `lm` the best way to fit regression models to a quantitative outcome like $\log(\text{bmi})$?

Can we build up our framework for developing regression models?

432 Class 03 Slides

thomaselove.github.io/432

2022-01-18

Today's Agenda

- Create a data set for week 2 analyses from `smart_ohio`
- Making cleaning / tidying decisions, then saving our work
- Simple imputation
- Splitting the sample with `rsample` tools
- Fitting a model (and then several more models) with `lm`
 - Incorporating an interaction between factors
- Regression Diagnostics via Residual Plots

Creating and Managing the Data for Week 2

Setup

```
knitr::opts_chunk$set(comment = NA)
options(width = 60)

library(here); library(knitr)
library(janitor); library(patchwork)
library(naniar); library(simputation)
library(skimr)          ## for a specific summary
library(equatiomatic)   ## print equations
library(broom)
library(rsample)         ## new today: data splitting
library(yardstick)       ## new today: evaluating fits
library(tidyverse)

theme_set(theme_bw())
options(dplyr.summarise.inform = FALSE) ## avoid message
```

Similar approach as last time...

```
smart_ohio <- read_csv(here("data/smart_ohio.csv"))

week2 <- smart_ohio %>%
  filter(hx_diabetes == 0,
         mmsa == "Cleveland-Elyria",
         complete.cases(bmi)) %>%
  select(bmi, inc_imp, fruit_day, drinks_wk,
         female, exerany, genhealth, race_eth,
         hx_diabetes, mmsa, SEQNO) %>%
  type.convert(as.is = FALSE) %>%
  mutate(ID = as.character(SEQNO - 2017000000)) %>%
  relocate(ID)
```

```
# A tibble: 894 x 12
  ID      bmi inc_imp fruit_day drinks_wk female exerany
  <chr>  <dbl>   <int>     <dbl>     <dbl>   <int>   <int>
1 2       23.0    86865      4         0        1        0
2 3       26.9     NA         3         0        1        1
3 4       26.5     NA         2         4.67     1        1
4 5       24.2    58311     0.57      0.93     0        1
5 7       23.0    2318       2         2        0        1
6 8       28.4    79667      1         0        0        1
7 9       30.1    47880     0.23      0        0        1
8 10      19.8    100136     0.77      0.47     1        1
9 11      27.2    73145      0.71      0        0        1
10 12     24.6    76917      1.07      0        1        1
# ... with 884 more rows, and 5 more variables:
#   genhealth <fct>, race_eth <fct>, hx_diabetes <int>,
#   mmsa <fct>, SEQNO <int>
```

Codebook for useful week2 variables

- 894 subjects in Cleveland-Elyria with bmi and no history of diabetes

Variable	Description
bmi	(outcome) Body-Mass index in kg/m ² .
inc_imp	income (imputed from grouped values) in \$
fruit_day	average fruit servings consumed per day
drinks_wk	average alcoholic drinks consumed per week
female	sex: 1 = female, 0 = male
exerany	any exercise in the past month: 1 = yes, 0 = no
genhealth	self-reported overall health (5 levels)
race_eth	race and Hispanic/Latinx ethnicity (5 levels)

- plus ID, SEQNO, hx_diabetes (all 0), MMSA (all Cleveland-Elyria)
- See Chapter 2 of the Course Notes for details on the variables

Basic Data Summaries

Available approaches include:

- `summary`
- `mosaic` package's `inspect()`
- `skimr` package's `skim_without_charts()`
- `Hmisc` package's `describe`

all of which can work nicely in an HTML presentation, but none of them fit well on one of these slides.

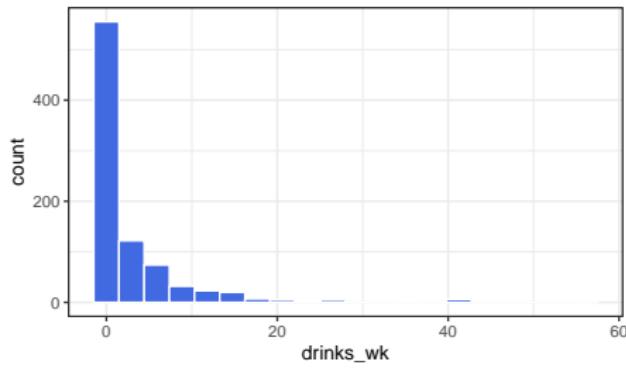
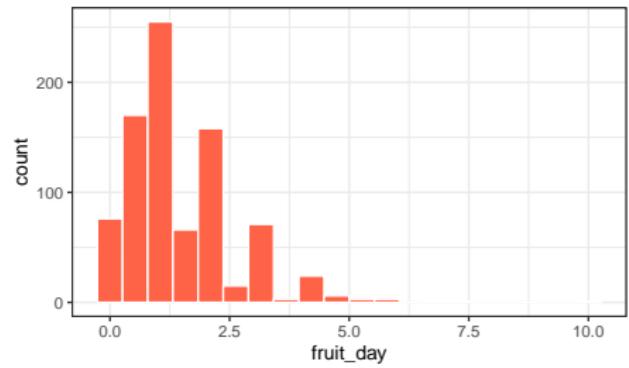
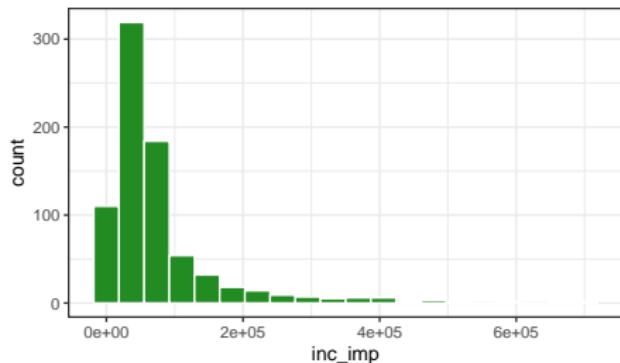
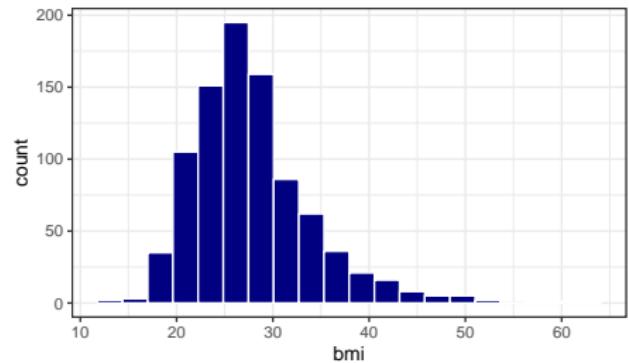
Summarizing the Quantities (Raw week2)

```
week2 %>% select(bmi, inc_imp, fruit_day, drinks_wk) %>%  
  skim_without_charts() %>%  
  yank(., "numeric") %>%  
  select(var = skim_variable, n_missing, min = p0,  
         median = p50, max = p100, mean, sd) %>%  
  kable(digits = 1)
```

var	n_missing	min	median	max	mean	sd
bmi	0	13.3	26.8	63	27.9	6.3
inc_imp	120	216.0	48224.5	700676	75673.5	90695.8
fruit_day	41	0.0	1.1	10	1.4	1.1
drinks_wk	39	0.0	0.5	56	3.0	6.1

- Any signs of trouble? (What are we looking for?)

Quick Histogram of each quantitative variable



Code for previous slide

```
p1 <- ggplot(week2, aes(x = bmi)) +  
  geom_histogram(fill = "navy", col = "white", bins = 20)  
p2 <- ggplot(week2, aes(x = inc_imp)) +  
  geom_histogram(fill = "forestgreen", col = "white",  
                 bins = 20)  
p3 <- ggplot(week2, aes(x = fruit_day)) +  
  geom_histogram(fill = "tomato", col = "white", bins = 20)  
p4 <- ggplot(week2, aes(x = drinks_wk)) +  
  geom_histogram(fill = "royalblue", col = "white",  
                 bins = 20)  
(p1 + p2) / (p3 + p4)
```

I also used `warning = FALSE` in the plot's code chunk label to avoid warnings about missing values, like this one for `inc_imp`:

Warning: Removed 120 rows containing non-finite values

Binary variables in raw week2

```
week2 %>% tabyl(female, exerany) %>% adorn_title()
```

		exerany		
		female	0	1
female	0	95	268	20
	1	128	361	22

- female is based on biological sex (1 = female, 0 = male)
- exerany comes from a response to “During the past month, other than your regular job, did you participate in any physical activities or exercises such as running, calisthenics, golf, gardening, or walking for exercise?” (1 = yes, 0 = no, don’t know and refused = missing)
- Any signs of trouble here?

Binary variables in raw week2

```
week2 %>% tabyl(female, exerany) %>% adorn_title()
```

		exerany		
		female	0	1
female	0	95	268	20
	1	128	361	22

- female is based on biological sex (1 = female, 0 = male)
- exerany comes from a response to “During the past month, other than your regular job, did you participate in any physical activities or exercises such as running, calisthenics, golf, gardening, or walking for exercise?” (1 = yes, 0 = no, don’t know and refused = missing)
- Any signs of trouble here?
- I think the 1/0 values and names are OK choices.

Multicategorical genhealth in raw week2

```
week2 %>% tabyl(genhealth)
```

	genhealth	n	percent	valid_percent
1_Excellent	148	0.165548098	0.16573348	
2_VeryGood	324	0.362416107	0.36282195	
3_Good	274	0.306487696	0.30683091	
4_Fair	112	0.125279642	0.12541993	
5_Poor	35	0.039149888	0.03919373	
<NA>	1	0.001118568		NA

- The variable is based on “Would you say that in general your health is ...” using the five specified categories (Excellent -> Poor), numbered for convenience after data collection.
- Don’t know / not sure / refused were each treated as missing.
- How might we manage this variable?

Changing the levels for genhealth

```
week2 <- week2 %>%
  mutate(health =
    fct_recode(genhealth,
      E = "1_Excellent",
      VG = "2_VeryGood",
      G = "3_Good",
      F = "4_Fair",
      P = "5_Poor"))
```

Might want to run a sanity check here, just to be sure...

Checking health vs. genhealth in week2

```
week2 %>% tabyl(genhealth, health) %>% adorn_title()
```

		health					
genhealth		E	VG	G	F	P	NA_
1_Excellent		148	0	0	0	0	0
2_VeryGood		0	324	0	0	0	0
3_Good		0	0	274	0	0	0
4_Fair		0	0	0	112	0	0
5_Poor		0	0	0	0	35	0
<NA>		0	0	0	0	0	1

- OK. We've preserved the order and we have much shorter labels. Sometimes, that's helpful.

Multicategorical race_eth in raw week2

```
week2 %>% count(race_eth)
```

```
# A tibble: 6 x 2
  race_eth                n
  <fct>                  <int>
1 Black non-Hispanic     167
2 Hispanic                 27
3 Multiracial non-Hispanic 19
4 Other race non-Hispanic 22
5 White non-Hispanic      646
6 <NA>                     13
```

“Don’t know”, “Not sure”, and “Refused” were treated as missing.

- What is this variable actually about?

Multicategorical race_eth in raw week2

```
week2 %>% count(race_eth)
```

```
# A tibble: 6 x 2
  race_eth                n
  <fct>                  <int>
1 Black non-Hispanic     167
2 Hispanic                 27
3 Multiracial non-Hispanic 19
4 Other race non-Hispanic 22
5 White non-Hispanic      646
6 <NA>                     13
```

“Don’t know”, “Not sure”, and “Refused” were treated as missing.

- What is this variable actually about?
- What is the most common thing people do here?

What is the question you are asking?

Collapsing race_eth levels *might* be rational for *some* questions.

- We have lots of data from two categories, but only two.
- Systemic racism affects people of color in different ways across these categories, but also *within* them.
- Is combining race and Hispanic/Latinx ethnicity helpful?

It's hard to see the justice in collecting this information and not using it in as granular a form as possible, though this leaves some small sample sizes. There is no magic number for "too small a sample size."

- Most people identified themselves in one of the categories.
- These data are not ordered, and (I'd argue) ordering them isn't helpful.
- Regression models are easier to interpret, though, if the "baseline" category is a common one.

Resorting the factor for race_eth

Let's sort all five levels, from most observations to least...

```
week2 <- week2 %>%  
  mutate(race_eth = fct_infreq(race_eth))
```

```
week2 %>% tabyl(race_eth)
```

	race_eth	n	percent	valid_percent
White	non-Hispanic	646	0.72259508	0.73325766
Black	non-Hispanic	167	0.18680089	0.18955732
	Hispanic	27	0.03020134	0.03064699
Other	race non-Hispanic	22	0.02460850	0.02497162
Multiracial	non-Hispanic	19	0.02125280	0.02156640
	<NA>	13	0.01454139	NA

- Not a perfect solution, certainly, but we'll try it out.

“Cleaned” Data and Missing Values

```
week2 <- week2 %>%
  select(ID, bmi, inc_imp, fruit_day, drinks_wk,
         female, exerany, health, race_eth, everything())
```

```
miss_var_summary(week2)
```

```
# A tibble: 13 x 3
```

	variable	n_miss	pct_miss
	<chr>	<int>	<dbl>
1	inc_imp	120	13.4
2	exerany	42	4.70
3	fruit_day	41	4.59
4	drinks_wk	39	4.36
5	race_eth	13	1.45
6	health	1	0.112
7	genhealth	1	0.112
8	ID	0	0

Single Imputation Approach?

```
set.seed(43203)
week2im <- week2 %>%
  select(ID, bmi, inc_imp, fruit_day, drinks_wk,
         female, exerany, health, race_eth) %>%
  data.frame() %>%
  impute_cart(health ~ bmi + female) %>%
  impute_pmm(exerany ~ female + health + bmi) %>%
  impute_rlm(inc_imp + drinks_wk + fruit_day ~
              bmi + female + health + exerany) %>%
  impute_cart(race_eth ~ health + inc_imp + bmi) %>%
  tibble()
```

```
prop_miss_case(week2im)
```

```
[1] 0
```

Saving the tidied data

Let's save both the unimputed and the imputed tidy data as R data sets.

```
saveRDS(week2, here("data", "week2.Rds"))
```

```
saveRDS(week2im, here("data", "week2im.Rds"))
```

To reload these files, we'd use `readRDS`.

- The main advantage here is that we've saved the whole R object, including all characteristics that we've added since the original download.

Splitting the Sample

Use `initial_split` from `rsample` to partition the data into:

- Model development (training) sample where we'll build models
- Model evaluation (testing) sample which we'll hold out for a while

```
set.seed(432)      ## to make the work replicable in the future
week2im_split <- initial_split(week2im, prop = 3/4)
```

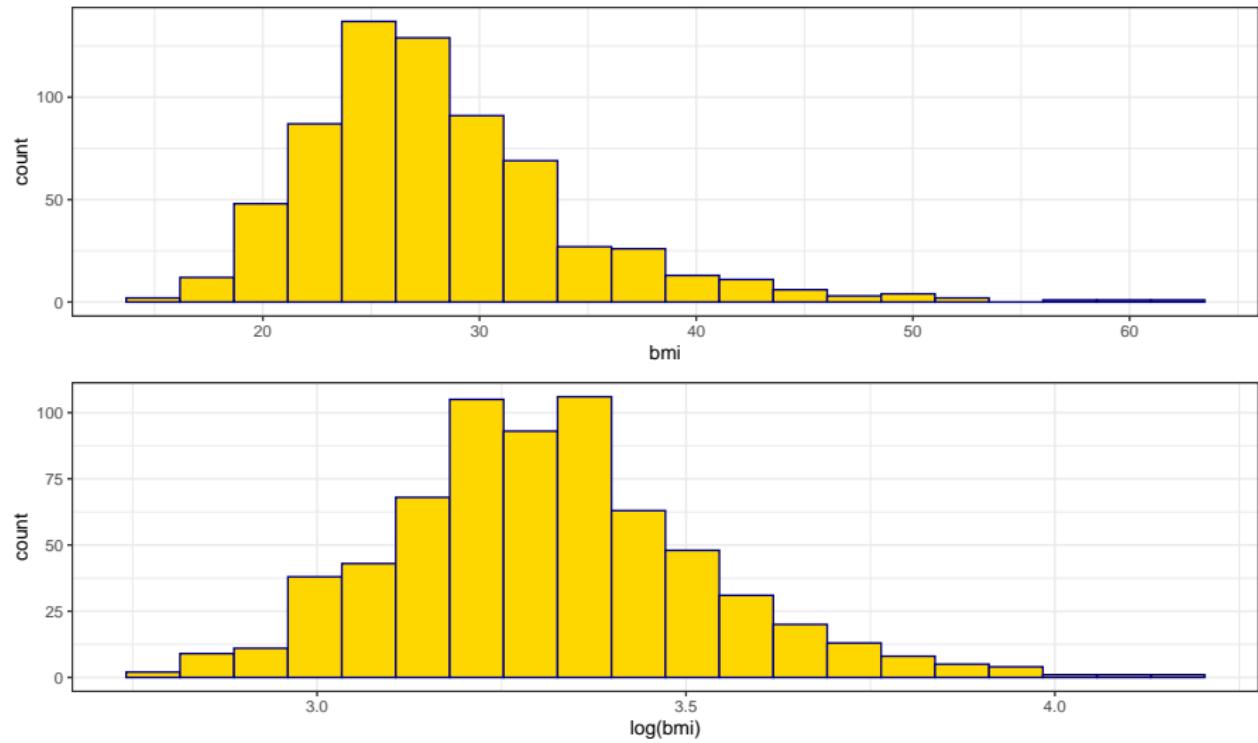
```
train_w2im <- training(week2im_split)
test_w2im <- testing(week2im_split)
```

```
dim(train_w2im); dim(test_w2im)
```

```
[1] 670    9
```

```
[1] 224    9
```

Should we transform our outcome?



Outcome: bmi, with key predictors exerany and health both categorical (two-way ANOVA!)

bmi means by exerany and health

```
summaries_1 <- train_w2im %>%
  group_by(exerany, health) %>%
  summarise(n = n(), mean = mean(bmi), stdev = sd(bmi))
summaries_1 %>% kable(digits = 2)
```

exerany	health	n	mean	stdev
0	E	18	27.49	3.56
0	VG	54	26.87	5.27
0	G	58	30.33	7.45
0	F	31	35.12	9.95
0	P	8	36.21	12.11
1	E	92	25.80	4.49
1	VG	191	26.80	4.89
1	G	152	29.12	6.26
1	F	49	27.21	5.55
1	P	17	28.50	8.61

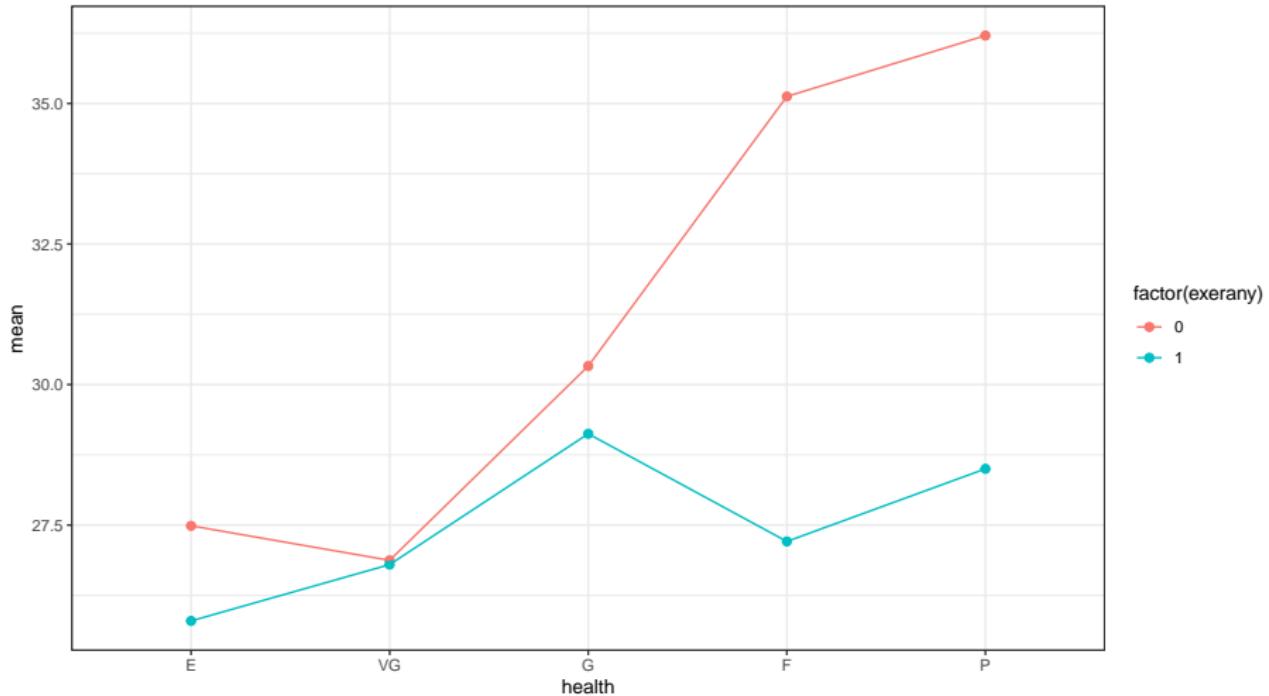
Code for Interaction Plot

```
ggplot(summaries_1, aes(x = health, y = mean,
                        col = factor(exerany))) +
  geom_point(size = 2) +
  geom_line(aes(group = factor(exerany))) +
  labs(title = "Observed Means of BMI",
       subtitle = "by Exercise and Overall Health")
```

- Note the use of `factor` here since the `exerany` variable is in fact numeric, although it only takes the values 1 and 0.
 - Sometimes it's helpful to treat 1/0 as a factor, and sometimes not.
- Where is the evidence of serious non-parallelism (if any) in the plot on the next slide that results from this code?

Resulting Interaction Plot

Observed Means of BMI
by Exercise and Overall Health



Models we'll build today

- `m_1` a linear model without interaction using `exerany` and `health` to predict `bmi`
- `m_1int` add the interaction term for `exerany` and `health` to `m_1`

We'll assess these models carefully (today) in the training sample and (next time) in the test sample.

- We'll also explore adding a covariate `fruit_day` to the models in several different ways.

Fitting ANOVA model m_1 without interaction

Building a Model (m_1) without interaction

```
m_1 <- lm(bmi ~ exerany + health,  
           data = train_w2im)
```

- How well does this model fit the training data?

```
glance(m_1) %>%  
  select(r.squared, adj.r.squared, sigma, nobs,  
         df, df.residual, AIC, BIC) %>%  
  kable(digits = c(3, 3, 2, 0, 0, 0, 1, 1))
```

r.squared	adj.r.squared	sigma	nobs	df	df.residual	AIC	BIC
0.089	0.082	6.12	670	5	664	4335.9	4367.5

ANOVA for the m_1 model

```
anova(m_1)
```

Analysis of Variance Table

Response: bmi

	Df	Sum Sq	Mean Sq	F value	Pr(>F)	
exerany	1	895.7	895.71	23.948	1.243e-06	***
health	4	1528.5	382.12	10.217	4.952e-08	***
Residuals	664	24834.7	37.40			

Signif. codes:

0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Tidied ANOVA for the m_1 model

```
tidy(anova(m_1)) %>%  
  kable(dig = c(0, 0, 2, 2, 2, 3))
```

term	df	sumsq	meansq	statistic	p.value
exerany	1	895.71	895.71	23.95	0
health	4	1528.47	382.12	10.22	0
Residuals	664	24834.72	37.40	NA	NA

A summary of m_1 coefficients

```
summary(m_1)$coeff
```

	Estimate	Std. Error	t value	Pr(> t)
(Intercept)	27.9094987	0.7428015	37.5732944	1.704557e-166
exerany	-2.1966833	0.5501802	-3.9926613	7.262660e-05
healthVG	0.6176707	0.7026030	0.8791177	3.796555e-01
healthG	3.1372434	0.7224634	4.3424255	1.629287e-05
healthF	3.7122198	0.9070315	4.0927131	4.788419e-05
healthP	4.5514459	1.3577495	3.3521985	8.472164e-04

Tidied summary of m_1 coefficients

```
tidy(m_1, conf.int = TRUE, conf.level = 0.90) %>%  
  kable(digits = c(0,2,2,2,3,2,2))
```

term	estimate	std.error	statistic	p.value	conf.low	conf.high
(Intercept)	27.91	0.74	37.57	0.000	26.69	29.13
exerany	-2.20	0.55	-3.99	0.000	-3.10	-1.29
healthVG	0.62	0.70	0.88	0.380	-0.54	1.77
healthG	3.14	0.72	4.34	0.000	1.95	4.33
healthF	3.71	0.91	4.09	0.000	2.22	5.21
healthP	4.55	1.36	3.35	0.001	2.32	6.79

Equation for Model without Interaction

From `m1` our equation is ...

```
extract_eq(m_1, use_coefs = TRUE, wrap = TRUE)
```

$$\widehat{\text{bmi}} = 27.91 - 2.2(\text{exerany}) + 0.62(\text{health}_{VG}) + 3.14(\text{health}_G) + 3.71(\text{health}_F) + 4.55(\text{health}_P) \quad (1)$$

- You need to use `results = "asis"` in the code chunk label to get this to work.
- This function `extract_eq` comes from the `equatiomatic` package.

Interpreting the m_1 model

$$\widehat{\text{bmi}} = 27.91 - 2.2(\text{exerany}) + 0.62(\text{health}_{VG}) + 3.14(\text{health}_G) + 3.71(\text{health}_F) + 4.55(\text{health}_P) \quad (2)$$

Name	exerany	health	predicted bmi
Harry	0	Excellent	27.91
Sally	1	Excellent	$27.91 - 2.20 = 25.71$
Billy	0	Fair	$27.91 + 3.71 = 31.62$
Meg	1	Fair	$27.91 - 2.20 + 3.71 = 29.42$

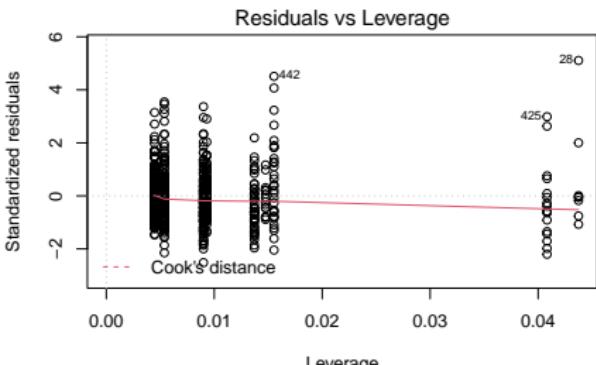
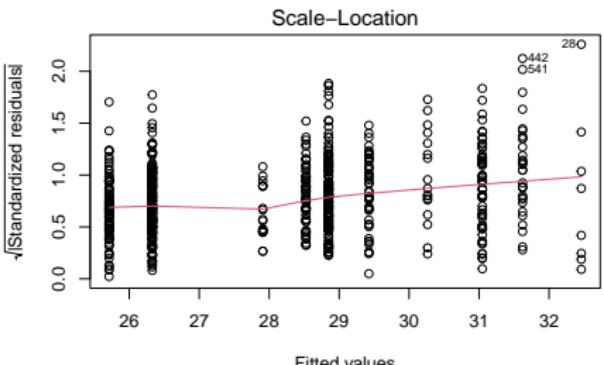
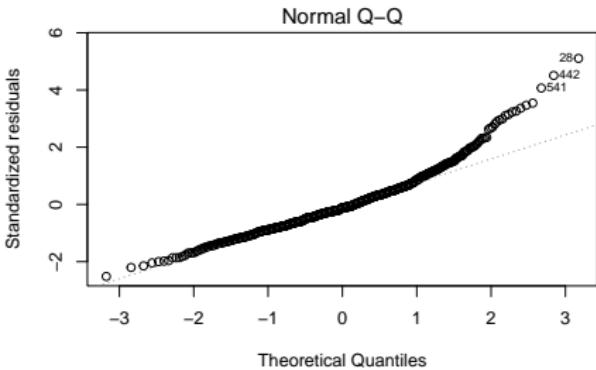
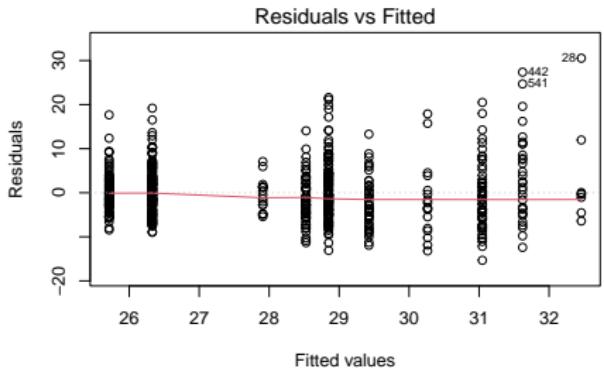
- Effect of exerany?
- Effect of health = Fair instead of Excellent?

Plot the Residuals from model m_1?

```
par(mfrow = c(2,2))
plot(m_1)
par(mfrow = c(1,1))
```

That's the simplest code to get the four key plots to show up in the most familiar pattern, as shown on the next slide...

m_1 Residual Plots (conclusions?)



Fitting ANOVA model `m_1int` including interaction

Adding the interaction term to m_1

```
m_1int <- lm(bmi ~ exerany * health,  
               data = train_w2im)
```

- How does this model compare in terms of fit to the training data?

```
bind_rows(glance(m_1), glance(m_1int)) %>%  
  mutate(mod = c("m_1", "m_1int")) %>%  
  select(mod, r.sq = r.squared, adj.r.sq = adj.r.squared,  
         sigma, nobs, df, df.res = df.residual, AIC, BIC) %>%  
  kable(digits = c(0, 3, 3, 2, 0, 0, 0, 1, 1))
```

mod	r.sq	adj.r.sq	sigma	nobs	df	df.res	AIC	BIC
m_1	0.089	0.082	6.12	670	5	664	4335.9	4367.5
m_1int	0.126	0.114	6.01	670	9	660	4315.8	4365.4

ANOVA for the m_1int model

```
tidy(anova(m_1int)) %>%  
  kable(dig = c(0, 0, 2, 2, 2, 3))
```

term	df	sumsq	meansq	statistic	p.value
exerany	1	895.71	895.71	24.82	0
health	4	1528.47	382.12	10.59	0
exerany:health	4	1020.50	255.13	7.07	0
Residuals	660	23814.22	36.08	NA	NA

ANOVA test comparing m_1 to m_1int

```
anova(m_1, m_1int)
```

Analysis of Variance Table

Model 1: bmi ~ exerany + health

Model 2: bmi ~ exerany * health

	Res.Df	RSS	Df	Sum of Sq	F	Pr(>F)
1	664	24835				
2	660	23814	4	1020.5	7.0707	1.411e-05 ***

Signif. codes:

0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

A summary of m_1int coefficients

```
summary(m_1int)$coeff
```

	Estimate	Std. Error	t value
(Intercept)	27.4872222	1.415826	19.4142627
exerany	-1.6917874	1.548148	-1.0927817
healthVG	-0.6140741	1.634855	-0.3756137
healthG	2.8419157	1.620700	1.7535108
healthF	7.6366487	1.780029	4.2901815
healthP	8.7202778	2.552417	3.4164785
exerany:healthVG	1.6167545	1.803846	0.8962818
exerany:healthG	0.4865311	1.804508	0.2696198
exerany:healthF	-6.2226958	2.072938	-3.0018726
exerany:healthP	-6.0145361	3.004914	-2.0015667
	Pr(> t)		
(Intercept)	9.227071e-67		
exerany	2.748884e-01		
healthVG	7.073248e-01		

Tidied summary of m_1int coefficients

```
tidy(m_1int, conf.int = TRUE, conf.level = 0.90) %>%
  rename(se = std.error, t = statistic, p = p.value) %>%
  kable(digits = c(0,2,2,2,3,2,2))
```

term	estimate	se	t	p	conf.low	conf.high
(Intercept)	27.49	1.42	19.41	0.000	25.16	29.82
exerany	-1.69	1.55	-1.09	0.275	-4.24	0.86
healthVG	-0.61	1.63	-0.38	0.707	-3.31	2.08
healthG	2.84	1.62	1.75	0.080	0.17	5.51
healthF	7.64	1.78	4.29	0.000	4.70	10.57
healthP	8.72	2.55	3.42	0.001	4.52	12.92
exerany:healthVG	1.62	1.80	0.90	0.370	-1.35	4.59
exerany:healthG	0.49	1.80	0.27	0.788	-2.49	3.46
exerany:healthF	-6.22	2.07	-3.00	0.003	-9.64	-2.81
exerany:healthP	-6.01	3.00	-2.00	0.046	-10.96	-1.06

Equation for Interaction Model

From m1_int our equation is ...

```
extract_eq(m_1int, use_coefs = TRUE,  
          wrap = TRUE, terms_per_line = 2)
```

$$\widehat{\text{bmi}} = 27.49 - 1.69(\text{exerany}) - 0.61(\text{health}_{VG}) + 2.84(\text{health}_G) + 7.64(\text{health}_F) + 8.72(\text{health}_P) + 1.62(\text{exerany} \times \text{health}_{VG}) + 0.49(\text{exerany} \times \text{health}_G) - 6.22(\text{exerany} \times \text{health}_F) - 6.01(\text{exerany} \times \text{health}_P) \quad (3)$$

Don't forget to use results = "asis" in the code chunk label.

Interpreting the m_1int model

$$\widehat{\text{bmi}} = 27.49 - 1.69(\text{exerany}) - 0.61(\text{health}_{VG}) + 2.84(\text{health}_G) + 7.64(\text{health}_F) + 8.72(\text{health}_P) + 1.62(\text{exerany} \times \text{health}_{VG}) + 0.49(\text{exerany} \times \text{health}_G) - 6.22(\text{exerany} \times \text{health}_F) - 6.01(\text{exerany} \times \text{health}_P) \quad (4)$$

Name	exerany	health	predicted bmi
Harry	0	Excellent	27.49
Sally	1	Excellent	$27.49 - 1.69 = 25.80$
Billy	0	Fair	$27.49 + 7.64 = 35.13$
Meg	1	Fair	$27.49 - 1.69 + 7.64 - 6.22 = 27.22$

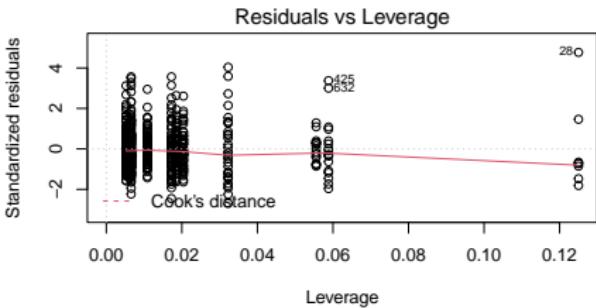
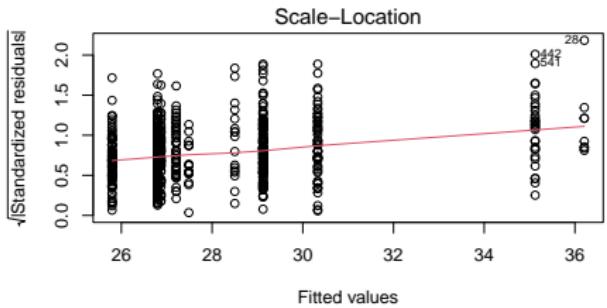
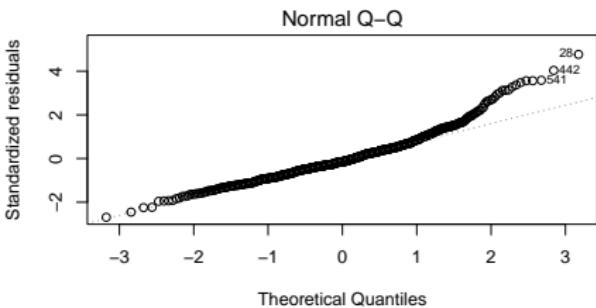
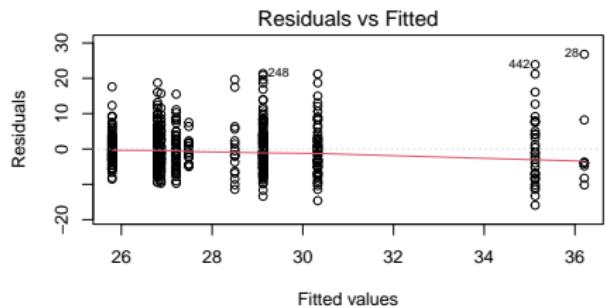
- How do we interpret effect sizes here?

Interpreting the m_1int model

Name	exerany	health	predicted bmi
Harry	0	Excellent	27.49
Sally	1	Excellent	$27.49 - 1.69 = 25.80$
Billy	0	Fair	$27.49 + 7.64 = 35.13$
Meg	1	Fair	$27.49 - 1.69 + 7.64 - 6.22 = 27.22$

- How do we interpret effect sizes here? **It depends.**
- Effect of exerany?
 - If health = Excellent, effect is -1.69
 - If health = Fair, effect is $(-1.69 - 6.22) = -7.91$
- Effect of health = Fair instead of Excellent?
 - If exerany = 0 (no), effect is 7.64
 - If exerany = 1 (yes), effect is $(7.64 - 6.22) = 1.42$

Plot the Residuals from model m_1int?



Incorporating a Covariate into our two-way ANOVA models

Taking Stock

So far, we've fit two models to predict bmi, using exerany and health, one with an interaction term and one without.

```
m_1 <- lm(bmi ~ exerany + health, data = train_w2im)  
m_1int <- lm(bmi ~ exerany * health, data = train_w2im)
```

Next, we'll fit models incorporating a covariate, specifically, fruit_day, a quantity (servings/day).

- m_2 and m_2int will add a linear term for fruit_day
- Later models (we'll fit next time) will add various non-linear terms in fruit_day
- We'll assess these models in our testing sample (next time) as well as our training sample.

Giving away the ending: We'll see that none of these augmented models will clearly improve the fit in our test sample over the performance of m_1 and m_1int.

Adding in the covariate fruit_day to m_1

```
m_2 <- lm(bmi ~ fruit_day + exerany + health,  
           data = train_w2im)
```

- How well does this model fit the training data?

```
bind_rows(glance(m_1), glance(m_2)) %>%  
  mutate(mod = c("m_1", "m_2")) %>%  
  select(mod, r.sq = r.squared, adj.r.sq = adj.r.squared,  
         sigma, df, df.res = df.residual, AIC, BIC) %>%  
  kable(digits = c(0, 3, 3, 2, 0, 0, 1, 1))
```

mod	r.sq	adj.r.sq	sigma	df	df.res	AIC	BIC
m_1	0.089	0.082	6.12	5	664	4335.9	4367.5
m_2	0.098	0.090	6.09	6	663	4331.2	4367.3

- Also available in `glance` for a model fit with `lm` are `statistic`, `p.value`, `logLik`, and `deviance`.

ANOVA for the m_2 model

```
tidy(anova(m_2)) %>%
  kable(dig = c(0, 0, 2, 2, 2, 3))
```

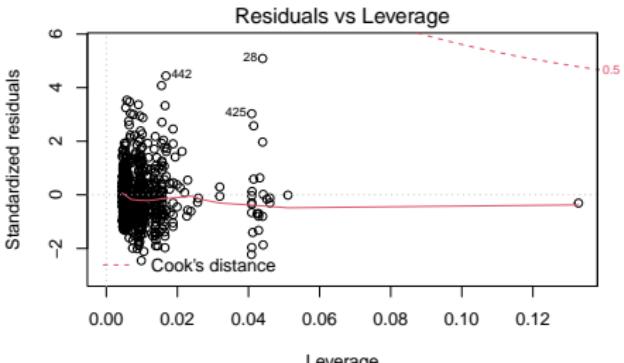
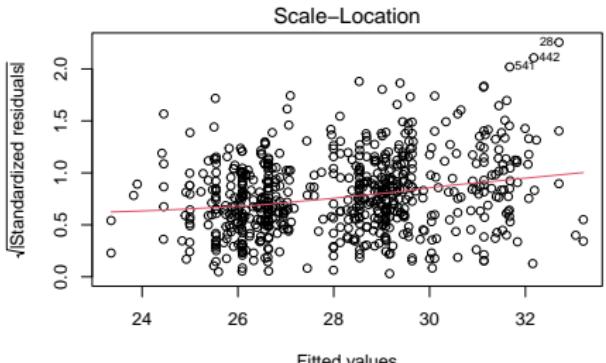
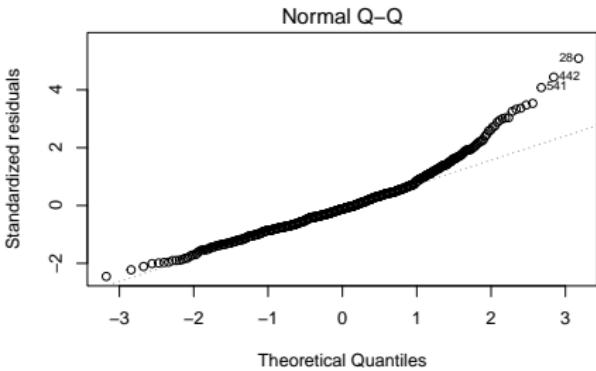
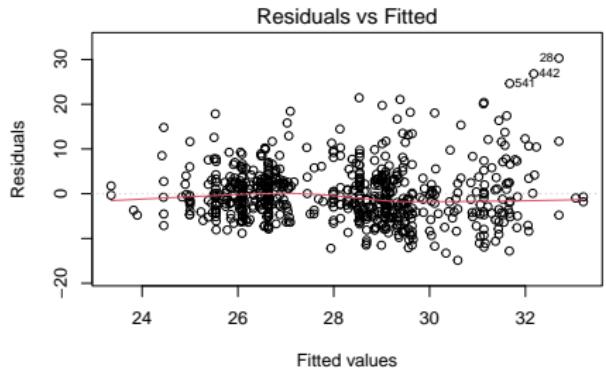
term	df	sumsq	meansq	statistic	p.value
fruit_day	1	468.10	468.10	12.62	0
exerany	1	760.50	760.50	20.51	0
health	4	1441.63	360.41	9.72	0
Residuals	663	24588.68	37.09	NA	NA

Tidied summary of m_2 coefficients

```
tidy(m_2, conf.int = TRUE, conf.level = 0.90) %>%
  kable(digits = c(0,2,2,2,3,2,2))
```

term	estimate	std.error	statistic	p.value	conf.low	conf.high
(Intercept)	28.68	0.80	35.94	0.000	27.37	30.00
fruit_day	-0.55	0.21	-2.58	0.010	-0.90	-0.20
exerany	-2.05	0.55	-3.71	0.000	-2.95	-1.14
healthVG	0.55	0.70	0.79	0.430	-0.60	1.71
healthG	3.00	0.72	4.16	0.000	1.81	4.19
healthF	3.55	0.91	3.92	0.000	2.06	5.04
healthP	4.57	1.35	3.38	0.001	2.34	6.79

m_2 Residual Plots (non-constant variance?)



Who is that poorest fit case?

Plot suggests we look at row 28

```
train_w2im %>% slice(28) %>%  
  select(ID, bmi, fruit_day, exerany, health) %>% kable()
```

ID	bmi	fruit_day	exerany	health
320	63		1	0 P

What is unusual about this subject?

```
train_w2im %$% sort(bmi) %>% tail()
```

```
[1] 50.46 51.22 51.54 56.31 58.98 63.00
```

What if we included the interaction term?

```
m_2int <- lm(bmi ~ fruit_day + exerany * health,  
               data = train_w2im)
```

Compare m_2int fit to previous models...

mod	r.sq	adj.r.sq	sigma	df	df.res	AIC	BIC
m_1	0.089	0.082	6.12	5	664	4335.9	4367.5
m_2	0.098	0.090	6.09	6	663	4331.2	4367.3
m_1int	0.126	0.114	6.01	9	660	4315.8	4365.4
m_2int	0.138	0.125	5.97	10	659	4309.1	4363.2

- m_1 = no fruit_day, no exerany*health interaction
- m_2 = fruit_day, but no interaction
- m_1int = no fruit_day, with interaction
- m_2int = both fruit_day and interaction

ANOVA for the m_2int model

```
tidy(anova(m_2int)) %>%  
  kable(dig = c(0, 0, 2, 2, 2, 3))
```

term	df	sumsq	meansq	statistic	p.value
fruit_day	1	468.10	468.10	13.12	0
exerany	1	760.50	760.50	21.32	0
health	4	1441.63	360.41	10.10	0
exerany:health	4	1080.39	270.10	7.57	0
Residuals	659	23508.29	35.67	NA	NA

Tidied summary of m_2int coefficients

```
tidy(m_2int, conf.int = TRUE, conf.level = 0.90) %>%
  rename(se = std.error, t = statistic, p = p.value) %>%
  kable(digits = c(0,2,2,2,3,2,2))
```

term	estimate	se	t	p	conf.low	conf.high
(Intercept)	28.28	1.43	19.73	0.000	25.91	30.64
fruit_day	-0.61	0.21	-2.93	0.004	-0.96	-0.27
exerany	-1.43	1.54	-0.93	0.353	-3.97	1.11
healthVG	-0.66	1.63	-0.40	0.686	-3.34	2.02
healthG	2.75	1.61	1.71	0.088	0.10	5.41
healthF	7.59	1.77	4.29	0.000	4.67	10.50
healthP	9.12	2.54	3.59	0.000	4.93	13.30
exerany:healthVG	1.59	1.79	0.88	0.377	-1.37	4.54
exerany:healthG	0.41	1.79	0.23	0.819	-2.54	3.37
exerany:healthF	-6.41	2.06	-3.11	0.002	-9.81	-3.02
exerany:healthP	-6.55	2.99	-2.19	0.029	-11.48	-1.62

ANOVA comparison of m_2 and m_2int

```
anova(m_2, m_2int)
```

Analysis of Variance Table

Model 1: bmi ~ fruit_day + exerany + health

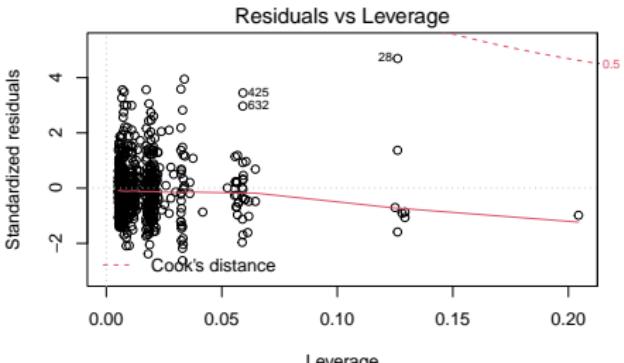
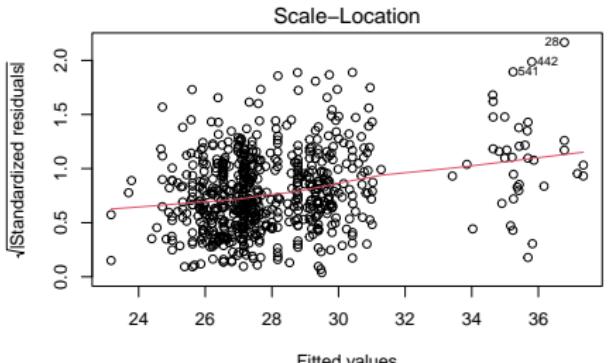
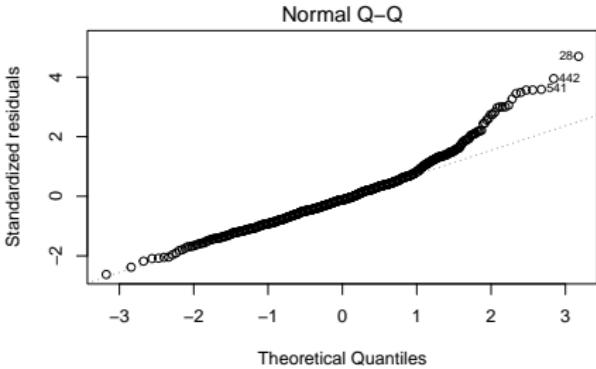
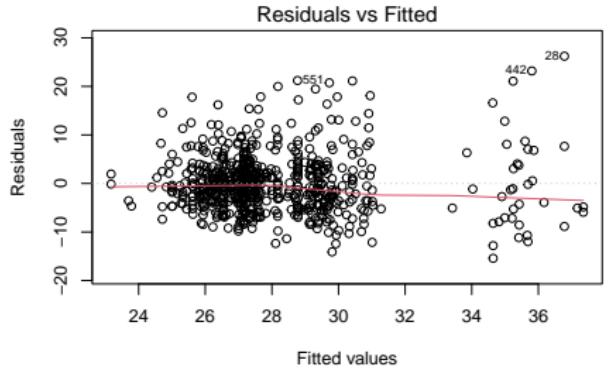
Model 2: bmi ~ fruit_day + exerany * health

Res.Df	RSS	Df	Sum of Sq	F	Pr(>F)
1	663	24589			
2	659	23508	4	1080.4	7.5716 5.751e-06 ***

Signif. codes:

0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual plots for model m_2int?



Which of the four models fits best?

In the **training** sample, we have...

mod	r.sq	adj.r.sq	sigma	df	df.res	AIC	BIC
m_1	0.089	0.082	6.12	5	664	4335.9	4367.5
m_2	0.098	0.090	6.09	6	663	4331.2	4367.3
m_1int	0.126	0.114	6.01	9	660	4315.8	4365.4
m_2int	0.138	0.125	5.97	10	659	4309.1	4363.2

- Adjusted R^2 , σ , AIC and BIC all improve as we move down from m1 towards m2_int.
- BUT the testing sample cannot judge between models accurately. Our models have already *seen* that data.
- For fairer comparisons, we'll need to also consider the (held out) testing sample.

Next Time

- Feedback from the Minute Paper after Class 03, due tomorrow at Noon, please.
- Assessing the models we've fit so far in the testing sample
- Incorporating polynomial terms and splines into linear regression (ANCOVA) models

432 Class 04 Slides

thomaselove.github.io/432

2022-01-20

Today's Agenda

- Data Load from .Rds built in Class 3
- Fitting models with lm
 - Incorporating an interaction between factors
 - Incorporating polynomial terms
 - Incorporating restricted cubic splines
- Evaluating results in a testing sample with yardstick

Setup

```
knitr::opts_chunk$set(comment = NA)
options(width = 60)

library(here); library(magrittr)
library(janitor); library(knitr)
library(patchwork); library(broom)
library(rsample); library(yardstick)

library(rms)          ## new today: from Frank Harrell

library(tidyverse)

theme_set(theme_bw())
options(dplyr.summarise.inform = FALSE)
```

From Class 3

We developed the week2 data and performed a simple imputation for it (into week2im) in Class 3. Here, we'll read in those saved results, and then split into testing and training samples, as we did in Class 3.

```
week2 <- readRDS(here("data", "week2.Rds"))

week2im <- readRDS(here("data", "week2im.Rds"))

set.seed(432)
week2im_split <- initial_split(week2im, prop = 3/4)

train_w2im <- training(week2im_split)
test_w2im <- testing(week2im_split)
```

Codebook for useful week2 variables

- 894 subjects in Cleveland-Elyria with bmi and no history of diabetes

Variable	Description
bmi	(outcome) Body-Mass index in kg/m ² .
inc_imp	income (imputed from grouped values) in \$
fruit_day	average fruit servings consumed per day
drinks_wk	average alcoholic drinks consumed per week
female	sex: 1 = female, 0 = male
exerany	any exercise in the past month: 1 = yes, 0 = no
genhealth	self-reported overall health (5 levels)
race_eth	race and Hispanic/Latinx ethnicity (5 levels)

- plus ID, SEQNO, hx_diabetes (all 0), MMSA (all Cleveland-Elyria)
- See Chapter 2 of the Course Notes for details on the variables

Class 03 and Class 04

In Class 03, we fit two models to predict bmi, using exerany and health, one with an interaction term and one without.

```
m_1 <- lm(bmi ~ exerany + health, data = train_w2im)
m_1int <- lm(bmi ~ exerany * health, data = train_w2im)
```

Adding in the covariate fruit_day...

```
m_2 <- lm(bmi ~ fruit_day + exerany + health,
           data = train_w2im)
m_2int <- lm(bmi ~ fruit_day + exerany * health,
              data = train_w2im)
```

Compare fits in the training sample?

mod	r.sq	adj.r.sq	sigma	df	df.res	AIC	BIC
m_1	0.089	0.082	6.12	5	664	4335.9	4367.5
m_2	0.098	0.090	6.09	6	663	4331.2	4367.3
m_1int	0.126	0.114	6.01	9	660	4315.8	4365.4
m_2int	0.138	0.125	5.97	10	659	4309.1	4363.2

- m_1 = no fruit_day
- m_2 = fruit_day included
- int = exerany*health interaction included

But the training sample cannot judge between models accurately. Our models have already *seen* that data. So fairer comparisons, consider the (held out) testing sample...

Model predictions of bmi in the test sample

We'll use augment from the broom package...

```
m1_test_aug <- augment(m_1, newdata = test_w2im)
m1int_test_aug <- augment(m_1int, newdata = test_w2im)
m2_test_aug <- augment(m_2, newdata = test_w2im)
m2int_test_aug <- augment(m_2int, newdata = test_w2im)
```

This adds fitted values (predictions) and residuals (errors) ...

```
m1_test_aug %>% select(ID, bmi, .fitted, .resid) %>%
  slice(1:2) %>% kable()
```

ID	bmi	.fitted	.resid
4	26.51	28.85006	-2.340059
5	24.25	28.85006	-4.600059

What does the yardstick package do?

For each subject in the testing set, we will need:

- estimate = model's prediction of that subject's `bmi`
- truth = the `bmi` value observed for that subject

Calculate a summary of the predictions across the n test subjects, such as:

- R^2 = square of the correlation between truth and estimate
- `mae` = mean absolute error ...

$$mae = \frac{1}{n} \sum |truth - estimate|$$

- `rmse` = root mean squared error ...

$$rmse = \sqrt{\frac{1}{n} \sum (truth - estimate)^2}$$

Testing Results (using R^2)

We can use the `yardstick` package and its `rsq()` function.

```
testing_r2 <- bind_rows(  
  rsq(m1_test_aug, truth = bmi, estimate = .fitted),  
  rsq(m1int_test_aug, truth = bmi, estimate = .fitted),  
  rsq(m2_test_aug, truth = bmi, estimate = .fitted),  
  rsq(m2int_test_aug, truth = bmi, estimate = .fitted)) %>%  
  mutate(model = c("m_1", "m_1int", "m_2", "m_2int"))  
testing_r2 %>% kable(dig = 4)
```

.metric	.estimator	.estimate	model
rsq	standard	0.0712	m_1
rsq	standard	0.0395	m_1int
rsq	standard	0.0647	m_2
rsq	standard	0.0361	m_2int

Mean Absolute Error?

Consider the mean absolute prediction error ...

```
testing_mae <- bind_rows(  
  mae(m1_test_aug, truth = bmi, estimate = .fitted),  
  mae(m1int_test_aug, truth = bmi, estimate = .fitted),  
  mae(m2_test_aug, truth = bmi, estimate = .fitted),  
  mae(m2int_test_aug, truth = bmi, estimate = .fitted)) %>%  
  mutate(model = c("m_1", "m_1int", "m_2", "m_2int"))  
testing_mae %>% kable(dig = 2)
```

.metric	.estimator	.estimate	model
mae	standard	4.43	m_1
mae	standard	4.63	m_1int
mae	standard	4.48	m_2
mae	standard	4.71	m_2int

Root Mean Squared Error?

How about the square root of the mean squared prediction error, or RMSE?

```
testing_rmse <- bind_rows(  
  rmse(m1_test_aug, truth = bmi, estimate = .fitted),  
  rmse(m1int_test_aug, truth = bmi, estimate = .fitted),  
  rmse(m2_test_aug, truth = bmi, estimate = .fitted),  
  rmse(m2int_test_aug, truth = bmi, estimate = .fitted)) %>%  
  mutate(model = c("m_1", "m_1int", "m_2", "m_2int"))  
testing_rmse %>% kable(digits = 3)
```

.metric	.estimator	.estimate	model
rmse	standard	5.736	m_1
rmse	standard	6.033	m_1int
rmse	standard	5.778	m_2
rmse	standard	6.091	m_2int

Other Summaries for Numerical Predictions

Within the `yardstick` package, there are several other summaries, including:

- `rsq_trad()` = defines R^2 using sums of squares.
 - The `rsq()` measure we showed a few slides ago is a squared correlation coefficient and is guaranteed to fall in $(0, 1)$.
- `mape()` = mean absolute percentage error
- `mpe()` = mean percentage error
- `huber_loss()` = Huber loss (often used in robust regression), which is less sensitive to outliers than `rmse()`.
- `ccc()` = concordance correlation coefficient, which attempts to measure both consistency/correlation (like `rsq()`) and accuracy (like `rmse()`).

See the `yardstick` home page for more details.

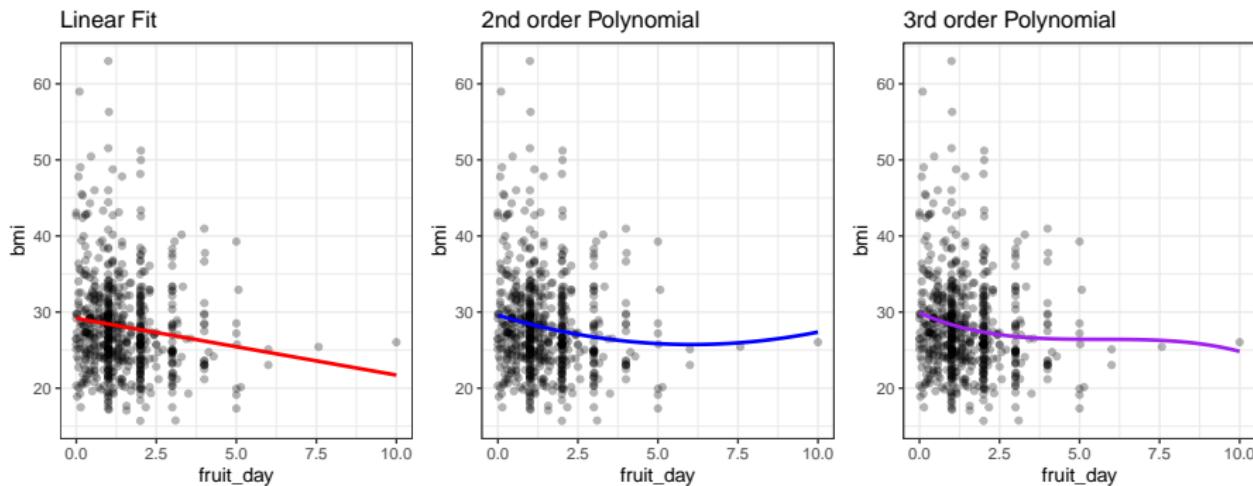
Incorporating a non-linear term for fruit_day

Suppose we wanted to include a polynomial term for fruit_day:

```
lm(bmi ~ fruit_day, data = train_w2im)
```

```
lm(bmi ~ poly(fruit_day, 2), data = train_w2im)
```

```
lm(bmi ~ poly(fruit_day, 3), data = train_w2im)
```



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: $y = \beta_0 + \beta_1 x$
- Quadratic: $y = \beta_0 + \beta_1 x + \beta_2 x^2$
- Cubic: $y = \beta_0 + \beta_1 x + \beta_2 x^2 + \beta_3 x^3$
- Quartic: $y = \beta_0 + \beta_1 x + \beta_2 x^2 + \beta_3 x^3 + \beta_4 x^4$
- Quintic: $y = \beta_0 + \beta_1 x + \beta_2 x^2 + \beta_3 x^3 + \beta_4 x^4 + \beta_5 x^5$

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

Raw Polynomials vs. Orthogonal Polynomials

Predict bmi using fruit_day with a polynomial of degree 2.

```
(temp1 <- lm(bmi ~ fruit_day + I(fruit_day^2),  
            data = train_w2im))
```

Call:

```
lm(formula = bmi ~ fruit_day + I(fruit_day^2), data = train_w2im)
```

Coefficients:

	fruit_day	I(fruit_day^2)
(Intercept)		
29.5925	-1.2733	0.1051

This uses raw polynomials. Predicted bmi for fruit_day = 2 is

$$\begin{aligned}\text{bmi} &= 29.5925 - 1.2733 \text{ (fruit_day)} + 0.1051 \text{ (fruit_day}^2\text{)} \\ &= 29.5925 - 1.2733 \text{ (2)} + 0.1051 \text{ (4)} \\ &= 27.466\end{aligned}$$

Does the raw polynomial match our expectations?

```
temp1 <- lm(bmi ~ fruit_day + I(fruit_day^2),  
             data = train_w2im)  
  
augment(temp1, newdata = tibble(fruit_day = 2)) %>%  
  kable(digits = 3)
```

fruit_day	.fitted
2	27.466

and this matches our “by hand” calculation. But it turns out most regression models use *orthogonal* rather than raw polynomials...

Fitting an Orthogonal Polynomial

Predict bmi using fruit_day with an *orthogonal* polynomial of degree 2.

```
(temp2 <- lm(bmi ~ poly(fruit_day, 2), data = train_w2im))
```

Call:

```
lm(formula = bmi ~ poly(fruit_day, 2), data = train_w2im)
```

Coefficients:

(Intercept)	poly(fruit_day, 2)1
28.089	-21.636
poly(fruit_day, 2)2	
8.009	

This looks very different from our previous version of the model.

- What happens when we make a prediction, though?

Prediction in the Orthogonal Polynomial Model

Remember that in our raw polynomial model, our “by hand” and “using R” calculations both concluded that the predicted bmi for a subject with fruit_day = 2 was 27.466.

Now, what happens with the orthogonal polynomial model temp2 we just fit?

```
augment(temp2, newdata = data.frame(fruit_day = 2)) %>%  
  kable(digits = 3)
```

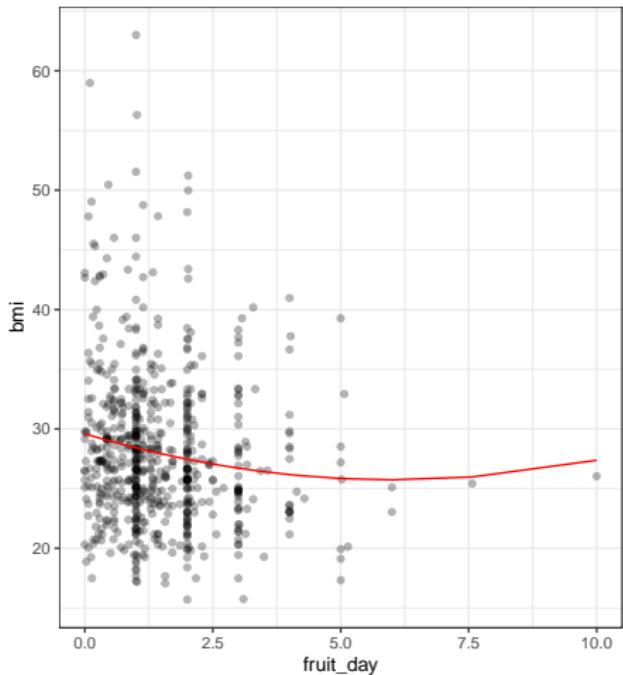
fruit_day	.fitted
2	27.466

- No change in the prediction.

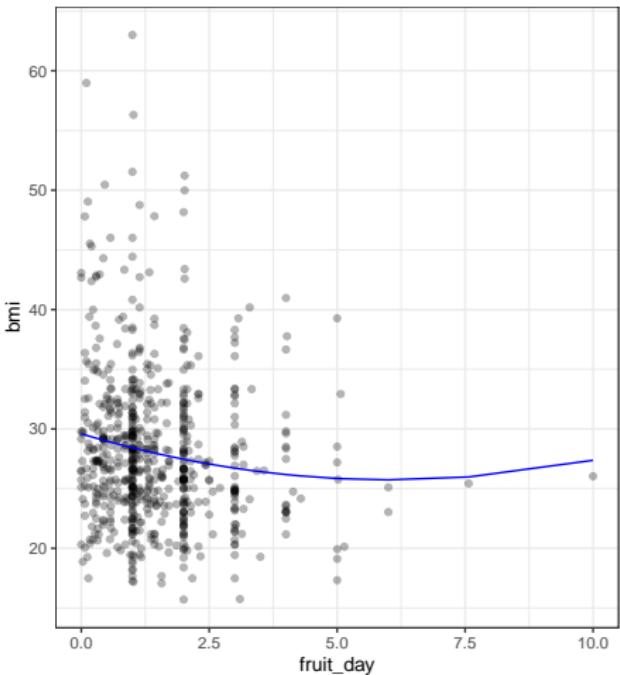
Fits of raw vs orthogonal polynomials

Comparing Two Methods of Fitting a Quadratic Polynomial

temp1: Raw fit, degree 2



temp2: Orthogonal fit, degree 2



- The two models are, in fact, identical.

Why do we 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...

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

```
fruit_day I(fruit_day^2)  
4.652178     4.652178
```

- Orthogonal polynomial terms are uncorrelated with one another, easing the process of identifying which terms add value to our model.

```
vif(temp2)
```

```
poly(fruit_day, 2)1 poly(fruit_day, 2)2  
1                      1
```

Why orthogonal rather than raw polynomials?

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 usually avoid polynomials in our practical work, and instead use splines, 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.

Adding a Second Order Polynomial to our Models

```
m_3 <- lm(bmi ~ poly(fruit_day, 2) + exerany + health,  
           data = train_w2im)
```

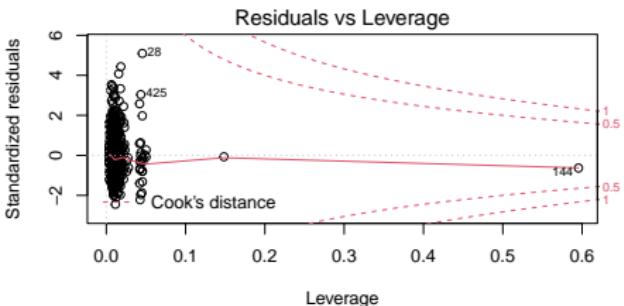
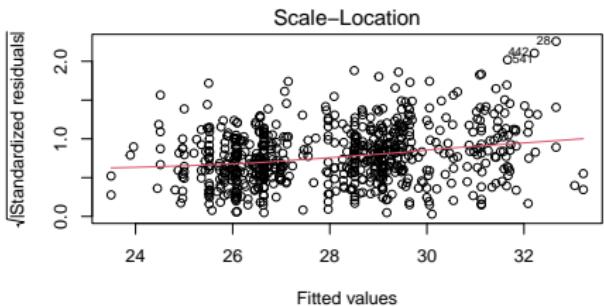
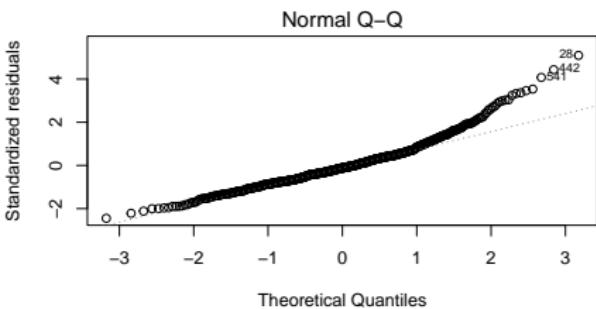
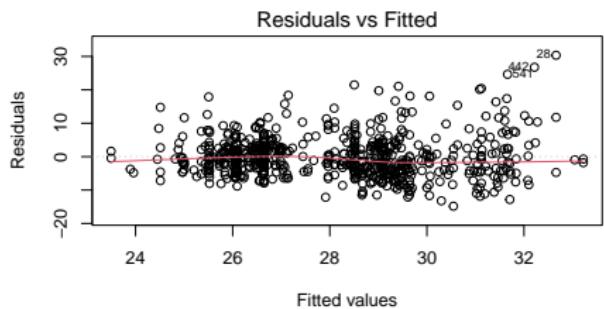
- Comparison to other models without the interaction...

mod	r.sq	adj.r.sq	sigma	df	df.res	AIC	BIC
m_1	0.0889	0.0821	6.12	5	664	4335.9	4367.5
m_2	0.0980	0.0898	6.09	6	663	4331.2	4367.3
m_3	0.0980	0.0885	6.09	7	662	4333.2	4373.8

Tidied summary of m_3 coefficients

term	est	se	t	p	conf.low	conf.high
(Intercept)	27.87	0.74	37.54	0.000	26.65	29.10
poly(fruit_day, 2)1	-15.88	6.16	-2.58	0.010	-26.04	-5.73
poly(fruit_day, 2)2	1.08	6.24	0.17	0.863	-9.20	11.35
exerany	-2.03	0.56	-3.63	0.000	-2.95	-1.11
healthVG	0.55	0.70	0.79	0.430	-0.60	1.71
healthG	3.00	0.72	4.15	0.000	1.81	4.19
healthF	3.55	0.91	3.92	0.000	2.06	5.04
healthP	4.54	1.36	3.33	0.001	2.29	6.78

m_3 Residual Plots



Add in the interaction

```
m_3int <- lm(bmi ~ poly(fruit_day, 2) + exerany * health,  
               data = train_w2im)
```

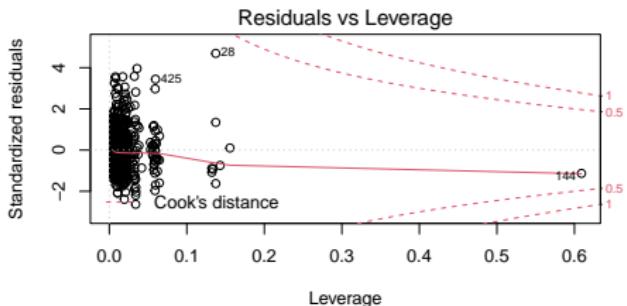
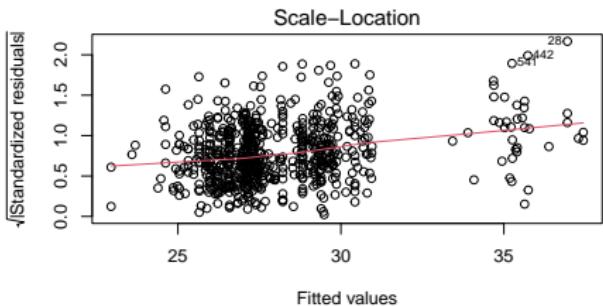
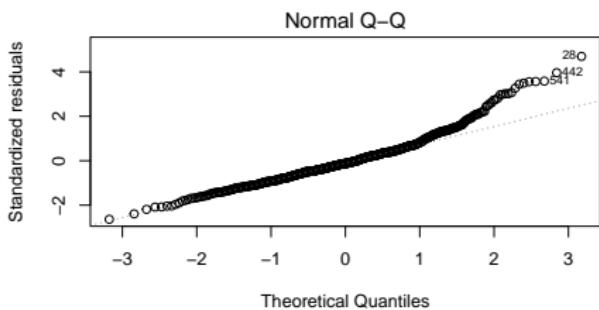
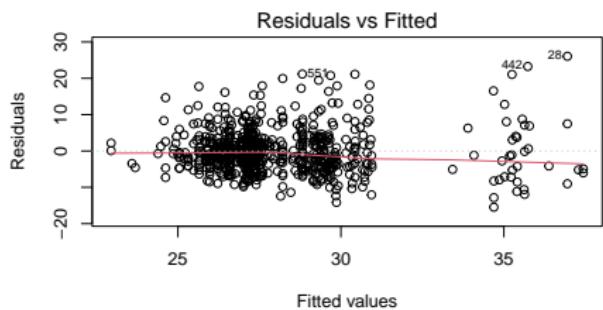
- Comparison to other models with the interaction...

mod	r.sq	adj.r.sq	sigma	df	df.res	AIC	BIC
m_1int	0.1264	0.1145	6.01	9	660	4315.8	4365.4
m_2int	0.1376	0.1245	5.97	10	659	4309.1	4363.2
m_3int	0.1377	0.1233	5.98	11	658	4311.1	4369.6

Tidied summary of m_3int coefficients

term	est	se	t	p	conf.low	conf.high
(Intercept)	27.40	1.41	19.43	0.000	25.07	29.72
poly(fruit_day, 2)1	-17.73	6.06	-2.93	0.004	-27.71	-7.75
poly(fruit_day, 2)2	-1.66	6.29	-0.26	0.792	-12.02	8.70
exerany	-1.46	1.55	-0.94	0.346	-4.00	1.09
healthVG	-0.66	1.63	-0.40	0.686	-3.34	2.02
healthG	2.75	1.61	1.71	0.089	0.09	5.41
healthF	7.58	1.77	4.28	0.000	4.66	10.50
healthP	9.27	2.61	3.55	0.000	4.97	13.56
exerany:healthVG	1.59	1.79	0.88	0.377	-1.37	4.54
exerany:healthG	0.42	1.80	0.23	0.814	-2.54	3.38
exerany:healthF	-6.40	2.06	-3.10	0.002	-9.80	-3.00
exerany:healthP	-6.71	3.05	-2.20	0.028	-11.74	-1.68

m_3int Residual Plots



How do models m_3 and m_3int do in testing?

```
m3_test_aug <- augment(m_3, newdata = test_w2im)
m3int_test_aug <- augment(m_3int, newdata = test_w2im)

testing_r2 <- bind_rows(
  rsq(m1_test_aug, truth = bmi, estimate = .fitted),
  rsq(m2_test_aug, truth = bmi, estimate = .fitted),
  rsq(m3_test_aug, truth = bmi, estimate = .fitted),
  rsq(m1int_test_aug, truth = bmi, estimate = .fitted),
  rsq(m2int_test_aug, truth = bmi, estimate = .fitted),
  rsq(m3int_test_aug, truth = bmi, estimate = .fitted)) %>%
  mutate(model = c("m_1", "m_2", "m_3", "m_1int",
                  "m_2int", "m_3int"))
```

- I've hidden my calculations for RMSE and MAE here.

Results comparing all six models (testing)

```
bind_cols(testing_r2 %>% select(model, rsquare = .estimate),  
          testing_rmse %>% select(rmse = .estimate),  
          testing_mae %>% select(mae = .estimate)) %>%  
kable(digits = c(0, 4, 3, 3))
```

model	rsquare	rmse	mae
m_1	0.0712	5.736	4.432
m_2	0.0647	5.778	4.480
m_3	0.0651	5.776	4.480
m_1int	0.0395	6.033	4.628
m_2int	0.0361	6.091	4.710
m_3int	0.0354	6.098	4.711

- Did the polynomial term in m_3 and m_3int improve our predictions?

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.
 - 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.

A simulated data set

```
set.seed(4322021)

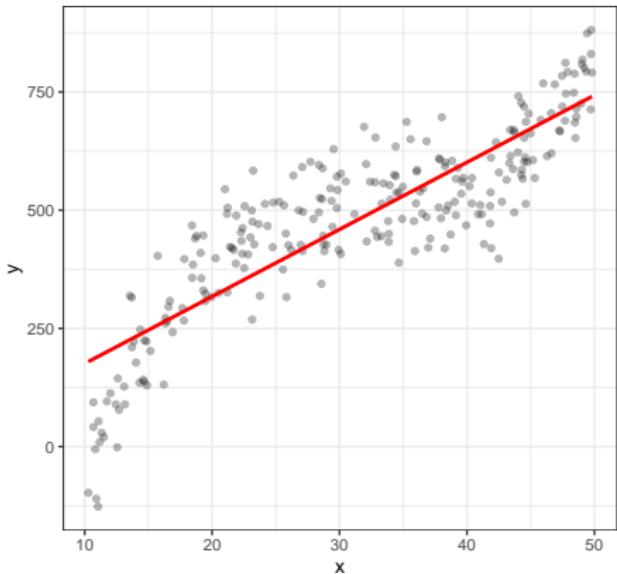
sim_data <- tibble(
  x = runif(250, min = 10, max = 50),
  y = 3*(x-30) - 0.3*(x-30)^2 + 0.05*(x-30)^3 +
    rnorm(250, mean = 500, sd = 70)
)

head(sim_data, 2)

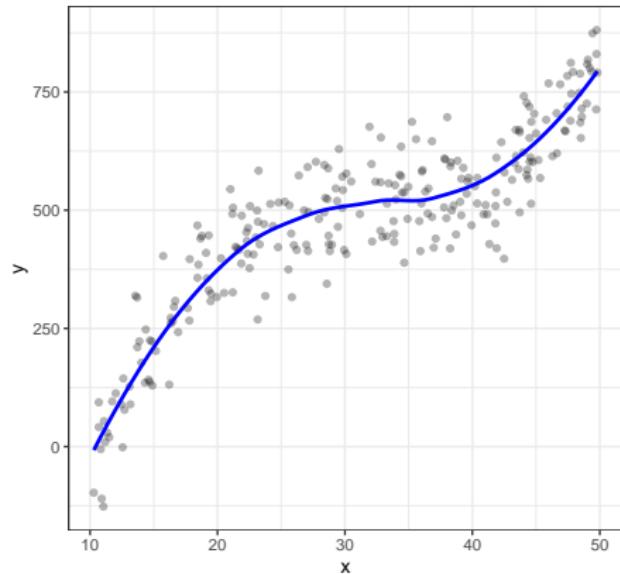
# A tibble: 2 x 2
  x     y
  <dbl> <dbl>
1 42.5  397.
2 35.9  414.
```

The sim_data, plotted.

With Linear Fit



With Loess Smooth

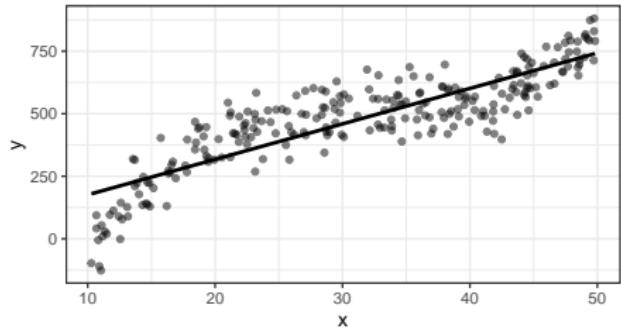


Fitting Restricted Cubic Splines with lm and rcs

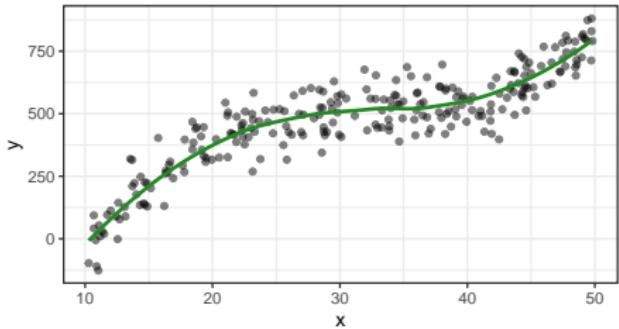
```
sim_linear <- lm(y ~ x, data = sim_data)
sim_poly2 <- lm(y ~ poly(x, 2), data = sim_data)
sim_poly3 <- lm(y ~ poly(x, 3), data = sim_data)
sim_rcs3 <- lm(y ~ rcs(x, 3), data = sim_data)
sim_rcs4 <- lm(y ~ rcs(x, 4), data = sim_data)
sim_rcs5 <- lm(y ~ rcs(x, 5), data = sim_data)
```

Looking at the Polynomial Fits

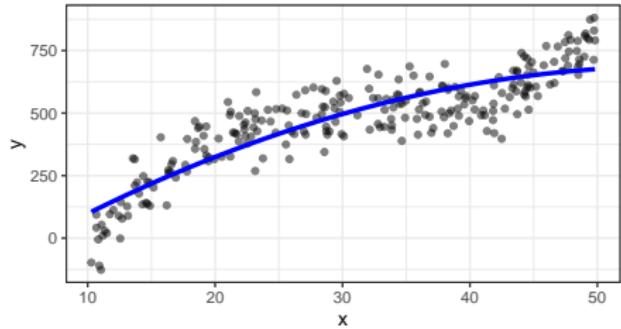
Linear Fit



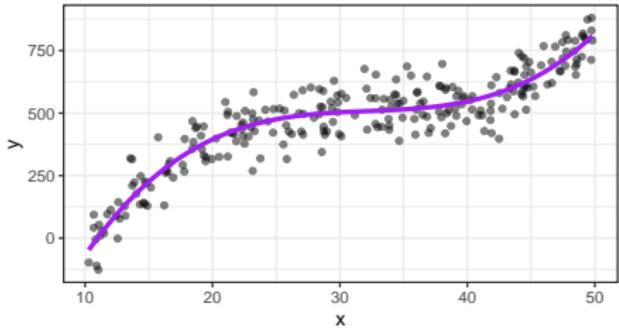
Loess Smooth



Quadratic Polynomial

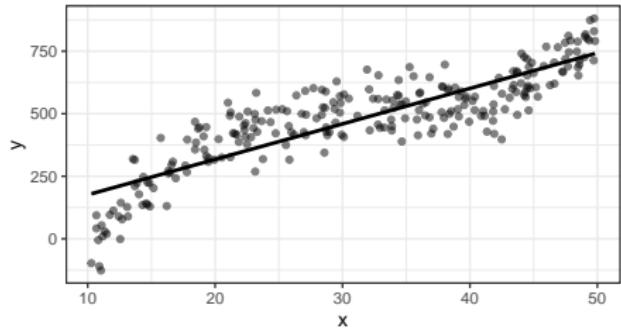


Cubic Polynomial

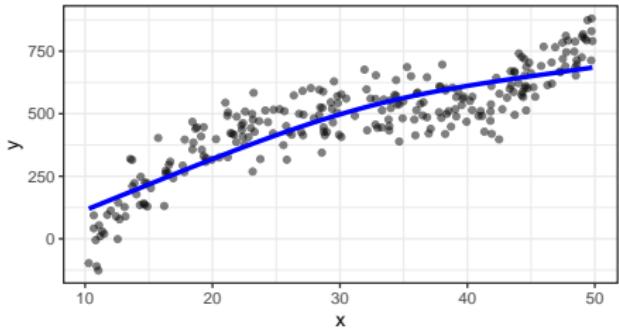


Looking at the Restricted Cubic Spline Fits

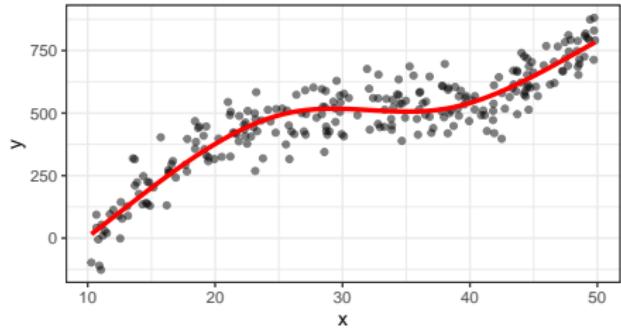
Linear Fit



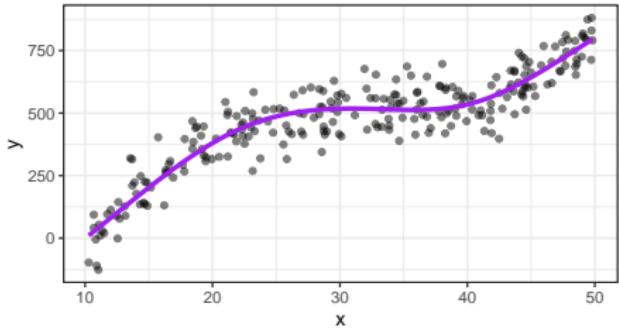
RCS with 3 knots



RCS with 4 knots



RCS with 5 knots



Fitting Restricted Cubic Splines with lm and rcs

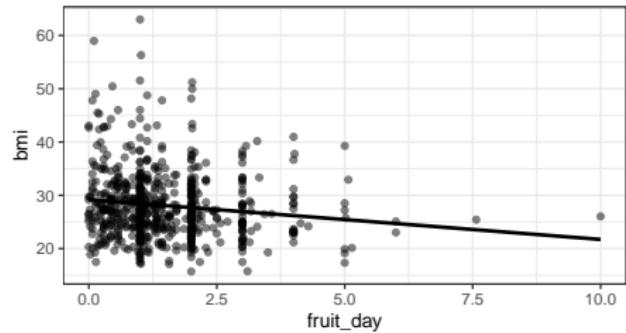
For most applications, three to five knots strike a nice balance between complicating the model needlessly and fitting data pleasingly. Let's consider a restricted cubic spline model for `bmi` based on `fruit_day` again, but now with:

- in `temp3`, 3 knots, and
- in `temp4`, 4 knots,

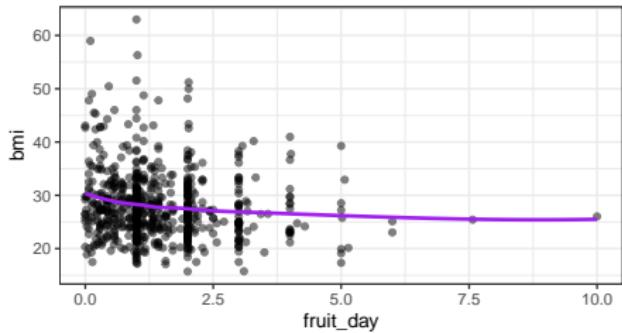
```
temp3 <- lm(bmi ~ rcs(fruit_day, 3), data = train_w2im)
temp4 <- lm(bmi ~ rcs(fruit_day, 4), data = train_w2im)
```

Spline models for bmi and fruit_day

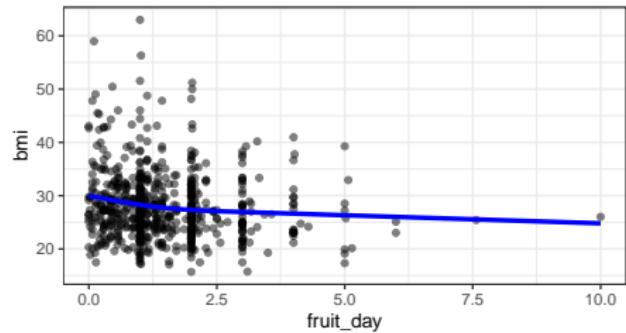
Linear Fit



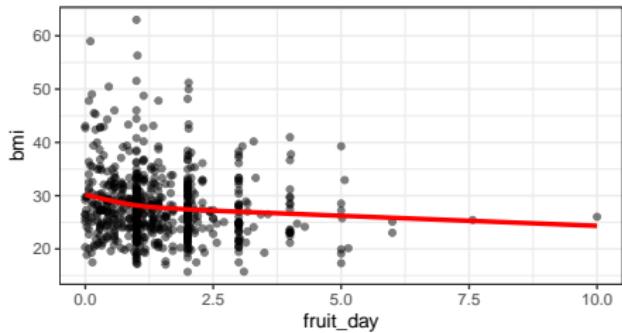
Loess Smooth



RCS, 3 knots



RCS, 4 knots



Let's try an RCS with 4 knots

```
m_4 <- lm(bmi ~ rcs(fruit_day, 4) + exerany + health,  
           data = train_w2im)  
  
m_4int <- lm(bmi ~ rcs(fruit_day, 4) + exerany * health,  
               data = train_w2im)
```

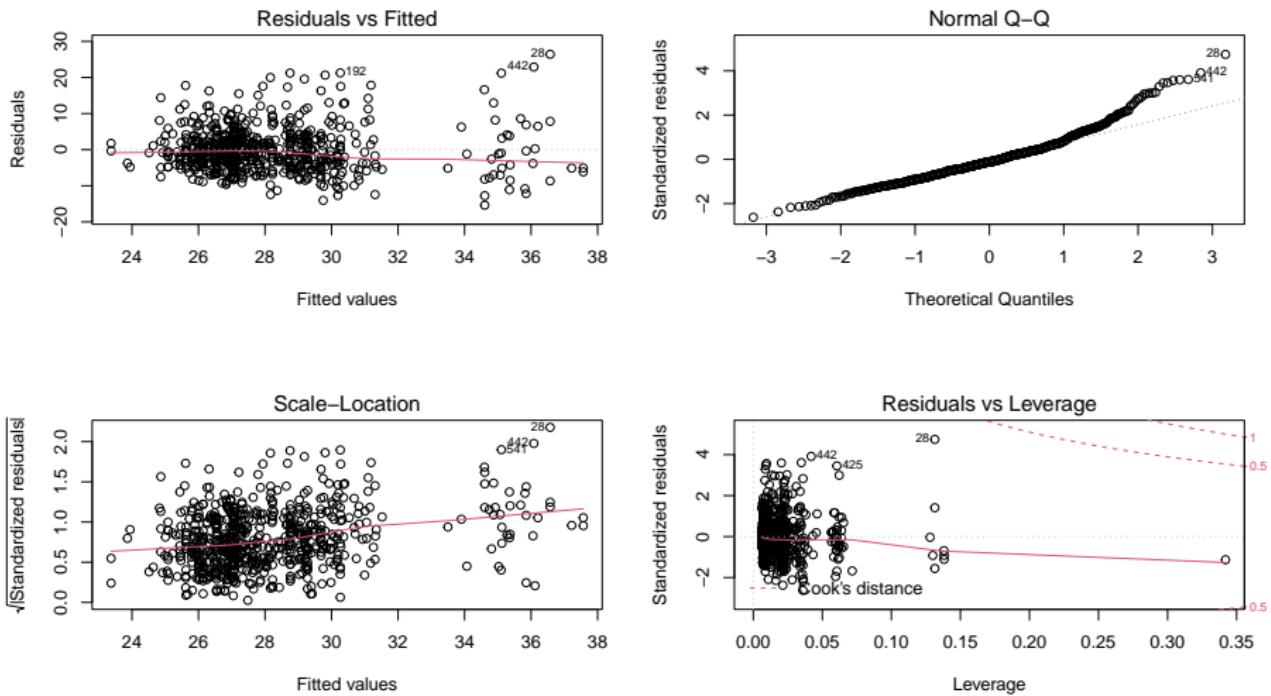
Comparing 4 models including the exerany*health interaction...

mod	fruit	r.sq	adj.r.sq	sigma	df	AIC	BIC
m_1int	not in	0.1264	0.1145	6.007	9	4315.8	4365.4
m_2int	linear	0.1376	0.1245	5.973	10	4309.1	4363.2
m_3int	poly(2)	0.1377	0.1233	5.977	11	4311.1	4369.6
m_4int	rcs(4)	0.1379	0.1222	5.981	12	4312.9	4376.0

Tidied summary of m_4int coefficients

term	est	se	t	p	lo90	hi90
(Intercept)	28.57	1.54	18.50	0.000	26.02	31.11
rcs(fruit_day, 4)fruit_day	-1.22	1.27	-0.96	0.337	-3.30	0.87
rcs(fruit_day, 4)fruit_day'	2.33	5.82	0.40	0.689	-7.26	11.92
rcs(fruit_day, 4)fruit_day''	-6.00	16.58	-0.36	0.717	-33.32	21.31
exerany	-1.34	1.55	-0.87	0.387	-3.90	1.21
healthVG	-0.64	1.63	-0.39	0.696	-3.32	2.04
healthG	2.78	1.61	1.72	0.086	0.12	5.44
healthF	7.64	1.77	4.30	0.000	4.71	10.56
healthP	9.09	2.56	3.54	0.000	4.86	13.31
exerany:healthVG	1.56	1.80	0.87	0.385	-1.40	4.52
exerany:healthG	0.36	1.80	0.20	0.841	-2.60	3.32
exerany:healthF	-6.47	2.07	-3.13	0.002	-9.87	-3.06
exerany:healthP	-6.55	3.02	-2.17	0.031	-11.53	-1.57

m_4int Residual Plots



How do models m_4 and m_4int do in testing?

model	rsquare	rmse	mae
m_1	0.0712	5.736	4.432
m_2	0.0647	5.778	4.480
m_3	0.0651	5.776	4.480
m_4	0.0682	5.771	4.474
m_1int	0.0395	6.033	4.628
m_2int	0.0361	6.091	4.710
m_3int	0.0354	6.098	4.711
m_4int	0.0393	6.099	4.707

I'll note that there's a fair amount of very repetitive code in the R Markdown file to create that table.

- What are our conclusions?

Next Week

- What if we want to build models for a binary outcome, rather than a quantitative one?
- Lab 1 due Monday at 9 PM.
- Tuesday's class will be a Zoom session, Thursday in person.

432 Class 05 Slides

thomaselove.github.io/432

2022-01-25

Moving Forward

- Predicting a Binary outcome
 - using a linear probability model
 - using logistic regression and `glm`
- Creating the `smart3` and `smart3_sh` data
 - A “shadow” to track what is imputed

Setup

```
library(conflicted)                      # a new idea
library(here); library(magrittr)
library(janitor); library(knitr)
library(patchwork); library(broom)
library(equatiomatic)
library(simputation); library(naniar)
library(faraway)                         # for orings data
library(rms)
library(tidyverse)

theme_set(theme_bw())
conflict_prefer("summarize", "dplyr") # choose over Hmisc
conflict_prefer("filter", "dplyr") # choose over stats
options(dplyr.summarise.inform = FALSE)
```

A First Example: Space Shuttle O-Rings

Challenger Space Shuttle Data

The US space shuttle Challenger exploded on 1986-01-28. An investigation ensued into the reliability of the shuttle's propulsion system. The explosion was eventually traced to the failure of one of the three field joints on one of the two solid booster rockets. Each of these six field joints includes two O-rings which can fail.

The discussion among engineers and managers raised concern that the probability of failure of the O-rings depended on the temperature at launch, which was forecast to be 31 degrees F. There are strong engineering reasons based on the composition of O-rings to support the judgment that failure probability may rise monotonically as temperature drops.

We have data on 23 space shuttle flights that preceded *Challenger* on primary o-ring erosion and/or blowby and on the temperature in degrees Fahrenheit. No previous liftoff temperature was under 53 degrees F.

The “O-rings” data

```
orings1 <- faraway::orings %>%
  tibble() %>%
  mutate(burst = case_when( damage > 0 ~ 1,
                           TRUE ~ 0))

orings1 %>% summary()
```

	temp	damage	burst
Min.	:53.00	:0.0000	:0.0000
1st Qu.	:67.00	:0.0000	:0.0000
Median	:70.00	:0.0000	:0.0000
Mean	:69.57	:0.4783	:0.3043
3rd Qu.	:75.00	:1.0000	:1.0000
Max.	:81.00	:5.0000	:1.0000

- damage = number of damage incidents out of 6 possible
- we set burst = 1 if damage > 0

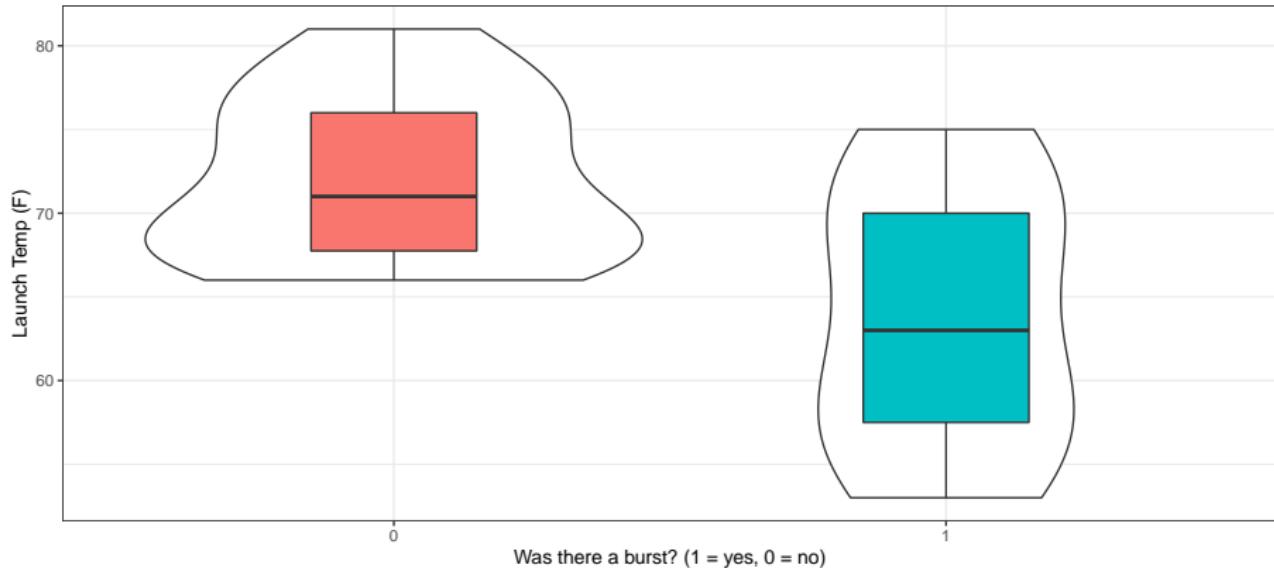
Code to plot burst and temp in our usual way...

```
ggplot(orings1, aes(x = factor(burst), y = temp)) +  
  geom_violin() +  
  geom_boxplot(aes(fill = factor(burst)), width = 0.3) +  
  guides(fill = "none") +  
  labs(title = "Are bursts more common at low temperatures?",  
       subtitle = "23 prior space shuttle launches",  
       x = "Was there a burst? (1 = yes, 0 = no)",  
       y = "Launch Temp (F)")
```

Plotted Association of burst and temp

Are bursts more common at low temperatures?

23 prior space shuttle launches



What if we want to predict Prob(burst) using temp?

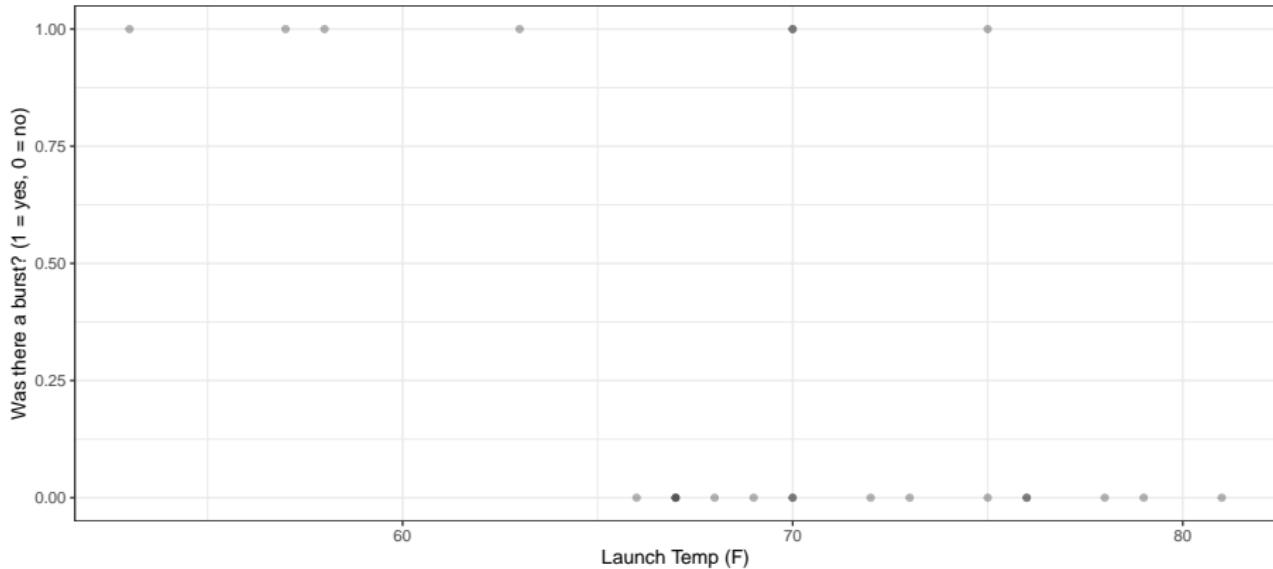
We want to treat the binary variable burst as the outcome, and temp as the predictor...

```
ggplot(orings1, aes(x = temp, y = burst)) +  
  geom_point(alpha = 0.3) +  
  labs(title = "Are bursts more common at low temperatures?",  
       subtitle = "23 prior space shuttle launches",  
       y = "Was there a burst? (1 = yes, 0 = no)",  
       x = "Launch Temp (F)")
```

Plot of Prob(burst) by temperature at launch

Are bursts more common at low temperatures

23 prior space shuttle launches



Fit a linear model to predict Prob(burst)?

```
mod1 <- lm(burst ~ temp, data = orings1)  
  
tidy(mod1, conf.int = T) %>% kable(digits = 3)
```

term	estimate	std.error	statistic	p.value	conf.low	conf.high
(Intercept)	2.905	0.842	3.450	0.002	1.154	4.656
temp	-0.037	0.012	-3.103	0.005	-0.062	-0.012

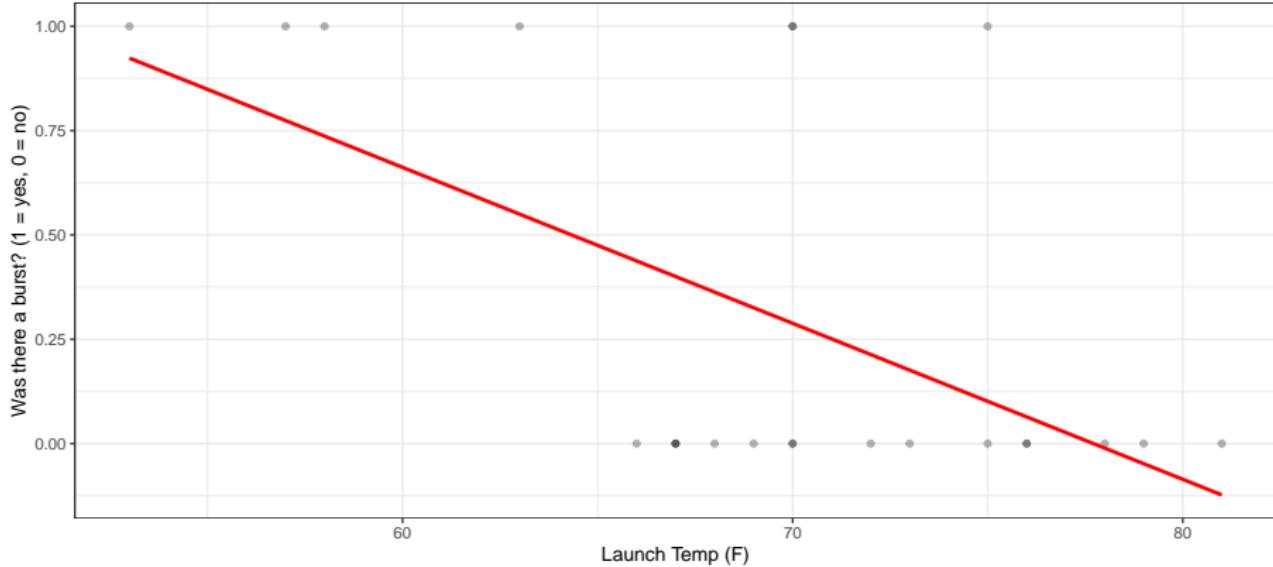
- This is a **linear probability model**.

```
extract_eq(mod1, use_coefs = TRUE, coef_digits = 3)
```

$$\widehat{\text{burst}} = 2.905 - 0.037(\text{temp}) \quad (1)$$

Add linear probability model to our plot?

Bursts more common at lower temperatures
23 prior space shuttle launches



- It would help if we could see the individual launches...

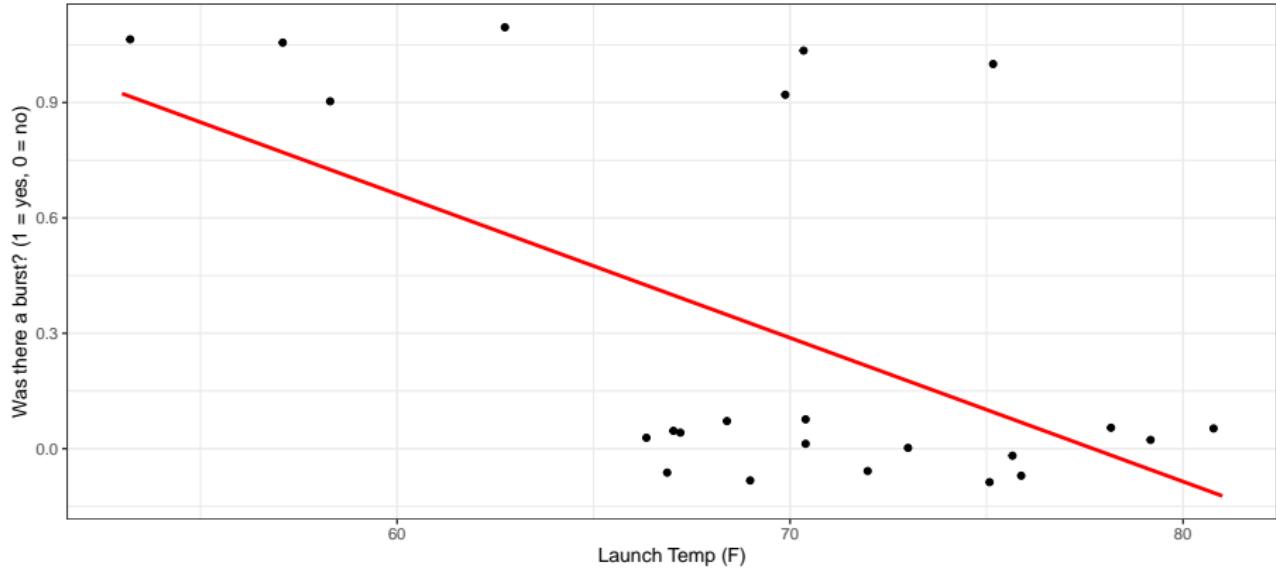
Add vertical jitter and our mod1 model?

```
ggplot(orings1, aes(x = temp, y = burst)) +  
  geom_jitter(height = 0.1) +  
  geom_smooth(method = "lm", se = F, col = "red",  
              formula = y ~ x) +  
  labs(title = "Bursts more common at lower temperatures",  
       subtitle = "23 prior space shuttle launches",  
       y = "Was there a burst? (1 = yes, 0 = no)",  
       x = "Launch Temp (F)")
```

Resulting plot with points jittered and linear model

Bursts more common at lower temperatures

23 prior space shuttle launches



- What's wrong with this picture?

Making Predictions with mod1

```
tidy(mod1, conf.int = T) %>%  
  kable(digits = c(0,5,3,3,3,3,3))
```

term	estimate	std.error	statistic	p.value	conf.low	conf.high
(Intercept)	2.90476	0.842	3.450	0.002	1.154	4.656
temp	-0.03738	0.012	-3.103	0.005	-0.062	-0.012

- What does mod1 predict for the probability of a burst if the temperature at launch is 70 degrees F?

$$Prob(\text{burst}) = 2.90476 - 0.03738(70) = 0.288$$

- What if the temperature was actually 60 degrees F?

Making Several Predictions with mod1

Let's use our linear probability model `mod1` to predict the probability of a burst at some other temperatures...

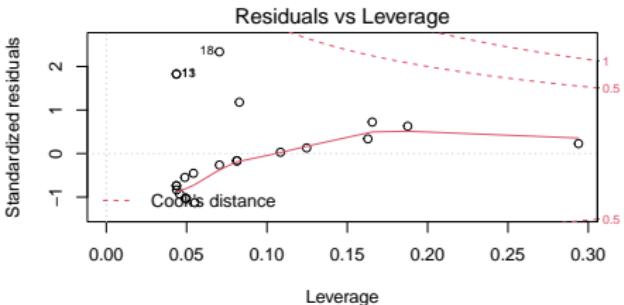
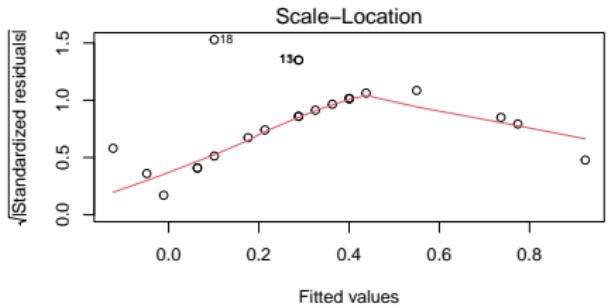
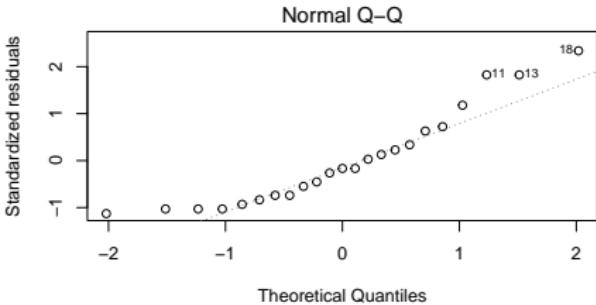
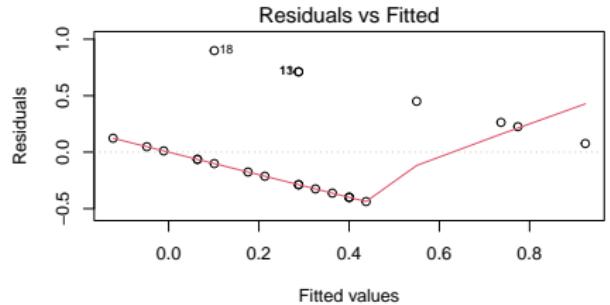
```
newtemps <- tibble(temp = c(80, 70, 60, 50, 31))
```

```
augment(mod1, newdata = newtemps)
```

```
# A tibble: 5 x 2
  temp .fitted
  <dbl>    <dbl>
1     80 -0.0857
2     70  0.288 
3     60  0.662 
4     50  1.04  
5     31  1.75
```

- Uh, oh.

Residual Plots for mod1?



- Uh, oh.

Models to predict a Binary Outcome

Our outcome takes on two values (zero or one) and we then model the probability of a “one” response given a linear function of predictors.

Idea 1: Use a *linear probability model*

- Main problem: predicted probabilities that are less than 0 and/or greater than 1
- Also, how can we assume Normally distributed residuals when outcomes are 1 or 0?

Idea 2: Build a *non-linear* regression approach

- Most common approach: logistic regression, part of the class of *generalized* linear models

The Logit Link and Logistic Function

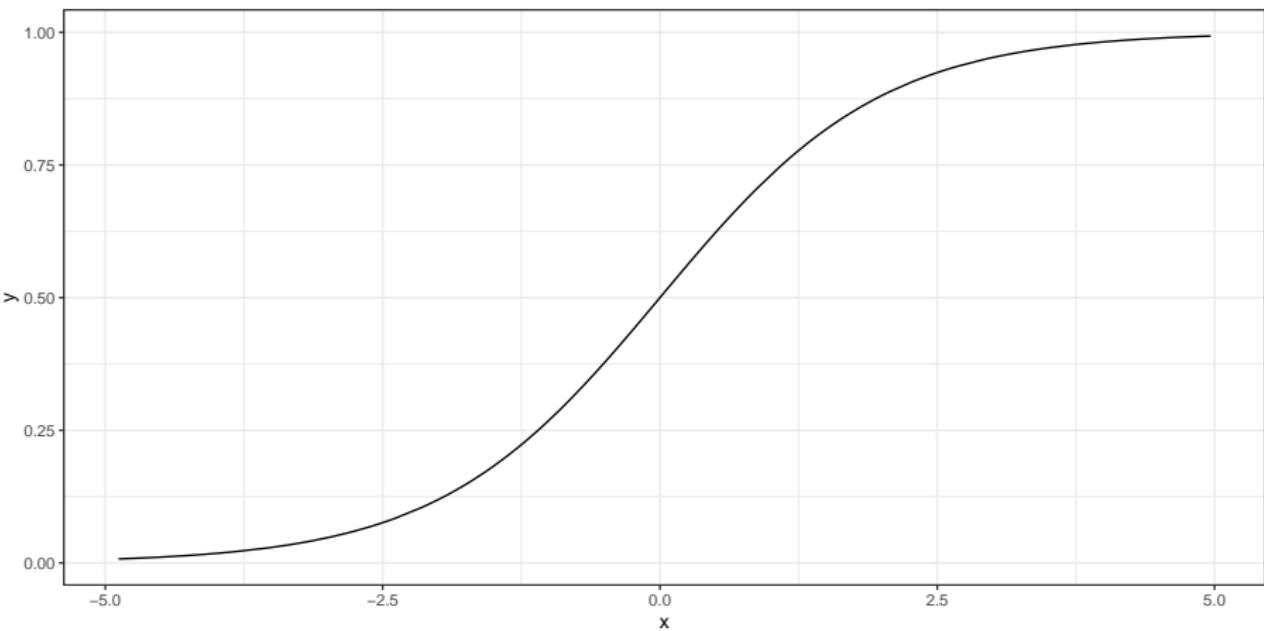
The function we use in logistic regression is called the **logit link**.

$$\text{logit}(\pi) = \log\left(\frac{\pi}{1-\pi}\right) = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \dots + \beta_k X_k$$

The inverse of the logit function is called the **logistic function**. If $\text{logit}(\pi) = \eta$, then $\pi = \frac{\exp(\eta)}{1+\exp(\eta)}$.

- The logistic function $\frac{e^x}{1+e^x}$ takes any value x in the real numbers and returns a value between 0 and 1.

The Logistic Function $y = \frac{e^x}{1+e^x}$



The logit or log odds

We usually focus on the **logit** in statistical work, which is the inverse of the logistic function.

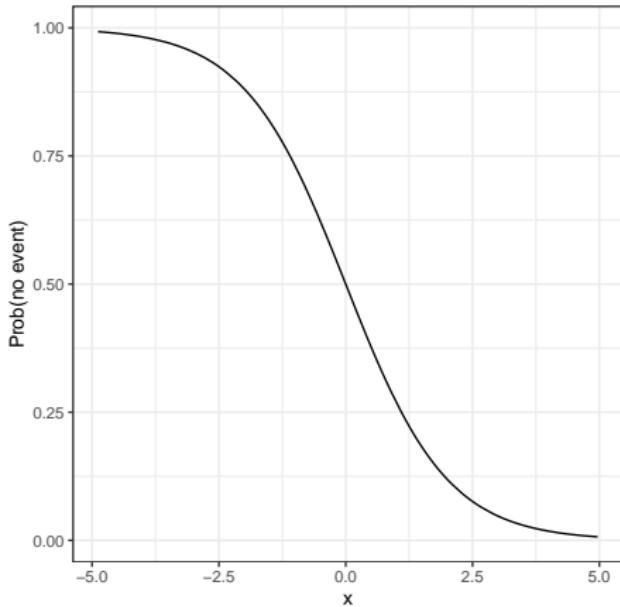
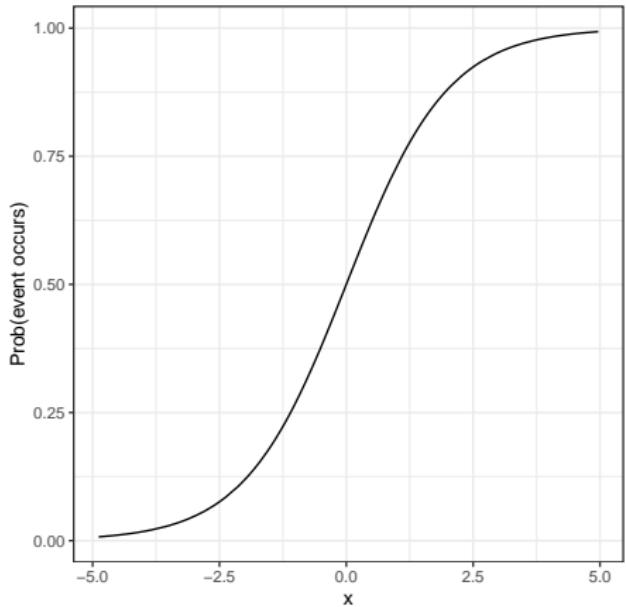
- If we have a probability $\pi < 0.5$, then $\text{logit}(\pi) < 0$.
- If our probability $\pi > 0.5$, then $\text{logit}(\pi) > 0$.
- Finally, if $\pi = 0.5$, then $\text{logit}(\pi) = 0$.

Why is this helpful?

- $\text{log}(\text{odds}(Y = 1))$ or $\text{logit}(Y = 1)$ covers all real numbers.
- $\text{Prob}(Y = 1)$ is restricted to $[0, 1]$.

Predicting $\text{Pr}(\text{event})$ or $\text{Pr}(\text{no event})$

- Can we flip the story?



Returning to the prediction of Prob(burst)

We'll use the `glm` function in R, specifying a logistic regression model.

- Instead of predicting $Pr(burst)$, we're predicting $\log(odd(burst))$ or $logit(burst)$.

```
mod2 <- glm(burst ~ temp, data = orings1,  
            family = binomial(link = "logit"))  
  
tidy(mod2, conf.int = TRUE) %>%  
  select(term, estimate, std.error, conf.low, conf.high) %>%  
  kable(digits = c(0,4,3,3,3))
```

term	estimate	std.error	conf.low	conf.high
(Intercept)	15.0429	7.379	3.331	34.342
temp	-0.2322	0.108	-0.515	-0.061

Our model mod2

```
extract_eq(mod2, use_coefs = TRUE, coef_digits = 4)
```

$$\log \left[\frac{\widehat{P(\text{burst} = 1)}}{1 - \widehat{P(\text{burst} = 1)}} \right] = 15.0429 - 0.2322(\text{temp}) \quad (2)$$

$$\textit{logit}(\text{burst}) = \log(\textit{odds}(\text{burst})) = 15.0429 - 0.2322\text{temp}$$

- For a temperature of 70 F at launch, what is the prediction?

Let's look at the results

- For a temperature of 70 F at launch, what is the prediction?

$$\log(\text{odds}(\text{burst})) = 15.0429 - 0.2322 (70) = -1.211$$

- Exponentiate to get the odds, on our way to estimating the probability.

$$\text{odds}(\text{burst}) = \exp(-1.211) = 0.2979$$

- so, we can estimate the probability by

$$Pr(\text{burst}) = \frac{0.2979}{(0.2979 + 1)} = 0.230.$$

Prediction from mod2 for temp = 60

What is the predicted probability of a burst if the temperature is 60 degrees?

- $\log(\text{odds}(\text{burst})) = 15.0429 - 0.2322(60) = 1.1109$
- $\text{odds}(\text{burst}) = \exp(1.1109) = 3.0371$
- $\Pr(\text{burst}) = 3.0371 / (3.0371 + 1) = 0.752$

Will augment do this, as well?

```
temp <- tibble(temp = c(60,70))

augment(mod2, newdata = temp, type.predict = "link")

# A tibble: 2 x 2
  temp .fitted
  <dbl>    <dbl>
1     60     1.11
2     70    -1.21

augment(mod2, newdata = temp, type.predict = "response")

# A tibble: 2 x 2
  temp .fitted
  <dbl>    <dbl>
1     60     0.753
2     70     0.230
```

Plotting the Logistic Regression Model

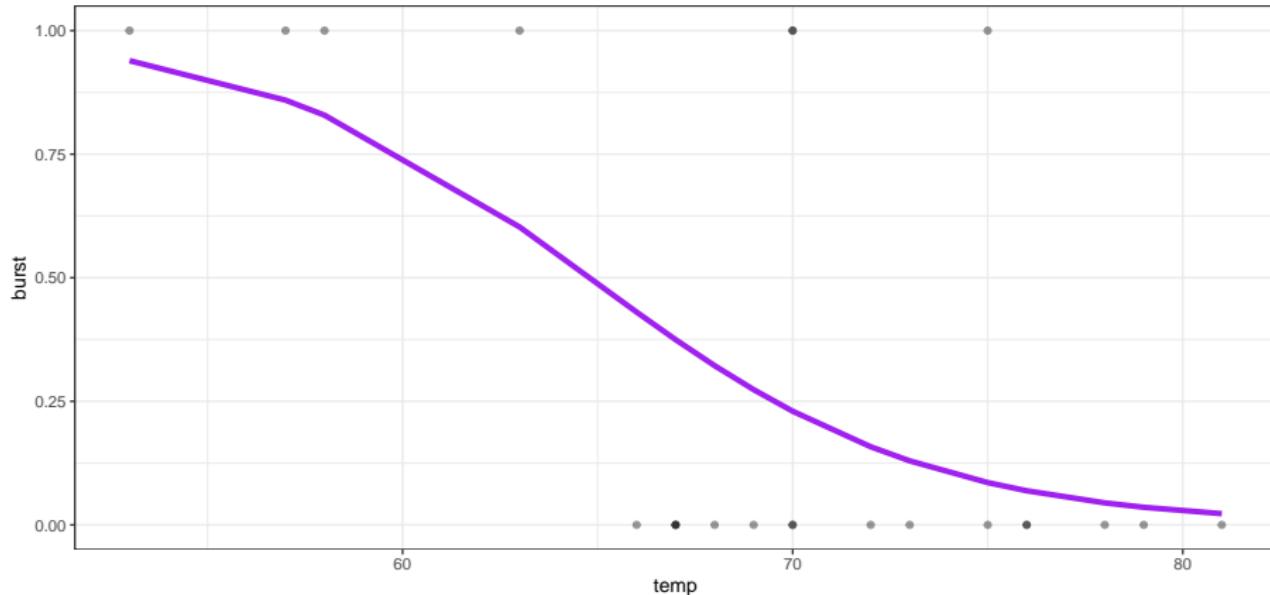
Use the augment function to get the fitted probabilities into the original data, then plot.

```
mod2_aug <- augment(mod2, type.predict = "response")  
  
ggplot(mod2_aug, aes(x = temp, y = burst)) +  
  geom_point(alpha = 0.4) +  
  geom_line(aes(x = temp, y = .fitted),  
            col = "purple", size = 1.5) +  
  labs(title = "Fitted Logistic mod2 for Pr(burst)")
```

- Results on next slide

Plotting Model m2

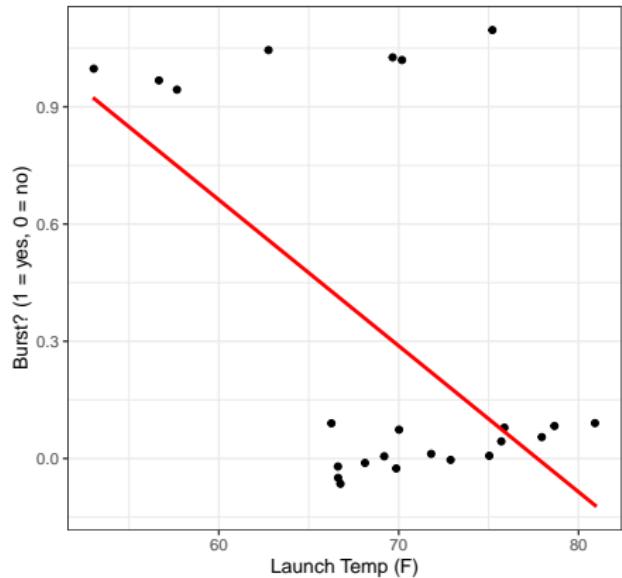
Fitted Logistic mod2 for Pr(burst)



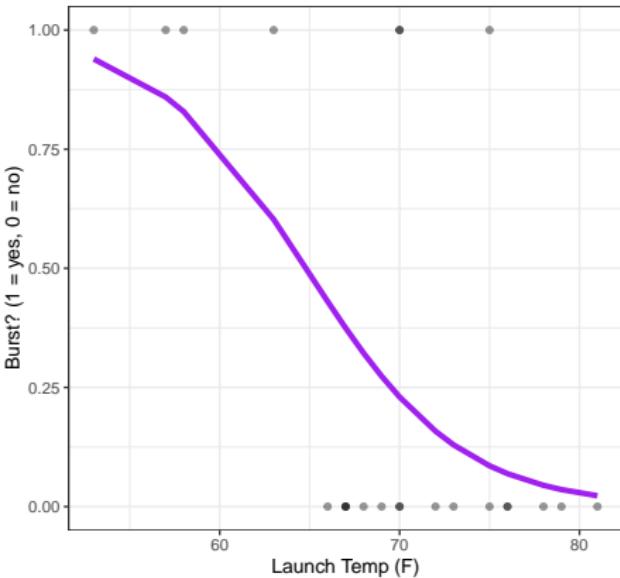
Note that we're just connecting the predictions made for observed temp values with `geom_line`, so the appearance of the function isn't as smooth as the actual logistic regression model.

Comparing the fits of mod1 and mod2...

Linear Probability mod1



Logistic Regression mod2



Could we try exponentiating the mod2 coefficients?

How can we interpret the coefficients of the model?

$$\text{logit}(burst) = \log(\text{odds}(burst)) = 15.043 - 0.232\text{temp}$$

Exponentiating the coefficients is helpful...

`exp(-0.232)`

[1] 0.7929461

Suppose Launch A's temperature was one degree higher than Launch B's.

- The **odds** of Launch A having a burst are 0.793 times as large as they are for Launch B.
- Odds Ratio estimate comparing two launches whose `temp` differs by 1 degree is 0.793

Exponentiated and tidied slope of temp (mod2)

```
tidy(mod2, exponentiate = TRUE, conf.int = TRUE) %>%
  filter(term == "temp") %>%
  select(term, estimate, std.error, conf.low, conf.high) %>%
  kable(digits = 3)
```

term	estimate	std.error	conf.low	conf.high
temp	0.793	0.108	0.597	0.941

- What would it mean if the Odds Ratio for temp was 1?
- How about an odds ratio that was greater than 1?

Regression on a Binary Outcome

Linear Probability Model (a linear model)

```
lm(event ~ predictor1 + predictor2 + ..., data = tiblename)
```

- $\text{Pr}(\text{event})$ is linear in the predictors

Logistic Regression Model (generalized linear model)

```
glm(event ~ pred1 + pred2 + ..., data = tiblename,  
     family = binomial(link = "logit"))
```

- Logistic Regression forces a prediction in $(0, 1)$
- $\log(\text{odds}(\text{event}))$ is linear in the predictors

The logistic regression model

$$\text{logit}(\text{event}) = \log \left(\frac{\Pr(\text{event})}{1 - \Pr(\text{event})} \right) = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \dots + \beta_k X_k$$

$$\text{odds}(\text{event}) = \frac{\Pr(\text{event})}{1 - \Pr(\text{event})}$$

$$\Pr(\text{event}) = \frac{\text{odds}(\text{event})}{\text{odds}(\text{event}) + 1}$$

$$\Pr(\text{event}) = \frac{\exp(\text{logit}(\text{event}))}{1 + \exp(\text{logit}(\text{event}))}$$

Building a smart3 tibble

BRFSS and SMART (Creating smart3)

```
smart3 <- read_csv(here("data/smart_ohio.csv")) %>%
  mutate(SEQNO = as.character(SEQNO)) %>%
  select(SEQNO, mmsa, mmsa_wt, landline,
         age_imp, healthplan, dm_status,
         fruit_day, drinks_wk, activity,
         smoker, physhealth, bmi, genhealth)
```

smart3 Variables, by Type

Variable	Type	Description
landline	Binary (1/0)	survey conducted by landline? (vs. cell)
healthplan	Binary (1/0)	subject has health insurance?
age_imp	Quantitative	age (imputed from groups - see Notes)
fruit_day	Quantitative	mean servings of fruit / day
drinks_wk	Quantitative	mean alcoholic drinks / week
bmi	Quantitative	body-mass index (in kg/m ²)
physhealth	Count (0-30)	of last 30 days, # in poor physical health
dm_status	Categorical	diabetes status (4 levels, we'll collapse to 2)
activity	Categorical	physical activity level (4 levels, we'll re-level)
smoker	Categorical	smoking status (4 levels, we'll collapse to 3)
genhealth	Categorical	self-reported overall health (5 levels)

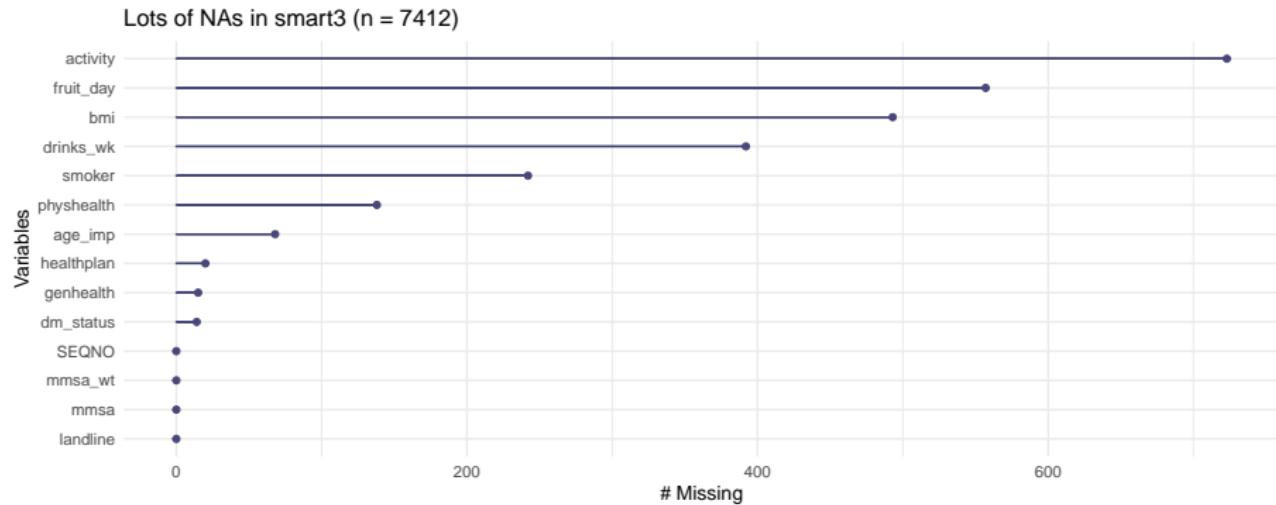
Collapsing Two Factors, Re-leveling another

```
smart3 <- smart3 %>% type.convert() %>%
  mutate(SEQNO = as.character(SEQNO)) %>%
  mutate(dm_status =
    fct_collapse(factor(dm_status),
                 Yes = "Diabetes",
                 No = c("No-Diabetes",
                        "Pre-Diabetes",
                        "Pregnancy-Induced")))) %>%
  mutate(smoker =
    fct_collapse(factor(smoker),
                 Current = c("Current_not_daily",
                            "Current_daily")))) %>%
  mutate(activity =
    fct_relevel(factor(activity),
                "Highly_Active", "Active",
                "Insufficiently_Active",
                "Inactive"))
```

Visualizing Missingness in Variables

```
gg_miss_var(smart3) +  
  labs(title = "Lots of NAs in smart3 (n = 7412)")
```

Warning: It is deprecated to specify `guide = FALSE` to remove a guide. Please use `guide = "none"` instead.



Creating a “Shadow” to track what is imputed

```
smart3_sh <- smart3 %>% bind_shadow()
```

smart3_sh creates new variables, ending in _NA

```
names(smart3_sh)
```

```
[1] "SEQNO"           "mmsa"            "mmsa_wt"  
[4] "landline"        "age_imp"          "healthplan"  
[7] "dm_status"       "fruit_day"        "drinks_wk"  
[10] "activity"        "smoker"           "physhealth"  
[13] "bmi"              "genhealth"        "SEQNO_NA"  
[16] "mmsa_NA"         "mmsa_wt_NA"      "landline_NA"  
[19] "age_imp_NA"      "healthplan_NA"   "dm_status_NA"  
[22] "fruit_day_NA"    "drinks_wk_NA"    "activity_NA"  
[25] "smoker_NA"       "physhealth_NA"  "bmi_NA"  
[28] "genhealth_NA"
```

What are the new variables tracking?

```
smart3_sh %>% count(smoker, smoker_NA)
```

```
# A tibble: 4 x 3
  smoker   smoker_NA     n
  <fct>    <fct>     <int>
1 Current  !NA        1290
2 Former   !NA        1999
3 Never    !NA        3881
4 <NA>     NA         242
```

The fct_explicit_na warning: A pain point

My general preference is to not use `fct_explicit_na`, and if I see a warning about that, I typically suppress it from printing.

“Simple” Imputation Strategy

```
set.seed(2022432)
smart3_sh <- smart3_sh %>%
  data.frame() %>%
  impute_rhd(dm_status + smoker ~ 1) %>%
  impute_rhd(healthplan + activity ~ 1) %>%
  impute_rlm(age_imp + fruit_day + drinks_wk + bmi ~
    mmsa + landline + healthplan) %>%
  impute_knn(phshealth ~ bmi) %>%
  impute_cart(genhealth ~ activity + phshealth +
    mmsa + healthplan) %>%
tibble()
```

Check to see that imputation worked...

Before imputation, what fraction of our cases are complete?

```
pct_complete_case(smart3)
```

```
[1] 81.08473
```

After imputation, do any of our cases have missing values?

```
pct_miss_case(smart3_sh)
```

```
[1] 0
```

Saving the smart3 and smart3_sh tibbles to .Rds

```
saveRDS(smart3, "data/smart3.Rds")
```

```
saveRDS(smart3_sh, "data/smart3_sh.Rds")
```

**Using diabetes status (yes/no) to predict
whether $\text{BMI} > 30$**

Create binary outcome variable

```
smart3_sh <- readRDS("data/smart3_sh.Rds") %>%
  mutate(bmigt30 = as.numeric(bmi > 30),
        dm_status = fct_relevel(dm_status, "No"))

smart3_sh %>%
  group_by(bmigt30) %>%
  summarize(n = n(), mean(bmi), min(bmi), max(bmi)) %>%
  kable(digits = 2)
```

bmigt30	n	mean(bmi)	min(bmi)	max(bmi)
0	5074	25.26	13.30	30.00
1	2338	35.85	30.01	75.52

Predicting BMI > 30 using diabetes status (a factor)

```
mod_DM <- smart3_sh %$%
  glm(bmigt30 ~ dm_status,
      family = binomial(link = logit))

tidy(mod_DM) %>% select(term, estimate) %>%
  kable(digits = 3)
```

term	estimate
(Intercept)	-0.949
dm_statusYes	1.048

$$\text{Equation: } \text{logit}(\text{BMI} > 30) = -0.949 + 1.048 \text{ (dm_statusYes)}$$

How can we interpret this result?

Interpreting the mod_DM Equation

$$\text{logit}(\text{BMI} > 30) = -0.949 + 1.048 (\text{dm_statusYes})$$

- Harry has diabetes.
 - His predicted $\text{logit}(\text{BMI} > 30)$ is $-0.949 + 1.048 (1) = 0.099$
- Sally does not have diabetes.
 - Her predicted $\text{logit}(\text{BMI} > 30)$ is $-0.949 + 1.048 (0) = -0.949$

Now, $\text{logit}(\text{BMI} > 30) = \log(\text{odds}(\text{BMI} > 30))$, so exponentiate to get the odds...

- Harry has predicted $\text{odds}(\text{BMI} > 30) = \exp(0.099) = 1.104$
- Sally has predicted $\text{odds}(\text{BMI} > 30) = \exp(-0.949) = 0.387$

Can we convert these odds into something more intuitive?

Converting Odds to Probabilities

- Harry has predicted odds($BMI > 30$) = $\exp(0.099) = 1.104$
- Sally has predicted odds($BMI > 30$) = $\exp(-0.949) = 0.387$

$$odds(BMI > 30) = \frac{Pr(BMI > 30)}{1 - Pr(BMI > 30)}$$

and

$$Pr(BMI > 30) = \frac{odds(BMI > 30)}{odds(BMI > 30) + 1}$$

- So Harry's predicted $Pr(BMI > 30) = 1.104 / 2.104 = 0.52$
- Sally's predicted $Pr(BMI > 30) = 0.387 / 1.387 = 0.28$
- odds range from 0 to ∞ , and $\log(\text{odds})$ range from $-\infty$ to ∞ .
- odds > 1 if probability > 0.5 . If odds = 1, then probability = 0.5.

What about the odds ratio?

$\text{logit}(\text{BMI} > 30) = -0.949 + 1.048 (\text{dm_statusYes})$

- Harry, with diabetes, has $\text{odds}(\text{BMI} > 30) = 1.104$
- Sally, without diabetes, has $\text{odds}(\text{BMI} > 30) = 0.387$

Odds Ratio for $\text{BMI} > 30$ associated with having diabetes (vs. not) =

$$\frac{1.104}{0.387} = 2.85$$

- Our model estimates that a subject with diabetes has 2.85 times the odds (285% of the odds) of a subject without diabetes of having $\text{BMI} > 30$.

Can we calculate the odds ratio from the equation's coefficients?

- Yes, $\exp(1.048) = 2.85$.

Tidy with exponentiation

```
tidy(mod_DM, exponentiate = TRUE,  
      conf.int = TRUE, conf.level = 0.9) %>%  
  select(term, estimate, conf.low, conf.high) %>%  
  kable(digits = 3)
```

term	estimate	conf.low	conf.high
(Intercept)	0.387	0.369	0.405
dm_statusYes	2.851	2.556	3.181

- The odds ratio for BMI > 30 among subjects with diabetes as compared to those without diabetes is 2.851
- The odds of BMI > 30 are 285.1% as large (2.851 times as large) for subjects with diabetes as they are for subjects without diabetes, according to this model.
- A 90% uncertainty interval for the odds ratio estimate includes (2.556, 3.181).

Interpreting these summaries

Connecting the Odds Ratio and Log Odds Ratio to probability statements...

- If the probabilities were the same (for diabetes and non-diabetes subjects) of having $BMI > 30$, then the odds would also be the same, and so the odds ratio would be 1.
- If the probabilities of $BMI > 30$ were the same and thus the odds were the same, then the log odds ratio would be $\log(1) = 0$.

$\text{logit}(BMI > 30) = -0.949 + 1.048 \ (\text{dm_statusYes})$

- ① If the log odds of a coefficient (like $\text{diabetes} = \text{Yes}$) are negative, then what does that imply?
- ② What if we flipped the order of the levels for diabetes so our model was about $\text{diabetes} = \text{No}$?

Flipping the model changes slope and intercept!

```
mod_DM_no <- smart3_sh %$%
  glm(bmigt30 ~ (dm_status == "No"),
      family = binomial(link = logit))

tidy(mod_DM_no) %>% select(term, estimate) %>%
  kable(digits = 3)
```

term	estimate
(Intercept)	0.098
dm_status == "No"TRUE	-1.048

Old: $\text{logit}(\text{BMI} > 30) = -0.949 + 1.048 (\text{dm_statusYes})$

New: $\text{logit}(\text{BMI} > 30) = 0.098 - 1.048 (\text{dm_status} = \text{No})$

Predictions from the two models?

DMYes: $\text{logit}(\text{BMI} > 30) = -0.949 + 1.048$ (`dm_status = Yes`)

DMNo: $\text{logit}(\text{BMI} > 30) = 0.098 - 1.048$ (`dm_status = No`)

Harry lives with diabetes. Sally does not.

Using the DMYes model:

- $\text{logit}(\text{Harry's BMI} > 30) = -0.949 + 1.048 = 0.098$
- $\text{logit}(\text{Sally's BMI} > 30) = -0.949$

Using the DMNo model:

- $\text{logit}(\text{Harry's BMI} > 30) = 0.098$
- $\text{logit}(\text{Sally's BMI} > 30) = 0.098 - 1.048 = -0.949$

Comparison to the 2x2 Table Results?

```
smart3_sh %>% tabyl(bmigt30, dm_status)
```

bmigt30	No	Yes
0	4551	523
1	1761	577

That's not quite the 2x2 table we want.

We want to switch the order of both variables

```
temp1 <- smart3_sh %>%
  select(bmigt30, dm_status, SEQNO)

temp1 <- temp1 %>%
  mutate(bmigt30 = fct_relevel(factor(bmigt30), "1"),
        dm_status = fct_relevel(dm_status, "Yes"))

temp1 %>% tabyl(bmigt30, dm_status)
```

	bmigt30	Yes	No
1	577	1761	
0	523	4551	

Resulting 2x2 Table Result

```
temp1 %$% Epi::twoby2(bmigt30, dm_status)
```

2 by 2 table analysis:

Outcome : Yes

Comparing : 1 vs. 0

	Yes	No	P(Yes)	95% conf. interval
1	577	1761	0.2468	0.2297 0.2647
0	523	4551	0.1031	0.0950 0.1117

	95% conf. interval		
Relative Risk:	2.3943	2.1498	2.6666
Sample Odds Ratio:	2.8512	2.5024	3.2486
Conditional MLE Odds Ratio:	2.8506	2.4969	3.2554
Probability difference:	0.1437	0.1246	0.1633

Using smoking status (multi-categorical) to predict diabetes (yes/no)

Can we use smoker to predict dm_status?

```
smart3_sh %>% tabyl(smoker, dm_status) %>%
  adorn_totals() %>%
  adorn_percentages(den = "row") %>%
  adorn_pct_formatting() %>%
  adorn_ns(position = "front")
```

smoker	No	Yes
Current	1170 (87.8%)	163 (12.2%)
Former	1689 (81.9%)	373 (18.1%)
Never	3453 (86.0%)	564 (14.0%)
Total	6312 (85.2%)	1100 (14.8%)

Logistic Regression for dm_status by smoker

```
mod_SM <- glm(dm_status ~ smoker,  
                family = binomial(link = "logit"),  
                data = smart3_sh)  
  
tidy(mod_SM) %>% select(term, estimate) %>% kable(dig = 3)
```

term	estimate
(Intercept)	-1.971
smokerFormer	0.461
smokerNever	0.159

What is being fit, exactly?

```
extract_eq(mod_SM, use_coefs = TRUE, coef_digits = 3,  
          wrap = TRUE, terms_per_line = 2, label = NA)
```

$$\log \left[\frac{\widehat{P(\text{dm_status} = \text{Yes})}}{1 - \widehat{P(\text{dm_status} = \text{Yes})}} \right] = -1.971 + 0.461(\text{smoker}_{\text{Former}}) + 0.159(\text{smoker}_{\text{Never}}) \quad (3)$$

Resulting Predictions from mod_SM

$\text{logit}(\text{dm} = \text{Yes}) = -1.971 + 0.461 \text{ Former} + 0.159 \text{ Never}$

- from logit to odds via exponentiation

Smoking Status	logit(DM = Yes)	odds(DM = Yes)
Current	-1.971	$\exp(-1.971) = 0.139$
Former	$-1.971 + 0.461 = -1.510$	$\exp(-1.510) = 0.221$
Never	$-1.971 + 0.159 = -1.812$	$\exp(-1.812) = 0.163$

- convert from odds to probabilities (do these match our table?)

Smoking Status	odds(DM = Yes)	Pr(DM = Yes)
Current	0.139	.139/1.139 = 0.122
Former	0.221	.221/1.221 = 0.181
Never	0.163	.163/1.163 = 0.140

Next Time

- Binary regression models with multiple predictors
- Assessing the quality of fit for a logistic model

432 Class 06 Slides

thomaselove.github.io/432

2022-01-27

Moving Forward

- Logistic Regression Models and the smart3_sh data

Setup

```
library(here); library(magrittr)
library(janitor); library(knitr)
library(patchwork); library(broom)
library(equatiomatic); library(simputation)
library(naniar)
library(rsample); library(yardstick)
library(tidyverse)

theme_set(theme_bw())
```

smart3 Variables, by Type

Variable	Type	Description
landline	Binary (1/0)	survey conducted by landline? (vs. cell)
healthplan	Binary (1/0)	subject has health insurance?
age_imp	Quantitative	age (imputed from groups - see Notes)
fruit_day	Quantitative	mean servings of fruit / day
drinks_wk	Quantitative	mean alcoholic drinks / week
bmi	Quantitative	body-mass index (in kg/m ²)
physhealth	Count (0-30)	of last 30 days, # in poor physical health
dm_status	Categorical	diabetes status (4 levels, we'll collapse to 2)
activity	Categorical	physical activity level (4 levels, we'll re-level)
smoker	Categorical	smoking status (4 levels, we'll collapse to 3)
genhealth	Categorical	self-reported overall health (5 levels)

The smart3 data (built last time)

```
smart3_sh <- readRDS(here("data", "smart3_sh.Rds"))
```

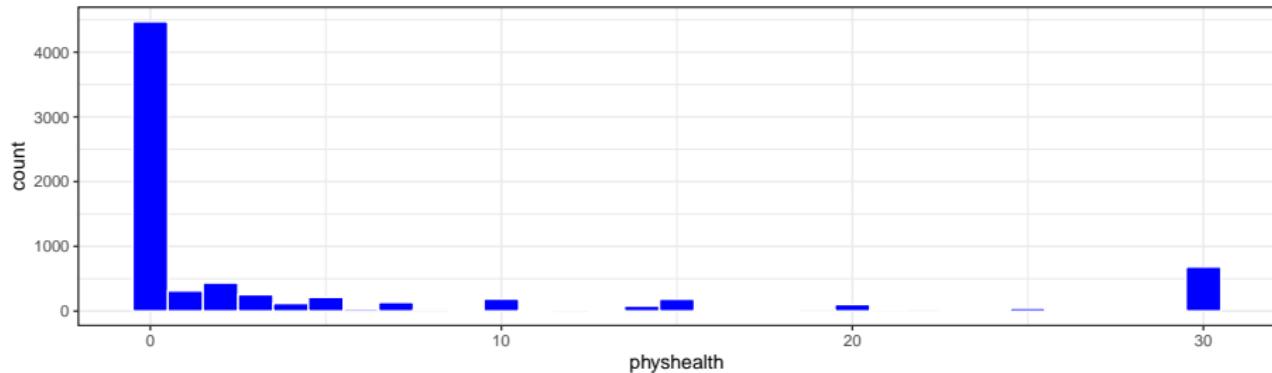
```
str(smart3_sh)
```

```
tibble [7,412 x 28] (S3: tbl_df/tbl/data.frame)
```

```
$ SEQNO      : chr [1:7412] "2017000001" "2017000002" "2017000003" ...
$ mmsa       : chr [1:7412] "Cincinnati" "Cincinnati" "Cincinnati" ...
$ mmsa_wt    : num [1:7412] 670 407 356 203 194 ...
$ landline   : int [1:7412] 1 1 1 1 1 1 1 1 1 1 ...
$ age_imp    : num [1:7412] 36 41 55 61 57 24 65 53 51 42 ...
$ healthplan : chr [1:7412] "1" "1" "1" "1" ...
$ dm_status  : Factor w/ 2 levels "Yes","No": 2 2 2 2 2 2 1 1 1 1 ...
$ fruit_day  : num [1:7412] 1.43 1 3 0.5 0.72 ...
$ drinks_wk  : num [1:7412] 4.67 0 0 0 0.23 1.87 0 0 0.23 0 ...
$ activity   : Factor w/ 4 levels "Highly_Active",...: 2 2 1 1 1 1 1 1 1 1 ...
$ smoker     : Factor w/ 3 levels "Current","Former",...: 3 3 3 3 3 3 3 3 3 3 ...
$ physhealth : int [1:7412] 0 0 2 0 2 0 0 30 2 30 ...
```

Days (in last 30) of poor physical health

```
ggplot(smart3_sh, aes(x = physhealth)) +  
  geom_histogram(binwidth = 1,  
                 fill = "blue", col = "white")
```



```
smart3_sh %$% tabyl(physhealth > 0)
```

physhealth > 0	n	percent
FALSE	4472	0.6033459
TRUE	2940	0.3966541

Create day6 data: predicting Pr(physhealth > 0)?

```
day6 <- smart3_sh %>%  
  mutate(sick = as.numeric(physhealth > 0),  
        id = as.character(  
          as.numeric(SEQNO)-2017000000)) %>%  
  select(id, sick, age = age_imp, dm_status, smoker,  
         bmi, physhealth)
```

```
slice(day6, 17:19) # show rows 17-19
```

```
# A tibble: 3 x 7  
  id      sick    age dm_status smoker      bmi physhealth  
  <chr> <dbl> <dbl> <fct>     <fct>   <dbl>     <int>  
1 17       0     72  No    Former    31.4      0  
2 18       1     82  No    Never    27.6      5  
3 19       0     62  Yes   Current  27.5      0
```

Before fitting models, let's split our sample

```
set.seed(4322022)

day6_split <- initial_split(day6, prop = 0.7,
                             strata = smoker)

d6_train <- training(day6_split)
d6_test <- testing(day6_split)
```

What does strata = smoker do?

Impact of strata = smoker in split

```
d6_train %>% tabyl(smoker)
```

smoker	n	percent
Current	933	0.1798728
Former	1443	0.2781955
Never	2811	0.5419318

```
d6_test %>% tabyl(smoker)
```

smoker	n	percent
Current	400	0.1797753
Former	619	0.2782022
Never	1206	0.5420225

The logistic regression model

$$\text{logit}(\text{event}) = \log \left(\frac{\Pr(\text{event})}{1 - \Pr(\text{event})} \right) = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \dots + \beta_k X_k$$

$$\text{odds}(\text{event}) = \frac{\Pr(\text{event})}{1 - \Pr(\text{event})}$$

$$\Pr(\text{event}) = \frac{\text{odds}(\text{event})}{\text{odds}(\text{event}) + 1}$$

$$\Pr(\text{event}) = \frac{\exp(\text{logit}(\text{event}))}{1 + \exp(\text{logit}(\text{event}))}$$

Model 1: predict sick from smoker and age

Fit the model with and without an interaction term?

```
mod1 <- glm(sick ~ age + smoker,  
            family = binomial(link = "logit"),  
            data = d6_train)
```

```
mod2 <- glm(sick ~ age * smoker,  
            family = binomial(link = "logit"),  
            data = d6_train)
```

- ① Can we use the models to make predictions?
- ② How should we interpret the model coefficients?
- ③ Can we compare the models based on in-sample performance?
- ④ How can we assess predictions using our test sample?

Model 1

```
extract_eq(mod1, wrap = TRUE, terms_per_line = 2,  
          operator_location = "start", use_coefs = TRUE,  
          coef_digits = 3)
```

$$\log \left[\frac{\widehat{P(\text{sick} = 1)}}{1 - \widehat{P(\text{sick} = 1)}} \right] = -0.322 + 0.005(\text{age}) - 0.318(\text{smoker}_{\text{Former}}) - 0.51(\text{smoker}_{\text{Never}}) \quad (1)$$

Likelihood Ratio Tests: Model 1

```
anova(mod1, test = "LRT")
```

Analysis of Deviance Table

Model: binomial, link: logit

Response: sick

Terms added sequentially (first to last)

	Df	Deviance	Resid. Df	Resid. Dev	Pr(>Chi)
NULL			5186	6964.6	
age	1	7.422	5185	6957.2	0.006442 **
smoker	2	45.045	5183	6912.1	1.654e-10 ***

Signif. codes:

0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Model 1

```
tidy(mod1, conf.int = TRUE, conf.level = 0.90) %>%
  select(term, estimate, std.error,
         conf.low, conf.high, p.value) %>%
kable(dig = 3)
```

term	estimate	std.error	conf.low	conf.high	p.value
(Intercept)	-0.322	0.105	-0.494	-0.150	0.002
age	0.005	0.002	0.002	0.007	0.003
smokerFormer	-0.318	0.086	-0.460	-0.176	0.000
smokerNever	-0.510	0.077	-0.636	-0.384	0.000

Model 1 Predictions for subjects A-F

- $\text{logit}(\text{sick}) = -0.322 + 0.005 \text{ age} - 0.318 \text{ Former} - 0.510 \text{ Never}$

ID	age	smoker	logit(sick)	odds(sick)	Pr(sick)
A	33	Current	-0.157	0.8547	0.461
B	33	Former	-0.475	0.6219	0.383
C	33	Never	-0.667	0.5132	0.339
D	55	Current	-0.047	0.9541	0.488
E	55	Former	-0.365	0.6942	0.410
F	55	Never	-0.557	0.5729	0.364

Sample Calculation (for E):

- $\text{logit}(\text{sick}) = -0.322 + 0.005 (55) - 0.318 (1) - 0.510 (0) = -0.365$
- $\text{odds}(\text{sick}) = \exp(-0.365) = 0.6942$
- $\text{Prob}(\text{sick}) = 0.6942 / (1 + 0.6942) = 0.410$

Model 1 (coefficients exponentiated)

```
tidy(mod1, exponentiate = TRUE, conf.int = TRUE,  
     conf.level = 0.90) %>%  
  select(term, estimate,  
         lo90 = conf.low, hi90 = conf.high) %>%  
  kable(dig = 3)
```

term	estimate	lo90	hi90
(Intercept)	0.725	0.610	0.861
age	1.005	1.002	1.007
smokerFormer	0.728	0.631	0.839
smokerNever	0.601	0.529	0.681

- So what can we conclude about, for instance, the effect of Never smoking (as compared to Current smoking)?

Model 1 (coefficients exponentiated)

```
tidy(mod1, exponentiate = TRUE, conf.int = TRUE,  
     conf.level = 0.90) %>%  
  select(term, estimate,  
         lo90 = conf.low, hi90 = conf.high) %>%  
  kable(dig = 3)
```

term	estimate	lo90	hi90
(Intercept)	0.725	0.610	0.861
age	1.005	1.002	1.007
smokerFormer	0.728	0.631	0.839
smokerNever	0.601	0.529	0.681

- So what can we conclude about, for instance, the effect of Never smoking (as compared to Current smoking)?
- Suppose Chloe and Nancy are the same age, where Nancy never smoked and Chloe is a current smoker.

Model 1 (Chloe and Nancy)

term	estimate	lo90	hi90
(Intercept)	0.725	0.610	0.861
age	1.005	1.002	1.007
smokerFormer	0.728	0.631	0.839
smokerNever	0.601	0.529	0.681

- Chloe and Nancy are the same age; Nancy never smoked and Chloe smokes currently. What can we conclude about the relative odds for Nancy of a sick day as compared to Chloe?

Model 1 (Chloe and Nancy)

term	estimate	lo90	hi90
(Intercept)	0.725	0.610	0.861
age	1.005	1.002	1.007
smokerFormer	0.728	0.631	0.839
smokerNever	0.601	0.529	0.681

- Chloe and Nancy are the same age; Nancy never smoked and Chloe smokes currently. What can we conclude about the relative odds for Nancy of a sick day as compared to Chloe?
- Nancy's odds of at least one sick day in the past 30 are 60.1% of Chloe's odds.

Model 1 (Chloe and Nancy)

term	estimate	lo90	hi90
(Intercept)	0.725	0.610	0.861
age	1.005	1.002	1.007
smokerFormer	0.728	0.631	0.839
smokerNever	0.601	0.529	0.681

- Chloe and Nancy are the same age; Nancy never smoked and Chloe smokes currently. What can we conclude about the relative odds for Nancy of a sick day as compared to Chloe?
- Nancy's odds of at least one sick day in the past 30 are 60.1% of Chloe's odds.
- 90% CI for this odds ratio is (0.529, 0.681).

Model 1 (Chloe and Nancy)

term	estimate	lo90	hi90
(Intercept)	0.725	0.610	0.861
age	1.005	1.002	1.007
smokerFormer	0.728	0.631	0.839
smokerNever	0.601	0.529	0.681

- Chloe and Nancy are the same age; Nancy never smoked and Chloe smokes currently. What can we conclude about the relative odds for Nancy of a sick day as compared to Chloe?
- Nancy's odds of at least one sick day in the past 30 are 60.1% of Chloe's odds.
- 90% CI for this odds ratio is (0.529, 0.681).
- Chloe's odds of a sick day are $(1/0.601 = 1.664)$ times those of Nancy.

Does this match the predictions we made?

- Suppose both Chloe and Nancy are 33 years old.
- We saw that Nancy's odds(sick) should be 0.601 times Chloe's odds(sick).

ID	age	smoker	logit(sick)	odds(sick)	Pr(sick)
Chloe	33	Current	-0.157	0.8547	0.461
Nancy	33	Never	-0.667	0.5132	0.339

- and we have $0.5132 / 0.8547 = 0.600$ for the ratio of Nancy's odds to Chloe's odds
- and Chloe's odds are $0.8547 / 0.5132 = 1.665$ times those of Nancy.
- These discrepancies are just due to rounding error in my table.

Model 1 results from glance

- We'll have some additional measures of fit quality, in time.
- Deviance = $-2(\log \text{Likelihood})$

```
glance(mod1) %>%
  select(nobs, df.null, null.deviance,
         deviance, df.residual) %>% kable(dig = 1)
```

nobs	df.null	null.deviance	deviance	df.residual
5187	5186	6964.6	6912.1	5183

```
glance(mod1) %>%
  select(nobs, logLik, AIC, BIC) %>% kable(dig = 1)
```

nobs	logLik	AIC	BIC
5187	-3456.1	6920.1	6946.4

Model 2

```
extract_eq(mod2, wrap = TRUE, terms_per_line = 1,  
          operator_location = "start", use_coefs = TRUE,  
          coef_digits = 3)
```

$$\log \left[\frac{\widehat{P(\text{sick} = 1)}}{1 - \widehat{P(\text{sick} = 1)}} \right] = -0.188 + 0.002(\text{age}) - 0.563(\text{smoker}_{\text{Former}}) - 0.642(\text{smoker}_{\text{Never}}) + 0.004(\text{age} \times \text{smoker}_{\text{Former}}) + 0.003(\text{age} \times \text{smoker}_{\text{Never}}) \quad (2)$$

Likelihood Ratio Tests: Model 2

```
anova(mod2, test = "LRT")
```

Analysis of Deviance Table

Model: binomial, link: logit

Response: sick

Terms added sequentially (first to last)

	Df	Deviance	Resid. Df	Resid. Dev	Pr(>Chi)	
NULL			5186	6964.6		
age	1	7.422	5185	6957.2	0.006442	**
smoker	2	45.045	5183	6912.1	1.654e-10	***
age:smoker	2	0.733	5181	6911.4	0.693211	

Signif. codes:

Model 2

```
tidy(mod2, conf.int = TRUE, conf.level = 0.90) %>%
  select(term, estimate, std.error, conf.low,
         conf.high, p.value) %>% kable(dig = 3)
```

term	estimate	std.error	conf.low	conf.high	p.value
(Intercept)	-0.188	0.214	-0.542	0.164	0.380
age	0.002	0.004	-0.005	0.009	0.608
smokerFormer	-0.563	0.300	-1.057	-0.070	0.060
smokerNever	-0.642	0.245	-1.046	-0.238	0.009
age:smokerFormer	0.004	0.005	-0.004	0.013	0.392
age:smokerNever	0.003	0.004	-0.005	0.010	0.564

- $\text{logit}(\text{sick}) = -0.188 + 0.002 \text{ age} - 0.563 \text{ Former} - 0.642 \text{ Never} + 0.004 \text{ age*Former} + 0.003 \text{ age*Never}$

Model 2 predictions for subjects A-F

- $\text{logit}(\text{sick}) = -0.188 + 0.002 \text{ age} - 0.563 \text{ Former} - 0.642 \text{ Never} + 0.004 \text{ age*Former} + 0.003 \text{ age*Never}$

ID	age	smoker	logit(sick)	odds(sick)	Pr(sick)
A	33	Current	-0.122	0.8851	0.470
B	33	Former	-0.553	0.5752	0.365
C	33	Never	-0.665	0.5143	0.340
D	55	Current	-0.078	0.9250	0.481
E	55	Former	-0.421	0.6564	0.396
F	55	Never	-0.555	0.5741	0.365

- Subject E: $\text{logit}(\text{sick}) = -0.188 + 0.002(55) - 0.563(1) - 0.642(0) + 0.004(55)(1) + 0.003(55)(0) = -0.421$
- $\text{odds}(\text{sick}) = \exp(-0.421) = 0.6564$ so
- $\text{Prob}(\text{sick}) = 0.6564 / (1 + 0.6564) = 0.396$ for subject E.

Model 2 (coefficients exponentiated)

```
tidy(mod2, exponentiate = TRUE, conf.int = TRUE,  
     conf.level = 0.90) %>%  
  select(term, estimate,  
         lo90 = conf.low, hi90 = conf.high) %>%  
  kable(dig = 3)
```

term	estimate	lo90	hi90
(Intercept)	0.828	0.582	1.178
age	1.002	0.995	1.009
smokerFormer	0.570	0.348	0.932
smokerNever	0.526	0.351	0.788
age:smokerFormer	1.004	0.996	1.013
age:smokerNever	1.003	0.995	1.010

Model 2 (Chloe and Nancy)

term	estimate	lo90	hi90
(Intercept)	0.828	0.582	1.178
age	1.002	0.995	1.009
smokerFormer	0.570	0.348	0.932
smokerNever	0.526	0.351	0.788
age:smokerFormer	1.004	0.996	1.013
age:smokerNever	1.003	0.995	1.010

- Chloe and Nancy are the same age; Nancy never smoked and Chloe smokes currently. What can we conclude about the relative odds for Nancy of a sick day as compared to Chloe?

Model 2 (Chloe and Nancy)

term	estimate	lo90	hi90
(Intercept)	0.828	0.582	1.178
age	1.002	0.995	1.009
smokerFormer	0.570	0.348	0.932
smokerNever	0.526	0.351	0.788
age:smokerFormer	1.004	0.996	1.013
age:smokerNever	1.003	0.995	1.010

- Chloe and Nancy are the same age; Nancy never smoked and Chloe smokes currently. What can we conclude about the relative odds for Nancy of a sick day as compared to Chloe?
- We cannot conclude anything **unless** we know what age Chloe and Nancy are, since the effect of smoking depends on age.

Model 2 (Chloe and Nancy)

If Chloe (current smoker) and Nancy (never smoker) are each 33, then ...

ID	age	smoker	logit(sick)	odds(sick)	Pr(sick)
Chloe	33	Current	-0.122	0.8851	0.470
Nancy	33	Never	-0.665	0.5143	0.340

- Chloe's odds of being sick are $0.8851 / 0.5143 = 1.72$ times that of Nancy, **if** they are each 33 years old.
- If Chloe and Nancy are each 55, then from the table below, Chloe's odds are $0.9250 / 0.5741 = 1.61$ times Nancy's odds of being sick.

ID	age	smoker	logit(sick)	odds(sick)	Pr(sick)
D	55	Current	-0.078	0.9250	0.481
F	55	Never	-0.555	0.5741	0.365

Comparing Model 1 to Model 2 with AIC and BIC

```
bind_rows(glance(mod1) %>% select(nobs, AIC, BIC),  
          glance(mod2) %>% select(nobs, AIC, BIC)) %>%  
  mutate(mod = c("m1 (no int.)", "m2 (interaction)")) %>%  
  kable(digits = 1)
```

nobs	AIC	BIC	mod
5187	6920.1	6946.4	m1 (no int.)
5187	6923.4	6962.7	m2 (interaction)

- Which model looks like it performs better in the training sample?

Comparison with Mallows' C_p statistic?

```
anova(mod1, mod2, test = "Cp")
```

Analysis of Deviance Table

Model 1: sick ~ age + smoker

Model 2: sick ~ age * smoker

	Resid. Df	Resid. Dev	Df	Deviance	Cp
1	5183	6912.1		6920.1	
2	5181	6911.4	2	0.73284	6923.4

- Same as what we got from glance for AIC in this case.

Can we compare the models with a Test?

```
anova(mod1, mod2, test = "LRT")
```

Analysis of Deviance Table

Model 1: sick ~ age + smoker

Model 2: sick ~ age * smoker

	Resid. Df	Resid. Dev	Df	Deviance	Pr(>Chi)
1	5183	6912.1			
2	5181	6911.4	2	0.73284	0.6932

Could also consider:

- Rao's efficient score test (test = "Rao")
- Pearson's chi-square test (test = "Chisq")

Let's get predicted probabilities in training sample

```
m1_aug <- augment(mod1, type.predict = "response")
m2_aug <- augment(mod2, type.predict = "response")
```

The predicted probabilities are in the .fitted column.

```
m1_aug %>% select(age, smoker, sick, .fitted) %>% slice(1)
```

```
# A tibble: 1 x 4
  age   smoker   sick   .fitted
  <dbl> <fct>     <dbl>
1     57 Current     1     0.486
```

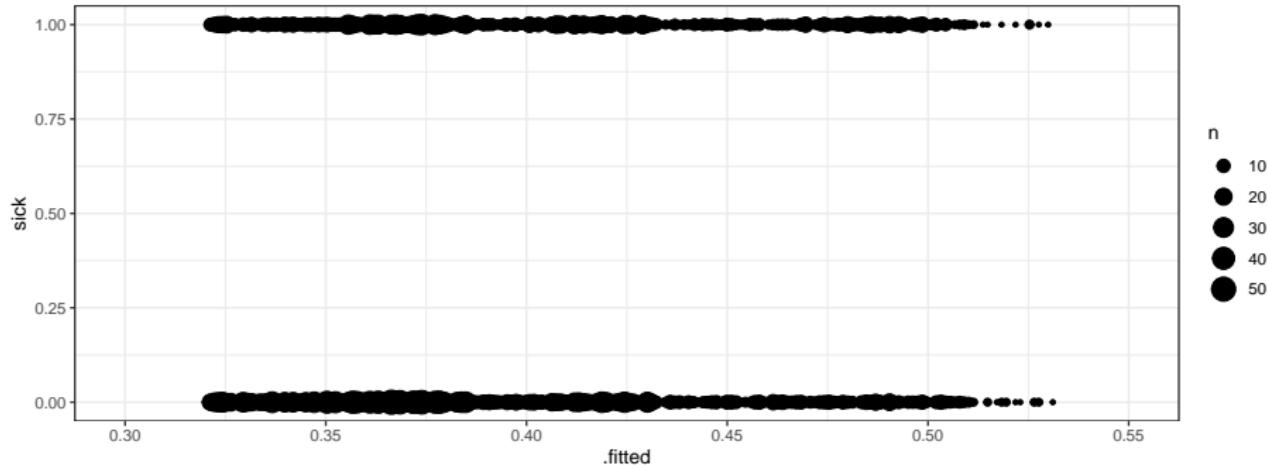
```
m2_aug %>% select(age, smoker, sick, .fitted) %>% slice(1)
```

```
# A tibble: 1 x 4
  age   smoker   sick   .fitted
  <dbl> <fct>     <dbl>
1     57 Current     1     0.482
```

Observed (sick status) vs. Model 1 fitted $\text{Pr}(\text{sick})$

```
ggplot(m1_aug, aes(x = .fitted, y = sick)) +  
  geom_count() + xlim(0.30, 0.55) +  
  labs(title = "Model 1 (no interaction)",  
       sub = "Training Data")
```

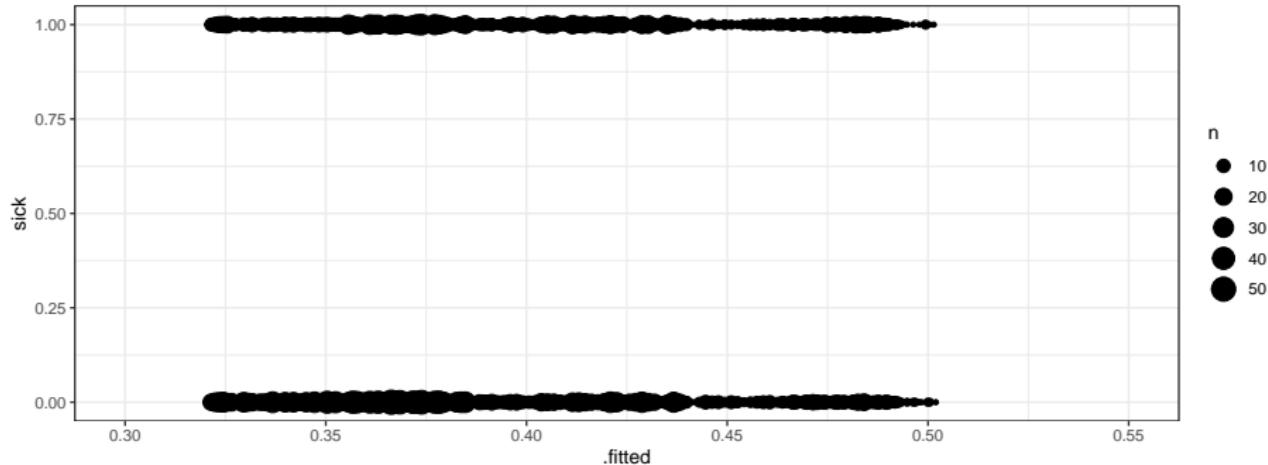
Model 1 (no interaction)



Observed (sick status) vs. Model 2 fitted Pr(sick)

```
ggplot(m2_aug, aes(x = .fitted, y = sick)) +  
  geom_count() + xlim(0.30, 0.55) +  
  labs(title = "Model 2 (with interaction)",  
       sub = "Training Data")
```

Model 2 (with interaction)



Making Classification Decisions

- Our outcome is `sick`, where `sick = 1` if `physact > 0`, otherwise `sick = 0`.
- We can establish a classification rule based on our model's predicted probabilities of `sick = 1`.
- 0.5 is a natural (but not inevitable) cut point.
 - if `.fitted` is below 0.50, we'll predict `sick = 0`
 - if `.fitted` is 0.50 or larger, we'll predict `sick = 1`.

```
m1_aug %$% table(.fitted >= 0.50, sick)
```

sick		
	0	1
FALSE	3058	2005
TRUE	75	49

Standard Epidemiological Format

Confusion matrix for Model mod1 in the training sample.

```
confuse_m1 <- m1_aug %>%
  mutate(sick_obs = factor(sick == "1"),
         sick_pred = factor(.fitted >= 0.50),
         sick_obs = fct_relevel(sick_obs, "TRUE"),
         sick_pred = fct_relevel(sick_pred, "TRUE")) %$%
  table(sick_pred, sick_obs)
```

```
confuse_m1
```

		sick_obs
		TRUE FALSE
sick_pred	TRUE	49 75
	FALSE	2005 3058

Terminology associated with the Confusion Matrix

```
confuse_m1
```

		sick_obs
sick_pred		TRUE FALSE
TRUE	49	75
FALSE	2005	3058

- Total Observations = $49 + 75 + 2005 + 3058 = 5187$
- Correct Predictions = $49 + 3058 = 3107$, or 59.9% accuracy
- Incorrect Predictions = $75 + 2005 = 2080$ (40.1%)
- Observed TRUE = $49 + 2005 = 2054$, or 39.6% prevalence
- Predicted TRUE = $49 + 75 = 124$, or 2.4% detection prevalence

Other Summaries from a Confusion Matrix

```
confuse_m1
```

		sick_obs
sick_pred	TRUE	FALSE
TRUE	49	75
FALSE	2005	3058

- Sensitivity = $49 / (49 + 2005) = 2.4\%$ (also called Recall)
 - if the subject actually was sick, our model predicts that 2.4% of the time
- Specificity = $3058 / (3058 + 75) = 97.6\%$
 - if the subject was actually not sick, our model predicts that 97.6% of the time
- Positive Predictive Value (or Precision) = $49 / (49 + 75) = 39.5\%$
 - our predictions of sick were correct 39.5% of the time
- Negative Predictive Value = $3058 / (3058 + 2005) = 60.4\%$
 - our predictions of “not sick” were correct 60.4% of the time

Confusion Matrix for mod2 (training sample)

We can obtain a similar confusion matrix for model mod2 using the same (arbitrary) cutoff of .fitted >= 0.5 to indicate sick.

```
confuse_m1
```

sick_pred	sick_obs	
	TRUE	FALSE
TRUE	49	75
FALSE	2005	3058

```
confuse_m2
```

sick_pred	sick_obs	
	TRUE	FALSE
TRUE	2	3
FALSE	2052	3130

Which of these confusion matrices looks better?

Get confusion matrix more easily?

Switch to a 0.45 cutoff...

```
m1_aug <- m1_aug %>%
  mutate(obs = factor(sick),
         pred = factor(ifelse(.fitted >= 0.45, 1, 0)))

conf_mat(data = m1_aug, truth = obs, estimate = pred)
```

		Truth
Prediction	0	1
	0 2679	1642
1 454	412	

Accuracy and Kappa Results for mod_1

```
metrics(data = m1_aug, truth = obs, estimate = pred) %>%  
  kable(digits = 6)
```

.metric	.estimator	.estimate
accuracy	binary	0.595913
kap	binary	0.061834

- Kappa = a correlation statistic from -1 to +1, with complete agreement +1 and complete disagreement -1.
- Kappa measures the inter-rater reliability of our predicted and true classifications.

Confusion Matrix for mod_2 with 0.45 cutoff

```
m2_aug <- m2_aug %>%
  mutate(obs = factor(sick),
         pred = factor(ifelse(.fitted >= 0.45, 1, 0)))

conf_mat(data = m2_aug, truth = obs, estimate = pred)
```

		Truth	
		0	1
Prediction	0	2582	1561
	1	551	493

- $493 + 2582 = 3075$ accurate predictions (59.3% accuracy)
- Sensitivity = $493 / (493 + 1561) = 24.0\%$
 - for the people who were actually sick, we made correct predictions 24% of the time
- Specificity = $2582 / (2582 + 551) = 82.4\%$
 - for the people who weren't actually sick, we made correct predictions 82.4% of the time with this decision rule and model mod_2.

Holdout Sample?

```
mod1_aug_test <- augment(mod1, newdata = d6_test,
                           type.predict = "response") %>%
  mutate(obs = factor(sick),
         pred = factor(ifelse(.fitted >= 0.45, 1, 0)))  
  
mod2_aug_test <- augment(mod2, newdata = d6_test,
                           type.predict = "response") %>%
  mutate(obs = factor(sick),
         pred = factor(ifelse(.fitted >= 0.45, 1, 0)))
```

metrics for test sample, models 1 and 2

```
bind_cols(  
  metrics(data = mod1_aug_test,  
          truth = obs, estimate = pred) %>%  
    select(.metric, mod1 = .estimate),  
  metrics(data = mod2_aug_test,  
          truth = obs, estimate = pred) %>%  
    select(mod2 = .estimate)  
)
```

```
# A tibble: 2 x 3  
  .metric     mod1     mod2  
  <chr>      <dbl>   <dbl>  
1 accuracy  0.609   0.604  
2 kap        0.0938  0.0979
```

What's Next?

Expanding our options with `tidymodels` and the Harrell-verse...

- Fitting linear and logistic regression models in new ways
- Evaluating the success of our models in new ways
- Incorporating imputation approaches more seamlessly

Please don't forget to submit your Project A proposal by Monday at 9 PM.

432 Class 07 Slides

thomaselove.github.io/432

2022-02-01

Setup

```
library(mosaic)          ## auto-loads mosaicData
```

Warning in register(): Can't find generic `scale_type` in package ggplot2 to register S3 method.

```
library(here)
library(janitor)
library(magrittr)
library(knitr)
library(broom)
library(patchwork)
library(GGally)          ## for scatterplot matrix
library(rms)              ## auto-loads Hmisc
library(tidyverse)

theme_set(theme_bw())
```

Today's Materials

- The HELP study (today's data) and preliminaries
- Using `ols` to fit a linear model
 - Obtaining coefficients and basic summaries
 - Validating summary statistics like R^2
 - ANOVA in `ols`
 - Plot Effects with `summary` and `Predict`
 - Building and using a nomogram
 - Evaluating Calibration
 - Influential points and `dfbeta`
- Spending Degrees of Freedom on Non-Linearity
 - The Spearman ρ^2 (rho-squared) plot
- Building Non-Linear Predictors in `ols`
 - Polynomial Functions
 - Restricted Cubic Splines
 - Restricting Interaction (Product) Terms

Today's Data, from the HELP study

Today's Data (day7, from the HELP study)

Today's 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. We will look at 453 subjects with complete data today.

```
day7 <- tibble(mosaicData::HELPrct) %>%
  select(id, cesd, age, sex, subst = substance,
         mcs, pcs, pss_fr)
```

```
dim(day7)
```

```
[1] 453    8
```

Key Variables for Today

Variable	Description
<code>id</code>	subject identifier
<code>cesd</code>	Center for Epidemiologic Studies Depression measure (higher scores indicate more depressive symptoms)
<code>age</code>	subject age (in years)
<code>sex</code>	female ($n = 107$) or male ($n = 346$)
<code>subst</code>	primary substance of abuse (alcohol, cocaine or heroin)
<code>mcs</code>	SF-36 Mental Component Score (lower = worse status)
<code>pcs</code>	SF-36 Physical Component Score (lower = worse status)
<code>pss_fr</code>	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/>.

The day7 categorical data

```
day7 %>% tabyl(sex, subst) %>%
  adorn_totals(where = c("row", "col")) %>%
  adorn_percentages() %>%
  adorn_pct_formatting() %>%
  adorn_ns(position = "front") %>%
  adorn_title(placement = "combined") %>%
  kable(align = 'lrrrr')
```

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%)

Summarizing the day7 quantitative data

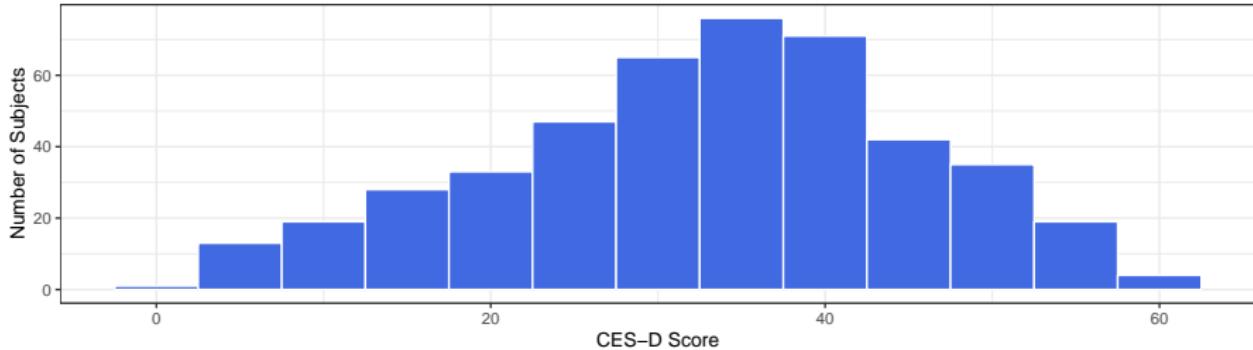
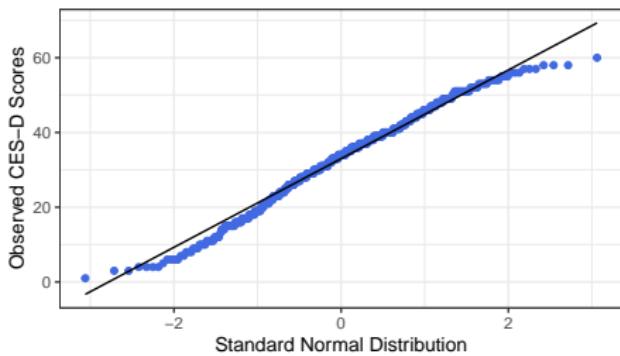
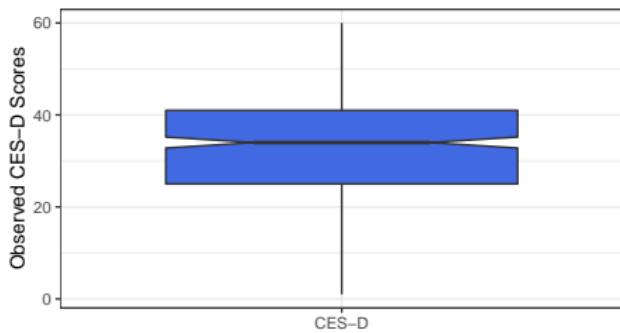
```
day7 %>% select(cesd, age, mcs, pcs, pss_fr) %>%  
  inspect(digits = 2)
```

quantitative variables:

	name	class	min	Q1	median	Q3	max	mean	sd	n	
...1	cesd	integer	1.0	25		34	41	60	32.8	12.5	453
...2	age	integer	19.0	30		35	40	60	35.7	7.7	453
...3	mcs	numeric	6.8	22		29	41	62	31.7	12.8	453
...4	pcs	numeric	14.1	40		49	57	75	48.0	10.8	453
...5	pss_fr	integer	0.0	3		7	10	14	6.7	4.0	453
		missing									
...1		0									
...2		0									
...3		0									
...4		0									
...5		0									

Our Outcome (CES-Depression score)

CES-D Depression Scores from day7 data
Higher CES-D indicates more depressive symptoms



n = 453, no missing data

Hmisc::describe() for our outcome CES-D

```
day7 %$% describe(cesd)
```

cesd : CESD at baseline

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

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

- Info measures the variable's information between 0 and 1: the higher the Info, the more continuous the variable is (the fewer ties there are.)
- Gmd = Gini's mean difference, a robust measure of variation. If you randomly selected two of the 453 subjects many times, the mean difference in cesd would be 14.23 points.

We have some labels in our data

```
str(day7)
```

```
tibble [453 x 8] (S3: tbl_df/tbl/data.frame)
$ id      : int [1:453] 1 2 3 4 5 6 7 8 9 10 ...
..- attr(*, "label")= chr "subject ID"
$ 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
..- attr(*, "label")= chr "sex"
$ subst   : Factor w/ 3 levels "alcohol","cocaine",...: 2 1 3 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"
```

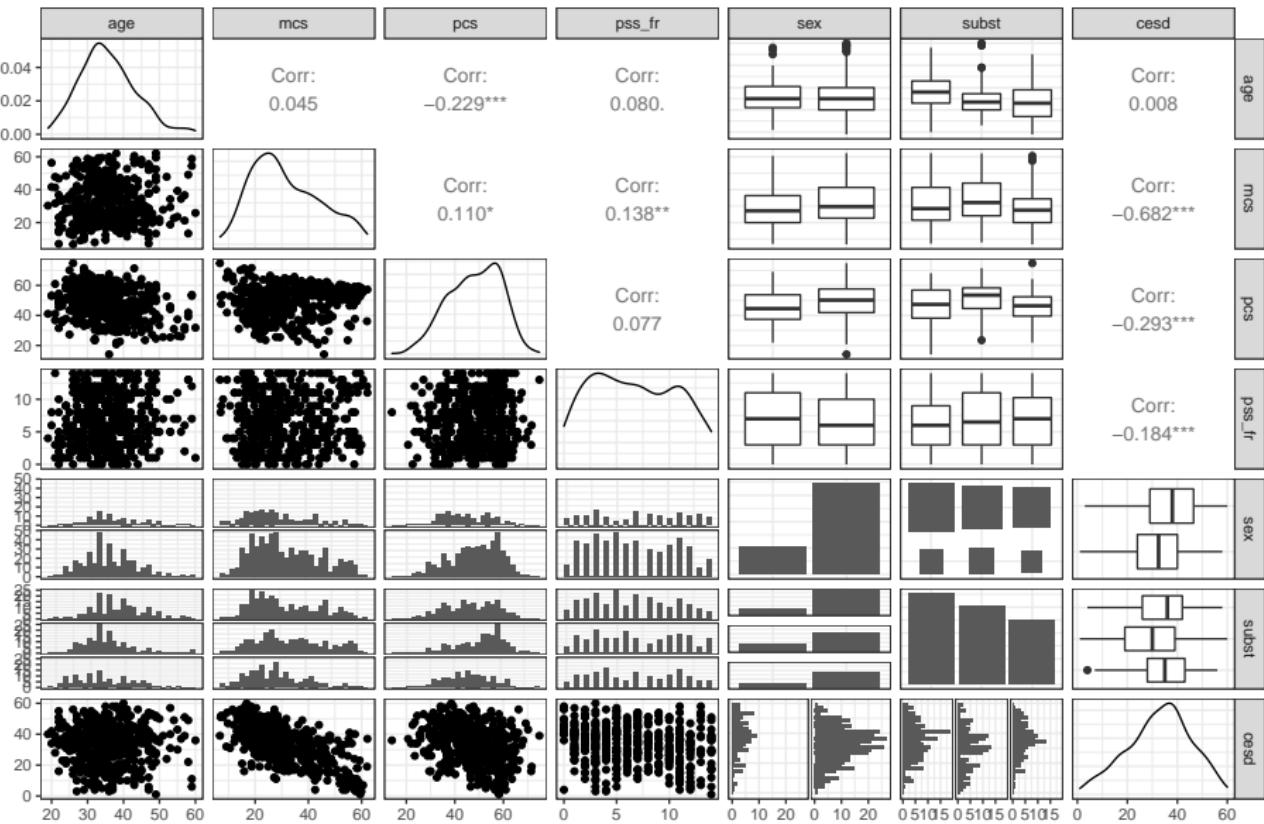
Scatterplot Matrix (code)

```
temp <- day7 %>%
  select(age, mcs, pcs, pss_fr, sex, subst, cesd)

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

Note that we're placing the outcome (cesd) last, and we'll want to set message = FALSE in the code chunk when producing the result (next slide.)

Scatterplot Matrix (result)



Saving the Data Set

```
saveRDS(day7, here("data", "day7.Rds"))
```

Could also have used

```
saveRDS(day7, here("data/day7.Rds"))
```

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:

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

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

This leaves the following questions:

- ① What's the `datadist` stuff doing?
- ② Why use `x = TRUE`, `y = TRUE` in the fit?

What is datadist?

Before we fit any ols model to data from `my_tibble`, we'll use:

```
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.

- excerpted from the datadist documentation

Why use x = TRUE, y = TRUE in the fit?

Once we've set up the distribution summaries with the datadist code, we fit linear regression models using the same fitting routines as lm with ols:

```
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 that we'll need in building plots and summaries of the fit.
- The defaults are x = FALSE, y = FALSE, but in this class, we'll always want to include these additional pieces.

Using ols to fit a Two-Predictor Model

Now, we'll fit an ols model predicting our outcome (cesd) using two predictors (mcs and subst) using the day7 tibble.

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

```
dd <- datadist(day7)
```

```
options(datadist = "dd")
```

```
mod1 <- ols(cesd ~ mcs + subst, data = day7,  
            x = TRUE, y = TRUE)
```

Contents of mod1?

mod1

```
> mod1
Linear Regression Model

ols(formula = cesd ~ mcs + subst, data = day7, 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

      Coef    S.E.      t      Pr(>|t|)
Intercept  55.3026 1.2724  43.46 <0.0001
mcs        -0.6570 0.0337 -19.48 <0.0001
subst=cocaine -3.4440 1.0055  -3.43 0.0007
subst=heroin  -1.7791 1.0681  -1.67 0.0965
```

- Likelihood Ratio Test?
- What is the discrimination index g?

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.

- 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 `ptsd` values, from `Hmisc::describe`, which was `Gmd = 14.23`.

Validate the summary statistics of an ols fit

- Can we validate summary statistics by resampling?

```
set.seed(4322022)  
validate(mod1)
```

	index.orig	training	test	optimism	index.corrected	n
R-square	0.4787	0.4827	0.4741	0.0086	0.4701	40
MSE	81.4606	81.6585	82.1755	-0.5170	81.9776	40
g	9.8272	9.9173	9.8011	0.1162	9.7110	40
Intercept	0.0000	0.0000	0.2719	-0.2719	0.2719	40
Slope	1.0000	1.0000	0.9914	0.0086	0.9914	40

- The data used to fit the model provide an over-optimistic view of the quality of fit.
- We're interested here in assessing how well the model might work in new data, and to do so, we can use a resampling approach.
- Consider R^2 here...

Interpreting the Resampling Validation Results

	index.orig	training	test	optimism	index.corrected	n
R-square	0.4787	0.4827	0.4741	0.0086	0.4701	40

- index.orig for R^2 is 0.4787. That's what we get from the data we used to fit the model, and is what we see in our standard output.
- 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 0.4827
 - Check each model in its test sample: average R^2 was 0.4741
- optimism = training result - test result = 0.0086
- index.corrected = index.orig - optimism = 0.4701

While our *nominal* R^2 is 0.4787 for this model, but correcting for optimism yields a *validated* R^2 of 0.4701.

- $R^2 = 0.4701$ better estimates how the model will perform in new data.

ANOVA for mod1 fit by ols

```
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 `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

```
summary(mod1)
```

Factor	Low	High	Diff.	Effect	S.E.	Lower	0.95	Upper	0.95
mcs	21.676	40.941	19.266	-12.6580	0.64984	-13.9350	-11.38100		
subst - cocaine:alcohol	1.000	2.000	NA	-3.4440	1.00550	-5.4200	-1.46790		
subst - heroin:alcohol	1.000	3.000	NA	-1.7791	1.06810	-3.8782	0.31993		

- How do we interpret the subst effects estimated by this model?
 - Effect of subst being cocaine instead of alcohol on ces_d is -3.4440 assuming no change in mcs, with 95% CI (-5.42, -1.47).
 - Effect of subst being heroin instead of alcohol on ces_d is -1.7791 assuming no change in mcs, with 95% CI (-3.88, +0.32).

But what about the mcs effect?

summary for mod1 fit by ols

```
summary(mod1)
```

Factor	Effects			Response : cesd					
	Low	High	Diff.	Effect	S.E.	Lower	0.95	Upper	0.95
mcs	21.676	40.941	19.266	-12.6580	0.64984	-13.9350	-11.38100		
subst - cocaine:alcohol	1.000	2.000	NA	-3.4440	1.00550	-5.4200	-1.46790		
subst - heroin:alcohol	1.000	3.000	NA	-1.7791	1.06810	-3.8782	0.31993		

- Effect of mcs: -12.6580 is the estimated change in cesd associated with a move from mcs = 21.676 (see Low value) to mcs = 40.941 (the High value) assuming no change in subst.
- ols chooses the Low and High values from the interquartile range.

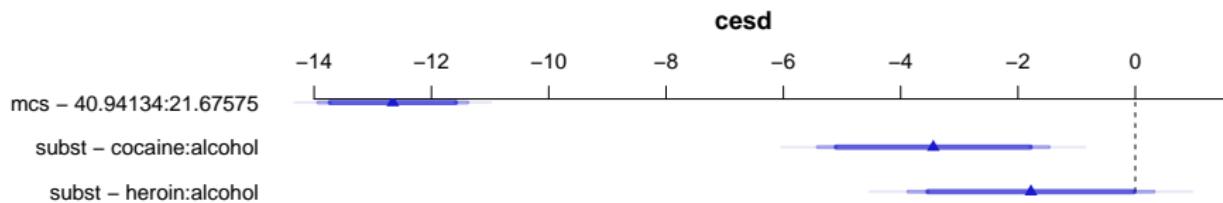
```
day7 %$% quantile(mcs, c(0.25, 0.75))
```

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

Plot the summary to see effect sizes

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

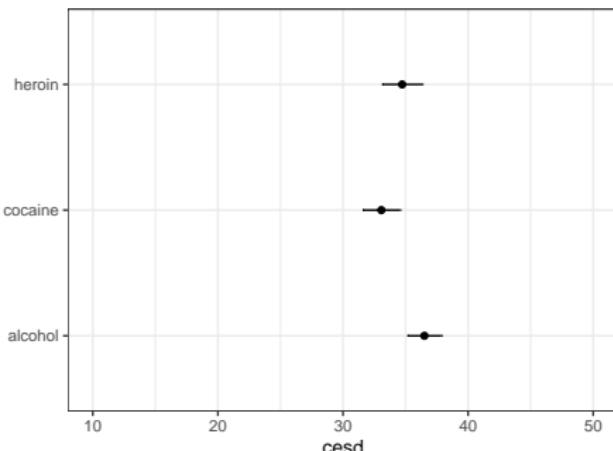
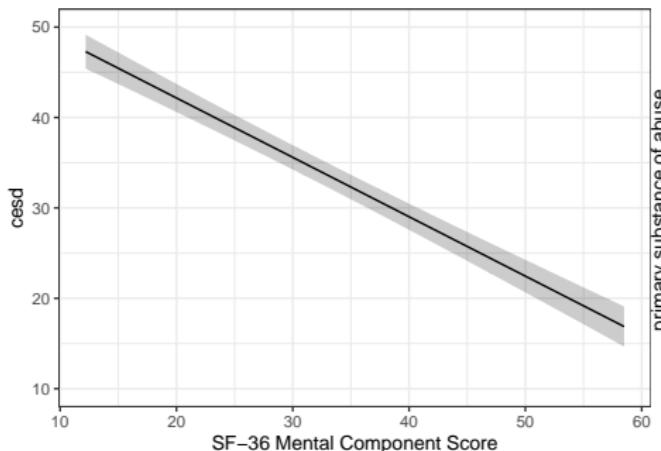
```
plot(summary(mod1))
```



- 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.

What do the individual effects look like?

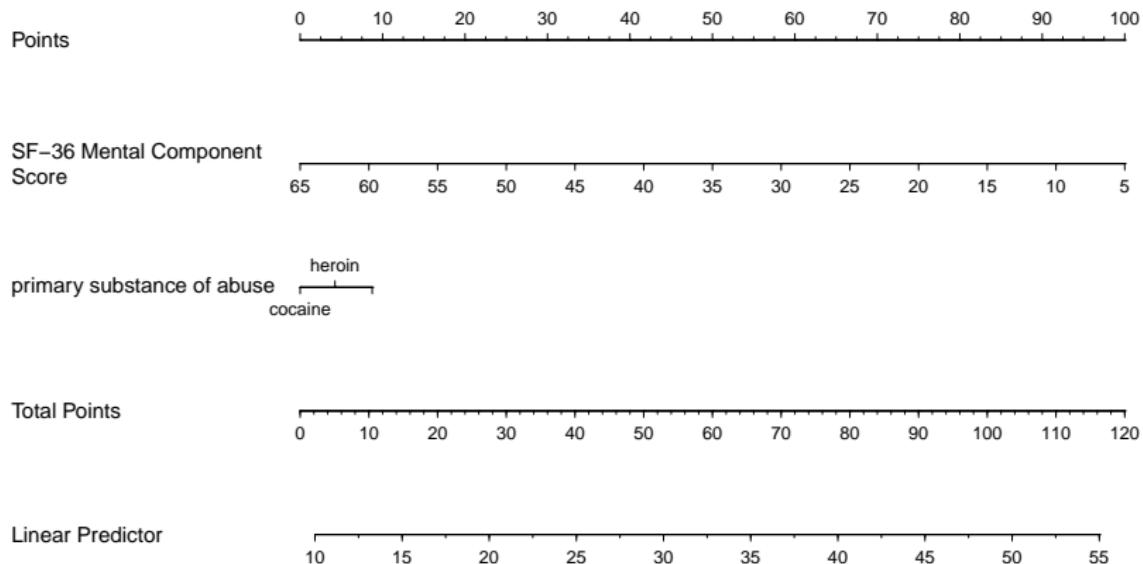
```
ggplot(Predict(mod1, conf.int = 0.95), layout = c(1,2))
```



- The left plot shows the impact of changing mcs on cesd holding subst at its baseline level (alcohol).
- The right plot shows the impact of changing subst on cesd holding mcs at its median value which is 28.602417.
- Defaults: add 95% CI bands and layout tries for a square.

Build a nomogram for the ols fit

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



Nomograms

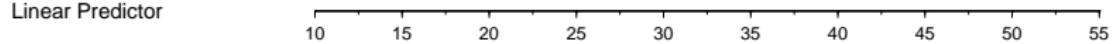
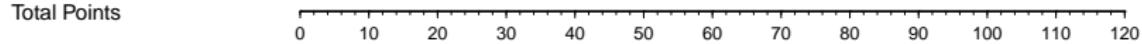
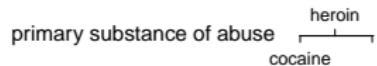
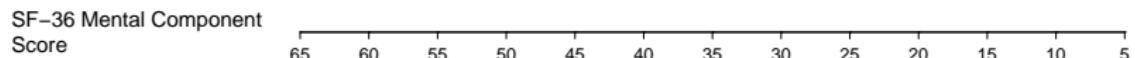
For complex models (this model isn't actually very complex) it can be helpful to have a tool that will help you see the modeled effects in terms of their impact on the predicted outcome.

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 the mod1 fit

Predicted cesd for a subject with mcs = 35 and subst = heroin?



Actual Prediction for such a subject...

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

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

```
1  
30.52766
```

- The broom package doesn't (really) support rms fits, and throws a warning (omitted here), but you could always refit the model with lm...

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

```
# A tibble: 1 x 3  
  mcs subst .fitted  
  <dbl> <chr>   <dbl>  
1     35 heroin    30.5
```

Assessing the Calibration of mod1

We would like our model to be well-calibrated, in the following sense . . .

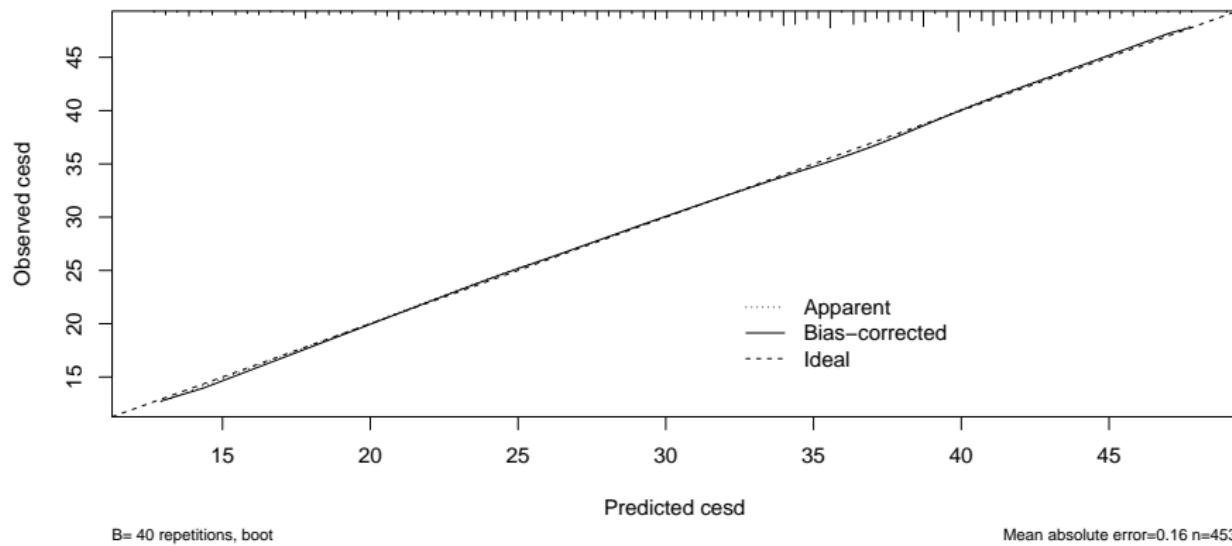
- Suppose our model assigns a predicted outcome of 6 to several subjects. If the model is well-calibrated, then we expect the mean of those subjects' actual outcomes to be very close to 6.

We'd like to look at the relationship between the observed cesd outcome and our predicted cesd from the model.

- The calibration plot we'll create provides two estimates (with and without bias-correction) of the predicted vs. observed values of our outcome, and compares these to the ideal scenario (predicted = observed).
- The plot uses resampling validation to produce bias-corrected estimates and uses lowess smooths to connect across predicted values.
- Calibration plots require $x = \text{TRUE}$, $y = \text{TRUE}$ in the ols fit.

Calibration Plot for mod1

```
set.seed(4320123); plot(calibrate(mod1))
```



n=453 Mean absolute error=0.16 Mean squared error=0.03522
0.9 Quantile of absolute error=0.32

Influential Points for mod1?

The dfbeta value for a particular subject and coefficient β is the change in the coefficient that happens when the subject is excluded from the model.

```
which.influence(mod1, cutoff = 0.2)
```

```
$Intercept  
[1] 8 351 405 433
```

```
$mcs  
[1] 351 402 450
```

```
$subst  
[1] 351
```

- These are the subjects that have absolute values of dfbetas that exceed the specified cutoff (default is 0.2 but it's an arbitrary choice.)

Show the influential points more directly?

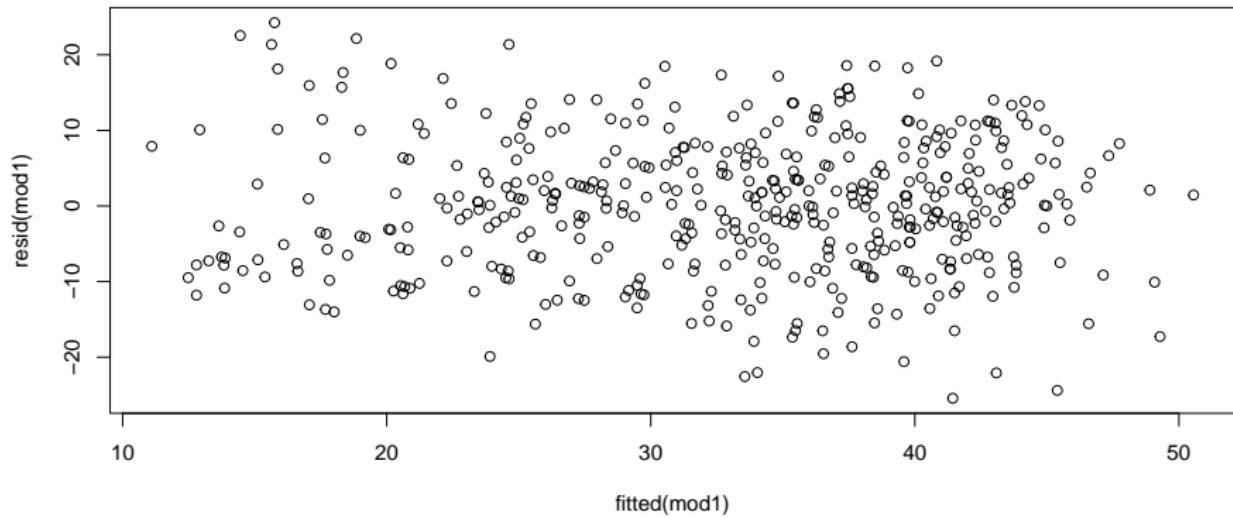
```
w <- which.influence(mod1, cutoff = 0.2)
d <- day7 %>% select(mcs, subst, cesd) %>% data.frame()
show.influence(w, d)
```

	Count	mcs	subst
8	1	9.16053	alcohol
351	3	*57.48944	*heroin
402	1	*55.47938	alcohol
405	1	15.07887	alcohol
433	1	18.59431	alcohol
450	1	*62.17550	alcohol

- Count = number of coefficients where this row appears influential.
- Use `day7 %>% slice(351)` to see row 351 in its entirety.
- Use residual plots (with an `lm` fit) to check Cook's distances.

Residuals vs. Fitted Values is easy from ols

```
plot(resid(mod1) ~ fitted(mod1))
```



Fitting all Residual Plots for mod1

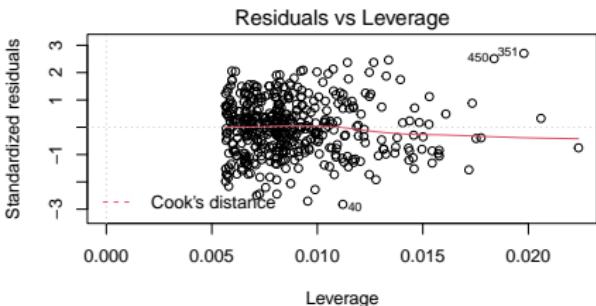
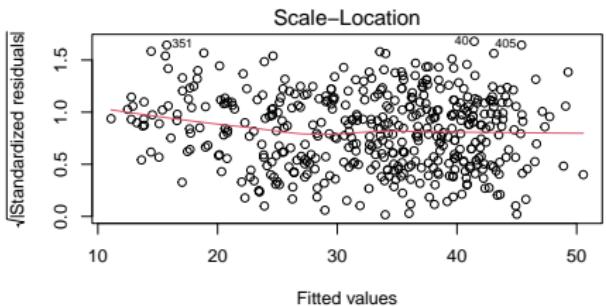
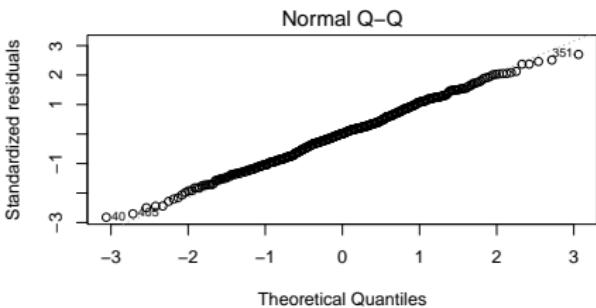
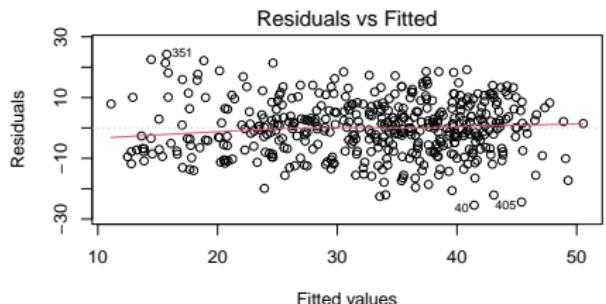
To fit more complete residual plots (and to do other things) we will fit the lm version of this same model...

```
mod1_lm <- lm(cesd ~ mcs + subst, data = day7)

par(mfrow = c(2,2))
plot(mod1_lm)
par(mfrow = c(1,1))
```

- Plots are shown on the next slide. While the subject in row 351 is more influential than most other points, it doesn't reach the standard of a problematic Cook's distance.

Residual Plots for mod1



Thinking about Non-Linear Terms?

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

Polynomial Regression

A polynomial in the variable x of degree D is a linear combination of the powers of x up to D . Fitting such a model creates a **polynomial regression**.

- Linear: $y = \beta_0 + \beta_1 x$
- Quadratic: $y = \beta_0 + \beta_1 x + \beta_2 x^2$
- Cubic: $y = \beta_0 + \beta_1 x + \beta_2 x^2 + \beta_3 x^3$
- Quartic: $y = \beta_0 + \beta_1 x + \beta_2 x^2 + \beta_3 x^3 + \beta_4 x^4$
- Quintic: $y = \beta_0 + \beta_1 x + \beta_2 x^2 + \beta_3 x^3 + \beta_4 x^4 + \beta_5 x^5$

An **orthogonal polynomial** sets up a model design matrix and then scales those columns so that each column is uncorrelated with the previous ones.

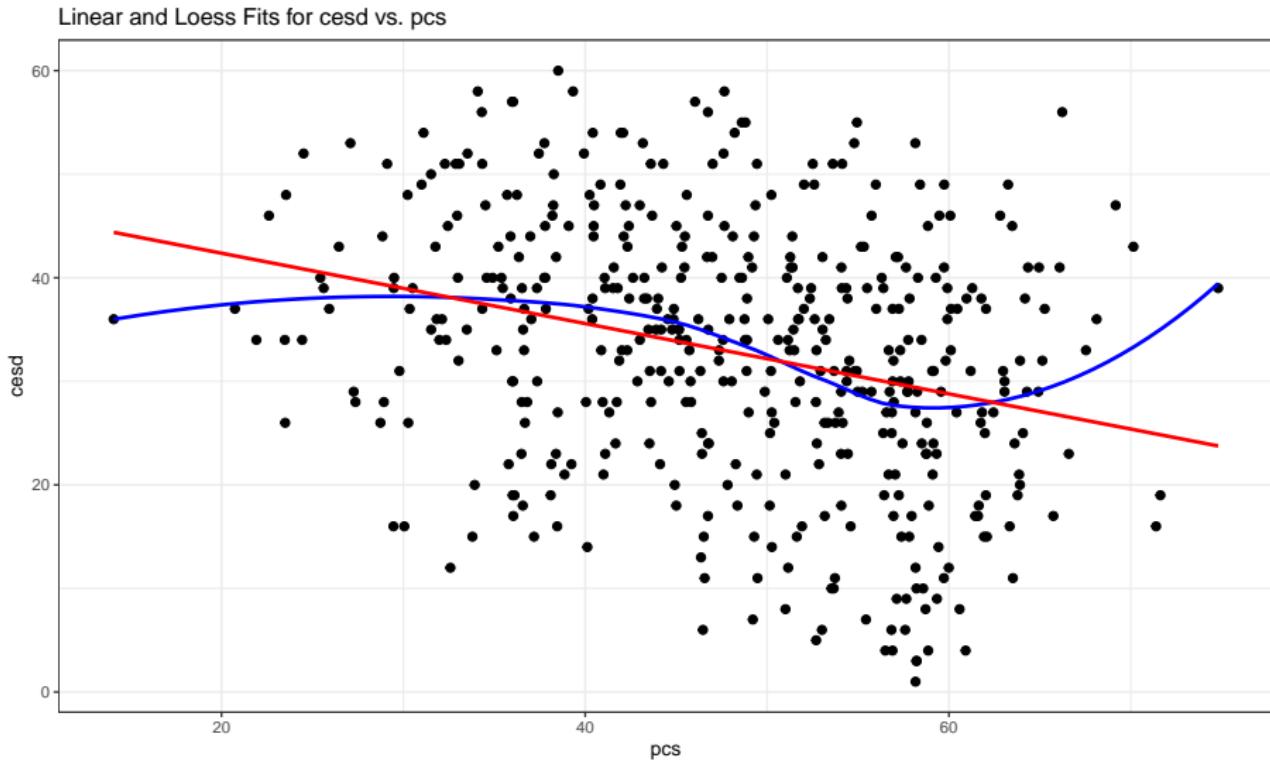
- This reduction in collinearity (correlation between predictors) lets us gauge whether the addition of any particular polynomial term improves model fit.

A new predictor: use pcs to predict cesd?

- Let's look at both a linear fit and a loess smooth to see if they indicate meaningfully different things about the association between pcs and cesd

```
ggplot(day7, 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")
```

Linear and Loess Fits for cesd with pcs



Fitting polynomial regressions with ols

```
dd <- datadist(day7)
options(datadist = "dd")

mod_B1 <- ols(cesd ~ pcs,
              data = day7, x = TRUE, y = TRUE)
mod_B2 <- ols(cesd ~ pol(pcs, 2),
              data = day7, x = TRUE, y = TRUE)
mod_B3 <- ols(cesd ~ pol(pcs, 3),
              data = day7, x = TRUE, y = TRUE)
```

- Note the use of `pol()` from the `rms` package here to fit orthogonal polynomials, rather than `poly()` which we used for an `lm` fit.

Model B1 (linear in pcs)

mod_B1

```
> mod_B1
Linear Regression Model

ols(formula = cesd ~ pcs, data = day7, 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
```

Model B2 (quadratic polynomial in pcs)

mod_B2

```
> mod_B2
Linear Regression Model

ols(formula = cesd ~ pol(pcs, 2), data = day7, x = TRUE, y = TRUE)

      Model Likelihood Discrimination
              Ratio Test      Indexes
Obs       453    LR chi2     40.68      R2      0.086
sigma11.9915   d.f.        2      R2 adj  0.082
d.f.       450  Pr(> chi2) 0.0000      g     4.199

Residuals

      Min      1Q Median      3Q      Max
-28.387 -7.750  0.591   8.634  29.697

      Coef    S.E.      t    Pr(>|t|)
Intercept 46.4007 8.7967  5.27 <0.0001
pcs        -0.2136 0.3867 -0.55 0.5809
pcs^2      -0.0014 0.0041 -0.33 0.7424
```

Model B3 (cubic polynomial in pcs)

mod_B3

```
> mod_B3
Linear Regression Model

ols(formula = cesd ~ pol(pcs, 3), data = day7, x = TRUE, y = TRUE)

      Model Likelihood Discrimination
      Ratio Test      Indexes
Obs     453    LR chi2     48.70      R2      0.102
sigmainv.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
```

Store the polynomial fits

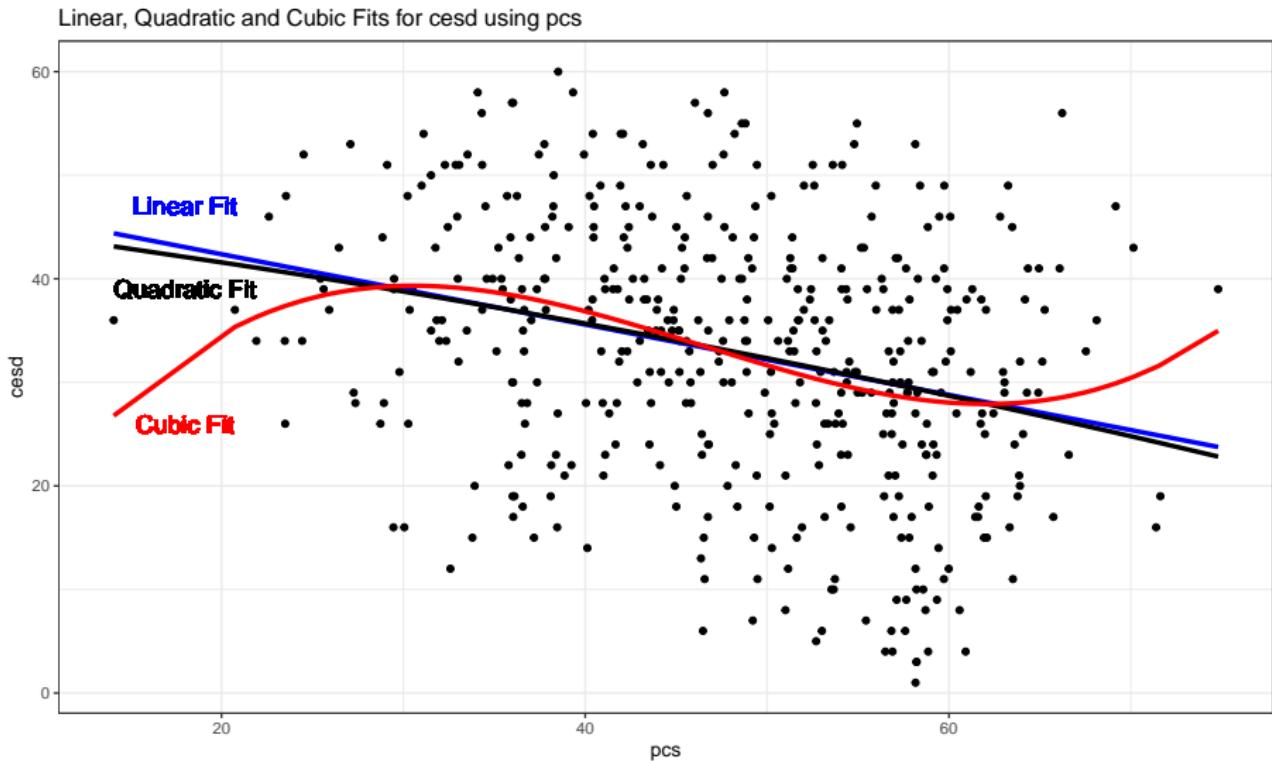
First, we need to store the values. Again broom doesn't play well with ols fits, so I'll just add the predictions as columns

```
cesd_fits <- day7 %>%  
  mutate(fitB1 = predict(mod_B1),  
        fitB2 = predict(mod_B2),  
        fitB3 = predict(mod_B3))
```

Code to plot polynomial fits

```
ggplot(cesd_fits, aes(x = pcs, y = cesd)) +
  geom_point() +
  geom_line(aes(x = pcs, y = fitB1),
            col = "blue", size = 1.25) +
  geom_line(aes(x = pcs, y = fitB2),
            col = "black", size = 1.25) +
  geom_line(aes(x = pcs, y = fitB3),
            col = "red", size = 1.25) +
  geom_text(x = 18, y = 47, label = "Linear Fit",
            size = 5, col = "blue") +
  geom_text(x = 18, y = 39, label = "Quadratic Fit",
            size = 5, col = "black") +
  geom_text(x = 18, y = 26, label = "Cubic Fit",
            size = 5, col = "red") +
  labs(title = "Linear, Quadratic and Cubic Fits for cesd us
```

The Polynomial Fits, plotted



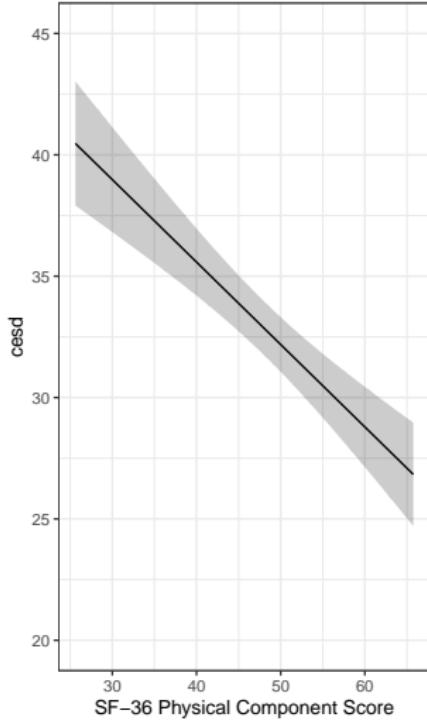
Code to plot polynomial fits with Predict

```
p1 <- ggplot(Predict(mod_B1)) + ggtitle("B1: Linear")
p2 <- ggplot(Predict(mod_B2)) + ggtitle("B2: Quadratic")
p3 <- ggplot(Predict(mod_B3)) + ggtitle("B3. Cubic")

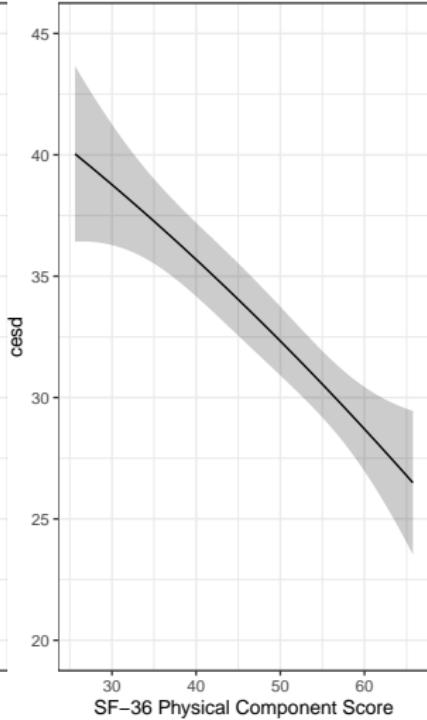
p1 + p2 + p3
```

Visualizing the polynomial fits with Predict

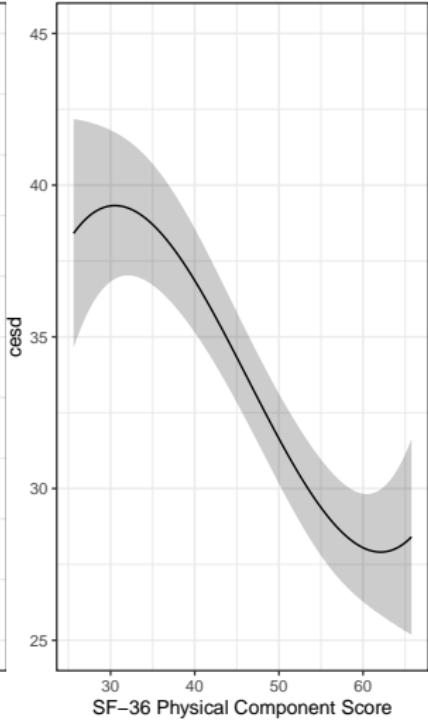
B1: Linear



B2: Quadratic



B3. Cubic



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.
 - 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.

Fitting Restricted Cubic Splines with ols

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

- 3 knots in modC3, 4 knots in modC4, and 5 knots in modC5

```
dd <- datadist(day7)
options(datadist = "dd")

mod_C3 <- ols(cesd ~ rcs(pcs, 3),
               data = day7, x = TRUE, y = TRUE)
mod_C4 <- ols(cesd ~ rcs(pcs, 4),
               data = day7, x = TRUE, y = TRUE)
mod_C5 <- ols(cesd ~ rcs(pcs, 5),
               data = day7, x = TRUE, y = TRUE)
```

Model C3 (3-knot spline in pcs)

mod_C3

```
> mod_C3
Linear Regression Model

ols(formula = cesd ~ rcs(pcs, 3), data = day7, 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
```

Model C4 (4-knot spline in pcs)

mod_C4

```
> mod_C4
Linear Regression Model

ols(formula = cesd ~ rcs(pcs, 4), data = day7, x = TRUE, y = TRUE)

      Model Likelihood   Discrimination
           Ratio Test       Indexes
Obs     453    LR chi2     51.31      R2      0.107
sigma1 8.648    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
```

Model C5 (5-knot spline in pcs)

mod_C5

```
> mod_C5
Linear Regression Model

ols(formula = cesd ~ rcs(pcs, 5), data = day7, x = TRUE, y = TRUE)

      Model Likelihood   Discrimination
              Ratio Test           Indexes
Obs       453    LR chi2     54.64      R2       0.114
sigma1 1.8345    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

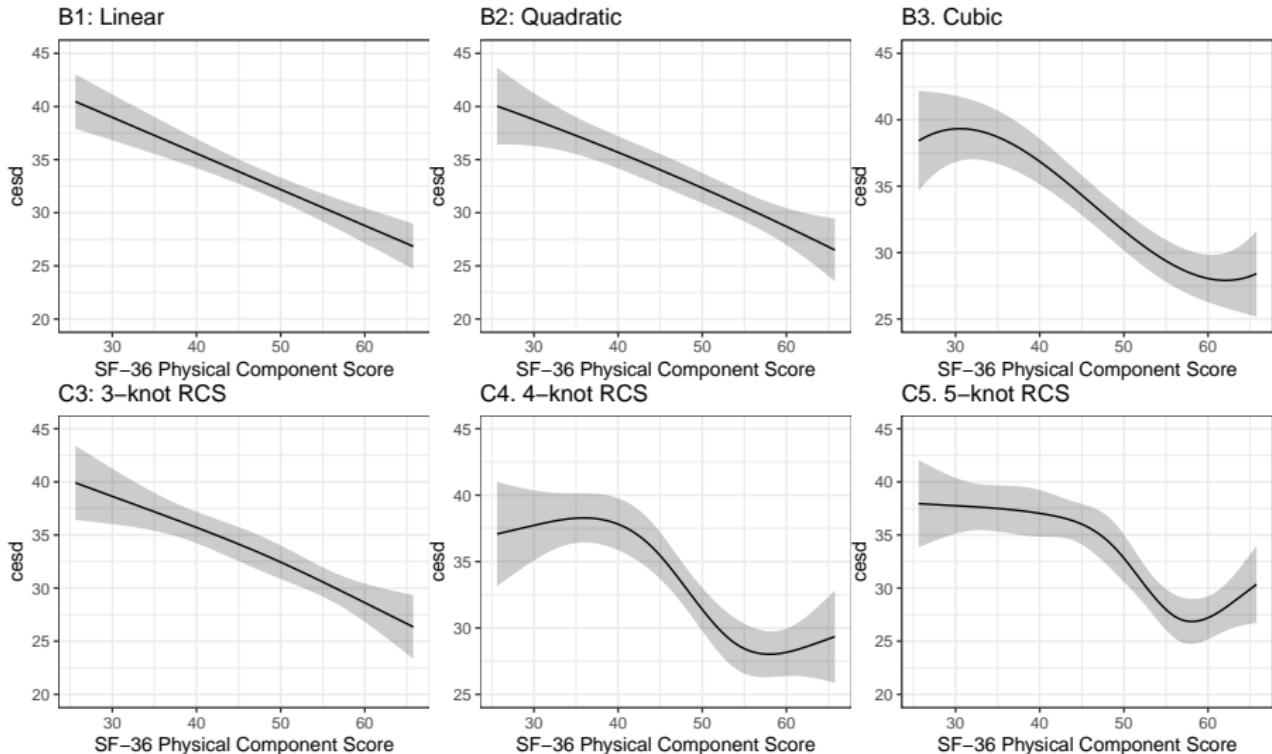
      Coef    S.E.      t    Pr(>|t|) 
Intercept 39.0631 7.8282  4.99 <0.0001
pcs        -0.0436 0.2332 -0.19 0.8517
pcs'       -0.2952 1.0079 -0.29 0.7697
pcs''      -3.1835 4.8079 -0.66 0.5082
pcs'''     14.4216 8.3721  1.72 0.0857
```

Code to plot all six fits

```
p1 <- ggplot(Predict(mod_B1)) + ggttitle("B1: Linear")
p2 <- ggplot(Predict(mod_B2)) + ggttitle("B2: Quadratic")
p3 <- ggplot(Predict(mod_B3)) + ggttitle("B3. Cubic")
p4 <- ggplot(Predict(mod_C3)) + ggttitle("C3: 3-knot RCS")
p5 <- ggplot(Predict(mod_C4)) + ggttitle("C4. 4-knot RCS")
p6 <- ggplot(Predict(mod_C5)) + ggttitle("C5. 5-knot RCS")

(p1 + p2 + p3) / (p4 + p5 + p6)
```

Visualizing the fits better?



Which of these models looks better?

- Compare our six models for the cesd to pcs association
- I used `set.seed(432)` then `validate(mod_B1)` etc.

Model	Index-Corrected R^2	Corrected MSE
B1 (linear)	0.0848	143.25
B2 (quadratic)	0.0752	142.49
B3 (cubic)	0.0909	143.73
C3 (3-knot RCS)	0.0732	143.31
C4 (4-knot RCS)	0.0870	144.00
C5 (5-knot RCS)	0.0984	141.44

- So which model has the best (validated) summaries?
- We'd need to look at residual plots, too, of course.

Data Spending: Non-Linearity Prior to Fits

Spending degrees of freedom wisely

- Suppose we have a data set with many possible predictors, and minimal theory or subject matter knowledge to guide us.
- We might want our final inferences to be as unbiased as possible. To accomplish this, we have to pay a penalty (in terms of degrees of freedom) for any “peeks” we make at the data in advance of fitting a model.
- So that rules out a lot of decision-making about non-linearity based on looking at the data, if our sample size isn’t much larger than 15 times the number of predictors we’re considering including in our model.
- In our case, we have $n = 453$ observations on 6 candidate predictors.
- In addition, adding non-linearity to our model costs additional degrees of freedom.
- What can we do?

Spearman's ρ^2 plot: A smart first step?

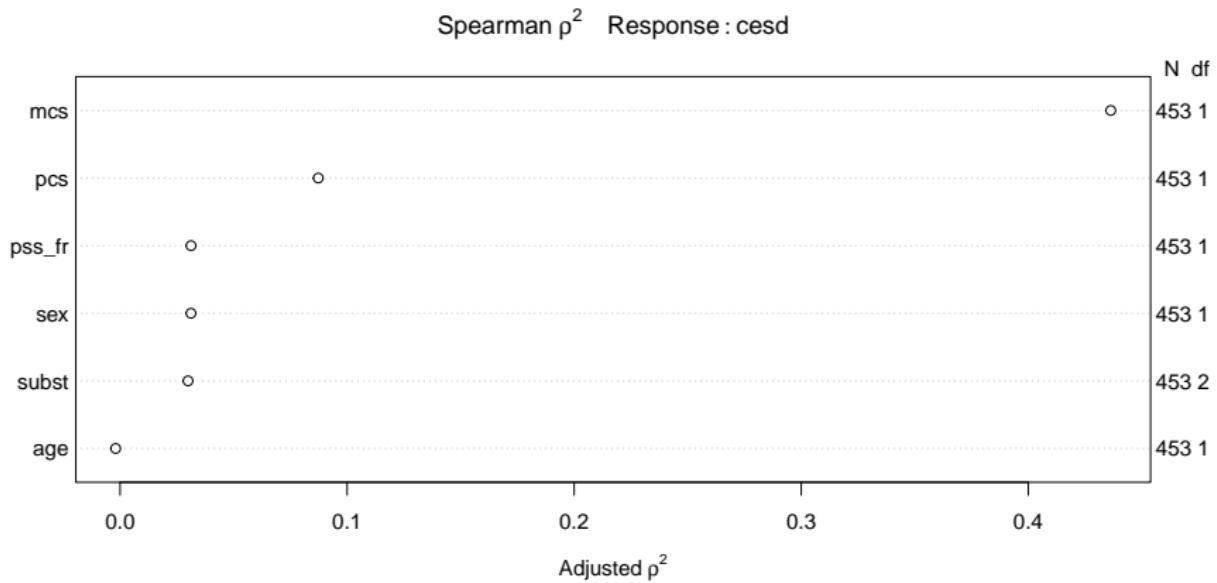
Spearman's ρ^2 is an indicator (not a perfect one) of potential predictive punch, but doesn't give away the game.

- Idea: Perhaps we should focus our efforts re: non-linearity on predictors that score better on this measure.

```
spear_cesd <- spearman2(cesd ~ mcs + subst + pcs +  
                           age + sex + pss_fr,  
                           data = day7)
```

Spearman's ρ^2 Plot

```
plot(spear_cesd)
```



Conclusions from Spearman ρ^2 Plot

- `mcs` is the most attractive candidate for a non-linear term, as it packs the most potential predictive punch, so if it does turn out to need non-linear terms, our degrees of freedom will be well spent.
 - This **does not** mean that `mcs` actually needs a non-linear term, or will show meaningfully better results if a non-linear term is included. We'd have to fit a model with and without non-linearity in `mcs` to know that.
 - Non-linearity will often take the form of a product term, a polynomial term, or a restricted cubic spline.
- `pcs`, also quantitative, has the next most potential predictive punch
- these are followed by `pss_fr` and `sex`.

Grim Reality

With 453 observations (452 df) we should be thinking about models with modest numbers of regression inputs.

- Non-linear terms (polynomials, splines) just add to the problem, as they need additional df to be estimated.

In this case, we might choose to include non-linear terms in just two or three variables (and that's it) and even that would be tough to justify with this modest sample size.

Contents of spear_cesd

spear_cesd

Spearman rho^2 Response variable:cesd

	rho2	F	df1	df2	P	Adjusted rho2	n
mcs	0.438	350.89	1	451	0.0000	0.436	453
subst	0.034	7.97	2	450	0.0004	0.030	453
pcs	0.089	44.22	1	451	0.0000	0.087	453
age	0.000	0.12	1	451	0.7286	-0.002	453
sex	0.033	15.56	1	451	0.0001	0.031	453
pss_fr	0.033	15.57	1	451	0.0001	0.031	453

Proposed New Model

Fit a model to predict cesd using:

- a 5-knot spline on mcs
- a 3-knot spline on pcs
- a linear term on pss_fr
- a linear term on age
- an interaction of sex with the main effect of mcs (restricting our model so that terms that are non-linear in both sex and mcs are excluded), and
- a main effect of subst

Perhaps more than we can reasonably do with 453 observations, but let's see how it looks.

Our new model mod2

```
dd <- datadist(day7)
options(datadist = "dd")

mod2 <- ols(cesd ~ rcs(mcs, 5) + rcs(pcs, 3) + sex +
            mcs %ia% sex + pss_fr + age + subst,
            data = day7, x = TRUE, y = TRUE)
```

- `%ia%` tells R to fit an interaction term with `sex` and the main effect of `mcs`.
- We have to include `sex` as a main effect for the interaction term (`%ia%`) to work here.

Our new, more complex model mod2

mod2

```
> mod2
Linear Regression Model

ols(formula = cesd ~ rcs(mcs, 5) + rcs(pcs, 3) + sex + mcs %ia%
    sex + pss_fr + age + subst, data = day7, x = TRUE, y = TRUE)

      Model Likelihood Discrimination
      Ratio Test      Indexes
Obs     453   LR chi2     349.44    R2      0.538
sigma8.6248 d.f.          12    R2 adj    0.525
d.f.     440 Pr(> chi2) 0.0000    g      10.439

Residuals

      Min       1Q   Median       3Q      Max
-26.7893 -5.9000  0.1545  5.5884  26.1304

                               Coef    S.E.      t    Pr(>|t|)
Intercept      76.3346 6.2540 12.21 <0.0001
mcs        -0.9306 0.2315 -4.02 <0.0001
mcs'         1.6607 2.5040  0.66 0.5075
mcs''        -2.8854 8.3945 -0.34 0.7312
mcs'''       0.2942 7.9390  0.04 0.9705
pcs        -0.2341 0.0883 -2.65 0.0083
pcs'        -0.0151 0.1000 -0.15 0.8797
sex=male     -2.0330 2.5456 -0.80 0.4249
mcs * sex=male -0.0129 0.0783 -0.17 0.8690
pss_fr       -0.2569 0.1046 -2.46 0.0144
age         -0.0466 0.0569 -0.82 0.4139
subst=cocaine -2.6999 0.9965 -2.71 0.0070
subst=heroin  -2.1741 1.0677 -2.04 0.0423
```

ANOVA for this model

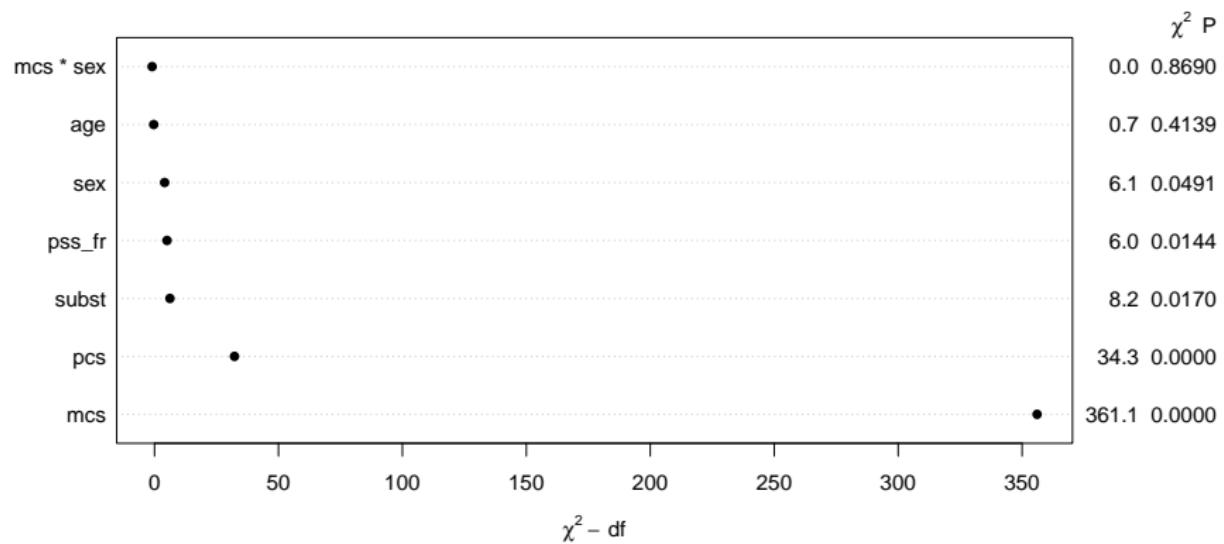
```
anova(mod2)
```

		Analysis of Variance		Response: cesd		
Factor		d.f.	Partial SS	MS	F	P
mcs	(Factor+Higher Order Factors)	5	26857.364670	5371.472934	72.21	<.0001
All Interactions		1	2.026255	2.026255	0.03	0.8690
Nonlinear		3	293.502251	97.834084	1.32	0.2688
pcs		2	2548.388579	1274.194290	17.13	<.0001
Nonlinear		1	1.705031	1.705031	0.02	0.8797
sex	(Factor+Higher Order Factors)	2	451.578352	225.789176	3.04	0.0491
All Interactions		1	2.026255	2.026255	0.03	0.8690
mcs * sex	(Factor+Higher Order Factors)	1	2.026255	2.026255	0.03	0.8690
pss_fr		1	448.812293	448.812293	6.03	0.0144
age		1	49.758786	49.758786	0.67	0.4139
subst		2	611.625952	305.812976	4.11	0.0170
TOTAL NONLINEAR		4	293.512204	73.378051	0.99	0.4146
TOTAL NONLINEAR + INTERACTION		5	294.601803	58.920361	0.79	0.5558
REGRESSION		12	38058.315322	3171.526277	42.64	<.0001
ERROR		440	32730.174744	74.386761		

- Remember that this ANOVA testing is sequential, other than the TOTALs.
- We can also plot the ANOVA results, for example...

Plotting ANOVA results for mod2

```
plot(anova(mod2))
```



Validation of Summary Statistics

```
set.seed(432); validate(mod2)
```

```
> set.seed(432); validate(mod2)
      index.orig training   test optimism index.corrected n
R-square      0.5376    0.5513  0.5233   0.0280      0.5096 40
MSE          72.2520   69.8358 74.4984  -4.6627     76.9147 40
g            10.4392   10.5053 10.2718   0.2335     10.2056 40
Intercept    0.0000    0.0000  0.7893  -0.7893      0.7893 40
Slope        1.0000    1.0000  0.9751   0.0249     0.9751 40
```

summary results for mod2

```
summary(mod2)
```

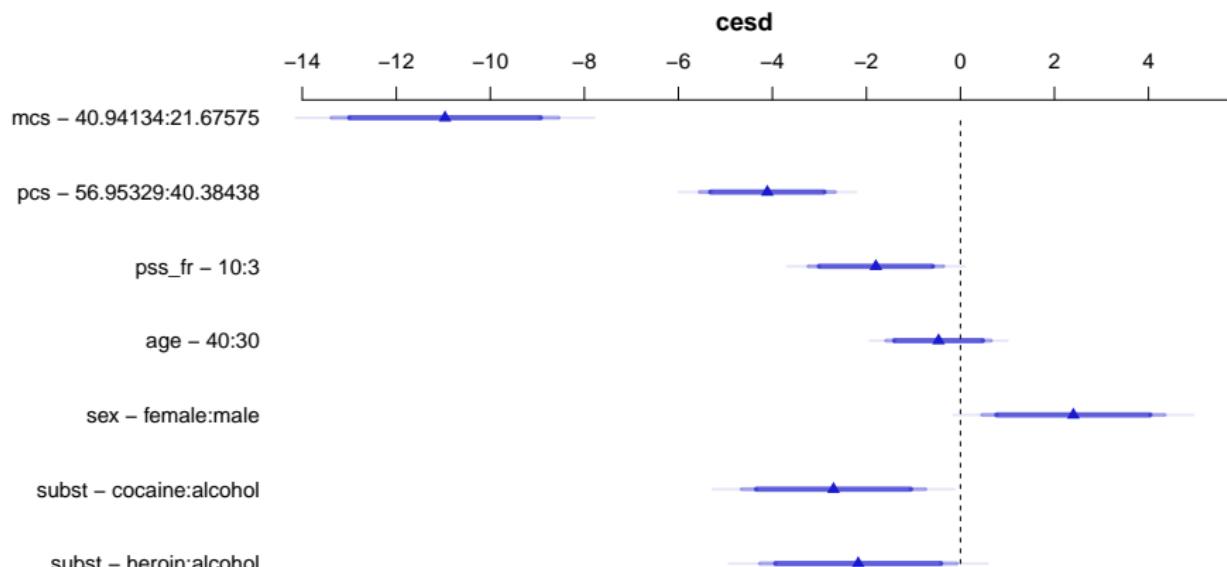
```
> summary(mod2)
      Effects          Response : cesd

    Factor       Low   High  Diff. Effect   S.E.   Lower 0.95 Upper 0.95
    mcs        21.676 40.941 19.266 -10.96400 1.23340 -13.38800 -8.539800
    pcs        40.384 56.953 16.569  -4.10790 0.73381 -5.55010 -2.665700
    pss_fr     3.000 10.000  7.000  -1.79860 0.73225 -3.23780 -0.359500
    age        30.000 40.000 10.000 -0.46552 0.56918 -1.58420 0.653130
    sex - female:male 2.000 1.000     NA  2.40260 0.99054  0.45577 4.349300
    subst - cocaine:alcohol 1.000 2.000     NA -2.69990 0.99647 -4.65830 -0.741430
    subst - heroin:alcohol 1.000 3.000     NA -2.17410 1.06770 -4.27250 -0.075632

Adjusted to: mcs=28.60242 sex=male
```

Plot of summary results for mod2

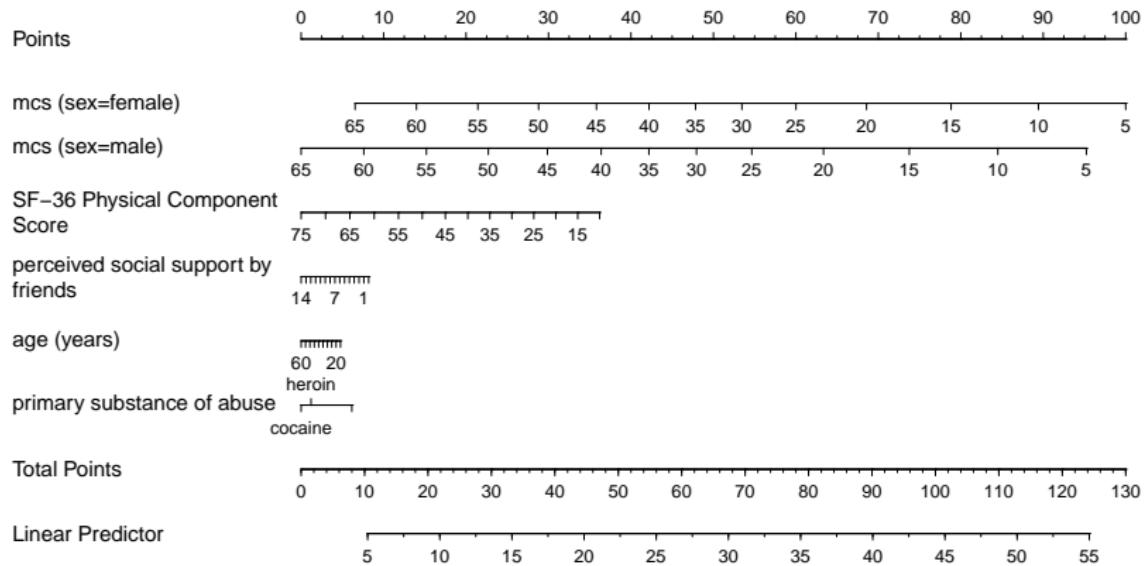
```
plot(summary(mod2))
```



Adjusted to:mcs=28.60242 sex=male

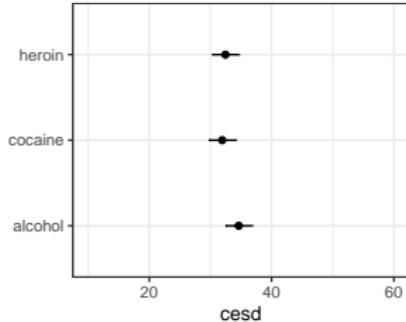
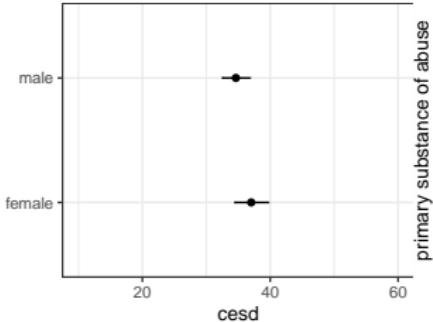
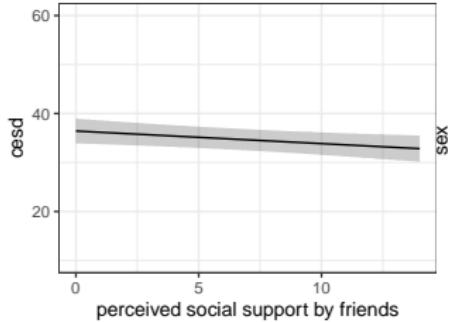
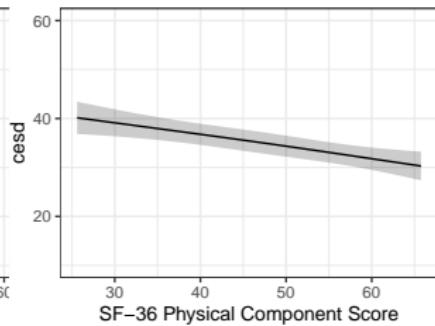
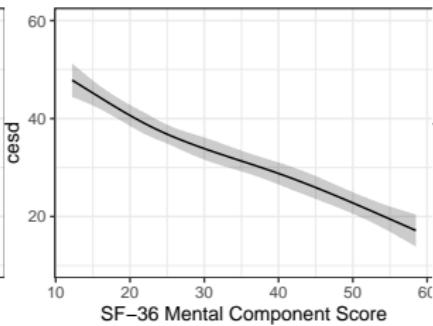
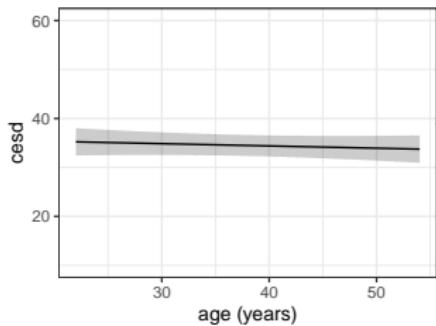
Nomogram for mod2

```
plot(nomogram(mod2))
```



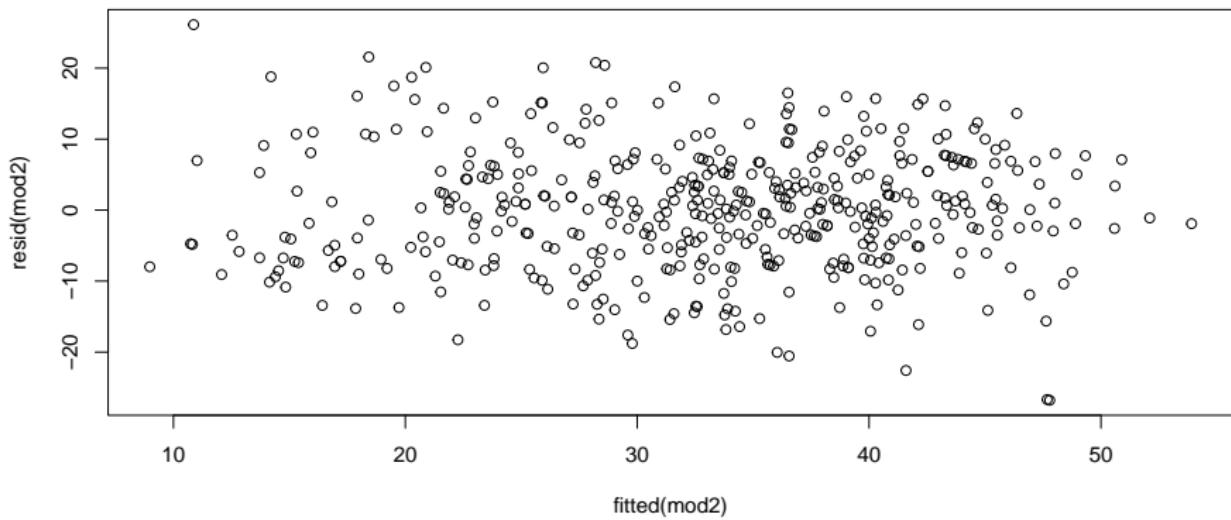
Seeing the impact of the modeling another way

```
ggplot(Predict(mod2))
```



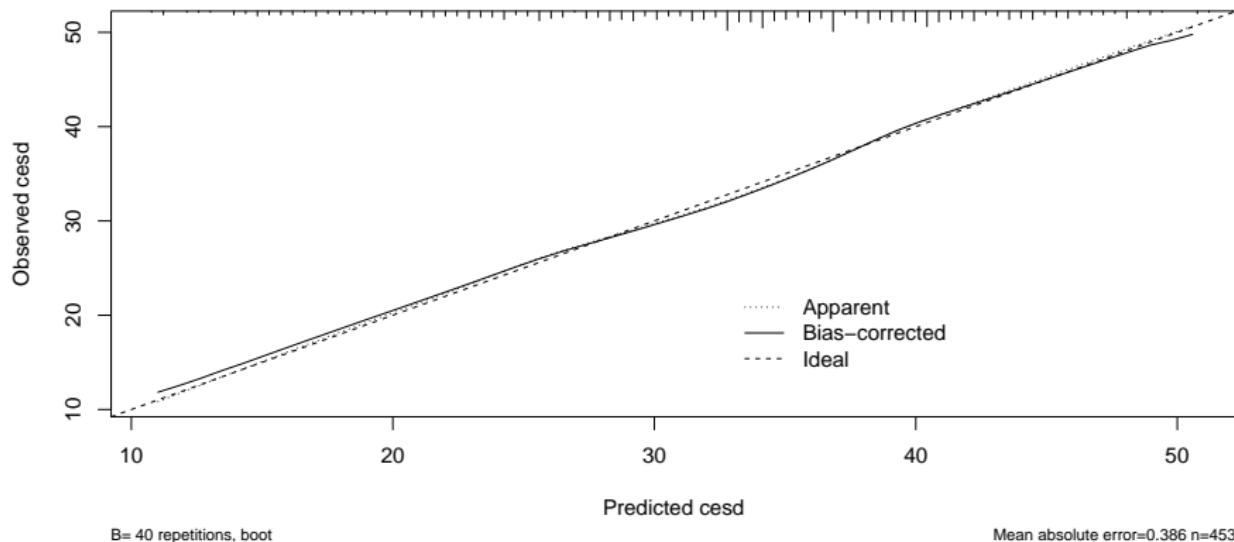
Residuals vs. Fitted Values to check assumptions

```
plot(resid(mod2) ~ fitted(mod2))
```



Checking the model's calibration

```
set.seed(432); plot(calibrate(mod2))
```



n=453 Mean absolute error=0.386 Mean squared error=0.19775
0.9 Quantile of absolute error=0.704

Limitations of `lm` for fitting complex linear models

We can certainly assess this big, complex model using `lm`, too:

- with in-sample summary statistics like adjusted R^2 , AIC and BIC,
- we can assess its assumptions with residual plots, and
- we can also compare out-of-sample predictive quality through cross-validation,

But to really delve into the details of how well this complex model works, and to help plot what is actually being fit, we'll probably want to fit the model using `ols`.

- In Project A, we expect some results that are most easily obtained using `lm` and others that are most easily obtained using `ols`.

Next Time

- The HERS data
- Fitting a more complex linear regression model
- Adding missing data into all of this, and running multiple imputation

Please remember to participate in the brief poll on Piazza about Chapter 2 of *The Signal and the Noise*.

432 Class 08 Slides

thomaselove.github.io/432

2022-02-03

Today's Agenda

- Data from the Heart and Estrogen/Progestin Study
- Using ols to fit linear regression models in the presence of missing values
- Using aregImpute to facilitate principled multiple imputation when fitting regressions
- Developing detailed regression results under a variety of imputation plans

Setup

```
library(magrittr); library(janitor)
library(here); library(knitr)
library(naniar); library(simputation)

library(rms)
library(tidyverse)
```

Today's Data

Heart and Estrogen/Progestin Study (HERS)

- Clinical trial of hormone therapy for the prevention of recurrent heart attacks and deaths among 2763 post-menopausal women with existing coronary heart disease (see Hulley et al 1998 and many subsequent references, including Vittinghoff, Chapter 4.)
- We're excluding the women in the trial with a diabetes diagnosis.

```
hers_raw <- read_csv(here("data/hersdata.csv")) %>%
  clean_names()

hers1 <- hers_raw %>%
  filter(diabetes == "no") %>%
  select(subject, ldl, ht, age, smoking, drinkany, sbp,
         physact, bmi, diabetes)
```

The Codebook (n = 2032)

Variable	Description
subject	subject code
HT	factor: hormone therapy or placebo
diabetes	yes or no (all are no in our sample)
ldl	LDL cholesterol in mg/dl
age	age in years
smoking	yes or no
drinkany	yes or no
sbp	systolic BP in mm Hg
physact	5-level factor, details next slide
bmi	body-mass index in kg/m ²

Goal Predict ldl using age, smoking, drinkany, sbp, physact and bmi, across both HT levels but restricted to women without diabetes.

The physact variable

```
hers1 %>% count(physact)
```

```
# A tibble: 5 x 2
```

	physact	n
	<chr>	<int>
1	about as active	674
2	much less active	107
3	much more active	252
4	somewhat less active	322
5	somewhat more active	677

Comparison is to activity levels for these women just before menopause.

Any missing data?

```
miss_var_summary(hers1)
```

```
# A tibble: 10 x 3
  variable n_miss pct_miss
  <chr>     <int>    <dbl>
1 ldl         7     0.344
2 drinkany    2     0.0984
3 bmi         2     0.0984
4 subject     0      0
5 ht          0      0
6 age         0      0
7 smoking     0      0
8 sbp          0      0
9 physact     0      0
10 diabetes   0      0
```

Single Imputation for drinkany, bmi and ldl

Since drinkany is a factor, we have to do some extra work to impute.

```
set.seed(432092)
```

```
hers2 <- hers1 %>%
  mutate(drinkany_n =
    ifelse(drinkany == "yes", 1, 0)) %>%
  impute_pmm(drinkany_n ~ age + smoking) %>%
  mutate(drinkany =
    ifelse(drinkany_n == 1, "yes", "no")) %>%
  impute_rlm(bmi ~ age + smoking + sbp) %>%
  impute_rlm(ldl ~ age + smoking + sbp + bmi)
```

Now, check missingness...

```
miss_var_summary(hers2)
```

```
# A tibble: 11 x 3
  variable   n_miss pct_miss
  <chr>       <int>    <dbl>
1 subject        0        0
2 ldl            0        0
3 ht             0        0
4 age            0        0
5 smoking        0        0
6 drinkany       0        0
7 sbp            0        0
8 physact        0        0
9 bmi            0        0
10 diabetes       0        0
11 drinkany_n     0        0
```

Multiple Imputation using aregImpute from Hmisc

Model to predict all missing values of any variables, using additive regression bootstrapping and predictive mean matching.

Steps are:

- ① aregImpute draws a sample with replacement from the observations where the target variable is observed, not missing.
- ② It then fits a flexible additive model to predict this target variable while finding the optimum transformation of it.
- ③ It then uses this fitted flexible model to predict the target variable in all of the original observations.
- ④ Finally, it imputes each missing value of the target variable with the observed value whose predicted transformed value is closest to the predicted transformed value of the missing value.

Fitting a Multiple Imputation Model

```
set.seed(4320132)
dd <- datadist(hers1)
options(datadist = "dd")
fit3 <- aregImpute(~ ldl + age + smoking + drinkany +
                     sbp + physact + bmi,
                     nk = c(0, 3:5), tlinear = FALSE,
                     data = hers1, B = 10, n.impute = 20)
```

Iteration 1 Iteration 2 Iteration 3 Iteration 4 Iteration 5 Iteration 6 Iteration 7 Iteration 8 Iteration 9 Iteration 10

Multiple Imputation using aregImpute from Hmisc

aregImpute requires specifications of all variables, and several other details:

- `n.impute` = number of imputations, we'll run 20
- `nk` = number of knots to describe level of complexity, with our choice `nk = c(0, 3:5)` we'll fit both linear models and models with restricted cubic splines with 3, 4, and 5 knots
- `tlinear` = `FALSE` allows the target variable to have a non-linear transformation when `nk` is 3 or more
- `B` = 10 specifies 10 bootstrap samples will be used
- `data` specifies the source of the variables

aregImpute Imputation Results (1 of 4)

fit3

Multiple Imputation using Bootstrap and PMM

```
aregImpute(formula = ~ldl + age + smoking + drinkany + sbp +
physact + bmi, data = hers1, n.impute = 20, nk = c(0, 3:5),
tlinear = FALSE, B = 10)
```

n: 2032 p: 7 Imputations: 20 nk: 0

Number of NAs:

ldl	age	smoking	drinkany	sbp	physact	bmi
7	0	0	2	0	0	2

fit3 Imputation Results (2 of 4)

R-squares for Predicting Non-Missing Values for Each Variable Using Last Imputations of Predictors

ldl	drinkany	bmi
0.041	0.014	0.109

fit3 Imputation Results (3 of 4)

Resampling results for determining the complexity of imputation models

Variable being imputed: ldl

Bootstrap bias-corrected summaries:

Statistic	nk = 0	nk = 3	nk = 4	nk = 5
R^2	0.0139	0.0149	0.00776	0.0124
mean absolute error	28.3594	42.9139	44.09937	39.8266
median abs. error	22.8301	35.5441	38.85302	32.6386

10-fold cross-validated:

Statistic	nk = 0	nk = 3	nk = 4	nk = 5
R^2	0.0214	0.0180	0.01517	0.0191
mean absolute error	145.7176	43.5007	45.02428	44.2456
median abs. error	141.4238	36.4102	38.88053	37.3141

fit3 Imputation Results (4 of 4)

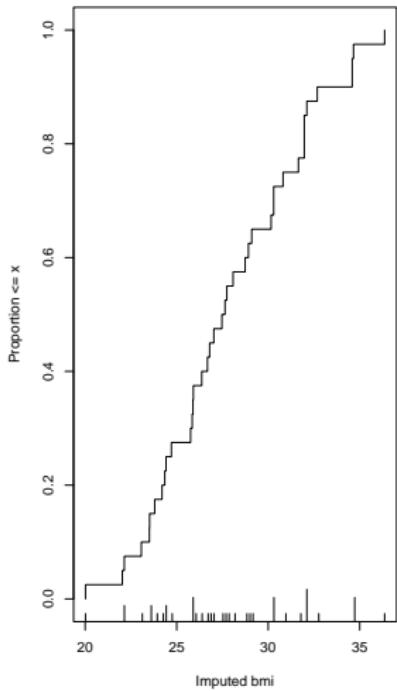
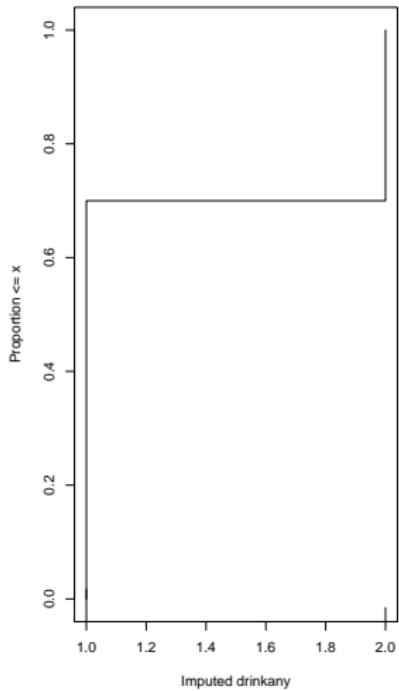
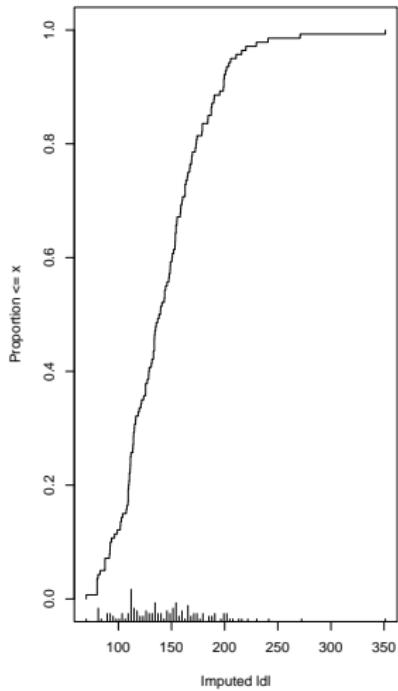
Variable being imputed: drinkany

		nk=0	nk=3	nk=4	nk=5
Bootstrap	R^2	0.0163	0.0113	0.0102	0.00986
10-fold cv	R^2	0.0205	0.0249	0.0163	0.01358
Bootstrap	mean error	0.4470	0.4568	0.4558	0.46624
10-fold cv	mean error	0.4450	0.4454	0.4476	0.44676
Bootstrap	median error	0.0000	0.0000	0.0000	0.00000
10-fold cv	median error	0.0000	0.0500	0.1000	0.00000

Variable being imputed: bmi

		nk=0	nk=3	nk=4	nk=5
Bootstrap	R^2	0.0845	0.0932	0.0946	0.0847
10-fold cv	R^2	0.0864	0.0903	0.0968	0.0899
Bootstrap	mean error	3.7829	4.8119	4.9226	5.1775
10-fold cv	mean error	27.6776	4.8359	4.9390	5.1136
Bootstrap	median error	2.9955	3.9704	3.9371	4.2634
10-fold cv	median error	27.0143	3.9894	3.9431	4.1876

A plot of the imputed values... (results)



A plot of the imputed values... (code)

```
par(mfrow = c(1,3))
plot(fit3)
par(mfrow = c(1,1))
```

- For ldl, we imputed most of the 7 missing subjects in most of the 20 imputation runs to values within a range of around 120 through 200, but occasionally, we imputed values that were substantially lower than 100.
- For drinkany we imputed about 70% no and 30% yes.
- For bmi, we imputed values ranging from about 23 to 27 in many cases, and up near 40 in other cases.
- This method never imputes a value for a variable that doesn't already exist in the data.

Kitchen Sink Model (Main Effects only)

```
mod_ks <- ols(ldl ~ age + smoking + drinkany + sbp +  
                  physact + bmi, data = hers2)  
anova(mod_ks)
```

Analysis of Variance Response: ldl

Factor	d.f.	Partial SS	MS	F	P
age	1	9330.911	9330.911	6.93	0.0085
smoking	1	8199.755	8199.755	6.09	0.0137
drinkany	1	6444.424	6444.424	4.79	0.0288
sbp	1	9274.287	9274.287	6.89	0.0087
physact	4	10874.528	2718.632	2.02	0.0891
bmi	1	15876.957	15876.957	11.80	0.0006
REGRESSION	9	60077.708	6675.301	4.96	<.0001
ERROR	2022	2721037.890	1345.716		

Spearman ρ^2 Plot

How should we prioritize the degrees of freedom we spend on non-linearity?

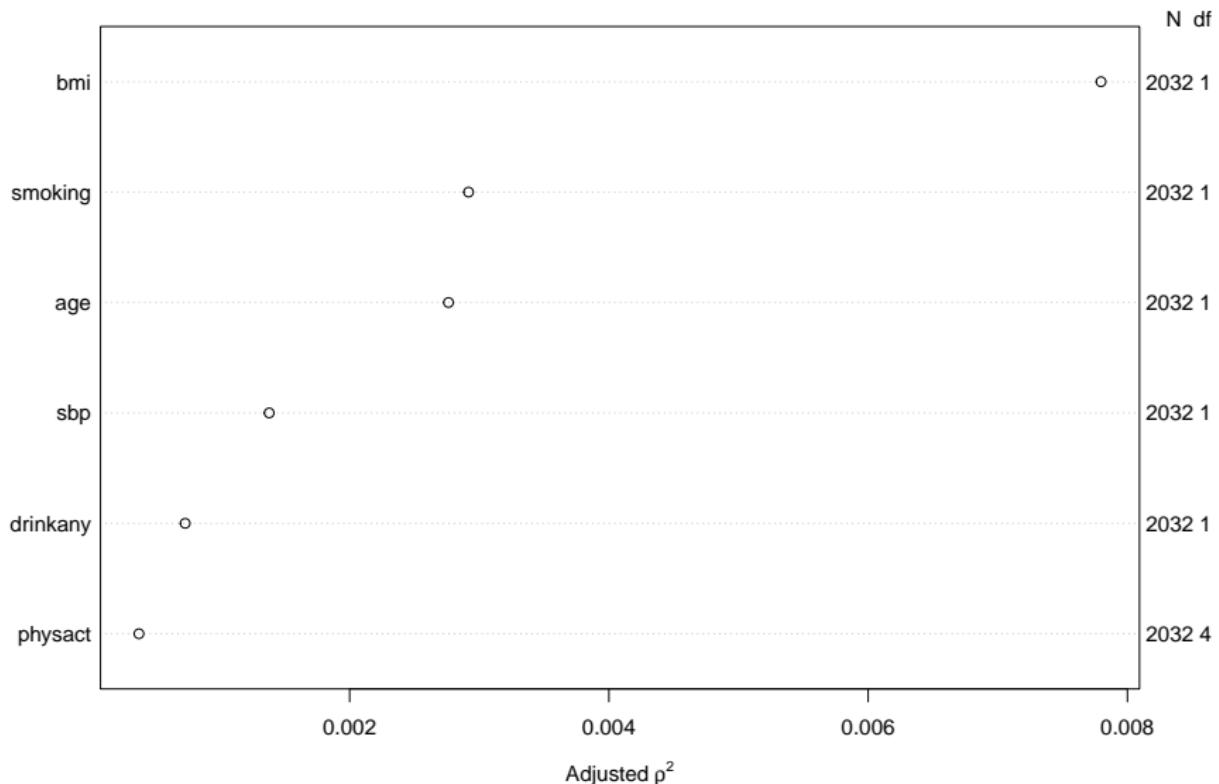
```
plot(spearman2(ldl ~ age + smoking + drinkany + sbp +  
    physact + bmi, data = hers2))
```

Plot's on the next page.

- Note the use of the simple imputation `hers2` data here. Why?

Spearman ρ^2 Plot Result

Spearman ρ^2 Response : ldl



Spending Degrees of Freedom

We're spending 9 degrees of freedom in our kitchen sink model. (We can verify this with anova or the plot.)

- Each quantitative main effect costs 1 df to estimate
- Each binary categorical variable also costs 1 df
- Multi-categorical variables with L levels cost L-1 df to estimate

Suppose we're willing to spend up to a total of **14** degrees of freedom (i.e. a combined 5 more on interaction terms and other ways to capture non-linearity.)

What should we choose?

What did we see in the Spearman ρ^2 Plot?

Group 1 (largest adjusted ρ^2)

- bmi, a quantitative predictor, is furthest to the right

Group 2 (next largest)

- smoking, a binary predictor, is next, followed closely by
- age, a quantitative predictor

Other predictors (rest of the group)

- sbp, quantitative
- drinkany, binary
- physact, multi-categorical (5 levels)

Impact of Adding Non-Linear Terms on Spent DF

What happens when we add a non-linear term?

- Adding a polynomial of degree D costs D degrees of freedom.
 - So a polynomial of degree 2 (quadratic) costs 2 df, or 1 more than the main effect alone.
- Adding a restricted cubic spline with K knots costs K-1 df.
 - So adding a rcs with 4 knots uses 3 df, or 2 more than the main effect.
 - We restrict ourselves to considering splines with 3, 4, or 5 knots.
- Adding an interaction (product term) depends on the main effects of the predictors we are interacting
 - If the product term's predictors have df1 and df2 degrees of freedom, product term adds $df1 \times df2$ degrees of freedom.
 - An interaction of a binary and quantitative variable adds $1 \times 1 = 1$ additional degree of freedom to the main effects model.
 - When we use a quantitative variable in a spline and interaction, we'll do the interaction on the main effect, not the spline.

Model we'll fit with ols

Fitting a model to predict ldl using

- bmi with a restricted cubic spline, 5 knots
- age with a quadratic polynomial
- sbp as a linear term
- drinkany indicator
- physact factor
- smoking indicator and its interaction with the main effect of bmi

We can fit this to the data

- restricted to complete cases (hers1, effectively)
- after simple imputation (hers2)
- after our multiple imputation (fit3)

Using only the Complete Cases

Fitting the model to the complete cases

```
d <- datadist(hers1)
options(datadist = "d")

m1 <- ols(ldl ~ rcs(bmi, 5) + pol(age, 2) + sbp +
           drinkany + physact + smoking +
           smoking %ia% bmi, data = hers1,
           x = TRUE, y = TRUE)
```

where %ia% identifies the linear interaction alone.

m1 results (screen 1/2)

m1

ldl	bmi	age	sbp	drinkany	physact	smoking
7	2	0	0	2	0	0

Linear Regression Model

```
ols(formula = ldl ~ rcs(bmi, 5) + pol(age, 2) + sbp + drinkany +
  physact + smoking + smoking %ia% bmi, data = hers1, x = TRUE,
  y = TRUE)
```

Obs	2021	Model Likelihood		Discrimination	
		LR	chi2	Ratio Test	Indexes
sigma	36.7430	d.f.	14	R2	0.026
d.f.	2006	Pr(> chi2)	0.0000	R2 adj	0.019
				g	6.629

Residuals

Min	1Q	Median	3Q	Max
-113.440	-24.519	-3.778	20.940	197.087

m1 results (screen 2/2)

m1

	Coef	S.E.	t	Pr(> t)
Intercept	121.6057	68.2000	1.78	0.0747
bmi	1.5687	1.0107	1.55	0.1208
bmi'	-8.6685	9.1577	-0.95	0.3440
bmi''	40.5712	37.4468	1.08	0.2787
bmi'''	-55.8872	44.5946	-1.25	0.2103
age	-0.5791	1.9657	-0.29	0.7683
age^2	0.0018	0.0149	0.12	0.9024
sbp	0.1221	0.0453	2.69	0.0072
drinkany=yes	-3.7427	1.6629	-2.25	0.0245
physact=much less active	-4.5660	3.8904	-1.17	0.2407
physact=much more active	-0.3291	2.7521	-0.12	0.9048
physact=somewhat less active	-0.0160	2.5270	-0.01	0.9950
physact=somewhat more active	3.7731	2.0293	1.86	0.0631
smoking=yes	-7.0832	12.0586	-0.59	0.5570
smoking=yes * bmi	0.4961	0.4391	1.13	0.2587

Fit Model after Single Imputation

Fitting the model after simple imputation

```
dd <- datadist(hers2)
options(datadist = "dd")

m2 <- ols(ldl ~ rcs(bmi, 5) + pol(age, 2) + sbp +
           drinkany + physact + smoking +
           smoking %ia% bmi, data = hers2,
           x = TRUE, y = TRUE)
```

where, again, %ia% identifies the linear interaction alone.

m2 results (screen 1/2)

m2

Linear Regression Model

```
ols(formula = ldl ~ rcs(bmi, 5) + pol(age, 2) + sbp + drinkany +
    physact + smoking + smoking %ia% bmi, data = hers2, x = TRUE,
    y = TRUE)
```

	Model	Likelihood Ratio Test	Discrimination
Obs	2032	LR chi2	53.14
sigma	36.6503	d.f.	14
d.f.	2017	Pr(> chi2)	0.0000
			R2 adj g
			0.026 0.019 6.631

Residuals

Min	1Q	Median	3Q	Max
-113.379	-24.326	-3.835	20.832	197.097

m2 results (screen 2/2)

m2

	Coef	S.E.	t	Pr(> t)
Intercept	120.2662	67.6113	1.78	0.0754
bmi	1.5508	1.0071	1.54	0.1237
bmi'	-8.4486	9.0978	-0.93	0.3532
bmi''	39.6413	37.1378	1.07	0.2859
bmi'''	-54.8924	44.2677	-1.24	0.2151
age	-0.5249	1.9490	-0.27	0.7877
age^2	0.0014	0.0148	0.10	0.9233
sbp	0.1209	0.0451	2.68	0.0074
drinkany=yes	-3.7023	1.6544	-2.24	0.0253
physact=much less active	-4.7408	3.8621	-1.23	0.2198
physact=much more active	-0.2635	2.7391	-0.10	0.9234
physact=somewhat less active	0.0130	2.5101	0.01	0.9959
physact=somewhat more active	3.8031	2.0193	1.88	0.0598
smoking=yes	-6.8961	12.0196	-0.57	0.5662
smoking=yes * bmi	0.4892	0.4375	1.12	0.2636

ANOVA results for m2 from ols

anova(m2)

Analysis of Variance		Response: ldl				
Factor		d.f.	Partial SS	MS	F	P
bmi	(Factor+Higher Order Factors)	5	2.758824e+04	5517.64861	4.11	0.0010
All Interactions		1	1.679813e+03	1679.81344	1.25	0.2636
Nonlinear		3	9.735452e+03	3245.15068	2.42	0.0647
age		2	9.175762e+03	4587.88077	3.42	0.0330
Nonlinear		1	1.244351e+01	12.44351	0.01	0.9233
sbp		1	9.657476e+03	9657.47569	7.19	0.0074
drinkany		1	6.726918e+03	6726.91809	5.01	0.0253
physact		4	9.709992e+03	2427.49791	1.81	0.1247
smoking	(Factor+Higher Order Factors)	2	1.085405e+04	5427.02463	4.04	0.0177
All Interactions		1	1.679813e+03	1679.81344	1.25	0.2636
smoking * bmi	(Factor+Higher Order Factors)	1	1.679813e+03	1679.81344	1.25	0.2636
TOTAL NONLINEAR		4	9.738807e+03	2434.70175	1.81	0.1237
TOTAL NONLINEAR + INTERACTION		5	1.171134e+04	2342.26845	1.74	0.1214
REGRESSION		14	7.178905e+04	5127.78931	3.82	<.0001
ERROR		2017	2.709327e+06	1343.24569		

Validation of summary statistics

```
set.seed(432); validate(m2)
```

	index.orig	training	test	optimism	index.corrected	n
R-square	0.0258	0.0307	0.0182	0.0125	0.0133	40
MSE	1333.3300	1323.5182	1343.7711	-20.2529	1353.5829	40
g	6.6306	7.1676	5.8338	1.3338	5.2968	40
Intercept	0.0000	0.0000	26.5316	-26.5316	26.5316	40
Slope	1.0000	1.0000	0.8174	0.1826	0.8174	40

summary(m2) results

summary(m2)

Effects	Response : ldl									
Factor	Low	High	Diff.	Effect	S.E.	Lower	0.95	Upper	0.95	
bmi	24.2	30.263	6.0625	5.1862	2.2217	0.82921	9.54330			
age	62.0	72.000	10.0000	-3.3412	1.3450	-5.97890	-0.70357			
sbp	120.0	145.000	25.0000	3.0218	1.1270	0.81165	5.23190			
drinkany - yes:no	1.0	2.000	NA	-3.7023	1.6544	-6.94690	-0.45779			
physact - about as active:somewhat more active	5.0	1.000	NA	-3.8031	2.0193	-7.76310	0.15695			
physact - much less active:somewhat more active	5.0	2.000	NA	-8.5439	3.9035	-16.19900	-0.88862			
physact - much more active:somewhat more active	5.0	3.000	NA	-4.0666	2.7125	-9.38630	1.25310			
physact - somewhat less active:somewhat more active	5.0	4.000	NA	-3.7901	2.5633	-8.81720	1.23690			
smoking - yes:no	1.0	2.000	NA	6.2635	2.4009	1.55500	10.97200			

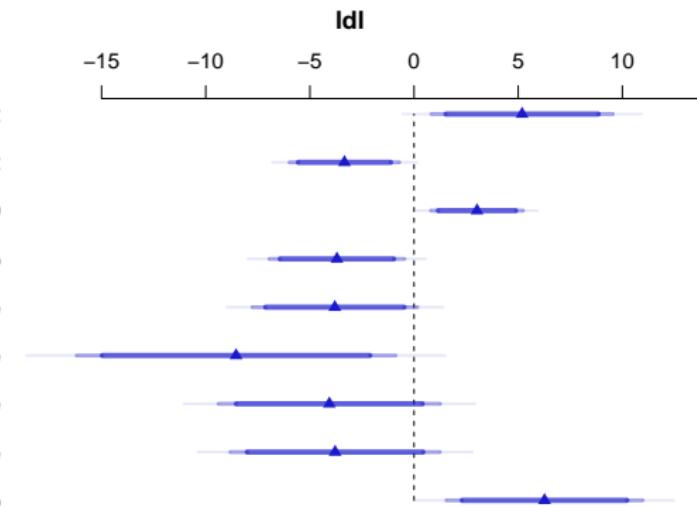
Adjusted to: bmi=26.9 smoking=no

- Of course, these should really be plotted...

Effect Size Plot for m2

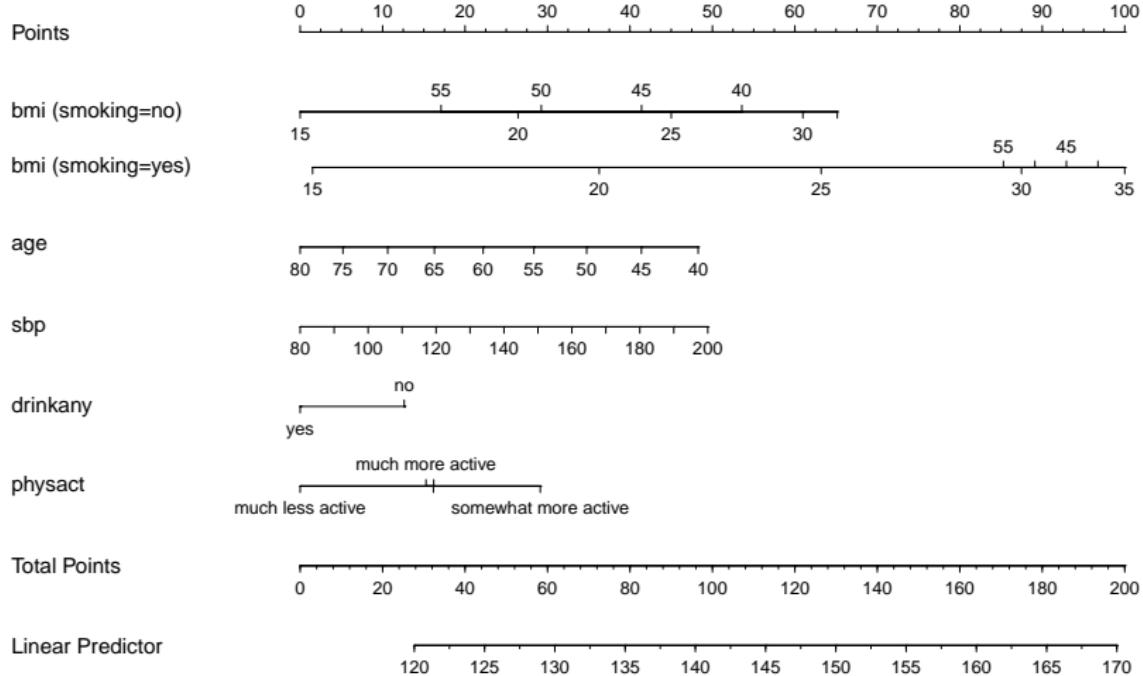
```
plot(summary(m2))
```

bmi – 30.2625:24.2
age – 72:62
sbp – 145:120
drinkany – yes:no
physact – about as active:somewhat more active
physact – much less active:somewhat more active
physact – much more active:somewhat more active
physact – somewhat less active:somewhat more active
smoking – yes:no



Adjusted to:bmi=26.9 smoking=no

Nomogram for m2



Making Predictions for an Individual

Suppose now that we want to use R to get a prediction for a new individual subject with `bmi = 30`, `age = 50`, `smoking = yes` and `physact = about as active`, `drinkany = yes` and `sbp` of 150.

```
predict(m2, expand.grid(bmi = 30, age = 50, smoking = "yes",
                         physact = "about as active",
                         drinkany = "yes", sbp = 150),
        conf.int = 0.95, conf.type = "individual")
```

\$linear.predictors	\$lower	\$upper
160.9399	88.48615	233.3936

Making Predictions for a Long-Run Mean

The other kind of prediction we might wish to make is for the mean of a series of subjects whose `bmi = 30`, `age = 50`, `smoking = yes` and `physact = about as active`, `drinkany = yes` and `sbp` of 150.

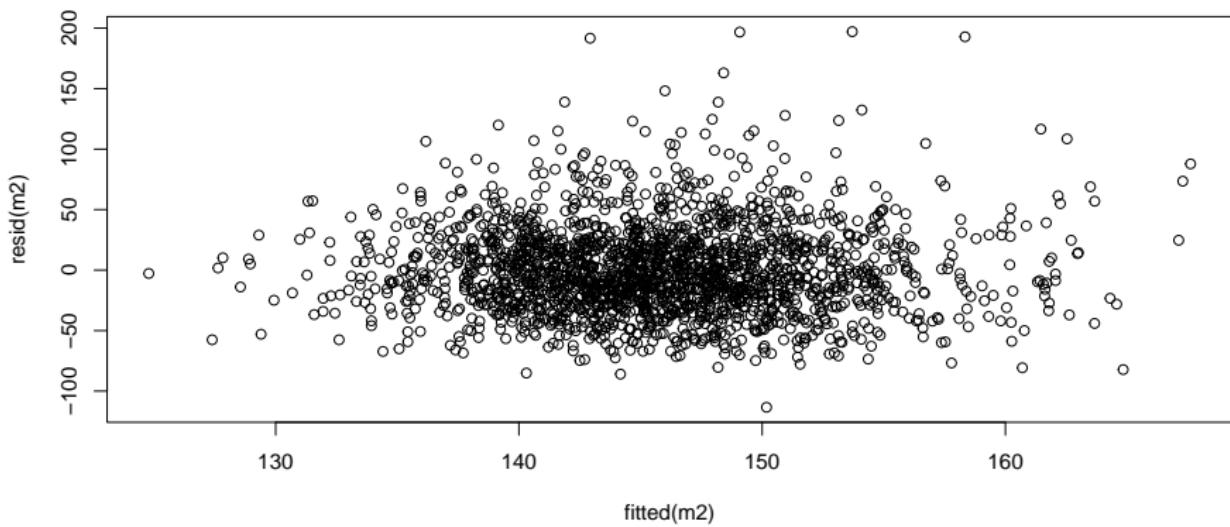
```
predict(m2, expand.grid(bmi = 30, age = 50, smoking = "yes",
                         physact = "about as active",
                         drinkany = "yes", sbp = 150),
        conf.int = 0.95, conf.type = "mean")
```

\$linear.predictors	\$lower	\$upper
160.9399	151.8119	170.0679

Of course, the confidence interval will always be narrower than the prediction interval given the same predictor values.

Residuals vs. Fitted Values?

```
plot(resid(m2) ~ fitted(m2))
```



Influential Points?

```
which.influence(m2, cutoff = 0.4)
```

```
$Intercept
```

```
[1] 1135
```

```
$age
```

```
[1] 1135
```

```
$smoking
```

```
[1] 132
```

```
$`smoking * bmi`
```

```
[1] 132
```

Using Multiple Imputation

Fitting the Model using Multiple Imputation

What do we have now?

- An imputation model fit3

```
fit3 <- aregImpute(~ ldl + age + smoking + drinkany + sbp +  
                    physact + bmi, nk = c(0, 3:5), tlinear = FALSE,  
                    data = hers1, B = 10, n.impute = 20, x = TRUE)
```

- A prediction model (from m1 or m2)

```
ols(ldl ~ rcs(bmi, 5) + pol(age, 2) + sbp +  
    drinkany + physact + smoking + smoking %ia% bmi,  
    x = TRUE, y = TRUE)
```

Now we put them together with the `fit.mult.impute` function...

Linear Regression & Imputation Model

```
m3imp <-  
  fit.mult.impute(ldl ~ rcs(bmi, 5) + pol(age, 2) + sbp +  
    drinkany + physact + smoking +  
    smoking %ia% bmi,  
    fitter = ols, xtrans = fit3,  
    data = hers1, pr = FALSE)
```

- When you run this without the `pr = FALSE` it generates considerable output related to the imputations, which we won't use today.
- Let's look at the rest of the output this yields...

m3imp results (screen 1/2)

m3imp

Linear Regression Model

```
fit.mult.impute(formula = ldl ~ rcs(bmi, 5) + pol(age, 2) + sbp +
  drinkany + physact + smoking + smoking %ia% bmi, fitter = ols,
  xtrans = fit3, data = hers1, pr = FALSE)
```

	Model Likelihood	Discrimination			
	Ratio Test	Indexes			
Obs	2032	LR chi2	52.74	R2	0.026
sigma	36.7331	d.f.	14	R2 adj	0.019
d.f.	2017	Pr(> chi2)	0.0000	g	6.621

Residuals

Min	1Q	Median	3Q	Max
-113.345	-24.510	-3.803	20.777	197.295

m3imp results (screen 2/2)

m3imp

	Coef	S.E.	t	Pr(> t)
Intercept	119.8951	67.8409	1.77	0.0773
bmi	1.5436	1.0097	1.53	0.1265
bmi'	-8.3664	9.1409	-0.92	0.3602
bmi''	39.2149	37.3458	1.05	0.2938
bmi'''	-54.2873	44.5323	-1.22	0.2230
age	-0.5002	1.9555	-0.26	0.7981
age^2	0.0012	0.0148	0.08	0.9351
sbp	0.1198	0.0454	2.64	0.0083
drinkany=yes	-3.7196	1.6613	-2.24	0.0253
physact=much less active	-4.7109	3.8716	-1.22	0.2238
physact=much more active	-0.2328	2.7512	-0.08	0.9326
physact=somewhat less active	-0.0417	2.5246	-0.02	0.9868
physact=somewhat more active	3.8197	2.0286	1.88	0.0599
smoking=yes	-6.8967	12.0503	-0.57	0.5672
smoking=yes * bmi	0.4866	0.4389	1.11	0.2677

ANOVA results for m3imp

```
anova(m3imp)
```

Analysis of Variance		Response: ldl				
Factor		d.f.	Partial SS	MS	F	P
bmi	(Factor+Higher Order Factors)	5	2.728300e+04	5456.600791	4.04	0.0012
All Interactions		1	1.658459e+03	1658.458931	1.23	0.2677
Nonlinear		3	9.585703e+03	3195.234412	2.37	0.0690
age		2	9.320445e+03	4660.222299	3.45	0.0318
Nonlinear		1	8.950493e+00	8.950493	0.01	0.9351
sbp		1	9.407603e+03	9407.602954	6.97	0.0083
drinkany		1	6.763854e+03	6763.853503	5.01	0.0253
physact		4	9.698175e+03	2424.543639	1.80	0.1268
smoking	(Factor+Higher Order Factors)	2	1.031090e+04	5155.452328	3.82	0.0221
All Interactions		1	1.658459e+03	1658.458931	1.23	0.2677
smoking * bmi	(Factor+Higher Order Factors)	1	1.658459e+03	1658.458931	1.23	0.2677
TOTAL NONLINEAR		4	9.587178e+03	2396.794504	1.78	0.1309
TOTAL NONLINEAR + INTERACTION		5	1.152744e+04	2305.487432	1.71	0.1293
REGRESSION		14	7.030149e+04	5021.535034	3.72	<.0001
ERROR		2017	2.721574e+06	1349.317884		

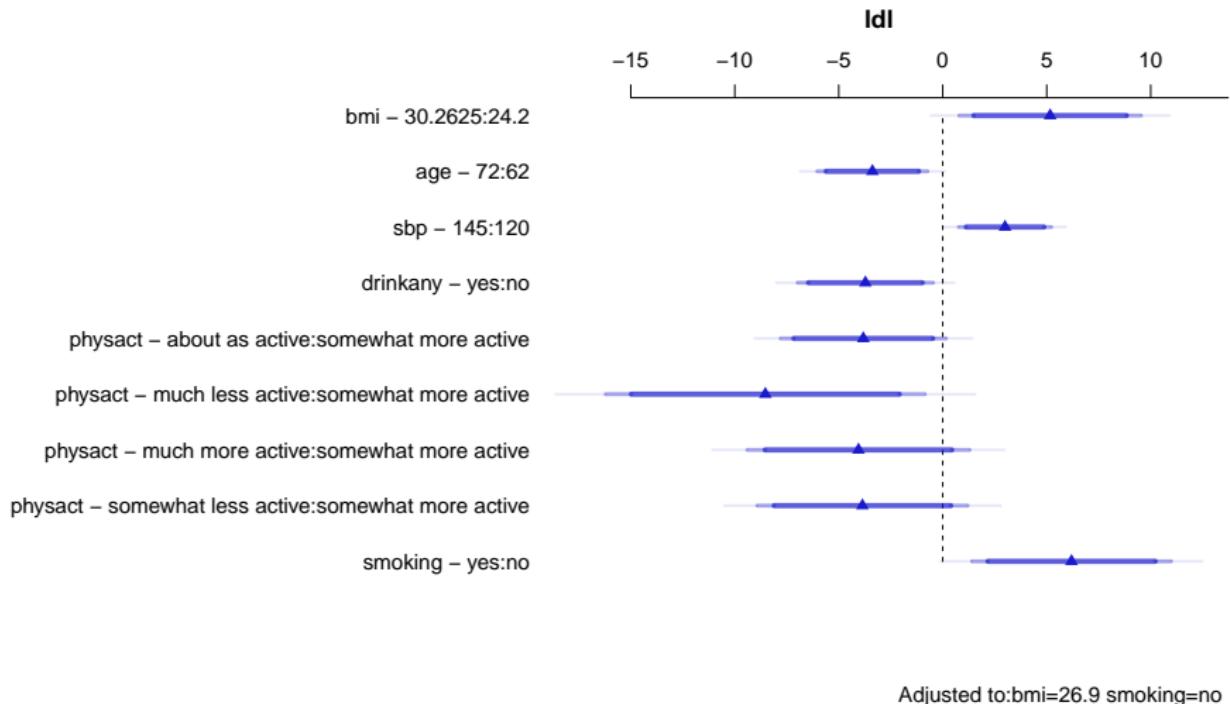
Summary of Effect Estimates for m3imp

```
summary(m3imp)
```

Effects	Response : ldl						
Factor	Low	High	Diff.	Effect	S.E.	Lower 0.95	Upper 0.95
bmi	24.2	30.263	6.0625	5.1643	2.2300	0.79099	9.53750
age	62.0	72.000	10.0000	-3.3824	1.3518	-6.03340	-0.73144
sbp	120.0	145.000	25.0000	2.9955	1.1345	0.77068	5.22040
drinkany - yes:no	1.0	2.000	NA	-3.7196	1.6613	-6.97780	-0.46150
physact - about as active:somewhat more active	5.0	1.000	NA	-3.8197	2.0286	-7.79800	0.15861
physact - much less active:somewhat more active	5.0	2.000	NA	-8.5306	3.9152	-16.20900	-0.85228
physact - much more active:somewhat more active	5.0	3.000	NA	-4.0525	2.7260	-9.39850	1.29350
physact - somewhat less active:somewhat more active	5.0	4.000	NA	-3.8614	2.5796	-8.92030	1.19760
smoking - yes:no	1.0	2.000	NA	6.1923	2.4427	1.40190	10.98300

Adjusted to: bmi=26.9 smoking=no

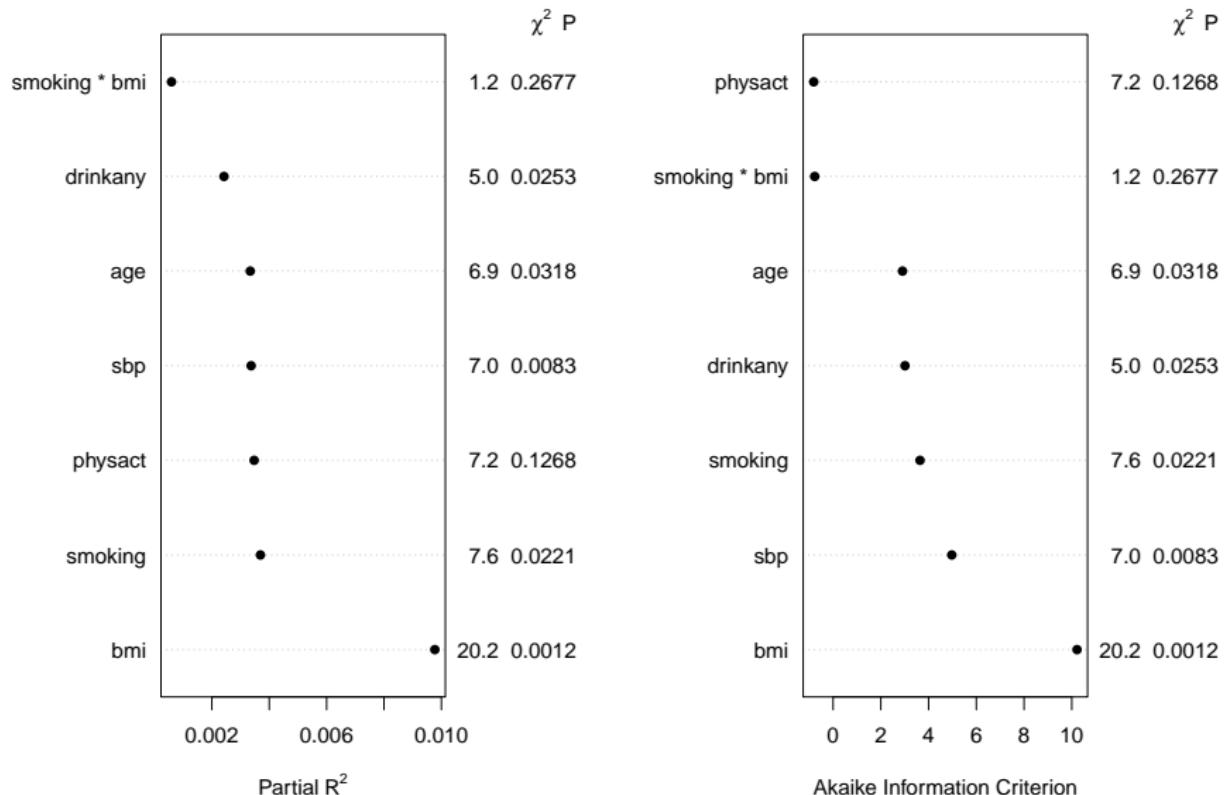
plot(summary(m3imp))



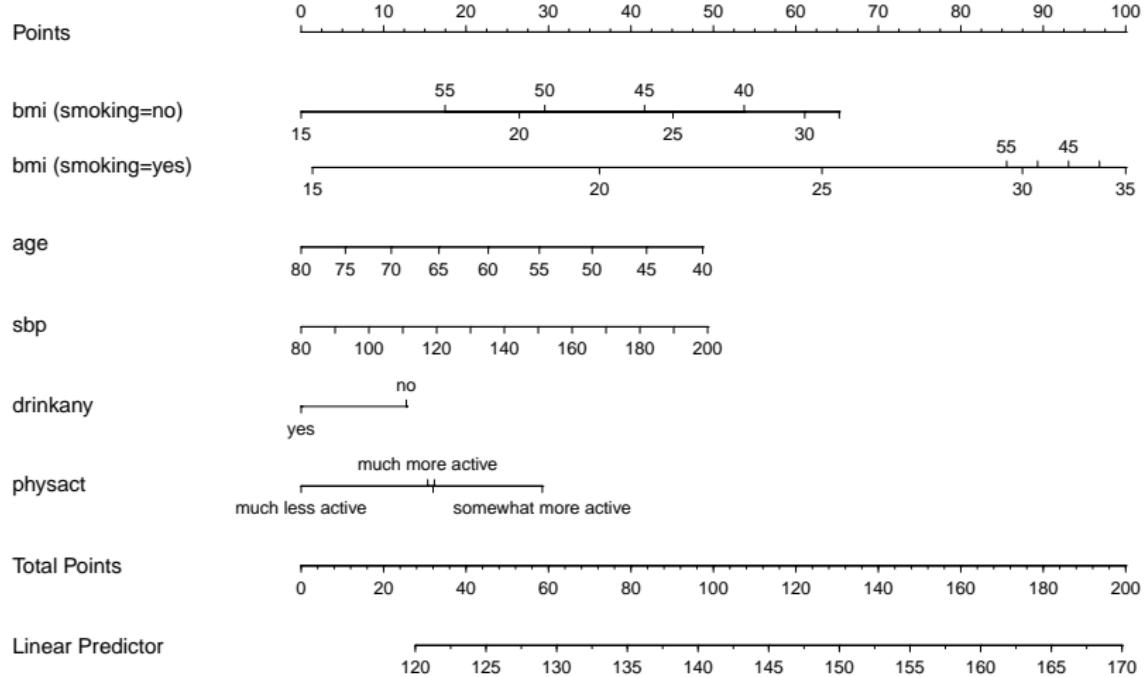
Evaluation via Partial R² and AIC (code)

```
par(mfrow = c(1,2))
plot(anova(m3imp), what="partial R2")
plot(anova(m3imp), what="aic")
par(mfrow = c(1,1))
```

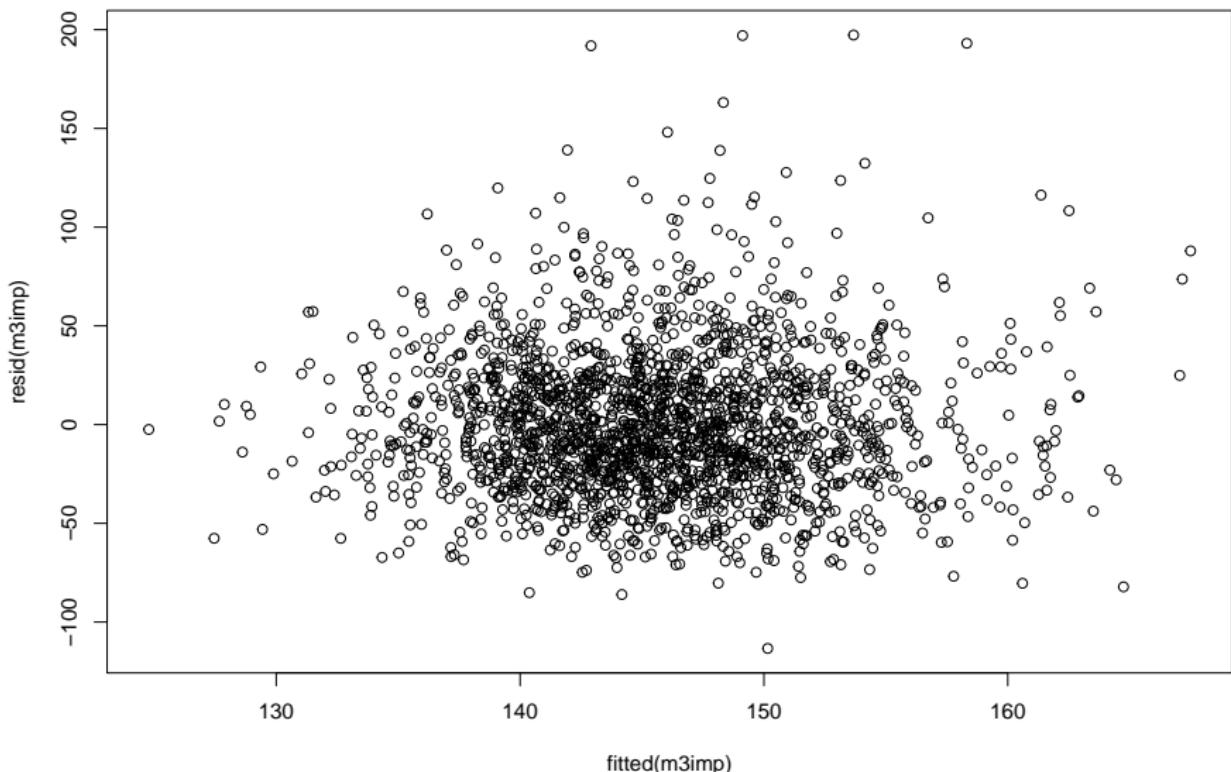
Evaluation via Partial R² and AIC (result)



```
plot(nomogram(m3imp))
```



```
plot(resid(m3imp) ~ fitted(m3imp))
```



Other Things I Might Need after aregImpute?

- How can I estimate the AIC (and BIC) of a model fit with `fit.mult.impute`?

`glance` won't work with an `ols` fit, but we can just use...

```
AIC(m3imp)
```

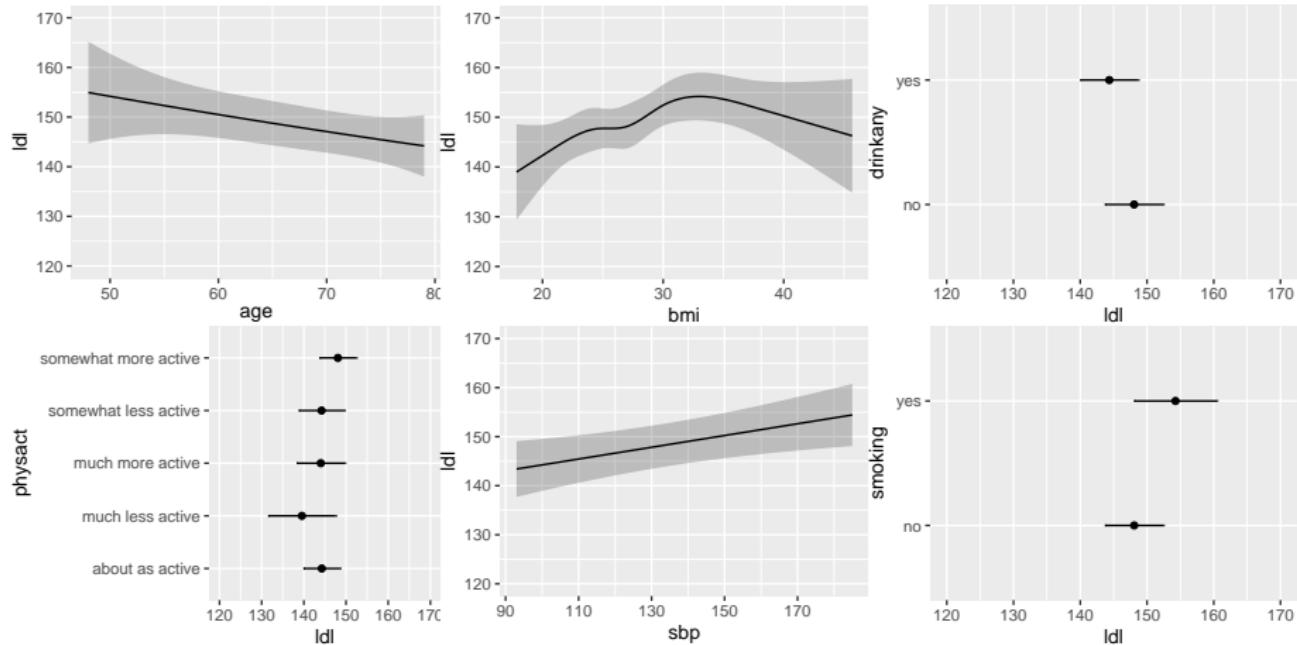
```
  d.f.  
20425.29
```

```
BIC(m3imp)
```

```
  d.f.  
20515.16
```

Can I run ggplot(Predict())?)?

```
ggplot(Predict(m3imp))
```



Pull out one imputation from aregImpute?

- How can I pull a single one (say, the fifth) of the imputations from aregImpute out?

Remember that fit3 was our imputation model here, build on the hers1 data, which keeps its subject identifiers in the subject column.

```
imputed_5 <-  
  impute.transcan(fit3, data = hers1, imputation = 5,  
                  list.out = T, pr = F, check = F)  
  
imputed_df5 <- as.data.frame(do.call(cbind, imputed_5))  
  
fifth_imp <-  
  bind_cols(subject = hers1$subject, imputed_df5) %>%  
  type.convert() %>% tibble()
```

Warning in type.convert.default(x[[i]], ...): 'as.is' should
be specified by the caller; using TRUE

Our fifth_imp tibble

fifth_imp

```
# A tibble: 2,032 x 8
  subject    ldl    age smoking drinkany    sbp physact    bmi
  <int>   <dbl>  <int>   <chr>      <int>  <int>   <chr>   <dbl>
1       1    122.     70 no          1    138 much mo~  23.7
2       2    242.     62 no          1    118 much le~  28.6
3       4    116.     64 yes         2    152 much le~  24.4
4       5    151.     65 no          1    175 somewha~ 21.9
5       6    138.     68 no          2    174 about a~  29.0
6       8    121.     69 no          1    178 much mo~  23.2
7       9    133      61 no          2    162 about a~  30.3
8      10    220      62 yes        2    111 somewha~ 45.7
9      11    173.     72 no          1    122 about a~  22.2
10     12    124.     73 no          1    158 somewha~ 25.3
# ... with 2,022 more rows
```

Create Residual Plots for this imputation?

```
model_for_resid_plots <-  
  lm(ldl ~ rcs(bmi, 5) + pol(age, 2) + sbp +  
      drinkany + physact + smoking +  
      smoking %ia% bmi, data = fifth_imp)
```

We can look at this model with `glance` or `tidy` to see that it gives similar results to what we see across the multiple imputations.

```
broom::glance(model_for_resid_plots) %>%  
  select(r.squared, AIC, BIC, nobs, df, df.residual) %>%  
  kable(digits = 3)
```

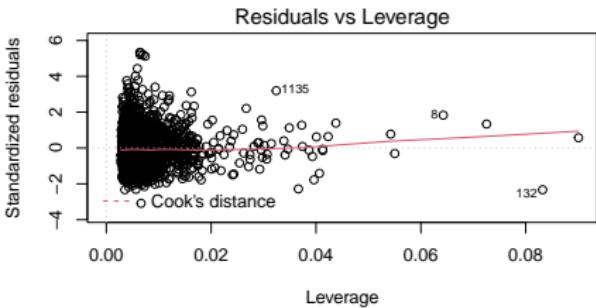
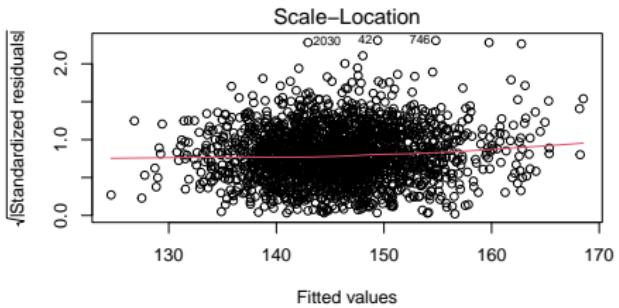
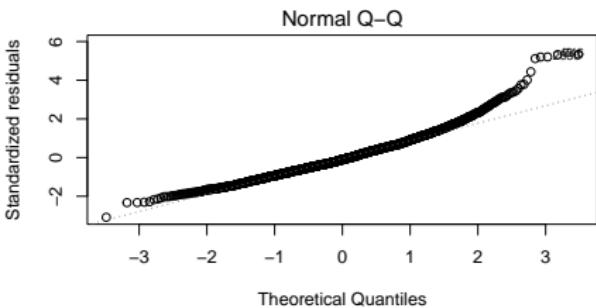
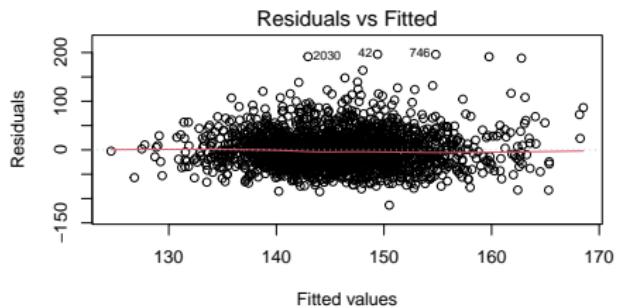
r.squared	AIC	BIC	nobs	df	df.residual
0.027	20451.36	20541.23	2032	14	2017

What else can we do?

We can plot residuals for the model fit to this single imputation, as shown on the next slide.

```
par(mfrow = c(2,2))
plot(model_for_resid_plots)
par(mfrow = c(1,1))
```

Residual Plots for Fifth Imputation



Next Step

Can we do all of this for a logistic regression model?

432 Class 09 Slides

thomaselove.github.io/432

2022-02-08

Today's Agenda

- Data from the Heart and Estrogen/Progestin Study
- Using ols to fit linear regression models in the presence of missing values
- Using aregImpute to facilitate principled multiple imputation when fitting regressions
- Developing detailed regression results under a variety of imputation plans

Setup

```
library(magrittr); library(janitor)
library(here); library(knitr)
library(naniar); library(simputation)

library(rms)
library(tidyverse)
```

Today's Data

Heart and Estrogen/Progestin Study (HERS)

- Clinical trial of hormone therapy for the prevention of recurrent heart attacks and deaths among 2763 post-menopausal women with existing coronary heart disease (see Hulley et al 1998 and many subsequent references, including Vittinghoff, Chapter 4.)
- We're excluding the women in the trial with a diabetes diagnosis.

```
hers_raw <- read_csv(here("data/hersdata.csv")) %>%
  clean_names()

hers1 <- hers_raw %>%
  filter(diabetes == "no") %>%
  select(subject, ldl, ht, age, smoking, drinkany, sbp,
         physact, bmi, diabetes)
```

The Codebook (n = 2032)

Variable	Description
subject	subject code
HT	factor: hormone therapy or placebo
diabetes	yes or no (all are no in our sample)
ldl	LDL cholesterol in mg/dl
age	age in years
smoking	yes or no
drinkany	yes or no
sbp	systolic BP in mm Hg
physact	5-level factor, details next slide
bmi	body-mass index in kg/m ²

Goal Predict ldl using age, smoking, drinkany, sbp, physact and bmi, across both HT levels but restricted to women without diabetes.

The physact variable

```
hers1 %>% count(physact)
```

```
# A tibble: 5 x 2
```

	physact	n
	<chr>	<int>
1	about as active	674
2	much less active	107
3	much more active	252
4	somewhat less active	322
5	somewhat more active	677

Comparison is to activity levels for these women just before menopause.

Any missing data?

```
miss_var_summary(hers1)
```

```
# A tibble: 10 x 3
  variable n_miss pct_miss
  <chr>     <int>    <dbl>
1 ldl         7     0.344
2 drinkany    2     0.0984
3 bmi         2     0.0984
4 subject     0      0
5 ht          0      0
6 age         0      0
7 smoking     0      0
8 sbp          0      0
9 physact     0      0
10 diabetes   0      0
```

Single Imputation for drinkany, bmi and ldl

Since drinkany is a factor, we have to do some extra work to impute.

```
set.seed(432092)
```

```
hers2 <- hers1 %>%
  mutate(drinkany_n =
    ifelse(drinkany == "yes", 1, 0)) %>%
  impute_pmm(drinkany_n ~ age + smoking) %>%
  mutate(drinkany =
    ifelse(drinkany_n == 1, "yes", "no")) %>%
  impute_rlm(bmi ~ age + smoking + sbp) %>%
  impute_rlm(ldl ~ age + smoking + sbp + bmi)
```

Now, check missingness...

```
miss_var_summary(hers2)
```

```
# A tibble: 11 x 3
  variable    n_miss pct_miss
  <chr>        <int>     <dbl>
1 subject         0         0
2 ldl             0         0
3 ht              0         0
4 age             0         0
5 smoking         0         0
6 drinkany        0         0
7 sbp             0         0
8 physact         0         0
9 bmi             0         0
10 diabetes        0         0
11 drinkany_n      0         0
```

Multiple Imputation using aregImpute from Hmisc

Model to predict all missing values of any variables, using additive regression bootstrapping and predictive mean matching.

Steps are:

- ① aregImpute draws a sample with replacement from the observations where the target variable is observed, not missing.
- ② It then fits a flexible additive model to predict this target variable while finding the optimum transformation of it.
- ③ It then uses this fitted flexible model to predict the target variable in all of the original observations.
- ④ Finally, it imputes each missing value of the target variable with the observed value whose predicted transformed value is closest to the predicted transformed value of the missing value.

Fitting a Multiple Imputation Model

```
set.seed(4320132)
dd <- datadist(hers1)
options(datadist = "dd")
fit3 <- aregImpute(~ ldl + age + smoking + drinkany +
                     sbp + physact + bmi,
                     nk = c(0, 3:5), tlinear = FALSE,
                     data = hers1, B = 10, n.impute = 20)
```

Iteration 1 Iteration 2 Iteration 3 Iteration 4 Iteration 5 Iteration 6 Iteration 7 Iteration 8 Iteration 9 Iteration 10 Iteration 11 Iteration 12 Iteration 13 Iteration 14 Iteration 15 Iteration 16 Iteration 17 Iteration 18 Iteration 19 Iteration 20

Multiple Imputation using aregImpute from Hmisc

aregImpute requires specifications of all variables, and several other details:

- `n.impute` = number of imputations, we'll run 20
- `nk` = number of knots to describe level of complexity, with our choice `nk = c(0, 3:5)` we'll fit both linear models and models with restricted cubic splines with 3, 4, and 5 knots
- `tlinear` = `FALSE` allows the target variable to have a non-linear transformation when `nk` is 3 or more
- `B` = 10 specifies 10 bootstrap samples will be used
- `data` specifies the source of the variables

aregImpute Imputation Results (1 of 4)

fit3

Multiple Imputation using Bootstrap and PMM

```
aregImpute(formula = ~ldl + age + smoking + drinkany + sbp +
physact + bmi, data = hers1, n.impute = 20, nk = c(0, 3:5),
tlinear = FALSE, B = 10)
```

n: 2032 p: 7 Imputations: 20 nk: 0

Number of NAs:

ldl	age	smoking	drinkany	sbp	physact	bmi
7	0	0	2	0	0	2

fit3 Imputation Results (2 of 4)

R-squares for Predicting Non-Missing Values for Each Variable Using Last Imputations of Predictors

ldl	drinkany	bmi
0.041	0.014	0.109

fit3 Imputation Results (3 of 4)

Resampling results for determining the complexity of imputation models

Variable being imputed: ldl

Bootstrap bias-corrected summaries:

Statistic	nk = 0	nk = 3	nk = 4	nk = 5
R^2	0.0139	0.0149	0.00776	0.0124
mean absolute error	28.3594	42.9139	44.09937	39.8266
median abs. error	22.8301	35.5441	38.85302	32.6386

10-fold cross-validated:

Statistic	nk = 0	nk = 3	nk = 4	nk = 5
R^2	0.0214	0.0180	0.01517	0.0191
mean absolute error	145.7176	43.5007	45.02428	44.2456
median abs. error	141.4238	36.4102	38.88053	37.3141

fit3 Imputation Results (4 of 4)

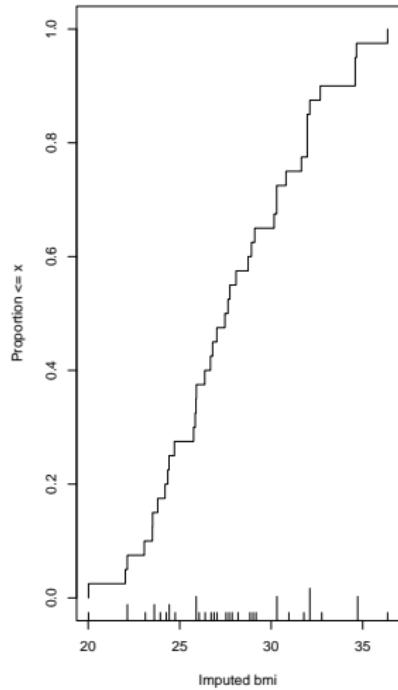
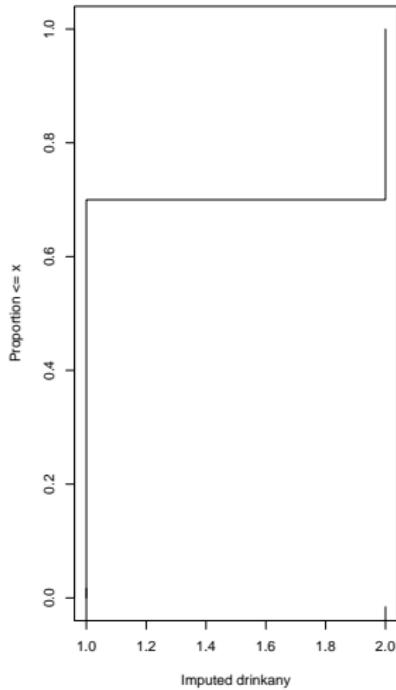
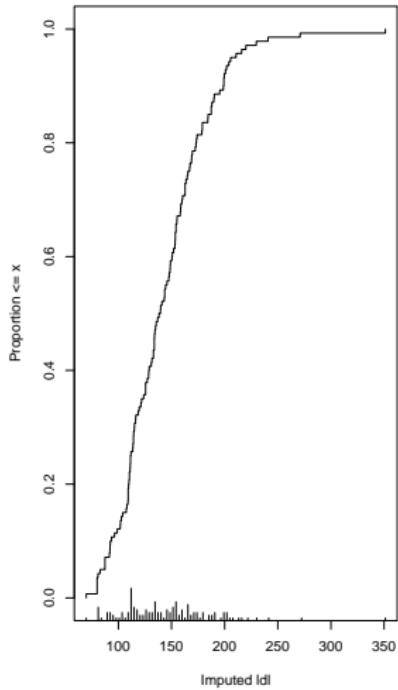
Variable being imputed: drinkany

		nk=0	nk=3	nk=4	nk=5
Bootstrap	R^2	0.0163	0.0113	0.0102	0.00986
10-fold cv	R^2	0.0205	0.0249	0.0163	0.01358
Bootstrap	mean error	0.4470	0.4568	0.4558	0.46624
10-fold cv	mean error	0.4450	0.4454	0.4476	0.44676
Bootstrap	median error	0.0000	0.0000	0.0000	0.00000
10-fold cv	median error	0.0000	0.0500	0.1000	0.00000

Variable being imputed: bmi

		nk=0	nk=3	nk=4	nk=5
Bootstrap	R^2	0.0845	0.0932	0.0946	0.0847
10-fold cv	R^2	0.0864	0.0903	0.0968	0.0899
Bootstrap	mean error	3.7829	4.8119	4.9226	5.1775
10-fold cv	mean error	27.6776	4.8359	4.9390	5.1136
Bootstrap	median error	2.9955	3.9704	3.9371	4.2634
10-fold cv	median error	27.0143	3.9894	3.9431	4.1876

A plot of the imputed values... (results)



A plot of the imputed values... (code)

```
par(mfrow = c(1,3))
plot(fit3)
par(mfrow = c(1,1))
```

- For `ldl`, we imputed most of the 7 missing subjects in most of the 20 imputation runs to values within a range of around 120 through 200, but occasionally, we imputed values that were substantially lower than 100.
- For `drinkany` we imputed about 70% no and 30% yes.
- For `bmi`, we imputed values ranging from about 23 to 27 in many cases, and up near 40 in other cases.
- This method never imputes a value for a variable that doesn't already exist in the data.

Kitchen Sink Model (Main Effects only)

```
mod_ks <- ols(ldl ~ age + smoking + drinkany + sbp +  
                 physact + bmi, data = hers2)  
anova(mod_ks)
```

Analysis of Variance Response: ldl

Factor	d.f.	Partial SS	MS	F	P
age	1	9330.911	9330.911	6.93	0.0085
smoking	1	8199.755	8199.755	6.09	0.0137
drinkany	1	6444.424	6444.424	4.79	0.0288
sbp	1	9274.287	9274.287	6.89	0.0087
physact	4	10874.528	2718.632	2.02	0.0891
bmi	1	15876.957	15876.957	11.80	0.0006
REGRESSION	9	60077.708	6675.301	4.96	<.0001
ERROR	2022	2721037.890	1345.716		

Spearman ρ^2 Plot

How should we prioritize the degrees of freedom we spend on non-linearity?

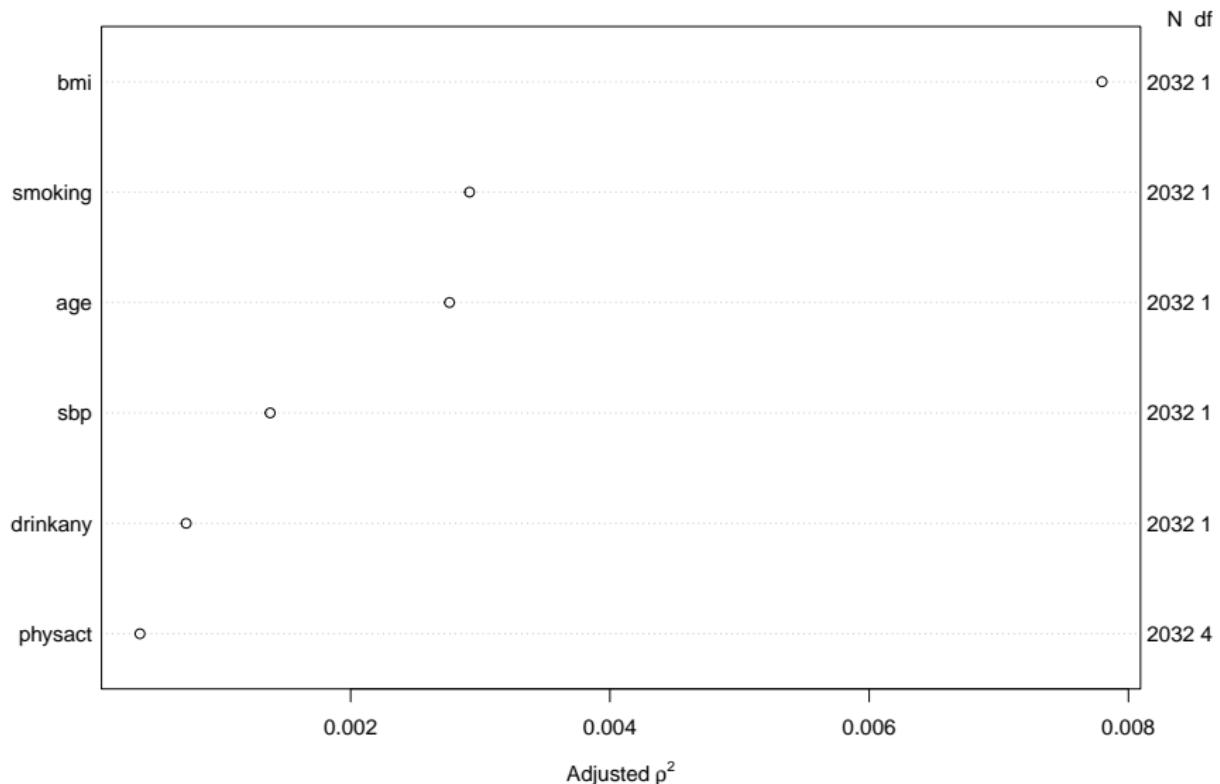
```
plot(spearman2(ldl ~ age + smoking + drinkany + sbp +  
    physact + bmi, data = hers2))
```

Plot's on the next page.

- Note the use of the simple imputation `hers2` data here. Why?

Spearman ρ^2 Plot Result

Spearman ρ^2 Response : ldl



Spending Degrees of Freedom

We're spending 9 degrees of freedom in our kitchen sink model. (We can verify this with anova or the plot.)

- Each quantitative main effect costs 1 df to estimate
- Each binary categorical variable also costs 1 df
- Multi-categorical variables with L levels cost L-1 df to estimate

Suppose we're willing to spend up to a total of **14** degrees of freedom (i.e. a combined 5 more on interaction terms and other ways to capture non-linearity.)

What should we choose?

What did we see in the Spearman ρ^2 Plot?

Group 1 (largest adjusted ρ^2)

- bmi, a quantitative predictor, is furthest to the right

Group 2 (next largest)

- smoking, a binary predictor, is next, followed closely by
- age, a quantitative predictor

Other predictors (rest of the group)

- sbp, quantitative
- drinkany, binary
- physact, multi-categorical (5 levels)

Impact of Adding Non-Linear Terms on Spent DF

What happens when we add a non-linear term?

- Adding a polynomial of degree D costs D degrees of freedom.
 - So a polynomial of degree 2 (quadratic) costs 2 df, or 1 more than the main effect alone.
- Adding a restricted cubic spline with K knots costs K-1 df.
 - So adding a spline with 4 knots uses 3 df, or 2 more than the main effect.
 - We restrict ourselves to considering splines with 3, 4, or 5 knots.
- Adding an interaction (product term) depends on the main effects of the predictors we are interacting
 - If the product term's predictors have df1 and df2 degrees of freedom, product term adds $df1 \times df2$ degrees of freedom.
 - An interaction of a binary and quantitative variable adds $1 \times 1 = 1$ additional degree of freedom to the main effects model.
 - When we use a quantitative variable in a spline and interaction, we'll do the interaction on the main effect, not the spline.

Model we'll fit with ols

Fitting a model to predict ldl using

- bmi with a restricted cubic spline, 5 knots
- age with a quadratic polynomial
- sbp as a linear term
- drinkany indicator
- physact factor
- smoking indicator and its interaction with the main effect of bmi

We can fit this to the data

- restricted to complete cases (hers1, effectively)
- after simple imputation (hers2)
- after our multiple imputation (fit3)

Using only the Complete Cases

Fitting the model to the complete cases

```
d <- datadist(hers1)
options(datadist = "d")

m1 <- ols(ldl ~ rcs(bmi, 5) + pol(age, 2) + sbp +
           drinkany + physact + smoking +
           smoking %ia% bmi, data = hers1,
           x = TRUE, y = TRUE)
```

where %ia% identifies the linear interaction alone.

m1 results (screen 1/2)

m1

ldl	bmi	age	sbp	drinkany	physact	smoking
7	2	0	0	2	0	0

Linear Regression Model

```
ols(formula = ldl ~ rcs(bmi, 5) + pol(age, 2) + sbp + drinkany +
  physact + smoking + smoking %ia% bmi, data = hers1, x = TRUE,
  y = TRUE)
```

Obs	2021	Model Likelihood		Discrimination	
		LR	chi2	Ratio Test	Indexes
sigma	36.7430	d.f.	14	R2	0.026
d.f.	2006	Pr(> chi2)	0.0000	R2 adj	0.019
				g	6.629

Residuals

Min	1Q	Median	3Q	Max
-113.440	-24.519	-3.778	20.940	197.087

m1 results (screen 2/2)

m1

	Coef	S.E.	t	Pr(> t)
Intercept	121.6057	68.2000	1.78	0.0747
bmi	1.5687	1.0107	1.55	0.1208
bmi'	-8.6685	9.1577	-0.95	0.3440
bmi''	40.5712	37.4468	1.08	0.2787
bmi'''	-55.8872	44.5946	-1.25	0.2103
age	-0.5791	1.9657	-0.29	0.7683
age^2	0.0018	0.0149	0.12	0.9024
sbp	0.1221	0.0453	2.69	0.0072
drinkany=yes	-3.7427	1.6629	-2.25	0.0245
physact=much less active	-4.5660	3.8904	-1.17	0.2407
physact=much more active	-0.3291	2.7521	-0.12	0.9048
physact=somewhat less active	-0.0160	2.5270	-0.01	0.9950
physact=somewhat more active	3.7731	2.0293	1.86	0.0631
smoking=yes	-7.0832	12.0586	-0.59	0.5570
smoking=yes * bmi	0.4961	0.4391	1.13	0.2587

Fit Model after Single Imputation

Fitting the model after simple imputation

```
dd <- datadist(hers2)
options(datadist = "dd")

m2 <- ols(ldl ~ rcs(bmi, 5) + pol(age, 2) + sbp +
           drinkany + physact + smoking +
           smoking %ia% bmi, data = hers2,
           x = TRUE, y = TRUE)
```

where, again, %ia% identifies the linear interaction alone.

m2 results (screen 1/2)

m2

Linear Regression Model

```
ols(formula = ldl ~ rcs(bmi, 5) + pol(age, 2) + sbp + drinkany +
    physact + smoking + smoking %ia% bmi, data = hers2, x = TRUE,
    y = TRUE)
```

	Model	Likelihood Ratio Test	Discrimination
Obs	2032	LR chi2	53.14
sigma	36.6503	d.f.	14
d.f.	2017	Pr(> chi2)	0.0000
			R2 adj g
			0.026 0.019 6.631

Residuals

Min	1Q	Median	3Q	Max
-113.379	-24.326	-3.835	20.832	197.097

m2 results (screen 2/2)

m2

	Coef	S.E.	t	Pr(> t)
Intercept	120.2662	67.6113	1.78	0.0754
bmi	1.5508	1.0071	1.54	0.1237
bmi'	-8.4486	9.0978	-0.93	0.3532
bmi''	39.6413	37.1378	1.07	0.2859
bmi'''	-54.8924	44.2677	-1.24	0.2151
age	-0.5249	1.9490	-0.27	0.7877
age^2	0.0014	0.0148	0.10	0.9233
sbp	0.1209	0.0451	2.68	0.0074
drinkany=yes	-3.7023	1.6544	-2.24	0.0253
physact=much less active	-4.7408	3.8621	-1.23	0.2198
physact=much more active	-0.2635	2.7391	-0.10	0.9234
physact=somewhat less active	0.0130	2.5101	0.01	0.9959
physact=somewhat more active	3.8031	2.0193	1.88	0.0598
smoking=yes	-6.8961	12.0196	-0.57	0.5662
smoking=yes * bmi	0.4892	0.4375	1.12	0.2636

ANOVA results for m2 from ols

anova(m2)

Analysis of Variance		Response: ldl				
Factor		d.f.	Partial SS	MS	F	P
bmi	(Factor+Higher Order Factors)	5	2.758824e+04	5517.64861	4.11	0.0010
All Interactions		1	1.679813e+03	1679.81344	1.25	0.2636
Nonlinear		3	9.735452e+03	3245.15068	2.42	0.0647
age		2	9.175762e+03	4587.88077	3.42	0.0330
Nonlinear		1	1.244351e+01	12.44351	0.01	0.9233
sbp		1	9.657476e+03	9657.47569	7.19	0.0074
drinkany		1	6.726918e+03	6726.91809	5.01	0.0253
physact		4	9.709992e+03	2427.49791	1.81	0.1247
smoking	(Factor+Higher Order Factors)	2	1.085405e+04	5427.02463	4.04	0.0177
All Interactions		1	1.679813e+03	1679.81344	1.25	0.2636
smoking * bmi	(Factor+Higher Order Factors)	1	1.679813e+03	1679.81344	1.25	0.2636
TOTAL NONLINEAR		4	9.738807e+03	2434.70175	1.81	0.1237
TOTAL NONLINEAR + INTERACTION		5	1.171134e+04	2342.26845	1.74	0.1214
REGRESSION		14	7.178905e+04	5127.78931	3.82	<.0001
ERROR		2017	2.709327e+06	1343.24569		

Validation of summary statistics

```
set.seed(432); validate(m2)
```

	index.orig	training	test	optimism	index.corrected	n
R-square	0.0258	0.0307	0.0182	0.0125	0.0133	40
MSE	1333.3300	1323.5182	1343.7711	-20.2529	1353.5829	40
g	6.6306	7.1676	5.8338	1.3338	5.2968	40
Intercept	0.0000	0.0000	26.5316	-26.5316	26.5316	40
Slope	1.0000	1.0000	0.8174	0.1826	0.8174	40

summary(m2) results

summary(m2)

Effects	Response : ldl									
Factor	Low	High	Diff.	Effect	S.E.	Lower	0.95	Upper	0.95	
bmi	24.2	30.263	6.0625	5.1862	2.2217	0.82921	9.54330			
age	62.0	72.000	10.0000	-3.3412	1.3450	-5.97890	-0.70357			
sbp	120.0	145.000	25.0000	3.0218	1.1270	0.81165	5.23190			
drinkany - yes:no	1.0	2.000	NA	-3.7023	1.6544	-6.94690	-0.45779			
physact - about as active:somewhat more active	5.0	1.000	NA	-3.8031	2.0193	-7.76310	0.15695			
physact - much less active:somewhat more active	5.0	2.000	NA	-8.5439	3.9035	-16.19900	-0.88862			
physact - much more active:somewhat more active	5.0	3.000	NA	-4.0666	2.7125	-9.38630	1.25310			
physact - somewhat less active:somewhat more active	5.0	4.000	NA	-3.7901	2.5633	-8.81720	1.23690			
smoking - yes:no	1.0	2.000	NA	6.2635	2.4009	1.55500	10.97200			

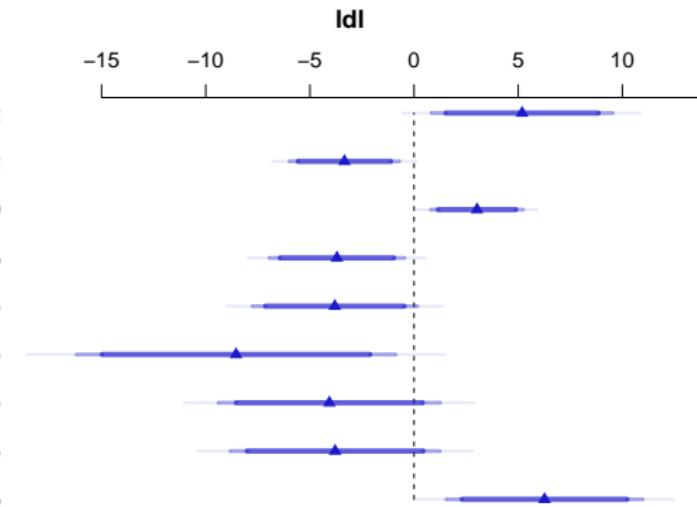
Adjusted to: bmi=26.9 smoking=no

- Of course, these should really be plotted...

Effect Size Plot for m2

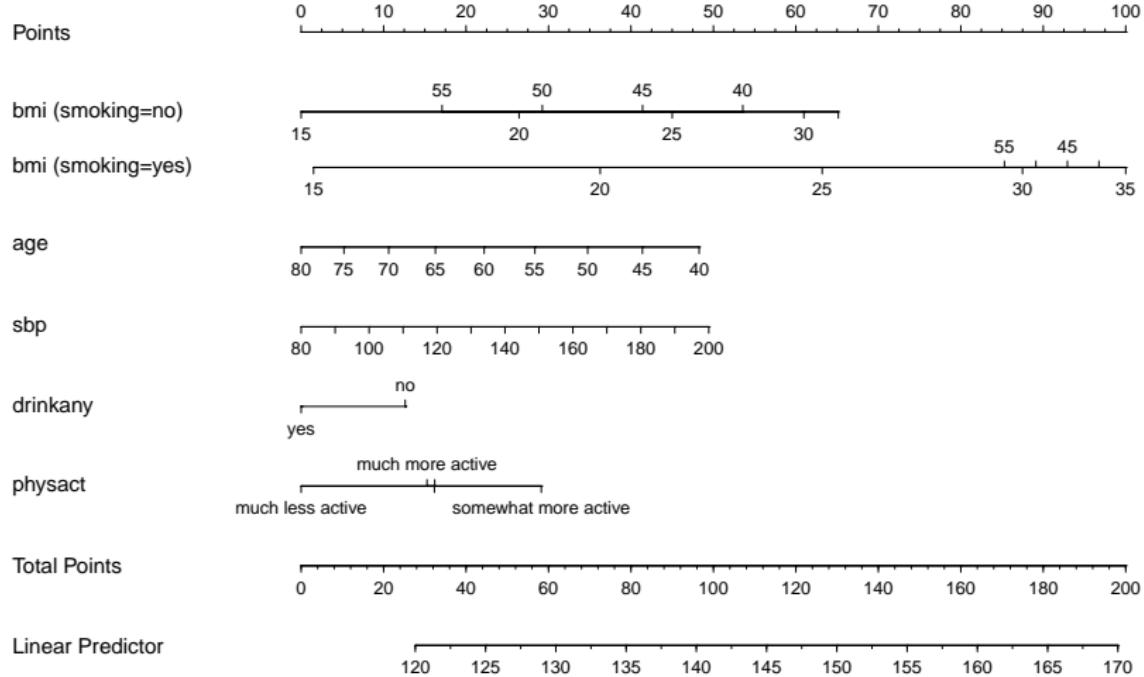
```
plot(summary(m2))
```

bmi – 30.2625:24.2
age – 72:62
sbp – 145:120
drinkany – yes:no
physact – about as active:somewhat more active
physact – much less active:somewhat more active
physact – much more active:somewhat more active
physact – somewhat less active:somewhat more active
smoking – yes:no



Adjusted to:bmi=26.9 smoking=no

Nomogram for m2



Making Predictions for an Individual

Suppose now that we want to use R to get a prediction for a new individual subject with `bmi = 30`, `age = 50`, `smoking = yes` and `physact = about as active`, `drinkany = yes` and `sbp` of 150.

```
predict(m2, expand.grid(bmi = 30, age = 50, smoking = "yes",
                         physact = "about as active",
                         drinkany = "yes", sbp = 150),
        conf.int = 0.95, conf.type = "individual")
```

\$linear.predictors	\$lower	\$upper
160.9399	88.48615	233.3936

Making Predictions for a Long-Run Mean

The other kind of prediction we might wish to make is for the mean of a series of subjects whose `bmi = 30`, `age = 50`, `smoking = yes` and `physact = about as active`, `drinkany = yes` and `sbp` of 150.

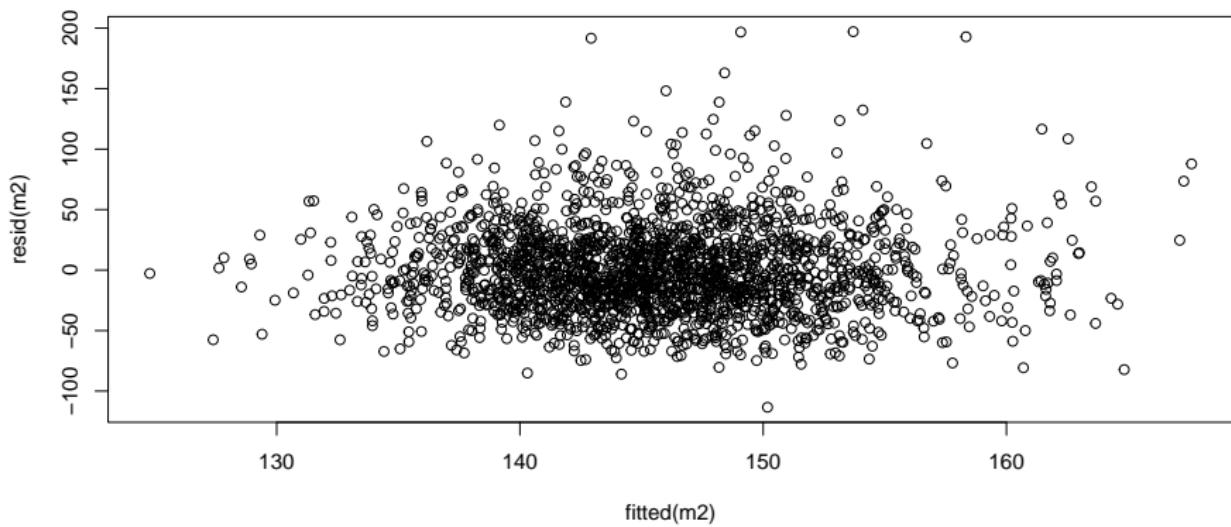
```
predict(m2, expand.grid(bmi = 30, age = 50, smoking = "yes",
                         physact = "about as active",
                         drinkany = "yes", sbp = 150),
        conf.int = 0.95, conf.type = "mean")
```

\$linear.predictors	\$lower	\$upper
160.9399	151.8119	170.0679

Of course, the confidence interval will always be narrower than the prediction interval given the same predictor values.

Residuals vs. Fitted Values?

```
plot(resid(m2) ~ fitted(m2))
```



Influential Points?

```
which.influence(m2, cutoff = 0.4)
```

```
$Intercept
```

```
[1] 1135
```

```
$age
```

```
[1] 1135
```

```
$smoking
```

```
[1] 132
```

```
$`smoking * bmi`
```

```
[1] 132
```

Using Multiple Imputation

Fitting the Model using Multiple Imputation

What do we have now?

- An imputation model fit3

```
fit3 <- aregImpute(~ ldl + age + smoking + drinkany + sbp +  
                    physact + bmi, nk = c(0, 3:5), tlinear = FALSE,  
                    data = hers1, B = 10, n.impute = 20, x = TRUE)
```

- A prediction model (from m1 or m2)

```
ols(ldl ~ rcs(bmi, 5) + pol(age, 2) + sbp +  
    drinkany + physact + smoking + smoking %ia% bmi,  
    x = TRUE, y = TRUE)
```

Now we put them together with the `fit.mult.impute` function...

Linear Regression & Imputation Model

```
m3imp <-  
  fit.mult.impute(ldl ~ rcs(bmi, 5) + pol(age, 2) + sbp +  
    drinkany + physact + smoking +  
    smoking %ia% bmi,  
    fitter = ols, xtrans = fit3,  
    data = hers1, pr = FALSE)
```

- When you run this without the `pr = FALSE` it generates considerable output related to the imputations, which we won't use today.
- Let's look at the rest of the output this yields...

m3imp results (screen 1/2)

m3imp

Linear Regression Model

```
fit.mult.impute(formula = ldl ~ rcs(bmi, 5) + pol(age, 2) + sbp +
  drinkany + physact + smoking + smoking %ia% bmi, fitter = ols,
  xtrans = fit3, data = hers1, pr = FALSE)
```

	Model Likelihood	Discrimination			
	Ratio Test	Indexes			
Obs	2032	LR chi2	52.74	R2	0.026
sigma	36.7331	d.f.	14	R2 adj	0.019
d.f.	2017	Pr(> chi2)	0.0000	g	6.621

Residuals

Min	1Q	Median	3Q	Max
-113.345	-24.510	-3.803	20.777	197.295

m3imp results (screen 2/2)

m3imp

	Coef	S.E.	t	Pr(> t)
Intercept	119.8951	67.8409	1.77	0.0773
bmi	1.5436	1.0097	1.53	0.1265
bmi'	-8.3664	9.1409	-0.92	0.3602
bmi''	39.2149	37.3458	1.05	0.2938
bmi'''	-54.2873	44.5323	-1.22	0.2230
age	-0.5002	1.9555	-0.26	0.7981
age^2	0.0012	0.0148	0.08	0.9351
sbp	0.1198	0.0454	2.64	0.0083
drinkany=yes	-3.7196	1.6613	-2.24	0.0253
physact=much less active	-4.7109	3.8716	-1.22	0.2238
physact=much more active	-0.2328	2.7512	-0.08	0.9326
physact=somewhat less active	-0.0417	2.5246	-0.02	0.9868
physact=somewhat more active	3.8197	2.0286	1.88	0.0599
smoking=yes	-6.8967	12.0503	-0.57	0.5672
smoking=yes * bmi	0.4866	0.4389	1.11	0.2677

ANOVA results for m3imp

```
anova(m3imp)
```

Analysis of Variance		Response: ldl				
Factor		d.f.	Partial SS	MS	F	P
bmi	(Factor+Higher Order Factors)	5	2.728300e+04	5456.600791	4.04	0.0012
All Interactions		1	1.658459e+03	1658.458931	1.23	0.2677
Nonlinear		3	9.585703e+03	3195.234412	2.37	0.0690
age		2	9.320445e+03	4660.222299	3.45	0.0318
Nonlinear		1	8.950493e+00	8.950493	0.01	0.9351
sbp		1	9.407603e+03	9407.602954	6.97	0.0083
drinkany		1	6.763854e+03	6763.853503	5.01	0.0253
physact		4	9.698175e+03	2424.543639	1.80	0.1268
smoking	(Factor+Higher Order Factors)	2	1.031090e+04	5155.452328	3.82	0.0221
All Interactions		1	1.658459e+03	1658.458931	1.23	0.2677
smoking * bmi	(Factor+Higher Order Factors)	1	1.658459e+03	1658.458931	1.23	0.2677
TOTAL NONLINEAR		4	9.587178e+03	2396.794504	1.78	0.1309
TOTAL NONLINEAR + INTERACTION		5	1.152744e+04	2305.487432	1.71	0.1293
REGRESSION		14	7.030149e+04	5021.535034	3.72	<.0001
ERROR		2017	2.721574e+06	1349.317884		

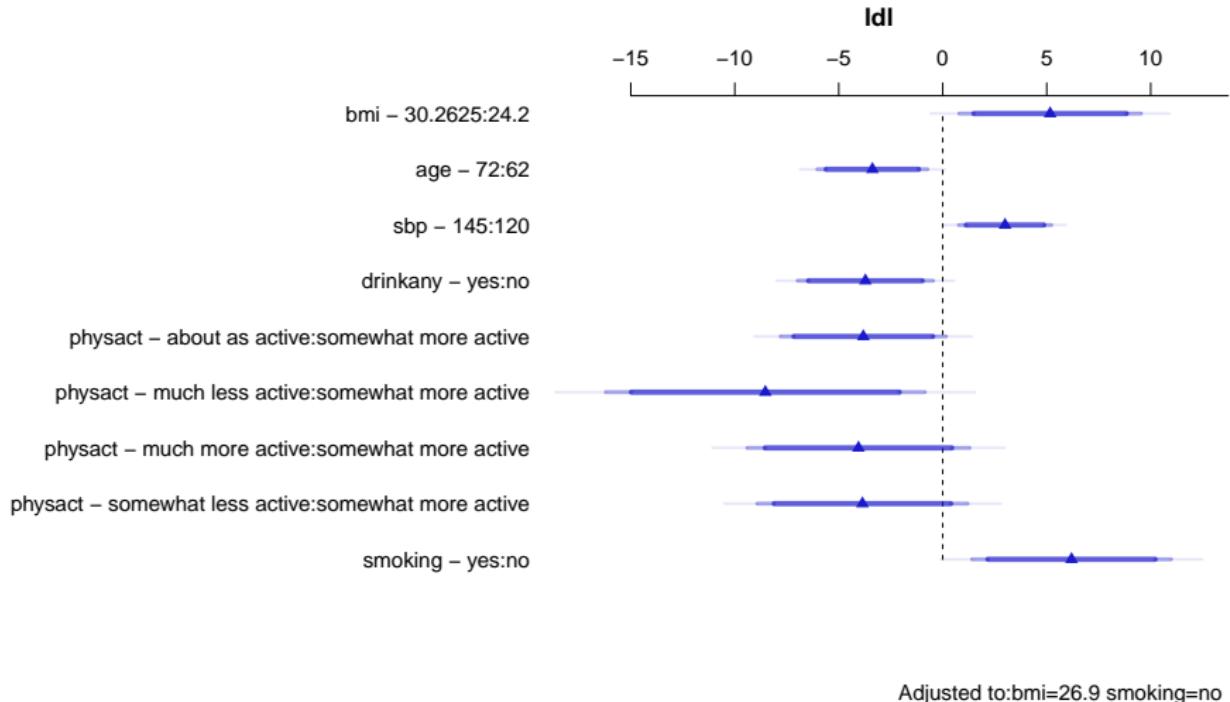
Summary of Effect Estimates for m3imp

summary(m3imp)

Effects	Response : ldl						
Factor	Low	High	Diff.	Effect	S.E.	Lower 0.95	Upper 0.95
bmi	24.2	30.263	6.0625	5.1643	2.2300	0.79099	9.53750
age	62.0	72.000	10.0000	-3.3824	1.3518	-6.03340	-0.73144
sbp	120.0	145.000	25.0000	2.9955	1.1345	0.77068	5.22040
drinkany - yes:no	1.0	2.000	NA	-3.7196	1.6613	-6.97780	-0.46150
physact - about as active:somewhat more active	5.0	1.000	NA	-3.8197	2.0286	-7.79800	0.15861
physact - much less active:somewhat more active	5.0	2.000	NA	-8.5306	3.9152	-16.20900	-0.85228
physact - much more active:somewhat more active	5.0	3.000	NA	-4.0525	2.7260	-9.39850	1.29350
physact - somewhat less active:somewhat more active	5.0	4.000	NA	-3.8614	2.5796	-8.92030	1.19760
smoking - yes:no	1.0	2.000	NA	6.1923	2.4427	1.40190	10.98300

Adjusted to: bmi=26.9 smoking=no

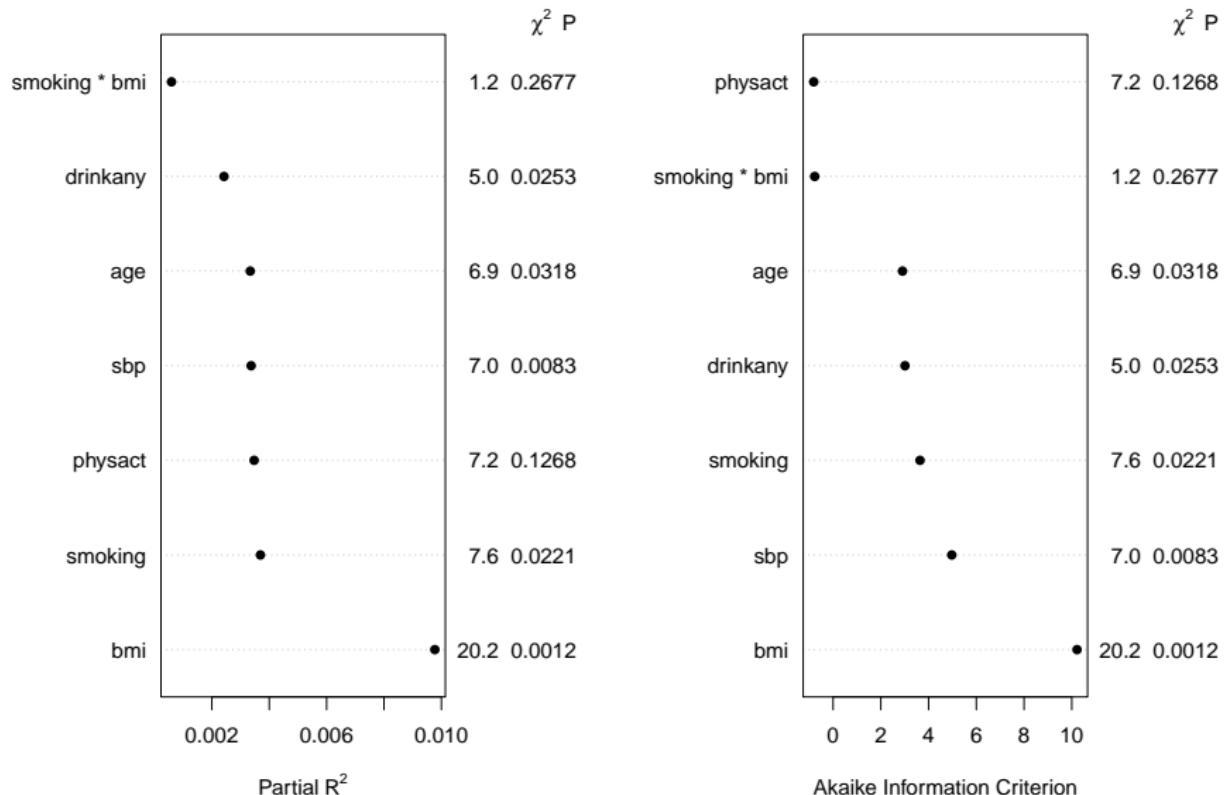
plot(summary(m3imp))



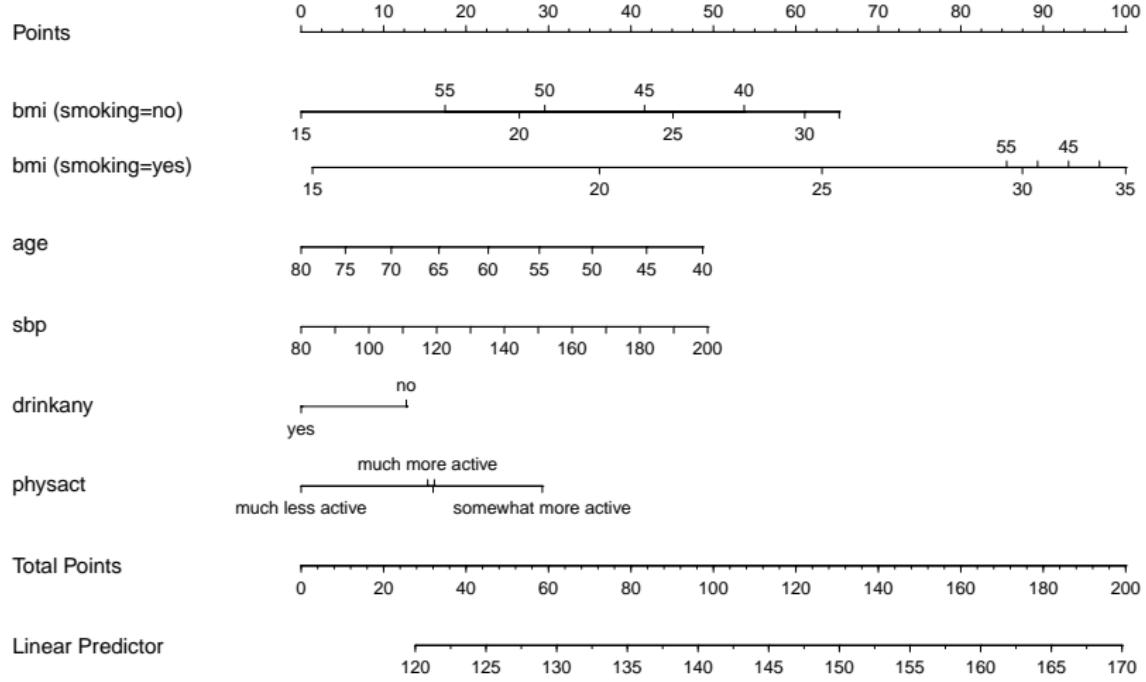
Evaluation via Partial R² and AIC (code)

```
par(mfrow = c(1,2))
plot(anova(m3imp), what="partial R2")
plot(anova(m3imp), what="aic")
par(mfrow = c(1,1))
```

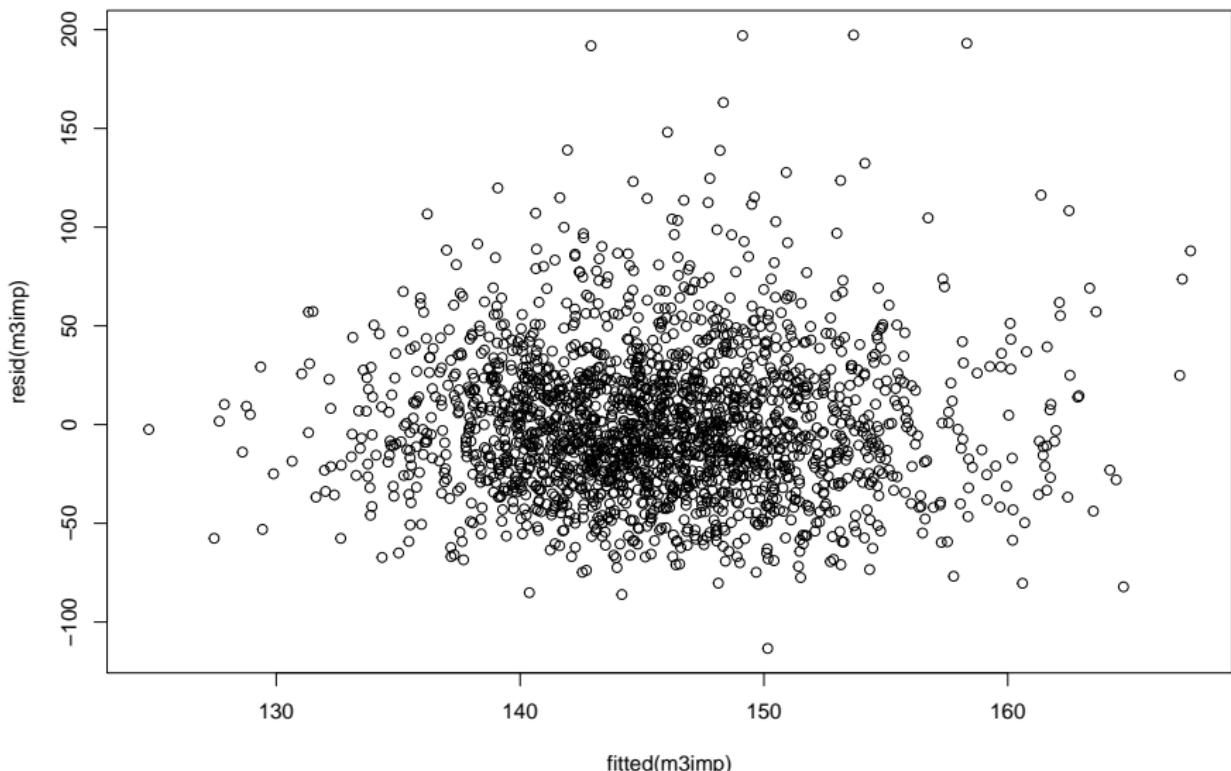
Evaluation via Partial R² and AIC (result)



plot(nomogram(m3imp))



```
plot(resid(m3imp) ~ fitted(m3imp))
```



Other Things I Might Need after aregImpute?

- How can I estimate the AIC (and BIC) of a model fit with `fit.mult.impute`?

`glance` won't work with an `ols` fit, but we can just use...

```
AIC(m3imp)
```

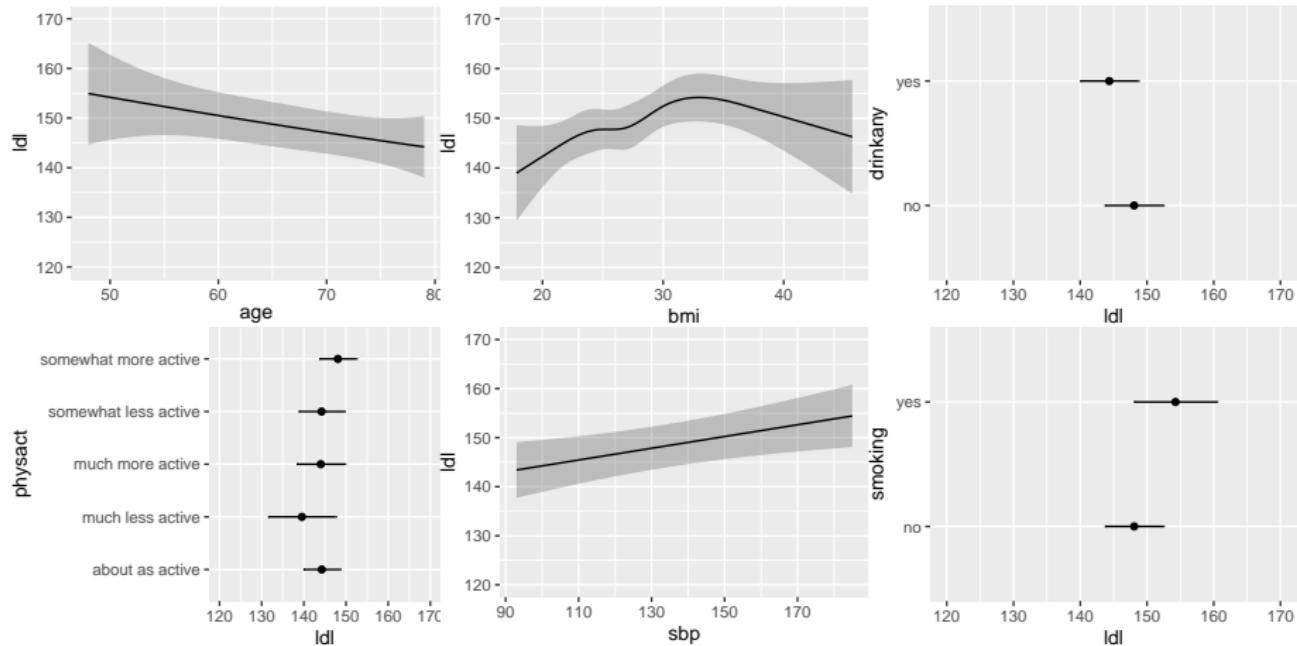
```
  d.f.  
20425.29
```

```
BIC(m3imp)
```

```
  d.f.  
20515.16
```

Can I run ggplot(Predict())?)?

```
ggplot(Predict(m3imp))
```



Pull out one imputation from aregImpute?

- How can I pull a single one (say, the fifth) of the imputations from aregImpute out?

Remember that fit3 was our imputation model here, build on the hers1 data, which keeps its subject identifiers in the subject column.

```
imputed_5 <-  
  impute.transcan(fit3, data = hers1, imputation = 5,  
                  list.out = T, pr = F, check = F)  
  
imputed_df5 <- as.data.frame(do.call(cbind, imputed_5))  
  
fifth_imp <-  
  bind_cols(subject = hers1$subject, imputed_df5) %>%  
  type.convert(as.is = FALSE) %>% tibble()
```

We'll show the result on the next slide.

Our fifth_imp tibble

fifth_imp

```
# A tibble: 2,032 x 8
  subject    ldl    age smoking drinkany    sbp physact    bmi
  <int>   <dbl>  <int>   <fct>      <int>  <int>   <fct>   <dbl>
1       1    122.     70 no          1    138 much mo~  23.7
2       2    242.     62 no          1    118 much le~  28.6
3       4    116.     64 yes         2    152 much le~  24.4
4       5    151.     65 no          1    175 somewha~  21.9
5       6    138.     68 no          2    174 about a~  29.0
6       8    121.     69 no          1    178 much mo~  23.2
7       9    133      61 no          2    162 about a~  30.3
8      10    220      62 yes        2    111 somewha~  45.7
9      11    173.     72 no          1    122 about a~  22.2
10     12    124.     73 no          1    158 somewha~  25.3
# ... with 2,022 more rows
```

Create Residual Plots for this imputation?

```
model_for_resid_plots <-
  lm(ldl ~ rcs(bmi, 5) + pol(age, 2) + sbp +
     drinkany + physact + smoking +
     smoking %ia% bmi, data = fifth_imp)
```

We can look at this model with `glance` or `tidy` to see that it gives similar results to what we see across the multiple imputations.

```
broom::glance(model_for_resid_plots) %>%
  select(r.squared, AIC, BIC, nobs, df, df.residual) %>%
  kable(digits = 3)
```

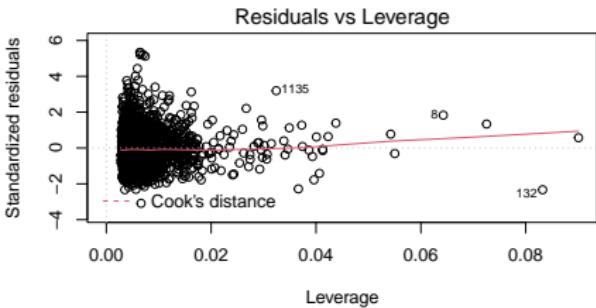
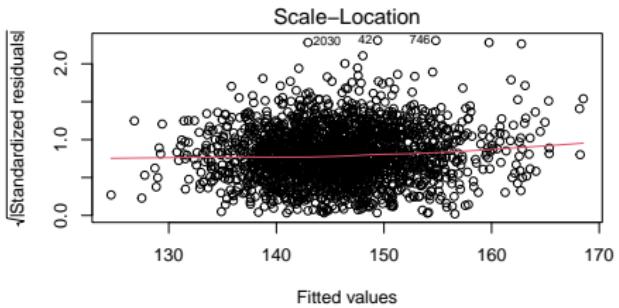
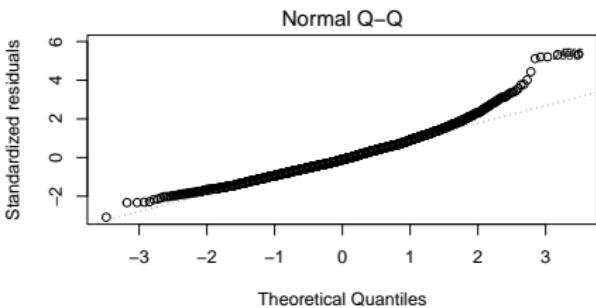
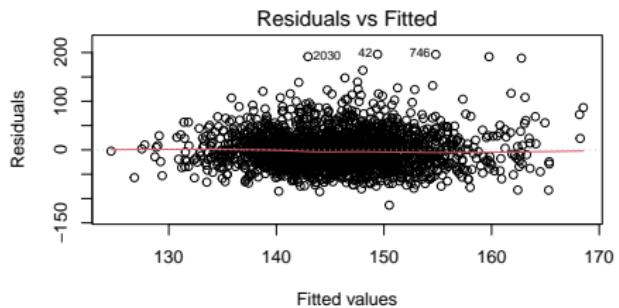
r.squared	AIC	BIC	nobs	df	df.residual
0.027	20451.36	20541.23	2032	14	2017

What else can we do?

We can plot residuals for the model fit to this single imputation, as shown on the next slide.

```
par(mfrow = c(2,2))
plot(model_for_resid_plots)
par(mfrow = c(1,1))
```

Residual Plots for Fifth Imputation



Next Step

Can we do all of this for a logistic regression model?

432 Class 10 Slides

thomaselove.github.io/432

2022-02-10

Today's Agenda

Fitting and evaluating logistic regression models with `lrm`

- The framingham example
 - Outcome: `chd10` = Developed coronary heart disease in next 10 years?
 - Creating “complete case” data: `fram_cc`
 - Single Imputation of Missing Values: `fram_sh`
- Use `lrm` to predict `chd10` using `glucose`, `smoker`, `sbp` and `educ`
 - on the complete cases (`fram_cc`)
 - accounting for missingness via single imputation (`fram_sh`)
 - accounting for missingness via multiple imputation
- Consider adding non-linear terms, refit and re-evaluate

Setup

```
library(magrittr); library(janitor)
library(here); library(knitr)
library(naniar); library(simputation)
library(ROCR)          # one way to draw ROC curves
library(rms)
library(tidyverse)

theme_set(theme_bw())
```

Today's Data

```
fram_raw <- read_csv(here("data/framingham.csv")) %>%
  clean_names()
```

See <https://www.framinghamheartstudy.org/> for more details. This particular data set has been used by lots of people, in varied settings, with variations all over the net. I don't know who the originators were.

Managing the Framingham Data

Data Cleanup

```
fram <- fram_raw %>%
  mutate(educ =
    fct_recode(factor(education),
      "Some HS" = "1",
      "HS grad" = "2",
      "Some Coll" = "3",
      "Coll grad" = "4")) %>%
  rename(smoker = "current_smoker",
    cigs = "cigs_per_day",
    stroke = "prevalent_stroke",
    highbp = "prevalent_hyp",
    chol = "tot_chol",
    sbp = "sys_bp", dbp = "dia_bp",
    hrate = "heart_rate",
    chd10 = "ten_year_chd") %>%
  select(subj_id, chd10, educ, glucose, sbp, smoker,
    everything()) %>% select(-education)
```

Data Descriptions (Main Variables Today)

The variables describe $n = 4238$ adults examined at baseline, then followed for 10 years to see if they developed incident coronary heart disease.

The main variables we'll use today in developing outcome models are:

Variable	Description
<code>subj_id</code>	identifying code added by Dr. Love
<code>chd10</code>	1 = coronary heart disease in next 10 years
<code>educ</code>	four-level factor: educational attainment
<code>glucose</code>	blood glucose level in mg/dl
<code>sbp</code>	systolic blood pressure (mm Hg)
<code>smoker</code>	1 = current smoker at time of examination, else 0

Data Descriptions (Other 11 variables)

Here are the other 11 variables in the fram data.

Variable	Description
male	1 = subject is male, else 0
age	in years (range is 32 to 70)
cigs	number of cigarettes smoked per day
bp_meds	1 = using anti-hypertensive medication at time of exam
stroke	1 = history of stroke, else 0
highbp	1 = under treatment for hypertension, else 0
diabetes	1 = history of diabetes, else 0
chol	total cholesterol (mg/dl)
dbp	diastolic blood pressure (mm Hg)
bmi	body mass index in kg/m^2
hrate	heart rate in beats per minute

Missing Data?

Our outcome chd10 has no missing values.

```
fram %>% tabyl(chd10) %>% adorn_pct_formatting(digits = 1)
```

chd10	n	percent
0	3594	84.8%
1	644	15.2%

- 3656 (86.3%) of the 4238 subjects in the fram data are complete.
- The remaining 582 observations have something missing.

```
n_case_complete(fram); pct_complete_case(fram)
```

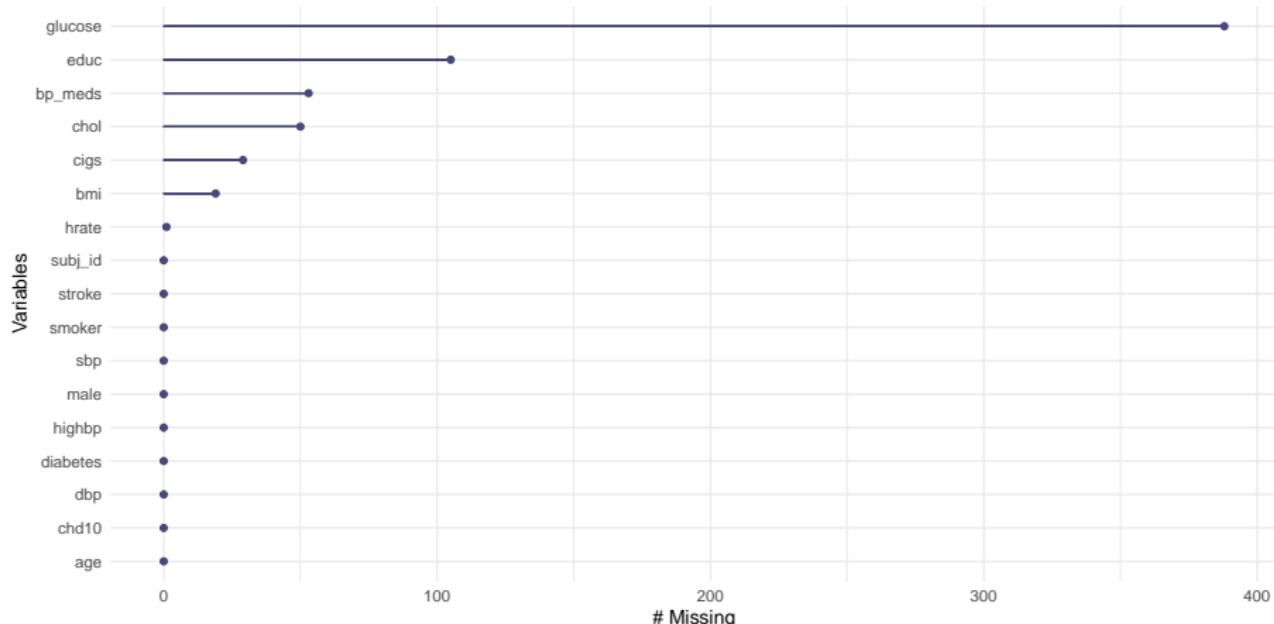
```
[1] 3656
```

```
[1] 86.26711
```

Which variables are missing data?

```
gg_miss_var(fram)
```

Warning: It is deprecated to specify `guide = FALSE` to remove a guide. Please use `guide = "none"` instead.



Counts of Missing Data, by Variable

```
miss_var_summary(fram) %>%
  filter(n_miss > 0)
```

```
# A tibble: 7 x 3
  variable n_miss pct_miss
  <chr>     <int>    <dbl>
1 glucose      388    9.16
2 educ         105    2.48
3 bp_meds      53    1.25
4 chol          50    1.18
5 cigs          29    0.684
6 bmi           19    0.448
7 hrate          1    0.0236
```

Single Imputation

We will impute:

- 5 quantitative variables (glucose, bmi, cigs, chol and hrate)
- 1 binary variable (bp_meds), and
- 1 multi-categorical variable (educ)

```
fram_sh <- bind_shadow(fram)
```

```
fram_sh <- fram_sh %>%
  data.frame() %>%
  impute_pmm(., bp_meds ~ highbp + sbp + dbp) %>%
  impute_cart(., educ ~ age + smoker + male) %>%
  impute_pmm(., cigs ~ smoker) %>%
  impute_rlm(., glucose + chol + hrate + bmi ~
    sbp + diabetes + age + highbp + stroke) %>%
  tibble()
```

Check multi-categorical single imputation?

```
fram_sh %>% count(educ_NA, educ)
```

```
# A tibble: 6 x 3
  educ_NA    educ      n
  <fct>     <fct>    <int>
1 !NA        Some HS   1720
2 !NA        HS grad  1253
3 !NA        Some Coll 687
4 !NA        Coll grad 473
5 NA         Some HS   80
6 NA         HS grad  25
```

Do the values seem reasonable?

Data Sets for the rest of our work

```
fram_start <- fram %>%
  select(subj_id, chd10, glucose, smoker, sbp, educ)
```

```
fram_cc <- fram_start %>%
  drop_na()
```

```
fram_sh <- fram_sh %>%
  select(subj_id, chd10, glucose, smoker, sbp, educ,
         glucose_NA, educ_NA)
```

- `fram_start` includes all 4238 rows and the 6 columns we'll use, including 388 rows missing glucose and 105 missing educ.
- `fram_cc` includes only the 3753 complete rows on the 6 columns.
- `fram_sh` uses single imputation to get 4238 complete rows, on 8 columns, including the useful missingness indicators.

Modeling Plan

Use `lrm` to fit a four-predictor logistic regression model to predict `chd10` using `glucose`, `smoker`, `sbp` and `educ`

- ① Using the complete cases (`fram_cc`)
- ② Accounting for missingness via single imputation (`fram_sh`)
- ③ Accounting for missingness via multiple imputation

Then, we'll consider adding several non-linear terms to the "four-predictor" models, and refit.

Fitting a Four-Predictor Model using Complete Cases

A “Four Predictor” model

First, we'll use the fram_cc data to perform a complete-case analysis and fix ideas.

```
d <- datadist(fram_cc)
options(datadist = "d")

mod_cc <- lrm(chd10 ~ glucose + smoker + sbp + educ,
               data = fram_cc, x = TRUE, y = TRUE)
```

This works very nicely when $\text{chd10} = 1$ (for Yes) or 0 (for No), as it does here. What if your outcome was actually a factor with values Yes and No? Use the following...

```
mod_cc <- lrm(outcome == "Yes" ~
                glucose + smoker + sbp + educ,
                data = fram_cc, x = TRUE, y = TRUE)
```

Main Output for mod_cc

mod_cc

Logistic Regression Model

```
lrm(formula = chd10 ~ glucose + smoker + sbp + educ, data = fram_cc,  
    x = TRUE, y = TRUE)
```

Obs	3753 0 1 max deriv	LR chi2 3174 579 2e-11	Model	Likelihood	Discrimination		Rank	Discrim.
			Ratio Test	223.29	R2	0.100	C	0.682
			d.f.	6	g	0.689	Dxy	0.363
			Pr(> chi2)	<0.0001	gr	1.992	gamma	0.364
					gp	0.092	tau-a	0.095
					Brier	0.122		

	Coef	S.E.	Wald Z	Pr(> z)
Intercept	-5.5622	0.3217	-17.29	<0.0001
glucose	0.0081	0.0016	4.93	<0.0001
smoker	0.3126	0.0955	3.27	0.0011
sbp	0.0237	0.0020	12.05	<0.0001
educ=HS grad	-0.4674	0.1157	-4.04	<0.0001
educ=Some coll	-0.3924	0.1423	-2.76	0.0058
educ=Coll grad	-0.1356	0.1549	-0.88	0.3815

- We'll walk through these summaries in the next few slides.
- Notes Section 12.14 provides additional details.

Deconstructing the mod_cc summaries (1/5)

Obs	3753
0	3174
1	579
max deriv	2e-11

- Obs = The number of observations used to fit the model, with 0 = the number of zeros and 1 = the number of ones in our outcome, chd10.
- Also specified is the maximum absolute value of the derivative at the point where the maximum likelihood function was estimated.

All you're likely to care about is whether the iterative function-fitting process converged, and R will warn you in other ways if it doesn't.

Deconstructing the mod_cc summaries (2/5)

```
Model Likelihood
Ratio Test
LR chi2      223.29
d.f.          6
Pr(> chi2) <0.0001
```

- This is a global likelihood ratio test (drop in deviance test.)
- Likelihood Ratio χ^2 statistic = null deviance - residual deviance
 - d.f. = null degrees of freedom - residual degrees of freedom
- $\text{Pr}(> \text{chi2})$ is a p value obtained from comparison to a χ^2 distribution with appropriate d.f.

It's not saying much to suggest that some part of this logistic regression model has some detectable predictive value.

- The null hypothesis here (that the model has no predictive value at all) is rarely interesting in practical work.

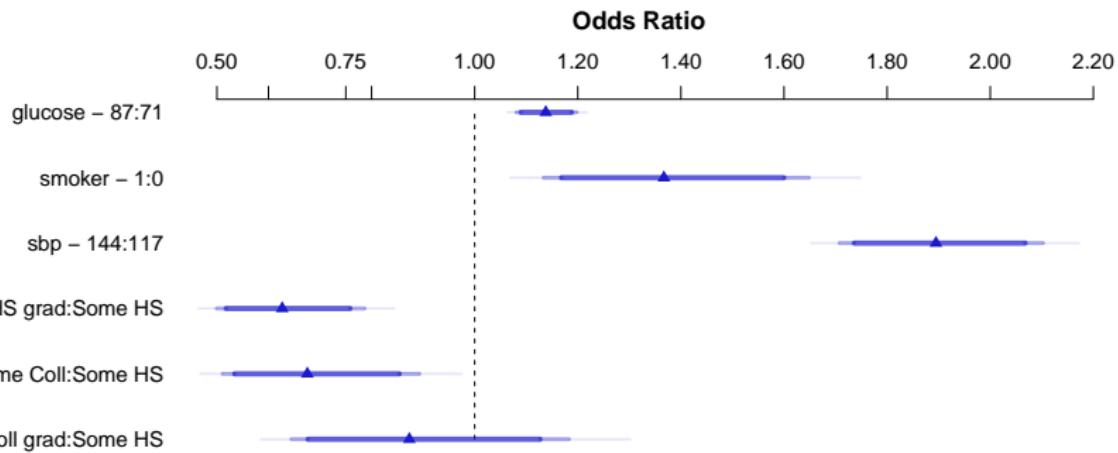
Deconstructing the mod_cc summaries (3/4)

	Coef	S.E.	Wald z	Pr(> z)
Intercept	-5.5622	0.3217	-17.29	<0.0001
glucose	0.0081	0.0016	4.93	<0.0001
smoker	0.3126	0.0955	3.27	0.0011
sbp	0.0237	0.0020	12.05	<0.0001
educ=HS grad	-0.4674	0.1157	-4.04	<0.0001
educ=Some Coll	-0.3924	0.1423	-2.76	0.0058
educ=Coll grad	-0.1356	0.1549	-0.88	0.3815

- How does each predictor appear to relate to 10-year risk?
 - Which is the baseline educ category?
 - Remember that these estimates are on the logit scale.
 - See the effect size discussion linked in today's README.

Plot of Effects using mod_cc

```
plot(summary(mod_cc))
```



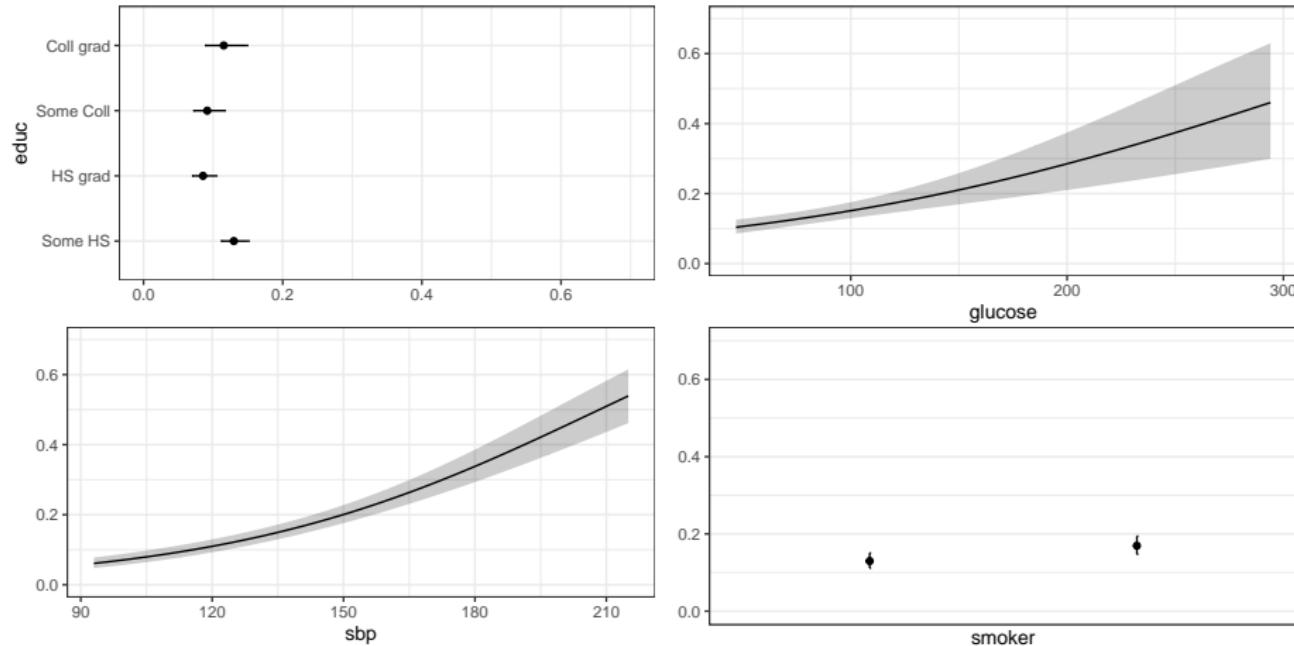
Effect Size Summary for mod_cc

```
summary(mod_cc)
```

	Effects			Response : chd10						
Factor	Low	High	Diff.	Effect	S.E.	Lower	0.95	Upper	0.95	
glucose	71 87		16	0.12912	0.026171	0.077828	0.18041			
Odds Ratio	71 87		16	1.13780	NA	1.080900	1.19770			
smoker	0 1		1	0.31259	0.095453	0.125510	0.49968			
Odds Ratio	0 1		1	1.36700	NA	1.133700	1.64820			
sbp	117 144		27	0.63907	0.053053	0.535080	0.74305			
Odds Ratio	117 144		27	1.89470	NA	1.707600	2.10230			
educ - HS grad:Some HS	1 2		NA	-0.46740	0.115720	-0.694220	-0.24059			
Odds Ratio	1 2		NA	0.62663	NA	0.499470	0.78616			
educ - Some Coll:Some HS	1 3		NA	-0.39238	0.142310	-0.671310	-0.11346			
Odds Ratio	1 3		NA	0.67544	NA	0.511040	0.89274			
educ - Coll grad:Some HS	1 4		NA	-0.13556	0.154910	-0.439180	0.16806			
Odds Ratio	1 4		NA	0.87323	NA	0.644570	1.18300			

Predict results for mod_cc

```
ggplot(Predict(mod_cc, fun = plogis))
```



Deconstructing the mod_cc summaries (4/4)

	Discrimination Indexes	Rank Discrim. Indexes
R2	0.100	C 0.682
g	0.689	Dxy 0.363
gr	1.992	gamma 0.364
gp	0.092	tau-a 0.095
Brier	0.122	

The key indexes for our purposes are:

- Nagelkerke R^2 , symbolized R2 here.
- The Brier score, symbolized Brier.
- The area under the ROC curve, or C statistic, shown as C.
- Somers' d statistic, symbolized Dxy here.

Let's walk through each of those, in turn.

Key Indexes (Nagelkerke R^2)

- In our model, Nagelkerke $R^2 = 0.100$

There are at least three ways to think about R^2 in linear regression, but when you move to a categorical outcome, not all of those ways can be expressed in the same statistic. See our Course Notes Section 10 for details.

The Nagelkerke R^2 :

- reaches 1 if the fitted model shows as much improvement as possible over the null model (which just predicts the mean response on the 0-1 scale for all subjects).
- is 0 for the null model
- is larger (closer to 1) as the fitted model improves, although it's been criticized for being misleadingly high,
- AND a value of 0.100 no longer means 10% of anything.

A value of 0.100 indicates a model of pretty poor quality.

An Alternative: McFadden's R^2

Consider the McFadden R-square, which can be defined as 1 minus the ratio of (the model deviance over the deviance for the intercept-only model.)

To obtain this for our `mod_cc` run with `lrm`, we can use:

```
1 - (mod_cc$deviance[2] / mod_cc$deviance[1])
```

```
[1] 0.069174
```

This McFadden R^2 corresponds well to the proportionate reduction in error interpretation of an R^2 , but some people don't like it as well.

Key Indexes (Brier Score = 0.122)

- The lower the Brier score, the better the predictions are calibrated.
- The maximum (worst) score is 1, the best is 0.

From Wikipedia: Suppose you're forecasting the probability P that it will rain on a given day.

- If the forecast is $P = 1$ (100%) and it rains, the Brier Score is 0.
- If the forecast is $P = 1$ (100%) and it doesn't rain, the Brier Score is 1.
- If the forecast is $P = 0.7$ and it rains, $\text{Brier} = (0.70 - 1)^2 = 0.09$.
- If the forecast is $P = 0.3$ and it rains, $\text{Brier} = (0.30 - 1)^2 = 0.49$.
- If the forecast is $P = 0.5$, the Brier score is $(0.50 - 1)^2 = 0.25$ regardless of whether it rains.

The Brier score can also be decomposed to assess calibration and discrimination separately.

Receiver Operating Characteristic Curve Analysis

One way to assess the predictive accuracy within the model development sample in a logistic regression is to consider analyses based on the receiver operating characteristic (ROC) curve. ROC curves are commonly used in assessing diagnoses in medical settings, and in signal detection applications.

The accuracy of a test can be evaluated by considering two types of errors: false positives and false negatives.

See Section 12.9 of our Course Notes for more details.

The C statistic (area under ROC curve) = 0.682

The C statistic and Somers' d (D_{xy}) are connected:

$$C = 0.5 + \frac{d}{2}, d = 2(C - .5)$$

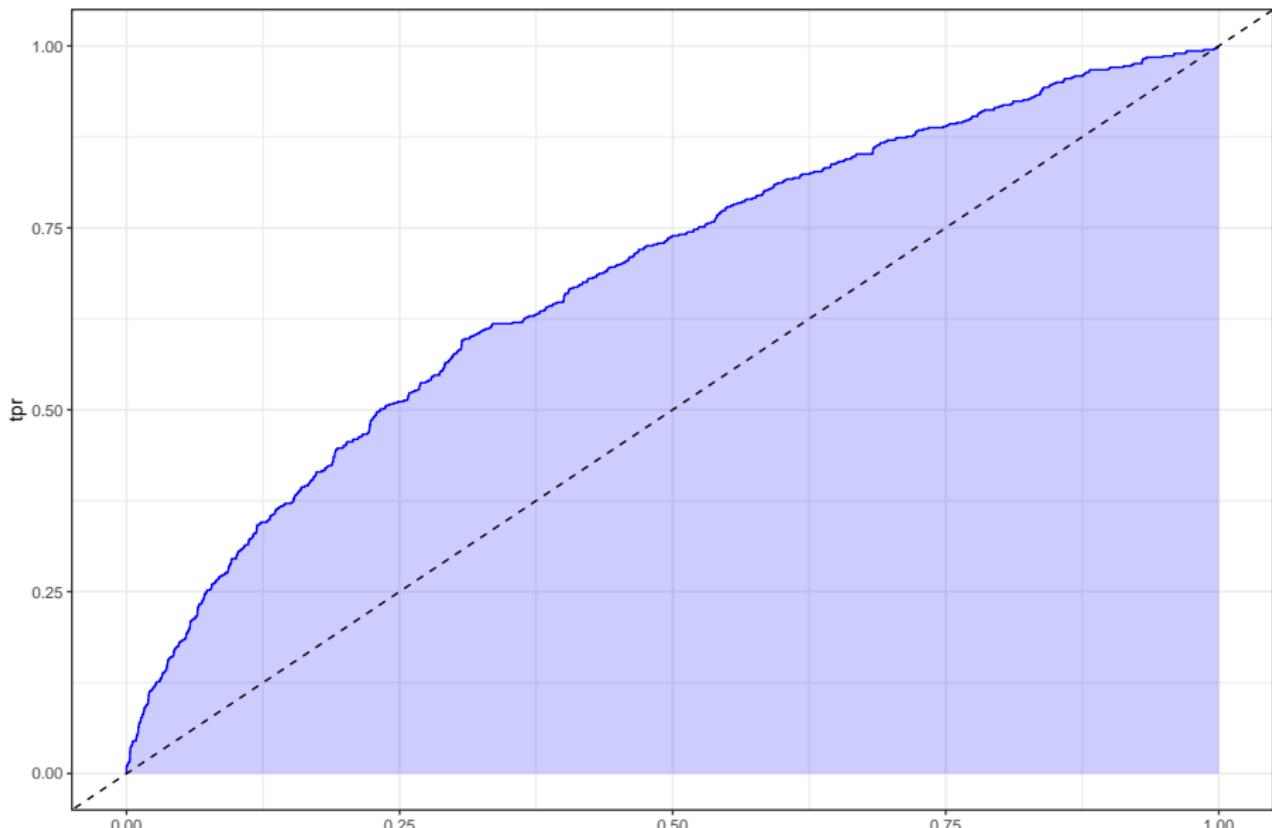
The C statistic ranges from 0 to 1.

- $C = 0.5$ describes a prediction that is exactly as good as random guessing
- $C = 1$ indicates a perfect prediction model, one that guesses “yes” for all patients with $chd10 = 1$ and which guesses “no” for all patients with $chd10 = 0$.
- Most of the time, the closer to 1, the happier we are:
 - $C \geq 0.8$ usually indicates a moderately strong model (good discrimination)
 - $C \geq 0.9$ indicates a very strong model (excellent discrimination)

So 0.682 isn't good.

ROC Curve for our mod_cc

mod_cc: ROC Curve w/ AUC=0.682



Code for Previous Slide

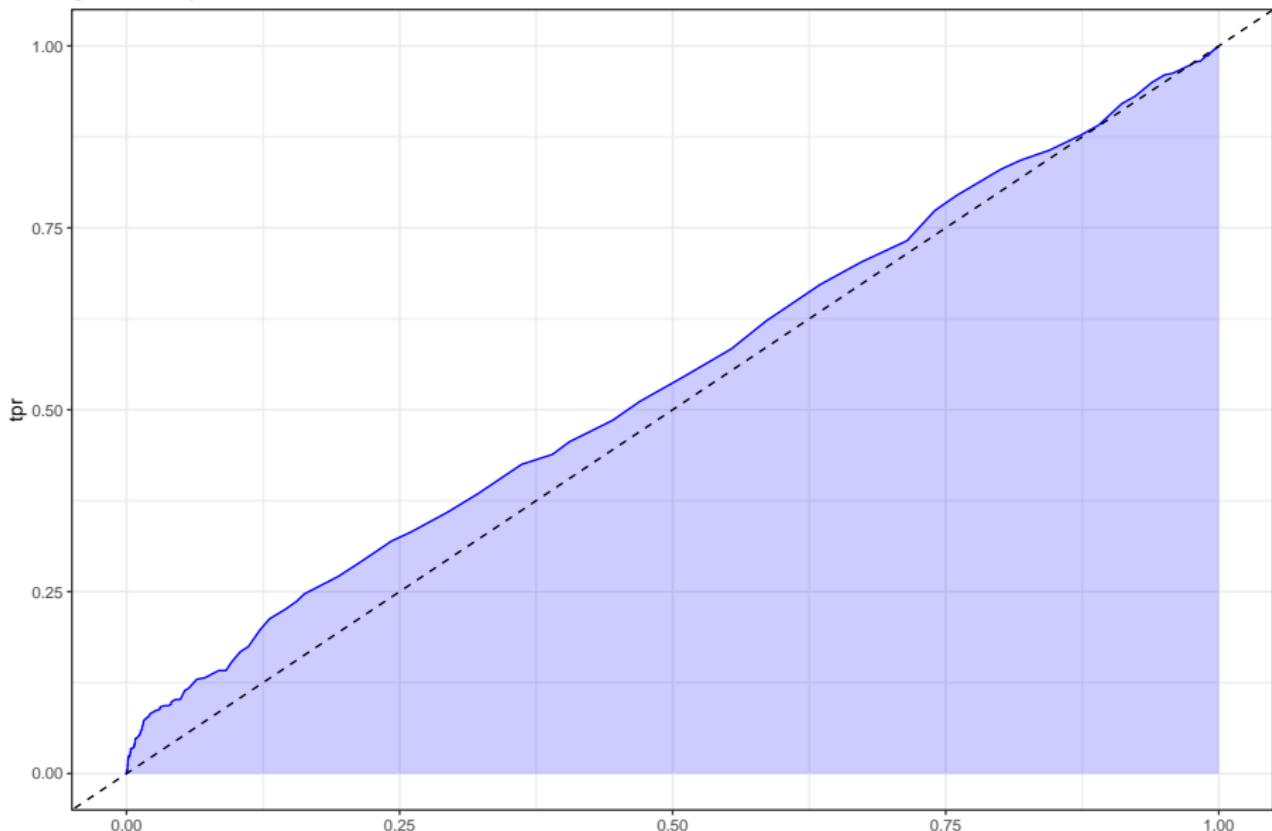
```
## requires ROCR package
prob <- predict(mod_cc, type="fitted")
pred <- prediction(prob, fram_cc$chd10)
perf <- performance(pred, measure = "tpr", x.measure = "fpr")
auc <- performance(pred, measure="auc")

auc <- round(auc@y.values[[1]],3)
roc.data <- data.frame(fpr=unlist(perf@x.values),
                        tpr=unlist(perf@y.values),
                        model="GLM")

ggplot(roc.data, aes(x=fpr, ymin=0, ymax=tpr)) +
  geom_ribbon(alpha=0.2, fill = "blue") +
  geom_line(aes(y=tpr), col = "blue") +
  geom_abline(intercept = 0, slope = 1, lty = "dashed") +
  labs(title = paste0("Model A: ROC Curve w/ AUC=", auc))
```

ROC Curve for a Simple Model (glucose only)

glucose only Model: ROC Curve w/ AUC=0.542



Validate Summary Statistics for mod_cc

- Usual approach (as in ols) to correcting for over-optimism through bootstrap validation, now using 50 bootstrap resamples instead of 40.

```
set.seed(432)  
validate(mod_cc, B = 50)
```

	index.orig	training	test	optimism	index.corrected	n
Dxy	0.3634	0.3655	0.3583	0.0072	0.3562	50
R2	0.1001	0.1007	0.0977	0.0029	0.0972	50
Intercept	0.0000	0.0000	-0.0196	0.0196	-0.0196	50
Slope	1.0000	1.0000	0.9873	0.0127	0.9873	50
Emax	0.0000	0.0000	0.0064	0.0064	0.0064	50
D	0.0592	0.0596	0.0578	0.0018	0.0574	50
U	-0.0005	-0.0005	0.0000	-0.0006	0.0000	50
Q	0.0598	0.0601	0.0577	0.0024	0.0574	50
B	0.1216	0.1215	0.1219	-0.0004	0.1220	50
g	0.6892	0.6933	0.6829	0.0105	0.6787	50
gp	0.0917	0.0918	0.0907	0.0011	0.0906	50

- Summaries we'll focus on here are Dxy, R2 and B
- Remember that $C = 0.5 + \frac{Dxy}{2}$, so our validated C statistic would be $0.5 + (0.3562/2) = 0.6781$

ANOVA for mod_cc

Model mod_cc uses 6 degrees of freedom.

```
anova(mod_cc)
```

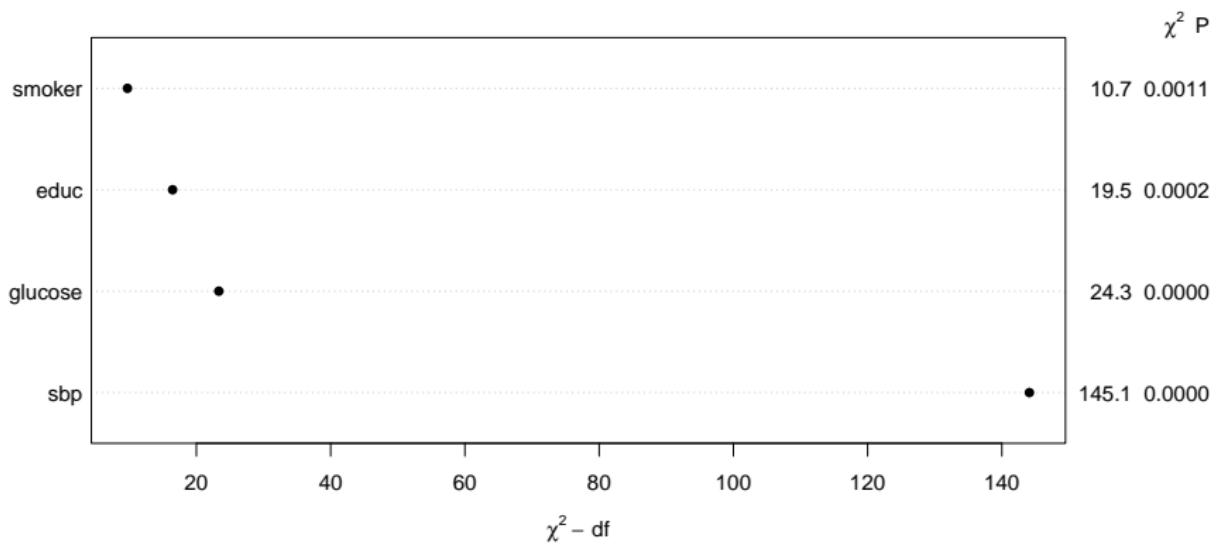
Wald Statistics

Response: chd10

Factor	Chi-Square	d.f.	P
glucose	24.34	1	<.0001
smoker	10.72	1	0.0011
sbp	145.10	1	<.0001
educ	19.45	3	0.0002
TOTAL	208.87	6	<.0001

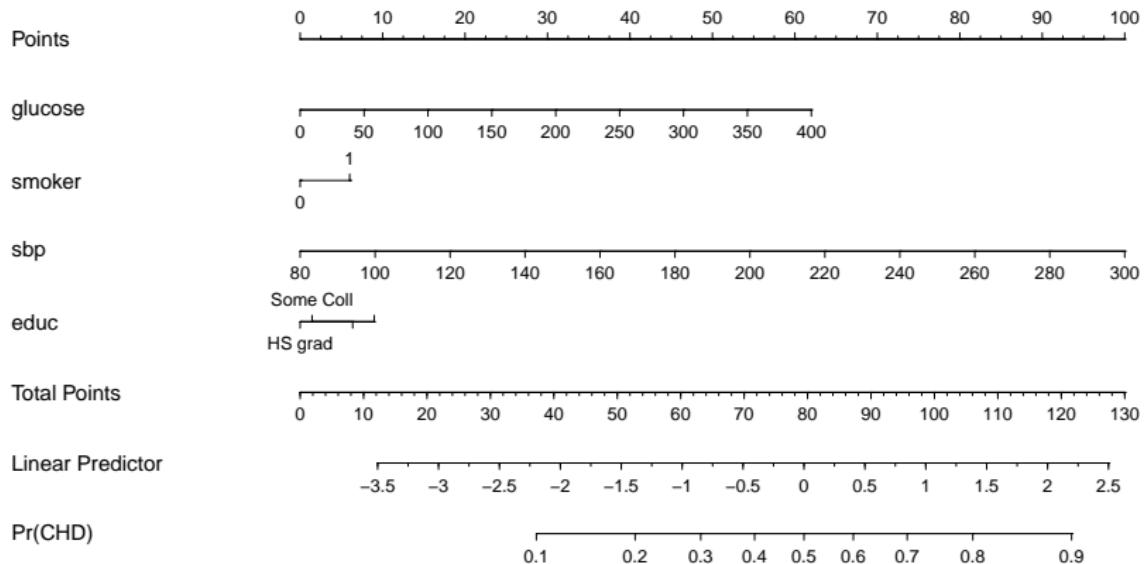
ANOVA for Model mod_cc

```
plot(anova(mod_cc))
```



Nomogram for mod_cc

```
plot(nomogram(mod_cc, fun = plogis,  
               funlabel = "Pr(CHD)"))
```



Using the Singly Imputed Data to fit the 4-predictor Model

Fit mod_si which is mod_cc after single imputation

```
d <- datadist(fram_sh)
options(datadist = "d")

mod_si <- lrm(chd10 ~ glucose + smoker + sbp + educ,
               data = fram_sh, x = TRUE, y = TRUE)
```

Model mod_si with single imputation

mod_si

```
Logistic Regression Model  
  
lrm(formula = chd10 ~ glucose + smoker + sbp + educ, data = fram_sh,  
    x = TRUE, y = TRUE)
```

Obs	4238 0 1 max deriv	Model Likelihood		Discrimination		Rank	Discrim. Indexes
		LR chi2	Ratio Test	R2	Indexes		
0	3594	d.f.	6	g	0.673	Dxy	0.354
1	644	Pr(> chi2)	<0.0001	gr	1.961	gamma	0.354
				gp	0.089	tau-a	0.091
				Brier	0.121		

	Coef	S.E.	Wald Z	Pr(> Z)
Intercept	-5.5649	0.3068	-18.14	<0.0001
glucose	0.0086	0.0016	5.32	<0.0001
smoker	0.3205	0.0901	3.56	0.0004
sbp	0.0231	0.0019	12.40	<0.0001
educ=HS grad	-0.4707	0.1098	-4.29	<0.0001
educ=Some Coll	-0.3055	0.1336	-2.29	0.0222
educ=Coll grad	-0.0816	0.1470	-0.56	0.5787

Comparing the Coefficients (exponentiated)

- Comparing the slopes as odds ratios

```
round_half_up(exp(mod_cc$coefficients), 3)
```

	Intercept	glucose	smoker	sbp
	0.004	1.008	1.367	1.024
educ=HS grad	educ=Some Coll	educ=Coll grad		
	0.627	0.675	0.873	

```
round_half_up(exp(mod_si$coefficients), 3)
```

	Intercept	glucose	smoker	sbp
	0.004	1.009	1.378	1.023
educ=HS grad	educ=Some Coll	educ=Coll grad		
	0.625	0.737	0.922	

Edited Summaries Comparing The Models

Summary	mod_si value	mod_cc value
Obs	4238	3753
0	3594	3174
1	644	579
Nagelkerke R^2	0.095	0.100
Brier Score	0.121	0.122
C	0.677	0.682
Dxy	0.354	0.363

- All of these results came from

mod_cc

mod_si

Validate mod_si Summary Statistics

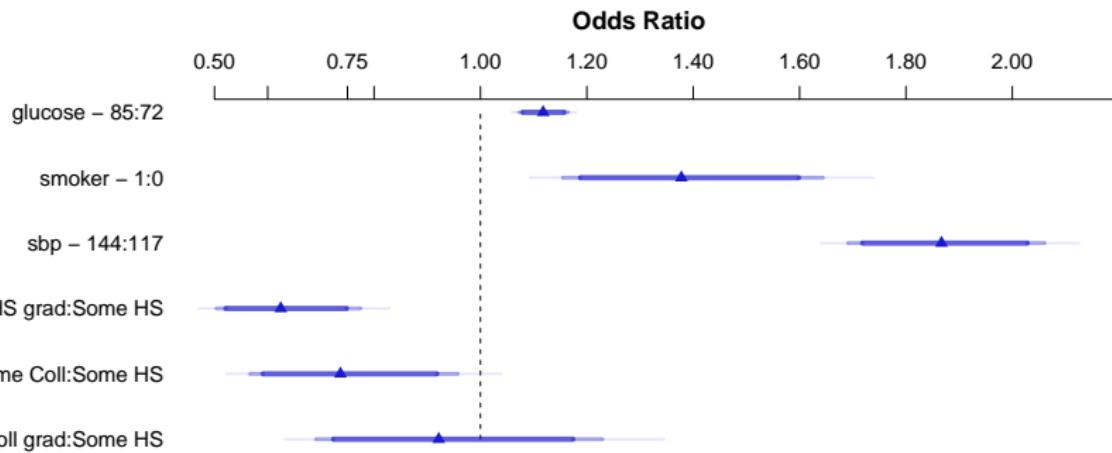
```
set.seed(432)  
validate(mod_si, B = 50)
```

	index.orig	training	test	optimism	index.corrected	n
Dxy	0.3538	0.3555	0.3496	0.0058	0.3480	50
R2	0.0954	0.0966	0.0933	0.0033	0.0921	50
Intercept	0.0000	0.0000	-0.0256	0.0256	-0.0256	50
Slope	1.0000	1.0000	0.9860	0.0140	0.9860	50
Emax	0.0000	0.0000	0.0079	0.0079	0.0079	50
D	0.0560	0.0568	0.0548	0.0021	0.0539	50
U	-0.0005	-0.0005	0.0000	-0.0005	0.0000	50
Q	0.0565	0.0573	0.0548	0.0026	0.0539	50
B	0.1206	0.1207	0.1208	-0.0001	0.1207	50

- Again, $C = 0.5 + \frac{D_{xy}}{2}$, so the corrected C statistic estimate will be $0.5 + (0.348/2) = 0.674$

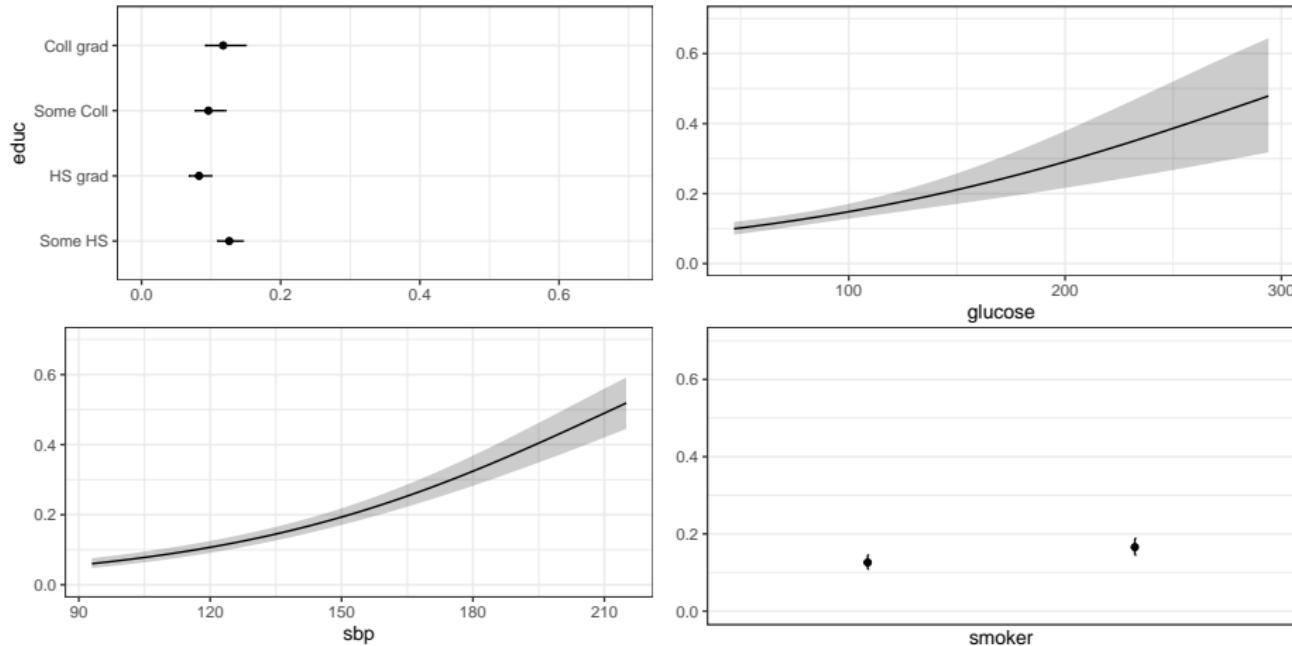
Plot of Effects using mod_si

```
plot(summary(mod_si))
```



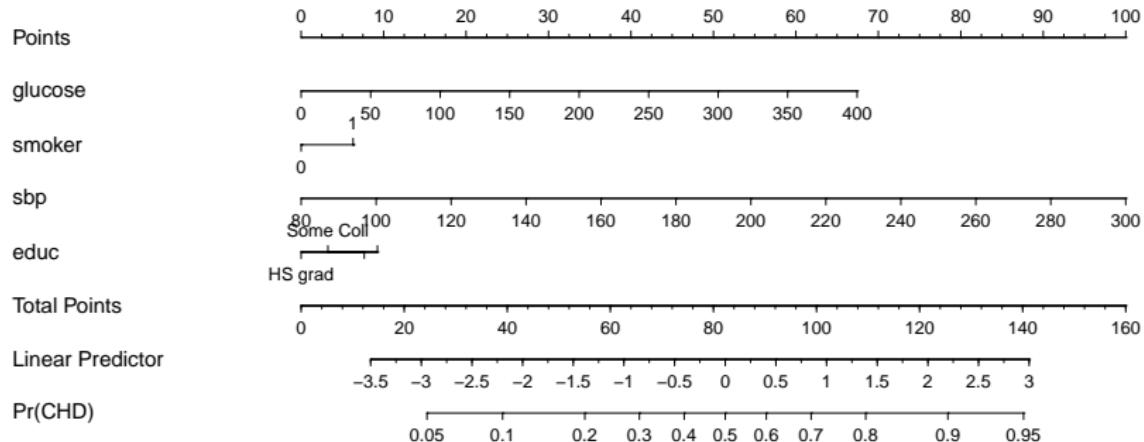
Predict results for mod_si

```
ggplot(Predict(mod_si, fun = plogis))
```



Nomogram for mod_si

```
plot(nomogram(mod_si, fun = plogis,
               fun.at = c(0.05, seq(0.1, 0.9, by = 0.1), 0.95),
               funlabel = "Pr(CHD)"))
```



Using Multiple Imputation: The 4-predictor Model

Fit the Imputation Model first

We'll use `aregImpute` here, and create 30 imputed sets.

```
set.seed(432)
dd <- datadist(fram)
options(datadist = "dd")

fit_imp <-
  aregImpute(~ chd10 + glucose + smoker + sbp + educ,
             nk = c(0, 3:5), tlinear = FALSE, data = fram,
             B = 10, n.impute = 30)
```

Iteration 1 Iteration 2 Iteration 3 Iteration 4 Iteration 5 It

Imputation Results (abbreviated output)

```
fit_imp
```

```
Multiple Imputation using Bootstrap and PMM

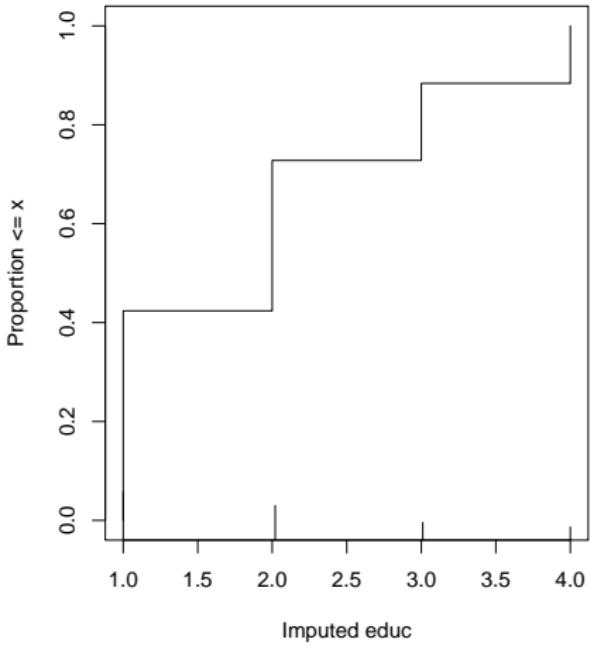
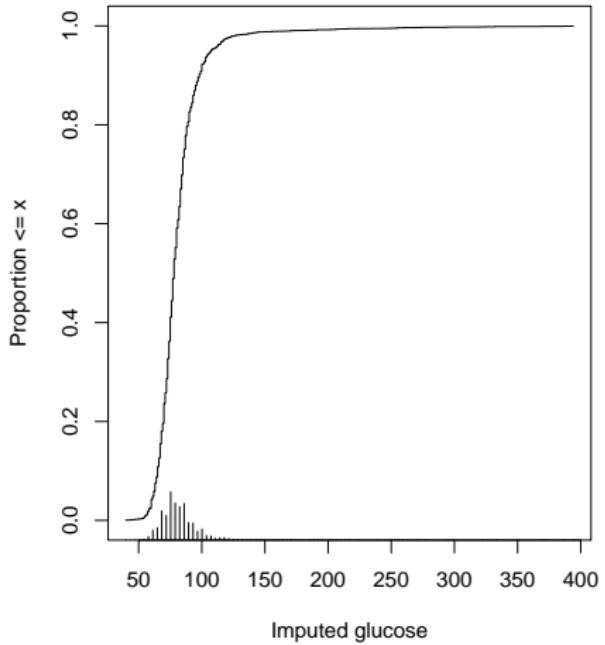
aregImpute(formula = ~chd10 + glucose + smoker + sbp + educ,
           data = fram, n.impute = 30, nk = c(0, 3:5), tlinear = FALSE,
           B = 10)

n: 4238          p: 5      Imputations: 30          nk: 0

Number of NAs:
  chd10   glucose   smoker     sbp     educ
    0       388       0         0      105

R-squares for Predicting Non-Missing Values for Each Variable
Using Last Imputations of Predictors
glucose   educ
  0.046   0.024
```

Multiply Imputed Values, via `plot(fit_imp)`



What do we need to do our multiple imputation?

- Imputation Model

```
fit_imp <-
  aregImpute(~ chd10 + glucose + smoker + sbp + educ,
             nk = c(0, 3:5), tlinear = FALSE, data = fram,
             B = 10, n.impute = 30)
```

- Outcome Model will be of the following form...

```
lrm(chd10 ~ glucose + smoker + sbp + educ,
     x = TRUE, y = TRUE)
```

Fitting mod_mi (mod_cc with multiple imputation)

```
mod_mi <-  
  fit.mult.impute(chd10 ~ glucose + smoker + sbp + educ,  
                  fitter = lrm, xtrans = fit_imp,  
                  data = fram_start, x = TRUE, y = TRUE,  
                  pr = FALSE)
```

- data = fram_start (which includes NA values)
- xtrans = fit_imp (results from multiple imputation)
- fitter = lrm (we could actually use glm too)
- pr = FALSE avoids a long printout we don't need

Model mod_mi with multiple imputation

mod_mi

Logistic Regression Model

```
fit.mult.impute(formula = chd10 ~ glucose + smoker + sbp + educ,
    fitter = lrm, xtrans = fit_imp, data = fram_start, pr = FALSE,
    x = TRUE, y = TRUE)
```

Obs	4238	Model Likelihood		Discrimination		Rank	Discrim.
		Ratio	Test	Indexes	Indexes		
0	3594	d.f.	6	g	0.670	Dxy	0.354
1	644	Pr(> chi2)	<0.0001	gr	1.955	gamma	0.354
				gp	0.088	tau-a	0.091
				Brier	0.121		

	Coef	S.E.	Wald Z	Pr(> Z)
Intercept	-5.5542	0.3083	-18.02	<0.0001
glucose	0.0083	0.0016	5.12	<0.0001
smoker	0.3188	0.0902	3.54	0.0004
sbp	0.0232	0.0019	12.40	<0.0001
educ=HS grad	-0.4551	0.1120	-4.06	<0.0001
educ=Some Coll	-0.3002	0.1340	-2.24	0.0251
educ=Coll grad	-0.0845	0.1478	-0.57	0.5674

Comparing the Coefficients (exponentiated)

- I'll just compare the two models using imputation...

```
round_half_up(exp(mod_mi$coefficients), 3)
```

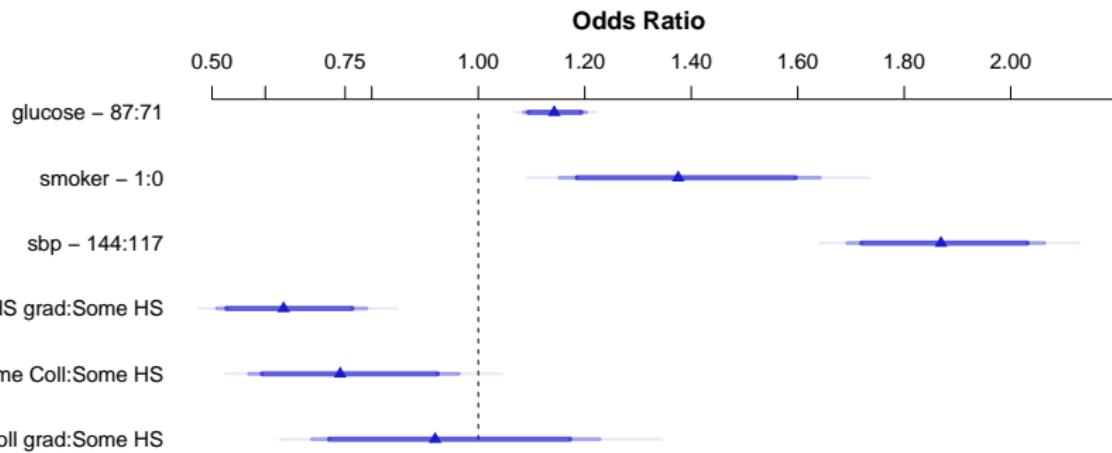
	Intercept	glucose	smoker	sbp
	0.004	1.008	1.376	1.023
educ=HS grad	educ=Some Coll	educ=Coll grad		
	0.634	0.741	0.919	

```
round_half_up(exp(mod_si$coefficients), 3)
```

	Intercept	glucose	smoker	sbp
	0.004	1.009	1.378	1.023
educ=HS grad	educ=Some Coll	educ=Coll grad		
	0.625	0.737	0.922	

Plot of Effects using mod_mi

```
plot(summary(mod_mi))
```



Edited Summaries Comparing Our 3 Models

Summary	mod_mi value	mod_si value	mod_cc value
Obs	4238	4238	3753
0	3594	3594	3174
1	644	644	579
Nagelkerke R^2	0.095	0.095	0.100
Brier Score	0.121	0.121	0.122
C	0.677	0.677	0.682
Dxy	0.354	0.354	0.363

- It's just a coincidence that the mod_mi and mod_si values are identical to the level of precision provided in this table.
- What might cause the values to look meaningfully different?

Validate mod_mi Summary Statistics

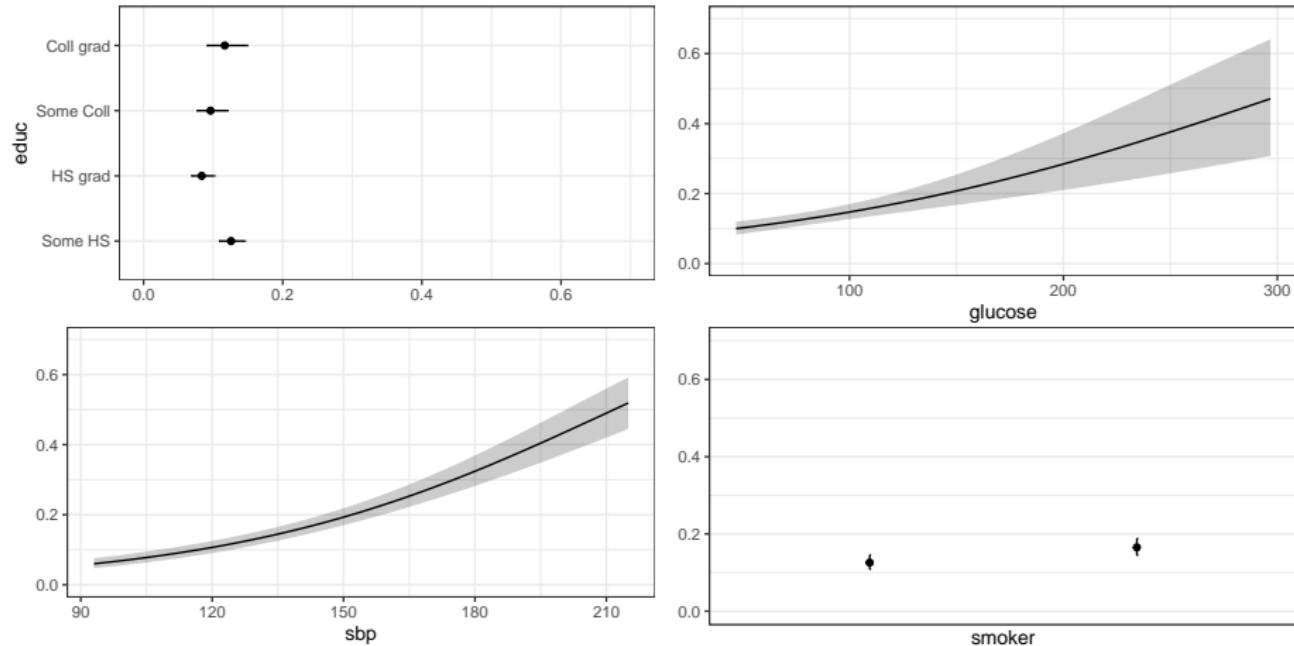
```
set.seed(432)
validate(mod_mi, B = 50)
```

	index.orig	training	test	optimism	index.corrected	n
Dxy	0.3535	0.3551	0.3493	0.0058	0.3477	50
R2	0.0952	0.0958	0.0925	0.0033	0.0919	50
Intercept	0.0000	0.0000	-0.0259	0.0259	-0.0259	50
Slope	1.0000	1.0000	0.9858	0.0142	0.9858	50
Emax	0.0000	0.0000	0.0080	0.0080	0.0080	50
D	0.0559	0.0564	0.0543	0.0021	0.0538	50
U	-0.0005	-0.0005	0.0000	-0.0005	0.0000	50
Q	0.0564	0.0569	0.0543	0.0026	0.0538	50
B	0.1207	0.1208	0.1209	-0.0001	0.1208	50

- Optimism-corrected C statistic estimate is $0.5 + (0.3477/2) = 0.674$

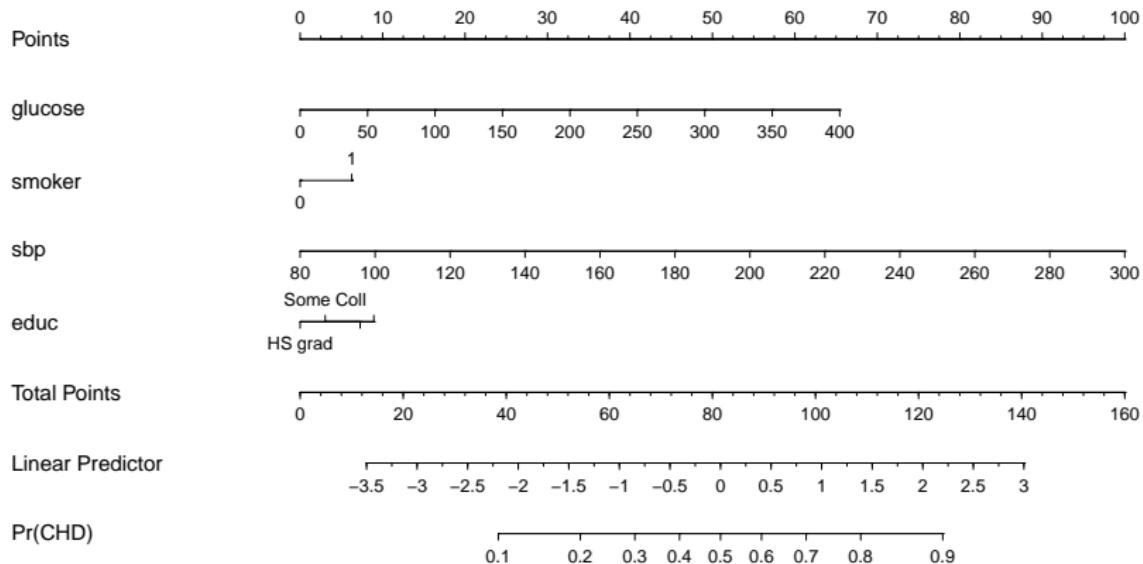
Predict results for mod_mi

```
ggplot(Predict(mod_mi, fun = plogis))
```



Nomogram for mod_mi

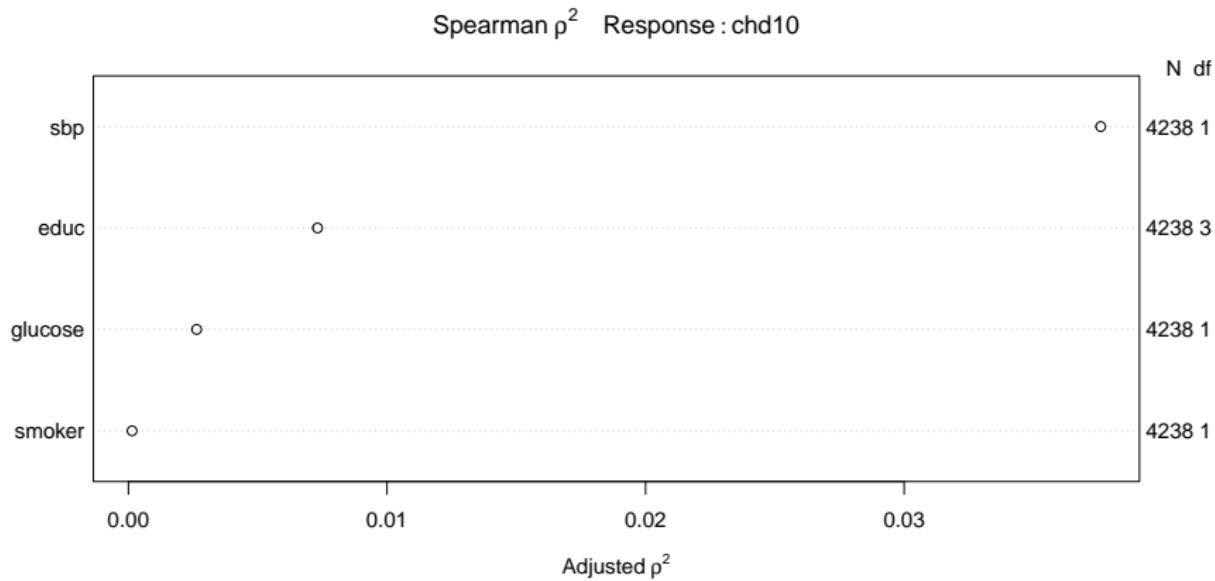
```
plot(nomogram(mod_mi, fun = plogis,  
               funlabel = "Pr(CHD)"))
```



Considering Non-Linear Terms

Spearman ρ^2 Plot

```
plot(spearman2(chd10 ~ glucose + smoker + sbp + educ,  
                data = fram_sh))
```



Adding some non-linear terms

- We'll add a restricted cubic spline with 5 knots in `sbp`
- and an interaction between the `educ` factor and the linear effect of `sbp`,
- and a quadratic polynomial in `glucose`

to our main effects model, just to show how to do them...

- I'll just show the results including the multiple imputation, since if you can get those, you should have little difficulty instead applying the single imputation or the complete case analysis.

mod_big incorporating multiple imputation

Our mod_big will incorporate several non-linear terms.

```
mod_big <-
  fit.mult.impute(
    chd10 ~ rcs(sbp, 5) + pol(glucose, 2) +
      smoker + educ + educ %ia% sbp,
    fitter = lrm, xtrans = fit_imp,
    data = fram_start, x = TRUE, y = TRUE,
    pr = FALSE)
```

The mod_big model with non-linear terms

Logistic Regression Model

```
fit.mult.impute(formula = chd10 ~ rcs(sbp, 5) + pol(glucose,  
2) + smoker + educ + educ %ia% sbp, fitter = lrm, xtrans = fit_imp,  
data = fram_start, pr = FALSE, x = TRUE, y = TRUE)
```

Obs	4238 0 1 max deriv	LR chi2 d.f. Pr(> chi2) <0.0001	Model Likelihood	Discrimination	Rank Discrim.
			Ratio Test	Indexes	Indexes
			R2	0.098	C 0.679
			g	0.710	Dxy 0.357
			gr	2.034	gamma 0.357
			gp	0.092	tau-a 0.092
			Brier	0.120	

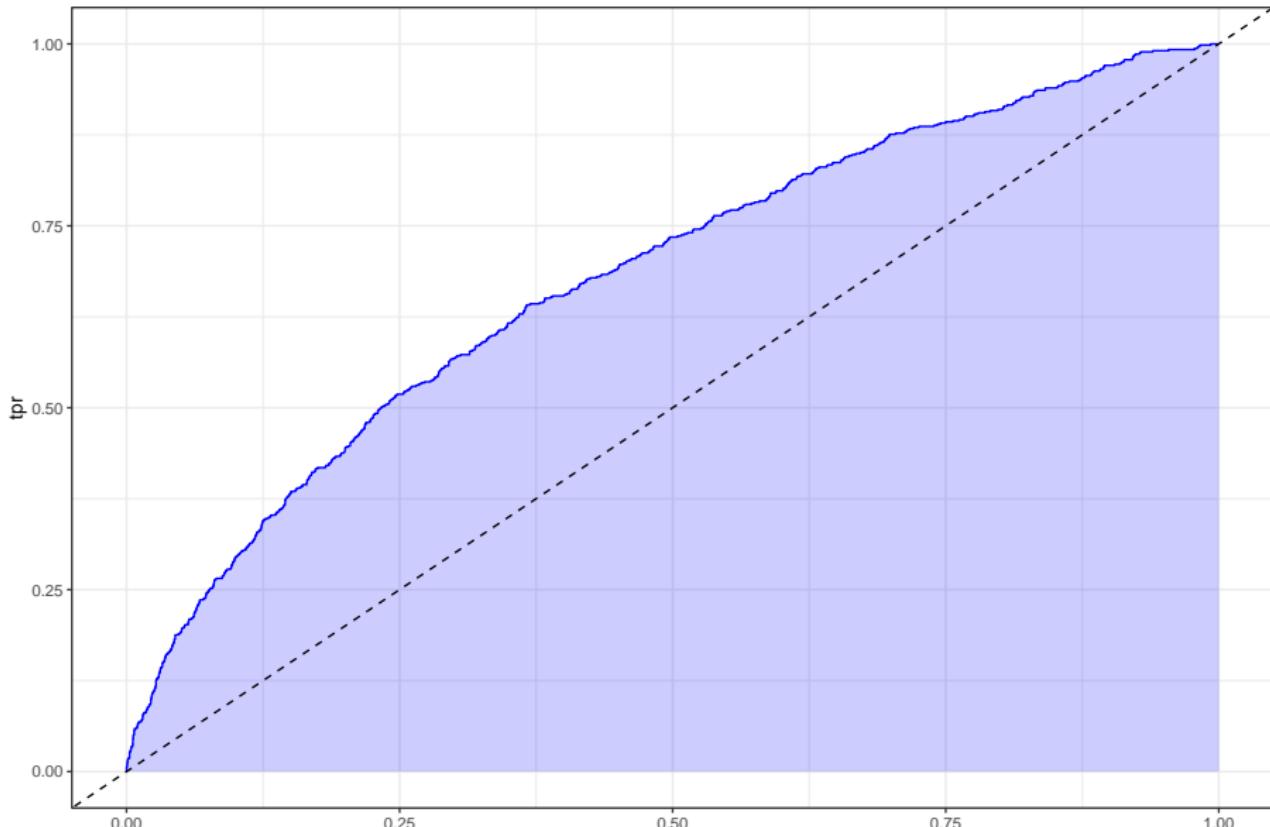
	Coef	S.E.	Wald z	Pr(> z)
Intercept	-3.2646	2.1123	-1.55	0.1222
sbp	0.0034	0.0190	0.18	0.8565
sbp'	0.1756	0.1837	0.96	0.3390
sbp''	-0.5056	0.6402	-0.79	0.4296
sbp'''	0.3651	0.6492	0.56	0.5738
glucose	0.0061	0.0054	1.12	0.2612
glucose^2	0.0000	0.0000	0.45	0.6495
smoker	0.3218	0.0903	3.56	0.0004
educ=HS grad	-0.4033	0.6438	-0.63	0.5310
educ=Some Coll	-1.4405	0.8055	-1.79	0.0737
educ=Coll grad	-1.1027	0.9379	-1.18	0.2397
educ=HS grad * sbp	-0.0004	0.0045	-0.09	0.9246
educ=Some Coll * sbp	0.0083	0.0057	1.44	0.1485
educ=Coll grad * sbp	0.0075	0.0068	1.10	0.2697

mod_big vs. mod_mi comparison

Summary	mod_big	mod_mi
Obs	4238	4238
0	3594	3594
1	644	644
Nagelkerke R^2	0.098	0.095
Brier Score	0.120	0.121
C	0.679	0.677
Dxy	0.357	0.354

ROC Curve for mod_big

Big Model: ROC Curve w/ AUC=0.68

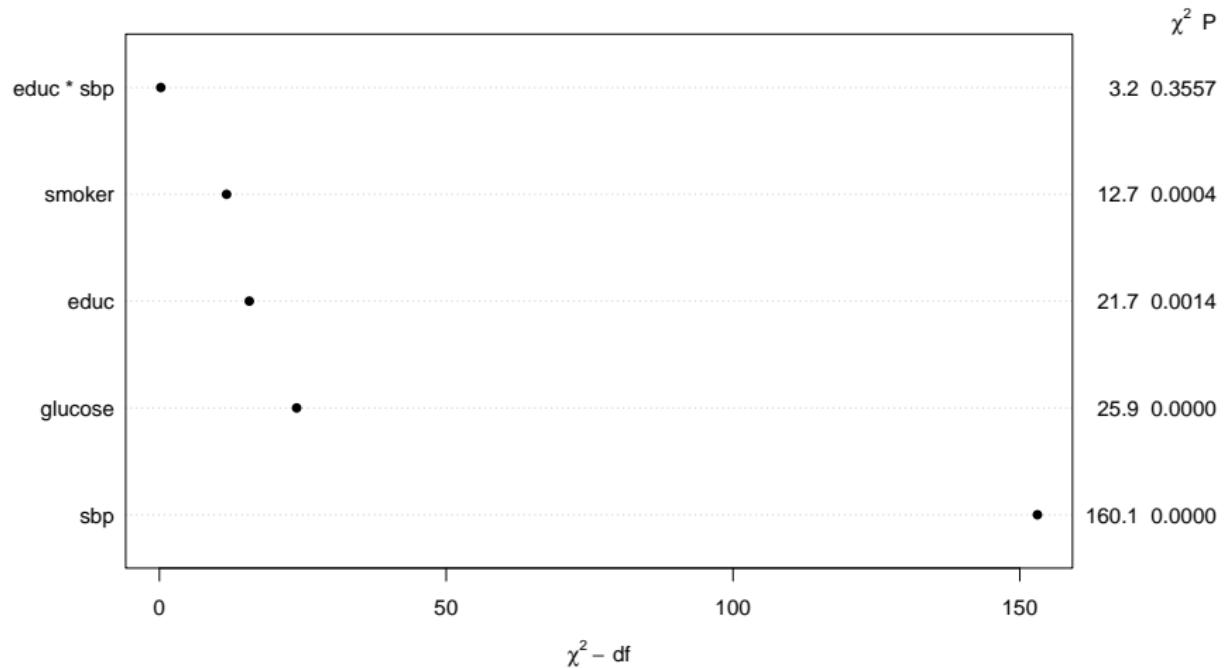


What does ANOVA suggest about the fit?

```
anova(mod_big)
```

	Wald Statistics	Response: chd10		
Factor		Chi-Square	d.f.	P
sbp (Factor+Higher Order Factors)		160.07	7	<.0001
All Interactions		3.24	3	0.3557
Nonlinear		3.03	3	0.3869
glucose		25.92	2	<.0001
Nonlinear		0.21	1	0.6495
smoker		12.71	1	0.0004
educ (Factor+Higher Order Factors)		21.68	6	0.0014
All Interactions		3.24	3	0.3557
educ * sbp (Factor+Higher Order Factors)		3.24	3	0.3557
TOTAL NONLINEAR		3.18	4	0.5280
TOTAL NONLINEAR + INTERACTION		7.14	7	0.4145
TOTAL		222.84	13	<.0001

`plot(anova(mod_big))` (model includes 13 df)



Validate mod_big Summary Statistics

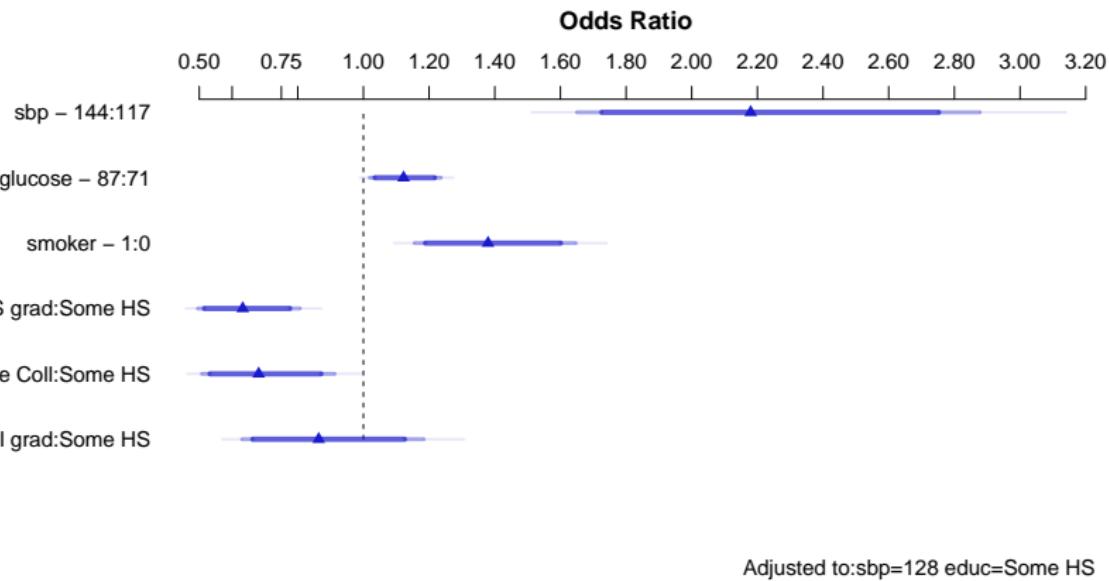
```
set.seed(432)  
validate(mod_big, B = 50)
```

	index.orig	training	test	optimism	index.corrected	n
Dxy	0.3577	0.3650	0.3507	0.0143	0.3434	50
R2	0.0980	0.1022	0.0922	0.0100	0.0880	50
Intercept	0.0000	0.0000	-0.0911	0.0911	-0.0911	50
Slope	1.0000	1.0000	0.9456	0.0544	0.9456	50
Emax	0.0000	0.0000	0.0296	0.0296	0.0296	50
D	0.0576	0.0603	0.0541	0.0062	0.0515	50
U	-0.0005	-0.0005	0.0003	-0.0007	0.0003	50
Q	0.0581	0.0607	0.0538	0.0069	0.0512	50
B	0.1204	0.1202	0.1209	-0.0007	0.1211	50

- Optimism-Corrected C = $0.5 + (.3434/2) = .672$

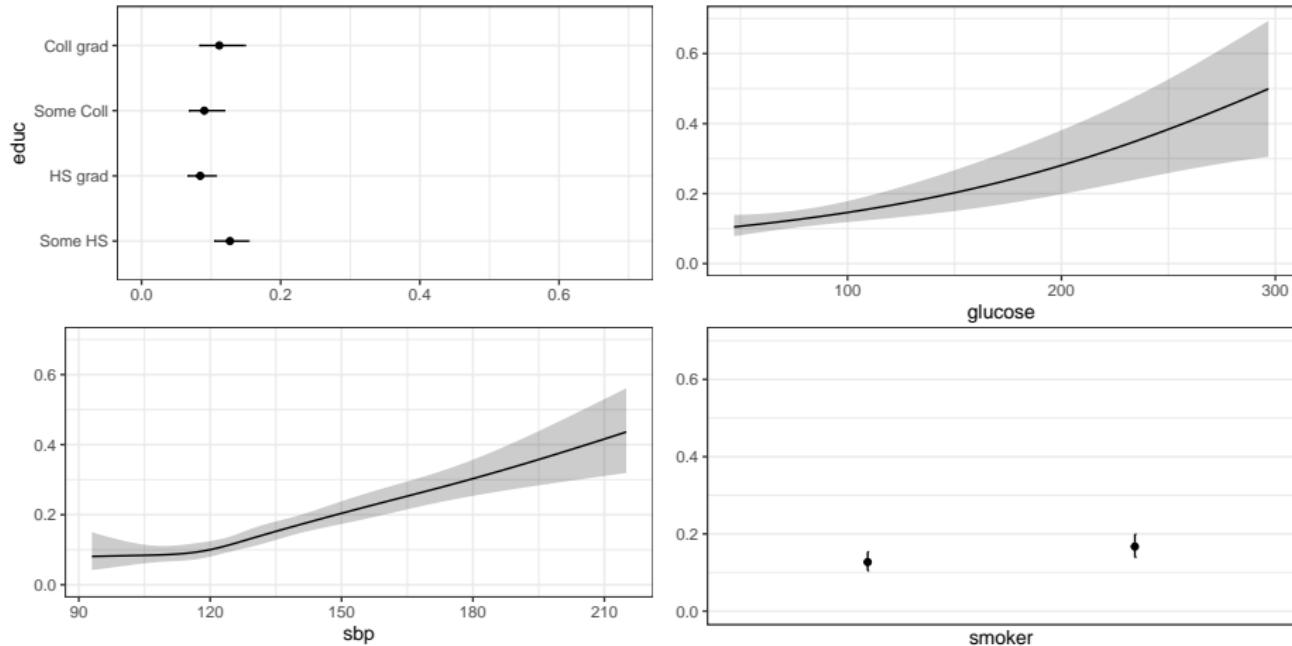
Plot of Effects using mod_big

```
plot(summary(mod_big))
```



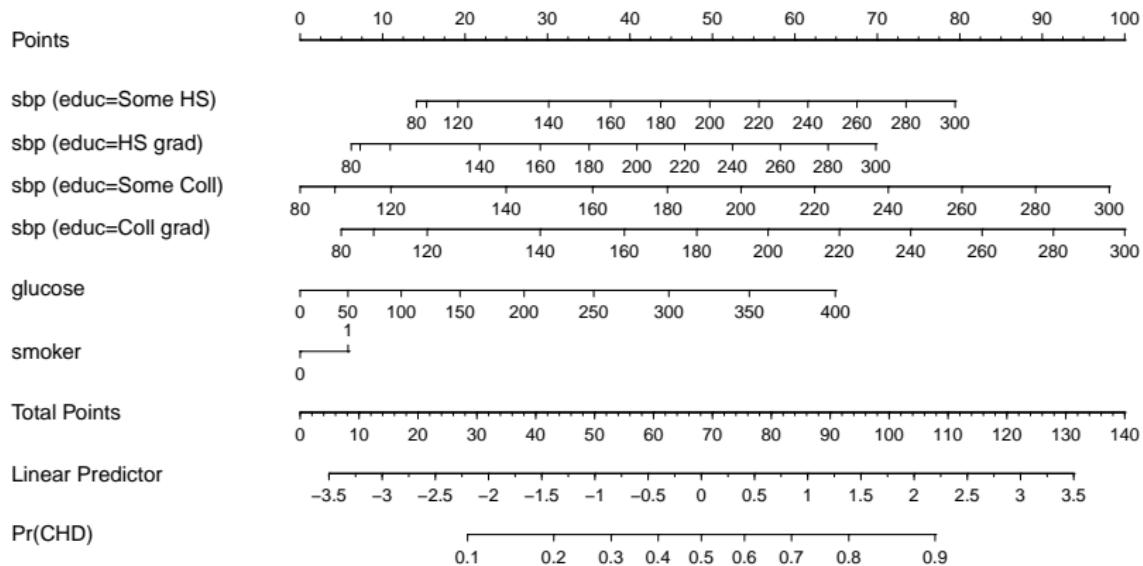
Predict results for mod_big

```
ggplot(Predict(mod_big, fun = plogis))
```



Nomogram for mod_big

```
plot(nomogram(mod_big, fun = plogis, funlabel = "Pr(CHD)"))
```



What's Next?

- Using the `tidymodels` framework to fit linear and logistic regression models

432 Class 11 Slides

thomaselove.github.io/432

2022-02-15

Today's Agenda

The tidymodels framework

- Using tidymodels tools to develop a linear regression model
 - Pre-processing activities
 - Model building (with multiple fitting engines)
 - Measuring model effectiveness
 - Creating a model workflow

Next Time (Class 12)

- Using tidymodels tools to develop a logistic regression model

Setup

```
library(here); library(conflicted)
library(knitr); library(magrittr); library(janitor)

library(tidymodels)
library(tidyverse)

theme_set(theme_bw())

conflict_prefer("select", "dplyr")
conflict_prefer("filter", "dplyr")
```

Regression Frameworks

Generally, regression allows us to summarize how predictions (or average values) of an outcome vary across individuals defined by a set of predictors. Some of the most important uses of regression are:

- **Prediction**, which involves both modeling existing observations and forecasting new data.
- **Exploring Associations**, where we summarize how well a set of variables predicts the outcome.
- **Extrapolation**, where we are adjusting for known differences between the observed sample of data and a population of interest.
- **Causal Inference**, where we are estimating the effect of a treatment, by comparing outcomes under treatment or control, or under different levels of a treatment¹.

Source: Gelman, Hill and Vehtari, *Regression and Other Stories*

¹My 500 course spends a whole semester on one important part of this subject.

Research Questions for Regression Models

- “How effectively can [insert quantitative outcome] be predicted using [insert predictor(s)]?” for a linear regression project, and
- “How effectively can [insert binary outcome] be predicted using [insert predictor(s)]?” for a logistic regression project.

If you’re struggling with this, or if your research question isn’t in the form of a question, consider these approaches. Advantages:

- ① regression can help provide an answer to these questions and in discussing your results you’ll need to answer the questions
- ② framing models in terms of exploring associations has some value for the tools we’re discussing and
- ③ it’s pretty clear what you’re doing, based just on your research question.

If you’re doing something else, I still need to think that you meet standards (1) and (3) at least.

Using R to fit Regression Models

For linear models, we have:

- `lm` to fit models for quantitative outcomes, compute and plot predictions and residuals, obtain confidence intervals, etc.
- `ols` from the `rms` package to save and explore additional components of the model's fit and to (slightly) expand the capacity for `lm` fits to incorporate non-linear terms and multiple imputations.

For logistic models, we have:

- `glm` to fit models for binary outcomes, compute and plot predictions, hypothesis tests and confidence intervals
- `lrm` from `rms` to save and explore additional components of the model's fit and to (slightly) expand the capacity for `glm` fits to incorporate non-linear terms and multiple imputations.

These are by no means the only options for fitting or working with models.

What are tidymodels?

The `tidymodels` collection of packages in R use tidyverse principles to facilitate modeling and machine learning work. The key idea is to develop a consistent framework for modeling, including:

- pre-processing data, which includes identifying variables and their roles, re-expression of outcomes, creation of features (predictors)
- building a model (potentially with multiple fitting “engines”)
- developing a re-usable workflow
- evaluating the fit of one model or various models with a variety of validation strategies

Visit the `tidymodels` website at <https://www.tidymodels.org/>.

Core Tidymodels Packages

Install many of the packages in the `tidymodels` ecosystem with `install.packages(tidymodels)`.

When you use `library(tidymodels)`, this makes the core packages available in your R session. They include:

- `rsample` which will help with data splitting and resampling
- `parsnip` which provides a tidy, unified interface for models
- `recipes` for data pre-processing and feature engineering
- `yardstick` for measuring model effectiveness
- `broom` for converting R objects into predictable formats
- `workflows` for bundling together pre-processing, modeling and post-processing work

as well as `dials` and `tune`, which help manage and optimize tuning parameters in certain types of models.

Today's Data (from Class 09)

Heart and Estrogen/Progestin Study (HERS)

- Clinical trial of hormone therapy for the prevention of recurrent heart attacks and deaths among 2763 post-menopausal women with existing coronary heart disease (see Hulley et al 1998 and many subsequent references, including Vittinghoff, Chapter 4.)
- We're excluding the women in the trial with a diabetes diagnosis and those with missing LDL values.

```
hers_raw <- read_csv(here("data/hersdata.csv")) %>%  
  clean_names()
```

```
hers_new <- hers_raw %>%  
  filter(diabetes == "no") %>%  
  filter(complete.cases(ldl1, ldl)) %>%  
  select(subject, ldl1, ldl, age, ht, globrat)
```

hers_new Codebook (n = 1925)

Variable	Description
subject	subject code
ht	factor: hormone therapy or placebo
ldl	baseline LDL cholesterol in mg/dl
age	baseline age in years
globrat	baseline self-reported health (5 levels)
ldl1	LDL at first annual study visit
diabetes	yes or no (all are no in our sample)

Goal Predict percentage change in ldl from baseline to followup, using baseline age, ht, ldl and globrat, restricted to women without diabetes.

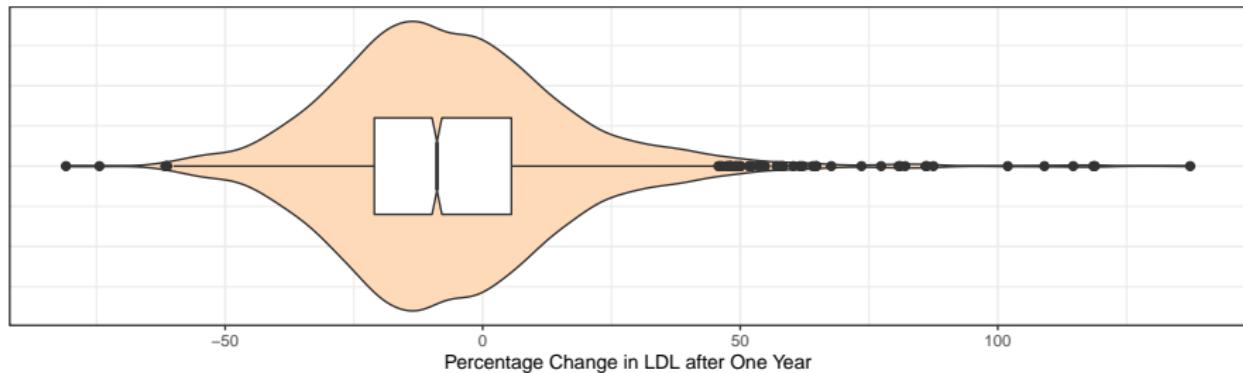
Steps we'll describe today

- ① Create our outcome and consider a transformation.
- ② Split the data into training and testing samples.
- ③ Build a recipe for our model.
 - Specify roles for outcome and predictors.
 - Deal with missing data in a reasonable way.
 - Complete all necessary pre-processing so we can fit models.
- ④ Specify a modeling engine for each fit we will create.
 - There are five available engines just for linear regression!
- ⑤ Create a workflow for each engine and fit model to the training data.
- ⑥ Compare coefficients graphically from two modeling approaches.
- ⑦ Assess performance in the models we create in the training data.
- ⑧ Compare multiple models based on their performance in test data.

Key Reference: Kuhn and Silge, *Tidy Modeling with R* or TMWR

Stage 1: Create our outcome

```
hers_new <- hers_new %>%
  mutate(ldl_pch = 100*(ldl1 - ldl)/ldl)
```



min	Q1	median	Q3	max	mean	sd	n	missing
-80.9	-21	-8.9	5.6	137.4	-6.5	22.8	1925	0

Stage 2: Creating Training and Test Samples



rsample

rsample provides infrastructure for efficient data splitting and resampling.

[Go to package ...](#)

Here, we'll use the `rsample` package to split our data.

```
set.seed(20210309)
hers_split <- initial_split(hers_new, prop = 0.8)

hers_train <- training(hers_split)
hers_test <- testing(hers_split)
```

We start with 1925 women in `hers_new`, which we split into 1540 women in the training sample, leaving 385 women in the testing sample.

What else can we do with rsample?

- Stratified sampling (splitting) on a categorical variable to ensure similar distributions of those categories in the training and testing groups.

```
initial_split(hers_new, prop = 0.8, strata = ht)
```

- What if you have time series data?
 - Use `initial_time_split()` to identify the first part of the data as the training set and the rest in the testing set; this assumes the data were pre-sorted in a sensible order.

The test set should **always** resemble new data that will be given to the model.

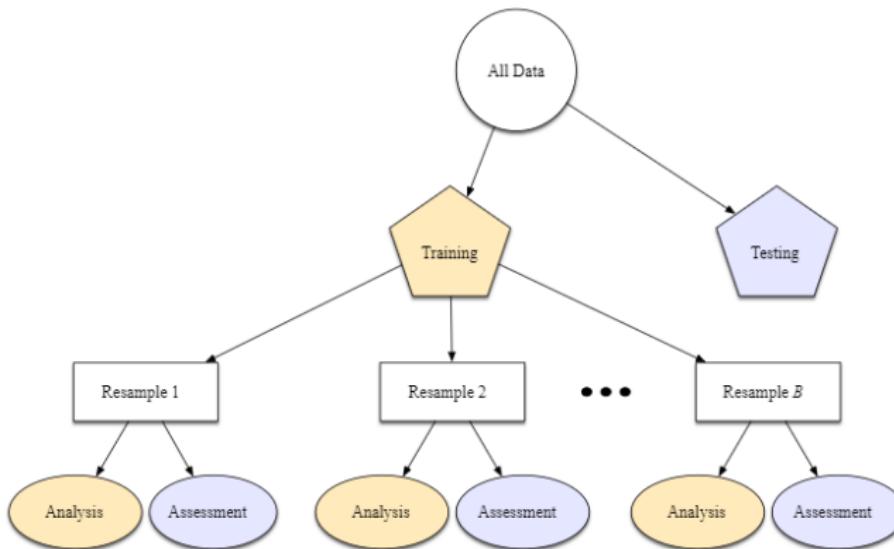
A test set should be avoided only when the data are pathologically small.

- TMWR, Section 5.2

What about a validation set?

- Would like to avoid overfitting (where the models do much better on the training set samples than you do on the test set)
- Idea is to hold back a validation set of data to measure performance while training prior to moving on with a model to the test set.
- This is really just a special case of a resampling method used on the training set, as described in TMWR section 10 (see next slide).

From TMWR, Section 10.2



Resampling is only conducted on the training set. The test set is not involved. For each iteration of resampling, the data are partitioned into two subsamples:

- The model is fit with the **analysis set**.
- The model is evaluated with the **assessment set**.

Stage 3: Pre-Processing the Data



recipes

recipes is a tidy interface to data pre-processing tools for feature engineering. [Go to package ...](#)

We'll build a **recipe** for our pre-modeling work. This might include:

- establishing the roles (outcome, predictors, identifiers) for variables
- pre-processing steps for predictors (feature engineering)
 - transforming predictors, including all of our usual power transformations, but also centering, scaling or normalizing and more complex mutations
 - creating dummy (indicator) variables for categorical data
 - dealing with factors and factor levels
 - including interactions, polynomials or splines
 - filtering out variables with zero variance
 - dealing with missing data via imputation or removal

<https://www.tidymodels.org/find/recipes/> lists all available recipes

Building a Recipe for our modeling

```
hers_rec <-
  recipe(ldl_pch ~ age + ht + ldl + globrat,
         data = hers_new) %>%
  step_impute_bag(all_predictors()) %>%
  step_poly(ldl, degree = 2) %>%
  step_dummy(all_nominal()) %>%
  step_normalize(all_predictors())
```

- ① Specify the roles for the outcome and the predictors.
- ② Impute missing predictors with bagged tree models.
- ③ Use an orthogonal polynomial of degree 2 with the baseline LDL data.
- ④ Form dummy variables to represent all categorical variables.
- ⑤ Normalize (subtract mean and divide by SD) all quantitative predictors.

Column Roles

```
hers_rec <-
  recipe(ldl_pch ~ age + ht + ldl + globrat,
        data = hers_new)
```

- Everything to the left of the ~ is an outcome.
- Everything to the right of the ~ is a predictor.

Sometimes we want to assign other roles, like “id” for an important identifier that isn’t either a predictor or an outcome, or “split” for a splitting variable.

- Any character string can be a role, and columns can have multiple roles
- `add_role()`, `remove_role()` and `update_role()` functions are helpful

Common steps used in building a recipe (1/5)

- Power Transformations of Predictors
 - `step_log(x1, base = 10)` (default base is `exp(1)`), `step_sqrt`, `step_inverse`
 - `step_BoxCox()` will transform predictors using a simple Box-Cox transformation to make them more symmetric (remember this does require a strictly positive variable, and will be something we'd use more for an outcome using the residuals for a statistical model).
 - `step_YeoJohnson()` uses the Yeo-Johnson transformation (again, typically on the outcome model) which is like Box-Cox but doesn't require the input variables to be strictly positive.
- `step_logit` and `step_invlogit`
- Non-Linear Terms for Quantitative Predictors
 - `step_poly()` produces orthogonal polynomial basis functions
 - `step_ns(x5, deg_free = 10)` from the `splines` package can create things called natural splines - the number of spline terms is a tuning parameter, `step_bs()` adds B-spline basis functions

Common steps used in building a recipe (2/5)

- Dealing with Categorical Predictors
 - `step_dummy(all_nominal())` which converts all factor or categorical variables into indicator (also called dummy) variables: numeric variables which take 1 and 0 as values to encode the categorical information
 - Other helpful selectors: `all_numeric()`, `all_predictors()` and `all_outcomes()`
 - If you want to select specific variables, you could use `step_dummy(x2, x3)`
 - `step_relevel()` reorders the provided factor columns so that a level you specify is first (the baseline)
 - If you have ordered factors in R, try `step_unorder()` to convert to regular factors or `step_ordinalscore()` to map specific numeric values to each factor level

Common steps used in building a recipe (3/5)

- Dealing with Categorical Predictors (continued)
 - `step_unknown()` to change missing values in a categorical variable to a dedicated factor level
 - `step_novel()` creates a new factor level that may be encountered in future data
 - `step_other()` converts infrequent values to a catch-all labeled “other” using a threshold
 - `step_other(x5, threshold = 0.05)` places bottom 5% of data in `x5` into “other”.
- Create Interaction Terms
 - `step_interact(~ interaction terms)` can be used to set up interactions
- Filter rows?
 - `step_filter()` can be used to filter rows using dplyr tools

Common steps used in building a recipe (4/5)

- `step_mutate()` can be used to conduct a variety of basic operations
- `step_ratio()` can be used to create ratios of current variables
- Centering and Scaling Predictors
 - `step_normalize()` to center and scale quantitative predictors
 - `step_center()` just centers predictors
 - `step_scale()` just scales numeric data and
 - `step_range()` to scale numeric data to a specific range
- Zero Variance Filters
 - `step_zv()` is the zero variance filter which removes variables that contain only a single value.
 - `step_nzv()` removes variables with very few unique values or for whom the ratio of the frequency of the most common value to the second most common value is large

Common steps used in building a recipe (5/5)

- Step options for imputation include things like
 - `step_meanimpute()` and `step_medianimpute()` to impute with mean or median,
 - `step_modelimpute()` to impute nominal data using the most common value,
 - `step_bagimpute()` for imputation via bagged trees,
 - `step_knnimpute()` to impute via k-nearest neighbors
- `step_naomit()` can be used to remove observations with missing values

<https://www.tidymodels.org/find/recipes/> lists all available recipes

Stage 4: Specify lm modeling engine for fit1



parsnip

parsnip is a tidy, unified interface to models that can be used to try a range of models without getting bogged down in the syntactical minutiae of the underlying packages. [Go to package ...](#)

```
hers_lm_model <- linear_reg() %>% set_engine("lm")
```

Other available engines for linear regression include:

- stan to fit Bayesian models
- spark
- keras

All parsnip models can be found at

<https://www.tidymodels.org/find/parsnip/>

Stage 4: Specify stan modeling engine for fit2

As an alternative, we'll often consider a Bayesian linear regression model as fit with the "stan" engine. This requires the pre-specification of a prior distribution for the coefficients, for instance:

```
prior_dist_int <- rstanarm::student_t(df = 1)
prior_dist_preds <- rstanarm::normal(0, 5)

hers_stan_model <- linear_reg() %>%
  set_engine("stan",
             prior_intercept = prior_dist_int,
             prior = prior_dist_preds)
```

Stage 5: Create a workflow for the lm model



workflows

workflows bundle your pre-processing, modeling, and post-processing together. [Go to package ...](#)

```
hers_lm_wf <- workflow() %>%  
  add_model(hers_lm_model) %>%  
  add_recipe(hers_rec)
```

Fit the lm model to the training sample

```
fit1 <- fit(hers_lm_wf, hers_train)
```

We'll show the fit1 results on the next slide.

```
> fit1
== workflow [trained] =====
Preprocessor: Recipe
Model: linear_reg()

-- Preprocessor -----
4 Recipe Steps

* step_bagimpute()
* step_poly()
* step_dummy()
* step_normalize()

-- Model -----
Call:
stats::lm(formula = ..y ~ ., data = data)

Coefficients:
(Intercept)           age      ldl_poly_1      ldl_poly_2
                 -6.0248     -1.6396     -8.0728     2.5596
ht_placebo       globrat_fair    globrat_good    globrat_poor
                 5.3921     -1.3379     -1.8050     -0.7685
globrat_very.good
                 -1.4063
```

Tidy the coefficients for fit1?



broom

broom converts the information in common statistical R objects into user-friendly, predictable formats. [Go to package ...](#)

term	estimate	std.error	conf.low	conf.high
(Intercept)	-6.368	0.523	-7.394	-5.342
age	-0.772	0.525	-1.802	0.258
ldl_poly_1	-7.857	0.524	-8.884	-6.829
ldl_poly_2	2.236	0.524	1.208	3.265
ht_placebo	4.893	0.523	3.866	5.920
globrat_fair	-0.723	1.002	-2.689	1.242
globrat_good	-1.569	1.196	-3.915	0.777
globrat_poor	-1.123	0.572	-2.244	-0.001
globrat_very.good	-1.390	1.114	-3.574	0.795

Want to glance at the fit1 summaries?

```
fit1 %>% extract_fit_parsnip() %>%  
  glance() %>% select(1:6) %>% kable(dig = 3)
```

r.squared	adj.r.squared	sigma	statistic	p.value	df
0.18	0.175	20.522	41.923	0	8

```
fit1 %>% extract_fit_parsnip() %>%  
  glance() %>% select(7:12) %>% kable(dig = 1)
```

logLik	AIC	BIC	deviance	df.residual	nobs
-6833.8	13687.6	13741	644801.3	1531	1540

Stage 5: Create a workflow for the stan model

```
hers_stan_wf <- workflow() %>%  
  add_model(hers_stan_model) %>%  
  add_recipe(hers_rec)
```

Fit the stan model to the training sample

```
set.seed(43202)  
fit2 <- fit(hers_stan_wf, hers_train)
```

We'll show the fit2 results on the next slide.

```
> fit2
== Workflow [trained] =====
Preprocessor: Recipe
Model: linear_reg()

-- Preprocessor -----
4 Recipe Steps

* step_bagimpute()
* step_poly()
* step_dummy()
* step_normalize()

-- Model -----
stan_glm
  family: gaussian [identity]
  formula: ...y ~ .
  observations: 1444
  predictors: 9
  -----
              Median MAD_SD
(Intercept)    -5.9    0.6
age            -1.6    0.5
ld1_poly_1     -8.0    0.6
ld1_poly_2      2.5    0.6
ht_placebo      5.3    0.6
globrat_fair    -1.1    1.0
globrat_good     -1.5    1.1
globrat_poor     -0.7    0.6
globrat_very.good -1.1    1.1

Auxiliary parameter(s):
  Median MAD_SD
sigma 20.8    0.4
  -----
* For help interpreting the printed output see ?print.stanreg
* For info on the priors used see ?prior_summary.stanreg
```

Tidy the fit2 coefficients?

The stan model requires the broom.mixed package to tidy the fit.

```
broom.mixed::tidy(fit2, conf.int = T) %>% kable(dig = 3)
```

term	estimate	std.error	conf.low	conf.high
(Intercept)	-6.286	0.521	-7.150	-5.445
age	-0.768	0.515	-1.579	0.104
ldl_poly_1	-7.761	0.527	-8.627	-6.902
ldl_poly_2	2.219	0.514	1.374	3.112
ht_placebo	4.833	0.540	3.989	5.694
globrat_fair	-0.634	0.933	-2.161	0.931
globrat_good	-1.407	1.112	-3.260	0.397
globrat_poor	-1.081	0.571	-1.988	-0.179
globrat_very.good	-1.258	1.077	-2.969	0.482

Stage 6: Compare the coefficients of the fits

```
coefs_lm <- tidy(fit1, conf.int = TRUE) %>%
  select(term, estimate, conf.low, conf.high) %>%
  mutate(mod = "lm")

coefs_stan <- tidy(fit2, conf.int = TRUE) %>%
  select(term, estimate, conf.low, conf.high) %>%
  mutate(mod = "stan")

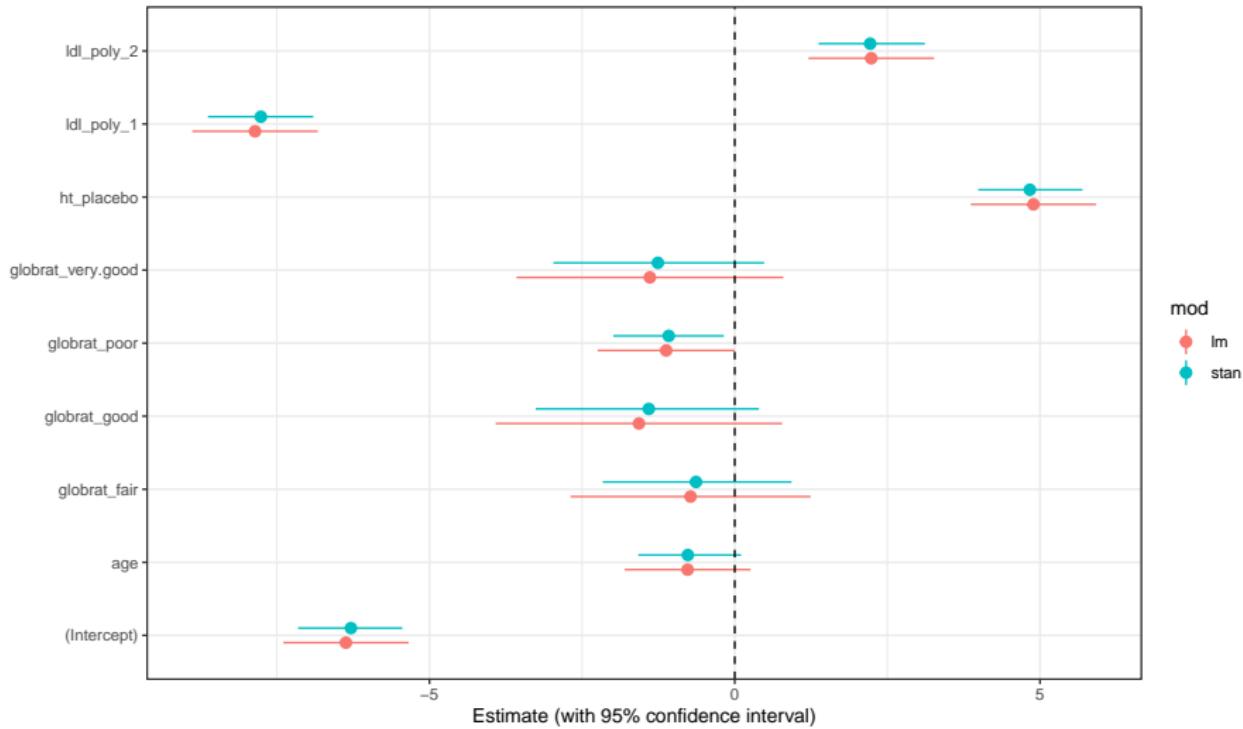
coefs_comp <- bind_rows(coefs_lm, coefs_stan)
```

Graph the coefficients from the two models

```
ggplot(coefs_comp, aes(x = term, y = estimate, col = mod,  
                      ymin = conf.low, ymax = conf.high)) +  
  geom_point(position = position_dodge2(width = 0.4)) +  
  geom_pointrange(position = position_dodge2(width = 0.4)) +  
  geom_hline(yintercept = 0, lty = "dashed") +  
  coord_flip() +  
  labs(x = "", y = "Estimate (with 95% confidence interval)",  
       title = "Comparing the lm and stan model coefficients")
```

Graph the coefficients from the two models

Comparing the lm and stan model coefficients



Stage 7. Assess performance in the training data



yardstick

yardstick measures the effectiveness of models using performance metrics.

[Go to package ...](#)

Available regression performance metrics include:

- `rsq` (r-squared, via correlation - always between 0 and 1)
- `rmse` (root mean squared error)
- `mae` (mean absolute error)
- `rsq_trad` (r-squared, calculated via sum of squares)

but there are many, many more. Let's select two...

```
mets <- metric_set(rsq, rmse)
```

Make predictions using fit1 in training sample

```
lm_pred_train <-  
  predict(fit1, hers_train) %>%  
  bind_cols(hers_train %>% dplyr::select(ldl_pch))  
  
# remember  
mets <- metric_set(rsq, rmse)  
  
lm_res_train <-  
  mets(lm_pred_train, truth = ldl_pch, estimate = .pred)
```

We'll see the results in a moment.

Make predictions using fit2 in training sample

```
stan_pred_train <-  
  predict(fit2, hers_train) %>%  
  bind_cols(hers_train %>% select(ldl_pch))  
  
# remember  
mets <- metric_set(rsq, rmse)  
  
stan_res_train <-  
  mets(stan_pred_train, truth = ldl_pch, estimate = .pred)
```

We'll see the results from each fit on the next slide.

fit1 and fit2 performance in the training sample

from fit1 with lm:

```
lm_res_train %>% kable()
```

.metric	.estimator	.estimate
rsq	standard	0.1796985
rmse	standard	20.4622118

from fit2 with stan:

```
stan_res_train %>% kable()
```

.metric	.estimator	.estimate
rsq	standard	0.1796909
rmse	standard	20.4626978

What about adjusted R^2 ?

The yardstick package doesn't use adjusted R^2 .

- tidyverse wants you to compute performance on a separate data set for comparing models rather than doing what adjusted R^2 tries to do, which is evaluate the model on the same data as were used to fit the model.

I wrote more on the adjusted R^2 statistic in the Class 11 README.

Stage 8. Compare model performance on test data

```
lm_pred_test <-
  predict(fit1, hers_test) %>%
  bind_cols(hers_test %>% dplyr::select(ldl_pch))

lm_res_test <-
  mets(lm_pred_test, truth = ldl_pch, estimate = .pred)

stan_pred_test <-
  predict(fit2, hers_test) %>%
  bind_cols(hers_test %>% select(ldl_pch))

stan_res_test <-
  mets(stan_pred_test, truth = ldl_pch, estimate = .pred)
```

fit1 and fit2 performance in the test sample

from fit1 with lm:

```
lm_res_test %>% kable()
```

.metric	.estimator	.estimate
rsq	standard	0.199772
rmse	standard	21.082747

from fit2 with stan:

```
stan_res_test %>% kable()
```

.metric	.estimator	.estimate
rsq	standard	0.1997987
rmse	standard	21.0858904

Where to Learn More

- Tidy Modeling with R by Max Kuhn and Julia Silge.
 - The Basics section (Chapters 4-9) as well as chapters 10-11 were my main tools for learning about these ideas.
- Julia Silge has many nice videos on YouTube demonstrating various things that `tidymodels` can accomplish.
 - I've recommended several in the Class 10 README.

Next Time

We'll apply ideas from the `tidymodels` framework to fit a logistic regression model.

432 Class 12 Slides

thomaselove.github.io/432

2022-02-17

Today's Agenda

Some reminders and loose ends

- for linear regression models
- for logistic regression models

We'll return to `tidymodels` next time.

Setup

```
library(here); library(knitr)
library(magrittr); library(janitor)
library(naniar); library(equatiomatic)
library(GGally); library(broom)
library(rms)

library(tidyverse)

theme_set(theme_bw())
```

Linear Regression

The day12 Data Set

These data are simulated.

```
dat12 <- readRDS(here("data/dat12.Rds"))

names(dat12)

[1] "subj"     "result"   "sur_s"    "typeA"    "sbp"      "sroh"

miss_case_table(dat12)

# A tibble: 1 x 3
  n_miss_in_case n_cases pct_cases
            <int>    <int>     <dbl>
1                  0       400      100
```

The dat12 codebook

Variable	Description	Type
result	Our outcome (0-500 scale)	quant.
sur_s	Survey sur_s (0-200 scale)	quant.
typeA	Type A (No or Yes)	binary
sbp	Systolic Blood Pressure	quant.
sroh	Self-Reported Health (E/VG/G/F)	4 cats.

Summary of dat12

```
summary(dat12 %>% select(-subj))
```

result	sur_s	typeA	sbp
Min. : 46.0	Min. : 39.0	No :203	Min. : 85.0
1st Qu.:160.0	1st Qu.: 87.0	Yes:197	1st Qu.:132.0
Median :168.0	Median :101.0		Median :147.0
Mean :168.8	Mean :100.2		Mean :148.1
3rd Qu.:177.0	3rd Qu.:114.0		3rd Qu.:165.0
Max. :483.0	Max. :185.0		Max. :215.0

sroh

E : 68

VG:147

G :139

F : 46

OLS Model for ‘result’ without Non-Linear Terms

Variable	Description
result	Our outcome (0-500 scale)
sur_s	Survey sur_s (0-200 scale)
typeA	Type A (No or Yes)
sbp	Systolic Blood Pressure
sroh	Self-Reported Health (E/VG/G/F)

```
d <- datadist(dat12)
options(datadist = "d")

modA <- ols(result ~ sur_s + typeA + sbp + sroh,
             data = dat12, x = TRUE, y = TRUE)
```

How many degrees of freedom does the model modA use?

Model modA

modA

```
> modA
Linear Regression Model

ols(formula = result ~ sur_s + typeA + sbp + sroh, data = dat12,
  x = TRUE, y = TRUE)

      Model Likelihood Discrimination
      Ratio Test      Indexes
Obs     400    LR chi2    296.82    R2      0.524
sigma16.8657   d.f.        6    R2 adj  0.517
d.f.     393    Pr(> chi2) 0.0000    g      19.692

Residuals

      Min       1Q     Median       3Q       Max
-69.1043 -7.1600 -0.7081  6.0485 250.0511

      Coef     S.E.     t     Pr(>|t|)
Intercept 134.0500 7.1558 18.73 <0.0001
sur_s      0.7180 0.0411 17.48 <0.0001
typeA=Yes 10.1832 1.6977  6.00 <0.0001
sbp       -0.2292 0.0365 -6.28 <0.0001
sroh=VG   -7.7635 2.4803 -3.13 0.0019
sroh=G    -11.1855 2.5028 -4.47 <0.0001
sroh=F    -12.9429 3.2348 -4.00 <0.0001
```

ANOVA results for modA

```
anova(modA)
```

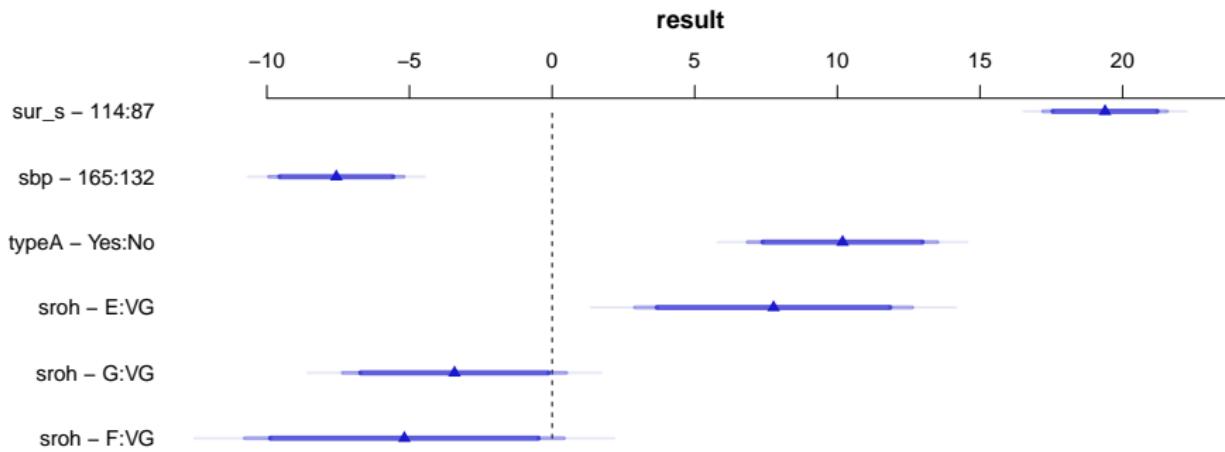
Analysis of Variance

Response: result

Factor	d.f.	Partial SS	MS	F	P
sur_s	1	86894.325	86894.3250	305.48	<.0001
typeA	1	10233.702	10233.7022	35.98	<.0001
sbp	1	11222.442	11222.4417	39.45	<.0001
sroh	3	6825.387	2275.1291	8.00	<.0001
REGRESSION	6	122994.902	20499.1504	72.07	<.0001
ERROR	393	111789.458	284.4515		

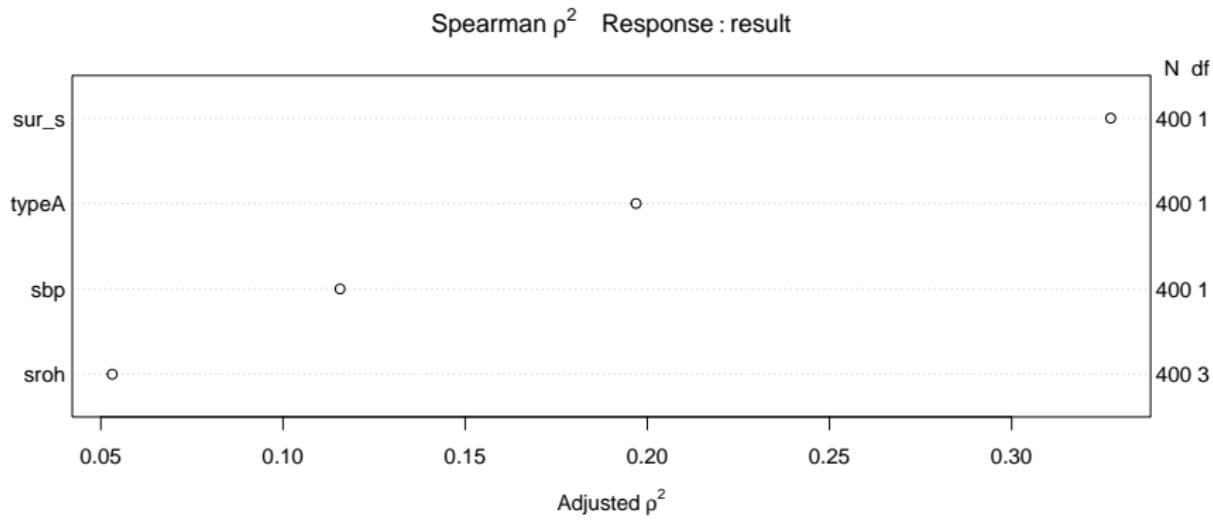
Plot Effect Sizes

```
plot(summary(modA))
```



Consider Potential Non-Linear Terms

```
plot(spearman2(result ~ sur_s + typeA + sbp + sroh,  
                data = dat12))
```



Using the Spearman plot as a guide...

Variable	Description	Adj. Spearman ρ^2
sur_s	Survey sur_s (0-200 scale)	Highest
typeA	Type A (No or Yes)	2nd Highest
sbp	Systolic Blood Pressure	3rd Highest
sroh	Self-Reported Health (E/VG/G/F)	Lowest

Using Polynomials or Splines

- Can we build a (polynomial or spline) non-linear term that will add one more degree of freedom to our original main-effects model?
- What if we can afford 2 additional df? Or 3?

Using Interaction terms

- How many df does the best categorical-categorical interaction use?
- How many df does the best categorical-quantitative interaction use?

Adding Polynomial Terms in sur_s

We'll look at a quadratic, then a cubic polynomial...

```
modP2 <- ols(result ~ pol(sur_s, 2) + typeA + sbp + sroh,  
               data = dat12, x = TRUE, y = TRUE)  
modP3 <- ols(result ~ pol(sur_s, 3) + typeA + sbp + sroh,  
               data = dat12, x = TRUE, y = TRUE)
```

Quadratic Polynomial adds 1 df to modA's 6

modP2

```
> modP2
Linear Regression Model

ols(formula = result ~ pol(sur_s, 2) + typeA + sbp + sroh, data = dat12,
    x = TRUE, y = TRUE)

      Model Likelihood Discrimination
              Ratio Test      Indexes
Obs      400   LR chi2     353.07    R2       0.586
sigma15.7405 d.f.          7   R2 adj    0.579
d.f.      392   Pr(> chi2) 0.0000    g       19.680

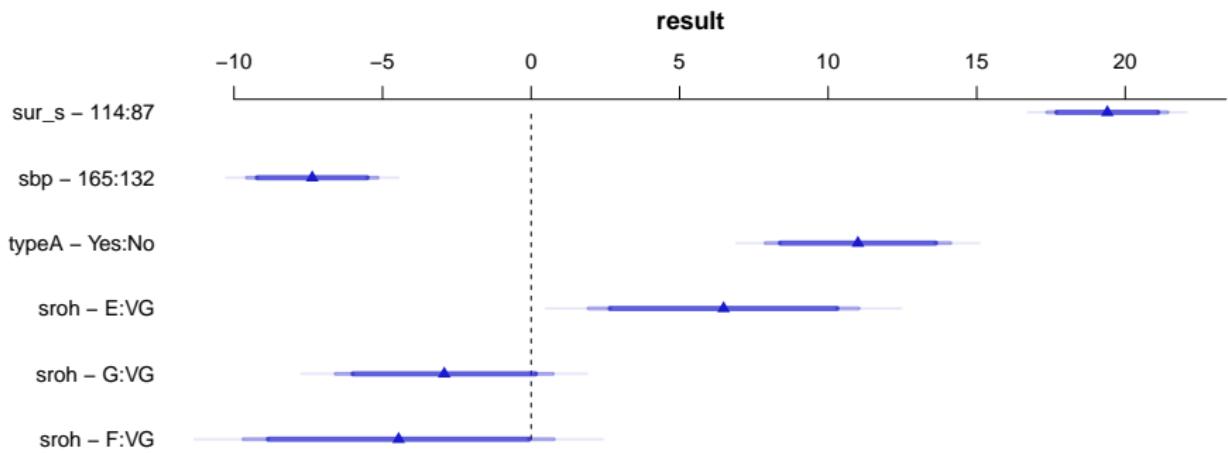
Residuals

      Min      1Q Median      3Q      Max
-100.7397 -7.3692  0.6981  7.5392 188.5970

      Coef    S.E.      t    Pr(>|t|)    
Intercept 222.3626 13.2799 16.74 <0.0001
sur_s     -1.1718  0.2486 -4.71 <0.0001
sur_s^2    0.0094  0.0012  7.69 <0.0001
typeA=Yes 11.0031  1.5881  6.93 <0.0001
sbp      -0.2232  0.0341 -6.55 <0.0001
sroh=VG   -6.4802  2.3209 -2.79 0.0055
sroh=G    -9.4029  2.3473 -4.01 <0.0001
sroh=F   -10.9390  3.0302 -3.61 0.0003
```

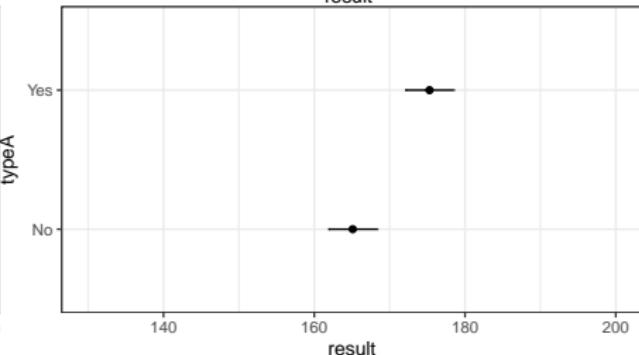
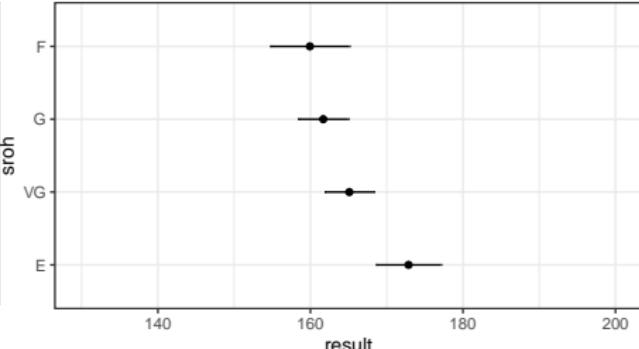
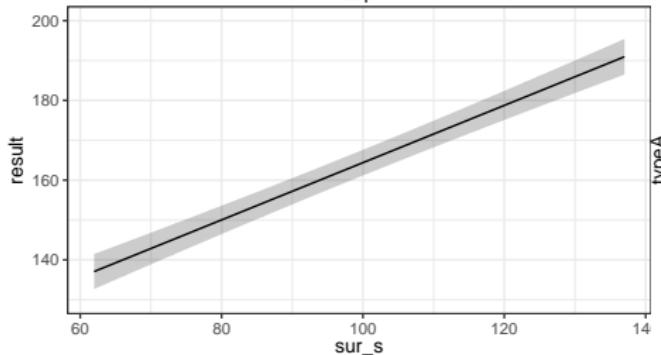
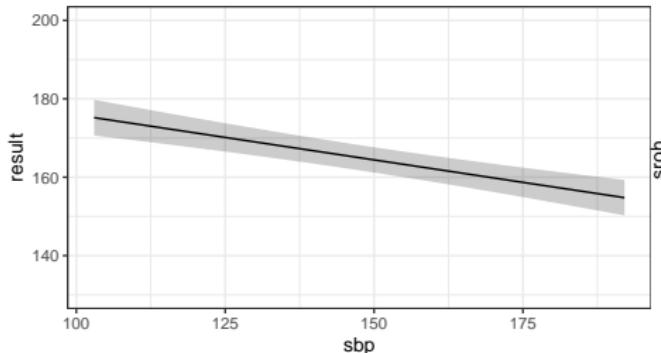
Plot Effect Sizes

```
plot(summary(modP2))
```



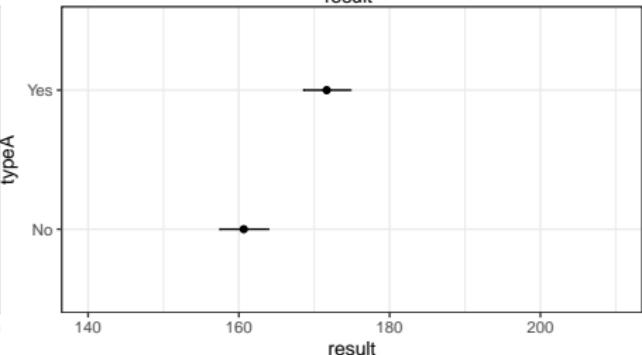
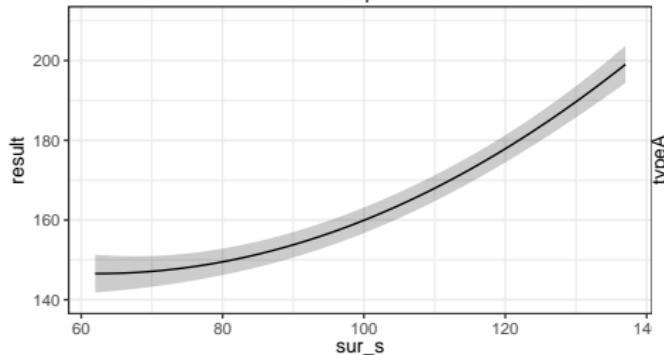
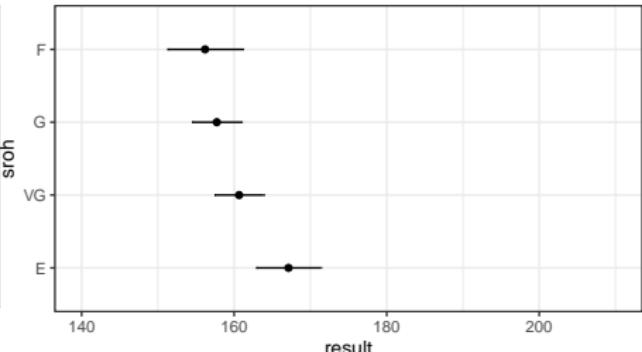
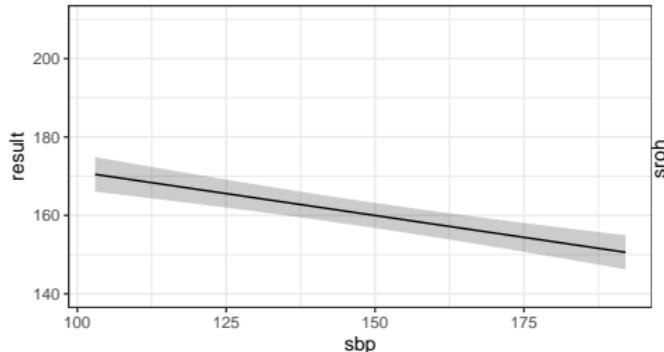
What does model modA look like?

```
ggplot(Predict(modA))
```



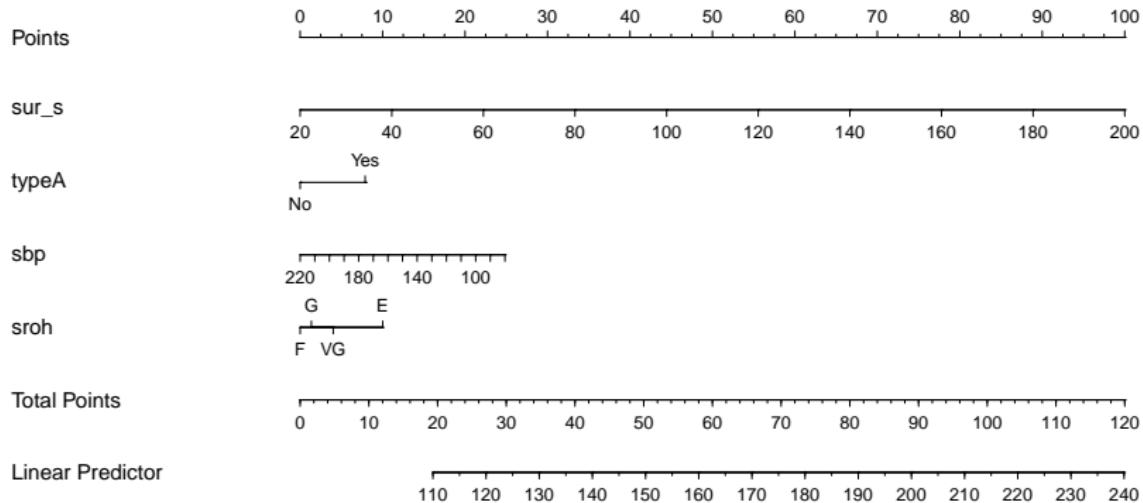
What does model modP2 look like?

```
ggplot(Predict(modP2))
```



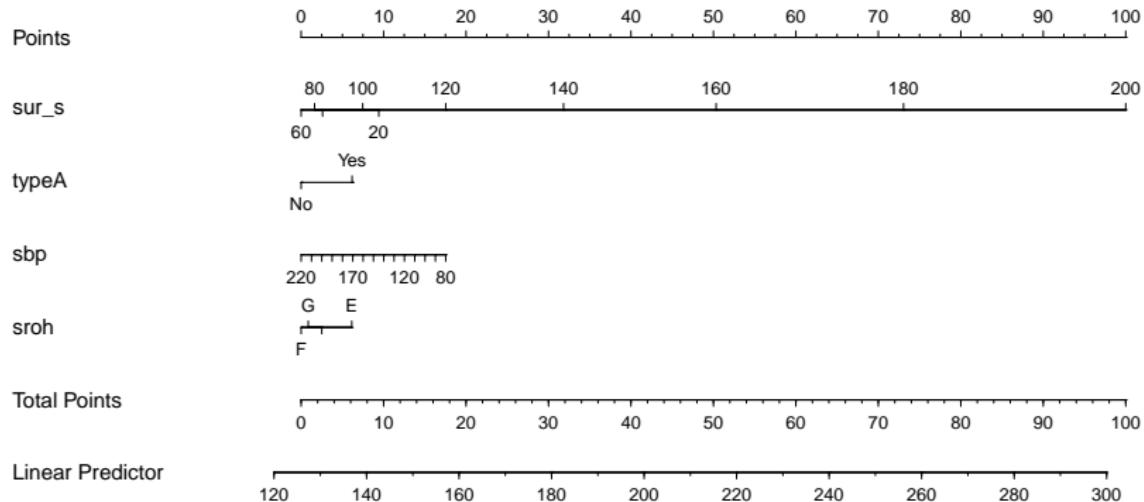
Nomogram for model modA

```
plot(nomogram(modA))
```



Nomogram for model modP2

```
plot(nomogram(modP2))
```



Do the non-linear terms in modP2 do much?

```
anova(modP2)
```

Analysis of Variance					Response: result	
Factor	d.f.	Partial SS	MS	F	P	
sur_s	2	101560.602	50780.3008	204.95	<.0001	
Nonlinear	1	14666.277	14666.2767	59.19	<.0001	
typeA	1	11894.005	11894.0047	48.01	<.0001	
sbp	1	10636.075	10636.0753	42.93	<.0001	
sroh	3	4795.273	1598.4244	6.45	3e-04	
REGRESSION	7	137661.179	19665.8827	79.37	<.0001	
ERROR	392	97123.181	247.7632			

Do the non-linear terms in modP2 help much?

AIC(modA); BIC(modA)

d.f.

3404.314

d.f.

3436.246

AIC(modP2); BIC(modP2)

d.f.

3350.059

d.f.

3385.982

Cubic (degree 3) polynomial adds 2 df to modA's 6

modP3

```
> modP3
Linear Regression Model

ols(formula = result ~ pol(sur_s, 3) + typeA + sbp + sroh, data = dat12,
    x = TRUE, y = TRUE)

      Model Likelihood Discrimination
          Ratio Test      Indexes
Obs     400   LR chi2    1227.39    R2      0.954
sigma5.2836 d.f.           8    R2 adj   0.953
d.f.     391   Pr(> chi2)  0.0000    g      17.754

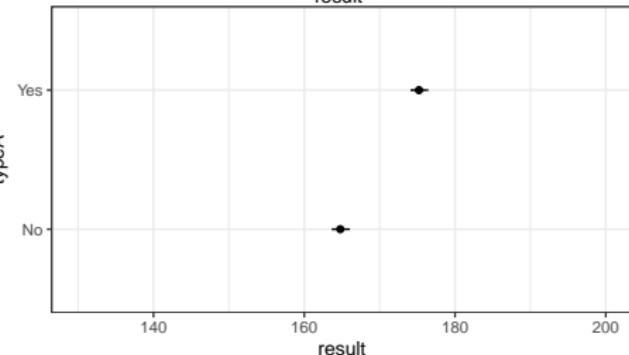
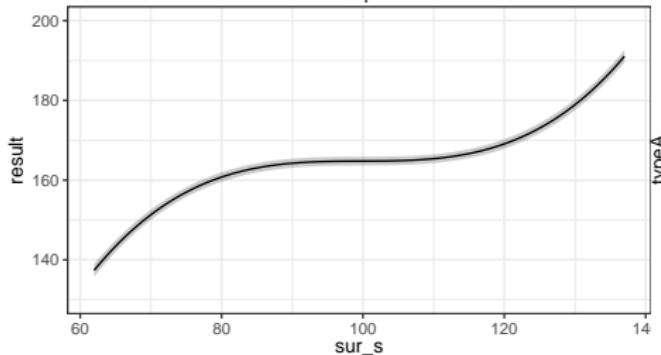
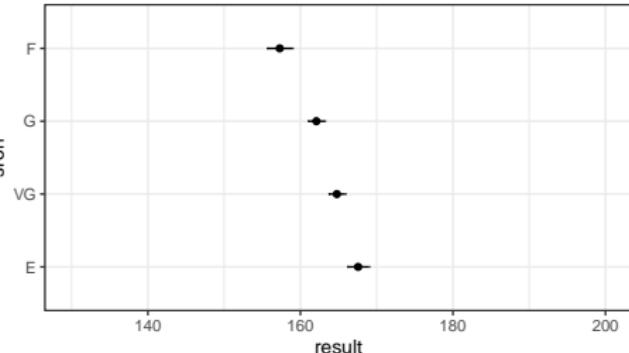
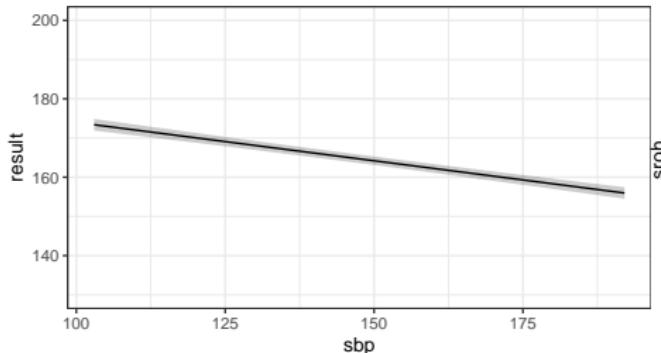
Residuals

      Min      1Q      Median      3Q      Max
-20.9404 -3.5022   0.1353   3.2907  14.6336

      Coef      S.E.      t      Pr(>|t|)
Intercept -304.4469 10.4759 -29.06 <0.0001
sur_s       15.0487  0.3036  49.57 <0.0001
sur_s^2     -0.1508  0.0029 -51.79 <0.0001
sur_s^3      0.0005  0.0000  55.57 <0.0001
typeA=Yes   10.4406  0.5332  19.58 <0.0001
sbp        -0.1955  0.0114 -17.08 <0.0001
sroh=VG     -2.7809  0.7819  -3.56 0.0004
sroh=G      -5.4697  0.7911  -6.91 <0.0001
sroh=F     -10.2759  1.0172 -10.10 <0.0001
```

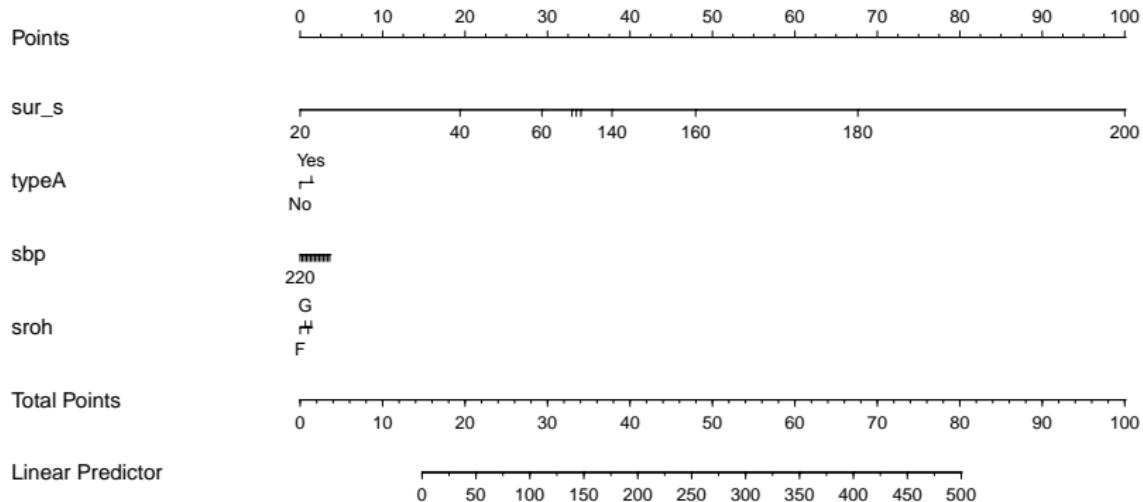
What does model modP3 look like?

```
ggplot(Predict(modP3))
```



Nomogram for model modP3

```
plot(nomogram(modP3))
```



How about a restricted cubic spline in cigs?

```
modC3 <- ols(result ~ rcs(sur_s,3) + typeA + sbp + sroh,  
               data = dat12, x = TRUE, y = TRUE)  
modC4 <- ols(result ~ rcs(sur_s,4) + typeA + sbp + sroh,  
               data = dat12, x = TRUE, y = TRUE)  
modC5 <- ols(result ~ rcs(sur_s,5) + typeA + sbp + sroh,  
               data = dat12, x = TRUE, y = TRUE)
```

RCS with 3 knots adds 1 df to modA's 6

modC3

```
> modC3
Linear Regression Model

ols(formula = result ~ rcs(sur_s, 3) + typeA + sbp + sroh, data = dat12,
    x = TRUE, y = TRUE)

      Model Likelihood Discrimination
          Ratio Test      Indexes
Obs     400   LR chi2    310.56    R2      0.540
sigma16.5997 d.f.        7    R2 adj  0.532
d.f.     392   Pr(> chi2) 0.0000      g    19.734

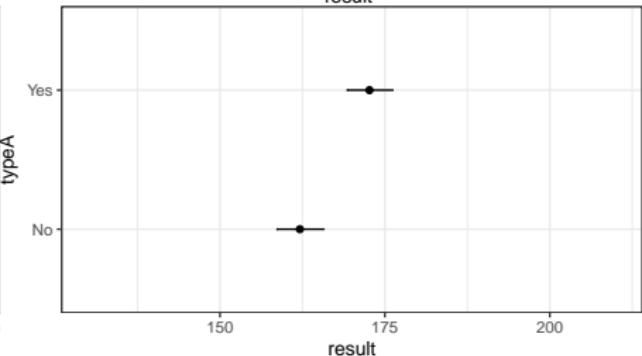
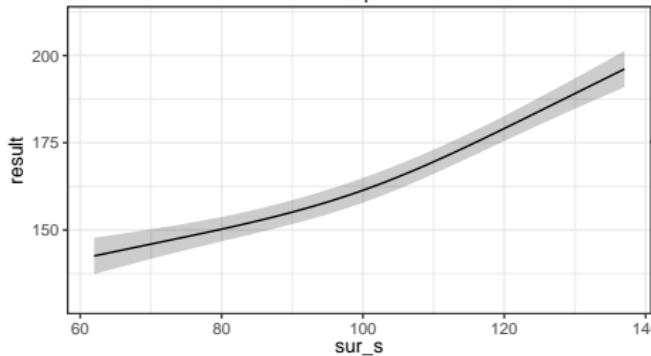
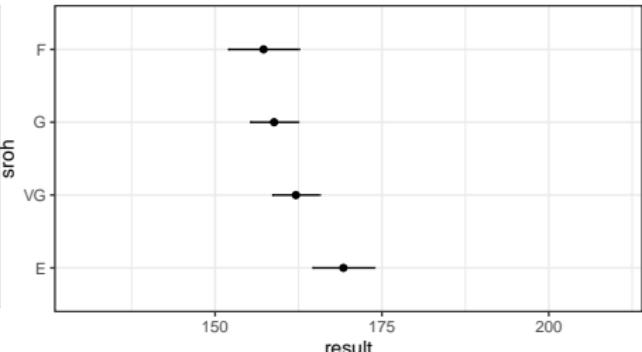
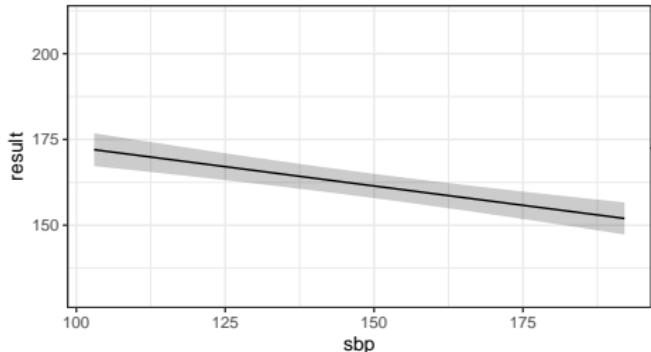
Residuals

      Min       1Q     Median       3Q       Max
-81.57619 -7.35312 -0.04281  6.90506 231.78407

      Coef    S.E.     t    Pr(>|t|) 
Intercept 156.6095 9.3149 16.81 <0.0001
sur_s      0.4221 0.0896  4.71 <0.0001
sur_s'     0.3544 0.0958  3.70 0.0002
typeA=Yes 10.5500 1.6739  6.30 <0.0001
sbp       -0.2251 0.0359 -6.26 <0.0001
sroh=VG    -7.1198 2.4474 -2.91 0.0038
sroh=G    -10.3836 2.4729 -4.20 <0.0001
sroh=F    -11.9694 3.1947 -3.75 0.0002
```

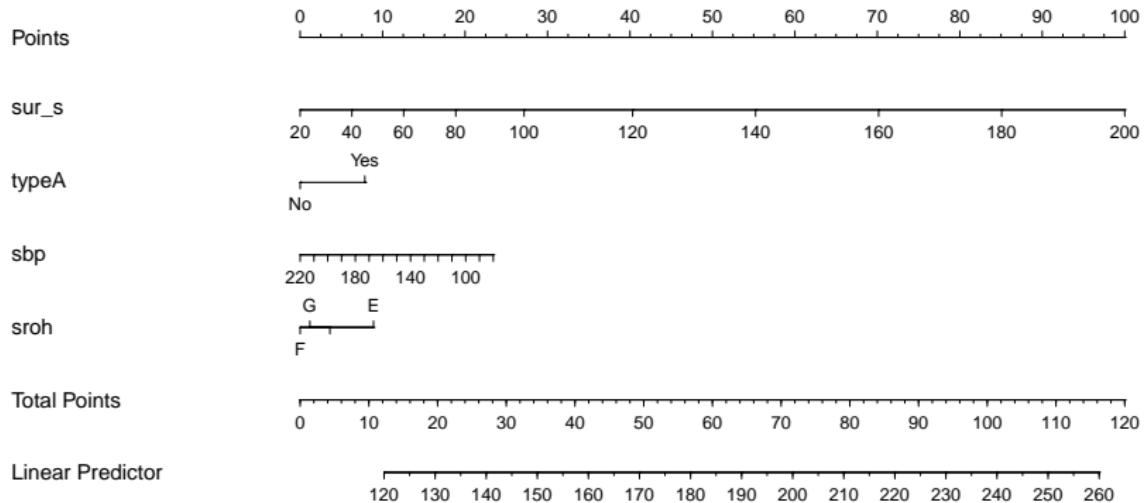
What does model modC3 look like?

```
ggplot(Predict(modC3))
```



What does the nomogram for modC3 look like?

```
plot(nomogram(modC3))
```



Do the non-linear terms help much in modC3?

AIC(modC3); BIC(modC3)

d.f.

3392.579

d.f.

3428.502

AIC(modA); BIC(modA)

d.f.

3404.314

d.f.

3436.246

ANOVA table for modC3?

```
anova(modC3)
```

Analysis of Variance					Response: result	
Factor	d.f.	Partial SS	MS	F	P	
sur_s	2	90667.755	45333.8775	164.52	<.0001	
Nonlinear	1	3773.430	3773.4300	13.69	2e-04	
typeA	1	10945.709	10945.7087	39.72	<.0001	
sbp	1	10806.525	10806.5247	39.22	<.0001	
sroh	3	5820.777	1940.2589	7.04	1e-04	
REGRESSION	7	126768.332	18109.7617	65.72	<.0001	
ERROR	392	108016.028	275.5511			

RCS with 4 knots adds 2 df to modA's 6

modC4

```
> modC4
Linear Regression Model

ols(formula = result ~ rcs(sur_s, 4) + typeA + sbp + sroh, data = dat12,
    x = TRUE, y = TRUE)

              Model Likelihood     Discrimination
                  Ratio Test          Indexes
Obs      400   LR chi2     617.42      R2       0.786
sigma11.3258 d.f.          8      R2 adj     0.782
d.f.      391   Pr(> chi2) 0.0000      g       20.628

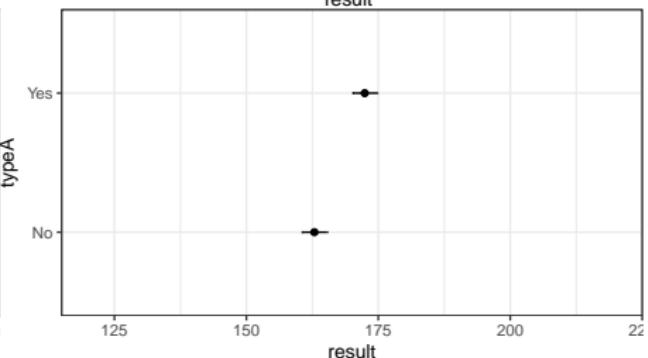
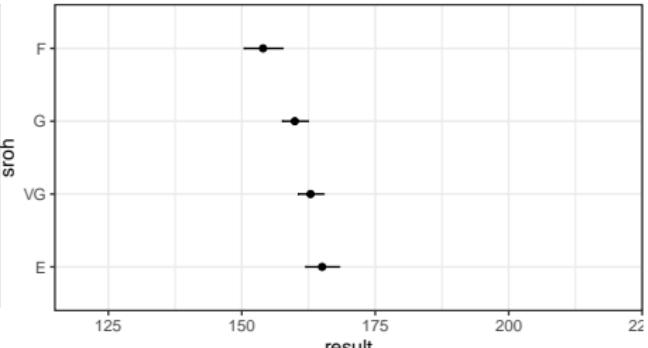
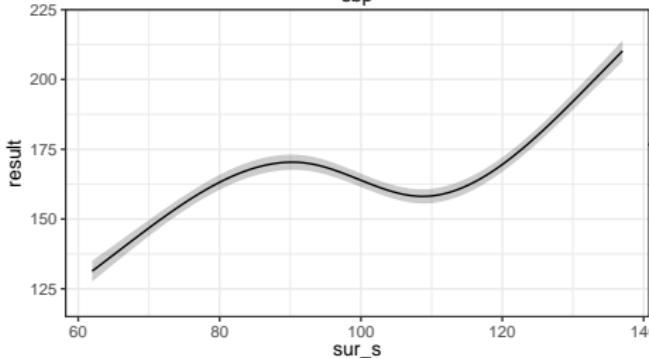
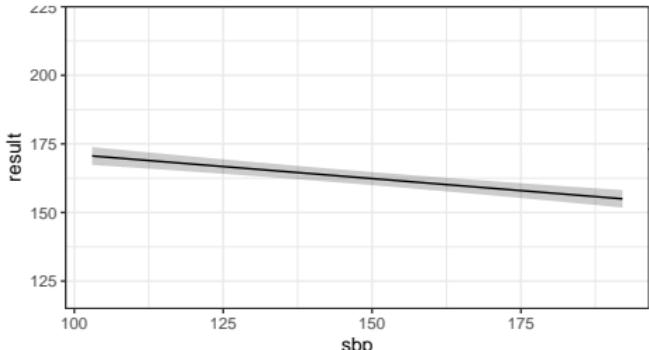
Residuals

    Min      1Q  Median      3Q      Max
-36.3707 -4.5731  0.2394  4.8794 147.0369

             Coef    S.E.      t    Pr(>|t|)    
Intercept 39.4204 8.7154   4.52 <0.0001
sur_s'     1.9340 0.0989  19.55 <0.0001
sur_s''    -4.9038 0.2683 -18.27 <0.0001
sur_s'''   23.5323 1.1447  20.56 <0.0001
typeA=Yes  9.5443 1.1433   8.35 <0.0001
sbp        -0.1753 0.0246  -7.12 <0.0001
sroh=VG    -2.1722 1.6858  -1.29 0.1983
sroh=G     -5.1198 1.7053  -3.00 0.0029
sroh=F     -11.0668 2.1802  -5.08 <0.0001
```

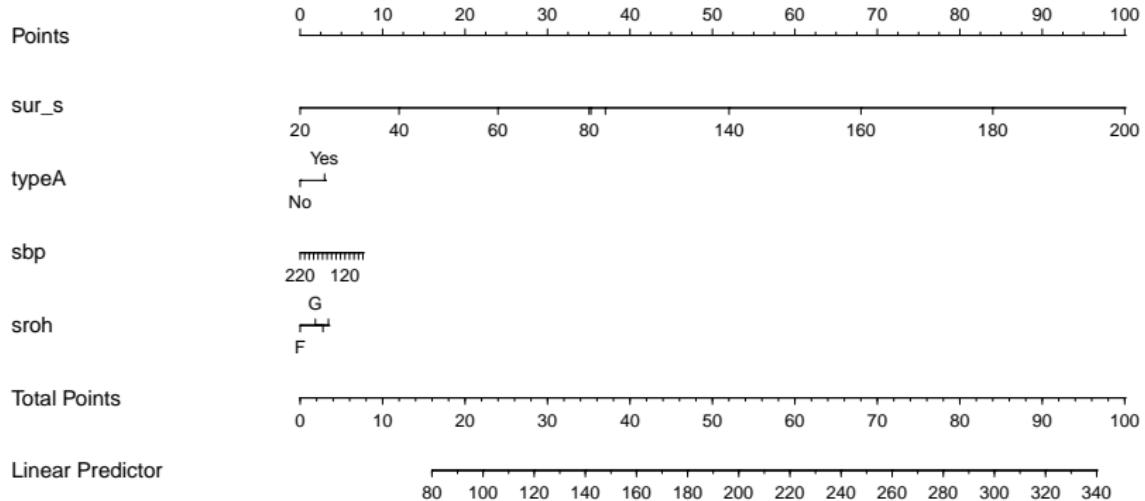
What does model modC4 look like?

```
ggplot(Predict(modC4))
```



What does the nomogram for modC4 look like?

```
plot(nomogram(modC4))
```



RCS with 5 knots adds 3 df to modA's 6

modC5

```
> modC5
Linear Regression Model

ols(formula = result ~ rcs(sur_s, 5) + typeA + sbp + sroh, data = dat12,
    x = TRUE, y = TRUE)

              Model Likelihood     Discrimination
                  Ratio Test      Indexes
obs        400   LR chi2   665.58      R2       0.811
sigma10 0.6778   d.f.          9      R2 adj    0.806
d.f.        390   Pr(> chi2) 0.0000      g       20.398

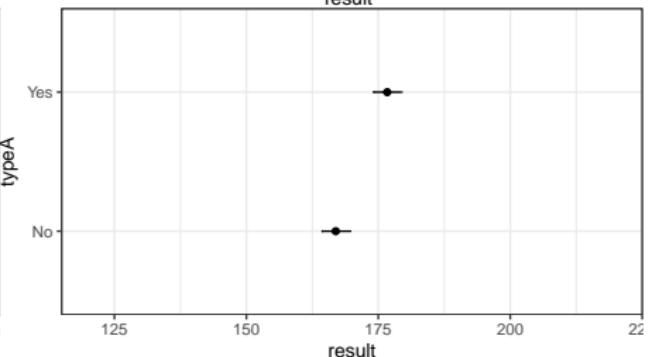
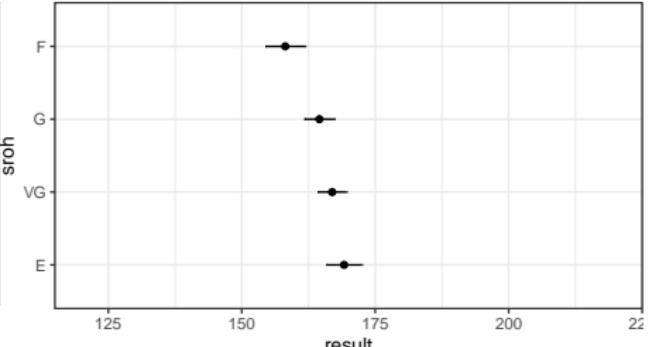
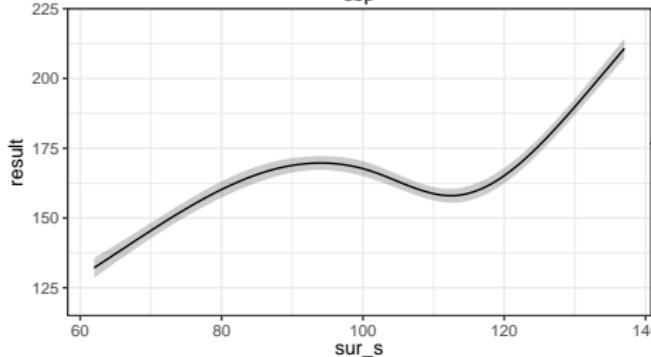
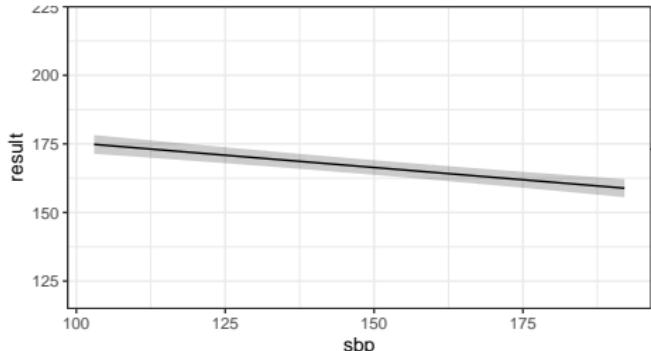
Residuals

    Min      1Q  Median      3Q      Max
-43.8254 -4.1905  0.2477  4.9296 126.8751

              Coef    S.E.      t    Pr(>|t|)
Intercept  57.3276 9.6251  5.96 <0.0001
sur_s      1.6682 0.1186 14.06 <0.0001
sur_s'     -3.2491 0.5682 -5.72 <0.0001
sur_s''    1.6160 3.0831  0.52 0.6005
sur_s'''   27.0995 5.2221  5.19 <0.0001
typeA=Yes  9.7539 1.0783  9.05 <0.0001
sbp       -0.1791 0.0233 -7.70 <0.0001
sroh=VG   -2.2273 1.5891 -1.40 0.1618
sroh=G    -4.6241 1.6095 -2.87 0.0043
sroh=F    -11.0161 2.0555 -5.36 <0.0001
```

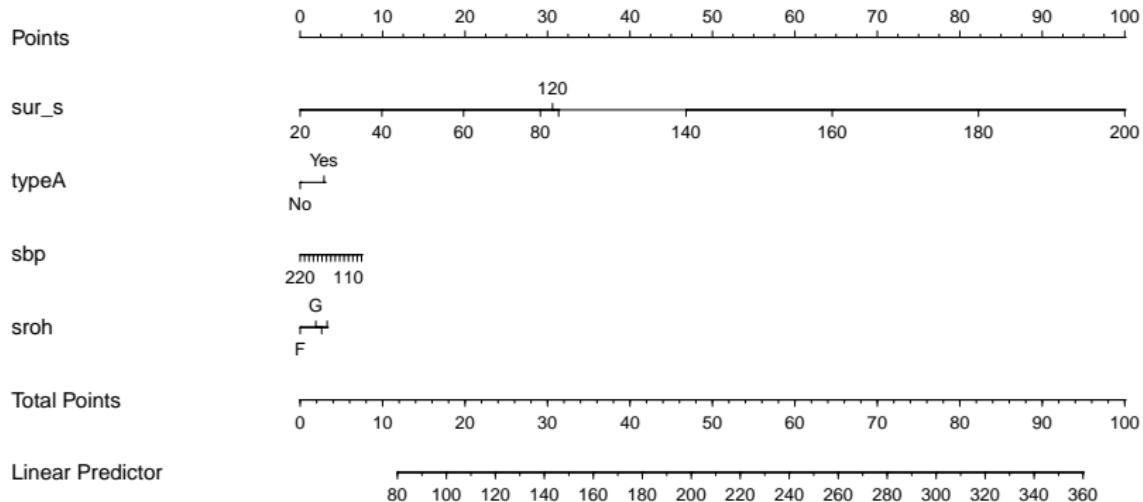
What does model modC5 look like?

```
ggplot(Predict(modC5))
```



What does the nomogram for modC5 look like?

```
plot(nomogram(modC5))
```



Splines and Polynomials with ols (or lrm)

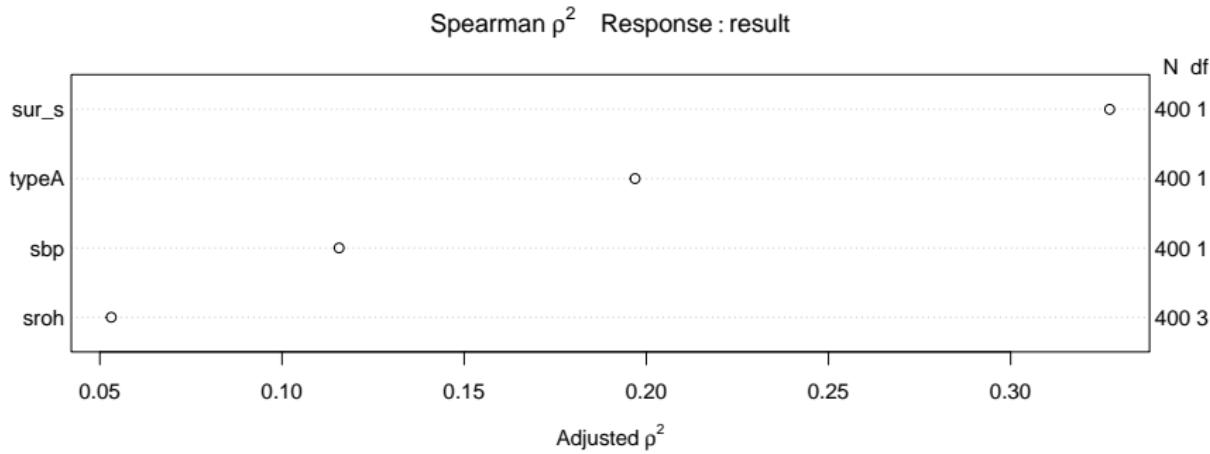
Model	Coeffs.	"Bends"	DF added
Main Effects (modA)	None	None	-
Polynomial, degree 2 (P2)	$\wedge 2$	1	1
Polynomial, degree 3 (P3)	$\wedge 2, \wedge 3$	2	2
RCS, 3 knots (C3)	'	2	1
RCS, 4 knots (C4)	', ''	3	2
RCS, 5 knots (C5)	', '', '''	4	3

- RCS = Restricted Cubic Spline

What about an interaction term instead?

- ① How many df does the best categorical-categorical interaction use?
- ② How many df does the best categorical-quantitative interaction use?

```
plot(spearman2(result ~ sur_s + typeA + sbp + sroh,  
                data = dat12))
```



Models with Interaction Terms

```
# already set up datadist for dat12

modI1 <- ols(result ~ sur_s * typeA + sbp + sroh,
              data = dat12, x = TRUE, y = TRUE)
modI2 <- ols(result ~ rcs(sur_s, 4) + typeA +
              sur_s %ia% typeA + sbp + sroh,
              data = dat12, x = TRUE, y = TRUE)
```

Model modI1 adds how many df to modA?

modI1

```
> modI1
Linear Regression Model

ols(formula = result ~ sur_s * typeA + sbp + sroh, data = dat12)

      Model Likelihood   Discrimination
             Ratio Test           Indexes
Obs       400    LR chi2     312.57      R2        0.542
sigma16.5580    d.f.          7      R2 adj     0.534
d.f.       392    Pr(> chi2) 0.0000      g       19.753

Residuals

      Min        1Q        Median         3Q        Max
-61.0473 -6.8744  -0.7916   6.0512 238.3297

      Coef    S.E.      t    Pr(>|t|) 
Intercept 118.8604 8.0008 14.86 <0.0001
sur_s      0.8596 0.0539 15.96 <0.0001
typeA=Yes 42.5882 8.3362  5.11 <0.0001
sbp       -0.2245 0.0359 -6.26 <0.0001
sroh=VG   -7.1991 2.4392 -2.95 0.0034
sroh=G    -10.3201 2.4668 -4.18 <0.0001
sroh=F    -12.7668 3.1761 -4.02 <0.0001
sur_s * typeA=Yes -0.3233 0.0815 -3.97 <0.0001
```

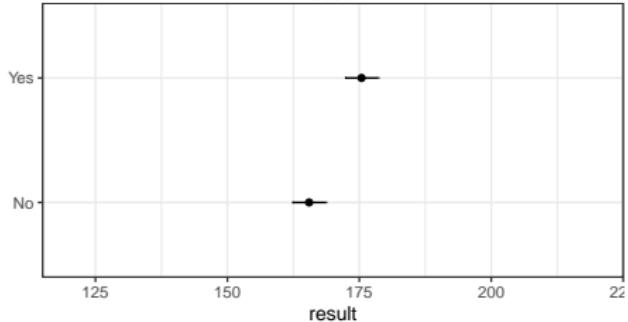
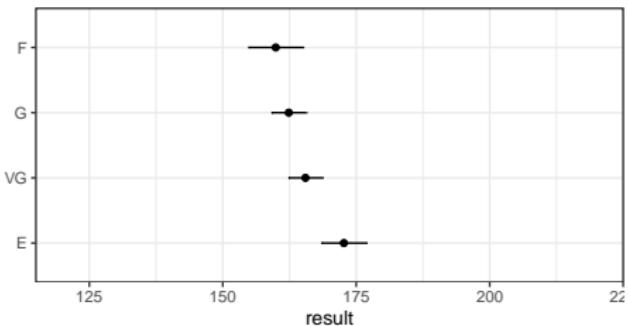
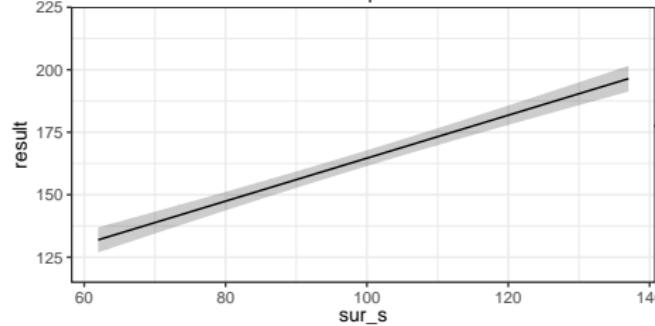
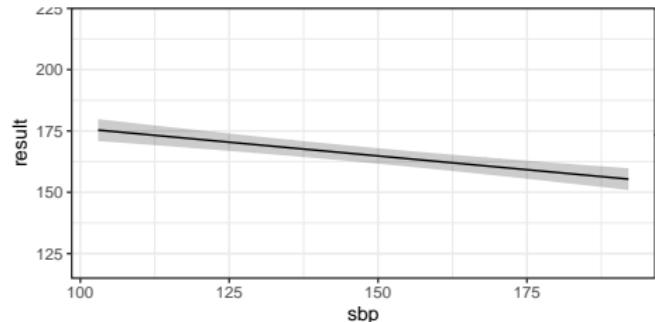
ANOVA for modI1

```
anova(modI1)
```

		Analysis of Variance			Response: result		
Factor		d.f.	Partial SS	MS	F	P	
sur_s	(Factor+Higher Order Factors)	2	91209.748	45604.8740	166.34	<.0001	
All Interactions		1	4315.423	4315.4231	15.74	1e-04	
typeA	(Factor+Higher Order Factors)	2	14549.125	7274.5626	26.53	<.0001	
All Interactions		1	4315.423	4315.4231	15.74	1e-04	
sbp		1	10749.174	10749.1735	39.21	<.0001	
sroh		3	6136.426	2045.4754	7.46	1e-04	
sur_s * typeA	(Factor+Higher Order Factors)	1	4315.423	4315.4231	15.74	1e-04	
REGRESSION		7	127310.325	18187.1893	66.34	<.0001	
ERROR		392	107474.035	274.1685			

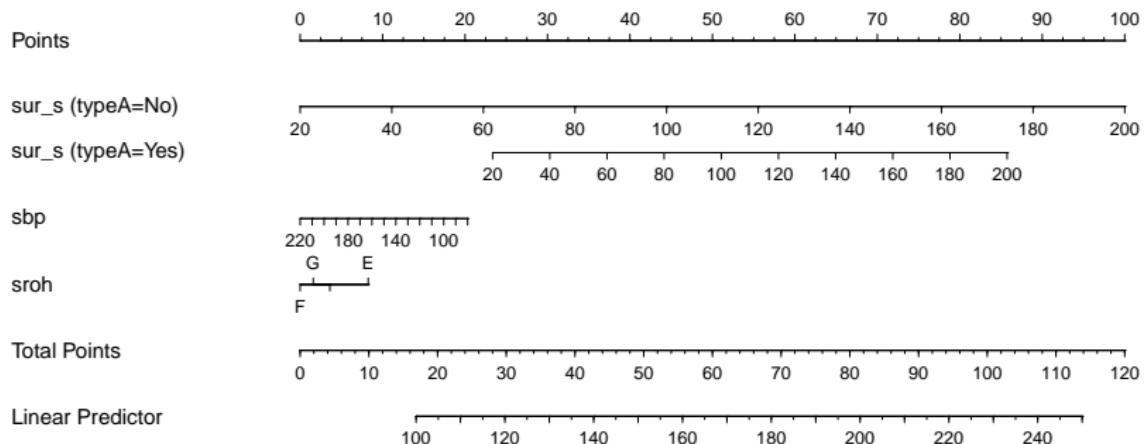
What does modI1 look like?

```
ggplot(Predict(modI1))
```



Nomogram for modI1

```
plot(nomogram(modI1))
```



Model modI2 adds how many df to modC4?

modI2

```
> modI2
Linear Regression Model

ols(formula = result ~ rcs(sur_s, 4) + typeA + sur_s %ia% typeA +
    sbp + sroh, data = datI2)

      Model Likelihood Discrimination
      Ratio Test      Indexes
Obs     400   LR chi2   634.39      R2      0.795
sigmAll.1022 d.f.          9      R2 adj  0.791
d.f.     390  Pr(> chi2) 0.0000      g      20.660

Residuals

      Min        1Q      Median        3Q       Max
-32.937 -4.914 -0.253  5.024 138.939

      Coef      S.E.      t      Pr(>|t|)
Intercept  33.6458  8.6580   3.89  0.0001
sur_s      1.9693  0.0973  20.23 <0.0001
sur_s'
sur_s''    -4.7400  0.2660 -17.82 <0.0001
sur_s'''   22.9316  1.1316  20.26 <0.0001
typeA=Yes  32.4408  5.6801   5.71 <0.0001
sur_s * typeA=Yes -0.2278  0.0554  -4.11 <0.0001
sbp        -0.1728  0.0242  -7.15 <0.0001
sroh=VG    -1.8306  1.6546  -1.11  0.2693
sroh=G     -4.5556  1.6773  -2.72  0.0069
sroh=F     -10.8748 2.1376  -5.09 <0.0001
```

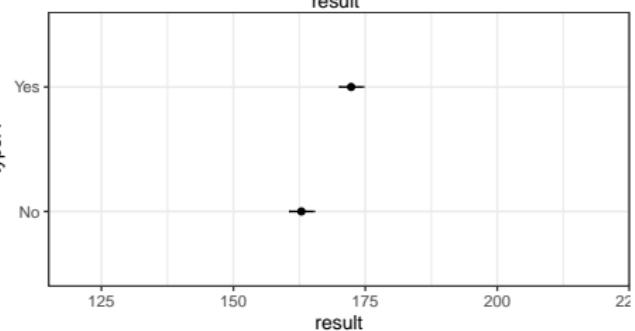
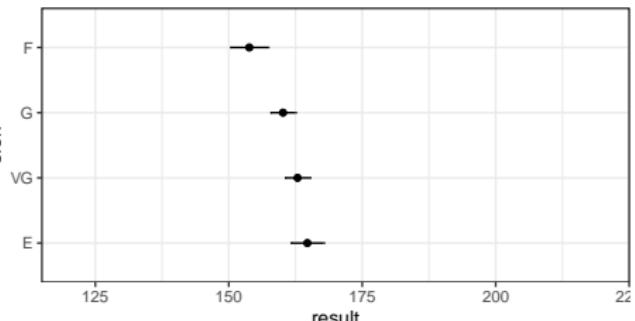
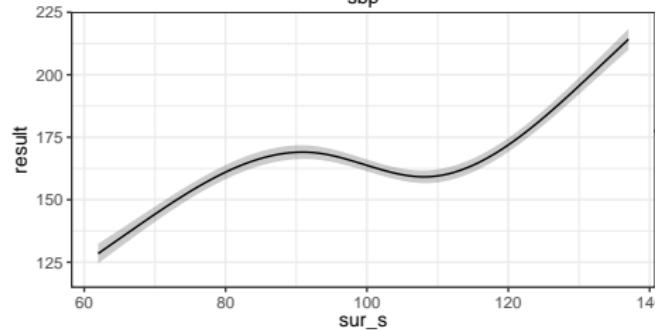
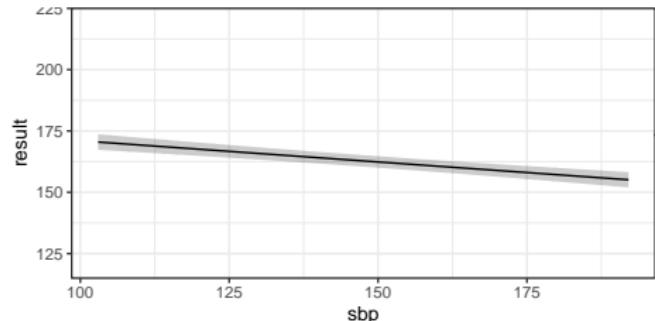
ANOVA for modI2

```
anova(modI2)
```

		Analysis of Variance				Response: result	
Factor		d.f.	Partial SS	MS	F	P	
sur_s	(Factor+Higher Order Factors)	4	150612.707	37653.1768	305.48	<.0001	
All Interactions		1	2083.922	2083.9220	16.91	<.0001	
Nonlinear		2	59402.959	29701.4795	240.97	<.0001	
typeA	(Factor+Higher Order Factors)	2	11023.724	5511.8620	44.72	<.0001	
All Interactions		1	2083.922	2083.9220	16.91	<.0001	
sur_s * typeA	(Factor+Higher Order Factors)	1	2083.922	2083.9220	16.91	<.0001	
sbp		1	6308.212	6308.2124	51.18	<.0001	
sroh		3	3849.970	1283.3234	10.41	<.0001	
TOTAL NONLINEAR + INTERACTION		3	63718.382	21239.4607	172.32	<.0001	
REGRESSION		9	186713.284	20745.9205	168.31	<.0001	
ERROR		390	48071.076	123.2592			

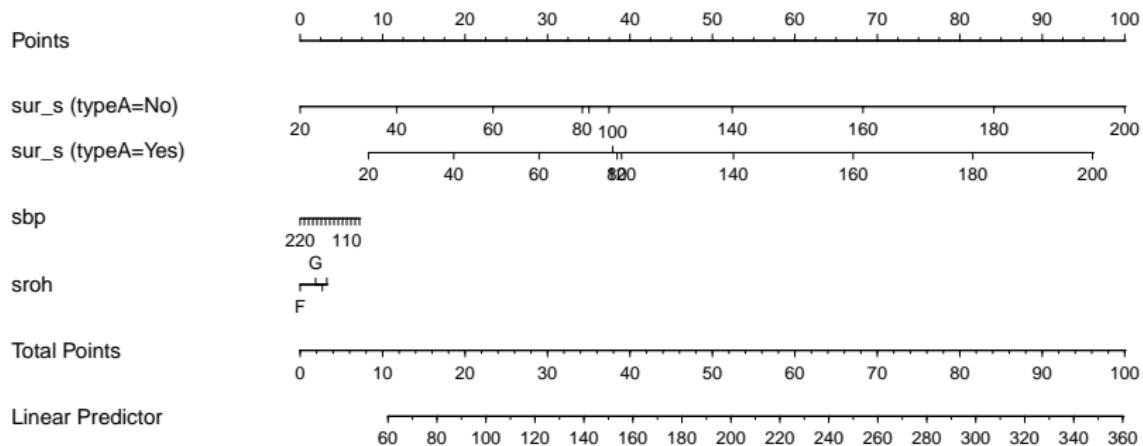
What does modI2 look like?

```
ggplot(Predict(modI2))
```



Nomogram for modI2

```
plot(nomogram(modI2))
```



Comparing Models?

```
set.seed(4321); validate(modA)
```

	index.orig	training	test	optimism
R-square	0.5239	0.5485	0.5035	0.0450
MSE	279.4736	309.0788	291.4385	17.6403
g	19.6922	20.4852	19.5334	0.9518
Intercept	0.0000	0.0000	5.5331	-5.5331
Slope	1.0000	1.0000	0.9670	0.0330
	index.corrected	n		
R-square	0.4788	40		
MSE	261.8333	40		
g	18.7404	40		
Intercept	5.5331	40		
Slope	0.9670	40		

- Ran validate for other models (see next slide)

Table of validate Results

Model	Raw R^2	Corrected R^2	Corrected MSE
modA (Main Effects)	0.5239	0.4788	261.8
modP2 (Quadr. Pol.)	0.5863	0.4756	293.8
modP3 (Cubic Pol.)	0.9535	0.9684	28.5
modC3 (RCS, 3 knots)	0.5399	0.4510	313.0
modC4 (RCS, 4 knots)	0.7864	0.7294	162.8
modC5 (RCS, 5 knots)	0.8106	0.7580	137.6
modI1 (interaction)	0.5422	0.4413	339.2
modI2 (int + RCS4)	0.7953	0.7337	161.4

Making Predictions

Suppose we want to predict the result for these new subjects:

```
new_people <- tibble(  
  name = c("Dave", "Edna"),  
  sur_s = c(100, 115), typeA = c("Yes", "No"),  
  sbp = c(140, 125), sroh = c("G", "E"))  
  
new_people %>% kable()
```

name	sur_s	typeA	sbp	sroh
Dave	100	Yes	140	G
Edna	115	No	125	E

Predicting Dave and Edna with modA

- Individual Prediction Intervals

```
predict(modA, newdata = data.frame(new_people),  
       conf.int = 0.95, conf.type = "individual")
```

\$linear.predictors

1	2
---	---

172.7522	187.9628
----------	----------

\$lower

1	2
---	---

139.4297	154.4792
----------	----------

\$upper

1	2
---	---

206.0747	221.4463
----------	----------

Predicting mean of people just like Dave and Edna with modA

- Mean Prediction Intervals

```
predict(modA, newdata = data.frame(new_people),  
        conf.int = 0.95, conf.type = "mean")
```

\$linear.predictors

	1	2
--	---	---

172.7522	187.9628
----------	----------

\$lower

	1	2
--	---	---

169.4477	183.3068
----------	----------

\$upper

	1	2
--	---	---

176.0567	192.6188
----------	----------

Predicting Dave and Edna with other models

```
predict(modP3, newdata = data.frame(new_people))
```

	1	2
173.8945	173.7410	

```
predict(modC4, newdata = data.frame(new_people))
```

	1	2
171.7091	167.9763	

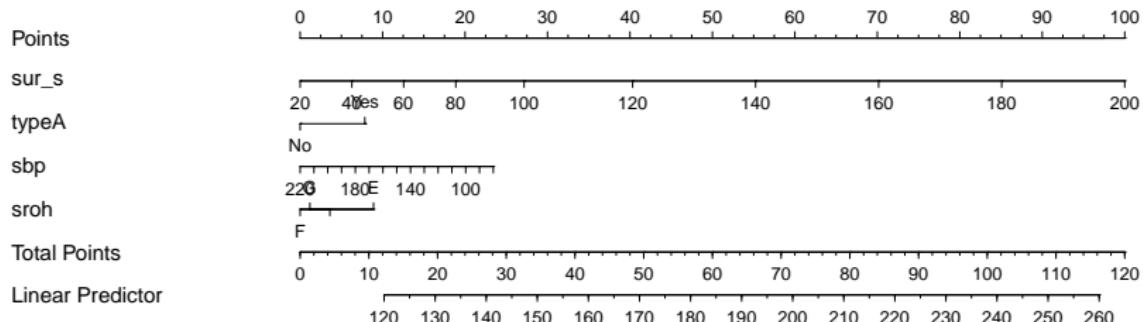
```
predict(modI2, newdata = data.frame(new_people))
```

	1	2
171.8875	169.4290	

Predicting Dave via the Nomogram for model modC3

- Dave has $\text{sur_s} = 100$, is typeA, Good sroh, $\text{sbp} = 140$.

```
plot(nomogram(modC3))
```



Dave's Actual Predicted Value (from modC3)

```
predict(modC3, newdata = data.frame(new_people))[1]
```

1

170.2422

Running the lm version of modC5

```
modC5_lm <- lm(result ~ rcs(sur_s,5) + typeA + sbp + sroh,  
                  data = dat12)  
  
anova(modC5_lm)
```

Analysis of Variance Table

Response: result

	Df	Sum Sq	Mean Sq	F value	Pr(>F)	
rcs(sur_s, 5)	4	171275	42819	375.551	< 2.2e-16	***
typeA	1	8141	8141	71.402	5.856e-16	***
sbp	1	7124	7124	62.479	2.789e-14	***
sroh	3	3779	1260	11.048	5.627e-07	***
Residuals	390	44466	114			

Signif. codes:

0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

The modC5_lm model equation

```
extract_eq(modC5_lm, use_coefs = TRUE, wrap = TRUE,  
           terms_per_line = 2)
```

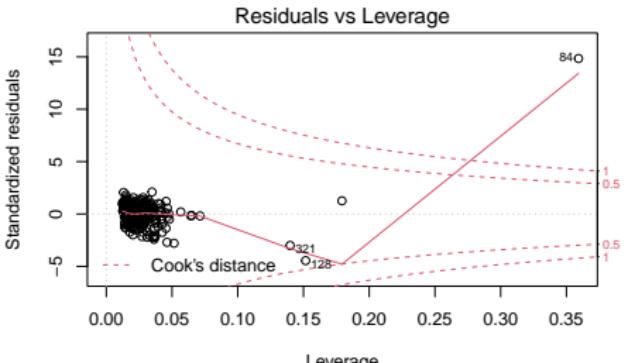
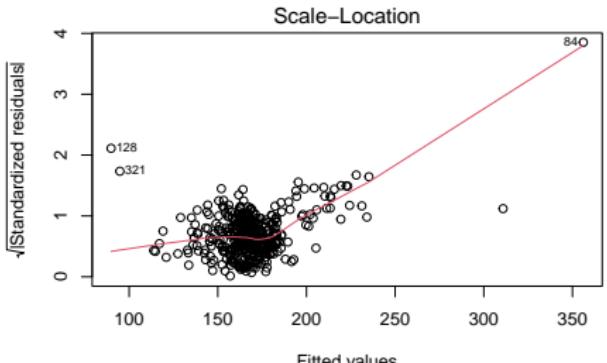
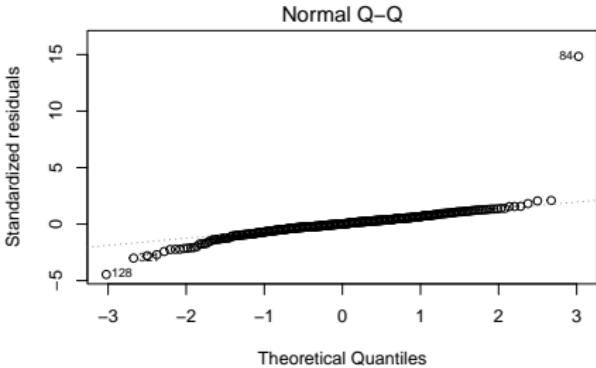
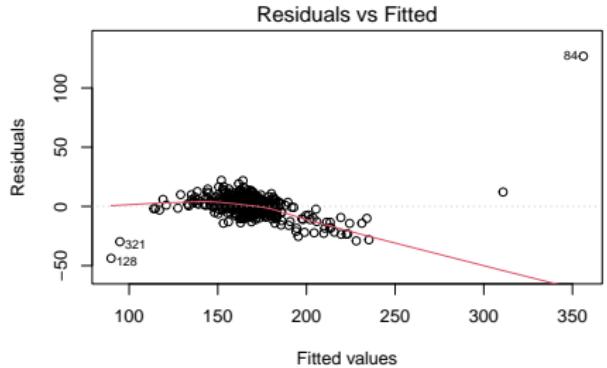
$$\widehat{\text{result}} = 57.33 + 1.67(\text{rcs}(\text{sur_s}, 5)_{\text{sur_s}}) - 3.25(\text{rcs}(\text{sur_s}, 5)_{\text{sur_s}'}) + 1.62(\text{rcs}(\text{sur_s}, 5)_{\text{sur_s}''}) + 27.1(\text{rcs}(\text{sur_s}, 5)_{\text{sur_s}'''}) + 9.75(\text{typeA}_{Y_{\text{es}}}) - 0.18(\text{sbp}) - 2.23(\text{sroh}_{VG}) - 4.62(\text{sroh}_G) - 11.02(\text{sroh}_F) \quad (1)$$

Residual Plots for modC5

```
par(mfrow = c(2,2)); plot(modC5_lm); par(mfrow = c(1,1))
```

- Results shown on next slide (not for the faint of heart)

Residual Plots for modC5



Oh dear...

```
dat12 %>% slice(84) %>% kable()
```

subj	result	sur_s	typeA	sbp	sroh
84	483	185	No	148	E

```
summary(dat12 %>% select(result, sur_s, sbp, sroh, typeA))
```

result	sur_s	sbp	sroh
Min. : 46.0	Min. : 39.0	Min. : 85.0	E : 68
1st Qu.:160.0	1st Qu.: 87.0	1st Qu.:132.0	VG:147
Median :168.0	Median :101.0	Median :147.0	G :139
Mean :168.8	Mean :100.2	Mean :148.1	F : 46
3rd Qu.:177.0	3rd Qu.:114.0	3rd Qu.:165.0	
Max. :483.0	Max. :185.0	Max. :215.0	

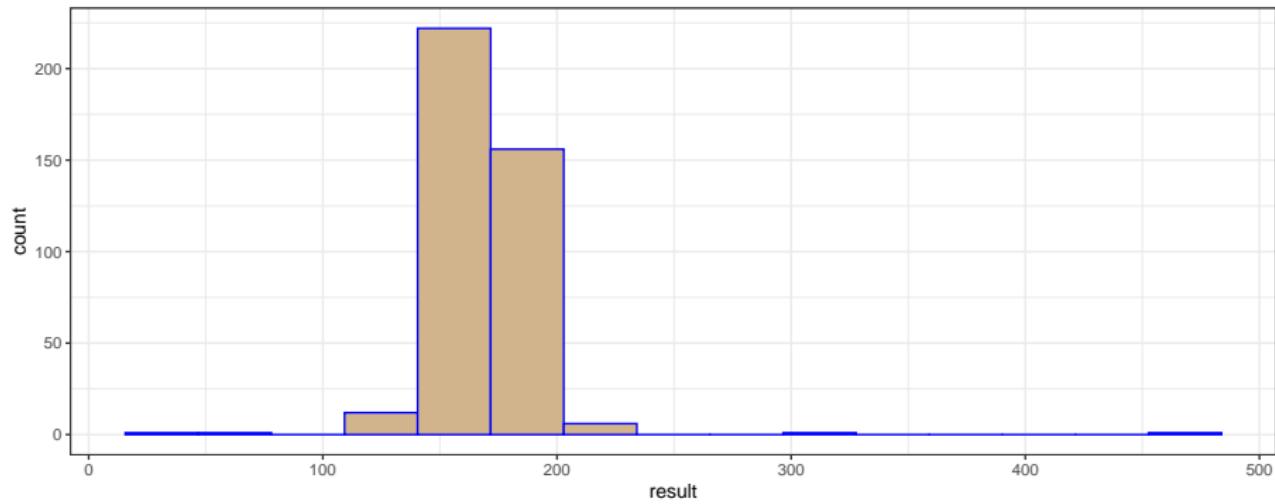
typeA

No :203

Veg:197

Was this foreseeable?

```
ggplot(data = dat12, aes(x = result)) +  
  geom_histogram(bins = 15, col = "blue", fill = "tan")
```



Logistic Regression

Framingham Data (from Class 10)

```
fram_raw <- read_csv(here("data/framingham.csv")) %>%  
  type.convert(as.is = FALSE) %>%  
  clean_names()
```

The variables describe $n = 4238$ adults examined at baseline, then followed for 10 years to see if they developed incident coronary heart disease. The binary outcome (below) has no missing values.

```
fram_raw %>% tabyl(ten_year_chd)
```

ten_year_chd	n	percent
0	3594	0.8480415
1	644	0.1519585

Data Cleanup

```
fram_new <- fram_raw %>%
  rename(cigs = "cigs_per_day",
         stroke = "prevalent_stroke",
         hrate = "heart_rate",
         sbp = "sys_bp",
         chd10_n = "ten_year_chd") %>%
  mutate(educ = fct_recode(factor(education),
                          "Some HS" = "1",
                          "HS grad" = "2",
                          "Some Coll" = "3",
                          "Coll grad" = "4")) %>%
  mutate(chd10_f = fct_recode(factor(chd10_n),
                           "chd" = "1", "chd_no" = "0")) %>%
  select(subj_id, chd10_n, chd10_f, age,
         cigs, educ, hrate, sbp, stroke)
```

Data Descriptions

Today, we'll only use the chd variables, plus age.

Variable	Description
subj_id	identifying code added by Dr. Love
chd10_n	(numeric) 1 = coronary heart disease in next 10 years
chd10_f	(factor) "chd" or "chd_no" in next ten years
age	in years (range is 32 to 70)
cigs	number of cigarettes smoked per day
educ	4-level factor: educational attainment
hrate	heart rate in beats per minute
sbp	systolic blood pressure in mm Hg
stroke	1 = history of stroke, else 0

Missing Data?

```
miss_var_summary(fram_new)
```

```
# A tibble: 9 x 3
  variable n_miss pct_miss
  <chr>     <int>    <dbl>
1 educ        105    2.48
2 cigs         29    0.684
3 hrate        1    0.0236
4 subj_id       0      0
5 chd10_n       0      0
6 chd10_f       0      0
7 age          0      0
8 sbp          0      0
9 stroke        0      0
```

Prepare our outcome.

We have our binary outcome as both a factor variable and a numeric (0/1) variable

```
fram_new %$% str(chd10_f)
```

```
Factor w/ 2 levels "chd_no","chd": 1 1 1 2 1 1 2 1 1 1 ...
```

```
fram_new %$% str(chd10_n)
```

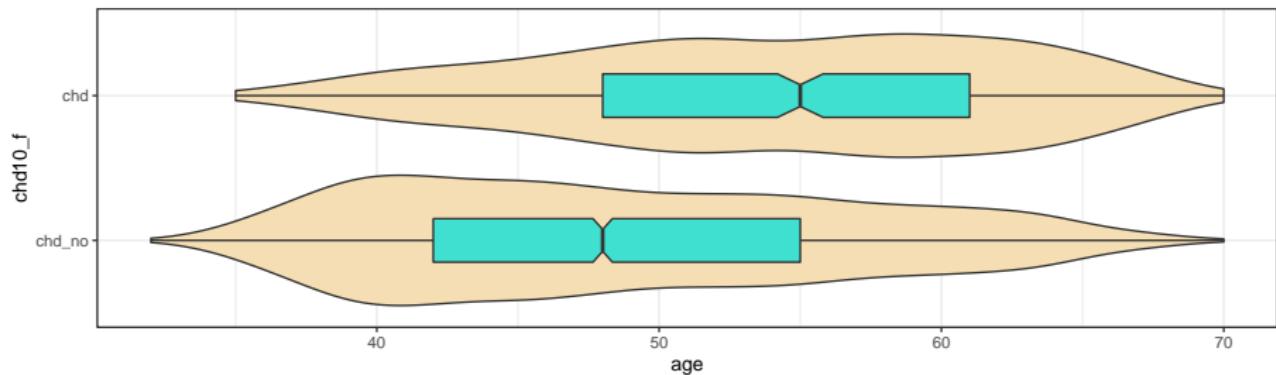
```
int [1:4238] 0 0 0 1 0 0 1 0 0 0 ...
```

```
fram_new %>% tabyl(chd10_f, chd10_n)
```

chd10_f	0	1
chd_no	3594	0
chd	0	644

Working with Binary Outcome Models

Does $\text{Pr}(\text{CHD in next ten years})$ look higher for *older* or *younger* people?



chd10_f	n	mean(age)	sd(age)	median(age)
chd_no	3594	48.77	8.41	48
chd	644	54.15	8.01	55

So what do we expect in this model?

$\text{Pr}(\text{CHD in next ten years})$ looks higher for *older* people?

If we predict $\log(\text{odds}(\text{CHD in next ten years}))$, we want to ensure that value will be **rising** with increased age.

So, for the `mage_1` model below, what sign do we expect for the slope of age?

```
mage_1 <- glm(chd10_f ~ age, family = binomial,  
                 data = fram_new)
```

Results for mage_1

```
tidy(mage_1) %>% kable(digits = 3)
```

term	estimate	std.error	statistic	p.value
(Intercept)	-5.558	0.284	-19.585	0
age	0.075	0.005	14.166	0

```
tidy(mage_1, exponentiate = TRUE) %>% kable(digits = 3)
```

term	estimate	std.error	statistic	p.value
(Intercept)	0.004	0.284	-19.585	0
age	1.077	0.005	14.166	0

Six ways to specify the outcome for this model

```
x1 <- glm(chd10_f ~ age,  
          family = binomial, data = fram_new)  
x2 <- glm(chd10_n ~ age,  
          family = binomial, data = fram_new)  
x3 <- glm((chd10_n == "1") ~ age,  
          family = binomial, data = fram_new)  
x4 <- glm((chd10_n == "0") ~ age,  
          family = binomial, data = fram_new)  
x5 <- glm((chd10_f == "chd") ~ age,  
          family = binomial, data = fram_new)  
x6 <- glm((chd10_f == "chd_no") ~ age,  
          family = binomial, data = fram_new)
```

What will happen to the age coefficient in these models?

Age Models x1 and x2

```
x1 <- glm(chd10_f ~ age,  
           family = binomial, data = fram_new)  
extract_eq(x1, use_coefs = TRUE)
```

$$\log \left[\frac{\widehat{P(\text{chd10_f} = \text{chd})}}{1 - \widehat{P(\text{chd10_f} = \text{chd})}} \right] = -5.56 + 0.07(\text{age}) \quad (2)$$

```
x2 <- glm(chd10_n ~ age,  
           family = binomial, data = fram_new)  
extract_eq(x2, use_coefs = TRUE)
```

$$\log \left[\frac{\widehat{P(\text{chd10_n} = 1)}}{1 - \widehat{P(\text{chd10_n} = 1)}} \right] = -5.56 + 0.07(\text{age}) \quad (3)$$

Age Models x3 and x4

```
x3 <- glm((chd10_n == "1") ~ age,  
           family = binomial, data = fram_new)  
extract_eq(x3, use_coefs = TRUE)
```

$$\log \left[\frac{\widehat{P(\text{chd10_n} = 1)}}{1 - \widehat{P(\text{chd10_n} = 1)}} \right] = -5.56 + 0.07(\text{age}) \quad (4)$$

```
x4 <- glm((chd10_n == "0") ~ age,  
           family = binomial, data = fram_new)  
extract_eq(x4, use_coefs = TRUE)
```

$$\log \left[\frac{\widehat{P(\text{chd10_n} = 0)}}{1 - \widehat{P(\text{chd10_n} = 0)}} \right] = 5.56 - 0.07(\text{age}) \quad (5)$$

Age Models x5 and x6

```
x5 <- glm((chd10_f == "chd") ~ age,  
           family = binomial, data = fram_new)  
extract_eq(x5, use_coefs = TRUE)
```

$$\log \left[\frac{P(\widehat{\text{chd10_f}} = \text{chd})}{1 - P(\widehat{\text{chd10_f}} = \text{chd})} \right] = -5.56 + 0.07(\text{age}) \quad (6)$$

```
x6 <- glm((chd10_f == "chd_no") ~ age,  
           family = binomial, data = fram_new)  
extract_eq(x6, use_coefs = TRUE)
```

$$\log \left[\frac{P(\widehat{\text{chd10_f}} = \text{chd_no})}{1 - P(\widehat{\text{chd10_f}} = \text{chd_no})} \right] = 5.56 - 0.07(\text{age}) \quad (7)$$

Making Predictions with a `glm` model

```
modelL1 <- glm(chd10_f == "chd" ~ age,  
                 family = binomial, data = fram_new)  
  
new_folks <- tibble(name = c("Frank", "Grace"),  
                      age = c(42, 56))  
  
new_folks %>% kable()
```

name	age
Frank	42
Grace	56

Predictions from a `glm` model (`modelL1`)

predictions on the logit scale

```
predict(modelL1, newdata = data.frame(new_folks))
```

1	2
-2.424935	-1.380563

or on the probability scale (reminder: `glm fit`)

```
predict(modelL1, newdata = data.frame(new_folks),  
        type = "response")
```

1	2
0.0812909	0.2009186

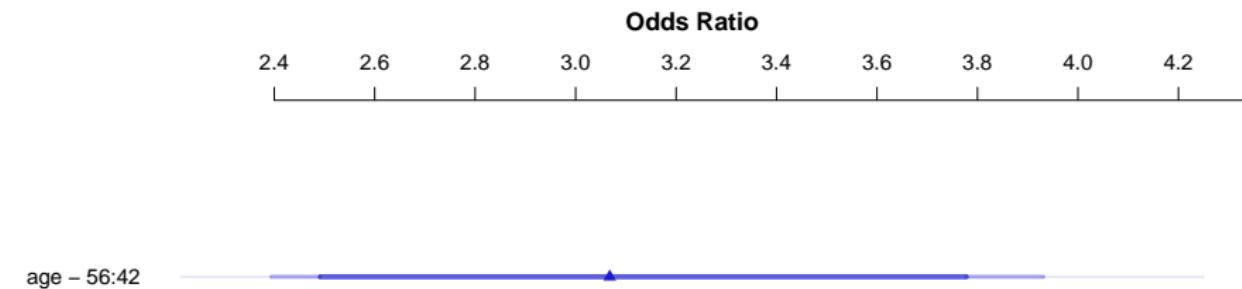
Building a different model with lrm

```
dd <- datadist(fram_new)
options(datadist = "dd")

modelL2 <- lrm(chd10_f == "chd" ~ rcs(age, 4),
                 data = fram_new, x = TRUE, y = TRUE)
```

Plot Effect Sizes from modelL2

```
plot(summary(modelL2))
```



Making Predictions with lrm (modelL2)

```
new_folks %>% kable()
```

name	age
Frank	42
Grace	56

- Predictions on the logit scale

```
predict(modelL2, newdata = data.frame(new_folks))
```

1	2
-2.465157	-1.344158

Useful Predictions with `lrm` (`modelL2`)

```
new_folks %>% kable()
```

name	age
Frank	42
Grace	56

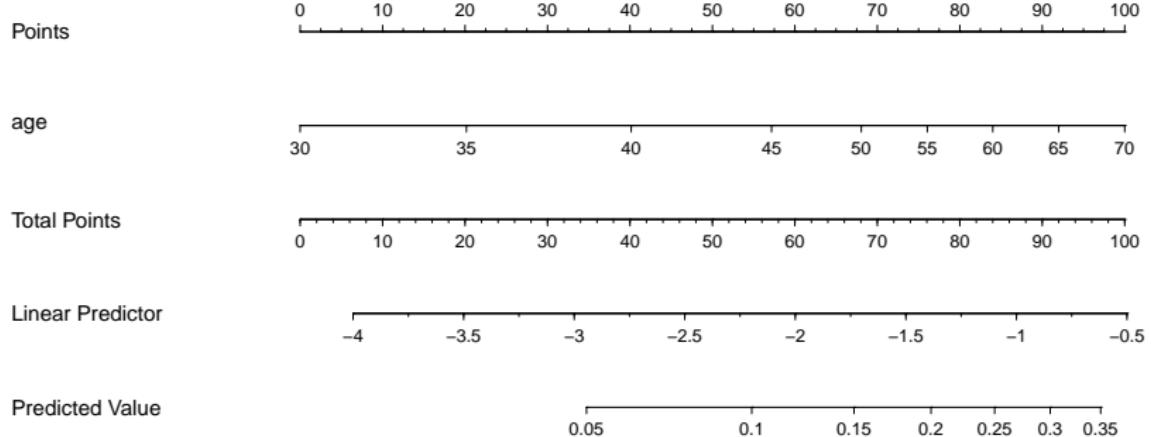
- Predicted probabilities after an `lrm` fit...

```
predict(modelL2, newdata = data.frame(new_folks),  
        type = "fitted")
```

1	2
0.07833722	0.20682714

Using the Nomogram to predict for Age 50

```
plot(nomogram(modell2, fun = plogis))
```



Compare our results from the nomogram...

- Predicted probabilities after an `lrm` fit...

```
predict(modelL2, newdata = data.frame(age = 50),  
        type = "fitted")
```

1
0.1542483

Validate C statistic, Nagelkerke R^2 , Brier score

```
set.seed(2022)
validate(modell2, B = 50)
```

```
> set.seed(2022)
> validate(modell2, B = 50)
      index.orig training   test optimism index.corrected n
Dxy        0.3581    0.3548  0.3581  -0.0033        0.3615 50
R2         0.0891    0.0887  0.0883   0.0004        0.0887 50
Intercept  0.0000    0.0000  0.0077  -0.0077        0.0077 50
Slope       1.0000    1.0000  1.0057  -0.0057        1.0057 50
Emax       0.0000    0.0000  0.0026   0.0026        0.0026 50
D          0.0522    0.0520  0.0517   0.0003        0.0519 50
U          -0.0005   -0.0005  0.0000  -0.0005        0.0000 50
Q          0.0527    0.0525  0.0517   0.0008        0.0519 50
B          0.1223    0.1225  0.1224   0.0001        0.1222 50
g          0.8169    0.8116  0.8124  -0.0008        0.8176 50
gp         0.0925    0.0918  0.0917   0.0001        0.0924 50
```

Next Time

- Logistic Regression using `tidymodels`
- Quiz 1 will be made available today at 5 PM. Good luck!