432 Class 03

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

2025-01-21

## Today’s Agenda

* Fitting linear regression models with lm
  + ANOVA: Incorporating an interaction between factors
  + ANCOVA: Incorporating a quantitative covariate
* Regression Diagnostics via check\_model()
* Validating / evaluating results in a test sample

### Appendix (see [Section 7](#sec-appendix))

How the smt\_im data were created from smart\_ohio.csv

## Today’s R Setup

knitr::opts\_chunk$set(comment = NA)  
  
library(janitor)  
library(naniar)  
library(broom)  
library(car)  
library(gt)  
library(mosaic) ## for df\_stats and favstats  
library(mice) ## imputation of missing data  
library(patchwork)   
library(rsample) ## data splitting  
library(easystats)  
library(tidyverse)   
  
theme\_set(theme\_lucid())

# The smt\_im data

## The smt\_im data

* 894 subjects in Cleveland-Elyria with bmi and no history of diabetes (missing values singly imputed: assume MAR)
* All subjects have hx\_diabetes (all 0), and are located in the MMSA labeled Cleveland-Elyria.
* See [Course Notes Chapter on BRFSS SMART data](https://thomaselove.github.io/432-notes/06-smart.html) for variable details
* Appendix provides details on data development.

## The Five Variables We’ll Use Today

9 variables in the data but we’ll use only these 5 today.

| Variable | Description |
| --- | --- |
| ID | subject identifying code |
| bmi | (outcome) Body-Mass index in . |
| exerany | any exercise in past month: 1 = yes, 0 = no |
| health | self-reported overall health (5 levels) |
| fruit\_day | average fruit servings consumed per day |

## Data Load

smt\_im <- read\_rds("c03/data/smt\_im.Rds") |>  
 select(ID, bmi, exerany, health, fruit\_day, everything())  
  
smt\_im

# A tibble: 894 × 9  
 ID bmi exerany health fruit\_day inc\_imp drinks\_wk female race\_eth   
 <chr> <dbl> <dbl> <fct> <dbl> <dbl> <dbl> <dbl> <fct>   
 1 2 23.0 0 E 4 86865 0 1 White non-Hisp…  
 2 3 26.9 1 G 3 20904 0 1 Other race non…  
 3 4 26.5 1 G 2 46946 4.67 1 White non-Hisp…  
 4 5 24.2 1 G 0.57 58311 0.93 0 White non-Hisp…  
 5 7 23.0 1 G 2 2318 2 0 White non-Hisp…  
 6 8 28.4 1 VG 1 79667 0 0 Other race non…  
 7 9 30.1 1 F 0.23 47880 0 0 Black non-Hisp…  
 8 10 19.8 1 E 0.77 100136 0.47 1 White non-Hisp…  
 9 11 27.2 1 E 0.71 73145 0 0 White non-Hisp…  
10 12 24.6 1 E 1.07 76917 0 1 Other race non…  
# ℹ 884 more rows

## Checking our Data

Are there any missing values?

smt\_im |> n\_miss()

[1] 0

Does each row have a unique ID value?

identical(nrow(smt\_im), n\_distinct(smt\_im$ID))

[1] TRUE

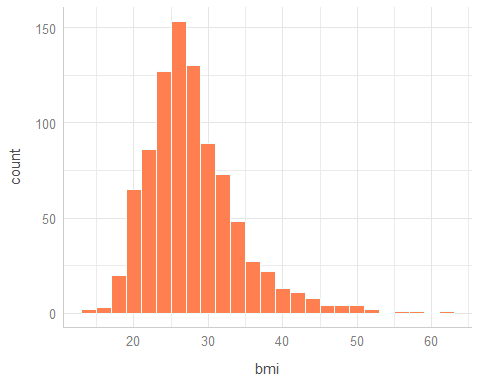
## Range and Level Checks?

data\_codebook(smt\_im |> select(bmi, exerany, health, fruit\_day))

select(smt\_im, bmi, exerany, health, fruit\_day) (894 rows and 4 variables, 4 shown)  
  
ID | Name | Type | Missings | Values | N  
---+-----------+-------------+----------+------------+------------  
1 | bmi | numeric | 0 (0.0%) | [13.3, 63] | 894  
---+-----------+-------------+----------+------------+------------  
2 | exerany | numeric | 0 (0.0%) | 0 | 232 (26.0%)  
 | | | | 1 | 662 (74.0%)  
---+-----------+-------------+----------+------------+------------  
3 | health | categorical | 0 (0.0%) | E | 148 (16.6%)  
 | | | | VG | 324 (36.2%)  
 | | | | G | 274 (30.6%)  
 | | | | F | 113 (12.6%)  
 | | | | P | 35 ( 3.9%)  
---+-----------+-------------+----------+------------+------------  
4 | fruit\_day | numeric | 0 (0.0%) | [0, 10] | 894  
------------------------------------------------------------------

## Our outcome, bmi

ggplot(smt\_im, aes(x = bmi)) +  
 geom\_histogram(binwidth = 2, col = "azure", fill = "coral")



## Key predictors: exerany, health

smt\_im |> tabyl(exerany, health) |>   
 adorn\_totals(where = c("row", "col")) |>  
 gt() |> tab\_options(table.font.size = 28)

| exerany | E | VG | G | F | P | Total |
| --- | --- | --- | --- | --- | --- | --- |
| 0 | 22 | 70 | 77 | 48 | 15 | 232 |
| 1 | 126 | 254 | 197 | 65 | 20 | 662 |
| Total | 148 | 324 | 274 | 113 | 35 | 894 |

Here, it doesn’t matter much whether we store the 1/0 in exerany as numeric or as a two-level factor in R. For binary variables, sometimes the numeric version will be more useful and sometimes a factor will be more useful.

## Our covariate, fruit\_day

We are mostly interested in whether accounting for the quantitative covariate fruit\_day changes the modeled association of our key predictors with bmi.

* Sometimes we center such a covariate (subtracting its mean.)

smt\_im <- smt\_im |>  
 mutate(fruit\_c = fruit\_day - mean(fruit\_day))

* Why? So that we can easily plug in the covariate’s mean (which will now be 0) when making predictions.

## Did we center fruit\_day properly?

Here’s a little “sanity check”:

df\_stats(~ fruit\_day + fruit\_c, data = smt\_im) |> gt() |>  
 fmt\_number(columns = min:sd, decimals = 3) |>  
 tab\_options(table.font.size = 24)

| response | min | Q1 | median | Q3 | max | mean | sd | n | missing |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| fruit\_day | 0.000 | 0.710 | 1.030 | 2.000 | 10.000 | 1.429 | 1.116 | 894 | 0 |
| fruit\_c | -1.429 | -0.719 | -0.399 | 0.571 | 8.571 | 0.000 | 1.116 | 894 | 0 |

The df\_stats() function comes from the mosaic package, and can be used to apply favstats() to multiple variables at once.

## Modeling Plan

1. Split smt\_im into training and testing samples.
2. Predict bmi using exer\_any and health
   * (fit1): without an interaction between the predictors
   * (fit2): and then with an interaction term
3. (fit3): Add in our (centered) covariate, fruit\_c to fit1
4. (fit4): Add in our (centered) covariate, fruit\_c to fit2
5. Assess all four models in training and testing samples.

## Splitting the Sample

We’ll partition our data set using some tools from the rsample package, into:

* a training sample containing 75% of the data
* a testing sample containing the remaining 25%

set.seed(432) ## for future replication  
  
smt\_im\_split <- initial\_split(smt\_im, prop = 3/4)  
  
train\_smt\_im <- training(smt\_im\_split)  
test\_smt\_im <- testing(smt\_im\_split)  
  
c(nrow(smt\_im), nrow(train\_smt\_im), nrow(test\_smt\_im))

[1] 894 670 224

# Building Our Four Models

## Modeling Plan

* Predict bmi using exer\_any and health
  + (fit1): without an interaction between the predictors
  + (fit2): and then with an interaction term
  + (fit3): Add in our (centered) covariate, fruit\_c to fit1
  + (fit4): Add in our (centered) covariate, fruit\_c to fit2

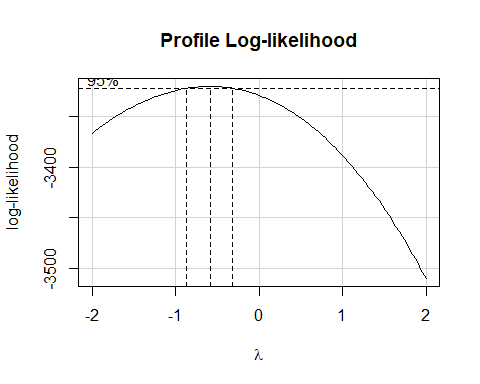
## Tukey’s ladder of power transformations

|  | 2 | 1 | 0.5 | 0 | -0.5 | -1 | -2 |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Transform |  |  |  |  |  |  |  |

* Used in combination with a Box-Cox plot from car
* Requires the variable to be **strictly positive**.
  + If desirable, you can add any constant to or multiply by any constant, before or after the transformation.
* Be sure you can back out of the transformation:
* , , ,

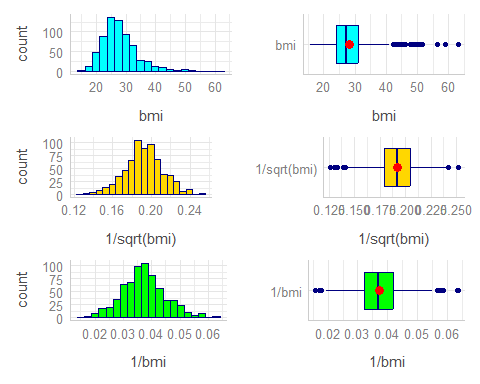
## Consider transforming bmi?

m0 <- lm(bmi ~ exerany + health + fruit\_c, data = train\_smt\_im)  
boxCox(m0)



## Should we transform bmi? (n = 670)

p1a <- ggplot(train\_smt\_im, aes(x = bmi)) +   
 geom\_histogram(col = "navy", fill = "cyan", bins = 20)  
  
p1b <- ggplot(train\_smt\_im, aes(x = bmi, y = "bmi")) +   
 geom\_boxplot(col = "navy", fill = "cyan") + labs(y = "") +  
 stat\_summary(geom = "point", col = "red", fun = mean, size = 3)  
  
p2a <- ggplot(train\_smt\_im, aes(x = 1/sqrt(bmi))) +   
 geom\_histogram(col = "navy", fill = "gold", bins = 20)  
  
p2b <- ggplot(train\_smt\_im, aes(x = 1/sqrt(bmi), y = "1/sqrt(bmi)")) +  
 geom\_boxplot(col = "navy", fill = "gold") + labs(y = "") +  
 stat\_summary(geom = "point", col = "red", fun = mean, size = 3)  
  
p3a <- ggplot(train\_smt\_im, aes(x = 1/bmi)) +   
 geom\_histogram(col = "navy", fill = "green", bins = 20)  
  
p3b <- ggplot(train\_smt\_im, aes(x = 1/bmi, y = "1/bmi")) +   
 geom\_boxplot(col = "navy", fill = "green") + labs(y = "") +  
 stat\_summary(geom = "point", col = "red", fun = mean, size = 3)  
  
(p1a + p1b) / (p2a + p2b) / (p3a + p3b)



## Re-scaling the transformation

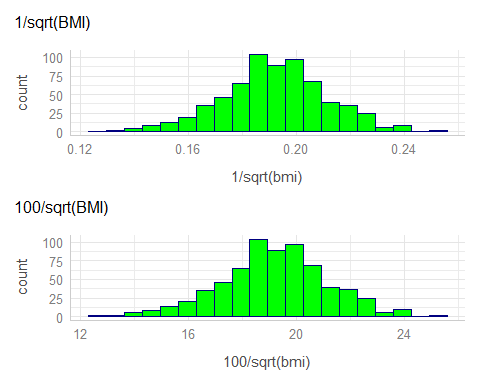
To ease interpretation of coefficients, I sometimes scale an outcome transformation so that its values fall in (10, 100), rather than between 0 and 1.

bind\_rows( favstats(~ 1/sqrt(bmi), data = train\_smt\_im),  
 favstats(~ 100/sqrt(bmi), data = train\_smt\_im)) |>  
 mutate(outcome = c("1/sqrt(bmi)", "100/sqrt(bmi)")) |>   
 relocate(outcome) |>  
 gt() |> fmt\_number(columns = min:sd, decimals = 3) |>   
 tab\_options(table.font.size = 24)

| outcome | min | Q1 | median | Q3 | max | mean | sd | n | missing |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1/sqrt(bmi) | 0.126 | 0.180 | 0.192 | 0.204 | 0.252 | 0.192 | 0.020 | 670 | 0 |
| 100/sqrt(bmi) | 12.599 | 17.958 | 19.194 | 20.447 | 25.230 | 19.195 | 1.987 | 670 | 0 |

## Shape doesn’t change

p2 <- ggplot(train\_smt\_im, aes(x = 1/sqrt(bmi))) +   
 geom\_histogram(col = "navy", fill = "green", bins = 20) +  
 labs(title = "1/sqrt(BMI)")  
  
p3 <- ggplot(train\_smt\_im, aes(x = 100/sqrt(bmi))) +  
 geom\_histogram(col = "navy", fill = "green", bins = 20) +   
 labs(title = "100/sqrt(BMI)")  
  
p2 / p3



## Means by exerany and health

summaries\_1 <- train\_smt\_im |>  
 group\_by(exerany, health) |>  
 summarise(n = n(), mean = mean(100/sqrt(bmi)),   
 stdev = sd(100/sqrt(bmi)))

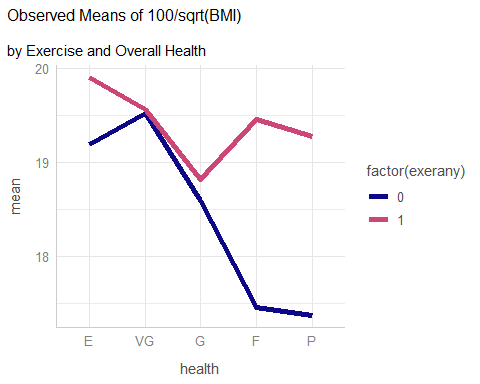
`summarise()` has grouped output by 'exerany'. You can override using the  
`.groups` argument.

summaries\_1

# A tibble: 10 × 5  
# Groups: exerany [2]  
 exerany health n mean stdev  
 <dbl> <fct> <int> <dbl> <dbl>  
 1 0 E 18 19.2 1.22  
 2 0 VG 55 19.5 1.90  
 3 0 G 60 18.6 2.22  
 4 0 F 32 17.5 2.43  
 5 0 P 9 17.4 2.25  
 6 1 E 92 19.9 1.64  
 7 1 VG 189 19.6 1.74  
 8 1 G 150 18.8 1.91  
 9 1 F 49 19.5 1.94  
10 1 P 16 19.3 2.79

## Code for Interaction Plot

ggplot(summaries\_1, aes(x = health, y = mean, col = factor(exerany))) +  
 geom\_line(aes(group = factor(exerany)), linewidth = 2) +  
 scale\_color\_viridis\_d(option = "C", end = 0.5) +  
 labs(title = "Observed Means of 100/sqrt(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 (see next slide) that results from this code?

# Fitting a Two-Way ANOVA model for

## Create our transformed outcome

We’ll want to do this in both our training and test samples.

train\_smt\_im <- train\_smt\_im |> mutate(bmi\_tr = 100 / sqrt(bmi))  
  
test\_smt\_im <- test\_smt\_im |> mutate(bmi\_tr = 100 / sqrt(bmi))

## Model fit1 without interaction

fit1 <- lm(bmi\_tr ~ exerany + health, data = train\_smt\_im)

Using the tidy() function from broom:

tidy(fit1, conf.int = TRUE, conf.level = 0.90) |>   
 gt() |> fmt\_number(columns = estimate:conf.high, decimals = 3) |>  
 tab\_options(table.font.size = 24)

| term | estimate | std.error | statistic | p.value | conf.low | conf.high |
| --- | --- | --- | --- | --- | --- | --- |
| (Intercept) | 19.306 | 0.233 | 82.845 | 0.000 | 18.922 | 19.689 |
| exerany | 0.572 | 0.172 | 3.329 | 0.001 | 0.289 | 0.854 |
| healthVG | -0.200 | 0.221 | -0.902 | 0.367 | -0.564 | 0.165 |
| healthG | -0.962 | 0.228 | -4.228 | 0.000 | -1.337 | -0.587 |
| healthF | -0.984 | 0.285 | -3.456 | 0.001 | -1.452 | -0.515 |
| healthP | -1.082 | 0.428 | -2.528 | 0.012 | -1.786 | -0.377 |

## Model Parameters for fit1

model\_parameters(fit1, ci = 0.90)

Parameter | Coefficient | SE | 90% CI | t(664) | p  
-------------------------------------------------------------------  
(Intercept) | 19.31 | 0.23 | [18.92, 19.69] | 82.84 | < .001  
exerany | 0.57 | 0.17 | [ 0.29, 0.85] | 3.33 | < .001  
health [VG] | -0.20 | 0.22 | [-0.56, 0.16] | -0.90 | 0.367   
health [G] | -0.96 | 0.23 | [-1.34, -0.59] | -4.23 | < .001  
health [F] | -0.98 | 0.28 | [-1.45, -0.51] | -3.46 | < .001  
health [P] | -1.08 | 0.43 | [-1.79, -0.38] | -2.53 | 0.012

Uncertainty intervals (equal-tailed) and p-values (two-tailed) computed  
 using a Wald t-distribution approximation.

## Model Parameters for fit1 (with gt())

Reformatting with gt()…

model\_parameters(fit1, ci = 0.90) |>   
 gt() |> fmt\_number(columns = -c(CI, df\_error), decimals = 3) |>  
 tab\_options(table.font.size = 24)

| Parameter | Coefficient | SE | CI | CI\_low | CI\_high | t | df\_error | p |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| (Intercept) | 19.306 | 0.233 | 0.9 | 18.922 | 19.689 | 82.845 | 664 | 0.000 |
| exerany | 0.572 | 0.172 | 0.9 | 0.289 | 0.854 | 3.329 | 664 | 0.001 |
| healthVG | -0.200 | 0.221 | 0.9 | -0.564 | 0.165 | -0.902 | 664 | 0.367 |
| healthG | -0.962 | 0.228 | 0.9 | -1.337 | -0.587 | -4.228 | 664 | 0.000 |
| healthF | -0.984 | 0.285 | 0.9 | -1.452 | -0.515 | -3.456 | 664 | 0.001 |
| healthP | -1.082 | 0.428 | 0.9 | -1.786 | -0.377 | -2.528 | 664 | 0.012 |

|  |
| --- |
| The fit1 equation |
| fit1: = 19.31 + .57 exerany - .20 (VG) - .96 (G) - .98 (F) - 1.08 (P) |

| Name | exerany | health | predicted |
| --- | --- | --- | --- |
| Harry | 0 | Excellent | 19.31 |
| Sally | 1 | Excellent | 19.31 + .57 = 19.88 |
| Billy | 0 | Fair | 19.31 - .98 = 18.33 |
| Meg | 1 | Fair | 19.31 + .57 - .98 = 18.90 |

* Effect of exerany on ?
* Effect of health = Fair instead of Excellent?

## How well does fit1 fit the training data?

n\_obs(fit1)

[1] 670

model\_performance(fit1)

# Indices of model performance  
  
AIC | AICc | BIC | R2 | R2 (adj.) | RMSE | Sigma  
------------------------------------------------------------------  
2786.766 | 2786.935 | 2818.317 | 0.069 | 0.062 | 1.916 | 1.925

glance(fit1) |>   
 select(r.squared, adj.r.squared, sigma, nobs,   
 df, df.residual, AIC, BIC) |>   
 gt() |> fmt\_number(columns = r.squared:sigma, decimals = 3) |>  
 fmt\_number(columns = AIC:BIC, decimals = 1) |>  
 tab\_options(table.font.size = 24)

| r.squared | adj.r.squared | sigma | nobs | df | df.residual | AIC | BIC |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0.069 | 0.062 | 1.925 | 670 | 5 | 664 | 2,786.8 | 2,818.3 |

## Tidied ANOVA for fit1

tidy(anova(fit1)) |> gt() |>   
 fmt\_number(columns = sumsq:statistic, decimals = 2) |>  
 fmt\_number(columns = p.value, decimals = 4) |>  
 tab\_options(table.font.size = 24)

| term | df | sumsq | meansq | statistic | p.value |
| --- | --- | --- | --- | --- | --- |
| exerany | 1 | 63.87 | 63.87 | 17.24 | 0.0000 |
| health | 4 | 118.71 | 29.68 | 8.01 | 0.0000 |
| Residuals | 664 | 2,459.85 | 3.70 | NA | NA |

## Model Checks

We’ll be checking assumptions related to:

* linearity
* homoscedasticity (constant variance)
* influential observations (outliers, leverage and influence)
* whether the residuals follow a Normal distribution
* collinearity (variance inflation factor)
* and a posterior predictive check of our predictions

## My slides and check\_model()

When building a regular HTML file, I would just use:

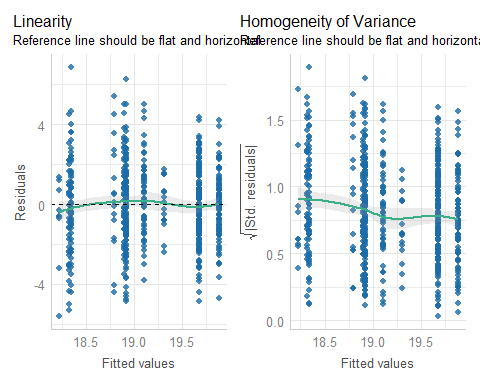
check\_model(fit1, detrend = FALSE)

with #| fig-height: 9 at the start of the code chunk so that the plots are taller than the default height (thus easier to read) but I will split out the plots for slides.

|  |
| --- |
| Problem with check\_model() |
| * The problem with check\_model() (particularly on Macs) now seems to be rectified. Update your packages (for instance, to performance version 0.13.0 or later) if you haven’t yet. |

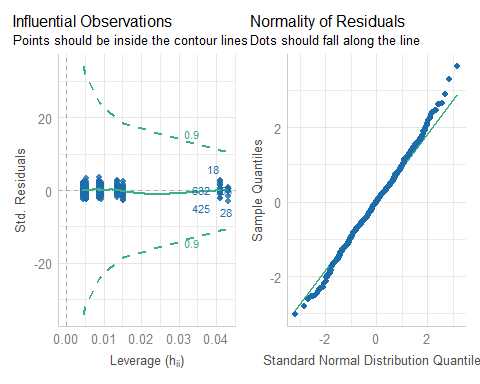
## Checking model fit1 (*n* = 670)

check\_model(fit1, check = c("linearity", "homogeneity"))



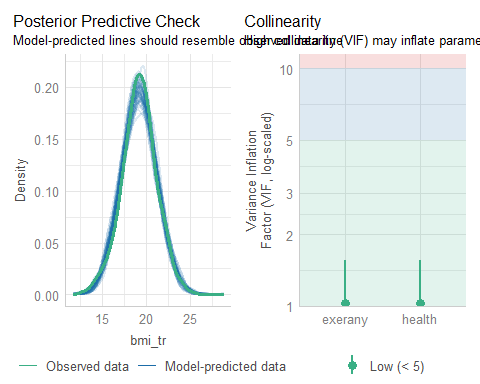
## Checking model fit1 (*n* = 670)

check\_model(fit1, check = c("outliers", "qq"), detrend = FALSE)



## Checking model fit1 (*n* = 670)

check\_model(fit1, check = c("pp\_check", "vif"))



# Fitting ANOVA model fit2 including interaction

## Adding the interaction term to fit1

fit2 <- lm(bmi\_tr ~ exerany \* health, data = train\_smt\_im)

* How do our models compare on fit to the training data?

bind\_rows(glance(fit1), glance(fit2)) |>  
 mutate(mod = c("fit1", "fit2")) |>  
 select(mod, r.sq = r.squared, adj.r.sq = adj.r.squared,   
 sigma, nobs, df, df.res = df.residual, AIC, BIC) |>   
 gt() |> fmt\_number(columns = r.sq:sigma, decimals = 3) |>  
 fmt\_number(columns = AIC:BIC, decimals = 1) |>  
 tab\_options(table.font.size = 24)

| mod | r.sq | adj.r.sq | sigma | nobs | df | df.res | AIC | BIC |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| fit1 | 0.069 | 0.062 | 1.925 | 670 | 5 | 664 | 2,786.8 | 2,818.3 |
| fit2 | 0.095 | 0.082 | 1.904 | 670 | 9 | 660 | 2,776.1 | 2,825.7 |

## ANOVA for the fit2 model

tidy(anova(fit2)) |> gt() |>   
 fmt\_number(columns = sumsq:statistic, decimals = 2) |>  
 fmt\_number(columns = p.value, decimals = 4) |>  
 tab\_options(table.font.size = 20)

| term | df | sumsq | meansq | statistic | p.value |
| --- | --- | --- | --- | --- | --- |
| exerany | 1 | 63.87 | 63.87 | 17.62 | 0.0000 |
| health | 4 | 118.71 | 29.68 | 8.19 | 0.0000 |
| exerany:health | 4 | 67.54 | 16.89 | 4.66 | 0.0010 |
| Residuals | 660 | 2,392.31 | 3.62 | NA | NA |

## ANOVA test comparing fit1 to fit2

anova(fit1, fit2)

Analysis of Variance Table  
  
Model 1: bmi\_tr ~ exerany + health  
Model 2: bmi\_tr ~ exerany \* health  
 Res.Df RSS Df Sum of Sq F Pr(>F)   
1 664 2459.8   
2 660 2392.3 4 67.544 4.6586 0.001029 \*\*  
---  
Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

## fit2 coefficients

tidy(fit2, conf.int = TRUE, conf.level = 0.90) |>  
 gt() |> fmt\_number(columns = estimate:conf.high, decimals = 3) |>  
 tab\_options(table.font.size = 20)

| term | estimate | std.error | statistic | p.value | conf.low | conf.high |
| --- | --- | --- | --- | --- | --- | --- |
| (Intercept) | 19.185 | 0.449 | 42.753 | 0.000 | 18.446 | 19.925 |
| exerany | 0.715 | 0.491 | 1.458 | 0.145 | -0.093 | 1.524 |
| healthVG | 0.333 | 0.517 | 0.643 | 0.520 | -0.519 | 1.184 |
| healthG | -0.593 | 0.512 | -1.159 | 0.247 | -1.436 | 0.250 |
| healthF | -1.729 | 0.561 | -3.083 | 0.002 | -2.653 | -0.805 |
| healthP | -1.818 | 0.777 | -2.339 | 0.020 | -3.098 | -0.537 |
| exerany:healthVG | -0.676 | 0.571 | -1.184 | 0.237 | -1.616 | 0.265 |
| exerany:healthG | -0.492 | 0.570 | -0.863 | 0.389 | -1.432 | 0.447 |
| exerany:healthF | 1.288 | 0.654 | 1.968 | 0.049 | 0.210 | 2.365 |
| exerany:healthP | 1.194 | 0.933 | 1.280 | 0.201 | -0.342 | 2.731 |

## Interpreting the fit2 model

| Name | exerany | health | predicted |
| --- | --- | --- | --- |
| Harry | 0 | Excellent | 19.19 |
| Sally | 1 | Excellent | 19.19 + .72 = 19.91 |
| Billy | 0 | Fair | 19.19 - 1.73 = 17.46 |
| Meg | 1 | Fair | 19.19 + .72 - 1.73 + 1.29 = 19.47 |

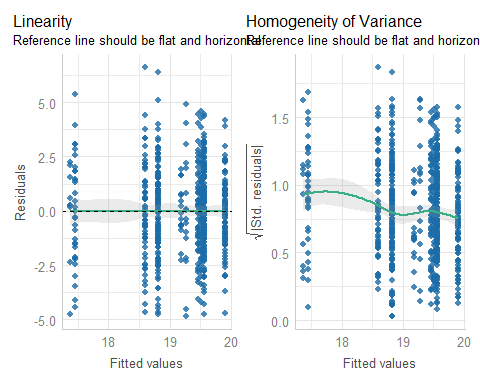
* How do we interpret effect sizes here? **It depends**…

## Interpreting the fit2 model

* Effect of exerany on predicted ?
  + If health = Excellent, effect is +0.72
  + If health = Fair, effect is (0.72 + 1.29) = +2.01
* Effect of health = Fair instead of Excellent?
  + If exerany = 0 (no), effect is -1.73
  + If exerany = 1 (yes), effect is (-1.73 + 1.29) = -0.44

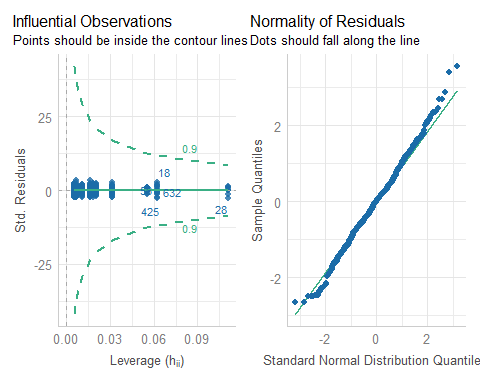
## Checking model fit2 (*n* = 670)

check\_model(fit2, check = c("linearity", "homogeneity"))



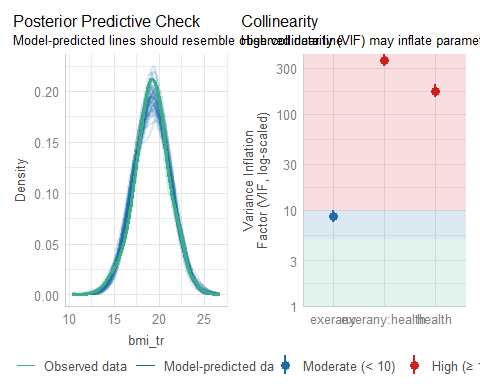
## Checking model fit2 (*n* = 670)

check\_model(fit2, check = c("outliers", "qq"), detrend = FALSE)



## Checking model fit2 (*n* = 670)

check\_model(fit2, check = c("pp\_check", "vif"))



# Incorporating a Covariate into our two-way ANOVA models

## Add fruit\_c to fit1

fit3 <- lm(bmi\_tr ~ fruit\_c + exerany + health, data = train\_smt\_im)

* How well does this model fit the training data?

bind\_rows(glance(fit1), glance(fit3)) |>  
 mutate(mod = c("fit1", "fit3")) |>  
 select(mod, r.sq = r.squared, adj.r.sq = adj.r.squared,   
 sigma, df, df.res = df.residual, AIC, BIC) |>   
 gt() |> fmt\_number(columns = r.sq:sigma, decimals = 3) |>  
 fmt\_number(columns = AIC:BIC, decimals = 1) |>  
 tab\_options(table.font.size = 24)

| mod | r.sq | adj.r.sq | sigma | df | df.res | AIC | BIC |
| --- | --- | --- | --- | --- | --- | --- | --- |
| fit1 | 0.069 | 0.062 | 1.925 | 5 | 664 | 2,786.8 | 2,818.3 |
| fit3 | 0.078 | 0.069 | 1.917 | 6 | 663 | 2,782.6 | 2,818.6 |

## ANOVA for the fit3 model

tidy(anova(fit3)) |> gt() |>   
 fmt\_number(columns = sumsq:statistic, decimals = 2) |>  
 fmt\_number(columns = p.value, decimals = 4) |>  
 tab\_options(table.font.size = 24)

| term | df | sumsq | meansq | statistic | p.value |
| --- | --- | --- | --- | --- | --- |
| fruit\_c | 1 | 42.86 | 42.86 | 11.66 | 0.0007 |
| exerany | 1 | 53.21 | 53.21 | 14.48 | 0.0002 |
| health | 4 | 109.19 | 27.30 | 7.43 | 0.0000 |
| Residuals | 663 | 2,437.18 | 3.68 | NA | NA |

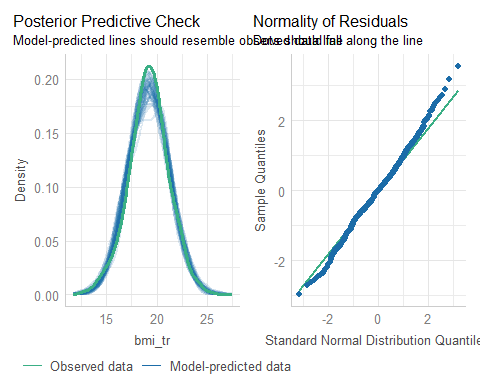
## fit3 coefficients

tidy(fit3, conf.int = TRUE, conf.level = 0.90) |>  
 gt() |> fmt\_number(columns = estimate:conf.high, decimals = 3) |>  
 tab\_options(table.font.size = 24)

| term | estimate | std.error | statistic | p.value | conf.low | conf.high |
| --- | --- | --- | --- | --- | --- | --- |
| (Intercept) | 19.306 | 0.232 | 83.170 | 0.000 | 18.924 | 19.689 |
| fruit\_c | 0.164 | 0.066 | 2.484 | 0.013 | 0.055 | 0.273 |
| exerany | 0.529 | 0.172 | 3.077 | 0.002 | 0.246 | 0.812 |
| healthVG | -0.183 | 0.221 | -0.829 | 0.408 | -0.546 | 0.181 |
| healthG | -0.920 | 0.227 | -4.047 | 0.000 | -1.294 | -0.545 |
| healthF | -0.923 | 0.285 | -3.246 | 0.001 | -1.392 | -0.455 |
| healthP | -1.083 | 0.426 | -2.542 | 0.011 | -1.785 | -0.381 |

## Checking model fit3 (*n* = 670)

check\_model(fit3, detrend = FALSE, check = c("pp\_check", "qq"))



## Include the interaction term?

fit4 <- lm(bmi\_tr ~ fruit\_c + exerany \* health,   
 data = train\_smt\_im)

### ANOVA for the fit4 model

tidy(anova(fit4)) |> gt() |>   
 fmt\_number(columns = sumsq:statistic, decimals = 2) |>  
 fmt\_number(columns = p.value, decimals = 4) |>  
 tab\_options(table.font.size = 20)

| term | df | sumsq | meansq | statistic | p.value |
| --- | --- | --- | --- | --- | --- |
| fruit\_c | 1 | 42.86 | 42.86 | 11.94 | 0.0006 |
| exerany | 1 | 53.21 | 53.21 | 14.83 | 0.0001 |
| health | 4 | 109.19 | 27.30 | 7.61 | 0.0000 |
| exerany:health | 4 | 72.14 | 18.04 | 5.03 | 0.0005 |
| Residuals | 659 | 2,365.03 | 3.59 | NA | NA |

## fit4 coefficients

tidy(fit4, conf.int = TRUE, conf.level = 0.90) |>  
 gt() |> fmt\_number(columns = estimate:conf.high, decimals = 3) |>  
 tab\_options(table.font.size = 18)

| term | estimate | std.error | statistic | p.value | conf.low | conf.high |
| --- | --- | --- | --- | --- | --- | --- |
| (Intercept) | 19.211 | 0.447 | 43.014 | 0.000 | 18.475 | 19.946 |
| fruit\_c | 0.180 | 0.065 | 2.757 | 0.006 | 0.073 | 0.288 |
| exerany | 0.640 | 0.489 | 1.308 | 0.191 | -0.166 | 1.445 |
| healthVG | 0.339 | 0.514 | 0.658 | 0.511 | -0.509 | 1.186 |
| healthG | -0.565 | 0.509 | -1.110 | 0.267 | -1.404 | 0.273 |
| healthF | -1.712 | 0.558 | -3.067 | 0.002 | -2.632 | -0.793 |
| healthP | -1.913 | 0.774 | -2.471 | 0.014 | -3.188 | -0.638 |
| exerany:healthVG | -0.662 | 0.568 | -1.165 | 0.245 | -1.597 | 0.274 |
| exerany:healthG | -0.471 | 0.568 | -0.829 | 0.407 | -1.406 | 0.464 |
| exerany:healthF | 1.357 | 0.651 | 2.084 | 0.038 | 0.284 | 2.431 |
| exerany:healthP | 1.331 | 0.929 | 1.432 | 0.153 | -0.200 | 2.862 |

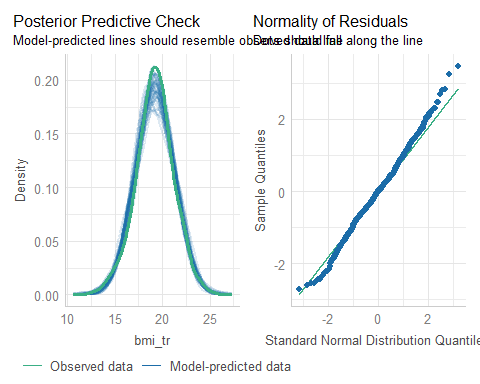
## ANOVA: Compare fit3 & fit4

anova(fit3, fit4)

Analysis of Variance Table  
  
Model 1: bmi\_tr ~ fruit\_c + exerany + health  
Model 2: bmi\_tr ~ fruit\_c + exerany \* health  
 Res.Df RSS Df Sum of Sq F Pr(>F)   
1 663 2437.2   
2 659 2365.0 4 72.145 5.0257 0.000539 \*\*\*  
---  
Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

## Checking model fit4 (*n* = 670)

check\_model(fit4, detrend = FALSE, check = c("pp\_check", "qq"))



# Comparing Our Models

## Which of the four models fits best?

In the **training** sample, our model results (note ordering):

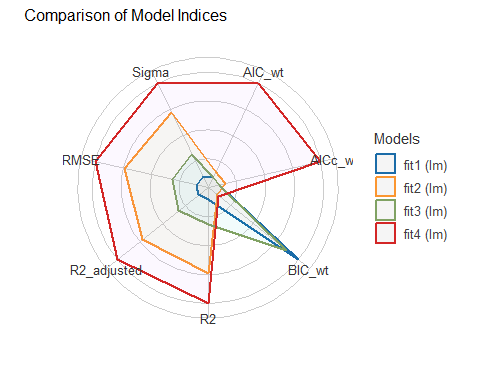
bind\_rows(glance(fit1), glance(fit3), glance(fit2), glance(fit4)) |>  
 mutate(mod = c("fit1", "fit3", "fit2", "fit4")) |>  
 select(mod, r.sq = r.squared, adj.r.sq = adj.r.squared,   
 sigma, df, df.res = df.residual, AIC, BIC) |>   
 gt() |> fmt\_number(columns = r.sq:sigma, decimals = 3) |>  
 fmt\_number(columns = AIC:BIC, decimals = 1) |>  
 tab\_options(table.font.size = 20)

| mod | r.sq | adj.r.sq | sigma | df | df.res | AIC | BIC |
| --- | --- | --- | --- | --- | --- | --- | --- |
| fit1 | 0.069 | 0.062 | 1.925 | 5 | 664 | 2,786.8 | 2,818.3 |
| fit3 | 0.078 | 0.069 | 1.917 | 6 | 663 | 2,782.6 | 2,818.6 |
| fit2 | 0.095 | 0.082 | 1.904 | 9 | 660 | 2,776.1 | 2,825.7 |
| fit4 | 0.105 | 0.091 | 1.894 | 10 | 659 | 2,770.4 | 2,824.5 |

* Adjusted , and AIC all improve as we move down this table. BIC likes fit1 and fit3.
* The training sample is the data our models have *already seen*, so we should be cautious.

## Comparison Plot: In-Sample Performance

plot(compare\_performance(fit1, fit2, fit3, fit4))



## What does augment() give us?

fit1\_test\_aug <- augment(fit1, newdata = test\_smt\_im)   
fit1\_test\_aug |> select(ID, bmi\_tr, bmi, .fitted, .resid, health, exerany) |>  
 slice(198:202) |> gt() |>   
 fmt\_number(columns = bmi\_tr:.resid, decimals = 2) |>  
 tab\_options(table.font.size = 20)

| ID | bmi\_tr | bmi | .fitted | .resid | health | exerany |
| --- | --- | --- | --- | --- | --- | --- |
| 1016 | 18.75 | 28.44 | 18.22 | 0.53 | P | 0 |
| 1018 | 19.36 | 26.68 | 19.88 | -0.52 | E | 1 |
| 1019 | 19.71 | 25.74 | 18.32 | 1.39 | F | 0 |
| 1020 | 22.05 | 20.57 | 19.68 | 2.37 | VG | 1 |
| 1024 | 20.19 | 24.52 | 18.34 | 1.85 | G | 0 |

Here, .fitted = predicted bmi\_tr and .resid = bmi\_tr - .fitted.

## Back-Transformation of bmi\_tr

Our models predict bmi\_tr = , but we want to predict bmi. How do we convert predicted to predicted bmi?

$$
1 / \left(\frac{100}{\sqrt{bmi}}\right) = \sqrt{bmi} / 100, \\
\mbox{so } 100 / \left(\frac{100}{\sqrt{bmi}}\right) = \sqrt{bmi}, \\
\mbox{and so } \left[100 / \left(\frac{100}{\sqrt{bmi}}\right)\right]^2 = bmi
$$

## augment() with results for bmi

We use for predicted bmi, then errors are bmi\_res = observed bmi - predicted bmi.

fit1\_test\_aug <- augment(fit1, newdata = test\_smt\_im) |>   
 mutate(bmi\_fit = (100/.fitted)^2, bmi\_res = bmi - bmi\_fit)  
  
fit1\_test\_aug |> select(ID, bmi, bmi\_fit, bmi\_res,   
 bmi\_tr, .fitted, .resid, exerany, health, fruit\_c) |>  
 slice(5:6) |> gt() |>   
 fmt\_number(columns = c(bmi:.resid, fruit\_c), decimals = 2) |>  
 tab\_options(table.font.size = 24)

| ID | bmi | bmi\_fit | bmi\_res | bmi\_tr | .fitted | .resid | exerany | health | fruit\_c |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 21 | 18.83 | 25.83 | -7.00 | 23.04 | 19.68 | 3.37 | 1 | VG | -0.43 |
| 29 | 44.66 | 29.72 | 14.94 | 14.96 | 18.34 | -3.38 | 0 | G | -0.72 |

## Augment all four models so far…

fit1\_test\_aug <- augment(fit1, newdata = test\_smt\_im) |>  
 mutate(bmi\_fit = (100/.fitted)^2, bmi\_res = bmi - bmi\_fit)  
  
fit2\_test\_aug <- augment(fit2, newdata = test\_smt\_im) |>  
 mutate(bmi\_fit = (100/.fitted)^2, bmi\_res = bmi - bmi\_fit)  
  
fit3\_test\_aug <- augment(fit3, newdata = test\_smt\_im) |>  
 mutate(bmi\_fit = (100/.fitted)^2, bmi\_res = bmi - bmi\_fit)  
  
fit4\_test\_aug <- augment(fit4, newdata = test\_smt\_im) |>  
 mutate(bmi\_fit = (100/.fitted)^2, bmi\_res = bmi - bmi\_fit)

## Four Key Error Summaries

We’ll look at all four of these summaries when we do linear regression, usually.

* Mean absolute prediction error (MAPE)
* Maximum absolute prediction error (Max. Error)
* Square root of mean squared prediction error (RMSPE)
* Squared correlation of observed and predicted bmi (validated )

## Key Summaries for fit1

fit1\_esum <- fit1\_test\_aug |>  
 summarise(MAPE = mean(abs(bmi\_res)),  
 Max\_E = max(abs(bmi\_res)),  
 RMSPE = sqrt(mean(bmi\_res^2)),  
 Val\_R2 = cor(bmi, bmi\_fit)^2) |>  
 mutate(Model = "fit1")  
  
fit1\_esum

# A tibble: 1 × 5  
 MAPE Max\_E RMSPE Val\_R2 Model  
 <dbl> <dbl> <dbl> <dbl> <chr>  
1 4.31 21.1 5.63 0.0752 fit1

* I built the key summaries for fit2, fit3 and fit4 in the same way (included in code, not shown in slides.)

fit2\_esum <- fit2\_test\_aug |>  
 summarise(MAPE = mean(abs(bmi\_res)),  
 Max\_E = max(abs(bmi\_res)),  
 RMSPE = sqrt(mean(bmi\_res^2)),  
 Val\_R2 = cor(bmi, bmi\_fit)^2) |>  
 mutate(Model = "fit2")  
  
fit3\_esum <- fit3\_test\_aug |>  
 summarise(MAPE = mean(abs(bmi\_res)),  
 Max\_E = max(abs(bmi\_res)),  
 RMSPE = sqrt(mean(bmi\_res^2)),  
 Val\_R2 = cor(bmi, bmi\_fit)^2) |>  
 mutate(Model = "fit3")  
  
fit4\_esum <- fit4\_test\_aug |>  
 summarise(MAPE = mean(abs(bmi\_res)),  
 Max\_E = max(abs(bmi\_res)),  
 RMSPE = sqrt(mean(bmi\_res^2)),  
 Val\_R2 = cor(bmi, bmi\_fit)^2) |>  
 mutate(Model = "fit4")

## Compare Models in Test Sample

bind\_rows(fit1\_esum, fit2\_esum, fit3\_esum, fit4\_esum) |>  
 relocate(Model) |> gt() |> fmt\_number(decimals = 3) |>  
 tab\_options(table.font.size = 24)

| Model | MAPE | Max\_E | RMSPE | Val\_R2 |
| --- | --- | --- | --- | --- |
| fit1 | 4.311 | 21.061 | 5.628 | 0.075 |
| fit2 | 4.492 | 19.512 | 5.837 | 0.037 |
| fit3 | 4.328 | 21.047 | 5.651 | 0.068 |
| fit4 | 4.515 | 19.991 | 5.875 | 0.034 |

|  |
| --- |
| Our Four Models |
| * fit1: exerany and health main effects; fit2: add interaction * fit3: add fruit\_c to fit1; fit4: add fruit\_c to fit2 |

## Next up…

Basics of logistic regression fitting and evaluation

* What if we have a binary (yes/no or 1/0) outcome?
* Predict “whether or not BMI < 30”, rather than BMI?
  + A linear probability model as a first idea
  + Using glm() rather than lm() to get a logistic model
  + Coefficients as log(odds ratios)
  + Changes in how we measure the model’s performance
  + Changes in the assumptions we make

# Appendix

## Creating Today’s Data Set

url1 <- "https://raw.githubusercontent.com/THOMASELOVE/432-data/master/data/smart\_ohio.csv"  
  
smart\_ohio <- read\_csv(url1)  
  
smt <- 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) |>   
 mutate(across(where(is.character), as\_factor)) |>  
 mutate(ID = as.character(SEQNO - 2017000000)) |>  
 relocate(ID)

## Codebook for useful smt variables (1)

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

| Variable | Description |
| --- | --- |
| bmi | (outcome) Body-Mass index in . |
| inc\_imp | income (imputed from grouped values) in $ |
| fruit\_day | average fruit servings consumed per day |
| drinks\_wk | average weekly alcoholic drinks consumed |
| female | sex: 1 = female, 0 = male |

## Codebook for useful smt variables (2)

* 894 subjects in Cleveland-Elyria without diabetes

| Variable | Description |
| --- | --- |
| exerany | any exercise in 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
* See [Course Notes Chapter 6](https://thomaselove.github.io/432-notes/smart.html) on BRFSS SMART data

## Basic Data Summaries

Available approaches include:

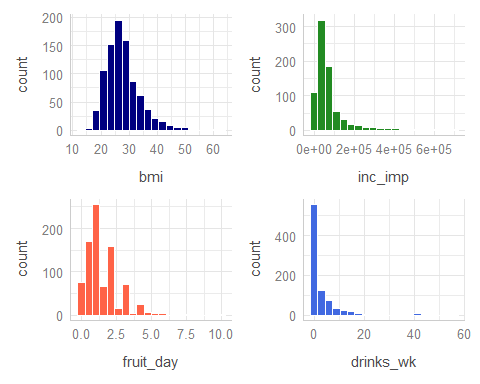
* data\_codebook() from datawizard in easystats
* Hmisc package’s describe(), or
* summary()

all of which can work nicely in an HTML presentation, but none of them fit well on a slide.

## Histogram of each quantity

|  |
| --- |
| Note |
| I used #| warning: false in this code chunk to avoid warnings about missing values, like this one for inc\_imp:  Warning: Removed 120 rows containing non-finite values |

p1 <- ggplot(smt, aes(x = bmi)) +   
 geom\_histogram(fill = "navy", col = "white", bins = 20)  
p2 <- ggplot(smt, aes(x = inc\_imp)) +   
 geom\_histogram(fill = "forestgreen", col = "white", bins = 20)  
p3 <- ggplot(smt, aes(x = fruit\_day)) +   
 geom\_histogram(fill = "tomato", col = "white", bins = 20)  
p4 <- ggplot(smt, aes(x = drinks\_wk)) +   
 geom\_histogram(fill = "royalblue", col = "white", bins = 20)  
  
(p1 + p2) / (p3 + p4)



## Binary variables in raw smt

smt |> tabyl(female, exerany) |> adorn\_title()

exerany   
 female 0 1 NA\_  
 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?

## Multicategorical genhealth in raw smt

smt |> tabyl(genhealth)

genhealth n percent valid\_percent  
 1\_Excellent 148 0.165548098 0.16573348  
 3\_Good 274 0.306487696 0.30683091  
 2\_VeryGood 324 0.362416107 0.36282195  
 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 treated as missing.
* How might we manage this variable?

## Changing the levels for genhealth

smt <- smt |>  
 mutate(health =   
 fct\_recode(genhealth,  
 E = "1\_Excellent",  
 VG = "2\_VeryGood",  
 G = "3\_Good",  
 F = "4\_Fair",  
 P = "5\_Poor"),  
 health = fct\_relevel(health, "E", "VG", "G", "F", "P"))

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

## Checking health vs. genhealth

smt |> tabyl(genhealth, health) |> adorn\_title()

health   
 genhealth E VG G F P NA\_  
 1\_Excellent 148 0 0 0 0 0  
 3\_Good 0 0 274 0 0 0  
 2\_VeryGood 0 324 0 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 adjusted the order to something more sensible, retained the missing value, and we have much shorter labels.

## Multicategorical race\_eth in raw smt

smt |> count(race\_eth)

# A tibble: 6 × 2  
 race\_eth n  
 <fct> <int>  
1 White non-Hispanic 646  
2 Other race non-Hispanic 22  
3 Black non-Hispanic 167  
4 Multiracial non-Hispanic 19  
5 Hispanic 27  
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 category.
* 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…

smt <- smt |>  
 mutate(race\_eth = fct\_infreq(race\_eth))  
  
smt |> 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

smt <- smt |>  
 select(ID, bmi, inc\_imp, fruit\_day, drinks\_wk,   
 female, exerany, health, race\_eth)  
  
miss\_var\_summary(smt)

# A tibble: 9 × 3  
 variable n\_miss pct\_miss  
 <chr> <int> <num>  
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 ID 0 0   
8 bmi 0 0   
9 female 0 0

## Single Imputation with mice

smt\_im <- mice(smt, m = 1, seed = 20250121, print = FALSE) |>  
 complete() |>  
 tibble()

Warning: Number of logged events: 1

|  |
| --- |
| Note |
| You may get a logged event for the ID variable expressed as a character, and that can be ignored. |

prop\_miss\_case(smt\_im)

[1] 0

dim(smt\_im)

[1] 894 9

## Saving the tidied data

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

write\_rds(smt, "c03/data/smt.Rds")  
  
write\_rds(smt\_im, "c03/data/smt\_im.Rds")

To reload these files, we’ll use read\_rds().

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