431 Class 19

https://thomaselove.github.io/431-2024/

2024-10-31

## Today’s Agenda

1. Exploration and Initial Data Summaries
2. Dealing with Missing Data (today via Single Imputation)
3. Partitioning into Training and Testing Samples
4. How might we transform our outcome? (Box-Cox? Scaling?)
5. Building Three Candidate Prediction Models
   * Assessing coefficients (model parameters)
   * Obtaining summaries of fit quality (performance)
6. Comparing some in-sample performance indices.

## 431 strategy: “most useful” model?

1. Split the data into a development (model training) sample of about 70-80% of the observations, and a holdout (model test) sample, containing the remaining observations.
2. Develop candidate models using the development sample.
3. Assess the quality of fit for candidate models within the development sample.

## 431 strategy: “most useful” model?

1. Check adherence to regression assumptions in the development sample.
2. When you have candidates, assess them based on the accuracy of the predictions they make for the data held out (and thus not used in building the models.)
3. Select a “final” model for use based on the evidence in steps 3, 4 and especially 5.

## R Packages

knitr::opts\_chunk$set(comment = NA)  
  
library(janitor)

Attaching package: 'janitor'

The following objects are masked from 'package:stats':  
  
 chisq.test, fisher.test

library(mice)

Attaching package: 'mice'

The following object is masked from 'package:stats':  
  
 filter

The following objects are masked from 'package:base':  
  
 cbind, rbind

library(naniar)  
library(patchwork)  
library(car)

Loading required package: carData

library(GGally) ## for ggpairs scatterplot matrix

Loading required package: ggplot2

Registered S3 method overwritten by 'GGally':  
 method from   
 +.gg ggplot2

library(easystats)

# Attaching packages: easystats 0.7.3  
✔ bayestestR 0.15.0 ✔ correlation 0.8.6   
✔ datawizard 0.13.0 ✔ effectsize 0.8.9   
✔ insight 0.20.5 ✔ modelbased 0.8.9   
✔ performance 0.12.4 ✔ parameters 0.23.0  
✔ report 0.5.9 ✔ see 0.9.0

library(tidyverse)

── Attaching core tidyverse packages ──────────────────────── tidyverse 2.0.0 ──  
✔ dplyr 1.1.4 ✔ readr 2.1.5  
✔ forcats 1.0.0 ✔ stringr 1.5.1  
✔ lubridate 1.9.3 ✔ tibble 3.2.1  
✔ purrr 1.0.2 ✔ tidyr 1.3.1

── Conflicts ────────────────────────────────────────── tidyverse\_conflicts() ──  
✖ dplyr::filter() masks mice::filter(), stats::filter()  
✖ dplyr::lag() masks stats::lag()  
✖ dplyr::recode() masks car::recode()  
✖ purrr::some() masks car::some()  
ℹ Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors

source("c19/data/Love-431.R")  
  
theme\_set(theme\_bw())

## Today’s Data

The dm500.Rds data contains four important variables + Subject ID on 500 adults with diabetes.

We want to predict the subject’s current Hemoglobin A1c level (a1c), using (up to) three predictors:

* a1c\_old: subject’s Hemoglobin A1c (in %) two years ago
* age: subject’s age in years (between 30 and 70)
* income: median income of subject’s neighborhood (3 levels)

## What roles will these variables play?

a1c is our outcome, which we’ll predict using three models …

1. Model 1: Use a1c\_old alone to predict a1c
2. Model 2: Use a1c\_old and age together to predict a1c
3. Model 3: Use a1c\_old, age, and income together to predict a1c

## The dm500 data

dm500 <- readRDS("c19/data/dm500.Rds")  
  
dm500

# A tibble: 500 × 5  
 a1c a1c\_old age income subject  
 <dbl> <dbl> <dbl> <fct> <chr>   
 1 6.3 11.4 62 Higher\_than\_50K S-001   
 2 11 16.3 54 Between\_30-50K S-002   
 3 8.7 10.7 47 <NA> S-003   
 4 6.5 5.8 53 Below\_30K S-004   
 5 6.7 6.3 64 Between\_30-50K S-005   
 6 5.8 6.5 48 Below\_30K S-006   
 7 9.6 NA 49 Below\_30K S-007   
 8 6.1 7.2 63 Between\_30-50K S-008   
 9 12.9 7.7 55 Below\_30K S-009   
10 6.7 6.8 63 Below\_30K S-010   
# ℹ 490 more rows

## More details on missing data

* What do we learn here?

miss\_var\_summary(dm500)

# A tibble: 5 × 3  
 variable n\_miss pct\_miss  
 <chr> <int> <num>  
1 a1c\_old 15 3   
2 income 5 1   
3 a1c 4 0.8  
4 age 0 0   
5 subject 0 0

miss\_case\_table(dm500)

# A tibble: 3 × 3  
 n\_miss\_in\_case n\_cases pct\_cases  
 <int> <int> <dbl>  
1 0 479 95.8  
2 1 18 3.6  
3 2 3 0.6

## Single Imputation

Today, we’ll assume all missing values are Missing at Random (MAR) and create 10 imputations but just use the 7th.

set.seed(20241031)  
  
dm500\_tenimps <- mice(dm500, m = 10, printFlag = FALSE)

Warning: Number of logged events: 1

dm500\_i <- complete(dm500\_tenimps, 7) |> tibble()  
  
n\_miss(dm500)

[1] 24

n\_miss(dm500\_i)

[1] 0

* Later, we’ll return and use all 10 imputations.

## Summarizing the dm500\_i tibble

data\_codebook(dm500\_i |> select(-subject))

select(dm500\_i, -subject) (500 rows and 4 variables, 4 shown)  
  
ID | Name | Type | Missings | Values | N  
---+---------+-------------+----------+-----------------+------------  
1 | a1c | numeric | 0 (0.0%) | [4.3, 16.7] | 500  
---+---------+-------------+----------+-----------------+------------  
2 | a1c\_old | numeric | 0 (0.0%) | [4.2, 16.3] | 500  
---+---------+-------------+----------+-----------------+------------  
3 | age | numeric | 0 (0.0%) | [31, 70] | 500  
---+---------+-------------+----------+-----------------+------------  
4 | income | categorical | 0 (0.0%) | Higher\_than\_50K | 124 (24.8%)  
 | | | | Between\_30-50K | 198 (39.6%)  
 | | | | Below\_30K | 178 (35.6%)  
---------------------------------------------------------------------

## Three candidate models for a1c

Our goal is accurate prediction of a1c values. Suppose we have decided to consider these three possible models…

1. Model 1: Use a1c\_old alone to predict a1c
2. Model 2: Use a1c\_old and age together to predict a1c
3. Model 3: Use a1c\_old, age, and income together to predict a1c

## How shall we be guided by our data?

It can scarcely be denied that the supreme goal of all theory is to make the irreducible basic elements as simple and as few as possible without having to surrender the adequate representation of a single datum of experience. (A. Einstein)

* Often, this is reduced to “make everything as simple as possible but no simpler”

## How shall we be guided by our data?

Entities should not be multiplied without necessity. (Occam’s razor)

* Often, this is reduced to “the simplest solution is most likely the right one”

## George Box’s aphorisms

On Parsimony: Since all models are wrong the scientist cannot obtain a “correct” one by excessive elaboration. On the contrary following William of Occam he should seek an economical description of natural phenomena. Just as the ability to devise simple but evocative models is the signature of the great scientist so overelaboration and overparameterization is often the mark of mediocrity.

## George Box’s aphorisms

On Worrying Selectively: Since all models are wrong the scientist must be alert to what is importantly wrong. It is inappropriate to be concerned about mice when there are tigers abroad.

* and, the most familiar version…

… all models are approximations. Essentially, all models are wrong, but some are useful. However, the approximate nature of the model must always be borne in mind.

# Partition the data: Training and Test Samples

## Partitioning the Data

* Select a random sample (without replacement) of 70% of dm500\_i (60-80% is common) for model training.
* Hold out the other 30% for model testing, using anti\_join() to pull subjects not in dm500\_i\_train.

set.seed(4312024)  
  
dm500\_i\_train <- dm500\_i |>   
 slice\_sample(prop = 0.7, replace = FALSE)  
dm500\_i\_test <-   
 anti\_join(dm500\_i, dm500\_i\_train, by = "subject")  
  
c(nrow(dm500\_i\_train), nrow(dm500\_i\_test), nrow(dm500\_i))

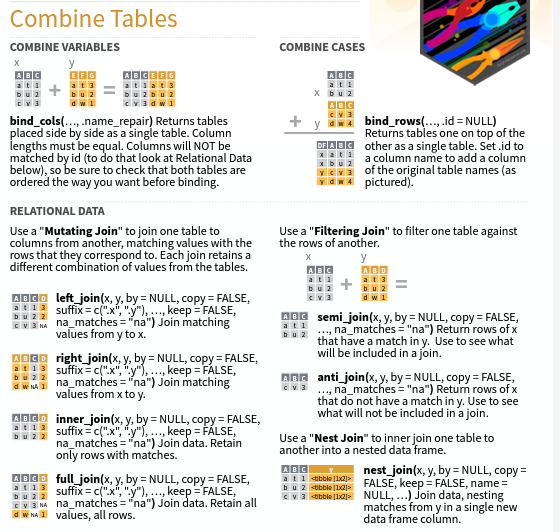
[1] 350 150 500

## Describing the join options

from [Posit’s Data Transformation Cheat Sheet](https://rstudio.github.io/cheatsheets/html/data-transformation.html)

“Mutating Joins” join one table to columns from another, matching values with the rows that the correspond to. Each join retains a different combination of values from the tables.

* left\_join(x, y): Join matching values from y to x.
* right\_join(x, y): Join matching values from x to y.
* inner\_join(x, y): Join data. retain only rows with matches.
* full\_join(x, y): Join data. Retain all values, all rows.



## Describing the join options

from [Posit’s Data Transformation Cheat Sheet](https://rstudio.github.io/cheatsheets/html/data-transformation.html)

“Filtering Joins” filter one table against the rows of another.

* semi\_join(x, y): Return rows of x that have a match in y. Use to see what will be included in a join.
* anti\_join(x, y): Return rows of x that do not have a match in y. Use to see what will not be included in a join.

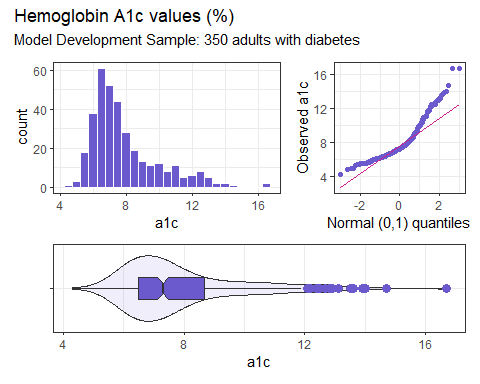
Use by = join\_by(col1, col2, ...) to specify one or more common columns to match on.

* For more, see [the Joins chapter in R4DS](https://r4ds.hadley.nz/joins)

# Consider transforming the outcome.

## Distribution of a1c (outcome)

p1 <- ggplot(dm500\_i\_train, aes(x = a1c)) +  
 geom\_histogram(binwidth = 0.5,   
 fill = "slateblue", col = "white")  
  
p2 <- ggplot(dm500\_i\_train, aes(sample = a1c)) +   
 geom\_qq(col = "slateblue") + geom\_qq\_line(col = "violetred") +  
 labs(y = "Observed a1c", x = "Normal (0,1) quantiles") +   
 theme(aspect.ratio = 1)  
  
p3 <- ggplot(dm500\_i\_train, aes(x = "", y = a1c)) +  
 geom\_violin(fill = "slateblue", alpha = 0.1) +   
 geom\_boxplot(fill = "slateblue", width = 0.3, notch = TRUE,  
 outlier.color = "slateblue", outlier.size = 3) +  
 labs(x = "") + coord\_flip()  
  
p1 + p2 - p3 +  
 plot\_layout(ncol = 1, height = c(3, 2)) +   
 plot\_annotation(title = "Hemoglobin A1c values (%)",  
 subtitle = str\_glue("Model Development Sample: ", nrow(dm500\_i\_train),   
 " adults with diabetes"))



## Transform the Outcome?

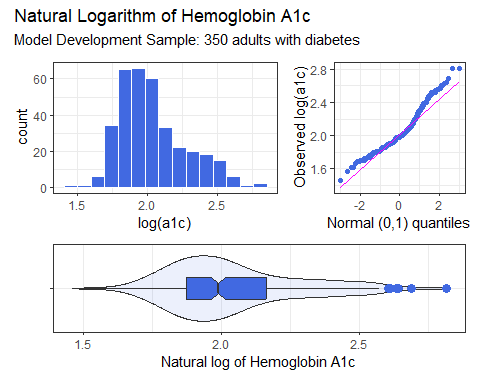
We want to try to identify a good transformation for the conditional distribution of the outcome, given the predictors, in an attempt to make the linear regression assumptions of linearity, Normality and constant variance more appropriate.

### (partial) Ladder of Power Transformations

| Transformation |  | y |  | log(y) |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | 2 | 1 | 0.5 | 0 | -1 | -2 |

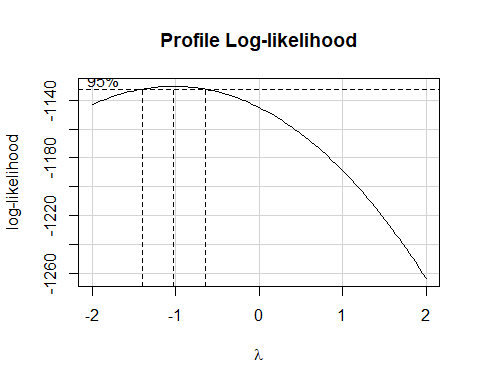
## Consider a log transformation?

p1 <- ggplot(dm500\_i\_train, aes(x = log(a1c))) +  
 geom\_histogram(bins = 15,   
 fill = "royalblue", col = "white")  
  
p2 <- ggplot(dm500\_i\_train, aes(sample = log(a1c))) +   
 geom\_qq(col = "royalblue") + geom\_qq\_line(col = "magenta") +  
 labs(y = "Observed log(a1c)", x = "Normal (0,1) quantiles") +   
 theme(aspect.ratio = 1)  
   
  
p3 <- ggplot(dm500\_i\_train, aes(x = "", y = log(a1c))) +  
 geom\_violin(fill = "royalblue", alpha = 0.1) +   
 geom\_boxplot(fill = "royalblue", width = 0.3, notch = TRUE,  
 outlier.color = "royalblue", outlier.size = 3) +  
 labs(x = "", y = "Natural log of Hemoglobin A1c") + coord\_flip()  
  
p1 + p2 - p3 +  
 plot\_layout(ncol = 1, height = c(3, 2)) +   
 plot\_annotation(title = "Natural Logarithm of Hemoglobin A1c",  
 subtitle = str\_glue("Model Development Sample: ", nrow(dm500\_i\_train),   
 " adults with diabetes"))



## Box-Cox to get started?

mod\_0 <- lm(a1c ~ a1c\_old + age + income,   
 data = dm500\_i\_train)  
boxCox(mod\_0) ## from car package



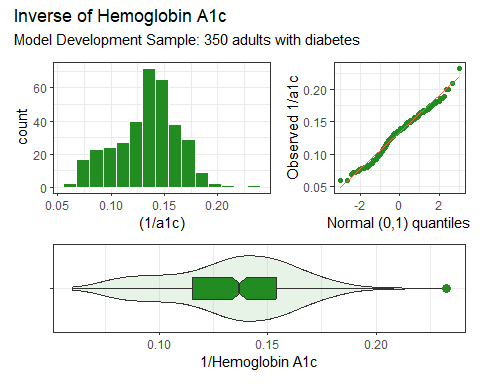
## Could Box-Cox be helpful?

summary(powerTransform(mod\_0)) ## also from car package

bcPower Transformation to Normality   
 Est Power Rounded Pwr Wald Lwr Bnd Wald Upr Bnd  
Y1 -1.0208 -1 -1.396 -0.6456  
  
Likelihood ratio test that transformation parameter is equal to 0  
 (log transformation)  
 LRT df pval  
LR test, lambda = (0) 29.14439 1 6.718e-08  
  
Likelihood ratio test that no transformation is needed  
 LRT df pval  
LR test, lambda = (1) 116.6815 1 < 2.22e-16

## Consider the inverse?

p1 <- ggplot(dm500\_i\_train, aes(x = (1/a1c))) +  
 geom\_histogram(bins = 15,   
 fill = "forestgreen", col = "white")  
  
p2 <- ggplot(dm500\_i\_train, aes(sample = (1/a1c))) +   
 geom\_qq(col = "forestgreen") + geom\_qq\_line(col = "tomato") +  
 labs(y = "Observed 1/a1c", x = "Normal (0,1) quantiles") +   
 theme(aspect.ratio = 1)  
  
p3 <- ggplot(dm500\_i\_train, aes(x = "", y = (1/a1c))) +  
 geom\_violin(fill = "forestgreen", alpha = 0.1) +   
 geom\_boxplot(fill = "forestgreen", width = 0.3, notch = TRUE,  
 outlier.color = "forestgreen", outlier.size = 3) +  
 labs(x = "", y = "1/Hemoglobin A1c") + coord\_flip()  
  
p1 + p2 - p3 +  
 plot\_layout(ncol = 1, height = c(3, 2)) +   
 plot\_annotation(title = "Inverse of Hemoglobin A1c",  
 subtitle = str\_glue("Model Development Sample: ", nrow(dm500\_i\_train),   
 " adults with diabetes"))



## Scale the inverse A1c?

dm500\_i\_train |> reframe(lovedist(1/a1c))

# A tibble: 1 × 10  
 n miss mean sd med mad min q25 q75 max  
 <int> <int> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>  
1 350 0 0.134 0.0291 0.137 0.0286 0.0599 0.115 0.154 0.233

dm500\_i\_train |> reframe(lovedist(100/a1c))

# A tibble: 1 × 10  
 n miss mean sd med mad min q25 q75 max  
 <int> <int> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>  
1 350 0 13.4 2.91 13.7 2.86 5.99 11.5 15.4 23.3

* If we use 1/A1c as our outcome, we’ll have some very small regression coefficients.
* Multiplying by 100 to get 100/A1c yields a new outcome with values between 5.99 and 23.3 instead of 0.0599 and 0.233.

## Correlation Matrix

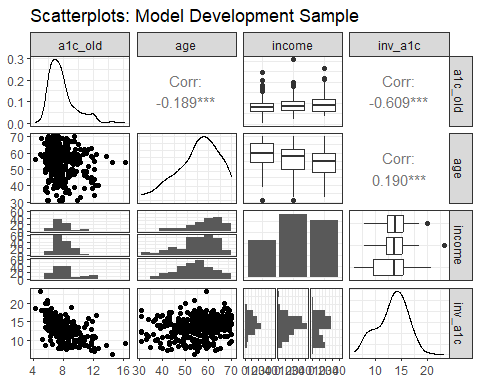
temp <- dm500\_i\_train |>   
 mutate(inv\_a1c = 100/a1c) |>  
 select(a1c\_old, age, income, inv\_a1c)  
  
correlation(temp)

# Correlation Matrix (pearson-method)  
  
Parameter1 | Parameter2 | r | 95% CI | t(348) | p  
---------------------------------------------------------------------  
a1c\_old | age | -0.19 | [-0.29, -0.09] | -3.59 | < .001\*\*\*  
a1c\_old | inv\_a1c | -0.61 | [-0.67, -0.54] | -14.34 | < .001\*\*\*  
age | inv\_a1c | 0.19 | [ 0.09, 0.29] | 3.60 | < .001\*\*\*  
  
p-value adjustment method: Holm (1979)  
Observations: 350

## Scatterplot Matrix

* I select the outcome last. Then, the bottom row will show the most important scatterplots, with the outcome on the Y axis, and each predictor, in turn on the X.
* ggpairs() comes from the GGally package.

temp <- dm500\_i\_train |>   
 mutate(inv\_a1c = 100/a1c) |>  
 select(a1c\_old, age, income, inv\_a1c)  
  
ggpairs(temp,   
 title = "Scatterplots: Model Development Sample",  
 lower = list(combo = wrap("facethist", bins = 10)))



## Three Regression Models We’ll Fit

* We continue to use the model training sample, and work with the (100/a1c) transformation.

fit1 <- lm((100/a1c) ~ a1c\_old, data = dm500\_i\_train)  
  
fit2 <- lm((100/a1c) ~ a1c\_old + age, data = dm500\_i\_train)  
  
fit3 <- lm((100/a1c) ~ a1c\_old + age + income,   
 data = dm500\_i\_train)  
  
c(n\_obs(fit1), n\_obs(fit2), n\_obs(fit3))

[1] 350 350 350

* n\_obs() gives us the number of observations used to fit the model. It comes from the insight package in easystats.

# Assess fit of candidate models in training sample.

## Estimated coefficients (fit1)

model\_parameters(fit1, ci = 0.95)

Parameter | Coefficient | SE | 95% CI | t(348) | p  
-------------------------------------------------------------------  
(Intercept) | 20.97 | 0.54 | [19.90, 22.04] | 38.49 | < .001  
a1c old | -0.98 | 0.07 | [-1.12, -0.85] | -14.34 | < .001

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

* Code: $$\hat{\frac{100}{A1c}} = 20.97 - 0.98 \mbox{ A1c\_old}$$

## Interpreting the fit1 equation (1/6)

* Interpret the Intercept?
* Our fit1 model estimates the value of (100/A1c) to be 20.97 if A1c\_old = 0, but we have no values of A1c\_old anywhere near 0 in our data[[1]](#footnote-72), nor is such a value plausible clinically, so the intercept doesn’t tell us much here.

## Interpreting the fit1 equation (2/6)

Our fit1 model estimates the slope of A1c\_old to be -0.98, with 95% CI (-1.12, -0.85). Interpret the point estimate…

* Suppose Harry has an A1c\_old value that is one percentage point (A1c is measured in percentage points) higher than Sally’s. Our fit1 model predicts the 100/A1c value for Harry to be 0.98 less than that of Sally, on average.

## Interpreting the fit1 equation (3/6)

Our fit1 model estimates the slope of A1c\_old to be -0.98, with 95% CI (-1.12, -0.85). What can we say about the CI?

* The confidence interval reflects imprecision in the population estimate, based only on assuming that the participants are selected at random from the population of interest.

## Interpreting the fit1 equation (4/6)

Slope of A1c\_old in fit1 is -0.98, with 95% CI (-1.12, -0.85).

* Model fit1 estimates a slope of -0.98 in study participants.
* When we generalize beyond study participants to the population they were selected at random from, then our data are **compatible** (at the 95% confidence level) with population slopes between -1.12 and -0.85, depending on the assumptions of our linear model fit1 being correct.

## Interpreting the fit1 equation (5/6)

* Practically, is our data a random sample of anything?

Slope of A1c\_old in fit1 is -0.98, with 95% CI (-1.12, -0.85).

* Our 95% confidence interval suggests that our data appear compatible with population slope values for A1c\_old between -1.12 and -0.85, assuming the participants are **representative** of the population of interest, **and** assuming the underlying linear model fit1 is correct.

## Interpreting the fit1 equation (6/6)

* Can we say “There is 95% probability that the population slope lies between …”?

To find such a probability interval, we’d need to **combine** our confidence interval (giving compatibility of data with population slope values) with **meaningful prior information**[[2]](#footnote-78) on which values for the population mean are plausible.

* For more, see [this article by Hilary Watt](https://academic.oup.com/ije/article/49/6/2083/5876177) 2020 Int J Epidemiology (<https://doi.org/10.1093/ije/dyaa080>)

## Summarize Fit Quality (fit1)

model\_performance(fit1)

# Indices of model performance  
  
AIC | AICc | BIC | R2 | R2 (adj.) | RMSE | Sigma  
------------------------------------------------------------------  
1583.106 | 1583.176 | 1594.680 | 0.371 | 0.370 | 2.303 | 2.309

* Adjusted = , where = number of predictors in the model, and = number of observations.
  + Adjusted is no longer a percentage of anything, just an index. Higher values, though, still indicate stronger fits.
* RMSE/Sigma = Residual Standard Error -> smaller values = better fit

## Summarize Fit Quality (fit1)

model\_performance(fit1)

# Indices of model performance  
  
AIC | AICc | BIC | R2 | R2 (adj.) | RMSE | Sigma  
------------------------------------------------------------------  
1583.106 | 1583.176 | 1594.680 | 0.371 | 0.370 | 2.303 | 2.309

* AIC = Akaike vs. BIC = Bayesian Information Criterion -> also smaller values = better fit
  + As we’ll see, the compare\_performance() function weights AIC and BIC measures so that higher values indicate a better fit.
  + Without the weights, we look for lower AIC and BIC.

## Estimated coefficients (fit2)

model\_parameters(fit2)

Parameter | Coefficient | SE | 95% CI | t(347) | p  
-------------------------------------------------------------------  
(Intercept) | 19.42 | 1.02 | [17.42, 21.43] | 19.06 | < .001  
a1c old | -0.96 | 0.07 | [-1.10, -0.82] | -13.79 | < .001  
age | 0.02 | 0.01 | [ 0.00, 0.05] | 1.79 | 0.074

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

* Suppose Harry is one year older than Sally and they have the same A1c\_old. On average, our fit2 model predicts Harry’s 100/A1c value to be 0.02 higher than Sally’s.

## Summarize Fit Quality (fit2)

model\_performance(fit1)

# Indices of model performance  
  
AIC | AICc | BIC | R2 | R2 (adj.) | RMSE | Sigma  
------------------------------------------------------------------  
1583.106 | 1583.176 | 1594.680 | 0.371 | 0.370 | 2.303 | 2.309

model\_performance(fit2)

# Indices of model performance  
  
AIC | AICc | BIC | R2 | R2 (adj.) | RMSE | Sigma  
------------------------------------------------------------------  
1581.884 | 1582.000 | 1597.316 | 0.377 | 0.374 | 2.292 | 2.302

* fit2 has higher , higher adjusted , lower RMSE, lower Sigma, lower values of AIC and corrected AIC.
* fit1 has a lower value of BIC.

## Compare Fit Quality (fit1 vs. fit2)

Remember compare\_performance() weights AIC and BIC so higher values indicate better fit.

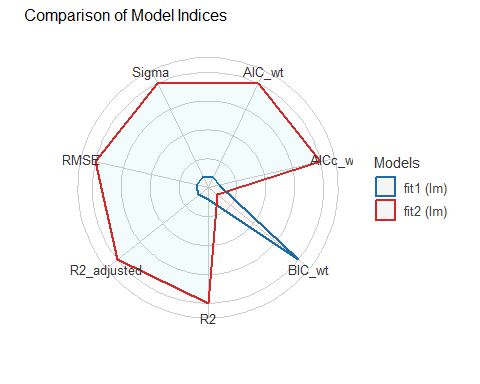
compare\_performance(fit1, fit2, rank = TRUE)

# Comparison of Model Performance Indices  
  
Name | Model | R2 | R2 (adj.) | RMSE | Sigma | AIC weights | AICc weights | BIC weights | Performance-Score  
---------------------------------------------------------------------------------------------------------------  
fit2 | lm | 0.377 | 0.374 | 2.292 | 2.302 | 0.648 | 0.643 | 0.211 | 85.71%  
fit1 | lm | 0.371 | 0.370 | 2.303 | 2.309 | 0.352 | 0.357 | 0.789 | 14.29%

* fit2 has higher , higher adjusted , lower RMSE, lower Sigma, higher (weighted) AIC and (weighted) AIC corrected
* fit1 has higher (weighted) BIC.

## fit1 vs. fit2 performance

plot(compare\_performance(fit1, fit2))



## Estimated coefficients (fit3)

model\_parameters(fit3)

Parameter | Coefficient | SE | 95% CI | t(345) | p  
-------------------------------------------------------------------------------  
(Intercept) | 19.49 | 1.07 | [17.39, 21.59] | 18.24 | < .001  
a1c old | -0.95 | 0.07 | [-1.09, -0.81] | -13.61 | < .001  
age | 0.02 | 0.01 | [ 0.00, 0.05] | 1.69 | 0.092   
income [Between\_30-50K] | 0.05 | 0.32 | [-0.58, 0.69] | 0.16 | 0.873   
income [Below\_30K] | -0.21 | 0.33 | [-0.87, 0.44] | -0.64 | 0.522

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

$$
\hat{\frac{100}{A1c}} = 19.49 - 0.95 \mbox{ A1c\_old} + 0.02 \mbox{ Age} \\ + 0.05 \mbox{(Inc 30-50)} - 0.21 \mbox{(Inc<30)}
$$

## Interpreting the slopes

$$
\hat{\frac{100}{A1c}} = 19.49 - 0.95 \mbox{ A1c\_old} + 0.02 \mbox{ Age} \\ + 0.05 \mbox{(Inc 30-50)} - 0.21 \mbox{(Inc<30)}
$$

* Suppose Harry and Sally are the same age and have the same A1c\_old, but Harry lives in a neighborhood with income < $30K, while Sally lives in a neighborhood with income > $50K. On average, our fit3 model predicts Harry’s 100/A1c value to be 0.21 lower than Sally’s.

## Summarize Fit Quality (All 3 models)

model\_performance(fit1)

# Indices of model performance  
  
AIC | AICc | BIC | R2 | R2 (adj.) | RMSE | Sigma  
------------------------------------------------------------------  
1583.106 | 1583.176 | 1594.680 | 0.371 | 0.370 | 2.303 | 2.309

model\_performance(fit2)

# Indices of model performance  
  
AIC | AICc | BIC | R2 | R2 (adj.) | RMSE | Sigma  
------------------------------------------------------------------  
1581.884 | 1582.000 | 1597.316 | 0.377 | 0.374 | 2.292 | 2.302

model\_performance(fit3)

# Indices of model performance  
  
AIC | AICc | BIC | R2 | R2 (adj.) | RMSE | Sigma  
------------------------------------------------------------------  
1584.938 | 1585.183 | 1608.086 | 0.379 | 0.372 | 2.289 | 2.306

## Compare Fit Quality (3 models)

Remember compare\_performance() weights AIC and BIC so higher values indicate better fit.

compare\_performance(fit1, fit2, fit3, rank = TRUE)

# Comparison of Model Performance Indices  
  
Name | Model | R2 | R2 (adj.) | RMSE | Sigma | AIC weights | AICc weights | BIC weights | Performance-Score  
---------------------------------------------------------------------------------------------------------------  
fit2 | lm | 0.377 | 0.374 | 2.292 | 2.302 | 0.568 | 0.568 | 0.211 | 83.06%  
fit3 | lm | 0.379 | 0.372 | 2.289 | 2.306 | 0.123 | 0.116 | 9.67e-04 | 43.26%  
fit1 | lm | 0.371 | 0.370 | 2.303 | 2.309 | 0.308 | 0.316 | 0.788 | 26.54%

## Which Model Looks Best? (1/4)

| Model |  | Adjusted | Predictors |
| --- | --- | --- | --- |
| fit1 | 0.371 | 0.370 | a1c\_old |
| fit2 | 0.377 | 0.374 | a1c\_old, age |
| fit3 | 0.379 | 0.372 | a1c\_old, age, income |

* By , the largest model (fit3) always looks best (raw is greedy)
* Adjusted penalizes for lack of parsimony. fit2 looks best.

## More Performance Indices (2/4)

| Model | RMSE | Sigma | Predictors |
| --- | --- | --- | --- |
| fit1 | 2.303 | 2.309 | a1c\_old |
| fit2 | 2.292 | 2.302 | a1c\_old, age |
| fit3 | 2.289 | 2.306 | a1c\_old, age, income |

* For and RMSE, smaller values, indicate better fits.
  + fit3 looks best by RMSE.
  + fit2 looks best by Sigma ().

## Still More Performance Indices (3/4)

Unweighted versions, from model\_performance()…

| Model | AIC | AIC\_c | BIC | Predictors |
| --- | --- | --- | --- | --- |
| fit1 | 1583.106 | 1583.176 | 1594.680 | a1c\_old |
| fit2 | 1581.884 | 1582.000 | 1597.316 | + age |
| fit3 | 1584.938 | 1585.183 | 1608.086 | + income |

* For unweighted AIC (both types) and BIC, smaller values (more negative, if relevant) indicate better fits.
  + fit2 looks best by AIC and corrected AIC.
  + fit1 looks best by BIC.

## Weighted Performance Indices (4/4)

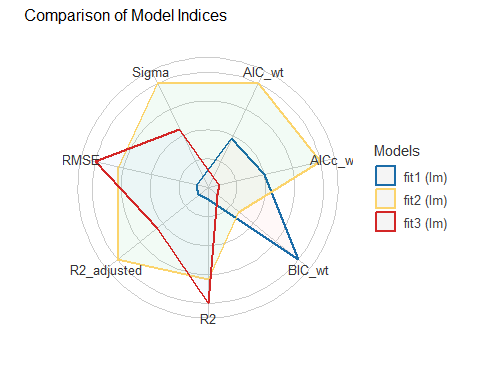
**WEIGHTED** versions, from compare\_performance()…

| Model | AIC (wtd) | AIC\_c (wtd) | BIC (wtd) | Predictors |
| --- | --- | --- | --- | --- |
| fit1 | 0.308 | 0.316 | 0.788 | a1c\_old |
| fit2 | 0.568 | 0.568 | 0.211 | + age |
| fit3 | 0.123 | 0.116 | 9.7e-04 | + income |

* After weighting of AIC (both types) and BIC, larger values indicate better fits.
  + fit2 looks best by AIC and corrected AIC.
  + fit1 looks best by BIC.

## Performance Indices for 3 Models

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



## Coming Soon

1. Checking model assumptions with check\_model() in the training sample
2. Assessing the candidate models more thoroughly, in both the training and test samples
   * MAPE, RMSPE, Maximum Prediction Error, Validated
3. Considering Bayesian alternative fits with weakly informative priors
4. Incorporating multiple imputation in building a final model

## Session Information

xfun::session\_info()

R version 4.4.1 (2024-06-14 ucrt)  
Platform: x86\_64-w64-mingw32/x64  
Running under: Windows 11 x64 (build 22631)  
  
Locale:  
 LC\_COLLATE=English\_United States.utf8   
 LC\_CTYPE=English\_United States.utf8   
 LC\_MONETARY=English\_United States.utf8  
 LC\_NUMERIC=C   
 LC\_TIME=English\_United States.utf8   
  
Package version:  
 abind\_1.4-8 askpass\_1.2.1 backports\_1.5.0   
 base64enc\_0.1.3 bayestestR\_0.15.0 bit\_4.5.0   
 bit64\_4.5.2 blob\_1.2.4 boot\_1.3-31   
 broom\_1.0.7 bslib\_0.8.0 cachem\_1.1.0   
 callr\_3.7.6 car\_3.1-3 carData\_3.0-5   
 cellranger\_1.1.0 cli\_3.6.3 clipr\_0.8.0   
 coda\_0.19-4.1 codetools\_0.2-20 colorspace\_2.1-1   
 compiler\_4.4.1 conflicted\_1.2.0 correlation\_0.8.6   
 cowplot\_1.1.3 cpp11\_0.5.0 crayon\_1.5.3   
 curl\_5.2.3 data.table\_1.16.2 datasets\_4.4.1   
 datawizard\_0.13.0 DBI\_1.2.3 dbplyr\_2.5.0   
 Deriv\_4.1.6 digest\_0.6.37 doBy\_4.6.24   
 dplyr\_1.1.4 dtplyr\_1.3.1 easystats\_0.7.3   
 effectsize\_0.8.9 emmeans\_1.10.5 estimability\_1.5.1   
 evaluate\_1.0.1 fansi\_1.0.6 farver\_2.1.2   
 fastmap\_1.2.0 fontawesome\_0.5.2 forcats\_1.0.0   
 foreach\_1.5.2 Formula\_1.2-5 fs\_1.6.4   
 gargle\_1.5.2 generics\_0.1.3 GGally\_2.2.1   
 ggplot2\_3.5.1 ggstats\_0.7.0 glmnet\_4.1-8   
 glue\_1.8.0 googledrive\_2.1.1 googlesheets4\_1.1.1   
 graphics\_4.4.1 grDevices\_4.4.1 grid\_4.4.1   
 gridExtra\_2.3 gtable\_0.3.6 haven\_2.5.4   
 highr\_0.11 hms\_1.1.3 htmltools\_0.5.8.1   
 httr\_1.4.7 ids\_1.0.1 insight\_0.20.5   
 isoband\_0.2.7 iterators\_1.0.14 janitor\_2.2.0   
 jomo\_2.7-6 jquerylib\_0.1.4 jsonlite\_1.8.9   
 knitr\_1.48 labeling\_0.4.3 lattice\_0.22-6   
 lifecycle\_1.0.4 lme4\_1.1-35.5 lubridate\_1.9.3   
 magrittr\_2.0.3 MASS\_7.3-61 Matrix\_1.7-0   
 MatrixModels\_0.5.3 memoise\_2.0.1 methods\_4.4.1   
 mgcv\_1.9.1 mice\_3.16.0 microbenchmark\_1.5.0  
 mime\_0.12 minqa\_1.2.8 mitml\_0.4-5   
 modelbased\_0.8.9 modelr\_0.1.11 multcomp\_1.4-26   
 munsell\_0.5.1 mvtnorm\_1.3-1 naniar\_1.1.0   
 nlme\_3.1-164 nloptr\_2.1.1 nnet\_7.3-19   
 norm\_1.0.11.1 numDeriv\_2016.8.1.1 openssl\_2.2.2   
 ordinal\_2023.12.4.1 pan\_1.9 parallel\_4.4.1   
 parameters\_0.23.0 patchwork\_1.3.0 pbkrtest\_0.5.3   
 performance\_0.12.4 pillar\_1.9.0 pkgconfig\_2.0.3   
 plyr\_1.8.9 prettyunits\_1.2.0 processx\_3.8.4   
 progress\_1.2.3 ps\_1.8.0 purrr\_1.0.2   
 quantreg\_5.99 R6\_2.5.1 ragg\_1.3.3   
 rappdirs\_0.3.3 RColorBrewer\_1.1-3 Rcpp\_1.0.13   
 RcppEigen\_0.3.4.0.2 readr\_2.1.5 readxl\_1.4.3   
 rematch\_2.0.0 rematch2\_2.1.2 report\_0.5.9   
 reprex\_2.1.1 rlang\_1.1.4 rmarkdown\_2.28   
 rpart\_4.1.23 rstudioapi\_0.17.1 rvest\_1.0.4   
 sandwich\_3.1-1 sass\_0.4.9 scales\_1.3.0   
 see\_0.9.0 selectr\_0.4.2 shape\_1.4.6.1   
 snakecase\_0.11.1 SparseM\_1.84.2 splines\_4.4.1   
 stats\_4.4.1 stringi\_1.8.4 stringr\_1.5.1   
 survival\_3.7-0 sys\_3.4.3 systemfonts\_1.1.0   
 textshaping\_0.4.0 TH.data\_1.1-2 tibble\_3.2.1   
 tidyr\_1.3.1 tidyselect\_1.2.1 tidyverse\_2.0.0   
 timechange\_0.3.0 tinytex\_0.53 tools\_4.4.1   
 tzdb\_0.4.0 ucminf\_1.2.2 UpSetR\_1.4.0   
 utf8\_1.2.4 utils\_4.4.1 uuid\_1.2.1   
 vctrs\_0.6.5 viridis\_0.6.5 viridisLite\_0.4.2   
 visdat\_0.6.0 vroom\_1.6.5 withr\_3.0.2   
 xfun\_0.48 xml2\_1.3.6 xtable\_1.8-4   
 yaml\_2.3.10 zoo\_1.8-12

1. Range of A1c\_old was 4.2 to 16.3 percentage points. [↑](#footnote-ref-72)
2. perhaps via a Bayesian model with an informative (not just weakly informative) prior. [↑](#footnote-ref-78)