

431 Class 24

Thomas E. Love

2019-11-21

Agenda for Today

- Linear Regression for Prediction in our dm431 data
 - Pre-Modeling Considerations
 - Consideration of Outcome Transformations
 - (Simple) Imputation to deal with Missing Data
 - Partitioning the Data (Development vs. Testing)
 - Building the Model
 - Evaluating Fit in the Development Sample
 - Considering Regression Assumptions
 - Post-Modeling Considerations
 - Evaluating Prediction Quality (Test Sample)
 - Back-Transformation of Outcome Predictions

In today's class, we will start with slide 48.

Our R Setup

```
library(simputation) # for simple imputation  
library(car) # for Box-Cox plot  
library(GGally) # for scatterplot matrix  
  
library(here); library(magrittr)  
library(patchwork); library(janitor); library(broom)  
library(tidyverse) # always load tidyverse last  
  
theme_set(theme_bw()) # now all ggplots use theme_bw()
```

Things discussed in Class 22 and 23

A change to the data!

All this time, we've had an error in the dm431 data, which I'll now call dm431_old.Rds. Can you spot it?

```
dm431_old <- readRDS(here("data", "dm431_old.Rds"))
```

```
head(dm431_old, 8)
```

```
# A tibble: 8 x 21
```

	subject	practice	age	race_eth	sex	a1c	insurance
	<chr>	<fct>	<dbl>	<fct>	<fct>	<dbl>	<fct>
1	S-001	Arlingt~	62	Black o~	F	6.3	Commerci~
2	S-002	Bristol	54	Black o~	F	11	Uninsured
3	S-003	Chester	47	Black o~	F	8.7	Uninsured
4	S-004	Dover	53	Non-His~	M	6.5	Commerci~
5	S-005	Franklin	64	Non-His~	F	6.7	Commerci~
6	S-006	Bristol	48	Black o~	F	5.8	Medicare
7	S-006	Franklin	49	Black o~	M	9.6	Commerci~
8	S-008	Dover	63	Black o~	F	6.1	Medicaid

So what exactly is the problem?

```
dm431_old %>% nrow()
```

```
[1] 431
```

```
dm431_old %$% n_distinct(subject)
```

```
[1] 430
```

```
dm431_old %>% slice(6:7) %>% select(subject, age)
```

```
# A tibble: 2 x 2
```

```
  subject    age
```

```
  <chr>    <dbl>
```

```
1 S-006      48
```

```
2 S-006      49
```

Fixing the Problem

```
dm431_fixed <- dm431_old %>%  
  mutate(subject = ifelse(subject == "S-006" & age == 49,  
    "S-007", subject))  
  
dm431_fixed %>% slice(6:7) %>% select(subject, age)  
  
# A tibble: 2 x 2  
  subject    age  
  <chr>    <dbl>  
1 S-006      48  
2 S-007      49  
  
saveRDS(dm431_fixed, file = here("data", "dm431_fixed.Rds"))
```

Focus on Four Variables (+ Subject)

```
dm431 <- readRDS(here("data", "dm431_fixed.Rds"))

dm_1 <- dm431 %>%
  select(a1c, a1c_old, age, income, subject)
```


Summarizing the dm_1 data set

```
summary(dm_1)
```

a1c	a1c_old	age
Min. : 4.300	Min. : 4.200	Min. :31.00
1st Qu.: 6.500	1st Qu.: 6.500	1st Qu.:51.00
Median : 7.300	Median : 7.300	Median :57.00
Mean : 7.884	Mean : 7.712	Mean :56.14
3rd Qu.: 8.600	3rd Qu.: 8.400	3rd Qu.:62.00
Max. :16.700	Max. :16.300	Max. :70.00
NA's :3	NA's :14	

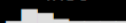


income	subject
Below_30K :146	Length:431
Between_30-50K :171	Class :character
Higher_than_50K:110	Mode :character
NA's : 4	

dm_1 %>% skimr::skim() results

```
> dm_1 %>% skimr::skim()
Skim summary statistics
n obs: 431
n variables: 5

-- Variable type:character -----
variable missing complete   n min max empty n_unique
subject      0      431 431   5   5     0      431

-- Variable type:factor -----
variable missing complete   n n_unique      top_counts ordered
income        4      427 431          3 Bet: 171, Bel: 146, Hig: 110, NA: 4  FALSE

-- Variable type:numeric -----
variable missing complete   n  mean  sd   p0  p25  p50  p75 p100  hist
alc           3      428 431  7.88 2.03  4.3  6.5  7.3  8.6 16.7  
alc_old       14      417 431  7.71 1.77  4.2  6.5  7.3  8.4 16.3  
age           0      431 431 56.14 8.41 31  51  57  62  70  
```

What roles will these variables play?

a1c is our outcome, which we'll predict with ...

- 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

What will we do about missing data?

```
dm_1 %>% summarise_all(~ sum(is.na(.)))
```

```
# A tibble: 1 x 5
```

	a1c	a1c_old	age	income	subject
	<int>	<int>	<int>	<int>	<int>
1	3	14	0	4	0

- We're missing 3 values of a1c, our outcome
- and 14 values of a1c_old, a predictor (Models 1-3)
- and 4 values of income, another predictor (Model 3)

Dealing with outcome missingness

I don't want to impute the outcome. We'll drop the 3 observations missing `a1c` from our data set.

```
dm_2 <- dm_1 %>% filter(complete.cases(a1c))  
dm_2 %>% summarise_all(~ sum(is.na(.)))
```

A tibble: 1 x 5

	a1c	a1c_old	age	income	subject
	<int>	<int>	<int>	<int>	<int>
1	0	12	0	4	0

How should we deal with the remaining missing values?

Simple Imputation of Missing a1c_old Values

We could use a robust linear model method to impute our quantitative a1c_old values on the basis of age, which is missing no observations in common with a1c_old (in fact, age is missing no observations.)

```
dm_3a <- impute_rlm(dm_2, a1c_old ~ age)
```

```
dm_3a %>% select(a1c_old, income) %>% summary()
```

a1c_old	income
Min. : 4.200	Below_30K :146
1st Qu.: 6.500	Between_30-50K :169
Median : 7.300	Higher_than_50K:109
Mean : 7.711	NA's : 4
3rd Qu.: 8.400	
Max. :16.300	

Simple Imputation of Missing income Values

We could use a decision tree (CART) method to impute our missing categorical income values, on the basis of age.

```
dm_3b <- impute_cart(dm_2, income ~ age)
```

```
dm_3b %>% select(a1c_old, income) %>% summary()
```

a1c_old	income
Min. : 4.200	Below_30K :148
1st Qu.: 6.500	Between_30-50K :171
Median : 7.300	Higher_than_50K:109
Mean : 7.716	
3rd Qu.: 8.400	
Max. :16.300	
NA's :12	

Chaining our Simple Imputations

Or we could put all of our imputations together in a chain. I encourage you to try `rlm` for quantitative variables, and `cart` for categorical variables, for now.

```
dm_4 <- dm_2 %>%  
  impute_rlm(a1c_old ~ age) %>%  
  impute_cart(income ~ age + a1c_old)  
  
dm_4 %>% select(a1c, a1c_old, income) %>%  
  summarise_all(~(sum(is.na(.))))
```

```
# A tibble: 1 x 3  
  a1c a1c_old income  
  <int>   <int>   <int>  
1     0       0     0
```

What did we do? What is the result?

dm_4 %>% skimr::skim() results

```
> dm_4 %>% skimr::skim()
Skim summary statistics
n obs: 428
n variables: 5

-- Variable type:character -----
variable missing complete  n min max empty n_unique
subject      0      428 428  5  5    0      428

-- Variable type:factor -----
variable missing complete  n n_unique top_counts ordered
income      0      428 428      3 Bet: 171, Bel: 148, Hig: 109, NA: 0 FALSE

-- Variable type:numeric -----
variable missing complete  n mean  sd  p0  p25  p50  p75 p100 hist
a1c          0      428 428  7.88 2.03  4.3  6.5  7.3  8.6 16.7
a1c_old      0      428 428  7.71 1.75  4.2  6.5  7.3  8.4 16.3
age          0      428 428 56.09 8.42 31  50.75 57  62  70
```

OK. Ready to proceed?

How will we decide which of the models is “best”?

Our goal is accurate prediction of a1c values.

Which of these models gives us the “best” result?

- 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”

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.

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.

431 approach: Which model is “most useful”?

- 1 Split the data into a model development (training) sample of about 70-80% of the observations, and a model test (holdout) 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.
- 4 Check adherence to regression assumptions in the development sample.
- 5 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.)
- 6 Select a “final” model for use based on the evidence in steps 3, 4 and especially 5.

Split the data into a model development (training) sample of about 70-80% of the observations, and a model test (holdout) sample, containing the remaining observations.

Partition the imputed data into development/test samples

```
set.seed(20191114)

dm4_dev <- sample_frac(dm_4, 0.75, replace = FALSE)

dm4_test <- anti_join(dm_4, dm4_dev, by = "subject")
```

```
nrow(dm_4); nrow(dm4_dev); nrow(dm4_test)
```

```
[1] 428
```

```
[1] 321
```

```
[1] 107
```

Develop candidate models using the development sample.

A look at the outcome (a1c) distribution

We'll study the outcome variable (a1c) in the development sample, to consider whether a transformation might be in order.

I did a little fancy work with the code (continues next slide)...

```
p1 <- ggplot(dm4_dev, aes(x = a1c)) +  
  geom_histogram(binwidth = 0.5,  
                 fill = "slateblue", col = "white")  
  
p2 <- ggplot(dm4_dev, aes(sample = a1c)) +  
  geom_qq(col = "slateblue") + geom_qq_line(col = "red")  
  
p3 <- ggplot(dm4_dev, aes(x = "", y = a1c)) +  
  geom_violin(fill = "slateblue", alpha = 0.3) +  
  geom_boxplot(fill = "slateblue", width = 0.3,  
               outlier.color = "red") +  
  labs(x = "") + coord_flip()
```


A look at the outcome (a1c) distribution

Putting the plots together, and titling them meaningfully...

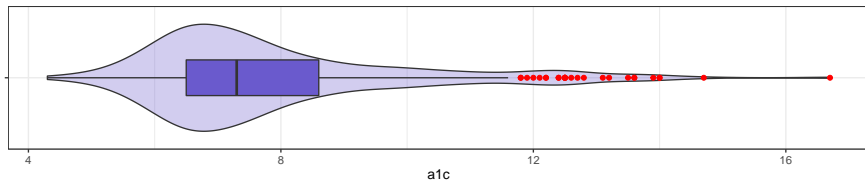
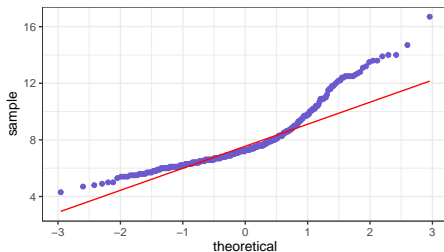
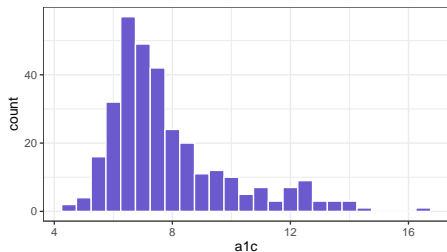
```
p1 + p2 - p3 +  
  plot_layout(ncol = 1, height = c(3, 2)) +  
  plot_annotation(title = "Hemoglobin A1c values (%)",  
                  subtitle = paste0("Model Development Sample: ",  
                                     nrow(dm4_dev),  
                                     " adults with diabetes"))
```

Result on the next slide...

Outcome (a1c): Model Development Sample

Hemoglobin A1c values (%)

Model Development Sample: 321 adults with diabetes



Why 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.

Ladder of Especially Useful (and often interpretable) transformations

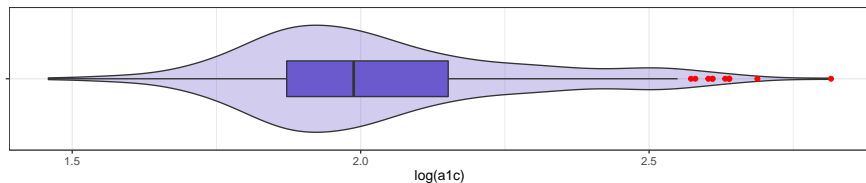
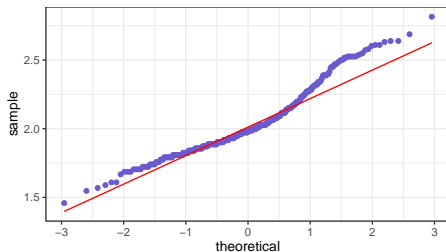
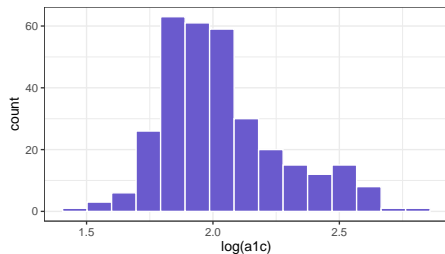
Transformation	y^2	y	\sqrt{y}	$\log(y)$	$1/y$	$1/y^2$
λ	2	1	0.5	0	-1	-2

- We see some sign of right skew in the a1c data. Let's try a log transformation.

Consider a log transformation?

Natural Logarithm of Hemoglobin A1c

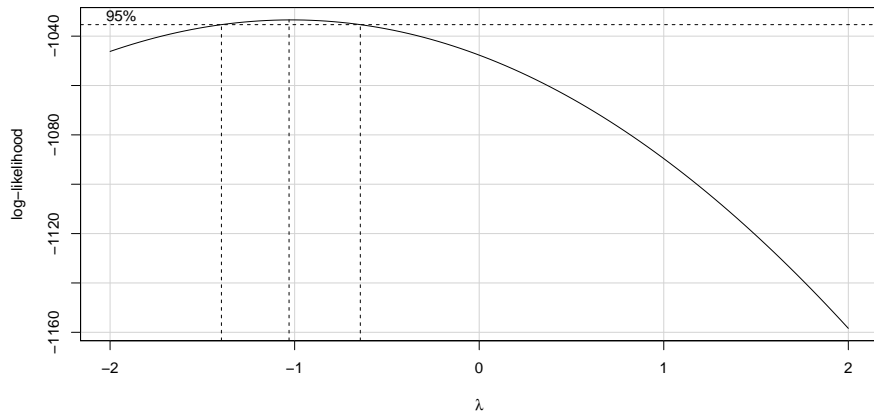
Model Development Sample: 321 adults with diabetes



Using Box-Cox to help select a transformation?

```
mod_0 <- lm(a1c ~ a1c_old + age + income, data = dm4_dev)
```

```
boxCox(mod_0)
```



Using Box-Cox to help select a transformation?

```
summary(powerTransform(mod_0))
```

bcPower Transformation to Normality

	Est Power	Rounded Pwr	Wald Lwr Bnd	Wald Upwr Bnd
Y1	-1.019	-1	-1.3953	-0.6427

Likelihood ratio test that transformation parameter is equal to 0
(log transformation)

	LRT	df	pval
LR test, lambda = (0)	28.53961	1	9.1801e-08

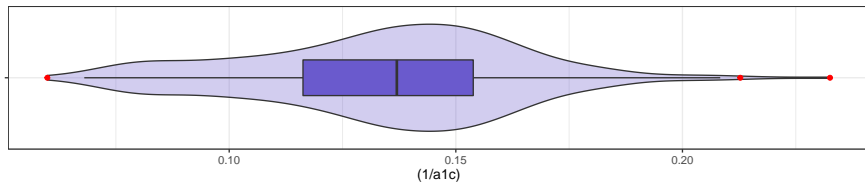
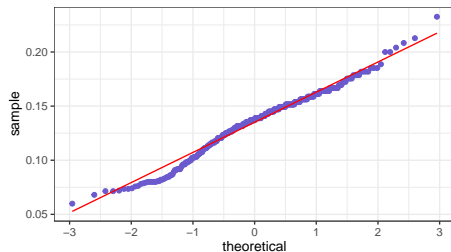
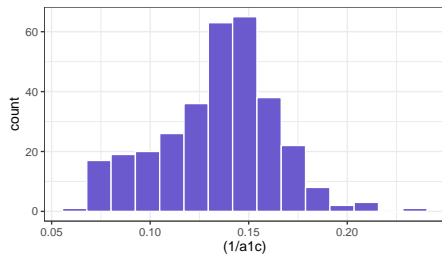
Likelihood ratio test that no transformation is needed

	LRT	df	pval
LR test, lambda = (1)	112.3945	1	< 2.22e-16

Consider the inverse?

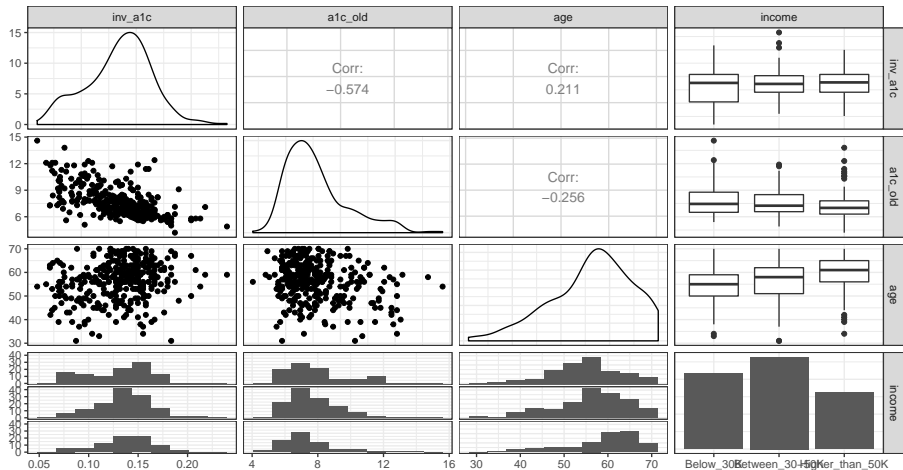
Inverse of Hemoglobin A1c

Model Development Sample: 321 adults with diabetes



Scatterplot Matrix

Scatterplots: Model Development Sample



Scatterplot Matrix (Code)

```
dm4_dev %>%  
  mutate(inv_a1c = 1/a1c) %>%  
  select(inv_a1c, a1c_old, age, income) %>%  
  ggpairs(., title = "Scatterplot Matrix for Model Development",  
    lower = list(combo = wrap("facethist", bins = 10)))
```

Note that ggpairs comes from the GGally package.

Three Regression Models We'll Fit

Remember we're using the model development sample here.

```
mod_1 <- lm((1/a1c) ~ a1c_old, data = dm4_dev)
```

```
mod_2 <- lm((1/a1c) ~ a1c_old + age, data = dm4_dev)
```

```
mod_3 <- lm((1/a1c) ~ a1c_old + age + income,  
            data = dm4_dev)
```

**Assess the quality of fit for candidate models
within the development sample.**

summary(mod_1) (edited to fit on screen)

```
Call:
lm(formula = (1/a1c) ~ a1c_old, data = dm4_dev)

Residuals:
    Min       1Q   Median       3Q      Max
-0.068553 -0.014349  0.000183  0.013117  0.078923

Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept)  0.2099494   0.0061836   33.95  <2e-16 ***
a1c_old      -0.0098553   0.0007868  -12.53  <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.02409 on 319 degrees of freedom
Multiple R-squared:  0.3297,    Adjusted R-squared:  0.3276
F-statistic: 156.9 on 1 and 319 DF,  p-value: < 2.2e-16
```

Summary of Fit Quality (mod_1)

```
g1 <- glance(mod_1) %>%  
  mutate(name = "mod_1") %>%  
  select(name, r.squared, adj.r.squared,  
         sigma, AIC, BIC) %>%  
  knitr::kable(digits = c(0, 3, 3, 3, 0, 0))
```

g1

name	r.squared	adj.r.squared	sigma	AIC	BIC
mod_1	0.33	0.328	0.024	-1477	-1466

Tidied coefficients (mod_1)

```
tidy(mod_1, conf.int = TRUE, conf.level = 0.95) %>%  
  select(term, estimate, std.error, p.value,  
         conf.low, conf.high) %>%  
  knitr::kable(digits = c(0, 4, 4, 4, 4, 4))
```

term	estimate	std.error	p.value	conf.low	conf.high
(Intercept)	0.2099	0.0062	0	0.1978	0.2221
a1c_old	-0.0099	0.0008	0	-0.0114	-0.0083

summary(mod_2) (edited to fit on screen)

```
Call:
lm(formula = (1/alc) ~ alc_old + age, data = dm4_dev)

Residuals:
    Min       1Q   Median       3Q      Max
-0.068387 -0.013588  0.000058  0.013243  0.076861

Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept)  0.1939204  0.0126114  15.377  <2e-16 ***
alc_old      -0.0095521  0.0008125 -11.756  <2e-16 ***
age           0.0002429  0.0001667   1.458   0.146
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.02405 on 318 degrees of freedom
Multiple R-squared:  0.3341,    Adjusted R-squared:  0.3299
F-statistic: 79.78 on 2 and 318 DF,  p-value: < 2.2e-16
```

Summary of Fit Quality (mod_2)

```
g2 <- glance(mod_2) %>%  
  mutate(name = "mod_2") %>%  
  select(name, r.squared, adj.r.squared,  
         sigma, AIC, BIC) %>%  
  knitr::kable(digits = c(0, 3, 3, 3, 0, 0))
```

g2

name	r.squared	adj.r.squared	sigma	AIC	BIC
mod_2	0.334	0.33	0.024	-1477	-1462

Tidied coefficients (mod_2)

```
tidy(mod_2, conf.int = TRUE, conf.level = 0.95) %>%  
  select(term, estimate, std.error, p.value,  
         conf.low, conf.high) %>%  
  knitr::kable(digits = c(0, 4, 4, 4, 4, 4))
```

term	estimate	std.error	p.value	conf.low	conf.high
(Intercept)	0.1939	0.0126	0.000	0.1691	0.2187
a1c_old	-0.0096	0.0008	0.000	-0.0112	-0.0080
age	0.0002	0.0002	0.146	-0.0001	0.0006

summary(mod_3) (edited to fit on screen)

```
Call:
lm(formula = (1/alc) ~ alc_old + age + income, data = dm4_dev)

Residuals:
    Min       1Q   Median       3Q      Max
-0.067448 -0.013845  0.000413  0.012895  0.077722

Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept)    0.1924566   0.0127526   15.092  <2e-16 ***
alc_old        -0.0095322   0.0008184  -11.647  <2e-16 ***
age             0.0002489   0.0001692    1.471   0.142
incomeBetween_30-50K 0.0024669  0.0031434    0.785   0.433
incomeHigher_than_50K -0.0001717  0.0036244   -0.047   0.962
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.02409 on 316 degrees of freedom
Multiple R-squared:  0.3359,    Adjusted R-squared:  0.3275
F-statistic: 39.96 on 4 and 316 DF,  p-value: < 2.2e-16
```

Summary of Fit Quality (mod_3)

```
g3 <- glance(mod_3) %>%  
  mutate(name = "mod_3") %>%  
  select(name, r.squared, adj.r.squared,  
         sigma, AIC, BIC) %>%  
  knitr::kable(digits = c(0, 3, 3, 3, 0, 0))
```

g3

name	r.squared	adj.r.squared	sigma	AIC	BIC
mod_3	0.336	0.328	0.024	-1474	-1451

Tidied coefficients (mod_3)

```
tidy(mod_3, conf.int = TRUE, conf.level = 0.95) %>%  
  select(term, est = estimate, se = std.error, p = p.value,  
         low95 = conf.low, high95 = conf.high) %>%  
  knitr::kable(digits = c(0, 4, 4, 3, 4, 4))
```

term	est	se	p	low95	high95
(Intercept)	0.1925	0.0128	0.000	0.1674	0.2175
a1c_old	-0.0095	0.0008	0.000	-0.0111	-0.0079
age	0.0002	0.0002	0.142	-0.0001	0.0006
incomeBetween_30-50K	0.0025	0.0031	0.433	-0.0037	0.0087
incomeHigher_than_50K	-0.0002	0.0036	0.962	-0.0073	0.0070

Could we have fit other predictor sets?

Perhaps an automated procedure like stepwise regression would suggest a better alternative?

- Three predictor candidates, so we could have used any of these predictor sets:
- a1c_old alone (our mod_1)
- age alone
- income alone
- a1c_old and age (our mod_2)
- a1c_old and income
- age and income
- a1c_old, age and income (our mod_3)

```
step(mod_3)
```

Stepwise Regression Results?

Start: AIC=-2387.04

(1/a1c) ~ a1c_old + age + income

	Df	Sum of Sq	RSS	AIC
- income	2	0.000502	0.18392	-2390.2
<none>			0.18342	-2387.0
- age	1	0.001256	0.18468	-2386.8
- a1c_old	1	0.078742	0.26216	-2274.4

Step: AIC=-2390.16

(1/a1c) ~ a1c_old + age

	Df	Sum of Sq	RSS	AIC
<none>			0.18392	-2390.2
- age	1	0.001229	0.18515	-2390.0
- a1c_old	1	0.079935	0.26386	-2276.3

Call:

lm(formula = (1/a1c) ~ a1c_old + age, data = dm4_dev)

Coefficients:

(Intercept)	a1c_old	age
0.1939204	-0.0095521	0.0002429

Comparing Summary Measures of Fit

in the development sample...

```
bind_rows(glance(mod_1), glance(mod_2), glance(mod_3)) %>%  
  mutate(name = c("mod_1", "mod_2", "mod_3")) %>%  
  select(name, r2 = r.squared, adj_r2 = adj.r.squared,  
         sigma, AIC, BIC, df, df_resid = df.residual) %>%  
  knitr::kable(digits = c(0, 4, 4, 4, 1, 0, 0, 0))
```

name	r2	adj_r2	sigma	AIC	BIC	df	df_resid
mod_1	0.3297	0.3276	0.0241	-1477.1	-1466	2	319
mod_2	0.3341	0.3299	0.0240	-1477.2	-1462	3	318
mod_3	0.3359	0.3275	0.0241	-1474.1	-1451	5	316

OK. What do we think?

Check adherence to regression assumptions in the development sample.

Checking Regression Assumptions

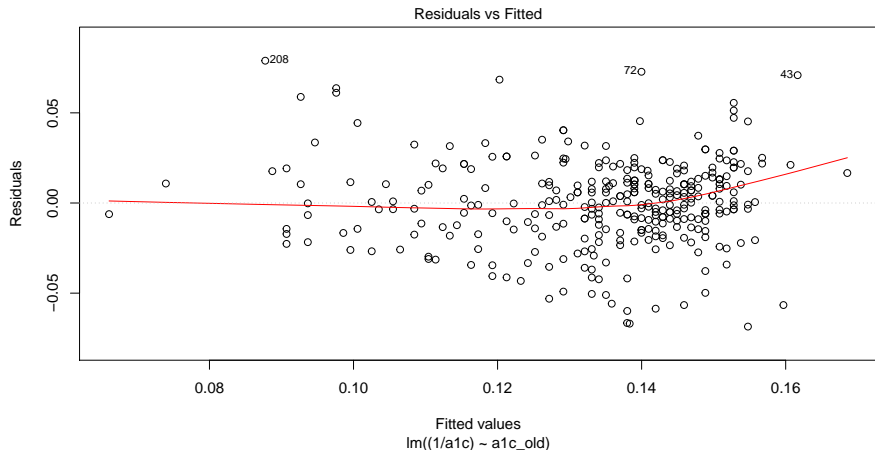
Four key assumptions we need to think about:

- 1 Linearity
- 2 Constant Variance (Homoscedasticity)
- 3 Normality
- 4 Independence

How do we assess 1, 2, and 3? Residual plots.

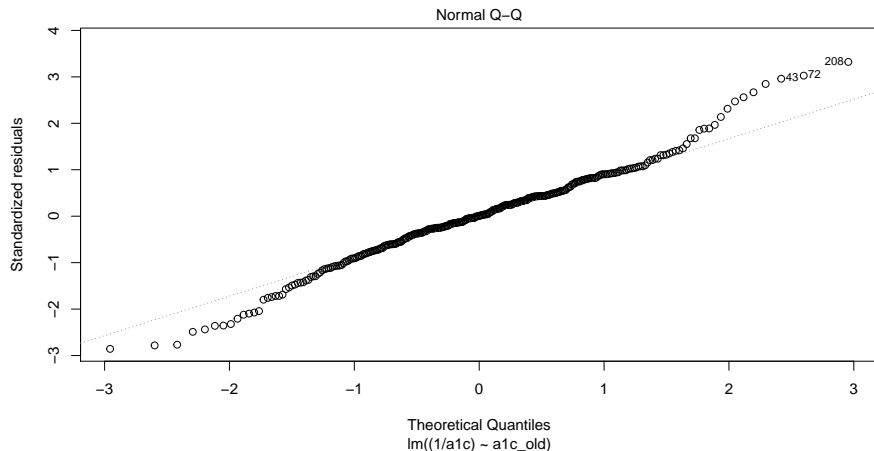
Residuals vs. Fitted Values Plot (Model mod_1)

```
plot(mod_1, which = 1)
```



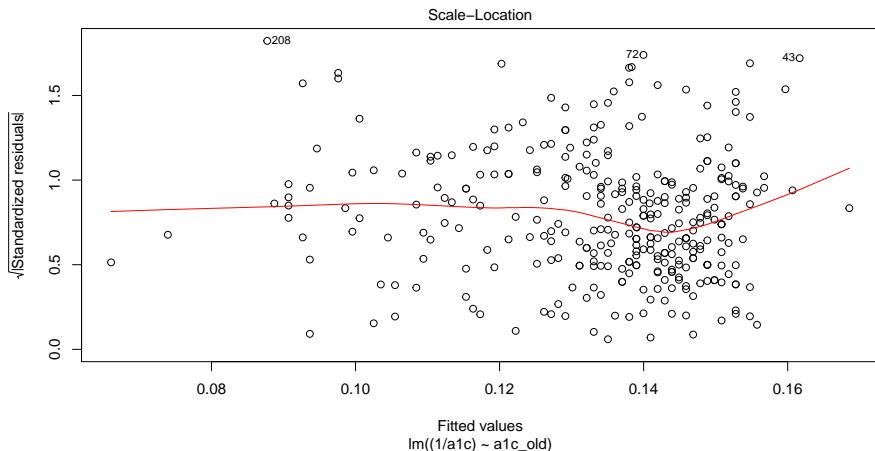
Normal Q-Q of Standardized Residuals (mod_1)

```
plot(mod_1, which = 2)
```



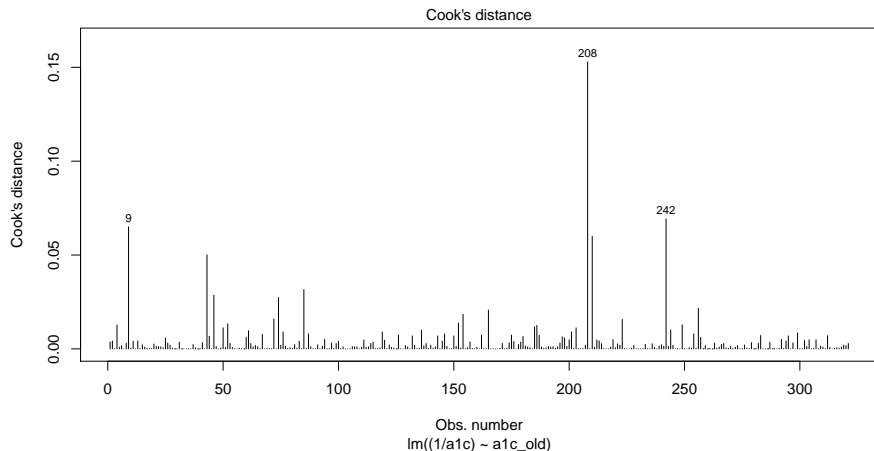
Scale-Location: Non-constant variance check (mod_1)

```
plot(mod_1, which = 3)
```



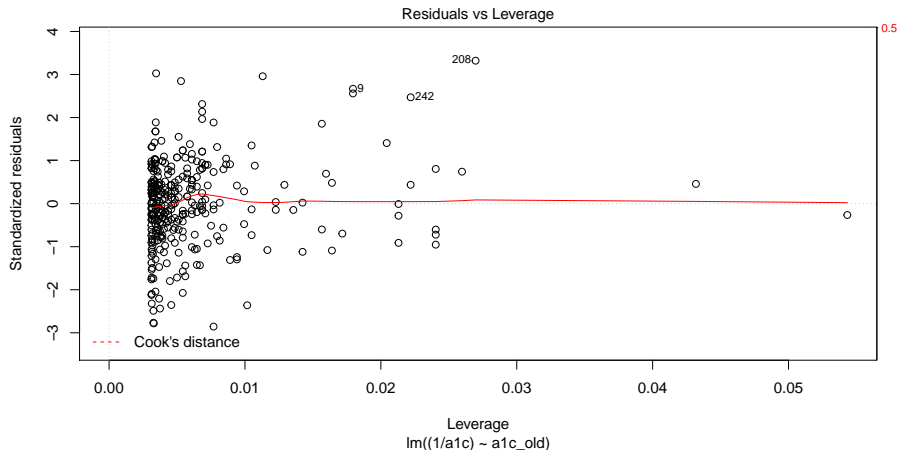
Index plot of Cook's distance for influence (mod_1)

```
plot(mod_1, which = 4)
```



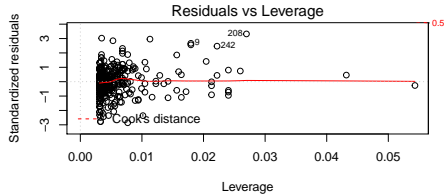
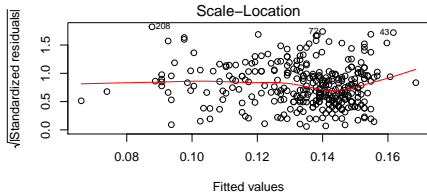
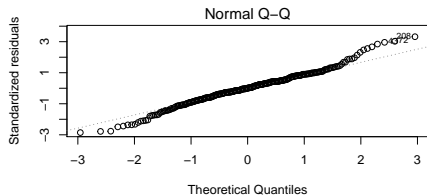
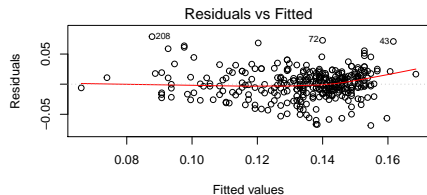
Residuals, Leverage and Influence plot (mod_1)

```
plot(mod_1, which = 5)
```

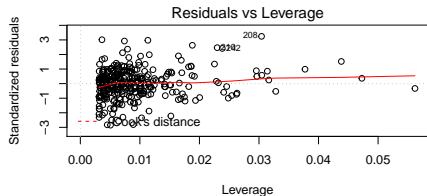
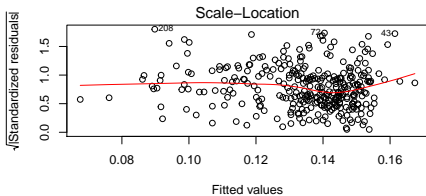
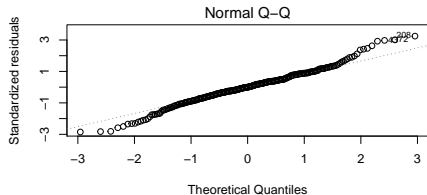
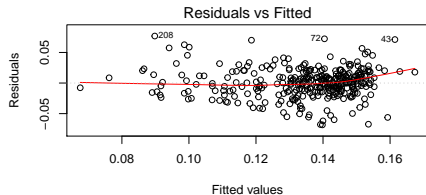


Residual Plots for Model mod_1

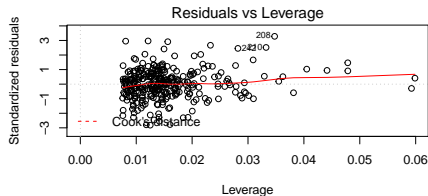
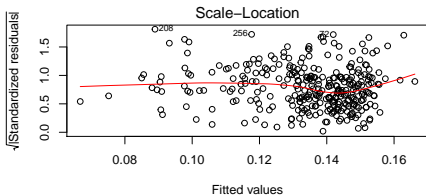
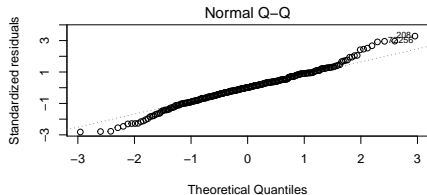
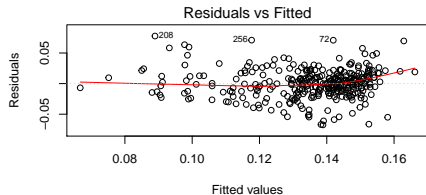
```
par(mfrow = c(2,2)); plot(mod_1); par(mfrow = c(1,1))
```



Residual Plots for Model `mod_2`



Residual Plots for Model `mod_3`



Conclusions so far?

- ① In-sample model predictions are about equally accurate for each of the three models. It's not clear yet that we need anything more than the simple regression on `a1c_old`.
- ② Residual plots look similarly reasonable for linearity, Normality and constant variance in all three models.
- ③ Probably worth considering all three models further, but it would depend on the context.

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.)

Calculate prediction errors for `mod_1` in test sample

The `augment` function in the `broom` package will create predictions within our new sample, but we want to back-transform these predictions so that they are on the original scale (`a1c`, rather than our transformed regression outcome $1/a1c$). Since the way to back out of the inverse transformation is to take the inverse again, we will take the inverse of the fitted values provided by `augment` and then calculate residuals on the original scale, as follows...

```
test_m1 <- augment(mod_1, newdata = dm4_test) %>%  
  mutate(name = "mod_1", fit_a1c = 1 / .fitted,  
         res_a1c = a1c - fit_a1c)
```

What does test_m1 now include?

```
test_m1 %>%  
  select(subject, a1c, fit_a1c, res_a1c, a1c_old,  
         age, income) %>%  
  head() %>%  
  knitr::kable(digits = c(0, 1, 2, 2, 1, 0, 0))
```

subject	a1c	fit_a1c	res_a1c	a1c_old	age	income
S-002	11.0	20.28	-9.28	16.3	54	Between_30-50K
S-005	6.7	6.76	-0.06	6.3	64	Between_30-50K
S-006	5.8	6.85	-1.05	6.5	48	Below_30K
S-009	12.9	7.46	5.44	7.7	55	Below_30K
S-013	8.1	6.95	1.15	6.7	55	Higher_than_50K
S-016	8.4	7.46	0.94	7.7	44	Between_30-50K

Gather test-sample prediction errors for models 2, 3

```
test_m2 <- augment(mod_2, newdata = dm4_test) %>%  
  mutate(name = "mod_2", fit_a1c = 1 / .fitted,  
         res_a1c = a1c - fit_a1c)  
  
test_m3 <- augment(mod_3, newdata = dm4_test) %>%  
  mutate(name = "mod_3", fit_a1c = 1 / .fitted,  
         res_a1c = a1c - fit_a1c)
```

Combine test sample results from the three models

```
test_comp <- bind_rows(test_m1, test_m2, test_m3) %>%  
  arrange(subject, name)  
  
test_comp %>% select(name, subject, a1c, fit_a1c, res_a1c,  
                    a1c_old, age, income) %>%  
  slice(1:3, 7:9) %>%  
  knitr::kable(digits = c(0, 0, 1, 2, 2, 1, 0, 0))
```

name	subject	a1c	fit_a1c	res_a1c	a1c_old	age	income
mod_1	S-002	11.0	20.28	-9.28	16.3	54	Between_30-5
mod_2	S-002	11.0	19.48	-8.48	16.3	54	Between_30-5
mod_3	S-002	11.0	18.87	-7.87	16.3	54	Between_30-5
mod_1	S-006	5.8	6.85	-1.05	6.5	48	Below_30K
mod_2	S-006	5.8	6.97	-1.17	6.5	48	Below_30K
mod_3	S-006	5.8	7.02	-1.22	6.5	48	Below_30K

What do we do to compare the test-sample errors?

Given this tibble, including predictions and residuals from the three models on our test data, we can now:

- 1 Visualize the prediction errors from each model.
- 2 Summarize those errors across each model.
- 3 Identify the “worst fitting” subject for each model in the test sample.

Visualize the prediction errors

```
ggplot(test_comp, aes(x = res_a1c, fill = name)) +  
  geom_histogram(bins = 20, col = "white") +  
  facet_grid (name ~ .) + guides(fill = FALSE)
```

or maybe

```
ggplot(test_comp, aes(x = name, y = res_a1c, fill = name)) +  
  geom_violin(alpha = 0.3) +  
  geom_boxplot(width = 0.3, outlier.shape = NA) +  
  geom_jitter(height = 0, width = 0.1) +  
  guides(fill = FALSE)
```

Test-Sample Prediction Errors

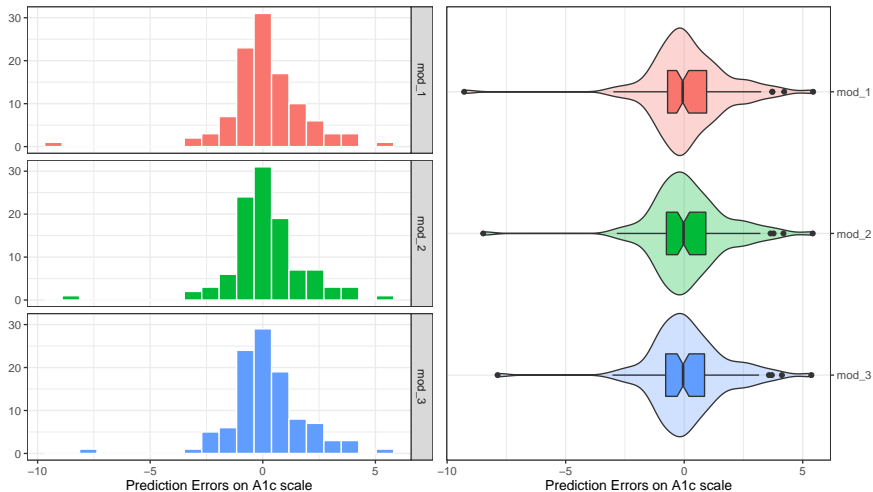


Table Comparing Model Prediction Errors

Calculate the mean absolute prediction error (MAPE), the mean squared prediction error (MSPE) and the maximum absolute error across the predictions made by each model.

```
test_comp %>%  
  group_by(name) %>%  
  summarize(n = n(),  
            MAPE = mean(abs(res_a1c)),  
            MSPE = mean(res_a1c^2),  
            max_error = max(abs(res_a1c)))
```

```
# A tibble: 3 x 5  
  name      n  MAPE  MSPE max_error  
  <chr> <int> <dbl> <dbl>      <dbl>  
1 mod_1  107  1.15  2.99      9.28  
2 mod_2  107  1.14  2.83      8.48  
3 mod_3  107  1.13  2.71      7.87
```

Identify the largest errors

Identify the subject(s) where that maximum prediction error was made by each model, and the observed and model-fitted values of a1c in each case.

```
temp1 <- test_m1 %>%  
  filter(abs(res_a1c) == max(abs(res_a1c)))  
  
temp2 <- test_m2 %>%  
  filter(abs(res_a1c) == max(abs(res_a1c)))  
  
temp3 <- test_m3 %>%  
  filter(abs(res_a1c) == max(abs(res_a1c)))
```

Identify the largest errors (Results)

Identify the subject(s) where that maximum prediction error was made by each model, and the observed and model-fitted values of a1c in each case.

```
bind_rows(temp1, temp2, temp3) %>%  
  select(subject, a1c, fit_a1c, res_a1c)
```

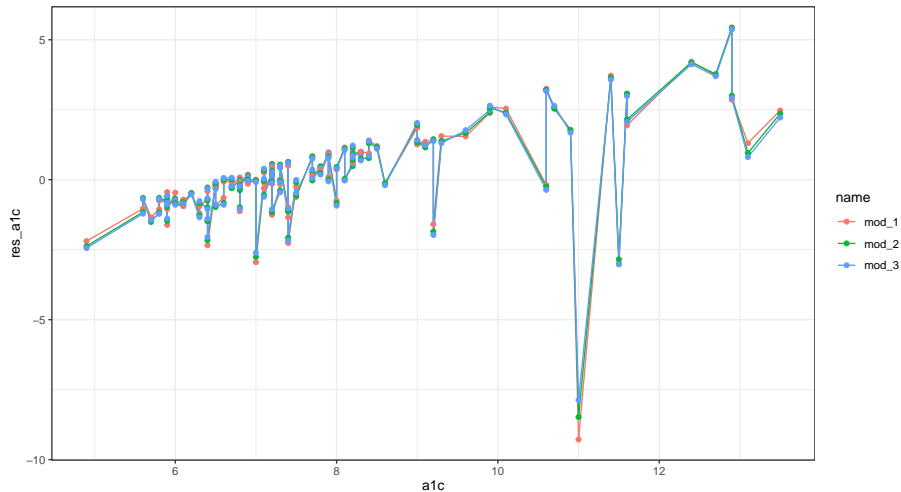
```
# A tibble: 3 x 4  
  subject    a1c fit_a1c res_a1c  
  <chr>    <dbl>   <dbl>   <dbl>  
1 S-002      11    20.3    -9.28  
2 S-002      11    19.5    -8.48  
3 S-002      11    18.9    -7.87
```

Line Plot of the Errors?

Compare the errors that are made at each level of observed A1c?

```
ggplot(test_comp, aes(x = a1c, y = res_a1c,  
                      group = name)) +  
  geom_line(aes(col = name)) +  
  geom_point(aes(col = name))
```

Line Plot of the Errors?



What if we ignored S-002 for a moment?

All three miss this subject substantially, but without S-002, we have:

```
test_comp %>% filter(subject != "S-002") %>%  
  group_by(name) %>%  
  summarize(n = n(),  
            MAPE = mean(abs(res_a1c)),  
            MSPE = mean(res_a1c^2),  
            max_error = max(abs(res_a1c)))
```

```
# A tibble: 3 x 5  
  name      n  MAPE  MSPE max_error  
  <chr> <int> <dbl> <dbl>      <dbl>  
1 mod_1   106  1.07  2.21      5.44  
2 mod_2   106  1.07  2.18      5.42  
3 mod_3   106  1.07  2.16      5.37
```

With the exception of subject S-002, the three models seem to make very similar errors in the test sample.

Conclusions now?

- 1 In-sample model predictions are about equally accurate for each of the three models. It's not clear yet that we need anything more than the simple regression on `a1c_old`. The addition of the other two predictors doesn't add predictive value that is statistically detectable.
- 2 Residual plots look similarly reasonable for linearity, Normality and constant variance in all three models.
- 3 Excluding a bad miss on one subject in the test sample, all three models do about equally well, with perhaps `mod_3` very slightly better according to all three metrics (MAPE, MSPE and `max_error`) in the test sample.

So, what should our “most useful” model be?

Repeating our 431 Strategy

Which model is “most useful” in a prediction context?

- ➊ Split the data into a model development (training) sample of about 70-80% of the observations, and a model test (holdout) sample, containing the remaining observations.
- ➋ Develop candidate models using the development sample.
- ➌ Assess the quality of fit for candidate models within the development sample.
- ➍ Check adherence to regression assumptions in the development sample.
- ➎ 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.)
- ➏ Select a “final” model for use based on the evidence in steps 3, 4 and especially 5.