Hints and Help for Carrying out Homework 4, which remains due to Canvas on 2020-03-23

432 Spring 2020 - From Dr. Love – Sent 2020-03-16

"The right answer alone isn't really a good indication of understanding."

Here's our setup and package loading section of Homework 4.

```
knitr::opts_chunk$set(comment = NA)

library(here); library(janitor); library(magrittr)

library(mice)
library(naniar)
library(caret)
library(simputation)
library(car)
library(rms)
library(broom)
library(knitr)

library(tidyverse)
```

```
phr <- read_csv(here("data", "phr.csv")) %>%
    clean_names()
```

In **Question 1**, you'll know you're on the right track if your tibble changes from having 20 variables and 10,746 rows (in the initial phr tibble) to having 9 variables and 10,743 rows (in the revised phr1 tibble.)

In **Question 2**, the key line of code after setting the seed would be something like

```
training.samples <- phr1$sbp %>% createDataPartition(p = 0.75, list = FALSE)
```

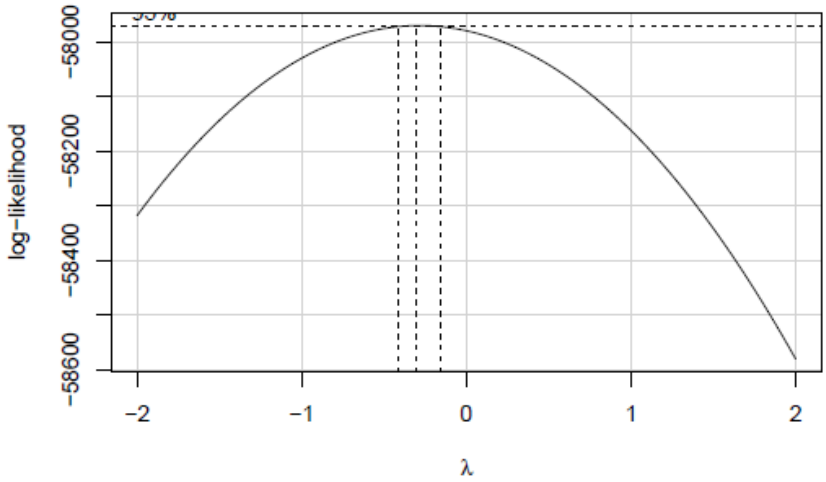
In **Question 3**, my result shows a tibble with 7 rows (variables) and 3 columns (labeled variable, n.miss and pct.miss) which I obtained using a function from the naniar package after selecting the seven variables of interest. You should find that the `ldl`, `bmi` and `hba1c` variables are missing data.

In **Question 4**, the key line of code (no reason to set a seed here, as it turns out) would be something like:

```
phr1_train_imp <- phr1_train %>%
         impute_rlm(., ldl + bmi + hba1c ~ age + female + caucasian + insurance)
```

You should then use a function from the naniar package to verify that the new **phr1_train_imp** tibble contains no missing values. Mine has 8058 rows.

In **Question 5**, my Box-Cox plot looks like this…



With the maximum value on the curve associated with a lambda value far closer to 0 than to -1, the choice of transformation (among the power transformations that we actually understand – like the square, square root, natural logarithm or inverse) should be straightforward.

In **Question 6**, I fit the following model:

mod_main <- phr1_train %$% lm(log(sbp) ~ hba1c + bmi + ldl + age + female + caucasian + insurance)

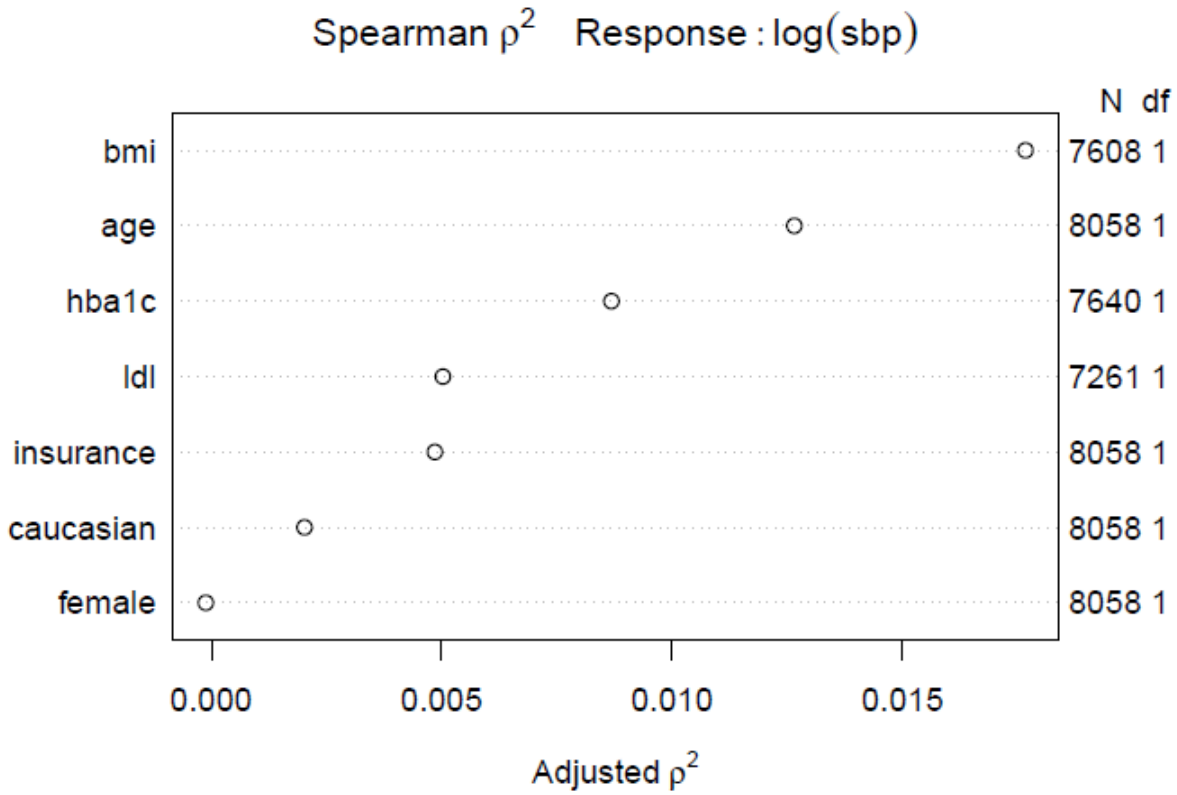
from which I obtained the following nicely cleaned up table…

model	r.squared	adj.r.squared	AIC	BIC	df
Main Effects	0.0517	0.0508	-9966	-9904.6	8

To get that clean table, I used the glance function from the broom package, which I then piped into a mutate function to establish the model's name as "Main Effects" for the leftmost column in the table, and then I piped that result into a select function to choose the columns of the glance output that I wanted in appropriate order, and finally I piped that into the kable function, for which I used

```
kable(digits = c(0, 4, 4, 1, 1, 0))
```

In **Question 7**, I decided to use the training data without any imputation to consider the relationship between log(sbp) and the predictors at hand. My plot looks like this…



I spent the additional 5 degrees of freedom using a pair of restricted cubic splines, with 5 knots for the quantitative variable with the highest potential predictive punch, and 4 knots for the quantitative variable with the second highest potential predictive punch according to my Spearman rho-squared plot.

- A 5-knot spline uses 4 degrees of freedom (which is 3 more than the 1 df used by the main effects model to represent the same predictor.)
- A 4-knot spline uses 3 degrees of freedom (which is 2 more than the 1 df used by the main effects model to represent the same predictor.)
- Hence, the addition of these two splines uses exactly 5 more degrees of freedom than the main effects model.

In **Question 8**, the first key bit of code fits the augmented model. I used:

```
mod_aug <- phr1_train %$%
          lm(log(sbp) ~ hba1c + rcs(bmi,5) + ldl + rcs(age, 4) + female +
                        caucasian + insurance)
```

I then adjusted the method used in Question 6 to produce the table of summary statistics…

model	r.squared	adj.r.squared	AIC	BIC	df
Augmented	0.0548	0.0531	-9977.8	-9882.3	13

Then I compared the results for each of the columns across the two tables.

I then used the anova function to compare mod_main to mod_aug, in part to look to see whether the augmented model appeared to provide a statistically detectable improvement over the main effects model, and (I suppose) to verify that the difference between the two models did in fact use exactly 5 degrees of freedom.

In **Question 9**, I first dropped all observations with missing values in the test sample, using
`phr1_test_cc <- phr1_test %>% drop_na()`

I encourage you to determine how many observations get dropped as a result of this move from phr1_test to phr1_test_cc.

To obtain predicted log(sbp) in the test sample using the main effects model, I used main_pred_logs <- mod_main **%>% predict**(phr1_test_cc). I then exponentiated those results to create a new set of values called main_pred_sbp which contain predicted sbp values in the test sample.

Then I created key summaries of the quality of predictions we made for sbp using mod_main, and placed them in a tibble called main_summaries. Here's the code that created the model name and the validation R-square value. I also created the other two summary statistics required to answer the Question, and you'll need to do that, too, by adding to my code.

```
main_summaries <- tibble(model = "Main Effects",
                         R2 = R2(main_pred_sbp, phr1_test_cc$sbp),
                         # insert more code here
                         )
```

Then I followed the same steps to create aug_pred_logs and aug_pred_sbp and finally aug_summaries for the augmented model's performance in the test sample.

Then I created a nicely formatted comparison table (with columns labeled model, R2, RMSE and MAE) using this code.

```
bind_rows(main_summaries, aug_summaries) %>% kable(digits = c(0, 4, 3, 3))
```

Of course, I then actually did the comparison. One of the models had a better result on two of the three validation statistics, so I went with that model.

In **Question 10**, I fit the model as specified and called it mod_cc. To obtain a sufficiently attractive table, I used the tidy function on mod_cc, and selected to show five columns. I didn't use kable in Question 10, but when I showed the tibble again in Question 11, I did, rounding everything off to 5 digits, so either way is fine, I suppose. My final tibble starts like this …

term	estimate	std.error	p.value	conf.low	conf.high
(Intercept)	4.59288	0.01509	0.00000	4.56330	4.62246
hba1c	0.00644	0.00091	0.00000	0.00466	0.00822
bmi	0.00254	0.00017	0.00000	0.00221	0.00288
ldl	0.00030	0.00004	0.00000	0.00022	0.00037

which gives you the idea that rounding to three decimals (in the case of ldl) would be a problem.

In **Question 11**, I used the mice package to do the multiple imputation as requested, after setting the seed that was specified in the question, and using m = 10 imputations.

I then fit the main effects model on each imputed data frame, and then pooled the models.

My multiple imputation result used summary on the pooled models, and I elected not to show either the df or the test statistic, but I actually did use kable(digits = 5) so the start of my results were:

	estimate	std.error	p.value	2.5 %	97.5 %
(Intercept)	4.58687	0.01374	0.00000	4.55993	4.61381
hba1c	0.00629	0.00085	0.00000	0.00463	0.00796
bmi	0.00244	0.00016	0.00000	0.00212	0.00276
ldl	0.00029	0.00004	0.00000	0.00022	0.00037

For **Question 12**, the description of the effect of the insurance variable on log(sbp) is what I'm looking for, rather than trying to describe the result in terms of an effect on sbp.

You do need to carefully explain what the insurance effect size means (it doesn't mean having insurance vs. not having insurance, for example.)

In building a response, I would focus on the changes in both the point and interval estimates (and even the much-maligned p value) in describing the impact of multiple imputation vs. a complete case analysis.

Good luck. Homework 4 is due to Canvas at