432 Class 12

Thomas E. Love, Ph.D.

2024-02-22

## Today’s Agenda

* Using caret to help with k-fold cross validation
* Building a Table One

## Today’s R Setup

knitr::opts\_chunk$set(comment = NA)  
  
library(janitor)  
library(broom)  
library(caret)  
library(tableone)  
library(tidyverse)  
  
theme\_set(theme\_bw())

# K-Fold Cross-Validation

## Tiny Data Set for a Linear Model

The maleptsd file contains information on PTSD (post traumatic stress disorder) symptoms following childbirth for 64 fathers[[1]](#footnote-22).

maleptsd <- read\_csv("c12/data/maleptsd.csv", show\_col\_types = FALSE) |>   
 clean\_names() |>  
 mutate(ptsd = log(ptsd\_raw + 1)) |>  
 relocate(ptsd, .after = id)

ptsd, the response, measures PTSD symptoms, and we have 10 candidate predictors for that response.

## The maleptsd data

maleptsd

# A tibble: 64 × 13  
 id ptsd over2 over3 over5 bond posit neg contr sup cons aff  
 <chr> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>  
 1 P01 0 0 0 10 26 48.8 4.6 58.8 18.8 65 10  
 2 P02 3.95 3 3 10 28 49.9 10.9 65 13.8 60 12  
 3 P03 2.20 10 10 10 22 22.5 34.2 59.5 10 65 12  
 4 P04 1.39 0 0 10 22 40.3 20.2 38.7 16 50 11  
 5 P05 1.61 8 7 9 22 43 32.1 41.8 14.2 51 9  
 6 P06 2.08 1 1 9 23 44.9 19.3 59.1 17.3 61 12  
 7 P07 2.08 0 0 10 24 39.3 25 45 14.3 49 10  
 8 P08 1.10 8 10 8 26 23.5 36.6 25.1 1.7 55 12  
 9 P09 3.33 9 10 4 18 11.5 38.9 24 10.3 14 0  
10 P10 0.693 5 0 9 19 35.6 25.1 29.9 2.3 48 10  
# ℹ 54 more rows  
# ℹ 1 more variable: ptsd\_raw <dbl>

## More predictors than we can handle

Only 64 observations and 10 potential predictors.

* Suppose we decide to try 5 of the predictors, specifically over3, bond, neg, sup and aff.

m1 <- lm(ptsd ~ over3 + bond + neg + sup + aff, data = maleptsd)  
  
glance(m1) |> select(r2 = r.squared, adjr2 = adj.r.squared, AIC, BIC)

# A tibble: 1 × 4  
 r2 adjr2 AIC BIC  
 <dbl> <dbl> <dbl> <dbl>  
1 0.260 0.197 206. 221.

* Note nominal of 0.260.

## Set up five-fold cross-validation

Use the **caret** package’s trainControl() function

set.seed(4322024)  
ctrl <- trainControl(method = "cv", number = 5)

Next, we train our model on those five folds:

ptsd\_mod <- train(ptsd ~ over3 + bond + neg + sup + aff,  
 data = maleptsd, method = "lm",   
 trControl = ctrl)

Results on next slide.

## ptsd\_mod results

ptsd\_mod

Linear Regression   
  
64 samples  
 5 predictor  
  
No pre-processing  
Resampling: Cross-Validated (5 fold)   
Summary of sample sizes: 52, 51, 50, 51, 52   
Resampling results:  
  
 RMSE Rsquared MAE   
 1.239869 0.1428023 1.072237  
  
Tuning parameter 'intercept' was held constant at a value of TRUE

* Compare this to the nominal we saw earlier of 0.2604.

## A New Model with Two Predictors

Perhaps we can justify a two-predictor model.

m2 <- lm(ptsd ~ aff + neg, data = maleptsd)  
  
glance(m2) |> select(r2 = r.squared, adjr2 = adj.r.squared, AIC, BIC)

# A tibble: 1 × 4  
 r2 adjr2 AIC BIC  
 <dbl> <dbl> <dbl> <dbl>  
1 0.203 0.176 205. 213.

## Train new model on same 5 folds

ptsd\_mod2 <- train(ptsd ~ neg + aff, data = maleptsd,   
 method = "lm", trControl = ctrl)  
  
ptsd\_mod2

Linear Regression   
  
64 samples  
 2 predictor  
  
No pre-processing  
Resampling: Cross-Validated (5 fold)   
Summary of sample sizes: 52, 52, 50, 51, 51   
Resampling results:  
  
 RMSE Rsquared MAE   
 1.154213 0.2030121 0.9863553  
  
Tuning parameter 'intercept' was held constant at a value of TRUE

## Model Summaries within each fold

ptsd\_mod2$resample

RMSE Rsquared MAE Resample  
1 1.1536878 0.18020759 0.9206545 Fold1  
2 1.2300847 0.02992272 1.0488347 Fold2  
3 0.9475468 0.21652976 0.8473368 Fold3  
4 1.3963512 0.03472384 1.1842472 Fold4  
5 1.0433949 0.55367662 0.9307033 Fold5

## Final Model from cross-validation

ptsd\_mod2$finalModel

Call:  
lm(formula = .outcome ~ ., data = dat)  
  
Coefficients:  
(Intercept) neg aff   
 -0.23311 0.04191 0.10648

glance(ptsd\_mod2$finalModel) |>  
 select(r2 = r.squared, adjr2 = adj.r.squared, AIC, BIC)

# A tibble: 1 × 4  
 r2 adjr2 AIC BIC  
 <dbl> <dbl> <dbl> <dbl>  
1 0.203 0.176 205. 213.

## Tidied Coefficients from C-V model 2

tidy(ptsd\_mod2$finalModel, conf.int = TRUE,   
 conf.level = 0.90)

# A tibble: 3 × 7  
 term estimate std.error statistic p.value conf.low conf.high  
 <chr> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>  
1 (Intercept) -0.233 0.527 -0.443 0.660 -1.11 0.646   
2 neg 0.0419 0.0125 3.34 0.00143 0.0210 0.0629  
3 aff 0.106 0.0473 2.25 0.0280 0.0275 0.185

### “Variable Importance”

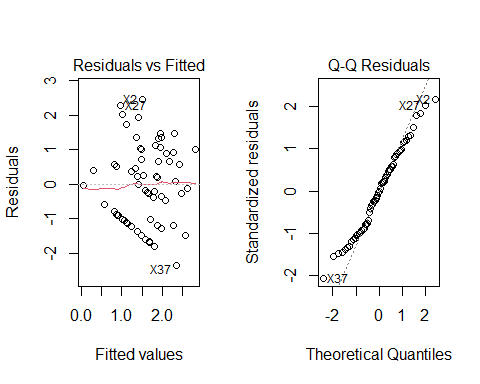
varImp(ptsd\_mod2)

lm variable importance  
  
 Overall  
neg 100  
aff 0

Linear Fit: based on absolute value of t statistic for each parameter

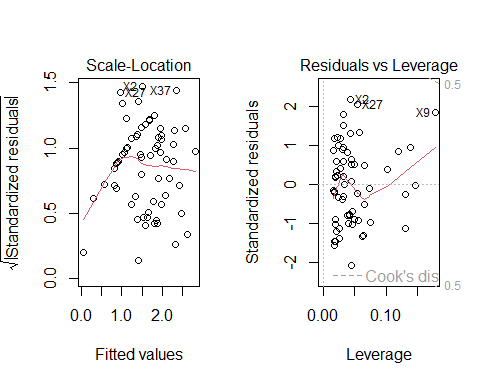
## Residual Plots (1) for final model

par(mfrow = c(1,2)); plot(ptsd\_mod2$finalModel, which = c(1:2))



## Residual Plots (2) for final model

par(mfrow = c(1,2)); plot(ptsd\_mod2$finalModel, which = c(3,5))



## More on K-fold Cross-Validation

Linear regression example in section 16.5 of our Course Notes.

* More on caret package at <https://topepo.github.io/caret/> although that’s older now, and the tidymodels approach will allow us to do a lot of the same things later this term.

Can you do something similar to this with a glm() fit in logistic regression?

* Yes, definitely. See the next few slides and also [Section 48 here](https://rforhr.com/kfold.html#estimate_kfold_logistic).

## K-Fold Cross-Validation for a Logistic Regression Fit using glm()

Suppose we look at a new outcome: is ptsd > 0?

maleptsd <- maleptsd |>  
 mutate(gg0 = factor(ifelse(ptsd == 0, "No", "Yes"))) |>  
 relocate(gg0)  
  
maleptsd |> tabyl(gg0)

gg0 n percent  
 No 18 0.28125  
 Yes 46 0.71875

Suppose we want to fit a logistic regression model to predict gg0 on the basis of aff and neg (the same two predictors as we used for ptsd)

## Set up five-fold cross-validation

set.seed(43220241)  
ctrl <- trainControl(method = "cv", number = 5)  
  
gg0\_mod <- train(gg0 ~ neg + aff, data = maleptsd,   
 method = "glm", family = binomial,   
 trControl = ctrl)  
gg0\_mod

Generalized Linear Model   
  
64 samples  
 2 predictor  
 2 classes: 'No', 'Yes'   
  
No pre-processing  
Resampling: Cross-Validated (5 fold)   
Summary of sample sizes: 52, 51, 51, 51, 51   
Resampling results:  
  
 Accuracy Kappa   
 0.7666667 0.3345462

## Interpreting these summaries

* Model accuracy shows the proportion of observations where the model classified the subject correctly.
  + Here, our model correctly classifies 76.7% of subjects
* Kappa measures classification accuracy but accounts for the baseline probabilities of the Yes/No groups.
  + Common Thresholds (from Landis and Koch 1977) are:
    - 0.81 - 1.0 Almost perfect, 0.61 - 0.80 Substantial, 0.41 - 0.60 Moderate, 0.21 - 0.40 Fair, 0.01 - 0.2 Slight
  + Here, our model does a fair job, with .

## Confusion Matrix

predictions <- predict(gg0\_mod)  
  
confusionMatrix(data = predictions, maleptsd$gg0)

Confusion Matrix and Statistics  
  
 Reference  
Prediction No Yes  
 No 6 5  
 Yes 12 41  
   
 Accuracy : 0.7344   
 95% CI : (0.6091, 0.837)  
 No Information Rate : 0.7188   
 P-Value [Acc > NIR] : 0.4528   
   
 Kappa : 0.2548   
   
 Mcnemar's Test P-Value : 0.1456   
   
 Sensitivity : 0.33333   
 Specificity : 0.89130   
 Pos Pred Value : 0.54545   
 Neg Pred Value : 0.77358   
 Prevalence : 0.28125   
 Detection Rate : 0.09375   
 Detection Prevalence : 0.17188   
 Balanced Accuracy : 0.61232   
   
 'Positive' Class : No

## Use summary() to obtain final model

summary(gg0\_mod)

Call:  
NULL  
  
Coefficients:  
 Estimate Std. Error z value Pr(>|z|)   
(Intercept) -2.49291 1.21035 -2.060 0.0394 \*   
neg 0.08451 0.02990 2.827 0.0047 \*\*  
aff 0.21229 0.10864 1.954 0.0507 .   
---  
Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
  
(Dispersion parameter for binomial family taken to be 1)  
  
 Null deviance: 76.049 on 63 degrees of freedom  
Residual deviance: 63.255 on 61 degrees of freedom  
AIC: 69.255  
  
Number of Fisher Scoring iterations: 4

## glance() and tidy() after c/v

glance(gg0\_mod$finalModel)

# A tibble: 1 × 8  
 null.deviance df.null logLik AIC BIC deviance df.residual nobs  
 <dbl> <int> <dbl> <dbl> <dbl> <dbl> <int> <int>  
1 76.0 63 -31.6 69.3 75.7 63.3 61 64

tidy(gg0\_mod$finalModel, exponentiate = TRUE,   
 conf.int = TRUE, conf.level = 0.90)

# A tibble: 3 × 7  
 term estimate std.error statistic p.value conf.low conf.high  
 <chr> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>  
1 (Intercept) 0.0827 1.21 -2.06 0.0394 0.00978 0.544  
2 neg 1.09 0.0299 2.83 0.00470 1.04 1.15   
3 aff 1.24 0.109 1.95 0.0507 1.04 1.49

## Variable Importance in model classification

varImp(gg0\_mod)

glm variable importance  
  
 Overall  
neg 100  
aff 0

So the information from neg is what’s making the difference here.

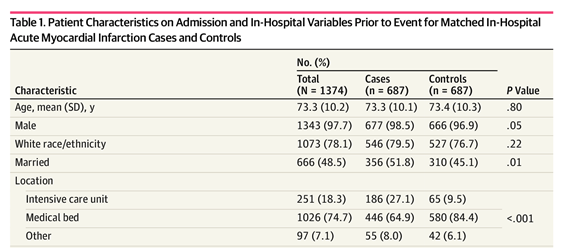
# Building a Table One

## An Original Clinical Investigation

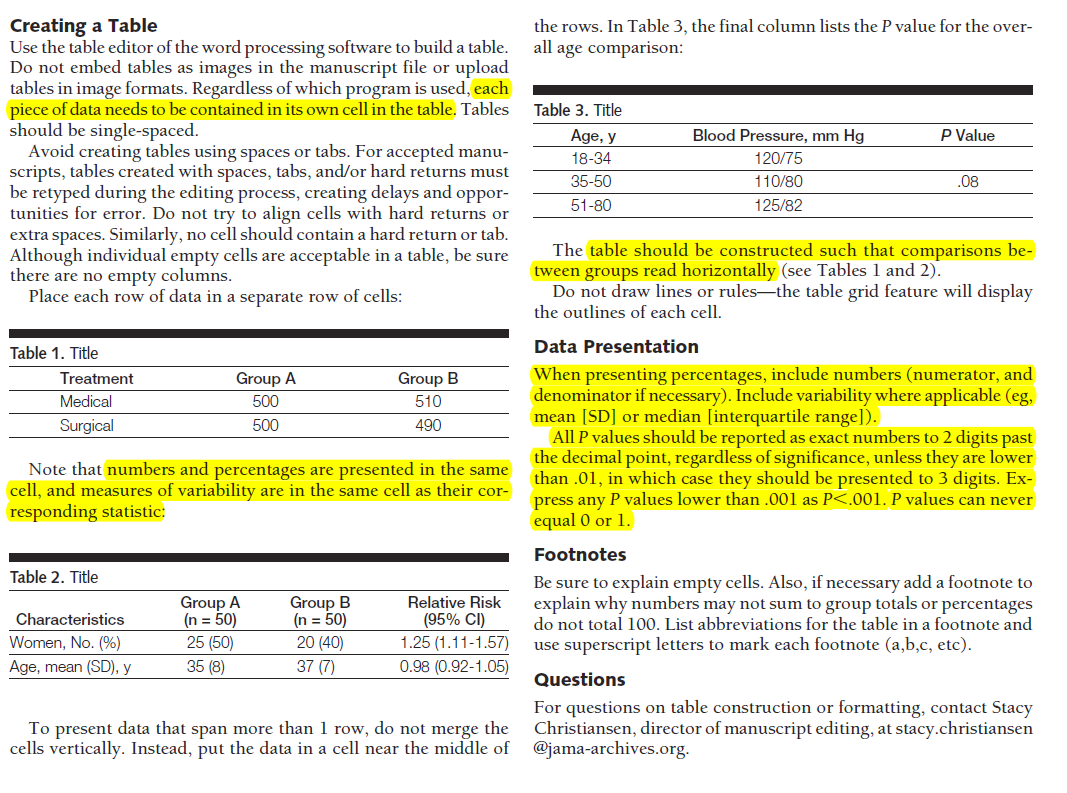


[Link to Source](https://jamanetwork.com/journals/jamanetworkopen/fullarticle/2720923)

## Part of Bradley et al.’s Table 1



## Table Creation Instructions, JAMA: [linked here](https://jama.jamanetwork.com/data/ifora-forms/jama/tablecreationinst.pdf)



## A Data Set

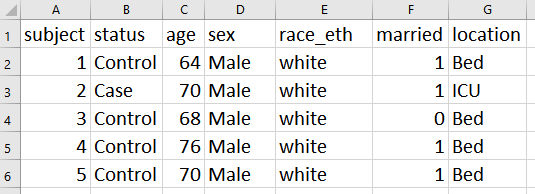
The bradley.csv data set is simulated, but consists of 1,374 observations (687 Cases and 687 Controls) containing:

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

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

## The bradley.csv data (first 5 rows)

* The bradley\_sim.md file on our web site shows you how I simulated the data.



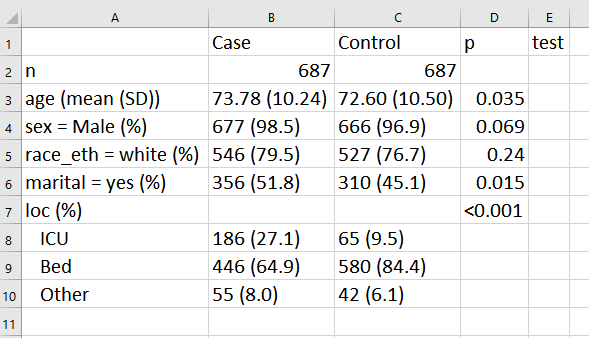
## To “Live” Coding

On our web site (Data and Code + Class 12 materials)

* In the data folder:
  + bradley.csv data file
* bradley\_table1.qmd Quarto script
* bradley\_table1.md Results of running Quarto
* bradley\_table1\_result.csv is the table generated by that Quarto script

# To The “Live Code”

## Opening bradley\_table1\_result.csv in Excel



## Learning More About Table 1

Chapter 18 of the Course Notes covers two larger examples, and more details, like…

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

## Next Time

* Thinking About Power: Retrospective Design
* Robust Linear Models

Good luck on the Quiz!

1. Source: Ayers et al. 2007 *J Reproductive and Infant Psychology*. The data are described in more detail in Wright DB and London K (2009) *Modern Regression Techniques Using R* Sage Publications. [↑](#footnote-ref-22)