Cleaning Description

Drew Faturos

10/22/2017

Questions Relating to the paper by Brady and Li

1 In the in-course exercises, we have been analyzing data with accident as the observation unit. This study uses a different observation unit. What is the unit of observation in the Brady and Li study? When you download the FARS data for a year, you get a zipped folder with several different datasets. Which of the FARS datasets provides information at this observation level (and so will be the one you want to use for this analysis)?

The Brady and Li study utilizes fatalities that occurred in less than one hour as the primary unit of study. The FARS dataset containing information on individual people titled person contains relevant information useful for the analysis conducted by Brady and Li.

2 This study only analyzes a subset of the available FARS data. Enumerate all of the constraints that are used by the study to create the subset of data they use in their study (e.g., years, states, person type, injury type). Go through the FARS documentation and provide the variable names for the variables you will use to create filter statements in R to limit the data to this subset. Provide the values that you will want to keep from each variable.

In the study Brady and Li utilize drivers from California, Hawaii, Illinois, New Hampsire, Rhode Island and West Virginia, who died as a result of an automobile less than one hour following the crash. The researchers utilize drug and alcohol test results in addition to driver characteristics such as sex, race, and age.

This variables utilized are:
AGE,
SEX, where 1 is male and 2 is female,
Person type (Per_typ) where 1 is driver
Injury Severity (INJ_SEV) where fatal injury/k = 4
Alcohol Test Result (ALC_RES)
0-93 = value of BAC test, 96 = no test given, 95/- = test refused/not reported
Drug Test Result (DRUGRES1, DRUGRES2, DRUGRES3) 0=Not Tested for Drugs, 1 = No Drugs Reported/Negative, –/95 = Not Reported, 100-295 = Narcotic, 300-395 = Depressant, 400-495 = Stimulant, 500-595 = Hallucinogen, 600-695 = Cannabinoid, 700-795 = Phencyclidine (PCP), 800-895 = Anabolic Steroid, 900-995 = Inhalant, 996 996 = Other Drugs, 997 997 = Tested for Drugs, Results Unknown, 998 = Tested for Drugs, Drugs Found, Type Unknown/Positive, 999/– Unknown if Tested/Not Reported, –/999 Unknown if Tested
Lag hours (LAG_HRS) time in hours between crash and death 99= unknown
Lag minutes (LAG_MIN) time in miutes between crash and death 99=unknown
State (STATE) where Illionis = 17, New Hamsphire = 33, Rhoide Island = 44, California = 6 Hawaii = 15 and west Virginia = 54

3 The study gives results stratified by age category, year, year category (e.g., 1999–2002), alcohol level, non-alcohol drug category, and sex. For each of these stratifications, give the variable name for the FARS variable you could use to make the split (i.e., the column name you would use in a group_by statement to create summaries within each category or the column name you would mutate to generate a categorical variable). Describe how each of these variables are coded in the data. Are there any values for missing data that you'll need to mutate to NA values in R? Are there any cases where coding has changed over the study period?
Age Category: use age variable (AGE) and group into the following three categories: < 25 years old, 25 -44 years old, 45- 64 years old, >= 65 years old. Variable coding changes and the pertinent values are as follows:

1999 – 2008 unknown = 99, 2009 – 2010 unknown = 999, in 2009 - = Not reported, and in 2010 998= Not Reported. I will need to recode the missing values (999, 998) as NA before starting the anaylsis

Sex: Use sex variable (Sex) and group between males and females (1 = male, 2 = Female, 9= unknown) There were slight coding changes involving not reported data. Not reported = - between 1999 – 2009 and in 2010 not reported changed to 8. I will need to recode the missinng variable as NA.

Year: define year as a variable after combining all the individual yearly data. Make categories for groups as follows: 1999-2002, 2003-2006, 2007-2010. There should be no missing values.

Alcohol Level: Use Alcohol test result variable (ALC_RES) positive test any value between 10 and 94. Codding changes between 2009 and 2010; however, the values pertinent to the analysis do not change because researchers only care if people test positive based on BAC and not issues surrounding test. There are missing values that I may have to recode to NA.

Non alcohol Drug use: Use drug variables (DRUGRES1, DRUGRES2, DRUGRES3) to create categories for: Cannibal: 600-695, Stimulant: 400-496, Narcotic: 100-295, Depressant (non-alcohol): 300-395, Other (hallocenogens, phencyclidine, anabolic steroids, inhalants or other): 500-996. Codding changes between 2009 and 2010; however, the values pertinent to the anaylsis do not change because only care about testing positive and not the issues surrounding the test.

Function used to clean the data, with comments indicating what is happening in each step.

comments of indivudal lines included after function

```
getwd()
```

```
## [1] "/Users/Drew/r_programming/fars_analysis/writting"
```

```
setwd("~/r_programming/fars_analysis")
library(tidyverse)

clean_yearly_person_file <- function(year) {
  # 1. Read data in.
person_file <- paste0("data-raw/yearly_person_data/person_", year, ".csv")
df <- readr::read_csv(person_file)
# 2. Convert all column names to lowercase.
colnames(df) <- tolower(colnames(df))

df <- df %>%
    # 3. Limit variables.
dplyr::select(st_case, veh_no, per_no, state, per_typ, lag_hrs,
              lag_mins, inj_sev, age, alc_res, contains("drugres"), sex) %>%

    # 4. Limit to relevant `per_typ` and `inj_sev` values, then remove those variables.
dplyr::filter(per_typ == 1 & inj_sev == 4) %>%
  dplyr::select(-per_typ, -inj_sev) %>%
    # 5. Create a `unique_id`. Note: to be unique, `year` needs to be pasted on.
tidyr::unite(unique_id, st_case, veh_no, per_no) %>%
  dplyr::mutate(year = year,
                unique_id = paste(unique_id, year, sep = "_")) %>%
    # 6. Limit to study states and then remove the `state` variable.
dplyr::filter(state %in% c(6, 15,
                           17,
```

```
                                33,
                                44,
                                54)) %>%
dplyr::select(-state) %>%
    # 7. Convert `sex` to a factor with levels "Male" and "Female".
dplyr::mutate(sex = ifelse(sex == 9, NA, sex),
              sex = factor(sex, levels = c(1, 2),
                           labels = c("Male", "Female"))) %>%
# 8. Use measured alcohol blood level to create `Alcohol` (logical for whether # alcohol was present).
dplyr::mutate(alc_res = ifelse(alc_res > 94, NA, alc_res / 10),
              Alcohol = alc_res >= 0.01) %>%
  dplyr::select(-alc_res) %>%

    # 9. Specify missing values for the lag minutes.
dplyr::mutate(lag_mins = ifelse(lag_mins == 99, NA, lag_mins))
  # 10. Save lag hours coded as missing as `NA`.
  if(year <= 2008){
    df <- df %>%
dplyr::mutate(lag_hrs = ifelse(lag_hrs %in% c(99, 999), NA, lag_hrs)) } else {
df <- df %>%
dplyr::mutate(lag_hrs = ifelse(lag_hrs == 999, NA, lag_hrs))
}

# 11. Limit to deaths within an hour of the accident then remove those variables.
df <- df %>%
dplyr::filter((lag_hrs < 1) | (lag_hrs == 1 & lag_mins == 0)) %>% dplyr::select(-lag_hrs, -lag_mins)
# 12. Save age values coded as missing as `NA`.
if(year <= 2008){
  df <- df %>%
dplyr::mutate(age = ifelse(age == 99, NA, age)) } else {
df <- df %>%
dplyr::mutate(age = ifelse(age %in% c(998, 999), NA, age))
}
# 13. Use age to create age categories and then remove `age` variable.
df <- df %>%
dplyr::mutate(agecat = cut(age, breaks = c(0, 25, 45, 65, 1000),
        labels = c("< 25 years",
           "25--44 years",
           "45--64 years",
           "65 years +"),
include.lowest = TRUE, right = FALSE)) %>%
  dplyr::select(-age)
# 14. Gather all the columns with different drug listings (i.e., `drugres1`, # `drugres2`, `drugres3`).
# drug categories.
gathered_df <- df %>%
tidyr::gather(drug_number, drug_type_raw, contains("drugres")) %>%
  dplyr::mutate(drug_type = ifelse(drug_type_raw %in% 100:295,
                    "Narcotic", NA),
              drug_type = ifelse(drug_type_raw %in% 300:395,
                  "Depressant", drug_type),
              drug_type = ifelse(drug_type_raw %in% 400:495,
                  "Stimulant", drug_type),
              drug_type = ifelse(drug_type_raw %in% 600:695,
```

```
                  "Cannabinoid", drug_type),
              drug_type = ifelse(drug_type_raw %in% c(500:595, 700:996),
                  "Other", drug_type),
              drug_type = ifelse(drug_type_raw == 1,
                  "None", drug_type),
              drug_type = factor(drug_type)) %>%
  dplyr::select(-drug_type_raw, -drug_number) %>%
  # 15. Filter out any observations where both alcohol and drug data is missing.
  dplyr::filter(!(is.na(Alcohol) & is.na(drug_type)))
# 16. Create a subset with only individuals with at least one non-missing
# listing for drugs. (Write a sentence or two for each step in this pipe chain.)
non_missing_drugs <- gathered_df %>%
  filter(!is.na(drug_type)) %>%
  group_by(unique_id, drug_type) %>%
  summarize(has_drug = TRUE) %>%
  ungroup() %>%
  mutate(row_num = 1:n()) %>%
  spread(drug_type, has_drug, fill = FALSE) %>%
  select(-row_num)
# 17. Join this back into the full dataset. (Write a sentence or two for each # step in this pipe chain
df <- df %>%
  dplyr::select(-contains("drugres")) %>% dplyr::full_join(non_missing_drugs, by = "unique_id") %>%
  dplyr::select(-None) %>%
  tidyr::gather(drug_type, positive_for_drug, Alcohol, Cannabinoid,
                Depressant, Narcotic, Other, Stimulant) %>%
  dplyr::mutate(drug_type = factor(drug_type)) %>%
  unique()

return(df)
}

# 18. Iterate the clean_yearly_person_file function across study years to
# create and save a single dataset.
# Note: map_df() is similar to map(), but it binds elements of the
# list resulting from map() together. To understand this step, try
# running this code with map instead of map_df, check out documentation # for map and map_df, and look
# `map_df` in your console.
clean_fars <- map_df(1999:2010, clean_yearly_person_file)
save(clean_fars, file = "data/clean_fars.RData")

# checking that it worked
dim(clean_fars)
```

```
## [1] 156413      6
```

```
length(unique(clean_fars$unique_id))
```

```
## [1] 25593
```

```
summary(clean_fars)
```

```
##   unique_id             sex              year              agecat
##  Length:156413      Male  :121072   Min.   :1999   < 25 years  :39149
##  Class :character   Female: 35335   1st Qu.:2002   25--44 years:61235
##  Mode  :character   NA's  :     6   Median :2004   45--64 years:39870
```

```
##                                     Mean   :2004   65 years +  :16108
##                                     3rd Qu.:2007   NA's        :   51
##                                     Max.   :2010
##        drug_type     positive_for_drug
##  Alcohol    :25593   Mode :logical
##  Cannabinoid:26260   FALSE:127597
##  Depressant :25988   TRUE :16894
##  Narcotic   :26086   NA's :11922
##  Other      :26179
##  Stimulant  :26307
```

Description of cleaning code:

1 Read data in.

Raw data is being read into R from a csv file and saved as a data frame tilted df. The data is for a specific year from the PERSON dataset contained with the FARS database. (The data has already been converted from a .dbf file to a .csv file.) The data is being read in so that it can be cleaned prior to being analyzed.

2 Convert all columns names to lowercase

The data from the FARS database contains variable names in all capitols. The variable names are being changed to lower case to make the data tidy and make the variable names easier to work with.

3 Limit Variables.

The PERSON data file within the FARS database contains a lot of variables that are not pertinent to the analysis. This step selects the variables needed for the analysis and removes all the extraneous variables. This makes the analysis cleaner and reduces the amount of computing power required.

4 Limit to relevant 'per_tep' and 'Inj_ser' values, then remove those variables.

This step limits the data set to drivers (`per_tep==1`) who sustained a fatal injury as a result of the accident (`inj_sev==4`). By getting rid of the variables afterwards, the data set is cleaned and all other accidents are removed from the data set.

5 Create a 'unique_id'. Note to be unique 'year' needs to be pasted on.

This step creates a unique ID for each fatality. This allows individual events to be distinguished once the data from all the years are combined. Year needs to be added in case there is overlap between individual recording numbers between years. This ID is also used for a merging step used later in the function.

6 Limit to study states and then remove the 'state' variable.

This code line limits the study to states that preformed drug screening. The numbers correlate to state names and the following states are selected: California, Illinois, New Hampshire, Rhode Island, Hawaii and West Virginia. By removing the state variable afterwards, the size of the data frame is reduced to only include states pertinent to the analysis.

7 Convert sex to a factor with levels "Male" and "Female"

This step makes sex a factor variable so `sex` is easier to analyze. Commands specificying male will be output as "Male" instead of as 1. Making sex a factor variable, also makes it easier to group the analysis by sex. This step also recodes missing variables as NA using an ifelse() statement. This allows R to recognize the missing variables.

8 Use measured alcohol blood level to create 'Alcohol' (logical for whether alcohol was present. Then remove the `alc_res` variable.

This command recoded the alcohol variable so all instances of no BAC testing or issues involving testing are recoded as missing (NA), which is done through an ifelse() statement. This statement also creates a new variable called `Alcohol` that says true for all instances where the driver had a BAC over 0.01g/dl. The variable `alc_res` is then removed because it is no longer pertinent because a new variable that is easier to interpret was created in its place.

9 Specify missing values for the lag minutes.

This mutate step recodes the variable `lag min` so that missing values are coded as NA instead of being coded as numeric values. This is accomplished through an ifelse() statement. By recoding missing values to NA, R is able to know there are missing values and will automatically exclude them from analysis unless specified otherwise.

10 Save lag hours codes as missing as 'NA'.

This command recoded the variable hours after death to make missing variables recognized in R using an if statement combined with an ifelse() command. The if statement is included to account for the variability of the coding for lag hours which changed in 2008.

11 Limit to deaths within an hour of the accident then remove those variables.

This filter step limits the dataset to fatalities that occurred in under one hour by specifying any time under one hour or a time of exactly one hour and zero minutes. The command then drops the variables measuring time after death because it is no longer needed. The data set now only contains accidents where the driver died less than hour following the accident.

12 Save age values coded as missing as 'NA'.

This step reformats the age variable to code missing as 'NA' so R recognizes them as opposed to treating 99 as an age. These lines of code also account for coding changes where missing values were coded one way before 2008 and recoded another way after 2008. This step is accomplished by using an if statement.

13 Use age to create age categories then remove 'age' variable.

This step reformats the variable `age` into a categorical variable with the following age categories: <25 years old, 25-44 years old, 45-64 years old, >65 years old. The age variable is then removed so individual ages are lost and replaced with age categories. This step was performed so the analysis can be stratified by individual age groups.

14 Gather all the columns with different drug listings. Convert from the numeric code listing to drug categories.

This step gathers all the different columns containing drug testing results and combines them to make one variable. This code chunk also changes the code used to specify different drug classes from a number range to the name of the corresponding drug. This step cleans the data because it makes a new variable titled `drug_type` that specifically identifies what drug was present as a result of drug testing. The code then removes the old varibles relating to drugs because they are no longer needed.

15 Filter our any observations where both alcohol and drug data is missing.

This step further cleans the data by removing all cases where information on alcohol and drugs is missing. As a result, only cases with an alcohol level above 0.01, positive drug test result, or both are included in the analysis. All other cases are now removed from the analysis. This step is important because it limits the analysis to only include people with pertinent data. This step creates a new dataframe titled `gathered_df`, which stores the results of this cleaning step.

16 Create a subset with only individuals with at least one non-missing listing for drugs.

Line 1. This step creates a new data frame titled `non_missing_drugs` using the `gathered_df` data frame.
Line 2: This step filters the data set to only include cases where information regarding type of drug is not missing.
Line 3: This step groups each case by the unique case identifier already created and the variable `drug_type` which no longer includes any missing values.
Line 4: This summarize command creates a new variable `has_drug` that is true for all unique cases that are positive for at least one drug test result.
Line 5: This step ungroups each individual case and the corresponding drug test results.
Line 6: This step creates a new variable `row_num` which gives each row in the data frame a sequential number starting at 1.
Line 7: This step takes the variable specifying individual drug and the variable indicating presence of any drug and codes all cases with no drug presence as false. The variable `has_drug` now reads true for all cases where a drug was present and reads false where no drugs are present. This step also creates a new variable for each category of drug type.
Line 8: This step removes the previously created variable row_num because it is no longer needed.

17 Join this back into the full dataset.

Line 1: This command takes the original cleaned data frame used up until comment 15 and applies the following changes to that data frame.
Line 2: This line removes all variables relating to drug testing results.
Line 3:This step joins the original cleaned data frame with the data frame created in comment 16 using the variable `unique_id`. This join does not delete any variables in either data frame.
Line 4: This line removes the variable `None` which was created as a result of the join.
Line 5: This step gathers all the different variables relating to drug type and creates a new variable titled `drug_type`.
Line 6: This step reclassifies the newly created variable drug_type as a factor variable.
Line 7: This step removes any duplications that resulted as a result of merging the cleaned data frame with the data frame containing data about drug exposure.

18 Iterate the clean_yearly_person_file function across study years to create and save a single dataset.

This step runs the function (`clean_yearly_person_file`) on each individual year of Fars data from 1999 to 2010 and combines the newly multiple year data into a new file titled `clean_fars`. This command cleans

all the data pertinent to the analysis and puts it in another folder. This clean data can now be used for analysis. The function `map_df` must be used instead of `map` because the function `map` does not combine all the individual years together. Instead the function `map` returns a data file with no data.

Creating Table of results

```
clean_fars %>%
  mutate(year_cat = cut(year, breaks = c(1999, 2002, 2006, 2010),
                        labels = c("1999-2002", "2003-2006", "2007-2010"),
                        include.lowest = TRUE, right = TRUE)) %>% filter(!is.na(sex))%>%
  group_by(drug_type, sex, year_cat) %>%
  summarize(n_non_missing = sum(!is.na(positive_for_drug)),
            positive_test = sum(positive_for_drug, na.rm = TRUE),
            perc_positive = round(100 * positive_test / n_non_missing, 1)) %>%
  select(drug_type, sex, year_cat, perc_positive) %>%
  unite(sex_year_cat, sex, year_cat) %>%
  spread(sex_year_cat, perc_positive) %>%
  knitr::kable(col.names = c("Drug type", "F 1999-2002",
                             "F 2003-2006", "F 2007-2010",
                             "M 1999-2002", "M 2003-2006",
                             "M 2007-2010"))
```

Drug type	F 1999-2002	F 2003-2006	F 2007-2010	M 1999-2002	M 2003-2006	M 2007-2010
Alcohol	26.4	24.3	27.1	43.2	42.9	43.3
Cannabinoid	2.8	5.7	7.3	5.8	10.3	11.8
Depressant	3.4	3.8	4.8	2.0	2.5	3.2
Narcotic	4.2	4.9	7.0	2.2	3.4	4.0
Other	5.6	6.6	7.2	4.3	4.5	4.2
Stimulant	7.2	9.1	8.7	10.5	11.9	9.2