Data Manipulations and Tables

Shay O'Brien

September 13, 2018

Loading Data and Subsetting

Let's load a CSV file with data on violent crime rates in different US states.

This file contains data on arrests per 100,000 residents for assault and murder in the 50 US states in 1973. Also given is the percent of the population living in urban areas.

```
library(dplyr)
library(xtable)
library(ggplot2)

# Set working directory
setwd("/Users/Shay/Dropbox/Soc500/2018 Materials/Precepts/Precept 1/2018_500")

# Read in data
state_crime <- read.csv(file = "uscrime.csv", header = TRUE)
# Preview your dataset
  #to see the first 6 rows
head(state_crime)
```

```
##        State Murder Assault UrbanPop
## 1    Alabama   13.2     236       58
## 2     Alaska   10.0     263       48
## 3    Arizona    8.1     294       80
## 4   Arkansas    8.8     190       50
## 5 California    9.0     276       91
## 6   Colorado    7.9     204       78
```

```
  #to see one cell
state_crime[3, 1]
```

```
## [1] Arizona
## 50 Levels: Alabama Alaska Arizona Arkansas California ... Wyoming
```

```
# Create a new variable from existing data
# Calculate total number of violent crimes
state_crime$tot_crime <- state_crime$Murder + state_crime$Assault
```

Now let's subset these data.

```
# Select all observations (rows) where total crime / 100,000 residents is greater
# than 200 / 100,000 residents.

# Three ways:
state_crime[state_crime$tot_crime > 200, ] # By indexing
subset(state_crime, tot_crime > 200) # Using the subset function
filter(state_crime, tot_crime > 200) # Using dplyr

##Note: these commands will print the entire subset. For large datasets, save subsets as new objects fi
```

```r
subset <- state_crime[state_crime$tot_crime > 200, ]
head(subset)
```

```r
# How many of these states are there?
nrow(state_crime[state_crime$tot_crime > 200, ])
```

```
## [1] 20
```

```r
nrow(subset)
```

```
## [1] 20
```

```r
# We can subset for many different combinations of conditions
state_crime[state_crime$Murder < 10 & state_crime$UrbanPop >= 50, ] # & is logical AND
```

```
##            State Murder Assault UrbanPop tot_crime
## 3        Arizona    8.1     294       80     302.1
## 4       Arkansas    8.8     190       50     198.8
## 5     California    9.0     276       91     285.0
## 6       Colorado    7.9     204       78     211.9
## 7    Connecticut    3.3     110       77     113.3
## 8       Delaware    5.9     238       72     243.9
## 11        Hawaii    5.3      46       83      51.3
## 12         Idaho    2.6     120       54     122.6
## 14       Indiana    7.2     113       65     120.2
## 15          Iowa    2.2      56       57      58.2
## 16        Kansas    6.0     115       66     121.0
## 17      Kentucky    9.7     109       52     118.7
## 19         Maine    2.1      83       51      85.1
## 21 Massachusetts    4.4     149       85     153.4
## 23     Minnesota    2.7      72       66      74.7
## 25      Missouri    9.0     178       70     187.0
## 26       Montana    6.0     109       53     115.0
## 27      Nebraska    4.3     102       62     106.3
## 29 New Hampshire    2.1      57       56      59.1
## 30    New Jersey    7.4     159       89     166.4
## 35          Ohio    7.3     120       75     127.3
## 36      Oklahoma    6.6     151       68     157.6
## 37        Oregon    4.9     159       67     163.9
## 38  Pennsylvania    6.3     106       72     112.3
## 39  Rhode Island    3.4     174       87     177.4
## 44          Utah    3.2     120       80     123.2
## 46      Virginia    8.5     156       63     164.5
## 47    Washington    4.0     145       73     149.0
## 49     Wisconsin    2.6      53       66      55.6
## 50       Wyoming    6.8     161       60     167.8
```

```r
state_crime[state_crime$Murder < 3 | state_crime$Assault < 50, ] # | is logical OR (inclusive)
```

```
##            State Murder Assault UrbanPop tot_crime
## 11        Hawaii    5.3      46       83      51.3
## 12         Idaho    2.6     120       54     122.6
## 15          Iowa    2.2      56       57      58.2
## 19         Maine    2.1      83       51      85.1
## 23     Minnesota    2.7      72       66      74.7
## 29 New Hampshire    2.1      57       56      59.1
```

```
## 34  North Dakota    0.8      45       44      45.8
## 45       Vermont    2.2      48       32      50.2
## 49     Wisconsin    2.6      53       66      55.6
```

```
high_murder <- state_crime[state_crime$Murder > mean(state_crime$Murder), ] # Murder rate above mean
```

Creating Tables of Summary Statistics

Let's find the mean and standard deviation of three variables in the dataset "diamonds."

```
head(diamonds)
```

```
## # A tibble: 6 x 10
##   carat       cut color clarity depth table price     x     y     z
##   <dbl>     <ord> <ord>   <ord> <dbl> <dbl> <int> <dbl> <dbl> <dbl>
## 1  0.23     Ideal     E     SI2  61.5    55   326  3.95  3.98  2.43
## 2  0.21   Premium     E     SI1  59.8    61   326  3.89  3.84  2.31
## 3  0.23      Good     E     VS1  56.9    65   327  4.05  4.07  2.31
## 4  0.29   Premium     I     VS2  62.4    58   334  4.20  4.23  2.63
## 5  0.31      Good     J     SI2  63.3    58   335  4.34  4.35  2.75
## 6  0.24 Very Good     J    VVS2  62.8    57   336  3.94  3.96  2.48
```

```
mn1 <- mean(diamonds$carat) # calculate mean
mn2 <- mean(diamonds$depth)
mn3 <- mean(diamonds$price)
sd1 <- sd(diamonds$carat) # calculate SD
sd2 <- sd(diamonds$depth)
sd3 <- sd(diamonds$price)
## Combine to form table
tbl <- cbind(c(mn1, mn2, mn3), c(sd1, sd2, sd3))
colnames(tbl) <- c("Mean", "SD") # add column names
rownames(tbl) <- c("carat", "depth", "price") # add row names
tbl
```

```
##               Mean           SD
## carat    0.7979397    0.4740112
## depth   61.7494049    1.4326213
## price 3932.7997219 3989.4397381
```

Let's make this table pretty:

```
print(xtable(tbl, caption = "Summary statistics for diamonds data"), comment = F)
```

	Mean	SD
carat	0.80	0.47
depth	61.75	1.43
price	3932.80	3989.44

Table 1: Summary statistics for diamonds data

But what happens if we want to calculate additional summary statistics? Unfortunately, the previous approach is not very scalable.

Let's use sapply instead to apply functions to multiple columns at once:

```r
# Select columns
vars <- c('carat', 'depth', 'price')
diamonds_r <- diamonds[, vars]

mns <- sapply(diamonds_r, mean)
sds <- sapply(diamonds_r, sd)

# Combine to form table
tbl2 <- cbind(mns, sds)
colnames(tbl2) <- c("Mean", "SD") # add column names
tbl2
```

```
##               Mean           SD
## carat    0.7979397    0.4740112
## depth   61.7494049    1.4326213
## price 3932.7997219 3989.4397381
```

We can better reuse our code by creating a function:

```r
get_summary <- function(data) {
  summary <- c(mean(data), sd(data))
  return(summary)
}

c(mean(diamonds_r$carat), sd(diamonds_r$carat))
```

```
## [1] 0.7979397 0.4740112
```

```r
# Apply this function to all variables
tbl3 <- t(sapply(diamonds_r, get_summary))
colnames(tbl3) <- c("Mean", "SD")
tbl3
```

```
##               Mean           SD
## carat    0.7979397    0.4740112
## depth   61.7494049    1.4326213
## price 3932.7997219 3989.4397381
```

Here is how you can also do this with dplyr and tidyr:

```r
diamond.summarized <- diamonds %>%
  select(carat, depth, price) %>%
  summarise_all(funs(Mean = mean, SD = sd))

diamond.summarized
```

```
## # A tibble: 1 x 6
##   carat_Mean depth_Mean price_Mean  carat_SD depth_SD price_SD
##        <dbl>      <dbl>      <dbl>     <dbl>    <dbl>    <dbl>
## 1  0.7979397    61.7494     3932.8 0.4740112 1.432621  3989.44
```

```r
# We can reshape this with tidyr
library(tidyr)

diamond.table <- diamond.summarized %>% gather(stat, val) %>% #from wide to long
  separate(stat, into = c("var", "stat"), sep = "_") %>%
  spread(stat, val)
```

```
#to see each step
diamond.summarized
```

```
## # A tibble: 1 x 6
##   carat_Mean depth_Mean price_Mean  carat_SD depth_SD price_SD
##        <dbl>      <dbl>      <dbl>     <dbl>    <dbl>    <dbl>
## 1  0.7979397    61.7494     3932.8 0.4740112 1.432621  3989.44
```

```
diamond.summarized %>% gather(stat, val)
```

```
## # A tibble: 6 x 2
##         stat          val
##        <chr>        <dbl>
## 1 carat_Mean    0.7979397
## 2 depth_Mean   61.7494049
## 3 price_Mean 3932.7997219
## 4   carat_SD    0.4740112
## 5   depth_SD    1.4326213
## 6   price_SD 3989.4397381
```

```
diamond.summarized %>% gather(stat, val) %>% #from wide to long
  separate(stat, into = c("var", "stat"), sep = "_")
```

```
## # A tibble: 6 x 3
##     var  stat          val
## * <chr> <chr>        <dbl>
## 1 carat  Mean    0.7979397
## 2 depth  Mean   61.7494049
## 3 price  Mean 3932.7997219
## 4 carat    SD    0.4740112
## 5 depth    SD    1.4326213
## 6 price    SD 3989.4397381
```

```
diamond.summarized %>% gather(stat, val) %>% #from wide to long
  separate(stat, into = c("var", "stat"), sep = "_") %>%
  spread(stat, val)
```

```
## # A tibble: 3 x 3
##     var         Mean           SD
## * <chr>        <dbl>        <dbl>
## 1 carat    0.7979397    0.4740112
## 2 depth   61.7494049    1.4326213
## 3 price 3932.7997219 3989.4397381
```

```
diamond.table
```

```
## # A tibble: 3 x 3
##     var         Mean           SD
## * <chr>        <dbl>        <dbl>
## 1 carat    0.7979397    0.4740112
## 2 depth   61.7494049    1.4326213
## 3 price 3932.7997219 3989.4397381
```