Example: Importing, Analyzing, and Plotting Data

R Workshop - 8/14/2018

Here is an example of importing a data set, performing a regression analysis, and creating a plot of the coefficients.

Importing the data

Let's work with the "R_Workshop_data.csv" file. Recall that the file contains 52 individuals and 4 variables. The variables are: respondents id ("id"), a binary variable indicating whether the respondent is male or female ("male" where "1" is male), a measure of the respondent's age ("age"), and a measure of risky behaviors engaged in by the respondent ("risky").

Let's go ahead and import it into R:

```
setwd("/...") #first, set the directory where the file is.
```

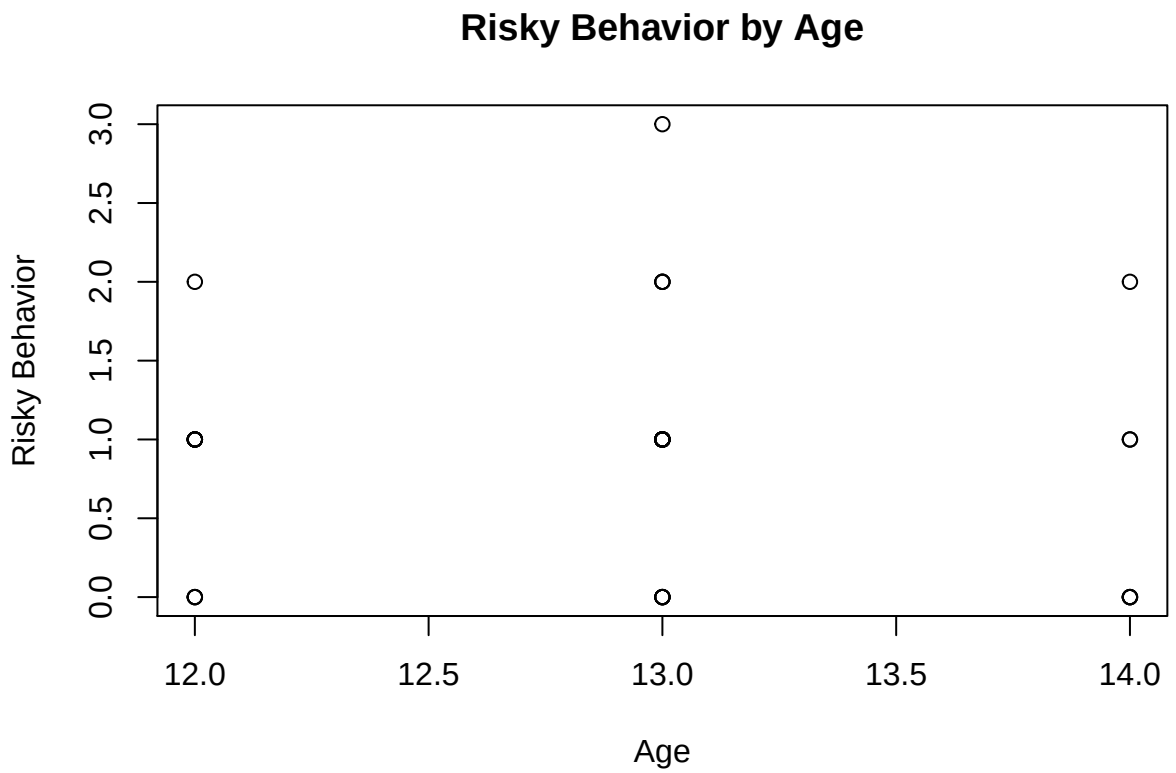
```
data <- read.csv(
  "https://www.jacobtnyoung.com/uploads/2/3/4/5/23459640/r_workshop_data.csv", #the url.
  header=TRUE, as.is=TRUE, na.strings="." #all the other arguments remain the same.
  )
data[1:20,] #look at the first 20 cases of the data.
```

```
##    id male age risky
## 1   1    0  13     1
## 2   2    1  NA     0
## 3   3    1  13     1
## 4   4   NA  12    NA
## 5   5    0  13     1
## 6   6    1  13    NA
## 7   7    1  12     1
## 8   8    1  12     1
## 9   9    0  12     1
## 10 10    0  13     3
## 11 11    0  13    NA
## 12 12    1  14     1
## 13 13    0  13     1
## 14 14    0  14     0
## 15 15   NA  12     1
## 16 16    1  13     0
## 17 17    0  13     1
## 18 18    1  14     0
## 19 19    0  14    NA
## 20 20    1  13     1
```

Analyzing the data

Suppose we are interesting if whether there is a linear relationship between a respondent's age and his/her risky behaviors. Let's plot the relationship between the variables to visually inspect them:

```
plot(
 data$age,                       # make age the x-axis.
 data$risky,                     # make risky behavior the y-axis.
 main = "Risky Behavior by Age", # set the title.
 xlab = "Age",                   # label the x-axis.
 ylab = "Risky Behavior"         # label the y-axis.
 )
```



Let's examine the correlation between the variables using the `cor()` function:

```
cor(data$age,data$risky) # this returns an error.
```

```
## [1] NA
```

Why do we get an error?

We can examine the missingness for each variable by using the `is.na()` function:

```
is.na(data$age)
```

```
##  [1] FALSE  TRUE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [12] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [23] FALSE FALSE FALSE FALSE FALSE  TRUE FALSE FALSE FALSE FALSE FALSE
## [34] FALSE FALSE FALSE FALSE  TRUE FALSE FALSE FALSE FALSE FALSE FALSE
## [45] FALSE FALSE FALSE FALSE FALSE FALSE  TRUE FALSE
```

```
is.na(data$risky)
```

```
##  [1] FALSE FALSE FALSE  TRUE FALSE  TRUE FALSE FALSE FALSE FALSE  TRUE
```

```
## [12] FALSE FALSE FALSE FALSE FALSE FALSE FALSE  TRUE FALSE FALSE FALSE
## [23] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE  TRUE FALSE FALSE
## [34] FALSE FALSE FALSE FALSE FALSE FALSE FALSE  TRUE FALSE FALSE FALSE
## [45] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
```

We can examine the missingness of a single variable by combining thre functions `is.na()`, `which()` and `length()`:

```
which(is.na(data$age)==TRUE) # which values are missing?
```

```
## [1]  2 28 38 51
```

```
length(which(is.na(data$age)==TRUE)) # how long is the vector of missing values from age?
```

```
## [1] 4
```

We can examine the missingness of both variables jointly by combining two functions `is.na()` and `table()`:

```
table(is.na(data$age),is.na(data$risky))
```

```
##
##         FALSE TRUE
##   FALSE    42    6
##   TRUE      4    0
```

Only take those cases that are complete by using the `use=` argument:

```
?cor
cor(data$age,data$risky, use="complete") # this does not return an error.
```

```
## [1] -0.0778548
```

We can estimate a linear regression model using the `lm()` function:

```
?lm
summary(lm(data$risky ~ data$age))
```

```
##
## Call:
## lm(formula = data$risky ~ data$age)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -0.9568 -0.7892  0.1270  0.1270  2.1270
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  1.96216    2.19190   0.895    0.376
## data$age    -0.08378    0.16964  -0.494    0.624
##
## Residual standard error: 0.7121 on 40 degrees of freedom
##   (10 observations deleted due to missingness)
## Multiple R-squared:  0.006061,   Adjusted R-squared:  -0.01879
## F-statistic: 0.2439 on 1 and 40 DF,  p-value: 0.6241
```

Let's make this model a bit more robust by adding male to the equation:

```
summary(lm(data$risky ~ data$age + data$male))
```

```
##
## Call:
## lm(formula = data$risky ~ data$age + data$male)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1.2618 -0.5590 -0.2045  0.3837  1.7955
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  1.94985    2.08145   0.937  0.35495
## data$age    -0.05734    0.16099  -0.356  0.72376
## data$male   -0.58817    0.20878  -2.817  0.00773 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.6587 on 37 degrees of freedom
##   (12 observations deleted due to missingness)
## Multiple R-squared:  0.1809, Adjusted R-squared:  0.1366
## F-statistic: 4.085 on 2 and 37 DF,  p-value: 0.02496
```

Plotting the data

Now we could create a plot of the estimate. Rather than manually entering the coefficients and standard errors, we can use the stored results. Since the estimates and standard errors are an object, we can just reference the particular values of the matrix we want in the plot. First, let's look at the results:

```
# make an object from the model.
 results <- summary(lm(data$risky ~ data$age + data$male))
 results
```

```
##
## Call:
## lm(formula = data$risky ~ data$age + data$male)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1.2618 -0.5590 -0.2045  0.3837  1.7955
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  1.94985    2.08145   0.937  0.35495
## data$age    -0.05734    0.16099  -0.356  0.72376
## data$male   -0.58817    0.20878  -2.817  0.00773 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.6587 on 37 degrees of freedom
##   (12 observations deleted due to missingness)
## Multiple R-squared:  0.1809, Adjusted R-squared:  0.1366
## F-statistic: 4.085 on 2 and 37 DF,  p-value: 0.02496
```

```r
class(results) # we see that it is a summary of an lm.
```

```
## [1] "summary.lm"
```

```r
names(results) # shows the names of the coefficents.
```

```
##  [1] "call"          "terms"         "residuals"     "coefficients"
##  [5] "aliased"       "sigma"         "df"            "r.squared"
##  [9] "adj.r.squared" "fstatistic"    "cov.unscaled"  "na.action"
```

```r
results$coefficients #gives just the 'coefficients' portion of the object 'results'.
```

```
##                Estimate Std. Error    t value    Pr(>|t|)
## (Intercept)  1.94985030  2.0814498  0.9367751 0.354948916
## data$age    -0.05733533  0.1609925 -0.3561366 0.723761046
## data$male   -0.58817365  0.2087760 -2.8172475 0.007726979
```

```r
#We can create the objects we need by referencing the matrix.
is.matrix(results$coefficients)
```

```
## [1] TRUE
```

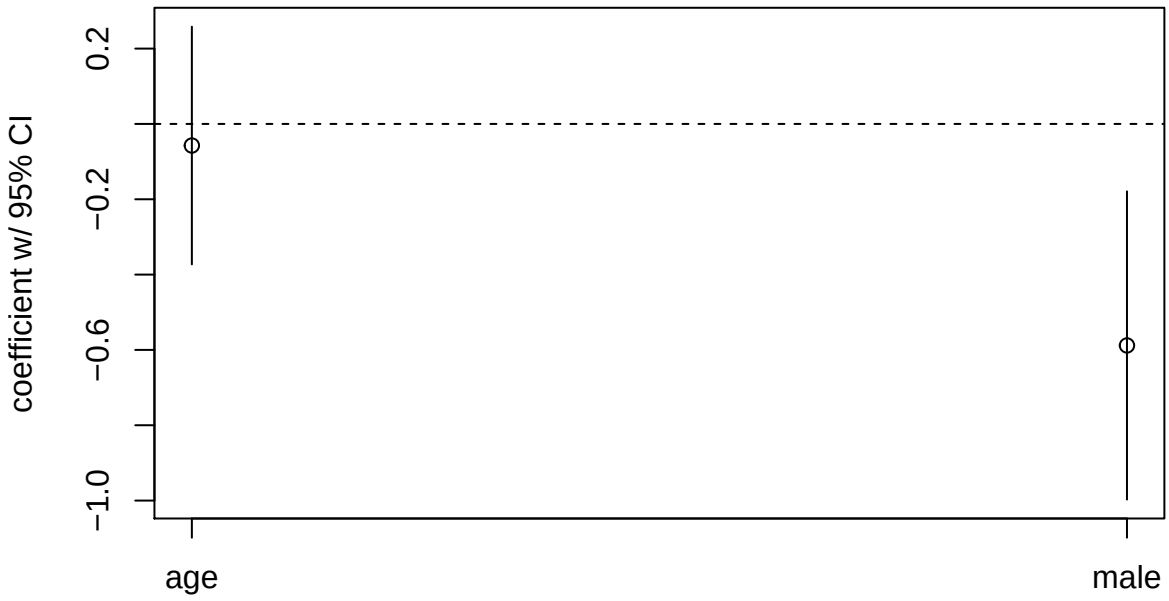
```r
point  <- c(results$coefficients[2,1],results$coefficients[3,1])
se     <- c(results$coefficients[2,2],results$coefficients[3,2])
upper.ci <- point+(1.96*se)
lower.ci <- point-(1.96*se)

#Now, we can plot:
n.coef = 2 # number of coefficients.
names = c("age","male") #coef names.

x.ax = seq(1,n.coef,length.out=n.coef) #dims of the x axis.
y.ax = seq(min(lower.ci),max(upper.ci),length.out=n.coef) #y axis.

plot(
 x.ax,
   y.ax,
   type="n",                      # do not plot anything yet.
   ylab="coefficient w/ 95% CI", # label for y axis.
   xlab="",                       # label for x axis .
   xaxt="n"                       # toggle x axis labels (for now).
)

points(x.ax,point) # plot the point estimates.
segments(x.ax,upper.ci,x.ax,lower.ci) #now the intervals.
abline(h=0,lty=2)  # add a line at zero.
axis(side=1,at=x.ax,labels=names) # add coefficient labels.
```



We can do a better job with the texreg package:

```
library("texreg")
```

```
## Version:  1.36.23
## Date:     2017-03-03
## Author:   Philip Leifeld (University of Glasgow)
##
## Please cite the JSS article in your publications -- see citation("texreg").
```

```
help(package="texreg")
plotreg(lm(data$risky ~ data$age + data$male))
```

```
## Model 1: bars denote 0.5 (inner) resp. 0.95 (outer) confidence intervals (computed from standard err
```