SRC R Session 2

Yiqing Xu

Resume Experiment

Preparation

```
## clean up memory, check/set working directory as necessary
rm(list = ls())
getwd()
```

```
## [1] "/Users/xyq/Library/CloudStorage/Dropbox/Teaching/_Undergraduate/R_sessions/session2"
```

```
dir()
```

```
## [1] "callback.pdf"  "resume.csv"    "session2_ex.R" "session2.pdf"
## [5] "session2.R"    "session2.Rmd"
```

Loading and examining the data

```
## load dataset
d <- read.csv("resume.csv")

## take a look at the dataset
head(d)
```

```
##   firstname    sex  race call
## 1   Allison female white    0
## 2   Kristen female white    0
## 3   Lakisha female black    0
## 4   Latonya female black    0
## 5    Carrie female white    0
## 6       Jay   male white    0
```

```
colnames(d)
```

```
## [1] "firstname" "sex"       "race"      "call"
```

```
summary(d)
```

```
##   firstname             sex                race                call
##  Length:4870        Length:4870        Length:4870        Min.   :0.00000
##  Class :character   Class :character   Class :character   1st Qu.:0.00000
##  Mode  :character   Mode  :character   Mode  :character   Median :0.00000
##                                                           Mean   :0.08049
##                                                           3rd Qu.:0.00000
##                                                           Max.   :1.00000
```

```
unique(d$firstname) # unique first names in the data
```

```
##  [1] "Allison"  "Kristen"  "Lakisha"  "Latonya"  "Carrie"   "Jay"
```

```
##  [7] "Jill"     "Kenya"    "Tyrone"   "Aisha"    "Geoffrey" "Matthew"
## [13] "Tamika"   "Leroy"    "Todd"     "Greg"     "Keisha"   "Brad"
## [19] "Laurie"   "Meredith" "Anne"     "Emily"    "Latoya"   "Ebony"
## [25] "Brendan"  "Hakim"    "Jamal"    "Neil"     "Tremayne" "Brett"
## [31] "Darnell"  "Sarah"    "Jermaine" "Tanisha"  "Rasheed"  "Kareem"
```

```
table(d$race)       # counts of race group for the names
```

```
##
## black white
##  2435  2435
```

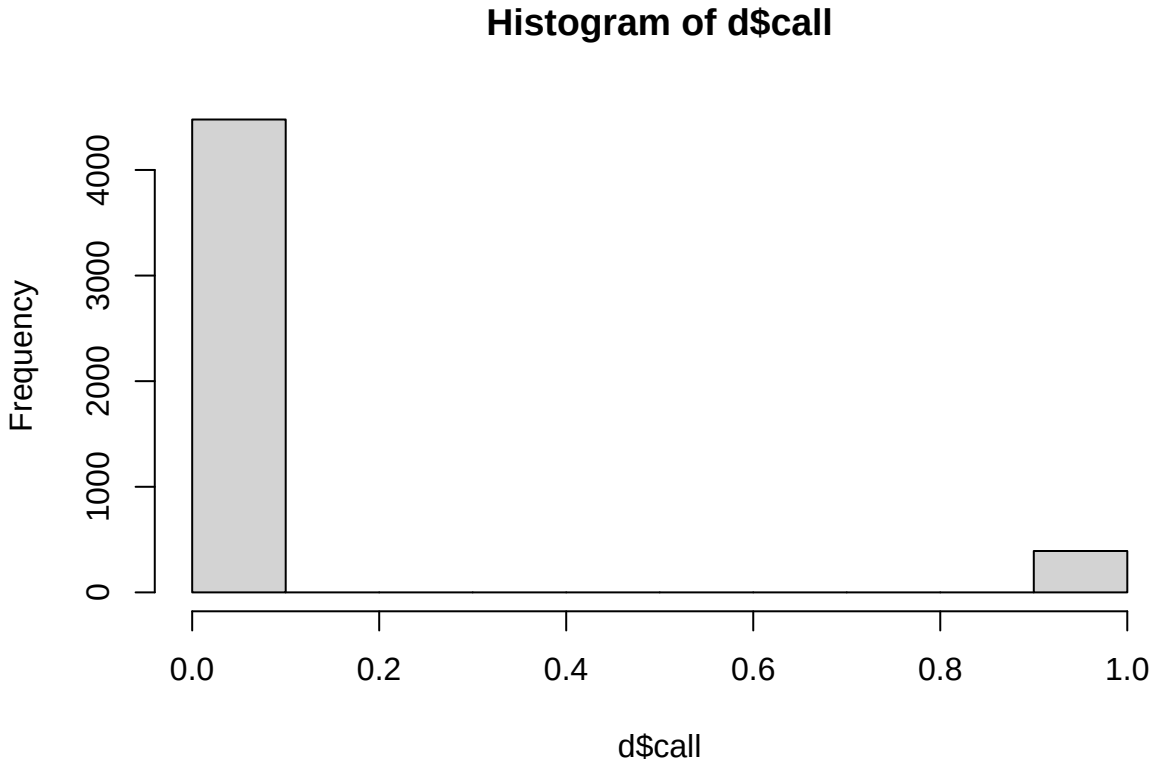
```
table(d$call)       # counts of whether or not there was a callback
```

```
##
##    0    1
## 4478  392
```

```
unique(d$firstname) # vector of unique first names in data
```

```
##  [1] "Allison"  "Kristen"  "Lakisha"  "Latonya"  "Carrie"   "Jay"
##  [7] "Jill"     "Kenya"    "Tyrone"   "Aisha"    "Geoffrey" "Matthew"
## [13] "Tamika"   "Leroy"    "Todd"     "Greg"     "Keisha"   "Brad"
## [19] "Laurie"   "Meredith" "Anne"     "Emily"    "Latoya"   "Ebony"
## [25] "Brendan"  "Hakim"    "Jamal"    "Neil"     "Tremayne" "Brett"
## [31] "Darnell"  "Sarah"    "Jermaine" "Tanisha"  "Rasheed"  "Kareem"
```

```
## draw the distribution of the callback rate
hist(d$call)
```



```r
## how many unique first names are there in the dataset?
length(unique(d$firstname))
```

```
## [1] 36
```

```r
## how many of these names are 'black' names and 'white' names, respectively?
length(unique(d$firstname[which(d$race == "white")]))
```

```
## [1] 18
```

```r
length(unique(d$firstname[which(d$race == "black")]))
```

```
## [1] 18
```

```r
## how many of them are black names and white names, respectively?
length(unique(d$firstname[which(d$race == "black")]))
```

```
## [1] 18
```

```r
length(unique(d$firstname[which(d$race == "white")]))
```

```
## [1] 18
```

```r
## average call rates for blacks and whites
call.wh <- mean(d$call[d$race=="white"])
call.bl <- mean(d$call[d$race=="black"])
```

```r
# calculate how much lower black callback rate is compared to white callback rate
(call.wh - call.bl)/call.wh
```

```
## [1] 0.3319149
```

```r
## what's the average call back rate for each first name
# first, get all first names
names.wh <- unique(d[d$race=="white", "firstname"])
names.bl <- unique(d[d$race=="black", "firstname"])
```

```r
# get the mean for a particular name
this.name <- names.bl[1] # first element of the black-sounding names vector
mean(d$call[d$firstname == this.name])
```

```
## [1] 0.055
```

Using a for loop to analyze the data

We can use a for loop to calculate the average callback rate for each unique name in the data. In general, a for loop is useful in any situation where you want to repeat the same operation over a collection of elements.

```r
## Step 1: Create a vector of elements over which the for loop will iterate
## In this case, we want a vector of unique first names for each subgroup
names.bl <- unique(d$firstname[which(d$race=="black")])
names.wh <- unique(d$firstname[which(d$race=="white")])
```

```r
# print the first five elements of each vector to check
names.bl[1:5]
```

```
## [1] "Lakisha" "Latonya" "Kenya"   "Tyrone"  "Aisha"
```

```r
names.wh[c(1,2,3,4,5)]
```

```
## [1] "Allison" "Kristen" "Carrie"  "Jay"     "Jill"
```

```r
## Step 2: Create an empty 'container object' to save the values of interest
## In this case, we want to save the average callback rate for each name
```

```r
# creates vector of 'NA's equal in length to the vectors we created in step 1
call.bl <- rep(NA, length(names.bl))
call.wh <- rep(NA, length(names.wh))
```

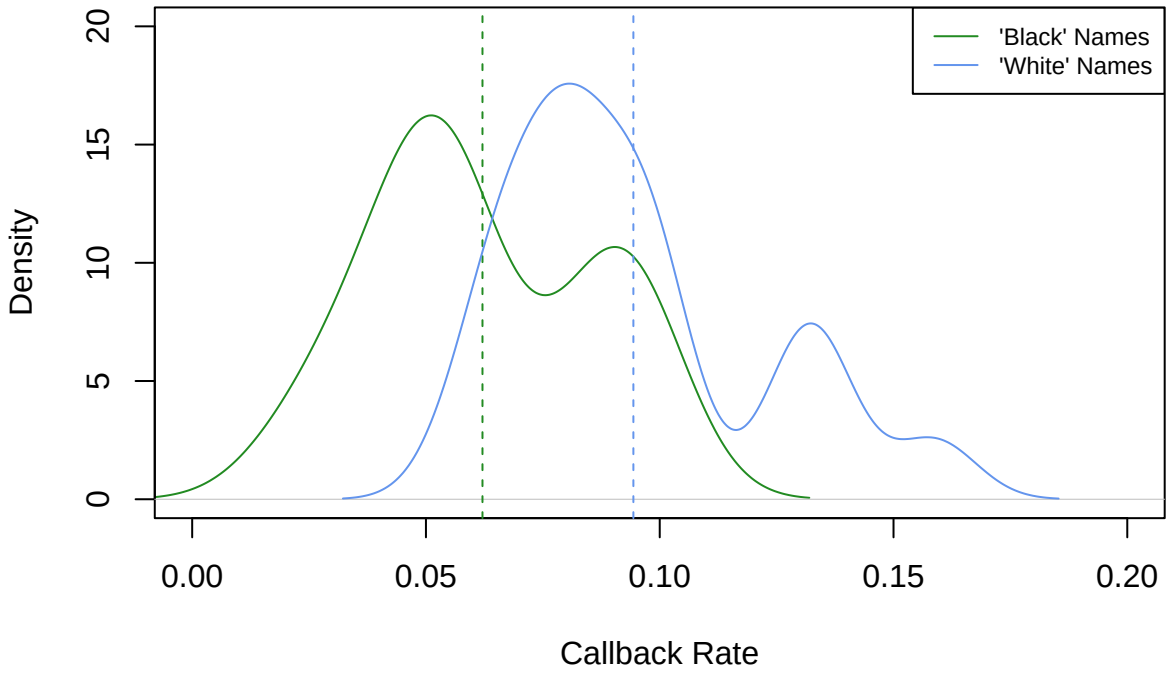
```r
## Step 3: Specify and run the for loop

## check what a for loop does
# counter variable 'i' will run from 1, 2, ... , 20
for (i in 1:20){
  print(i) # iterates over every value of 'i'
}
```

```
## [1] 1
## [1] 2
## [1] 3
## [1] 4
## [1] 5
## [1] 6
## [1] 7
## [1] 8
## [1] 9
## [1] 10
## [1] 11
## [1] 12
## [1] 13
## [1] 14
## [1] 15
## [1] 16
## [1] 17
## [1] 18
## [1] 19
## [1] 20
```

```r
## this is some code we will run inside the for loop
## we will keep it here for debugging
# code for debugging outside of the loop (commented out here)
# i <- 1
# names.bl[i]

## now we want to iterate from 1 to the length of names.bl vector
## to get every name in the names.bl vector
# counter variable 'i' will run from 1, 2, ... , length(names.bl)
for (i in 1:length(names.bl)) {
  this.name  <- names.bl[i] # what is the i-th name in names.bl
  name.mean  <- mean(d$call[d$firstname==this.name]) # avg callback rate for that name
  call.bl[i] <- name.mean   # save avg callback rate in i-th index of container
```

```
}

# can also simplify the above to one line of code
for (i in 1:length(names.wh)) {
  call.wh[i] <- mean(d$call[d$firstname==names.wh[i]])
}

## same task for white-sounding names
for (i in 1:length(names.wh)){
  # take the ith element of 'names.wh'
  this.name <- names.wh[i]
  # store the mean of callbacks for this.name as the ith element of
  # call.wh vector
  call.wh[i] <- mean(d$call[d$firstname == this.name])
}
```

```
# Compare the distributions of average call back rate for black and white names

## Include the first and last lines in this chunk of code if you want
## to save the plot as a PDF file in the current working directory.
## Can also save as bmp(), jpeg(), or png() files (see help manual).

# pdf("plotname.pdf", height = 4, width = 6) [include this line to save plot]
plot(density(call.bl), col="forestgreen",
     xlim=c(0, 0.2), ylim=c(0, 20),
     xlab="Callback Rate", main="Callback Rate by Race")
abline(v=mean(call.bl), lty=2, col="forestgreen")
lines(density(call.wh), col="cornflowerblue")
abline(v=mean(call.wh), lty=2, col="cornflowerblue")
legend("topright", legend=c("'Black' Names", "'White' Names"),
       col=c("forestgreen", "cornflowerblue"), lwd=1, cex=0.75)
```




```
# graphics.off() [include this line to save plot]
```

```
# get the list of unique first names
firstnames <- unique(d$firstname)

# calculate its number of elements
n <- length(unique(d$firstname))


### Finding call back rates for black sounding names

# step 1: create storage
call.rate <- rep(NA, n) # this is a vector of length n; all elements are NAs

# step 2: loop

## we first do a trial run with the first element ("Allison")

i <- 1 # create an index

# subset the data to all rows whose first names are Allison, calcuate the mean call-back rate
mean(d$call[which(d$firstname == firstnames[i])])
```

```
## [1] 0.09482759
```

```
# check whether this is the same as writing Allison directly
mean(d$call[which(d$firstname == "Allison")])
```

```
## [1] 0.09482759
```

```r
# yes it is! success!

# now we can write a for loop
for (i in 1:n) { # let i run from 1 to n
  # the ith element of the vector call.rate is the average callback rate for the ith first name from th
  call.rate[i] <- mean(d$call[which(d$firstname == firstnames[i])])
}

# check this vector
call.rate
```

```
##  [1] 0.09482759 0.13145540 0.05500000 0.09130435 0.13095238 0.13432836
##  [7] 0.08374384 0.08673469 0.05333333 0.02222222 0.06779661 0.08955224
## [13] 0.05468750 0.09375000 0.05882353 0.07843137 0.03825137 0.15873016
## [19] 0.09743590 0.10160428 0.08264463 0.07929515 0.08407080 0.09615385
## [25] 0.07692308 0.05454545 0.06557377 0.06578947 0.04347826 0.06779661
## [31] 0.04761905 0.09844560 0.09615385 0.05797101 0.02985075 0.04687500
```

```r
# we can create a dataframe combining the call rate vector and the first names

cbind.data.frame(firstnames, call.rate)
```

```
##    firstnames  call.rate
## 1     Allison 0.09482759
## 2     Kristen 0.13145540
## 3     Lakisha 0.05500000
## 4     Latonya 0.09130435
## 5      Carrie 0.13095238
## 6         Jay 0.13432836
## 7        Jill 0.08374384
## 8       Kenya 0.08673469
## 9      Tyrone 0.05333333
## 10      Aisha 0.02222222
## 11   Geoffrey 0.06779661
## 12    Matthew 0.08955224
## 13     Tamika 0.05468750
## 14      Leroy 0.09375000
## 15       Todd 0.05882353
## 16       Greg 0.07843137
## 17     Keisha 0.03825137
## 18       Brad 0.15873016
## 19     Laurie 0.09743590
## 20   Meredith 0.10160428
## 21       Anne 0.08264463
## 22      Emily 0.07929515
## 23     Latoya 0.08407080
## 24      Ebony 0.09615385
## 25    Brendan 0.07692308
## 26      Hakim 0.05454545
## 27      Jamal 0.06557377
## 28       Neil 0.06578947
## 29   Tremayne 0.04347826
## 30      Brett 0.06779661
## 31    Darnell 0.04761905
## 32      Sarah 0.09844560
```

```
## 33   Jermaine 0.09615385
## 34    Tanisha 0.05797101
## 35    Rasheed 0.02985075
## 36     Kareem 0.04687500
```

```
# or we can give names to elements of this callback rate vector
names(call.rate) <- firstnames
```