7/6/2019 read names.R

```
source("lib/utils.R")
 1
 2
 3
   # load the election results
 4 results <- read.csv("data/results.csv", colClasses=c(rep("character", 3),
    rep("numeric", 3)))
   results$OTHER <- results$TOTAL - (results$CLINTON + results$TRUMP)
 6
   results$TRUMP_PERCENT <- 100 * results$TRUMP / results$TOTAL
 7
   results$CLINTON PERCENT <- 100 * results$CLINTON / results$TOTAL
 8
   results$TRUMP MARGIN <- results$TRUMP PERCENT - results$CLINTON PERCENT
   results$CLINTON_MARGIN <- results$CLINTON_PERCENT - results$TRUMP_PERCENT
 9
   results$MARGIN
                            <- abs(results$CLINTON MARGIN)</pre>
10
11
12 | state abbrs <- results$ABBR
13
   f <- function(abbr) {</pre>
14
15
      print(paste("Reading", abbr))
16
      read_state(abbr, 2015)
17
18
19
    names <- do.call(rbind, lapply(state_abbrs, f))</pre>
20
   # If you're on the right OS, you can run this command line to make sure we got every
21
22
    system("grep -o ',2015,' data/states/*.TXT | wc -l")
23
    names$HANDLE <- paste(names$NAME, "_", names$GENDER, sep="")</pre>
24
25
   # tally state counts--number of states a name appears in--for each name+gender, and
26
    join them
    name counts <- as.data.frame(table(names$HANDLE))</pre>
27
    colnames(name counts) <- c("HANDLE", "STATE COUNT")</pre>
28
29
30
   names <- merge(names, name counts, by="HANDLE")</pre>
31
32
   # sort by state rank, with tie going to state with more of that name
    sorted <- names[order(names$GENDER, names$HANDLE, names$RANK, -names$VALUE),]</pre>
33
34
35
   # reduce to those in 10+ states
   top_10_roster <- unique(names$HANDLE[names$STATE_COUNT >= 10])
36
37
   top_10_names <- subset(sorted, sorted$STATE_COUNT >= 10)
38
39
   # for each name, rank which states are highest now that we've sorted them
40
   top 10 names$RANK N <- 0
41
   for (handle in top 10 roster) {
      top_10_names$RANK_N[top_10_names$HANDLE == handle] <-</pre>
42
    seq(1,NROW(top 10 names$RANK[top 10 names$HANDLE == handle]))
43
    }
44
45
   # and filter down to top ten
   filtered <- subset(top_10_names, top_10_names$RANK_N <= 10)</pre>
46
47
48 # join names with political results
   filtered <- merge(filtered, results, by="ABBR")</pre>
49
50 # and re-sort because merges always mess that up
   filtered <- filtered[order(filtered$GENDER, filtered$HANDLE, filtered$RANK N),]</pre>
51
52
```

7/6/2019 read names.R

```
# Thx, http://stackoverflow.com/questions/3443687/formatting-decimal-places-in-r
 53
 54
    specify decimal <- function(x, k) format(round(x, k), nsmall=k)</pre>
 55
    # reduce long decimals for csv files
 56
 57
    cleaned <- filtered</pre>
    cleaned$TRUMP_PERCENT <- specify_decimal(filtered$TRUMP_PERCENT, 2)</pre>
 58
 59 cleaned$CLINTON PERCENT <- specify decimal(filtered$CLINTON PERCENT, 2)
 60 cleaned$TRUMP_MARGIN
                              <- specify_decimal(filtered$TRUMP_MARGIN, 2)</pre>
    cleaned$CLINTON_MARGIN <- specify_decimal(filtered$CLINTON_MARGIN, 2)</pre>
 61
 62 cleaned$MARGIN
                              <- specify decimal(filtered$MARGIN, 2)</pre>
 63
 64 # eyeball this
 65 for (i in 1:100) {
      for (c in 15:19) {
 66
 67
         print(paste(paste(filtered[i,c], collapse=" "), paste(cleaned[i,c], collapse=" "),
     sep=" -- "))
 68
       }
 69
    }
 70
    # let's put these values in the baby box
 71
    write.csv(cleaned, "csv/all_names.csv", row.names=FALSE)
 72
 73
 74
    f <- function(handle) {</pre>
 75
       print(handle)
 76
       write.csv(subset(cleaned, cleaned$HANDLE==handle), paste("csv/names/", handle,
     ".csv", sep=""), row.names=FALSE)
 77
    }
 78
 79
    # write csvs for every name
    lapply(top_10_roster, f)
 80
 81
 82
    # total votes for each name among it's top-ten states
    names_trump <- aggregate(TRUMP ~ HANDLE, FUN=sum, data=filtered)</pre>
 83
    names clinton <- aggregate(CLINTON ~ HANDLE, FUN=sum, data=filtered)</pre>
 84
    names other <- aggregate(OTHER ~ HANDLE, FUN=sum, data=filtered)
 85
 86
    names_total <- aggregate(TOTAL ~ HANDLE, FUN=sum, data=filtered)</pre>
 87
 88
    # add to roster
 89
    roster <- merge(names trump, names clinton, by="HANDLE")</pre>
    roster <- merge(roster, names_other, by="HANDLE")</pre>
90
 91
    roster <- merge(roster, names total, by="HANDLE")</pre>
92
 93
    roster$TRUMP_PERCENT <- 100 * roster$TRUMP</pre>
                                                       / roster$TOTAL
 94
    roster$CLINTON PERCENT <- 100 * roster$CLINTON / roster$TOTAL
                           <- roster$TRUMP_PERCENT - roster$CLINTON_PERCENT</pre>
95
    roster$SPLIT
96
    # join with some of the original info that didn't survive aggregation
97
    info <- top_10_names[,c("HANDLE", "NAME", "GENDER", "STATE_COUNT")]</pre>
98
99
    info <- info[!duplicated(info), ]</pre>
100
    roster <- merge(roster, info, by="HANDLE")</pre>
101
102
    # add nat'l data
103
    national names <- read national(2015)</pre>
104
    national_names$HANDLE <- paste(national_names$NAME, "_", national_names$GENDER, sep="")</pre>
105
106
107
    # tally state counts
```

7/6/2019 read_names.R

```
roster <- merge(roster, national_names[,c("HANDLE", "VALUE", "RANK")], by="HANDLE" )</pre>
108
    roster$WINNER <- ""</pre>
109
110 roster$WINNER[roster$TRUMP < roster$CLINTON] <- "D"
111
    roster$WINNER[roster$TRUMP > roster$CLINTON] <- "R"</pre>
112
113
    roster$D_COUNT <- 0</pre>
    roster$R_COUNT <- 0
114
115
    for (i in 1:NROW(roster)) {
116
117
       handle <- roster[i,]$HANDLE
118
       print(handle)
119
       top_10 <- subset(filtered, filtered$HANDLE == handle)</pre>
120
       roster[i,]$D_COUNT <- NROW(subset(top_10, top_10$CLINTON > top_10$TRUMP))
       roster[i,]$R_COUNT <- NROW(subset(top_10, top_10$CLINTON < top_10$TRUMP))</pre>
121
    }
122
123
124 # we're done!
    write.csv(roster[,c(9:16,1:8)], "csv/roster.csv", row.names=FALSE)
125
```