Chaudhary_week_03

Jyoti Chaudhary

October 2, 2016

a. Create a data frame of players who played at least 200 games in their career according to the Fielding data frame. You'll have to group by player id, sum over the variable G, filter, and then do some sort of join with the Master data frame.

```
players <- group_by(Fielding, playerID)
players <- mutate(players, G_sum = sum(G))
players <- select(players, playerID, G_sum)

players.full <- left_join(unique(players), Master)
```

```
## Joining, by = "playerID"
```

```
head(players.full)
```

```
## # A tibble: 6 x 27
## # Groups:   playerID [6]
##   playerID  G_sum birthYear birthMonth birthDay birthCountry birthState
##   <chr>     <int>     <int>      <int>    <int> <chr>        <chr>
## 1 abercda01     1      1850          1        2 USA          OK
## 2 addybo01    276      1842          2       NA CAN          ON
## 3 allisar01   174      1849          1       29 USA          PA
## 4 allisdo01   344      1846          7       12 USA          PA
## 5 ansonca01  2592      1852          4       17 USA          IA
## 6 armstbo01    12      1850         NA       NA USA          MD
## # ... with 20 more variables: birthCity <chr>, deathYear <int>,
## #   deathMonth <int>, deathDay <int>, deathCountry <chr>,
## #   deathState <chr>, deathCity <chr>, nameFirst <chr>, nameLast <chr>,
## #   nameGiven <chr>, weight <int>, height <int>, bats <fct>, throws <fct>,
## #   debut <chr>, finalGame <chr>, retroID <chr>, bbrefID <chr>,
## #   deathDate <date>, birthDate <date>
```

b. Create a data frame similar to the babynames, but based on your data frame in (a). Use the variables nameFirst and birthYear.

```
players.full <- group_by(players.full, nameFirst, birthYear)
babynames.new <- select(players.full, nameFirst, birthYear)
babynames.new <- arrange(babynames.new, nameFirst, birthYear)
```

```
## Warning: package 'bindrcpp' was built under R version 3.4.4
```

```
head(babynames.new)
```

```
## # A tibble: 6 x 2
## # Groups:   nameFirst, birthYear [6]
##   nameFirst birthYear
##   <chr>         <int>
## 1 A. J.          1965
## 2 A. J.          1974
## 3 A. J.          1976
## 4 A. J.          1977
## 5 A. J.          1981
## 6 A. J.          1982
```

c. Combine the babynames data frame, restricted to male babies, and the one that you created in (b).

```
babynames.male <- filter(babynames, sex == "M")
babynames.male <- select(babynames.male, year, name, n, prop)
babynames.combined <- full_join(babynames.new, babynames.male,
                      by = c("nameFirst" = "name", "birthYear" = "year"))
head(babynames.combined)
```

```
## # A tibble: 6 x 4
## # Groups:   nameFirst, birthYear [6]
##   nameFirst birthYear     n  prop
##   <chr>         <dbl> <int> <dbl>
## 1 A. J.         1965.    NA    NA
## 2 A. J.         1974.    NA    NA
## 3 A. J.         1976.    NA    NA
## 4 A. J.         1977.    NA    NA
## 5 A. J.         1981.    NA    NA
## 6 A. J.         1982.    NA    NA
```

d. Determine the 5 most popular names for male babies from the babynames dataset and the 5 most popular names for baseball players, based on your dataset in (b). Do this by pooling all the names from 1890 to 1990—that is, find 10 names total, not 10 names per year. The total might actually be less than 10 if there is overlap in the names.

```
babynames.m <- filter(babynames, year>= 1890 & year <= 1990, sex == "M")
babynames.m <- select(babynames.m, name, n)
babynames.m <- group_by(babynames.m, name)
babynames.m <- mutate(babynames.m, sum = sum(n))
babynames.m <- select(babynames.m, name, sum)
babynames.m <- arrange(unique(babynames.m), desc(sum))
head(babynames.m)
```

```
## # A tibble: 6 x 2
## # Groups:   name [6]
##   name        sum
##   <chr>     <int>
## 1 James   4608320
## 2 John    4569150
## 3 Robert  4455132
## 4 Michael 3577553
## 5 William 3499276
## 6 David   3106604
```

```
babynames.n <- count(babynames.new)
babynames.n <- group_by(babynames.n, nameFirst)
babynames.n <- mutate(babynames.n, sum = sum(n))
babynames.n <- group_by(babynames.n, nameFirst)
babynames.n <- select(babynames.n, nameFirst, sum)
babynames.n <- arrange(unique(babynames.n), desc(sum))
head(babynames.n)
```

```
## # A tibble: 6 x 2
## # Groups:   nameFirst [6]
##   nameFirst   sum
##   <chr>     <int>
## 1 Bill        533
## 2 John        472
## 3 Mike        433
## 4 Jim         427
## 5 Joe         387
## 6 Bob         335
```

e. If you plot a name in the general population (i.e., from babynames) against baseball player names, the difference in scale will make it hard to interpret. For both general population and baseball names, create a new variables for each: the proportion of all names from that year equal to that name (e.g., if 2% of all babies in 1961 were names “Steven”, this new variable would equal 0.02 for Steven for 1961).

```
babynames.d <- group_by(babynames, year)
babynames.d <- filter(babynames.d, sex == "M")
head(babynames.d)
```

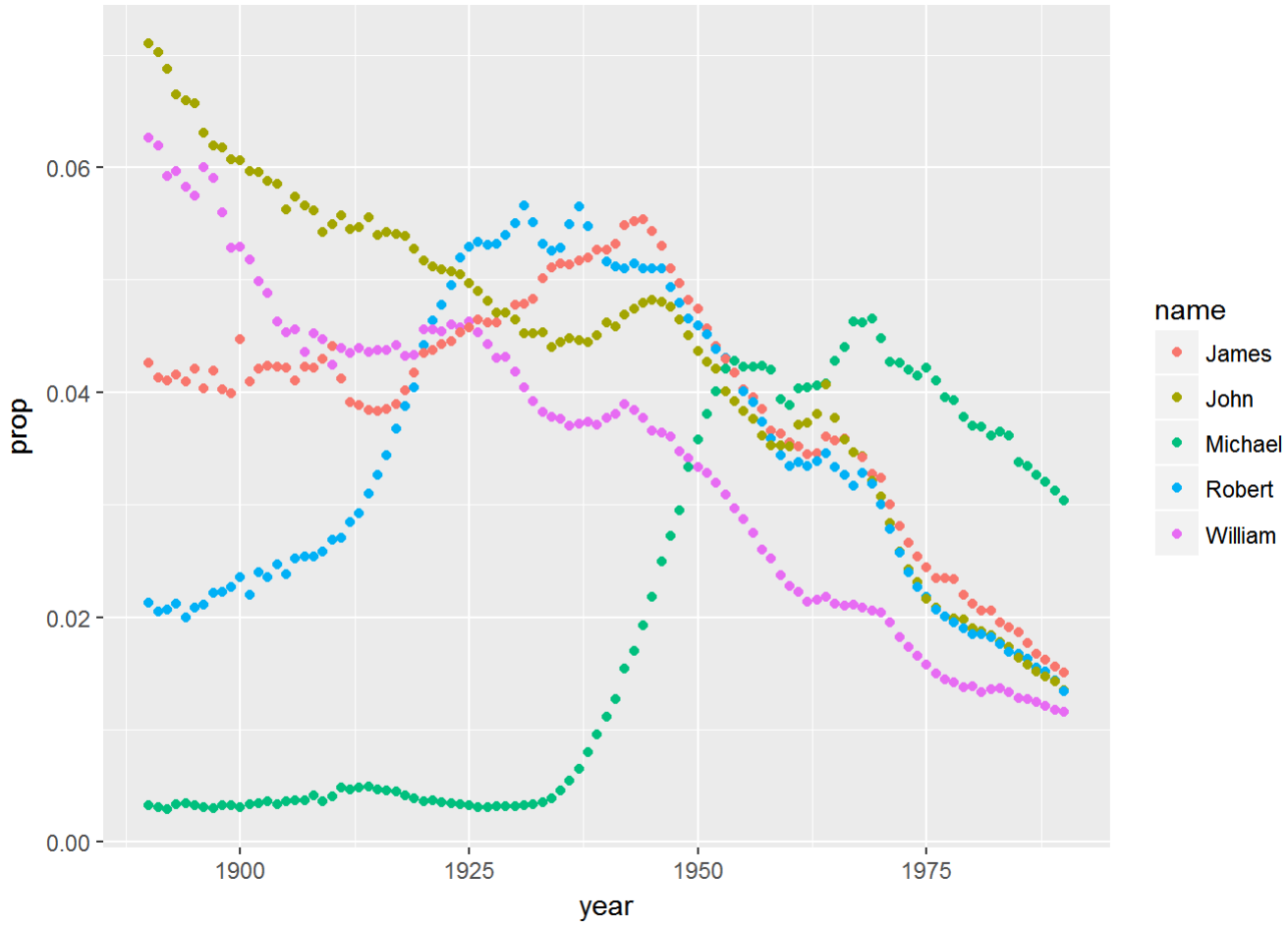
```
## # A tibble: 6 x 5
## # Groups:   year [1]
##    year sex   name        n   prop
##   <dbl> <chr> <chr>   <int>  <dbl>
## 1 1880. M     John     9655 0.0815
## 2 1880. M     William  9531 0.0805
## 3 1880. M     James    5927 0.0501
## 4 1880. M     Charles  5348 0.0452
## 5 1880. M     George   5126 0.0433
## 6 1880. M     Frank    3242 0.0274
```

```
babynames.p <- count(babynames.new)
babynames.p <- group_by(babynames.p, birthYear)
babynames.p <- mutate(babynames.p, sum = sum(n))
babynames.p <- mutate(babynames.p, prop = (n/sum))
head(babynames.p)
```

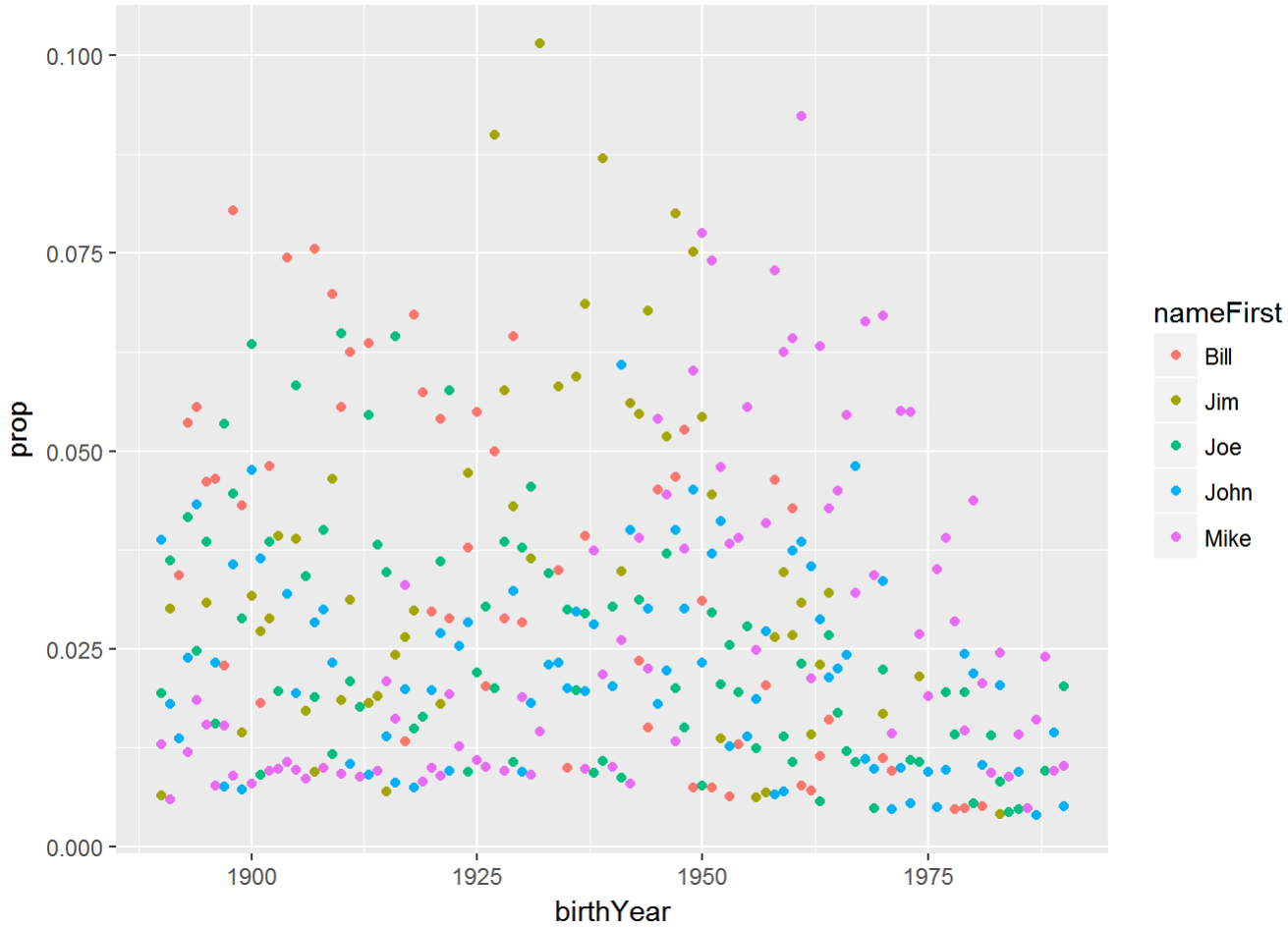
```
## # A tibble: 6 x 5
## # Groups:   birthYear [6]
##   nameFirst birthYear     n   sum    prop
##   <chr>         <int> <int> <int>   <dbl>
## 1 A. J.          1965     1   178 0.00562
## 2 A. J.          1974     1   186 0.00538
## 3 A. J.          1976     1   200 0.00500
## 4 A. J.          1977     1   205 0.00488
## 5 A. J.          1981     1   194 0.00515
## 6 A. J.          1982     1   213 0.00469
```

f. For each of the names you determined in (d), plot the relative popularity, using the variable you created in (e). Each figure should have different colors for general population names and for baseball player names. The horizontal axis should be year of birth, from 1890 to 1990.

```
###general names
ggplot(data = filter(babynames, sex == "M", year >=1890 & year <=1990,
                                  name == "James" |
                                  name == "John" |
                                  name == "Robert" |
                                  name =="Michael"|
                                  name == "William")) +
                                  geom_point(aes(year,prop, colour = name))
```




```
###baseball player names
ggplot(data = filter(babynames.p, birthYear >=1890 & birthYear <=1990,
                                  nameFirst == "Bill" |
                                  nameFirst == "Joe" |
                                  nameFirst == "John" |
                                  nameFirst =="Mike"|
                                  nameFirst == "Jim")) +
                                  geom_point(aes(birthYear,prop, colour = nameFirst))
```



2). Restaurant Inspection

- Starting with the restaurant health inspection dataset, create a new one that has a single observation for each inspection date. The observation should include all of the descriptive information about the restaurant, but from the many inspection- and violation-specific variables, please include only the mean score for that day. Now create a new data frame that for each restaurant (uniquely identifies by the CAMIS variable) has only the date and score of the most recent inspection. Please plot boxplots of scores for each boro (use “aes(BORO, SCORE)” and “geom_boxplot()”).

```
library(plyr)
```

```
## Warning: package 'plyr' was built under R version 3.4.4
```

```
## ------------------------------------------------------------------------------
```

```
## You have loaded plyr after dplyr - this is likely to cause problems.
## If you need functions from both plyr and dplyr, please load plyr first, then dplyr:
## library(plyr); library(dplyr)
```

```
## ------------------------------------------------------------------------------
```

```
##
## Attaching package: 'plyr'
```

```
## The following objects are masked from 'package:dplyr':
##
##     arrange, count, desc, failwith, id, mutate, rename, summarise,
##     summarize
```

```
## The following object is masked from 'package:purrr':
##
##     compact
```

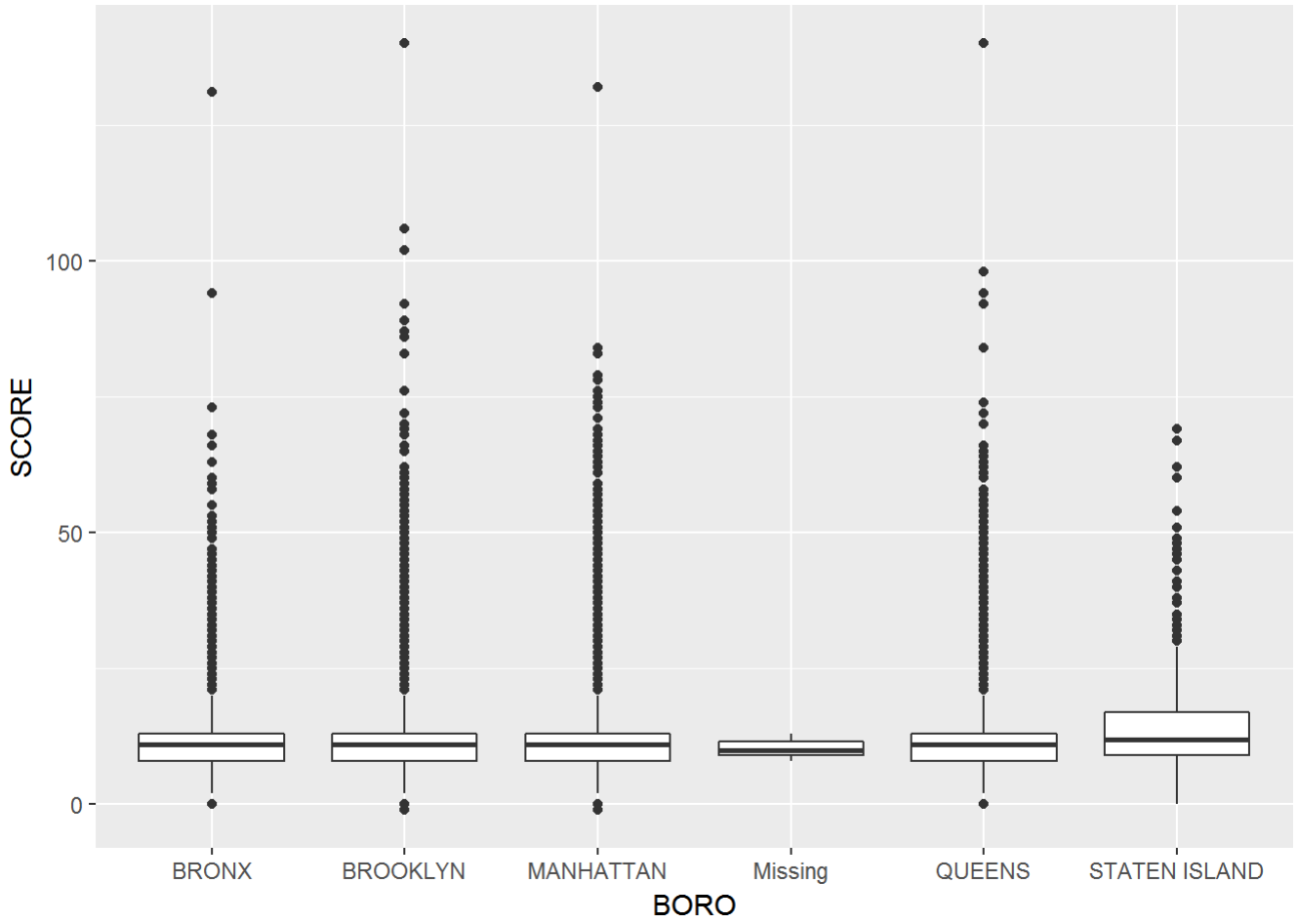
```
names(restaurant.df) <- make.names(names(restaurant.df))
restaurant.df <- group_by(restaurant.df, INSPECTION.DATE, DBA)
restaurant.df <- mutate(restaurant.df, mean = mean(SCORE))

restaurant.sp <- select(restaurant.df, DBA, INSPECTION.DATE)
restaurant.sp <- ddply(restaurant.sp,"DBA", summarise, INSPECTION.DATE = max(INSPECTION.DATE))
comb <- inner_join(restaurant.sp, restaurant.df, by= c("DBA", "INSPECTION.DATE"))

comb <- select(comb, DBA, INSPECTION.DATE, BORO, BUILDING,STREET, ZIPCODE,
               PHONE, CUISINE.DESCRIPTION, SCORE)
comb <- unique(comb)

ggplot(data= comb)+geom_boxplot(mapping = aes(BORO, SCORE))
```

```
## Warning: Removed 3352 rows containing non-finite values (stat_boxplot).
```



3). Legally-Operating-Businesses

- The data at `https://nycopendata.socrata.com/Business/Legally-Operating-Businesses/w7w3-xahh` (you'll have to figure out how to export it) does not include restaurants. Using the "Contact Phone Number", see if there are any businesses that share a phone number with restaurants in the dataset you created in (2). Because of formatting and other issues, you may not find every possible match; do not worry about that for now. For the matches that you find, use the functions sort and table to find the top ten business license categories where the business shares a phone number or address with a restaurant.

```
names(legal) <- make.names(names(legal))
comb<- group_by(comb, PHONE)
match <- inner_join(comb, legal, by=c("PHONE" = "Contact.Phone.Number"))
head(match)
```

```
## # A tibble: 6 x 25
## # Groups:   PHONE [6]
##   DBA         INSPECTION.DATE BORO    BUILDING STREET       ZIPCODE PHONE
##   <chr>       <chr>           <chr>   <chr>    <chr>          <int> <chr>
## 1 101 DELI    10/13/2016      QUEENS  10016    ROCKAWAY BE~   11694 71863~
## 2 107 WEST R~ 05/25/2017      MANHAT~ 2787     BROADWAY       10025 21286~
## 3 10TH AVENU~ 11/05/2015      MANHAT~ 156      10 AVENUE      10011 21292~
## 4 111 RESTAU~ 11/01/2017      QUEENS  0        CENTRAL TAX~   11430 71899~
## 5 1803        10/23/2017      MANHAT~ 78       READE ST       10007 21226~
## 6 1ST AVE GO~ 09/19/2017      MANHAT~ 1274     1ST AVE        10065 21247~
## # ... with 18 more variables: CUISINE.DESCRIPTION <chr>, SCORE <int>,
## #   DCA.License.Number <chr>, License.Type <chr>,
## #   License.Expiration.Date <chr>, License.Category <chr>,
## #   Business.Name <chr>, Business.Name.2 <chr>, Address.Building <chr>,
## #   Address.Street.Name <chr>, Secondary.Address.Street.Name <chr>,
## #   Address.City <chr>, Address.State <chr>, Address.ZIP <chr>,
## #   Address.Borough <chr>, Detail <chr>, Longitude <dbl>, Latitude <dbl>
```