Untitled

Jyoti Chaudhary

October 25, 2016

Problem 1

1. We would like to visually compare first names of baseball players with those of male babies in the population at large. This will require several steps.

```
## Loading tidyverse: ggplot2
## Loading tidyverse: tibble
## Loading tidyverse: tidyr
## Loading tidyverse: readr
## Loading tidyverse: purrr
## Loading tidyverse: dplyr
```

```
## Conflicts with tidy packages ----------------------------------------------
```

```
## filter(): dplyr, stats
## lag():    dplyr, stats
```

```
##
## Attaching package: 'curl'
```

```
## The following object is masked from 'package:readr':
##
##     parse_date
```

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_200 <- Fielding %>% group_by(playerID) %>%
               mutate(Gsum = sum(G)) %>%
                filter(Gsum >= 200) %>%
            left_join(Master, by=c('playerID'))

head(Players_200, 10)
```

```
## Source: local data frame [10 x 44]
## Groups: playerID [4]
##
##     playerID yearID stint teamID   lgID   POS     G    GS InnOuts    PO
##        <chr>  <int> <int> <fctr> <fctr> <chr> <int> <int>   <int> <int>
## 1   addybo01   1871     1    RC1     NA    2B    22    NA      NA    67
## 2   addybo01   1871     1    RC1     NA    SS     3    NA      NA     8
## 3  allisdo01   1871     1    WS3     NA     C    27    NA      NA    68
## 4  ansonca01   1871     1    RC1     NA    1B     1    NA      NA     7
## 5  ansonca01   1871     1    RC1     NA    2B     2    NA      NA     3
## 6  ansonca01   1871     1    RC1     NA    3B    20    NA      NA    38
## 7  ansonca01   1871     1    RC1     NA     C     5    NA      NA    10
## 8  ansonca01   1871     1    RC1     NA    OF     1    NA      NA     0
## 9  barnero01   1871     1    BS1     NA    2B    16    NA      NA    42
## 10 barnero01   1871     1    BS1     NA    SS    15    NA      NA    44
## # ... with 34 more variables: A <int>, E <int>, DP <int>, PB <int>,
## #   WP <int>, SB <int>, CS <int>, ZR <int>, Gsum <int>, birthYear <int>,
## #   birthMonth <int>, birthDay <int>, birthCountry <chr>,
## #   birthState <chr>, 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 <fctr>, throws <fctr>, 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_200_prop <- Players_200 %>%
  group_by(birthYear) %>%
  mutate(nprop = n())

babyname_df <- Players_200_prop %>%
              group_by(nameFirst, birthYear) %>%
              mutate(n = n(), prop = n/nprop, sex = "M" ) %>%
              select(birthYear, sex, nameFirst, n, prop)

head(babyname_df)
```

```
## Source: local data frame [6 x 5]
## Groups: nameFirst, birthYear [3]
##
##   birthYear   sex nameFirst     n      prop
##       <int> <chr>     <chr> <int>     <dbl>
## 1      1842     M       Bob    11 0.3666667
## 2      1842     M       Bob    11 0.3666667
## 3      1846     M      Doug    22 0.2784810
## 4      1852     M       Cap    68 0.3383085
## 5      1852     M       Cap    68 0.3383085
## 6      1852     M       Cap    68 0.3383085
```

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

```
filter(babynames, sex == "M") %>%
inner_join(babyname_df, by = c("name" = "nameFirst"))
```

```
## # A tibble: 14,533,741 × 9
##     year sex.x  name   n.x     prop.x birthYear sex.y   n.y     prop.y
##    <dbl> <chr> <chr> <int>      <dbl>     <int> <chr> <int>      <dbl>
## 1   1880     M  John  9655 0.08154561      1850     M    29 0.11934156
## 2   1880     M  John  9655 0.08154561      1847     M    14 0.09929078
## 3   1880     M  John  9655 0.08154561      1847     M    14 0.09929078
## 4   1880     M  John  9655 0.08154561      1847     M    14 0.09929078
## 5   1880     M  John  9655 0.08154561      1849     M    13 0.05531915
## 6   1880     M  John  9655 0.08154561      1849     M    13 0.05531915
## 7   1880     M  John  9655 0.08154561      1851     M    30 0.14150943
## 8   1880     M  John  9655 0.08154561      1851     M    30 0.14150943
## 9   1880     M  John  9655 0.08154561      1851     M    30 0.14150943
## 10  1880     M  John  9655 0.08154561      1850     M    29 0.11934156
## # ... with 14,533,731 more rows
```

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.

```
popular_babynames <- filter(babynames, year >= "1890" & year <= "1990") %>%
                      group_by(name) %>%
                      mutate(n1=sum(n)) %>%
                      select(name, n1) %>%
                      arrange(desc(n1))

popular_babynames5 <- unique(popular_babynames)

## Popular babynames
head(popular_babynames5, 5)
```

```
## Source: local data frame [5 x 2]
## Groups: name [5]
##
##      name      n1
##     <chr>   <int>
## 1   James 4629892
## 2    John 4589273
## 3  Robert 4474257
## 4    Mary 3911906
## 5 Michael 3596317
```

```
popular_baseball <- filter(babyname_df, birthYear >= "1890" & birthYear <= "1990") %>%
  group_by(nameFirst) %>%
  mutate(n1=sum(n)) %>%
  select(nameFirst, n1) %>%
  arrange(desc(n1))

popular_baseball5 <- unique(popular_baseball)

## Popular baseball player names
head(popular_baseball5, 5)
```

```
## Source: local data frame [5 x 2]
## Groups: nameFirst [5]
##
##   nameFirst     n1
##       <chr>  <int>
## 1      Mike 307661
## 2       Jim 178924
## 3      Dave 155031
## 4      John 120804
## 5     Jerry 101798
```

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).

```
popular_babynames_year <- filter(babynames, sex == "M") %>%
                          group_by(year) %>%
                        mutate(year_sum = sum(n))
popular_babynames_year_name <- group_by(popular_babynames_year, year, name) %>%
             mutate(name_sum = sum(n), prop_name_year = name_sum/year_sum)

head(popular_babynames_year_name)
```

```
## Source: local data frame [6 x 8]
## Groups: year, name [6]
##
##    year   sex    name     n       prop year_sum name_sum prop_name_year
##   <dbl> <chr>   <chr> <int>      <dbl>    <int>    <int>          <dbl>
## 1  1880     M    John  9655 0.08154561   110491     9655     0.08738268
## 2  1880     M William  9532 0.08050676   110491     9532     0.08626947
## 3  1880     M   James  5927 0.05005912   110491     5927     0.05364238
## 4  1880     M Charles  5348 0.04516892   110491     5348     0.04840213
## 5  1880     M  George  5126 0.04329392   110491     5126     0.04639292
## 6  1880     M   Frank  3242 0.02738176   110491     3242     0.02934176
```

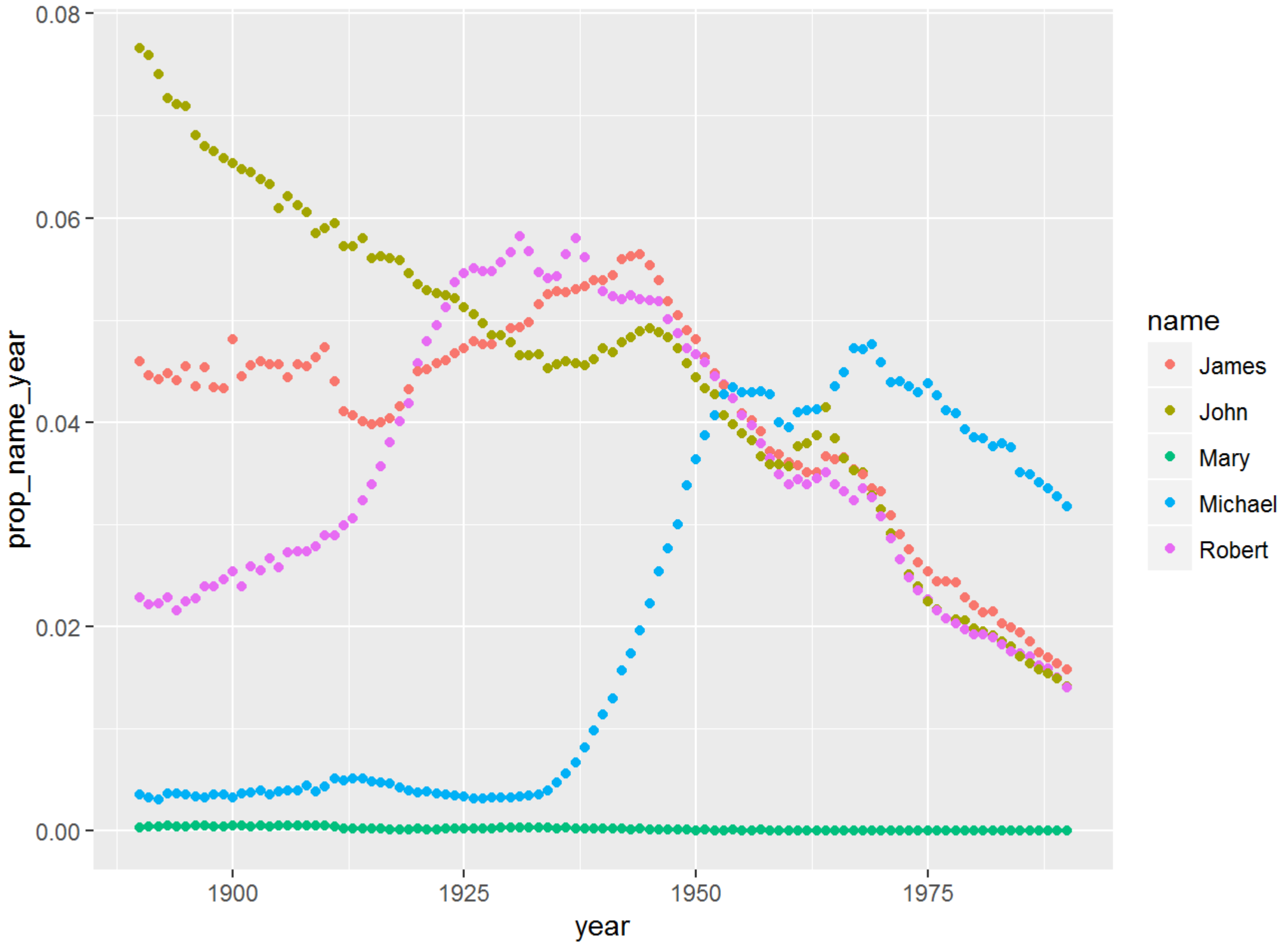
```
popular_baseball_year <- group_by(babyname_df, birthYear) %>%
                        mutate(year_sum = sum(n))
popular_baseball_year_name <- group_by(popular_baseball_year, birthYear, nameFirst) %>%
             mutate(name_sum = sum(n), prop_name_year = name_sum/year_sum)

head(popular_baseball_year_name)
```

```
## Source: local data frame [6 x 8]
## Groups: birthYear, nameFirst [3]
##
##   birthYear   sex nameFirst     n      prop year_sum name_sum
##       <int> <chr>     <chr> <int>     <dbl>    <int>    <int>
## 1      1842     M       Bob    11 0.3666667      482      121
## 2      1842     M       Bob    11 0.3666667      482      121
## 3      1846     M      Doug    22 0.2784810     1623      484
## 4      1852     M       Cap    68 0.3383085     7887     4624
## 5      1852     M       Cap    68 0.3383085     7887     4624
## 6      1852     M       Cap    68 0.3383085     7887     4624
## # ... with 1 more variables: prop_name_year <dbl>
```

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.

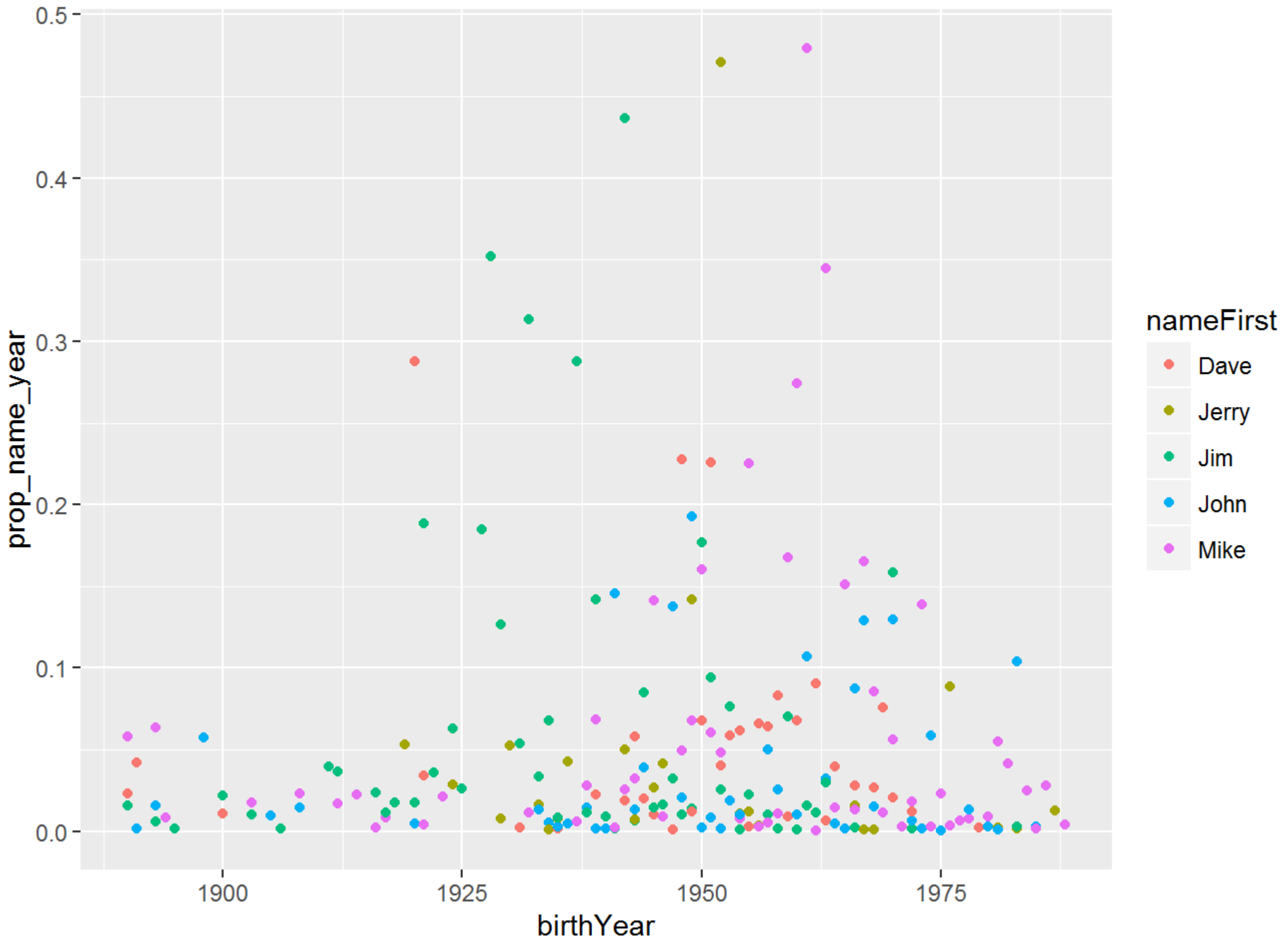
GENERAL POPULATION NAMES

```
ggplot(data = filter(popular_babynames_year_name, sex == "M", year >=1890 & year <=1990,
                                  name == "James" |
                                  name == "John" |
                                  name == "Robert" |
                                  name =="Mary"|
                                  name == "Michael")) +
                                  geom_point(aes(year,prop_name_year, colour = name))
```



BASEBALL PLAYER NAMES

```
ggplot(data = filter(popular_baseball_year_name, birthYear >=1890 & birthYear <=1990,
                                  nameFirst == "Mike" |
                                  nameFirst == "Jim" |
                                  nameFirst == "Dave" |
                                  nameFirst =="John"|
                                  nameFirst == "Jerry")) +
                                  geom_point(aes(birthYear,prop_name_year, colour = nameFirst))
```



Problem 2

The join in Week 3’s third homework likely missed some entries because of differences in how the phone numbers were formatted. Reformat the phone numbers in the two data sets to a common format (your choice) and then repeat your analysis from last week.

```
# Joined data frame of restaurant health inspection dataset and Legal business dataset. Join on contact phone number.
head(Legal_match, 10)
```

```
## Source: local data frame [10 x 24]
## Groups: PHONE [9]
##
##                     DBA INSPECTION.DATE      BORO BUILDING
##                   <chr>           <chr>     <chr>    <chr>
## 1              101 DELI      08/20/2013    QUEENS    10016
## 2   107 WEST RESTAURANT      12/07/2013 MANHATTAN     2787
## 3  10TH AVENUE COOKSHOP      11/20/2013 MANHATTAN      156
## 4        111 RESTAURANT      09/18/2013    QUEENS        0
## 5        111 RESTAURANT      09/18/2013    QUEENS        0
## 6      129 GOURMET DELI      08/27/2015 MANHATTAN      129
## 7            15 FLAVORS      09/04/2015     BRONX     3815
## 8    1ST AVENUE GOURMET      12/02/2015 MANHATTAN     1274
## 9             25TH DELI      11/08/2014    QUEENS     4819
## 10                   2A      10/29/2013 MANHATTAN       25
## # ... with 20 more variables: STREET <chr>, ZIPCODE <int>, PHONE <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>
```

```
# Top 10 Licence categories
head(sort(table(Legal_match$License.Category),decreasing = TRUE), 10)
```

```
##
## Home Improvement Salesperson      Cigarette Retail Dealer
##                        55502                        20895
##            Electronics Store  Home Improvement Contractor
##                        17320                        11439
##  Secondhand Dealer - General             Tow Truck Driver
##                         8320                         5266
##                Sidewalk Cafe             Stoop Line Stand
##                         4508                         3912
##       Debt Collection Agency                      Laundry
##                         3699                         3535
```

Problem 3

3. Read the post at http://www.sumsar.net/blog/2016/09/whats-on-the-menu/ (http://www.sumsar.net/blog/2016/09/whats-on-the-menu/) and follow the steps yourself. (Please include the R code in the RMarkdown file up through the creation of the data frame “d”—a terrible name, by the way.)

```
## Warning: 23 parsing failures.
##   row      col expected         actual
## 13943 image_id a double ps_rbk_637
## 13944 image_id a double ps_rbk_657
## 13945 image_id a double ps_rbk_661
## 13946 image_id a double psnypl_rbk_951
## 13947 image_id a double psnypl_rbk_952
## ..... ........ ........ ..............
## See problems(...) for more details.
```

```
## # A tibble: 10 × 6
##     year        location menu_id                         dish_name price
##    <dbl>           <chr>   <int>                             <chr> <dbl>
## 1   1900 Claremont Hotel   12882       Consomme printaniere royal  0.40
## 2   1900 Claremont Hotel   12882                    Chicken gumbo  0.60
## 3   1900 Claremont Hotel   12882              Tomato aux croutons  0.40
## 4   1900 Claremont Hotel   12882                   Onion au gratin  0.50
## 5   1900  La Noche Buena   13472                        St. Emilion  0.50
## 6   1900 Claremont Hotel   12882                          Radishes  0.10
## 7   1900 Claremont Hotel   12882                   Clam broth (cup)  0.25
## 8   1900 Claremont Hotel   12882 Cream of new asparagus, croutons  0.75
## 9   1900 Claremont Hotel   12882               Clear green turtle  0.75
## 10  1900 Claremont Hotel   12882           Chicken soup with rice  0.60
## # ... with 1 more variables: place <chr>
```

Interesting observation 1

MASHED POTATOES were more common and presumably more popular food item in comparison to BROWNED POTATOES and GERMAN FRIED POTATOES.

```
d$decennium = floor(d$year / 10) * 10

foods <- c("coffee", "tea", "pancake", "ice cream", "french frie",
           "french peas", "apple", "banana", "strawberry", "Mashed potatoes", "BROWNED POTATOES"
, "German fried potatoes")

food_over_time <- map_df(foods, function(food) {
  d %>%
    filter(d$year >= 1900 & d$year <= 1980) %>%
    group_by(decennium, menu_id)  %>%
    summarise(contains_food =
                any(str_detect(dish_name, regex(paste0("\\b", food), ignore_case = TRUE)),
                    na.rm = TRUE))  %>%
    summarise(prop_food = mean(contains_food, na.rm = TRUE)) %>%
    mutate(food = food)
})

food_time_plot <- list(
  geom_line(),
  geom_point(),
  scale_y_continuous("% of menus include",labels = scales::percent,
                     limits = c(0, NA)),
  scale_x_continuous(""),
  facet_wrap(~ food),
  theme_minimal(),
  theme(legend.position = "none"))


# Could not generate the plot out of below code. Getting error - "Aesthetics must be either length 1 or the same as the data". Tried to fix the issue, but could not resolve.

#food_over_time %>% filter(food %in% c("Mashed potatoes", "BROWNED POTATOES", "German fried potatoes")) %>%
#  ggplot(aes(d$decennium, prop_food, color = food)) + food_time_plot
```