Homework 1 : review and coding questions

TA : Arjama Das

Due: 01/31 at 11:59 PM

This assignment is meant to serve multiple objectives:

- You will gain familiarity with R, RStudio, R Markdown, and GitHub
- You will perform at least one iteration of the courses's data science inspired workflow
- You will gain experience with typing mathematics
- You will learn some `dplyr` and `ggplot2` basics

STAT 528 is a collaborative course environment, especially for assignments that involve coding, modeling, and/or data analysis. You are encouraged to ask for help from other students. Coding and data science work flow can be very tedious. Having someone else look over your work or answering a basic question can save you a lot of time. However, direct copying is not accepting. All final work must be your own.

Mathematical review questions

Problem 1: Prove that the Binomial distribution arises as a sum of n iid Bernoulli trials each with success probability p.

Solution: To see this, we check the equivalence of moment generating function. Denote the Bernoulli Trials to be $X_1, \ldots, X_n$. First for a rv $Y \sim \text{Binom}(n, p)$, recall

$$f_Y(y) = \left(\begin{array}{c} n \\ y \end{array} \right) p^y (1-p)^{n-y}, y = 0, \ldots, n$$

Its generating function gives

$$M_Y(t) = \mathbb{E}\left[e^{tY}\right] = \sum_{y=0}^{\infty} e^{ty} \cdot f_Y(y) = \sum_{y=0}^{n} \left(\begin{array}{c} n \\ y \end{array} \right) \cdot e^{ty} p^y \cdot (1-p)^{n-y}$$

Then for $X := \sum_{i=1}^{n} X_i$, apply the iid assumption and Binomial Theorem, it follows

$$\begin{aligned}
M_X(t) &= \mathbb{E}\left[e^{tX}\right] \\
&= \mathbb{E}\left[e^{t\sum_{i=1}^{n} X_i}\right] \\
&= \mathbb{E}\left[\prod_{i=1}^{n} e^{tX_i}\right] \\
&= \prod_{i=1}^{n} \mathbb{E}\left[e^{tX_i}\right] \\
&= \prod_{i=1}^{n} \left(e^{t\cdot 1}\cdot p + e^{t\cdot 0}\cdot(1-p)\right) \\
&= \left(e^t p + (1-p)\right)^n \\
&= \sum_{k=0}^{n} \left(\begin{array}{c} n \\ k \end{array} \right) \cdot \left(e^t p\right)^k \cdot (1-p)^{n-k} \\
&= \sum_{k=0}^{n} \left(\begin{array}{c} n \\ k \end{array} \right) \cdot e^{tk} \cdot p^k \cdot (1-p)^{n-k} \\
&= M_Y(t)
\end{aligned}$$

Problem 2: Let $l(\theta)$ denote a twice continuously differentiable log likelihood corresponding to an iid sample under density f_θ where n is the sample size. The score function is defined as

$$u(\theta) = \frac{\partial l(\theta)}{\partial \theta},$$

and the Fisher information matrix is defined as

$$I(\theta) = -\mathrm{E}\left(\frac{\partial^2 l(\theta)}{\partial \theta^2}\right),$$

where the expectation is over the assumed distribution for the data when the parameter value is θ. Prove that

$$\mathrm{E}(u(\theta)) = 0 \qquad \text{and} \qquad \mathrm{Var}(u(\theta)) = I(\theta).$$

Solution: For (i), by definition, apply Chain rule and twice cont. differentiability it follows

$$\begin{aligned}
\mathbb{E}[u(\theta)] &= \mathbb{E}\left[\frac{\partial l(\theta)}{\partial \theta}\right] \\
&= \mathbb{E}\left[\frac{\partial \log f_\theta}{\partial \theta}\right] \\
&= \int_X \frac{\partial}{\partial \theta} \log f_\theta(x) \cdot f_\theta(x) dx \\
&= \int_X \frac{1}{f_\theta(x)} \cdot \frac{\partial}{\partial \theta} f_\theta(x) \cdot f_\theta(x) dx \\
&= \int_X \frac{\partial}{\partial \theta} f_\theta(x) dx \\
&= \frac{\partial}{\partial \theta} \int_X f_\theta(x) dx \\
&= \frac{\partial}{\partial \theta} 1 \\
&= 0
\end{aligned}$$

For (ii), we equivalently want to show $\mathbb{E}\left[u(\theta)^2\right] - \mathbb{E}[u(\theta)]^2 = I(\theta)$. We have seen $\mathbb{E}[u(\theta)] = 0$, then that is to show $-\mathbb{E}\left[\left(\frac{\partial l(\theta)}{\partial \theta}\right)^2\right] = \mathbb{E}\left[\frac{\partial^2 l(\theta)}{\partial \theta^2}\right]$. Note

$$\frac{\partial^2 l(\theta)}{\partial \theta^2} = \frac{\partial^2}{\partial \theta^2} \log f_\theta = \frac{\partial}{\partial \theta}\left(\frac{\partial}{\partial \theta} \log f_\theta\right) = \frac{\partial}{\partial \theta}\left(\frac{1}{f_\theta} \cdot \frac{\partial}{\partial \theta} f_\theta\right) = \frac{1}{f_\theta} \cdot \frac{\partial^2}{\partial \theta^2} f_\theta - \frac{1}{f_\theta^2} \cdot \left(\frac{\partial}{\partial \theta} f_\theta\right)^2 = \frac{1}{f_\theta} \cdot \frac{\partial^2}{\partial \theta^2} f_\theta - \left(\frac{\partial}{\partial \theta} l(\theta)\right)^2$$

Take expectation on both side to have

$$\begin{aligned}
\mathbb{E}\left[\frac{\partial^2 l(\theta)}{\partial \theta^2}\right] &= \mathbb{E}\left[\frac{1}{f_\theta} \cdot \frac{\partial^2}{\partial \theta^2} f_\theta\right] - \mathbb{E}\left[\left(\frac{\partial}{\partial \theta} l(\theta)\right)^2\right] \\
&= \int_X \frac{\partial^2}{\partial \theta^2} f_\theta dx - \mathbb{E}\left[\left(\frac{\partial}{\partial \theta} l(\theta)\right)^2\right] \\
&= \frac{\partial^2}{\partial \theta^2} \int_X f_\theta dx - \mathbb{E}\left[\left(\frac{\partial}{\partial \theta} l(\theta)\right)^2\right] \\
&= \frac{\partial^2}{\partial \theta^2} 1 - \mathbb{E}\left[\left(\frac{\partial}{\partial \theta} l(\theta)\right)^2\right] \\
&= 0 - \mathbb{E}\left[\left(\frac{\partial}{\partial \theta} l(\theta)\right)^2\right] \\
&= -\mathbb{E}\left[\left(\frac{\partial l(\theta)}{\partial \theta}\right)^2\right]
\end{aligned}$$

Coding questions

Problem 3: The data we will use to accomplish this task will come from Lahman's Baseball Database. Thankfully, there is an R package, `Lahman`, that makes importing this data into R very easy. If you have

not done so previously, install this package using:

```
install.packages("Lahman")
```

While there many metrics that could be used to determine who is the "best" baseball player, because we are focusing on batters, we will use the on-base plus slugging (OPS) statistic. This statistic measures both a batter's ability to "get on base" and "hit for power."

- YouTube: Moneyball, "He Gets on Base"

Additionally, our definition of "best" will be based on a player's career statistics, but an alternative argument could be made based on single season efforts.

After loading the Lahman package, you will have access to several data frames containing historical baseball data from 1871 - 2023. You will need to interact with the following data frames:

- Schools
- CollegePlaying
- Batting
- People

You should spend some time exploring these datasets and reading the relevant documentation.

Create a tibble named `illini_mlb_batters` that contains the following elements, in this order:

- playerID
- nameFirst
- nameLast
- birthYear
- G
- AB
- R
- H
- X2B
- X3B
- HR
- RBI
- SB
- CS
- BB
- SO
- IBB
- HBP
- SH
- SF
- GIDP
- PA
- TB
- BA
- OBP
- SLG
- OPS

The rows of the tibble should be sorted from highest OPS to lowest OPS. Each row should represent the career statistics for the player with ID playerID. Only include players that had at least one at-bat and one plate appearance. Except for PA, TB, AVG, OBP, SLG, and OPS, the (sometimes season-level) variables listed can be found in one of the four data frames listed above. The remaining values can be calculated as follows:

- PA = AB + BB + HBP + SH + SF
- TB = H + X2B + 2 * X3B + 3 * HR
- BA = H / AB
- OBP = (H + BB + HBP) / (PA - SH)
- SLG = TB / AB
- OPS = OBP + SLG

Round any rate statistics to three decimals places, as is customary in baseball.

Solution:

```
# intall the packages and upload the tibbles
# install.packages("Lahman")
library(Lahman)
library(tibble)
library(tidyverse)
as_tibble(Schools)
```

```
## # A tibble: 1,241 x 5
##    schoolID   name_full                       city        state country
##    <chr>      <chr>                           <chr>       <chr> <chr>
##  1 abilchrist Abilene Christian University    Abilene     TX    USA
##  2 adelphi    Adelphi University              Garden City NY    USA
##  3 adrianmi   Adrian College                  Adrian      MI    USA
##  4 akron      University of Akron             Akron       OH    USA
##  5 alabama    University of Alabama           Tuscaloosa  AL    USA
##  6 alabamaam  Alabama A&M University          Normal      AL    USA
##  7 alabamast  Alabama State University        Montgomery  AL    USA
##  8 albanyst   Albany State University         Albany      GA    USA
##  9 albertsnid Albertson College               Caldwell    ID    USA
## 10 albevil    Bevill State Community College Sumiton     AL    USA
## # i 1,231 more rows
```

```
as_tibble(CollegePlaying)
```

```
## # A tibble: 17,350 x 3
##    playerID  schoolID yearID
##    <chr>     <chr>     <int>
##  1 aardsda01 pennst     2001
##  2 aardsda01 rice       2002
##  3 aardsda01 rice       2003
##  4 abadan01  gamiddl    1992
##  5 abadan01  gamiddl    1993
##  6 abbeybe01 vermont    1889
```

```
##  7 abbeybe01 vermont    1890
##  8 abbeybe01 vermont    1891
##  9 abbeybe01 vermont    1892
## 10 abbotje01 kentucky   1991
## # i 17,340 more rows
```

```
as_tibble(Batting)
```

```
## # A tibble: 113,799 x 22
##    playerID  yearID stint teamID lgID      G    AB     R     H   X2B   X3B    HR
##    <chr>      <int> <int> <fct>  <fct> <int> <int> <int> <int> <int> <int> <int>
##  1 aardsda01   2004     1 SFN    NL       11     0     0     0     0     0     0
##  2 aardsda01   2006     1 CHN    NL       45     2     0     0     0     0     0
##  3 aardsda01   2007     1 CHA    AL       25     0     0     0     0     0     0
##  4 aardsda01   2008     1 BOS    AL       47     1     0     0     0     0     0
##  5 aardsda01   2009     1 SEA    AL       73     0     0     0     0     0     0
##  6 aardsda01   2010     1 SEA    AL       53     0     0     0     0     0     0
##  7 aardsda01   2012     1 NYA    AL        1     0     0     0     0     0     0
##  8 aardsda01   2013     1 NYN    NL       43     0     0     0     0     0     0
##  9 aardsda01   2015     1 ATL    NL       33     1     0     0     0     0     0
## 10 aaronha01   1954     1 ML1    NL      122   468    58   131    27     6    13
## # i 113,789 more rows
## # i 10 more variables: RBI <int>, SB <int>, CS <int>, BB <int>, SO <int>,
## #   IBB <int>, HBP <int>, SH <int>, SF <int>, GIDP <int>
```

```
as_tibble(People)
```

```
## # A tibble: 21,010 x 26
##    playerID  birthYear birthMonth birthDay birthCity    birthCountry birthState
##    <chr>         <int>      <int>    <int> <chr>        <chr>        <chr>
##  1 aardsda01      1981         12       27 Denver       USA          CO
##  2 aaronha01      1934          2        5 Mobile       USA          AL
##  3 aaronto01      1939          8        5 Mobile       USA          AL
##  4 aasedo01       1954          9        8 Orange       USA          CA
##  5 abadan01       1972          8       25 Palm Beach   USA          FL
##  6 abadfe01       1985         12       17 La Romana    D.R.         La Romana
##  7 abadijo01      1850         11        4 Philadelphia USA          PA
##  8 abbated01      1877          4       15 Latrobe      USA          PA
##  9 abbeybe01      1869         11       11 Essex        USA          VT
## 10 abbeych01      1866         10       14 Falls City   USA          NE
## # i 21,000 more rows
## # i 19 more variables: 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>, bbrefID <chr>, finalGame <chr>, retroID <chr>,
## #   deathDate <date>, birthDate <date>
```

```
# Create a tibble to select the players in Illinois
illiniIDs = CollegePlaying %>%
  filter(schoolID == "illinois") %>%
  pull(playerID) %>%
  unique()
```

```r
foo = People %>%
  select(playerID, nameFirst, nameLast, birthYear)

# Create the illini_mlb_batters tibble as required
illini_mlb_batters = Batting %>%
  filter(playerID %in% illiniIDs) %>%
  select(playerID, G:GIDP) %>%
  mutate(across(G:GIDP, ~replace_na(.x,0))) %>%
  group_by(playerID) %>%
  summarise(across(G:GIDP, sum)) %>%
  mutate(PA=AB+BB+HBP+SH+SF,
    TB=H+X2B+2*X3B+3*HR,
    BA=H/AB,
    OBP=(H+BB+HBP)/(PA-SH),
    SLG=TB/AB,
    OPS=OBP+SLG) %>%
  left_join(foo, by="playerID") %>%
  select(playerID, nameFirst, nameLast, birthYear, everything()) %>%
  mutate(across(BA:OPS, ~ round(.x,3))) %>%
  arrange(desc(OPS)) %>%
  filter(AB >= 1)

print.data.frame(head(illini_mlb_batters))
```

```
##    playerID nameFirst nameLast birthYear    G   AB   R    H X2B X3B  HR RBI SB
## 1 boudrlo01       Lou Boudreau      1917 1646 6029 861 1779 385  66  68 789 51
## 2 evershо01      Hoot    Evers      1921 1142 3801 556 1055 187  41  98 565 45
## 3 halleto01       Tom   Haller      1937 1294 3935 461 1011 153  31 134 504 14
## 4 spiezsc01     Scott  Spiezio      1972 1274 3899 517  996 225  27 119 549 33
## 5 mccurha01     Harry  McCurdy      1899  543 1157 148  326  71  12   9 148 12
## 6 fletcda01    Darrin Fletcher      1966 1245 3902 377 1048 214   8 124 583  2
##   CS  BB  SO IBB HBP  SH SF GIDP   PA   TB    BA   OBP   SLG   OPS
## 1 50 796 309   0  34 164  0  155 7023 2500 0.295 0.380 0.415 0.795
## 2 36 415 420   0  27  65  2  116 4310 1618 0.278 0.353 0.426 0.778
## 3 30 477 593  96  35  35 37   60 4519 1628 0.257 0.340 0.414 0.753
## 4 23 412 594  35  35  25 41   77 4412 1632 0.255 0.329 0.419 0.747
## 5  9 129 108   0   3  25  0    0 1314  448 0.282 0.355 0.387 0.743
## 6  6 255 399  31  49  13 51  122 4270 1650 0.269 0.318 0.423 0.740
```

Problem 4: The data we will use to accomplish this task will come from the `Teams` data frame in Lahman's Baseball Database. In this problem we will visualize the Pythagorean Theorem of Baseball. This "Theorem" states that winning percentage is given by the following nonlinear equation:

$$WP = \frac{R^2}{R^2 + RA^2}$$

where

- WP is winning percentage
- R is total runs scored by a baseball team

- RA is total runs allowed by a baseball team

For this problem, plot the estimated number of wins as predicted by the Pythagorean equation and actual wins (denoted W). The estimated number of wins as predicted by the Pythagorean equation

$$162 * \frac{R^2}{R^2 + RA^2}.$$

Provide a line of best fit. Restrict attention to the 1990 season and beyond. Note that there are two shortened seasons that need to be treated separately from the remaining seasons. These seasons are 1994 and 2020. The 1994 season was cut short because of a labor strike. The 2020 season was cut short due to COVID.

Solution:

```
library(ggplot2)
as_tibble(Teams)
```

```
## # A tibble: 3,045 x 48
##    yearID lgID  teamID franchID divID  Rank     G Ghome     W     L DivWin WCWin
##     <int> <fct> <fct>  <fct>    <chr> <int> <int> <int> <int> <int> <chr>  <chr>
##  1   1871 NA    BS1    BNA      <NA>      3    31    NA    20    10 <NA>   <NA>
##  2   1871 NA    CH1    CNA      <NA>      2    28    NA    19     9 <NA>   <NA>
##  3   1871 NA    CL1    CFC      <NA>      8    29    NA    10    19 <NA>   <NA>
##  4   1871 NA    FW1    KEK      <NA>      7    19    NA     7    12 <NA>   <NA>
##  5   1871 NA    NY2    NNA      <NA>      5    33    NA    16    17 <NA>   <NA>
##  6   1871 NA    PH1    PNA      <NA>      1    28    NA    21     7 <NA>   <NA>
##  7   1871 NA    RC1    ROK      <NA>      9    25    NA     4    21 <NA>   <NA>
##  8   1871 NA    TRO    TRO      <NA>      6    29    NA    13    15 <NA>   <NA>
##  9   1871 NA    WS3    OLY      <NA>      4    32    NA    15    15 <NA>   <NA>
## 10   1872 NA    BL1    BLC      <NA>      2    58    NA    35    19 <NA>   <NA>
## # i 3,035 more rows
## # i 36 more variables: LgWin <chr>, WSWin <chr>, R <int>, AB <int>, H <int>,
## #   X2B <int>, X3B <int>, HR <int>, BB <int>, SO <int>, SB <int>, CS <int>,
## #   HBP <int>, SF <int>, RA <int>, ER <int>, ERA <dbl>, CG <int>, SHO <int>,
## #   SV <int>, IPouts <int>, HA <int>, HRA <int>, BBA <int>, SOA <int>, E <int>,
## #   DP <int>, FP <dbl>, name <chr>, park <chr>, attendance <int>, BPF <int>,
## #   PPF <int>, teamIDBR <chr>, teamIDlahman45 <chr>, teamIDretro <chr>
```

```
# Create the PTB tibble as required
PTB = Teams %>%
  filter(yearID >= 1990) %>%
  select(yearID, W, R, RA) %>%
  mutate(WP = 162*R^2/(R^2 + RA^2)) %>%
  mutate(seasons = case_when(
    yearID == 1994 ~ "labor strike",
    yearID == 2020 ~ "COVID",
    .default ="normal"
  ))

# Provide the line with "lm" method in ggplot to get the best fit of W and WP
ggplot(PTB) +
```

```
  aes(x = WP, y = W, color = seasons) +
  geom_point() +
  geom_smooth(method = "lm") +
  labs(title = "The Line of Best Fit between Actual Wins and Estimated Number of Wins by Pythagorean Eq
       x = "Estimated Number of Wins by Pythagorean Equation",
       y = "Actual Wins") +
  theme_minimal()
```