# SML 201 Problem Set 1 Solutions

*Bill Haarlow & Weston Carpenter*

*2019-10-01*

**Problem set 1 is due by 11:59pm on Tuesday October 1.** Please submit both a .Rmd and a .pdf file on Blackboard by the deadline **and** drop off a hard copy of the pdf file at 26 Prospect Avenue by 5pm of the **next day** of the due date. To look for the drop-off cabinet, after you enter the building turn to the left to enter the lounge area and the file cabinet is to your right with an open slot with the label "SML 201 Homework"; note that the building might be locked after 6pm and on the weekends. You are also welcome to bring your PDF copy to any lecture **before** the deadline and I will drop off the copy for you.

This problem set can be completed in groups of up to 3 students. It is okay to work by yourself, if this is preferable. You are welcome to get help (you can either ask questions on Piazza or talk to instructors in person during office hours) from instructors; however, please do not post code/solutions on Piazza on a public post.

When working in a group it is your responsibility to make sure that you are satisfied with all parts of the report and the submission is on time (e.g., we will not entertain arguments that deficiencies are the responsibility of other group members). We expect that you each work independently first and then compare your answers with each other once you all finish, or you all work together on your own laptops. Failing to make contributions and then putting your name on a report will be considered a violation of the honor code. **Please do not divide work among group mates.** Everyone in your group is responsible for being able to explain the solutions in the report.

For all parts of this problem set, you **MUST** use R commands to print the output as part of your R Markdown file. You are not permitted to find the answer in the R console and then copy and paste the answer into this document.

**If you are completing this problem set in a group**, please have only **one** person in your group turn in the .Rmd and .pdf files; the other person in your group should turn in the list of the people in your group in the *Text Submission* field on the submission page. This means that **everyone should make a submission**–either a file-upload or a text submission–regardless of whether you are working in a group or not.

---

Please type your name(s) after "Digitally signed:" below the honor pledge to serve as digital signature(s). Put the pledge and your signature(s) at the beginning of each document that you turn in.

> I pledge my honor that I have not violated the honor code when completing this assignment.

Digitally signed: Bill Haarlow & Weston Carpenter

---

**In order to receive full credits, please have sensible titles and axis labels for all your graphs and adjust values of the relevant graphical parameters so that your plots are informative. Also, all answers must be written in complete sentences.**

There are numerous webpages online that provide the popularity rank of a given name. These webpages are particularly interesting to soon-to-be parents (see http://www.bbc.com/news/uk-england-40774941). We will use the `babynames` dataset in the `babynames` package for this problem set; we will show you how this kind of rankings are generated. By the end of this problem set you will be able to produce popularity rankings of names for any time period that you choose and to answer some interesting questions with the dataset.

Remember that to access datasets in a package you will need to first load the package where the datasets are coming from.

```
library(babynames)  # Loads package `babynames`
```

## (9 pts) (3 pts each: code runs, code has annotations, answers are in complete sentences)

## (1 pt) (Did not print out large datasets that serve no purpose for the report)

## (14 pts) Question 1: Getting familiar with the data

This question is the last exercise in Precept1.

**Before you start**:

- Please make sure that you have the required versions of R and Rstudio installed (see the first announcement of the semester on Blackboard for the version numbers). Not using the up-to-date versions of R and Rstudio might result in incorrect answers to the questions and point deductions.
- Please make sure that you have read the instructions at the beginning of this document.

### (1 pts) Part a

What kind of R object is `babynames`?

```
class(babynames)  # Outputs the kind of R object for `babynames`
[1] "tbl_df"     "tbl"        "data.frame"
```

Answer: `babynames` is a data frame, according to the `class()` function.

### (2 pts) Part b

How many rows and columns does `babynames` have?

```
dim(babynames)  # Outputs the dimensions of `babynames`
[1] 1924665       5
```

Answer: `babynames` has 1,924,665 rows & 5 columns.

### (2 pts) Part c

Recall that for any built-in datasets or datasets from a package the information about the dataset is available on the dataset's help manual. Please make sure that you read the information about the dataset `babynames` on its help manual page and make sure that you understand the definitions of the variables in the dataset. What command line do you use to access the help manual page of `babynames`? List the names of the variables along with their definitions/meanings–it is okay to just copy and paste the definitions from the help manual if they are available.

```
help(babynames)  # This opens `babynames` on its help manual page
```

Answer: There are five variables in the data frame `babynames`. They are: year (year born), sex (sex of babies), name (name of babies), n (number of name uses per year), and prop (n divided by total number of applicants in that year, which means proportions are of people of that gender with that name born in that year).

### (2 pts) Part d

What are the data types of the variables in `babynames`?

```
# Displays the interal structure of `babynames`,
# including the variable data types
str(babynames)
Classes 'tbl_df', 'tbl' and 'data.frame':   1924665 obs. of  5 variables:
 $ year: num  1880 1880 1880 1880 1880 1880 1880 1880 1880 1880 ...
 $ sex : chr  "F" "F" "F" "F" ...
 $ name: chr  "Mary" "Anna" "Emma" "Elizabeth" ...
 $ n   : int  7065 2604 2003 1939 1746 1578 1472 1414 1320 1288 ...
 $ prop: num  0.0724 0.0267 0.0205 0.0199 0.0179 ...
```

Answer: The data types for the variables within `babynames` are as follows: `year` is numeric, `sex` is character, `name` is character, `n` is integer, and `prop` is numeric.

### (4 pts) Part e

What is the range of the values for the column `n` in `babynames` (i.e., what are the smallest and the largest values)?

```
range(babynames$n)  # Outputs the range of column `n` within `babynames`
[1]     5 99686
```

Answer: The range of `n` is [5, 99686].

Do you see any names with `n=1` in the dataset? Does this mean that for every baby that was born in a particular year there was always at least one other baby born in the same year sharing the same name?

```
min(babynames$n)  # Outputs the min value of `n` within `babynames`
[1] 5
```

Answer: Considering that the minimum value for `n` is 5, there are no names with `n=1` in the dataset. This means that for every baby born in a particular year, there are other babies born in that year that have the same name.

### (3 pts) Part f

What are the earliest and the latest years the dataset `babynames` records?

```
range(babynames$year)  # Gives the range of years in `babynames`
[1] 1880 2017
```

Answer: The `range()` function tells us that the earliest recorded year in the dataset `babynames` is 1880 and that the most recently recorded year is 2017.

## (51 pts) Question 2: Most popular baby girl names

Parts a and b of this question are the last exercise in Precept2.

Suppose that most of you in this class are turning 19 to 22 this year; this means that you were born between years 1997 and 2000. We will investigate the baby names in the dataset for this time period .

### (4 pts) Part a

Extract out the rows in `babynames` that correspond to the names of the babies born between years 1997 and 2000, inclusively and name the resulting subset `names1997.2000`. How many rows and columns does `names1997.2000` have?

```
placeholder = babynames$year %in% c(1997:2000)
# Extracts out the data for years [1997:2000]
names1997.2000 = babynames[placeholder, ]
# Assigns the corresponding rows in `babynames` to
# `names1997.2000`
dim(names1997.2000)  # Outputs the dimensions of subset `names1997.2000`
[1] 113189      5
```

Answer: The subset `names1997.2000` has 113,189 rows and 5 columns.

## (25 pts) Part b

Use `names1997.2000` from Part a for this Part. For the period between years 1997 to 2000 find out the top 20 most popular female names each year (it is okay to just print these four lists of names out in the code chunk; no need to include these names in the write-up). Among these four lists of top 20 female names, which names stay in the top 20 for all four years for the time period 1997 to 2000?

Please make sure that you print out only the requested names; printing out all the names in your dataset will likely cause Rstudio to have compiling issues since there will be many pages of names to be printed out.

```
female_baby = names1997.2000$sex == "F"  # Assigns female names to `female_baby`
top.names = names1997.2000[female_baby, ]  # Creates data frame `top.names`

year1997 = top.names$year %in% c(1997)  # Extracts all data from 1997
top.names1997 = top.names[year1997, ]
head(top.names1997$name, n = 20)  # Ouputs the top 20 names from 1997
 [1] "Emily"     "Jessica"   "Ashley"    "Sarah"     "Hannah"
 [6] "Samantha"  "Taylor"    "Alexis"    "Elizabeth" "Madison"
[11] "Megan"     "Kayla"     "Rachel"    "Lauren"    "Alyssa"
[16] "Amanda"    "Brianna"   "Jennifer"  "Victoria"  "Brittany"

year1998 = top.names$year %in% c(1998)  # Extracts all data from 1998
top.names1998 = top.names[year1998, ]
head(top.names1998$name, n = 20)  # Ouputs the top 20 names from 1998
 [1] "Emily"     "Hannah"    "Samantha"  "Sarah"     "Ashley"
 [6] "Alexis"    "Taylor"    "Jessica"   "Madison"   "Elizabeth"
[11] "Alyssa"    "Kayla"     "Megan"     "Lauren"    "Rachel"
[16] "Victoria"  "Brianna"   "Abigail"   "Amanda"    "Jennifer"

year1999 = top.names$year %in% c(1999)  # Extracts all data from 1999
top.names1999 = top.names[year1999, ]
head(top.names1999$name, n = 20)  # Ouputs the top 20 names from 1999
 [1] "Emily"     "Hannah"    "Alexis"    "Sarah"     "Samantha"
 [6] "Ashley"    "Madison"   "Taylor"    "Jessica"   "Elizabeth"
[11] "Alyssa"    "Lauren"    "Kayla"     "Brianna"   "Megan"
[16] "Victoria"  "Emma"      "Abigail"   "Rachel"    "Olivia"

year2000 = top.names$year %in% c(2000)  # Extracts all data from 2000
top.names2000 = top.names[year2000, ]
head(top.names2000$name, n = 20)  # Ouputs the top 20 names from 2000
 [1] "Emily"     "Hannah"    "Madison"   "Ashley"    "Sarah"
 [6] "Alexis"    "Samantha"  "Jessica"   "Elizabeth" "Taylor"
[11] "Lauren"    "Alyssa"    "Kayla"     "Abigail"   "Brianna"
[16] "Olivia"    "Emma"      "Megan"     "Grace"     "Victoria"
```

```
# This is essentially an expanded `%in%` comparing
# the top 20 names from all 4 of the years
top.names1997.8 = head(top.names1998$name, n = 20)[head(top.names1998$name,
    n = 20) %in% c("Emily", "Jessica", "Ashley", "Sarah",
    "Hannah", "Samantha", "Taylor", "Alexis", "Elizabeth",
    "Madison", "Megan", "Kayla", "Rachel", "Lauren",
    "Alyssa", "Amanda", "Brianna", "Jennifer", "Victoria",
    "Brittany")]
top.names1997.8.9 = head(top.names1999$name, n = 20)[head(top.names1999$name,
    n = 20) %in% top.names1997.8]
top.names1997.8.9.20 = head(top.names2000$name, n = 20)[head(top.names2000$name,
    n = 20) %in% top.names1997.8.9]
top.names1997.8.9.20  # Ouputs the consistent top names
 [1] "Emily"     "Hannah"    "Madison"   "Ashley"    "Sarah"
 [6] "Alexis"    "Samantha"  "Jessica"   "Elizabeth" "Taylor"
[11] "Lauren"    "Alyssa"    "Kayla"     "Brianna"   "Megan"
[16] "Victoria"
```

## (12 pts) Part c

Extract out the most popular 100 female names for the year 2017. Print out the names (for this part it is okay to just print them out in your code chunk; you do not need to list them again in the written part of your report) along with their counts; list the names in decreasing order of the counts; e.g., the first female name of the list should be the most popular female name in 2017.

```
year2017 = babynames$year %in% 2017  # Extracts out the data for year 2017
names2017 = babynames[year2017, ]  # Assigns rows of 2017 data to `names2017`
female_baby = names2017$sex == "F"  # Assigns female names to `female_baby`
a = names2017[female_baby, ]
b = sort(unique(a$n))  # Removes duplicate values from the list
names2017 = a[b, ]
# Displays top 100 female names for 2017
knitr::kable(head(names2017[c("name", "n")], n = 100))
```

| name | n |
|---|---|
| Sophia | 14831 |
| Mia | 13437 |
| Charlotte | 12893 |
| Amelia | 11800 |
| Evelyn | 10675 |
| Abigail | 10551 |
| Harper | 10451 |
| Emily | 9746 |
| Elizabeth | 8915 |
| Avery | 8186 |
| Sofia | 8134 |
| Ella | 8014 |
| Madison | 7847 |
| Scarlett | 7679 |
| Victoria | 7267 |
| Aria | 7132 |
| Grace | 6991 |
| Chloe | 6912 |
| Camila | 6752 |

| name | n |
|---|---|
| Penelope | 6639 |
| Riley | 6343 |
| Layla | 6274 |
| Lillian | 6132 |
| Nora | 6036 |
| Zoey | 6026 |
| Mila | 5941 |
| Aubrey | 5891 |
| Hannah | 5872 |
| Lily | 5816 |
| Addison | 5593 |
| Eleanor | 5519 |
| Natalie | 5516 |
| Luna | 5320 |
| Savannah | 5222 |
| Brooklyn | 5168 |
| Leah | 5159 |
| Zoe | 5129 |
| Stella | 5038 |
| Hazel | 5004 |
| Ellie | 4993 |
| Paisley | 4927 |
| Audrey | 4808 |
| Skylar | 4706 |
| Violet | 4699 |
| Claire | 4683 |
| Bella | 4611 |
| Aurora | 4573 |
| Lucy | 4564 |
| Anna | 4520 |
| Samantha | 4303 |
| Caroline | 4270 |
| Genesis | 4241 |
| Aaliyah | 4160 |
| Kennedy | 4137 |
| Kinsley | 4035 |
| Allison | 4017 |
| Maya | 4008 |
| Sarah | 3986 |
| Madelyn | 3939 |
| Adeline | 3902 |
| Alexa | 3883 |
| Ariana | 3865 |
| Elena | 3863 |
| Gabriella | 3862 |
| Naomi | 3823 |
| Alice | 3804 |
| Sadie | 3695 |
| Hailey | 3691 |
| Eva | 3614 |
| Emilia | 3581 |
| Autumn | 3575 |

| name | n |
|---|---|
| Quinn | 3575 |
| Nevaeh | 3562 |
| Piper | 3542 |
| Ruby | 3540 |
| Serenity | 3537 |
| Willow | 3529 |
| Everly | 3505 |
| Cora | 3422 |
| Kaylee | 3390 |
| Lydia | 3311 |
| Aubree | 3302 |
| Arianna | 3264 |
| Eliana | 3254 |
| Peyton | 3244 |
| Melanie | 3227 |
| Gianna | 3183 |
| Isabelle | 3145 |
| Julia | 3111 |
| Valentina | 3027 |
| Nova | 3026 |
| Clara | 3022 |
| Vivian | 3013 |
| Reagan | 2999 |
| Mackenzie | 2941 |
| Madeline | 2938 |
| Brielle | 2880 |
| Delilah | 2873 |
| Isla | 2863 |
| Rylee | 2847 |

### (10 pts) Part d

Did the list of female names staying in the top 20 consistently every year during the period 1997 to 2000 make it to the top 100 list for 2017? Print out the name(s) that did not make it to the top 100 list in your code chunk. How many are they? Make use of the function `%in%` to answer this question.

```
# This extracts the names that were consistent from
# 1997-2000 to 2017 from the top 20 1997-2000 data
# set
top.names1997.8.9.20[!top.names1997.8.9.20 %in% head(names2017$name,
    n = 100)]
[1] "Ashley"  "Alexis"  "Jessica" "Taylor"  "Lauren"  "Alyssa"  "Kayla"
[8] "Brianna" "Megan"
```

Answer: There are 9 names that did not make it top the 2017 top 100 list from the top 20 consistent names during the period 1997 to 2000.

## (15 pts) Question 3

For the values of the variable `n` in year 2017 how many names (consider female and male names all together) have their `n` values that are far from the majority of the values for `n` in 2017? Use the rule that we discussed

in lecture for identifying outliers to find out the number of outliers for the `n` values (you do not need to print out these values). What percentage of the `n` values in year 2017 are outliers?

```
year2017 = babynames$year %in% 2017  # Extracts out the data for year 2017
data2017 = babynames[year2017, ]  # Assigns that data to `data2017`
```

```
summary(data2017$n)  # Outputs the 5 number summary for the `n` values of data2017
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
    5.0     7.0    12.0   109.2    30.0 19738.0
# Assigns the IQR for the `n` values of data2017 to
# `IQR.2017`
IQR.2017 = IQR(data2017$n, na.rm = T)
IQR.2017  # Outputs the IQR for the `n` values of data2017
[1] 23
30 + 1.5 * IQR.2017  # Used to identify upper outliers using the 3rd Quartile value `30`
[1] 64.5
# There are no lower outliers since Q1 = 7 and
# 7-34.5 is less than the min value of 5
```

```
outliers = (data2017$n >= 30 + 1.5 * IQR.2017)  # Assigns the upper outliers to `outliers`
# Extracts out the corresponding logical values for
# `outliers` and assigns them to `outliers2017`
outliers2017 = data2017[outliers == TRUE, ]
```

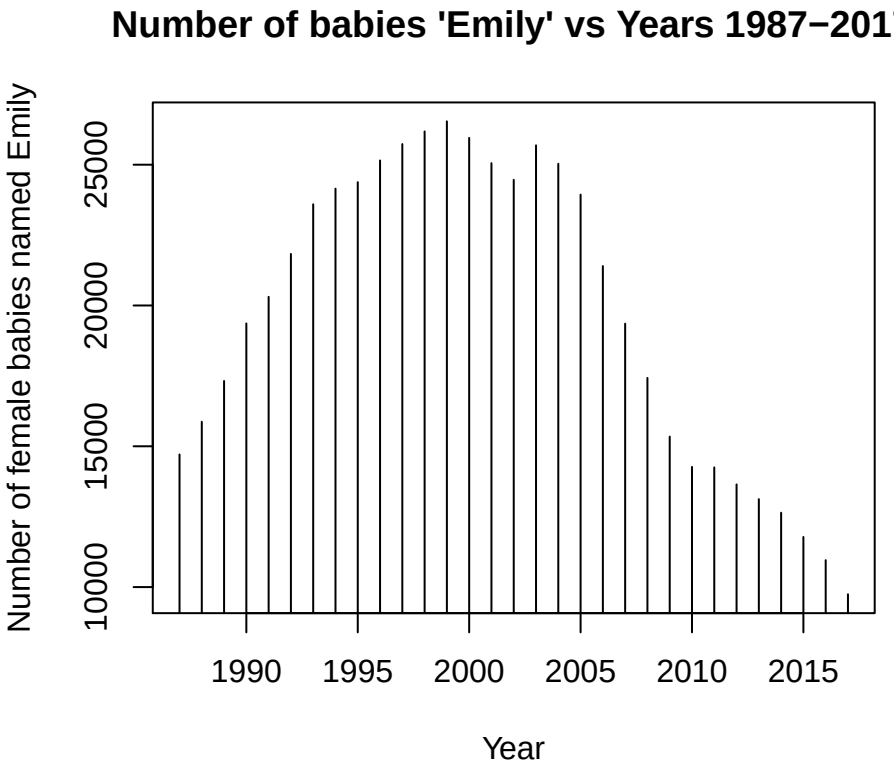
```
# Outputs the length of the `n` column within
# `outliers2017` and assigns it to `n.outliers2017
n.outliers2017 = length(outliers2017$n)
# Outputs the fraction of `n` values in 2017 that
# are outliers
n.outliers2017/length(data2017$n)
[1] 0.1473713
```

Answer: 14.73713% of the `n` values in year 2017 are outliers. This was determined using the 1.5*IQR rule for determining outliers in a given data set.

## (10 pts) Question 4

You are going to meet Emily tomorrow. You do not know anything about Emily except her first name, her gender, that the age she is turning this year is between 2 and 32, inclusively, and that she was born in the U.S. What is the most likely year that she was born (you can assume that the mortality rate for the people in this age group is negligible)?

```
# Extracts out the data from year 1987 to year 2017
years_87_17 = babynames$year %in% c(1987:2017)
data_87_17 = babynames[years_87_17, ]  # Converts `years_87_17` back into a data frame
female_baby = data_87_17$sex == "F"  # Extracts out all the females from `data_87_17`
females_87_17 = data_87_17[female_baby, ]  # Converts `data_87_17` back into a data frame
Emily_87_17 = females_87_17[females_87_17$name == "Emily",
    ]  # Extracts out all Emilies
# Plots the number of Emilies vs years
plot(Emily_87_17$year, Emily_87_17$n, type = "h", xlab = "Year",
    ylab = "Number of female babies named Emily", main = "Number of babies 'Emily' vs Years 1987-2017")
```




Answer: Emily was most likely born in 1999, according to the plot generated.

## Something for you to think about

You do not need to turn in anything for this part. Here is just a question for you to think about: The students in this class are from a population different from the population recorded in the `babynames` dataset; why? Think about whom in the class were recorded in the dataset and whom were not. As a result the names that were popular according to the dataset for the period that we were interested in might not necessarily be popular among the students in our class.