STAT346: Statistical Data Science I

Midterm Exam: Monday, November 2, 2020, 07:00–09:30 p.m.

ShinBaekROk / 2019150445

Instructions

1. This exam covers material from **Introduction to Data Science** (https://rafalab.github.io/dsbook/), Chapter 1–19.
2. You may use any books or online resources you want during this examination, but you may not communicate with any person other than your examiner.
3. You are required to use the RStudio IDE for this exam. You may use either the desktop edition or rstudio.cloud as you prefer.
4. You should work on the provided exam template. When you finalize your exam, you should submit your paper in pdf as well as its .rmd source file. They should have the following name:
   - `stat346_mid_yourID.pdf`
   - `stat346_mid_yourID.rmd`
5. You should submit your paper no later than 9:30 p.m. If you are late, you will get 20% penalty per 10 minutes.

Problem Set #1 (10 Points)

Using the famous `Galton` data set from the `mosaicData` package:

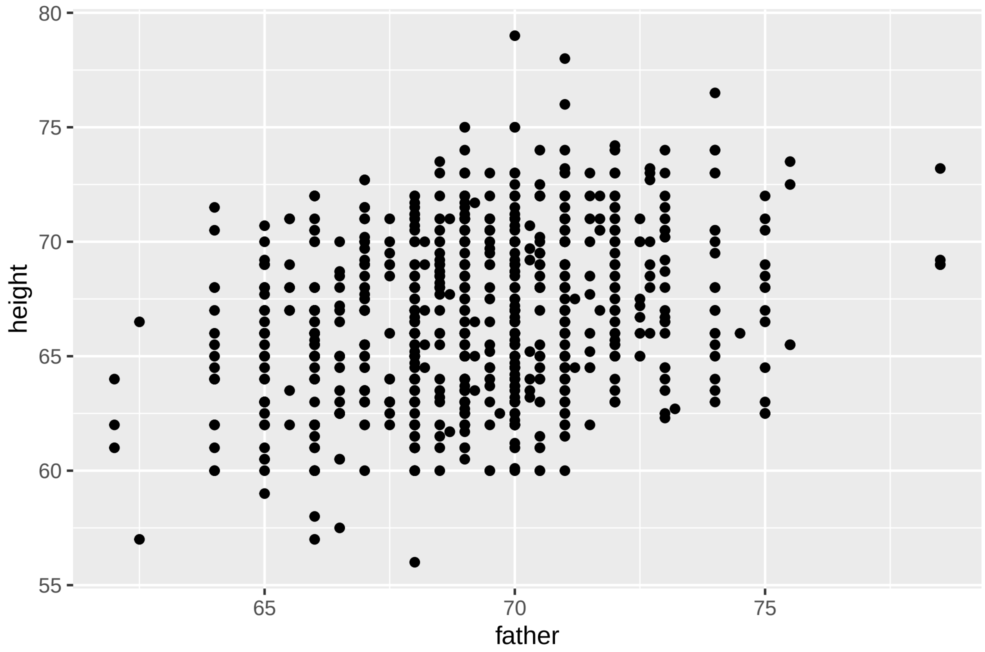
```
library(mosaic)
data(Galton)
```

Use the `ggplot2` package to answer the followings:

(a) [5 points] Create a scatterplot of each person's height against their father's height.

- (sol)

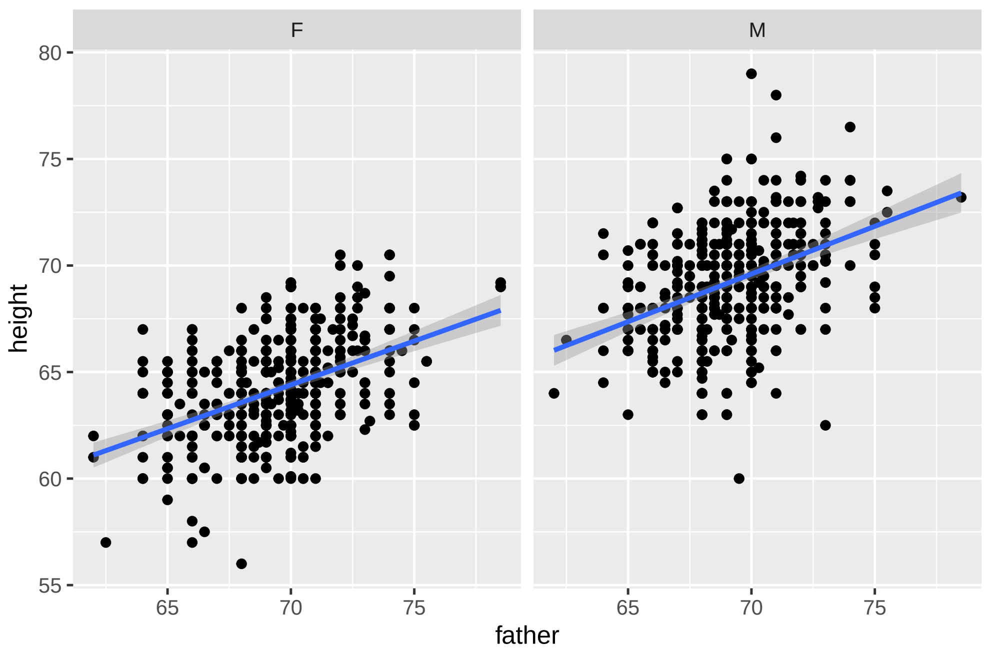
```r
Galton %>% ggplot(aes(y=height,x=father))+geom_point()
```



(b) [5 points] Seprate your plot into facets by sex. Add regression lines to all of your facets.

- (sol)

```r
Galton %>% ggplot(aes(y=height,x=father))+geom_point()+facet_grid(.~sex)+geom_smooth(method=
```



Problem Set #2 (15 Points)

The file `ranking.csv` contains two columns:

- The ID of an item being rated.
- A rating, which is one of `negative`, `positive`, `indifferent`, or `wtf` (meaning the respondent didn't understand the question).

There are multiple ratings for each item. The plot below shows this data:

- Each dot represents one item `i`.
- The size of the circles shows the total number of ratings for item `i`.
- The `X` coordinate for item `i` is the percentage of ratings for that item that are `negative`.
- The `Y` coordinate for item `i` is the percentage of ratings for that item that are `positive`.
- The regression line is created using the `lm` method.

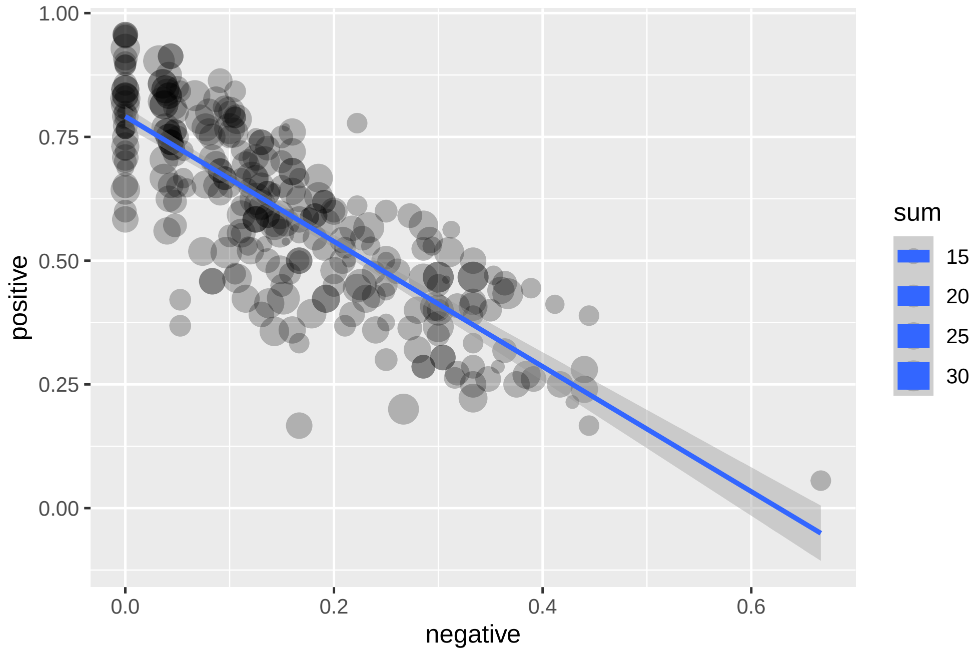
Re-create this plot using the `tidyverse` and `ggplot2`, fixing any mistakes you notice along the way.

- (sol)

```
ranking<-read_csv('C:/ranking.csv')
x<-ranking %>% group_by(rank, item) %>% summarize(n=n()) %>%
   spread(key='rank',value='n')
x$negative[is.na(x$negative)]<-0
x$positive[is.na(x$positive)]<-0
x$indifferent[is.na(x$indifferent)]<-0
x$wtf[is.na(x$wtf)]<-0
x<-x %>% mutate(neg_rate=negative/(negative+positive+indifferent+wtf),pos_rate=positive/(negat
x
```

```
## # A tibble: 284 x 8
##     item indifferent negative positive   wtf neg_rate pos_rate   sum
##    <dbl>       <dbl>    <dbl>    <dbl> <dbl>    <dbl>    <dbl> <dbl>
##  1     1           7        0       12     1  0        0.6        20
##  2     2           4        1       16     0  0.0476   0.762      21
##  3     3           7       11        7     0  0.44     0.28       25
##  4     4           3        4       18     0  0.16     0.72       25
##  5     5           4        0       14     2  0        0.7        20
##  6     6           0        1       21     1  0.0435   0.913      23
##  7     7           1        2       16     1  0.1      0.8        20
##  8     9           4        4        8     0  0.25     0.5        16
##  9    10           3        1       16     0  0.05     0.8        20
## 10    12           7        9       14     0  0.3      0.467      30
## # ... with 274 more rows
```

```
x %>% ggplot(aes(x=neg_rate,y=pos_rate, size=sum))+geom_point(alpha=.25)+geom_smooth(method='lm
```



Problem Set #3 (20 Points)

Read the file `measurements.csv` to create a tibble called `measurements`. (The strings `rad`, `sal`, and `temp` in the quantity column stand for `radiation`, `salinity`, and `temperature`, respectively.)

(a) [5 points] Create a tibble containing only rows where none of the values are `NA` and save in a tibble called `cleaned`.

- (sol)

```
measurements<-read_csv('C:/measurements.csv')
cleaned<- measurements %>% filter(!is.na(visitor) & !is.na(reading))# NA are only in visitor a
cleaned
```

```
## # A tibble: 18 x 4
##    visit_id visitor quantity reading
##       <dbl> <chr>   <chr>      <dbl>
##  1      619 dyer    rad         9.82
##  2      619 dyer    sal         0.13
##  3      622 dyer    rad         7.8
##  4      622 dyer    sal         0.09
##  5      734 pb      rad         8.41
##  6      734 lake    sal         0.05
```

```
##  7      734 pb      temp      -21.5
##  8      735 pb      rad         7.22
##  9      751 pb      rad         4.35
## 10      751 pb      temp      -18.5
## 11      752 lake    rad         2.19
## 12      752 lake    sal         0.09
## 13      752 lake    temp      -16
## 14      752 roe     sal        41.6
## 15      837 lake    rad         1.46
## 16      837 lake    sal         0.21
## 17      837 roe     sal        22.5
## 18      844 roe     rad        11.2
```

(b) [5 points] Count the number of measurements of each type of quantity in `cleaned`. Your result should have one row for each quantity `rad`, `sal`, and `temp`.

- (sol)

```r
rad<-cleaned %>% filter(quantity=='rad') %>% summarize(n=n()) %>% pull(n)

sal<-cleaned %>% filter(quantity=='sal') %>% summarize(n=n()) %>% pull(n)

temp<-cleaned %>% filter(quantity=='temp') %>% summarize(n=n()) %>% pull(n)

data.frame(number=c(rad=rad,sal=sal,temp=temp))
```

```
##      number
## rad       8
## sal       7
## temp      3
```

(c) [5 points] Display the minimum and maximum value of reading separately for each quantity in `cleaned`. Your result should have one row for each quantity `rad`, `sal`, and `temp`.

- (sol)

```r
rad_max<-cleaned %>% filter(quantity=='rad') %>% .$reading %>% max()
rad_min<-cleaned %>% filter(quantity=='rad') %>% .$reading %>% min()

sal_max<-cleaned %>% filter(quantity=='sal') %>% .$reading %>% max()
sal_min<-cleaned %>% filter(quantity=='sal') %>% .$reading %>% min()

temp_max<-cleaned %>% filter(quantity=='temp') %>% .$reading %>% max()
temp_min<-cleaned %>% filter(quantity=='temp') %>% .$reading %>% min()

x<-data.frame(max=c(rad_max,sal_max,temp_max),min=c(rad_min,sal_min,temp_min))
rownames(x)<-c('rad','sal','temp')
x
```

```
##          max    min
## rad   11.25   1.46
## sal   41.60   0.05
## temp -16.00 -21.50
```

(d) [5 points] Create a tibble in which all salinity (`sal`) readings greater than 1 are divided by 100. (This is needed because some people wrote percentages as numbers from 0.0 to 1.0, but others wrote them as 0.0 to 100.0.)

- (sol)

```
cleaned %>%  filter(quantity=='sal' & reading>1) %>% mutate(reading=reading/100) %>% pull(readi
```

```
## [1] 0.416 0.225
```

```
cleaned[14,4]<-0.416
cleaned[17,4]<-0.225
cleaned
```

```
## # A tibble: 18 x 4
##    visit_id visitor quantity reading
##       <dbl> <chr>   <chr>      <dbl>
##  1      619 dyer    rad        9.82
##  2      619 dyer    sal        0.13
##  3      622 dyer    rad        7.8
##  4      622 dyer    sal        0.09
##  5      734 pb      rad        8.41
##  6      734 lake    sal        0.05
##  7      734 pb      temp     -21.5
##  8      735 pb      rad        7.22
##  9      751 pb      rad        4.35
## 10      751 pb      temp     -18.5
## 11      752 lake    rad        2.19
## 12      752 lake    sal        0.09
## 13      752 lake    temp     -16
## 14      752 roe     sal        0.416
## 15      837 lake    rad        1.46
## 16      837 lake    sal        0.21
## 17      837 roe     sal        0.225
## 18      844 roe     rad       11.2
```

Problem Set #4 (35 Points)

For this problem, we will be using the data from the survey collected by the United States National Center for Health Statistics (NCHS). This center has conducted a series of health and nutrition surveys since the 1960's. Starting in 1999, about 5,000 individuals of all ages have been interviewed every year and they complete the health examination component of the survey.

Part of the data is made available via the NHANES package. Once you install the NHANES package, you can load the data like this:

```
library(NHANES)
data(NHANES)
```

Let's now explore the NHANES data.

(a) [5 points] We will provide some basic facts about blood pressure. First let's select a group to set the standard. We will use 20-to-29-year-old females. AgeDecade is a categorical variable with these ages. Note that the category is coded like " 20-29", with a space in front! What is the average and standard deviation of systolic blood pressure as saved in the BPSysAve variable? Save it to a variable called ref.

- (sol)

```
ref<-NHANES %>% filter(AgeDecade==' 20-29', Gender=='female', !is.na(BPSysAve)) %>% summarize(m
ref
```

```
## # A tibble: 1 x 2
##    mean    sd
##   <dbl> <dbl>
## 1  108.  10.1
```

(b) [5 points] Using a pipe, assign the average to a numeric variable ref_avg.

- (sol)

```
ref_avg <-ref %>% pull(mean)
```

(c) [5 points] Now report the min and max values for the same group.

- (sol)

```
NHANES %>% filter(AgeDecade==' 20-29', Gender=='female', !is.na(BPSysAve)) %>% summarize(min=mi
```

```
## # A tibble: 1 x 2
##     min   max
##   <int> <int>
## 1    84   179
```

(d) [5 points] Compute the average and standard deviation for females, but for each age group separately rather than a selected decade as in (a). Note that the age groups are defined by `AgeDecade`.

- (sol)

```
NHANES %>% filter(Gender=='female' & !is.na(BPSysAve)) %>% group_by(AgeDecade) %>% summarize(me
```

```
## # A tibble: 9 x 3
##   AgeDecade  mean    sd
##   <fct>     <dbl> <dbl>
## 1 " 0-9"     100.  9.07
## 2 " 10-19"   104.  9.46
## 3 " 20-29"   108. 10.1
## 4 " 30-39"   111. 12.3
## 5 " 40-49"   115. 14.5
## 6 " 50-59"   122. 16.2
## 7 " 60-69"   127. 17.1
## 8 " 70+"     134. 19.8
## 9  <NA>      142. 22.9
```

(e) [5 points] Repeat (d) for males.

- (sol)

```
NHANES %>% filter(Gender=='male' & !is.na(BPSysAve)) %>% group_by(AgeDecade) %>% summarize(mea
```

```
## # A tibble: 9 x 3
##   AgeDecade  mean    sd
##   <fct>     <dbl> <dbl>
## 1 " 0-9"     97.4  8.32
## 2 " 10-19"  110.  11.2
## 3 " 20-29"  118.  11.3
## 4 " 30-39"  119.  12.3
## 5 " 40-49"  121.  14.0
## 6 " 50-59"  126.  17.8
## 7 " 60-69"  127.  17.5
## 8 " 70+"    130.  18.7
## 9  <NA>     136.  23.5
```

(f) [5 points] We can actually combine both summaries for (d) and (e) into one line of code. This is because `group_by` permits us to group by more than one variable. Obtain one big summary table using `group_by(AgeDecade, Gender)`.

- (sol)

```r
NHANES %>% filter(!is.na(BPSysAve)) %>% group_by(AgeDecade,Gender) %>% summarize(mean=mean(BPSy
```

```
## # A tibble: 18 x 4
## # Groups:   AgeDecade [9]
##    AgeDecade Gender  mean    sd
##    <fct>     <fct>  <dbl> <dbl>
##  1 " 0-9"    female 100.   9.07
##  2 " 0-9"    male    97.4  8.32
##  3 " 10-19"  female 104.   9.46
##  4 " 10-19"  male   110.  11.2
##  5 " 20-29"  female 108.  10.1
##  6 " 20-29"  male   118.  11.3
##  7 " 30-39"  female 111.  12.3
##  8 " 30-39"  male   119.  12.3
##  9 " 40-49"  female 115.  14.5
## 10 " 40-49"  male   121.  14.0
## 11 " 50-59"  female 122.  16.2
## 12 " 50-59"  male   126.  17.8
## 13 " 60-69"  female 127.  17.1
## 14 " 60-69"  male   127.  17.5
## 15 " 70+"    female 134.  19.8
## 16 " 70+"    male   130.  18.7
## 17  <NA>     female 142.  22.9
## 18  <NA>     male   136.  23.5
```

(g) [5 points] For males between the ages of 40-49, compare systolic blood pressure across race as reported in the `Race1` variable. Order the resulting table from lowest to highest average systolic blood pressure.

- (sol)

```r
NHANES %>% filter(Gender == "male" & AgeDecade == " 40-49" & !is.na(BPSysAve)) %>%
    group_by(Race1) %>%
    summarize(mean=mean(BPSysAve),sd=sd(BPSysAve)) %>%
    arrange(mean)
```

```
## # A tibble: 5 x 3
##   Race1     mean    sd
##   <fct>    <dbl> <dbl>
## 1 White     120.  13.4
## 2 Other     120.  16.2
## 3 Hispanic  122.  11.1
## 4 Mexican   122.  13.9
## 5 Black     126.  17.1
```

Problem Set #5 (20 Points)

(a) [5 points] Two teams, say the Celtics and the Cavs, are playing a seven game series. The Cavs are a better team and have a 60% chance of winning each game. What is the probability that the Celtics win **at least** one game? Calculate it by hand.

- (sol)

```
Cavs_win_prob<-0.6
1-(Cavs_win_prob)^4
```

```
## [1] 0.8704
```

(b) [5 points] Create a Monte Carlo simulation to confirm your answer to the previous problem. Use `B <- 10000` simulations.

- (sol)

```
B<-10000
result<-replicate(B,{
  x<-sample(c(0,1),4,replace=TRUE,prob=c(0.6,0.4))
  sum(x)>=1
})
mean(result)
```

```
## [1] 0.8697
```

(c) [5 points] Again, two teams are playing a seven game championship series. The first to win four games, therefore, wins the series. Now, the teams are equally good so they each have a 50-50 chance of winning each game. If the Cavs lose the first game, what is the probability that they win the series? Calculate it by hand.

- (sol)

```
expand.grid(rep(list(0:1),6)) %>% rowSums(.)>=4->x
mean(x)
```

```
## [1] 0.34375
```

(d) [5 points] Confirm the results of the previous question with a Monte Carlo simulation. Use `B <- 10000` simulations.

- (sol)

```r
B<-10000
result<-replicate(B,{
   x<-sample(c(0,1),6,replace=TRUE,prob=c(0.5,0.5))
   sum(x)>=4
})
mean(result)
```

```
## [1] 0.3274
```

Problem Set #6 (20 Points)

Bootstrapping is any test or metric that uses random sampling with replacement and falls under the broader class of resampling methods. This technique allows estimation of the sampling distribution of almost any statistic using random sampling methods.

For illustration, read the following code:

```r
library(tidyverse)
d=tibble(x=c(1,2,3,4,5,6,7,8,9,10),
         y=c(7,3,4,2,10,1,9,8,5,6))
cor(d$x,d$y)
```

```
## [1] 0.2242424
```

The correlation between `x` and `y` is given by 0.224. Now suppose that we wish to find its standard error (SE), which is in fact not a simple task. Bootstrapping comes to its rescue, providing approximated SE estimates.

```r
B = 1000; n=nrow(d)
cor_boot=replicate(B,{
    d_temp = sample_n(d,n,replace=TRUE)
    cor(d_temp$x,d_temp$y)
})
sd(cor_boot)
```

```
## [1] 0.2353844
```

With this idea, answer the followings for `Gestation` data set from the `mosaicData` package.

```r
library(mosaicData)
data(Gestation)
```

(a) [6 points] Calculate and interpret a 95% confidence interval for the mean age of mothers.

- (sol)

```r
mean<-Gestation$age %>% mean(na.rm=T)
sd<-Gestation$age %>% sd(na.rm=T)
se<-sd/sqrt(length(Gestation$age))
CI<-c(mean-qnorm(0.975)*se,mean+qnorm(0.975)*se)
CI
```

```
## [1] 26.93296 27.57758
```

when we were to take repeated samples and construct for 95%CI for each sample,then 95% of these intervals would be expected to contain the true mean value.

(b) [6 points] Use the bootstrap to generate and interpret a 95% confidence interval for the median age of mothers.

- (sol)

```r
B=1000
median<-replicate(B,{
  x<-sample(Gestation$age,nrow(Gestation),replace=TRUE)
  median(x,na.rm=T)
})
sd<-sd(median)
sd
```

```
## [1] 0.4513574
```

```r
med<-Gestation$age %>% median(na.rm=T)
CI95<-c(med-qnorm(0.975)*sd,med+qnorm(0.975)*sd)
CI95
```

```
## [1] 25.11536 26.88464
```

when we were to take repeated samples and construct for 95%CI for each sample,then 95% of these intervals would be expected to contain the true median value. when we were to take repeated samples and construct for 95%CI for each sample,then 95% of these intervals would be expected to contain the true median value.

(d) [8 points] Use the bootstrap to generate a 95% confidence interval for the regression parameters in a model for weight (`wt`) as a function of `age`.

- (sol)

```r
B<-1000
reg<-replicate(B,{

   dat<-sample(Gestation,nrow(Gestation),replace=TRUE)
   mu_x<-dat %>% summarize(m=mean(age,na.rm=T)) %>% pull(m)
   mu_y<-dat %>% summarize(m=mean(wt,na.rm=T))%>% pull(m)
   s_x<-dat %>% summarize(s=sd(age,na.rm=T))%>% pull(s)
   s_y<-dat %>% summarize(s=sd(wt,na.rm=T))%>% pull(s)
   r<-dat %>% summarize(c=cor(age,wt ,use="complete.obs"))%>% pull(c)
   r*s_y/s_x
})
CI<-c(mean(reg)-sd(reg)*qnorm(0.975),mean(reg)+sd(reg)*qnorm(0.975))
CI
```

```
## [1] -0.08002906  0.28307685
```