STAT UN2102 Midterm

Name and UNI

3/01/2018

The STAT UN2102 Spring 2018 Midterm is open notes, open book(s), open computer and online resources are allowed. Students are **not** allowed to communicate with any other people regarding the exam with the exception of the instructor (Gabriel Young) and TA (Elliot Gordon). This includes emailing fellow students, using WeChat and other similar forms of communication. If there is any suspicion of one or more students cheating, further investigation will take place. If students do not follow the guidelines, they will receive a zero on the exam and potentially face more severe consequences. The exam will be posted on Canvas at 4:05PM. Students are required to submit both the .pdf and .Rmd files on Canvas (or .html if you must) by 5:25PM. Late exams will not be accepted. If for some reason you are unable to upload the completed exam on Canvas by 5:25PM, then immediately email markdown file to the course TA (eg2912@columbia.edu).

Part 1 (Outdated Baseball Data)

We consider a dataset of Major League Baseball players who played at least one game in both the 1991 and 1992 seasons, excluding pitchers. This dataset contains the 1992 salaries for that population, along with performance measures for each player from 1991. The four variables are: (1) salary in thousands; (2) batting average; (3) on-base percentage; (4) name of player.

Load in the **Baseball.csv** dataset.

```
setwd("/Users/salimmjahad/Desktop/STAT_COMP/midterm1")
baseball <- read.csv("Baseball.csv",as.is=TRUE)
dim(baseball)
```

```
## [1] 337   4
```

```
names(baseball)
```

```
## [1] "Salary" "AVG"    "OBP"    "Name"
```

Problem 1.1

Write one line of code that accomplishes the same task as the loop below.

```
m <- 0
vec <- baseball$Salary
for (i in 1:length(vec)){
        m <- m + vec[i]/length(vec)
}
m
```

```
## [1] 1248.528
```

```
# Code goes here
mean(vec)
```

```
## [1] 1248.528
```

Problem 1.2

Create a new column in the **baseball** dataframe that contains the players first name. Name the column **FirstName**. Display the head of the updated dataframe.

```
# Code goes here
tmpv <- c()
for (i in 1:dim(baseball["Name"])[1]) {
  tmp <- baseball["Name"][i,]
  tmpv[i] <- strsplit(tmp, split=" ")[[1]][1]
}
baseball["FirstName"] <- tmpv
head(baseball)
```

```
##   Salary   AVG   OBP              Name FirstName
## 1   3300 0.272 0.302 Andre Dawson          Andre
## 2   2600 0.269 0.335 Steve Buchele         Steve
## 3   2500 0.249 0.337 Kal Daniels             Kal
## 4   2475 0.260 0.292 Shawon Dunston       Shawon
## 5   2313 0.273 0.346 Mark Grace             Mark
## 6   2175 0.291 0.379 Ryne Sandberg          Ryne
```

Problem 1.3

Display the first ten most common names.

```
# Code goes here
smtg <- unlist(baseball["FirstName"])
names(smtg) <- NULL
#sum(head(summary(as.factor(smtg)), 10))
common <- names(head(summary(as.factor(smtg)), 10))
common
```

```
##  [1] "Mike"   "Dave"   "Gary"   "John"   "Kevin"  "Jeff"   "Jose"
##  [8] "Mark"   "Carlos" "Chris"
```

Problem 1.4

Use the **grep** function to extract the rows of the players whose last name is **Clark**.

```
# Code goes here
baseball[grep("Clark", baseball$Name),]
```

```
##     Salary   AVG   OBP              Name FirstName
## 133    200 0.228 0.295 Jerald Clark         Jerald
## 139   4275 0.301 0.359 Will Clark             Will
## 167   2900 0.249 0.374 Jack Clark             Jack
```

I understand that if anyone's first name is Clark this would not work.

Problem 1.5

Write a loop that finds the average salary for the top ten most common first names. Display the ten average salaries next to their corresponding names. Note that a player's name should not correlate with thier salary. This task is purely intended to assess iterative coding.

```r
# Code goes here
avgSalary = 0
commonSal <- subset(baseball, baseball$FirstName %in% common)$Salary
for (i in commonSal) {
  avgSalary <- avgSalary + i/length(commonSal)
}
avgSalary
```

```
## [1] 966.8243
```

Part 2 (CDC Cancer Data)

Consider the following dataset **BYSITE.TXT** taken directly from the Center of Disease Control's website. The dataset describes incidence and mortality crude rates of several types of cancer over time and also includes demographic variables such as **RACE** and **SEX**. The variables of interest in this exercise are: **YEAR**, **RACE**, **SITE**, **EVENT_TYPE**, and **CRUDE_RATE**.

Load in the **BYSITE.TXT** dataset. Also look at the levels of the variable **RACE**.

```r
cancer <- read.table("BYSITE.TXT",sep = "|",header=T,
                     na.strings=c("~","."))
dim(cancer)
```

```
## [1] 44982    13
```

```r
names(cancer)
```

```
##  [1] "YEAR"                   "RACE"
##  [3] "SEX"                    "SITE"
##  [5] "EVENT_TYPE"             "AGE_ADJUSTED_CI_LOWER"
##  [7] "AGE_ADJUSTED_CI_UPPER" "AGE_ADJUSTED_RATE"
##  [9] "COUNT"                  "POPULATION"
## [11] "CRUDE_CI_LOWER"         "CRUDE_CI_UPPER"
## [13] "CRUDE_RATE"
```

```r
levels(cancer$RACE)
```

```
## [1] "All Races"                      "American Indian/Alaska Native"
## [3] "Asian/Pacific Islander"         "Black"
## [5] "Hispanic"                       "White"
```

Problem 2.1

Create a new dataframe named **Prostate** that includes only the rows for prostate cancer. Check that the **Prostate** dataframe has 408 rows.

```r
#levels(cancer$SITE)
# Code goes here
Prostate <- cancer[grep("Prostate",cancer$SITE),]
dim(Prostate)
```

```
## [1] 408  13
```

Problem 2.2

Compute the average incidence rate for for each level of **RACE**. To accomplish this task, use the appropriate function from the **apply** family.

```
#levels(cancer$EVENT_TYPE)
# Code goes here
incidents <- subset(Prostate, EVENT_TYPE =="Incidence")
tapply(incidents$CRUDE_RATE, incidents$RACE , mean)
```

```
##                     All Races American Indian/Alaska Native
##                     140.92941                      41.68824
##        Asian/Pacific Islander                         Black
##                      51.67647                     152.40000
##                      Hispanic                         White
##                      53.65882                     141.75882
```

Problem 2.3

Refine the **Prostate** dataframe by removing rows corresponding to **YEAR** level **2010-2014** and removing rows corresponding to **RACE** level **All Races**. After removing the rows, convert **YEAR** into a numeric variable. Check that the new **Prostate** dataframe has 320 rows.

```
#levels(cancer$YEAR)
# Code goes here
New_Prostate <- subset(Prostate, YEAR!="2010-2014" & RACE !="All Races")
New_Prostate$YEAR <- strtoi(New_Prostate$YEAR)
mode(New_Prostate$YEAR)
```

```
## [1] "numeric"
```

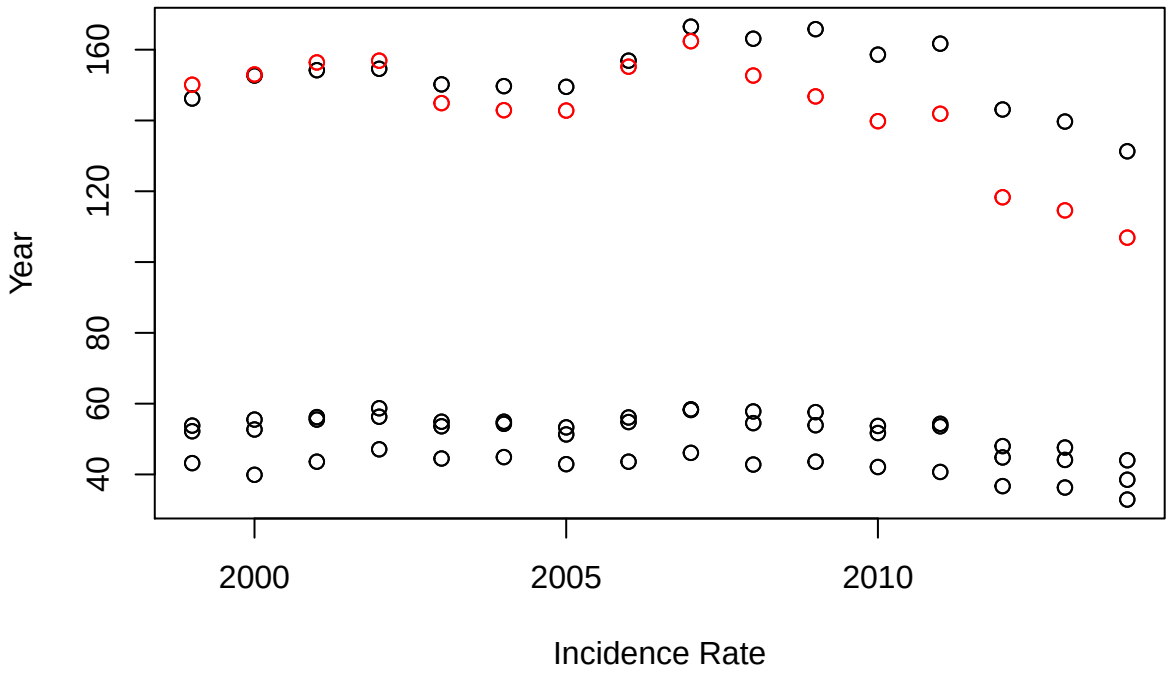
```
dim(New_Prostate)
```

```
## [1] 320  13
```

Problem 2.4

Create a **Base R** plot that shows the incidence rate as a function of time (**YEAR**). Split the scatterplot by whether or not **RACE** is white. Make sure to include a legend and label the graphic appropriately.

```
# Code goes here
New_incidents <- subset(New_Prostate, EVENT_TYPE =="Incidence")
#dim(New_incidents)
coloring = New_incidents$RACE == "White"
#length(coloring)
x <- plot(New_incidents$YEAR, New_incidents$CRUDE_RATE, col = factor(coloring), xlab = "Incidence Rate"
```

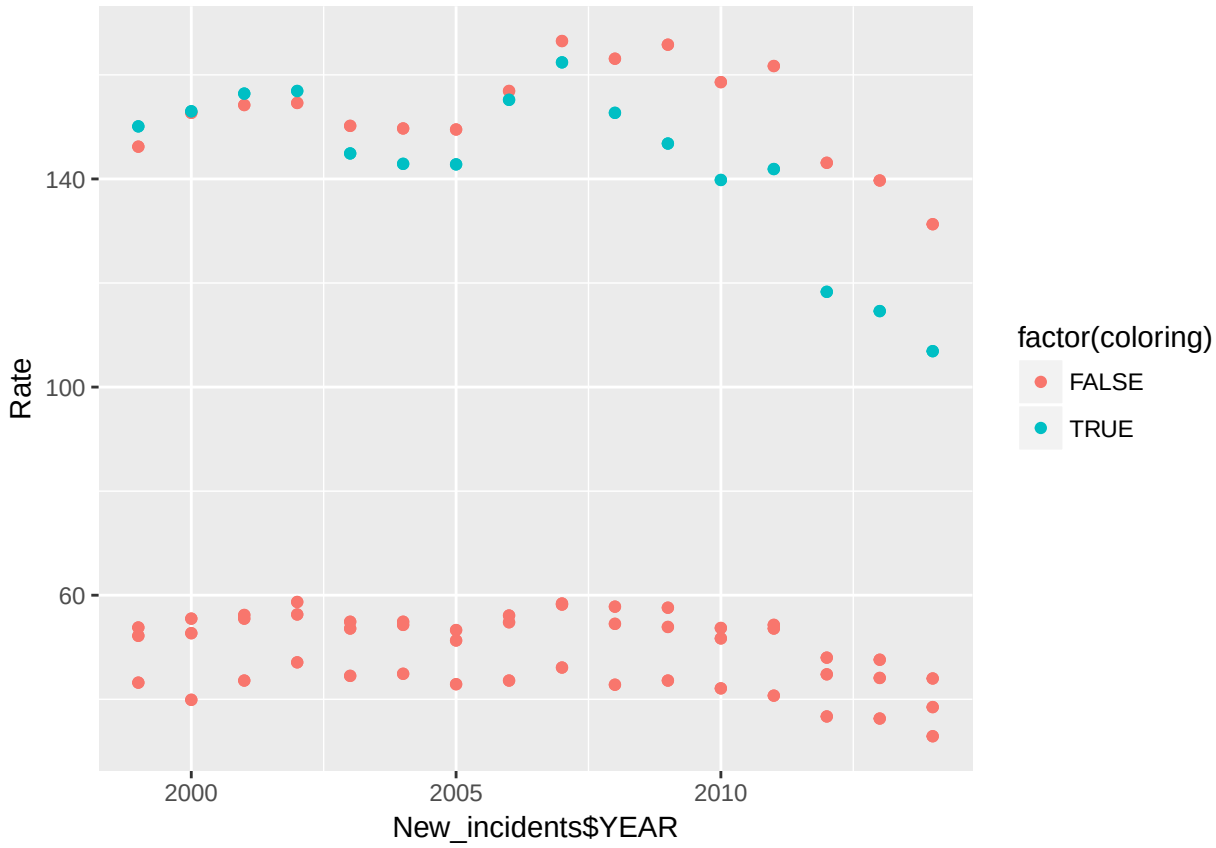


Problem 2.5

Create a **gglpot** plot that shows the crude incidence rate as a function of time (**YEAR**) split by **RACE**. Here you will use the other levels of **RACE**. Also add smoothers (without the confidence bands) for each level of **RACE**. Make sure to include a legend and label the graphic appropriately.

```
# Code goes here
library(ggplot2)
ggplot(data = New_incidents) + geom_point(aes(x = New_incidents$YEAR, y = New_incidents$CRUDE_RATE, col
```

```
## Warning: Ignoring unknown parameters: method, se
```


Rate
140
100
60
2000
2005
2010
New_incidents$YEAR
factor(coloring)
FALSE
TRUE