BIO247 Exam 1

Whitney Domico

2022-09-29

# Part 1 - Written

## Problem 1

Robustness refers to how well the researchers have controlled the experiment to ensure that it (1) is measuring what they actually want it to measure, (2) is assessing for confounding variables, and (3) is designed to be able to be applied to other similar (but not identical) experiments. Reproducibility refers to how well the researchers have (1) dictated the steps in the experiment so that it can be replicated by other researchers and (2) made an experiment that is able to be replicated.

## Problem 2

*on paper exam*

## Problem 3

This study is reproducible. It is a simple survey, and assuming the actual paper gives the exact questions that the researchers put in the survey, it would be very easy to replicate. I would argue that it is not robust, though, because of several reasons. (1) This is a survey, which is generally not a good way to get a random sample because of selection bias. (2) The only people surveyed were undergraduate students from a single university in southeastern USA. This would be difficult to generalize to the whole population of “humans,” since this is a narrow demographic. (3) Perhaps they explain in more in the actual paper; however, it is difficult to find the connection between these survey results and their assertion that “snake outreach and management strategies should account for both cognitive and affective antecedents of behavior.” I will cede that there is some robustness in that this experiment could feasibly be replicated with other demographic groups to make it more robust; however, it does not seem like these researchers plan to continue with other groups.

## Problem 4

This study seems reproducible in that it lays out exactly what steps the researchers performed. They do leave out some information (“few preprocessing steps”, “a model was created”), which could perhaps make it less reproducible. I would hope that they also published their code, in which case it would definitely be reproducible. This study also seems robust. They discuss that they used nine different feature selection models and eight different classification methods to choose the best one, which gives them merit. Also, they tested several different ratios and noted that they tried to avoid overfitting. They also ran significant statistical analyses to make sure that the data was not “obtained by chance,” which helps to get rid of the worry of having a confounding variable. It is also robust in that, even though these researchers were looking at liver cancer, it is feasible to assume that this method may also be applied to other types of cancers.

# Part 2 - Practical

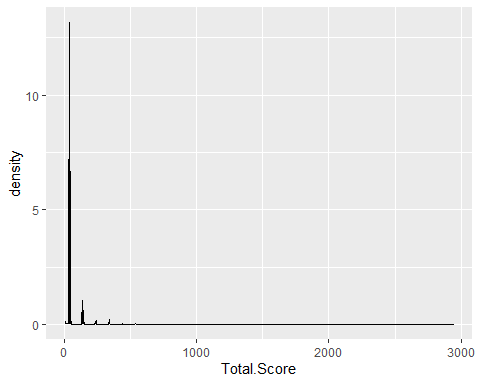
## Problem 5

I can make a vector that just includes “\_1” and “\_0” and then use the paste function to “add” the two vectors into a new vector (called intsuff)

interact <- rep("interactions",25)  
suffixes <- c("\_1", "\_0")  
intsuff <- paste(interact, suffixes, sep="")

## Problem 6

ggplot(GBM\_data)+geom\_density(aes(x=Total.Score), fill="black")



## Problem 7

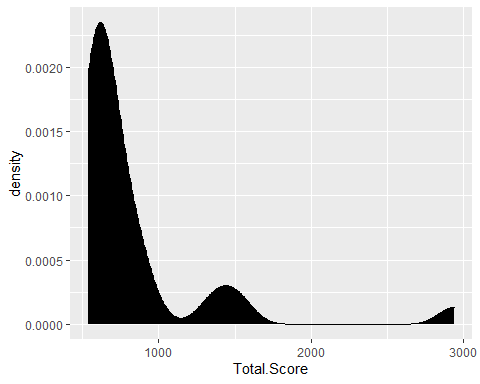
The density plot from Problem 6 appears to change form after about Total.Score = 440 (in other words, there seems to be less high “spikes” after this cutoff), so I’ve decided to only include Total.Scores that are greater than 440. Looking at the dataset, that includes rows 1 through 29.

Part of this solution was sourced from the Rmd created for the lab where we had to submit the exercises in a markdown file (Lab 3).

GBMSubset <- GBM\_data[1:29,]

ggplot just to see if the threshold makes sense

ggplot(GBMSubset)+geom\_density(aes(x=Total.Score), fill="black")



## Problem 8

Part of this solution was sourced from the script for the filtering code in Lab 4.

removing junk characters

GBMSubset$Paper.ID <- gsub("\\]", "", GBMSubset$Paper.ID)  
GBMSubset$Paper.ID <- gsub("\\[", "", GBMSubset$Paper.ID)  
GBMSubset$Paper.ID <- gsub("\\'", "", GBMSubset$Paper.ID)  
GBMSubset$Paper.ID <- gsub(" ", "", GBMSubset$Paper.ID)

making a column that is the number of unique papers for each row

for (each in nrow(GBMSubset):1){  
 uniquepaperID <- length(unique(unlist(strsplit(GBMSubset$Paper.ID[each], ","))))  
 GBMSubset$Num.Unique.Paper.ID <-uniquepaperID  
}

because each cell =1, I want a new column that is just the paper ID for each row

for (each in nrow(GBMSubset):1){  
 GBMSubset$ActualPaperID[each] <- unlist(strsplit(GBMSubset$Paper.ID[each], ","))  
}

## Warning in GBMSubset$ActualPaperID[each] <-  
## unlist(strsplit(GBMSubset$Paper.ID[each], : number of items to replace is not a  
## multiple of replacement length  
  
## Warning in GBMSubset$ActualPaperID[each] <-  
## unlist(strsplit(GBMSubset$Paper.ID[each], : number of items to replace is not a  
## multiple of replacement length  
  
## Warning in GBMSubset$ActualPaperID[each] <-  
## unlist(strsplit(GBMSubset$Paper.ID[each], : number of items to replace is not a  
## multiple of replacement length  
  
## Warning in GBMSubset$ActualPaperID[each] <-  
## unlist(strsplit(GBMSubset$Paper.ID[each], : number of items to replace is not a  
## multiple of replacement length  
  
## Warning in GBMSubset$ActualPaperID[each] <-  
## unlist(strsplit(GBMSubset$Paper.ID[each], : number of items to replace is not a  
## multiple of replacement length  
  
## Warning in GBMSubset$ActualPaperID[each] <-  
## unlist(strsplit(GBMSubset$Paper.ID[each], : number of items to replace is not a  
## multiple of replacement length  
  
## Warning in GBMSubset$ActualPaperID[each] <-  
## unlist(strsplit(GBMSubset$Paper.ID[each], : number of items to replace is not a  
## multiple of replacement length  
  
## Warning in GBMSubset$ActualPaperID[each] <-  
## unlist(strsplit(GBMSubset$Paper.ID[each], : number of items to replace is not a  
## multiple of replacement length  
  
## Warning in GBMSubset$ActualPaperID[each] <-  
## unlist(strsplit(GBMSubset$Paper.ID[each], : number of items to replace is not a  
## multiple of replacement length  
  
## Warning in GBMSubset$ActualPaperID[each] <-  
## unlist(strsplit(GBMSubset$Paper.ID[each], : number of items to replace is not a  
## multiple of replacement length  
  
## Warning in GBMSubset$ActualPaperID[each] <-  
## unlist(strsplit(GBMSubset$Paper.ID[each], : number of items to replace is not a  
## multiple of replacement length  
  
## Warning in GBMSubset$ActualPaperID[each] <-  
## unlist(strsplit(GBMSubset$Paper.ID[each], : number of items to replace is not a  
## multiple of replacement length  
  
## Warning in GBMSubset$ActualPaperID[each] <-  
## unlist(strsplit(GBMSubset$Paper.ID[each], : number of items to replace is not a  
## multiple of replacement length  
  
## Warning in GBMSubset$ActualPaperID[each] <-  
## unlist(strsplit(GBMSubset$Paper.ID[each], : number of items to replace is not a  
## multiple of replacement length  
  
## Warning in GBMSubset$ActualPaperID[each] <-  
## unlist(strsplit(GBMSubset$Paper.ID[each], : number of items to replace is not a  
## multiple of replacement length  
  
## Warning in GBMSubset$ActualPaperID[each] <-  
## unlist(strsplit(GBMSubset$Paper.ID[each], : number of items to replace is not a  
## multiple of replacement length  
  
## Warning in GBMSubset$ActualPaperID[each] <-  
## unlist(strsplit(GBMSubset$Paper.ID[each], : number of items to replace is not a  
## multiple of replacement length  
  
## Warning in GBMSubset$ActualPaperID[each] <-  
## unlist(strsplit(GBMSubset$Paper.ID[each], : number of items to replace is not a  
## multiple of replacement length  
  
## Warning in GBMSubset$ActualPaperID[each] <-  
## unlist(strsplit(GBMSubset$Paper.ID[each], : number of items to replace is not a  
## multiple of replacement length  
  
## Warning in GBMSubset$ActualPaperID[each] <-  
## unlist(strsplit(GBMSubset$Paper.ID[each], : number of items to replace is not a  
## multiple of replacement length  
  
## Warning in GBMSubset$ActualPaperID[each] <-  
## unlist(strsplit(GBMSubset$Paper.ID[each], : number of items to replace is not a  
## multiple of replacement length  
  
## Warning in GBMSubset$ActualPaperID[each] <-  
## unlist(strsplit(GBMSubset$Paper.ID[each], : number of items to replace is not a  
## multiple of replacement length  
  
## Warning in GBMSubset$ActualPaperID[each] <-  
## unlist(strsplit(GBMSubset$Paper.ID[each], : number of items to replace is not a  
## multiple of replacement length  
  
## Warning in GBMSubset$ActualPaperID[each] <-  
## unlist(strsplit(GBMSubset$Paper.ID[each], : number of items to replace is not a  
## multiple of replacement length  
  
## Warning in GBMSubset$ActualPaperID[each] <-  
## unlist(strsplit(GBMSubset$Paper.ID[each], : number of items to replace is not a  
## multiple of replacement length  
  
## Warning in GBMSubset$ActualPaperID[each] <-  
## unlist(strsplit(GBMSubset$Paper.ID[each], : number of items to replace is not a  
## multiple of replacement length  
  
## Warning in GBMSubset$ActualPaperID[each] <-  
## unlist(strsplit(GBMSubset$Paper.ID[each], : number of items to replace is not a  
## multiple of replacement length  
  
## Warning in GBMSubset$ActualPaperID[each] <-  
## unlist(strsplit(GBMSubset$Paper.ID[each], : number of items to replace is not a  
## multiple of replacement length  
  
## Warning in GBMSubset$ActualPaperID[each] <-  
## unlist(strsplit(GBMSubset$Paper.ID[each], : number of items to replace is not a  
## multiple of replacement length

now I can just find the number of unique papers in the new row

length(unique(GBMSubset$ActualPaperID))

## [1] 21

Therefore, the number of unique papers in my subset is 21.