Benford’s Law

Santonastaso

2022-08-14

# Benford’s Law Tool

Reading through ’Analytics Stories” (Winston, 2021) I was fascinated both by Benford’s Law, and by the fact that I’d never heard of it before. I wanted to conduct a study but I couldn’t really think of a dataset the interested me that would apply. So I just built a calculator that could be used any time. Also of note, I’m psyched because this is the first time I’ve made use of functions instead of blocks of code.

library(readxl)  
library(dplyr)

##   
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':  
##   
## filter, lag

## The following objects are masked from 'package:base':  
##   
## intersect, setdiff, setequal, union

library(Dict)  
library(stringr)  
setwd('C:/Users/tsant/Documents/Data')  
df <- as.data.frame(read\_excel("CountryPopRaw.xlsx"))  
digits <- seq(1,9)  
tally <- nrow(df)   
  
#First function creates our theoretical ideal.  
expected\_Bernhoff\_values\_function <- function(number\_list){  
 theory <- c() #set up list of theoretical values  
 for(i in digits){ #set up loop to go through digits 1 - 9   
 calc <- log10(1+i) \* 100 #calculate the number of expected values for each number  
 if(i == 1){ #special case: '1'; if '1' do the following:  
 theory <- append(theory, calc)  
 calc\_prev <- calc  
 }else{ # for 2 - 9 do the following:  
 theory <- append(theory, calc - calc\_prev)  
 calc\_prev <- calc  
 }  
 }  
 return(theory)  
}  
theory <- expected\_Bernhoff\_values\_function(digits)  
  
#Second function extracts all of the first digits in our population sample  
firsts\_function <- function(df){  
 firsts <- c() #set a variable empty list for all the first numbers  
 for(i in df) { #start loop through data list  
 tempstr <- as.character(i) #turn the number into a character string  
 tempnum <- as.numeric(substr(tempstr, 1, 1))#get the first character (number) and turn it back into a number  
 firsts <- append(firsts, tempnum) #add to my list of first numbers  
 }  
 firsts  
}  
#nums <- c(12, 22, 32, 42, 52, 11)  
firsts <- (firsts\_function(df))  
  
#Third function tallies each unique number in from the second function ('firsts\_function')  
tallies\_function <- function(num\_list){ #set up function to build list of tallies of first digits  
 tally\_list <- c() #set up list of tallies (1st digits)  
 for(i in digits){ #set up loop through digits list  
 temp\_num <- length(num\_list[num\_list == i]) #find number of 1s, 2s, etc...  
 tally\_list <- append(tally\_list, temp\_num) #make list of tallies of each number  
 }  
 return(tally\_list)  
}  
tallies <- (tallies\_function(firsts))  
  
#Fourth function calculates the percentages for each number  
calculate\_sample\_percents\_function <- function(tally\_list){ #set up function  
 percents <- c() #set up percents list   
 for(i in tally\_list){ #iterate through tallies list (from tallies\_function)  
 percents <- append(percents, i/tally\*100) #build list of percents  
 }  
 return(percents)  
}  
percents <- calculate\_sample\_percents\_function(tallies)  
  
#Last stanza: Make our grouped bar garaph  
x <- data.frame("Sample" = percents, "Theory" = theory) #make a dataframe  
z <- do.call(rbind, x) #convert it to a matrix I can use  
barplot(z, beside=TRUE, names.arg = digits, #make a barplot, and identify the names of the groups  
 ylab = "Percent", xlab = "First digit", #label x and y axes  
 legend.text = rownames(z), args.legend = list( #make a legend, and place it on the graph  
 x = "topright", bty = "n"))

