HW01p

Joseph Peltroche

February 19, 2018

Welcome to HW01p where the "p" stands for "practice" meaning you will use R to solve practical problems. This homework is due 11:59 PM Satuday 2/24/18.

You should have RStudio installed to edit this file. You will write code in places marked "TO-DO" to complete the problems. Some of this will be a pure programming assignment. The tools for the solutions to these problems can be found in the class practice lectures. I want you to use the methods I taught you, not for you to google and come up with whatever works. You won't learn that way.

To "hand in" the homework, you should compile or publish this file into a PDF that includes output of your code. Once it's done, push by the deadline.

R Basics

First, install the package `testthat` (a widely accepted testing suite for R) from https://github.com/r-lib/testthat using `pacman`. If you are using Windows, this will be a long install, but you have to go through it for some of the stuff we are doing in class. LINUX (or MAC) is preferred for coding. If you can't get it to work, install this package from CRAN (still using `pacman`), but this is not recommended long term.

```
if (!require("pacman")){install.packages("pacman")}
```

```
## Loading required package: pacman
```

```
pacman::p_load(devtools)
pacman::p_load_gh("testthat")
```

1. Use the `seq` function to create vector `v` consisting of all numbers from -100 to 100.

```
v=seq(-100,100)
```

Test using the following code:

```
expect_equal(v, -100 : 100)
```

If there are any errors, the `expect_equal` function will tell you about them. If there are no errors, then it will be silent.

2. Create a function `my_reverse` which takes as required input a vector and returns the vector in reverse where the first entry is the last entry, etc. No function calls are allowed inside your function (otherwise that would defeat the purpose of the exercise).

```
my_reverse=function(x){
    l=length(x)
    d=x
    if (is.vector(x)!=TRUE){
        print("Error: The input must be a vector")
    } else {
        for (i in 0:(l-1)){
            x[i+1]=d[l-i]
        }
    x
```

```
    }
}
```

Test using the following code:

```
expect_equal(my_reverse(c("A", "B", "C")), c("C", "B", "A"))
expect_equal(my_reverse(v), rev(v))
```

3. Let `n = 50`. Create a nxn matrix `R` of exactly 50% entries 0's, 25% 1's 25% 2's in random locations.

```
n = 50
R=matrix(sample(rep(c(0,0,1,2),625)), nrow = n, ncol = n)
#The nxn matrix has 2500 entries. The value of 625 in the repitition function is the

#repeat the four entries 625 times: 625x4.
```

Test using the following and write two more tests as specified below:

```
#test #1
expect_equal(dim(R), c(n, n))
unique(c(R))
```

```
## [1] 2 1 0
```

```
#test #1
x=0
r=c(R)
for(i in 1:2500){
  if(r[i]==2){
    x=x+1
  }
}
x
```

```
## [1] 625
```

4. Randomly punch holes (i.e. `NA`) values in this matrix so that approximately 30% of the entries are missing.

```
r=c(R)
r_l=length(R)*0.3
R_1=matrix(sample(replace(r,1:r_l,NA)),nrow = n,ncol = n)
```

Test using the following code. Note this test may fail 1/100 times.

```
num_missing_in_R = sum(is.na(c(R_1)))
expect_lt(num_missing_in_R, qbinom(0.995, n^2, 0.3))
expect_gt(num_missing_in_R, qbinom(0.005, n^2, 0.3))
```

5. Sort the rows matrix `R` by the largest row sum to lowest. See 2/3 way through practice lecture 3 for a hint.

```
row_sums=rep(NA,50);
d=matrix(NA,nrow = 50,ncol=50)
for(i in 1:50){
  row_sums[i]=sum(R[i,])
}
rownames(R)=row_sums
row_sorted=rownames(R)[order(rownames(R),decreasing=TRUE)]
```

```
for(i in 1:50){
  if (row_sorted[i]!=rownames(R)[i]){
    d[i,]=R[row_sorted[i],]
  }else{
    d[i,]=R[i,]
  }
}
R=d
R
```

```
##       [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10] [,11] [,12] [,13]
##  [1,]    2    2    0    0    2    1    2    0    2     1     1     2     0
##  [2,]    2    2    0    1    1    2    2    2    2     2     0     0     1
##  [3,]    1    0    1    0    2    2    2    0    2     0     2     2     1
##  [4,]    1    1    2    1    0    0    1    0    2     1     2     0     1
##  [5,]    1    1    2    1    0    0    1    0    2     1     2     0     1
##  [6,]    0    1    0    0    0    1    2    2    2     0     1     0     0
##  [7,]    0    1    0    0    0    1    2    2    2     0     1     0     0
##  [8,]    0    1    0    0    0    1    2    2    2     0     1     0     0
##  [9,]    0    1    0    0    0    1    2    2    2     0     1     0     0
## [10,]    0    1    0    0    0    1    2    2    2     0     1     0     0
## [11,]    0    1    0    0    0    1    2    2    2     0     1     0     0
## [12,]    2    1    0    0    1    2    0    0    2     1     0     1     0
## [13,]    0    0    1    2    0    0    2    1    2     1     0     2     2
## [14,]    0    0    1    2    0    0    2    1    2     1     0     2     2
## [15,]    2    0    1    2    2    1    0    1    1     0     2     1     0
## [16,]    2    0    1    2    2    1    0    1    1     0     2     1     0
## [17,]    2    0    1    2    2    1    0    1    1     0     2     1     0
## [18,]    2    0    1    2    2    1    0    1    1     0     2     1     0
## [19,]    1    2    0    2    1    2    0    1    0     0     1     0     0
## [20,]    1    2    0    2    1    2    0    1    0     0     1     0     0
## [21,]    1    2    0    2    1    2    0    1    0     0     1     0     0
## [22,]    1    0    0    1    1    0    0    2    0     0     1     2     0
## [23,]    1    0    0    1    1    0    0    2    0     0     1     2     0
## [24,]    1    0    0    1    1    0    0    2    0     0     1     2     0
## [25,]    0    0    1    0    0    0    0    0    0     1     2     1     0
## [26,]    2    0    0    0    0    0    0    0    2     0     2     2     0
## [27,]    2    0    0    0    0    0    0    0    2     0     2     2     0
## [28,]    0    1    1    0    0    2    1    0    1     0     2     0     1
## [29,]    2    1    0    0    2    2    2    0    0     0     0     0     2
## [30,]    2    1    0    0    2    2    2    0    0     0     0     0     2
## [31,]    2    1    0    0    2    2    2    0    0     0     0     0     2
## [32,]    1    2    0    2    0    0    1    0    0     0     1     0     0
## [33,]    0    0    1    0    2    0    2    1    0     1     1     1     1
## [34,]    1    0    0    2    0    0    2    1    0     1     1     0     1
## [35,]    1    0    0    2    0    0    2    1    0     1     1     0     1
## [36,]    1    0    0    2    0    0    2    1    0     1     1     0     1
## [37,]    1    0    0    2    0    0    2    1    0     1     1     0     1
## [38,]    1    1    0    1    0    2    2    2    1     1     1     2     0
## [39,]    1    1    0    1    0    2    2    2    1     1     1     2     0
## [40,]    1    1    0    1    0    2    2    2    1     1     1     2     0
## [41,]    0    1    0    2    1    2    2    2    0     0     0     0     0
## [42,]    0    1    0    2    1    2    2    2    0     0     0     0     0
## [43,]    0    1    0    2    1    2    2    2    0     0     0     0     0
```

```
## [44,]    0    0    0    0    1    2    1    1    0     0     1     2     1
## [45,]    0    2    2    0    0    1    0    0    2     0     2     2     0
## [46,]    0    2    2    0    0    1    0    0    2     0     2     2     0
## [47,]    0    0    0    0    2    0    1    0    1     2     1     1     2
## [48,]    0    0    0    0    2    0    1    0    1     2     1     1     2
## [49,]    2    0    1    0    0    1    0    0    0     0     1     0     1
## [50,]    0    0    0    1    0    2    2    0    1     2     0     0     0
##       [,14] [,15] [,16] [,17] [,18] [,19] [,20] [,21] [,22] [,23] [,24]
##  [1,]     0     0     0     2     2     2     0     0     0     1     0
##  [2,]     0     0     0     0     0     0     0     0     0     2     0
##  [3,]     2     0     0     2     1     0     2     1     1     2     1
##  [4,]     1     1     0     0     0     0     1     0     2     0     0
##  [5,]     1     1     0     0     0     0     1     0     2     0     0
##  [6,]     2     0     1     2     2     0     1     0     0     2     1
##  [7,]     2     0     1     2     2     0     1     0     0     2     1
##  [8,]     2     0     1     2     2     0     1     0     0     2     1
##  [9,]     2     0     1     2     2     0     1     0     0     2     1
## [10,]     2     0     1     2     2     0     1     0     0     2     1
## [11,]     2     0     1     2     2     0     1     0     0     2     1
## [12,]     0     2     1     0     2     2     0     0     2     1     2
## [13,]     0     1     2     0     0     0     2     0     2     0     0
## [14,]     0     1     2     0     0     0     2     0     2     0     0
## [15,]     0     2     0     0     1     0     0     0     0     0     0
## [16,]     0     2     0     0     1     0     0     0     0     0     0
## [17,]     0     2     0     0     1     0     0     0     0     0     0
## [18,]     0     2     0     0     1     0     0     0     0     0     0
## [19,]     1     2     1     2     0     0     0     1     0     1     2
## [20,]     1     2     1     2     0     0     0     1     0     1     2
## [21,]     1     2     1     2     0     0     0     1     0     1     2
## [22,]     2     0     1     2     0     2     0     2     0     2     1
## [23,]     2     0     1     2     0     2     0     2     0     2     1
## [24,]     2     0     1     2     0     2     0     2     0     2     1
## [25,]     2     0     2     2     0     1     2     0     0     2     2
## [26,]     1     0     0     2     2     0     1     0     1     0     0
## [27,]     1     0     0     2     2     0     1     0     1     0     0
## [28,]     0     1     1     1     0     0     2     2     0     0     0
## [29,]     0     0     1     2     0     2     0     1     0     2     0
## [30,]     0     0     1     2     0     2     0     1     0     2     0
## [31,]     0     0     1     2     0     2     0     1     0     2     0
## [32,]     0     0     1     0     1     0     1     2     2     0     0
## [33,]     0     0     1     0     0     2     0     0     0     1     0
## [34,]     0     0     0     2     0     0     0     1     0     0     2
## [35,]     0     0     0     2     0     0     0     1     0     0     2
## [36,]     0     0     0     2     0     0     0     1     0     0     2
## [37,]     0     0     0     2     0     0     0     1     0     0     2
## [38,]     0     0     1     0     1     0     1     0     0     0     0
## [39,]     0     0     1     0     1     0     1     0     0     0     0
## [40,]     0     0     1     0     1     0     1     0     0     0     0
## [41,]     0     1     0     0     1     0     1     2     0     0     1
## [42,]     0     1     0     0     1     0     1     2     0     0     1
## [43,]     0     1     0     0     1     0     1     2     0     0     1
## [44,]     0     0     0     0     0     0     1     0     0     0     1
## [45,]     0     1     0     0     0     0     0     1     0     0     2
## [46,]     0     1     0     0     0     0     0     1     0     0     2
```

```
## [47,]    0    0    0    1    1    0    0    0    0    0    2
## [48,]    0    0    0    1    1    0    0    0    0    0    2
## [49,]    2    0    2    0    1    0    1    0    0    1    0
## [50,]    0    1    0    2    1    0    0    0    0    0    0
##       [,25] [,26] [,27] [,28] [,29] [,30] [,31] [,32] [,33] [,34] [,35]
##  [1,]    1    2    2    2    2    2    2    2    1    1    0
##  [2,]    0    1    2    0    2    2    0    2    1    0    2
##  [3,]    1    2    0    1    2    0    1    1    1    0    1
##  [4,]    2    2    2    2    2    2    2    0    0    0    0
##  [5,]    2    2    2    2    2    2    2    0    0    0    0
##  [6,]    2    2    2    1    2    0    0    0    2    1    2
##  [7,]    2    2    2    1    2    0    0    0    2    1    2
##  [8,]    2    2    2    1    2    0    0    0    2    1    2
##  [9,]    2    2    2    1    2    0    0    0    2    1    2
## [10,]    2    2    2    1    2    0    0    0    2    1    2
## [11,]    2    2    2    1    2    0    0    0    2    1    2
## [12,]    1    0    1    0    2    2    1    0    0    0    0
## [13,]    1    1    1    0    2    1    0    0    0    2    1
## [14,]    1    1    1    0    2    1    0    0    0    2    1
## [15,]    1    0    1    2    1    2    2    1    2    0    0
## [16,]    1    0    1    2    1    2    2    1    2    0    0
## [17,]    1    0    1    2    1    2    2    1    2    0    0
## [18,]    1    0    1    2    1    2    2    1    2    0    0
## [19,]    0    0    1    1    0    0    0    0    0    0    1
## [20,]    0    0    1    1    0    0    0    0    0    0    1
## [21,]    0    0    1    1    0    0    0    0    0    0    1
## [22,]    2    0    2    0    0    0    0    2    0    2    0
## [23,]    2    0    2    0    0    0    0    2    0    2    0
## [24,]    2    0    2    0    0    0    0    2    0    2    0
## [25,]    2    0    2    0    0    2    1    1    1    0    0
## [26,]    1    0    2    2    1    0    1    0    1    1    0
## [27,]    1    0    2    2    1    0    1    0    1    1    0
## [28,]    0    2    2    2    0    0    0    0    1    0    2
## [29,]    1    0    0    0    1    0    0    2    0    2    0
## [30,]    1    0    0    0    1    0    0    2    0    2    0
## [31,]    1    0    0    0    1    0    0    2    0    2    0
## [32,]    0    1    0    2    0    2    0    2    2    0    0
## [33,]    0    0    2    0    2    1    2    1    0    0    0
## [34,]    1    2    1    2    0    0    1    0    1    1    2
## [35,]    1    2    1    2    0    0    1    0    1    1    2
## [36,]    1    2    1    2    0    0    1    0    1    1    2
## [37,]    1    2    1    2    0    0    1    0    1    1    2
## [38,]    2    0    0    0    0    0    1    0    1    2    0
## [39,]    2    0    0    0    0    0    1    0    1    2    0
## [40,]    2    0    0    0    0    0    1    0    1    2    0
## [41,]    1    2    1    0    0    1    0    0    0    2    0
## [42,]    1    2    1    0    0    1    0    0    0    2    0
## [43,]    1    2    1    0    0    1    0    0    0    2    0
## [44,]    2    0    0    0    1    0    1    2    0    2    0
## [45,]    0    0    0    1    2    2    2    0    0    2    0
## [46,]    0    0    0    1    2    2    2    0    0    2    0
## [47,]    0    2    1    2    2    0    0    0    0    1    0
## [48,]    0    2    1    2    2    0    0    0    0    1    0
## [49,]    1    0    2    0    0    2    0    0    0    0    2
```

```
## [50,]     0     0     0     0     1     0     0     0     2     0     0
##       [,36] [,37] [,38] [,39] [,40] [,41] [,42] [,43] [,44] [,45] [,46]
##  [1,]     2     1     1     0     2     0     0     1     2     1     2
##  [2,]     2     2     1     0     1     0     1     2     1     0     2
##  [3,]     0     0     1     0     0     0     0     1     1     2     2
##  [4,]     1     0     1     2     0     1     2     0     0     0     1
##  [5,]     1     0     1     2     0     1     2     0     0     0     1
##  [6,]     0     1     0     2     0     1     0     2     0     1     2
##  [7,]     0     1     0     2     0     1     0     2     0     1     2
##  [8,]     0     1     0     2     0     1     0     2     0     1     2
##  [9,]     0     1     0     2     0     1     0     2     0     1     2
## [10,]     0     1     0     2     0     1     0     2     0     1     2
## [11,]     0     1     0     2     0     1     0     2     0     1     2
## [12,]     0     0     1     1     2     1     2     1     1     2     2
## [13,]     1     1     0     0     2     0     2     2     2     0     0
## [14,]     1     1     0     0     2     0     2     2     2     0     0
## [15,]     1     0     0     2     0     0     1     2     2     0     2
## [16,]     1     0     0     2     0     0     1     2     2     0     2
## [17,]     1     0     0     2     0     0     1     2     2     0     2
## [18,]     1     0     0     2     0     0     1     2     2     0     2
## [19,]     0     2     2     0     0     0     2     1     2     1     2
## [20,]     0     2     2     0     0     0     2     1     2     1     2
## [21,]     0     2     2     0     0     0     2     1     2     1     2
## [22,]     1     0     0     1     2     1     0     0     2     2     0
## [23,]     1     0     0     1     2     1     0     0     2     2     0
## [24,]     1     0     0     1     2     1     0     0     2     2     0
## [25,]     2     0     0     1     0     1     1     0     0     2     2
## [26,]     1     1     2     1     0     0     0     2     2     2     0
## [27,]     1     1     2     1     0     0     0     2     2     2     0
## [28,]     0     0     2     0     2     1     0     0     0     0     1
## [29,]     0     2     0     0     0     0     2     0     0     0     2
## [30,]     0     2     0     0     0     0     2     0     0     0     2
## [31,]     0     2     0     0     0     0     2     0     0     0     2
## [32,]     2     0     0     1     2     0     0     0     2     0     0
## [33,]     0     0     2     0     2     0     0     2     0     2     0
## [34,]     0     0     2     0     0     0     2     1     0     0     2
## [35,]     0     0     2     0     0     0     2     1     0     0     2
## [36,]     0     0     2     0     0     0     2     1     0     0     2
## [37,]     0     0     2     0     0     0     2     1     0     0     2
## [38,]     0     2     0     0     0     0     0     1     1     1     1
## [39,]     0     2     0     0     0     0     0     1     1     1     1
## [40,]     0     2     0     0     0     0     0     1     1     1     1
## [41,]     1     0     0     2     0     2     0     0     0     0     0
## [42,]     1     0     0     2     0     2     0     0     0     0     0
## [43,]     1     0     0     2     0     2     0     0     0     0     0
## [44,]     1     1     2     2     0     0     0     0     0     1     1
## [45,]     0     0     2     0     0     0     1     0     0     2     1
## [46,]     0     0     2     0     0     0     1     0     0     2     1
## [47,]     0     2     0     0     0     1     0     0     0     0     0
## [48,]     0     2     0     0     0     1     0     0     0     0     0
## [49,]     0     0     0     1     2     0     0     0     0     0     0
## [50,]     0     0     0     1     0     0     2     0     0     2     1
##       [,47] [,48] [,49] [,50]
##  [1,]     0     1     0     0
```

```
##  [2,]    2    1    2    2
##  [3,]    2    0    1    1
##  [4,]    1    2    1    2
##  [5,]    1    2    1    2
##  [6,]    0    0    0    1
##  [7,]    0    0    0    1
##  [8,]    0    0    0    1
##  [9,]    0    0    0    1
## [10,]    0    0    0    1
## [11,]    0    0    0    1
## [12,]    0    0    0    1
## [13,]    0    2    1    0
## [14,]    0    2    1    0
## [15,]    2    1    0    0
## [16,]    2    1    0    0
## [17,]    2    1    0    0
## [18,]    2    1    0    0
## [19,]    1    0    2    2
## [20,]    1    0    2    2
## [21,]    1    0    2    2
## [22,]    0    0    1    0
## [23,]    0    0    1    0
## [24,]    0    0    1    0
## [25,]    2    0    0    0
## [26,]    0    0    0    2
## [27,]    0    0    0    2
## [28,]    1    1    2    1
## [29,]    2    0    2    0
## [30,]    2    0    2    0
## [31,]    2    0    2    0
## [32,]    2    0    1    2
## [33,]    2    2    0    1
## [34,]    0    1    0    1
## [35,]    0    1    0    1
## [36,]    0    1    0    1
## [37,]    0    1    0    1
## [38,]    1    1    0    2
## [39,]    1    1    0    2
## [40,]    1    1    0    2
## [41,]    2    0    1    1
## [42,]    2    0    1    1
## [43,]    2    0    1    1
## [44,]    1    2    0    0
## [45,]    0    0    0    0
## [46,]    0    0    0    0
## [47,]    1    2    0    0
## [48,]    1    2    0    0
## [49,]    0    0    0    2
## [50,]    0    0    0    0
```

Test using the following code.

```
for (i in 2 : n){
  expect_gte(sum(R[i - 1, ], na.rm = TRUE), sum(R[i, ], na.rm = TRUE))
}
```

6. Create a vector v consisting of a sample of 1,000 iid normal realizations with mean -10 and variance 10.

```
num_rvs=1000;
v=rnorm(num_rvs, mean=-10, sd = 10)
v
```

```
##    [1]  -8.102021196 -18.198882846   8.767064745 -13.457299219
##    [5] -13.596093511  -2.336074495  -3.773000700 -14.220364986
##    [9] -12.393346703 -12.529706306  -9.133868802 -26.069505320
##   [13] -10.608973937 -11.569216857 -11.019707841 -11.676312089
##   [17]   8.984392450  -6.509576501  -7.927973234  -0.119495071
##   [21]  -2.064585937  -4.856546892  -1.984721813  -8.509625328
##   [25]   6.093479662 -18.058189373  -9.303840300   0.498195277
##   [29]   2.289862212   3.102618189 -10.246936643 -14.279270641
##   [33] -14.636788867  -2.844636083  -9.105006105  -7.878667516
##   [37] -27.877816495  -0.232344015 -20.557639740  -8.953453896
##   [41]  -3.264578102  -9.714056773  -7.822042405  -7.172486863
##   [45]  -5.779389048  -5.590272245   6.264861758  -8.576696056
##   [49] -23.773521015 -20.325154544  -1.506429555 -19.790573872
##   [53]   0.240386523 -12.601648048 -15.421678626 -17.448866104
##   [57] -14.070784908  -8.929949475  -5.008685702  11.180070068
##   [61] -16.638237987  14.094808611 -10.801120871 -12.488485864
##   [65] -18.785808579  -8.768492197 -16.384959808   1.932614976
##   [69] -12.393577575 -25.114111248 -22.419456973 -26.244113253
##   [73]   3.499871702  -2.160181871 -14.908204496 -16.248914752
##   [77] -15.280493242 -13.599711637   7.842518860 -14.896345921
##   [81]  -0.592101776 -18.006634346  -0.633672001  -8.645190884
##   [85] -27.294594482  -7.097753324 -21.563459918 -18.337872902
##   [89] -11.407978054 -15.505710306 -35.479828360 -20.682978312
##   [93]  -5.331651280  -9.672345289  -9.987939553 -20.663106748
##   [97] -19.909111812 -14.179978236  -5.897019461 -17.310670484
##  [101] -19.253967451 -14.486775234 -19.886561573 -10.089426629
##  [105] -17.059311576  -9.265174129 -17.966955911   3.842403707
##  [109] -25.041202059   7.647819833 -15.802078452 -13.298371300
##  [113] -22.999855778 -11.224833959  -9.445617512  -5.588122719
##  [117] -29.364729026  -6.487970290  -6.653197809  -9.169555605
##  [121] -22.364238724  -3.242157718 -17.277172229 -17.725247465
##  [125]   3.888870654  -1.294986120 -19.085952005  -2.924119730
##  [129]  -8.340581847 -23.913586940  -5.573804755  -0.512274512
##  [133]  -9.638508486 -19.225431691  -7.967250067  -4.871016465
##  [137]  -3.365290635 -24.110280849   3.171038600 -17.126806575
##  [141] -14.913534458 -16.713039205 -20.523005646 -10.697168783
##  [145] -10.183720259  -9.682509870 -19.239216177  -3.122178605
##  [149] -12.380962563  -0.175314116 -17.603251917 -22.885346695
##  [153]  11.174968622 -16.149087155   0.064576470  -9.735135819
##  [157]   8.869439592  -1.675508039  -2.720568896  -8.190698426
##  [161] -23.283527370  11.972973428 -14.712413290 -23.112130170
##  [165]  -6.657350057 -17.542237468 -12.217238416 -19.578075945
##  [169] -17.696802195  -4.003116090 -20.502859356  -9.297193253
##  [173]  -8.632991449  -3.830869185  -7.213351390 -19.061523873
##  [177]  -8.699030307 -10.276079394 -29.069287258  -4.433084038
##  [181]  14.832844111 -12.563197746 -11.182467720  -3.147524841
##  [185] -13.087394265  17.208462427 -25.255931792 -21.030423217
##  [189] -11.794531038 -33.790540917  -5.353643501 -43.776575978
##  [193] -22.158903328 -16.042632910 -12.908633089 -17.406579306
```

```
##  [197]   2.071309098 -24.822211624  -9.605933077   6.209023544
##  [201] -11.246911096   0.662140722  -5.659721888 -14.017160631
##  [205] -14.069457044  -9.401587140 -11.662560286 -12.641840153
##  [209]   0.512570496 -24.680458443 -37.865938244  -1.166031506
##  [213]  -5.984340887  11.551407453   4.452728995   3.645199105
##  [217] -10.346230334 -18.103503727 -14.683723900 -11.829107872
##  [221]   3.253520389 -15.060815475  -9.168735013 -16.688411495
##  [225]  -6.594990327 -13.265963590  17.613870417   0.460303859
##  [229]  10.056357311  -4.141256324  -6.608396426 -14.118233507
##  [233]  -8.682786789 -10.354819365 -10.040895427 -24.169710355
##  [237] -11.790119600   9.862507837 -12.017179318  -3.015030267
##  [241] -31.527700460 -29.283958394  -4.540680326 -29.496247134
##  [245]  -5.415248026 -19.478976599  -3.782905152  12.401743502
##  [249]  -1.556997052 -14.230151143  -2.272379699 -32.231312949
##  [253]  -6.326269260  -3.142802726  -7.148826978 -20.236343869
##  [257] -12.637268369  -7.404001705 -14.727211360  -7.074912570
##  [261] -13.069662699 -24.212828207  -9.614933967 -28.977787236
##  [265] -20.152946137 -10.589593407 -12.309213242 -10.670832731
##  [269]   5.165983626  -0.590564687  -3.668314117   1.691684276
##  [273]  -3.762825660  -5.174557648 -10.961710669  -0.149879449
##  [277]   0.931957982  -5.918439995   0.452814750  -6.917606919
##  [281]  -2.777036772   2.898592986  -2.520931421   7.192571495
##  [285]  -6.842042945  -4.316419897 -12.594846805 -29.576963812
##  [289] -17.011123944  -5.378989449 -13.055299854  -4.673985519
##  [293]  -5.333410203  -1.113099924 -11.651245359 -11.858791187
##  [297] -13.404765427 -26.925056260 -25.555223302 -23.749542030
##  [301] -11.266896219  -7.050664783  -4.590748204 -16.864324837
##  [305]   6.115171702  -0.437482002   4.333395303 -14.223887865
##  [309] -12.038022937  -6.537118638 -16.267117120 -11.679711031
##  [313]   3.340757267   1.692330488 -31.465512888 -22.423151052
##  [317]  -4.130524952 -17.022366769  -5.299949510 -23.291033605
##  [321]  -6.722068085 -24.694974788 -17.395029071 -17.697626541
##  [325] -10.043357583  -0.789185087 -11.830945012 -14.139542808
##  [329]   1.296679505 -14.380009363 -15.976494819 -15.767069202
##  [333] -15.164962943  -4.572074169   1.984331770 -13.810494612
##  [337]  -7.239229527   5.921574387 -10.948183655 -13.153188207
##  [341]  -9.106612536  -7.166688948   6.274602768  14.454898160
##  [345]  -5.014504566  -7.307521924  -8.005727017 -12.810852540
##  [349]  -8.549591082 -29.848661095  -6.346617661 -15.455624495
##  [353] -10.878092141  -5.805997552 -12.928850798  -3.107863724
##  [357]  -5.253260195   4.292490934 -14.524809489  -0.126981749
##  [361] -12.359865606   6.519682154  -7.175465016   8.265865182
##  [365]   2.377137565  -8.997906198  -4.024681636  -5.873140642
##  [369] -10.620489918 -23.512415650  14.407727571 -17.798549818
##  [373]  -6.774887284 -18.869367508  -5.725495833  12.127169969
##  [377] -27.062895108 -18.469545036  -8.465528343 -21.747896510
##  [381] -10.894109214  -1.819890045  -6.518990053   3.527546422
##  [385]   9.253468233 -26.694997997 -15.392995393   7.796638867
##  [389]  -3.618167576  -9.017040770  -2.411242598  -5.087214896
##  [393]  -5.668361187 -22.124204507 -11.048804312  -8.313646620
##  [397]  -3.069831511 -25.472174849 -11.432287507 -22.403062834
##  [401] -30.905309261   4.789509665  -4.460119824 -21.830760300
##  [405] -12.079235071 -18.897986260   1.928290041   4.054051522
##  [409]  -2.132372237   8.297187280 -22.011676408  -0.569008999
```

```
##  [413] -12.070676046  -7.601177307 -17.863506351  -3.184859767
##  [417] -20.672054459   0.759603635 -17.801632886 -12.367268679
##  [421] -25.205639883  -3.465612546  -8.460834224  -6.586827774
##  [425]  -8.690337377   1.308099318 -17.330968247  -6.004756914
##  [429]   0.705972910 -18.065008569  -7.785976630   6.235672315
##  [433]  -2.229297831  -0.085615304 -10.943184952  -3.830112439
##  [437]  -7.886161845   0.010943150  -5.435614153 -21.295498860
##  [441]  -3.463050258  -8.615298296 -10.298811204 -17.042001014
##  [445]  -6.288834991   3.618973840 -15.784377441 -15.902971567
##  [449] -20.563328405  -3.221192954  -7.413503855 -29.128662819
##  [453] -23.752136324 -14.088348658  -9.585021060  -4.296641871
##  [457] -18.672108375  -2.109186349 -16.110291166  -0.583970372
##  [461]   0.168475563 -27.006765125 -26.269496247 -25.394128628
##  [465]   2.355771059   7.451264103 -15.248904963 -16.500108125
##  [469] -12.821567943  -8.683852264 -16.981216641   2.429844210
##  [473]  -7.755791270   5.411537882 -17.577247663  10.169056700
##  [477]  -8.624263203   1.249420800 -11.485360310  -9.011655457
##  [481] -28.311559530 -12.744220159 -11.153841352 -14.323156219
##  [485] -14.484828101  -2.905187522 -16.929508154  -3.842241189
##  [489]  -1.049251737 -11.704926180 -10.049407689  -1.686677921
##  [493]  10.070509683  -9.228050882 -16.045364269 -10.078173013
##  [497] -18.233903385 -14.031195328  -7.078349800  -3.625902154
##  [501]  -2.709357024 -14.588941207  -9.339214019  -9.943774570
##  [505]  -3.868506060  -9.609832944  -5.325208929 -11.138277762
##  [509] -32.382692843 -20.572204229   3.637567098 -19.896873413
##  [513]  -0.301648366 -12.414096861   5.348208097  -9.610337466
##  [517]  -6.089053283 -13.500000983  -8.811121173  -9.225519210
##  [521]  -0.363152029 -15.340731223   6.639542737  -1.260427438
##  [525]   1.952511611  -9.588445206  -7.926974126  -1.944147233
##  [529] -14.035845851   5.086525258   0.377890976  -2.003694550
##  [533]  -3.275179495  15.015546525   1.405172069 -12.297238053
##  [537]  -6.344713271 -22.098251872   6.103251904   7.168934546
##  [541] -11.438113283 -15.846113315 -10.723998442  -0.977868578
##  [545] -10.721498558   2.890832062 -10.869262497 -16.123256995
##  [549]  -3.189071418 -17.006870283 -17.755909911 -17.453449275
##  [553] -19.004515805  -8.549597306 -12.280216437 -10.103473725
##  [557] -15.427362969  -5.369961599  -5.873264197   4.684953633
##  [561]  -1.440731178 -20.462760741  -5.334031885  -5.213537244
##  [565]   3.289662061  12.490983616 -20.930962205  -7.928065186
##  [569] -17.441873559 -21.741173319   0.742950722   8.027175687
##  [573] -13.501414343  -8.204032678  -2.719072975  -1.269072983
##  [577]  14.062792712 -16.893726474  -7.313558170 -23.104496200
##  [581] -11.335359615  -8.616960139 -10.020535807  -8.046520805
##  [585]  -0.619371009 -14.074934156 -14.054958312  -0.322245379
##  [589]  -6.803561345 -14.332757704 -13.446795624 -16.895171518
##  [593] -15.958864670   1.838926820  -6.412317762 -19.564419553
##  [597] -12.985290891 -19.199914351  -5.284902115 -26.588835598
##  [601]  -7.290282016 -17.213522781 -18.150320082 -17.990699625
##  [605]  -6.082679113 -14.015371826 -16.488646973  -0.235232556
##  [609]   8.757569625 -33.312847176   6.647858901 -16.996259759
##  [613]  -3.738400368 -17.281800343  -6.448564457 -12.471619343
##  [617] -16.621729195 -10.823518628  -3.451096614 -13.476596939
##  [621] -23.914112794 -13.742847314  -2.927393557 -10.860912472
##  [625]  -7.611969082 -22.475539859  -0.609098336  -1.781765347
```

```
##  [629] -16.761299569   0.276750374 -10.154549105  -6.999390332
##  [633]  -3.807601541 -19.225759181  -4.936914901 -10.818498976
##  [637] -20.074127685 -15.325443600  -5.561257253 -24.661184926
##  [641]  -2.174908622 -16.727619790 -17.431826898  -9.882229601
##  [645] -18.476378341 -13.857458228  -0.024803243 -25.035787517
##  [649] -15.434637593  -8.535279073 -32.684864936 -11.967592928
##  [653] -10.947401288 -19.253987975  -8.094914200 -13.776970311
##  [657] -17.137181476 -23.810986472  -4.110972673  -4.312877559
##  [661] -23.478278256  -7.399486117 -27.562189530  -8.656893987
##  [665] -16.162612574   3.793389485 -18.605614727  -5.112079598
##  [669] -11.646862275 -13.237828932 -12.012638879 -21.384883824
##  [673]  -5.075671415 -25.233750900 -26.944941209  -0.522311710
##  [677]  -1.727688524   1.033993335  -0.982488746  -0.720175567
##  [681]  -1.068101177  -4.126351777   7.745564184 -14.694949561
##  [685] -26.363203865   8.205145725  -7.583959930 -10.221369142
##  [689]   5.740379644 -17.691195415 -17.004232273   0.612041826
##  [693]  -1.167945628 -11.829685951 -13.866855573 -14.372598808
##  [697]  -5.274913928  -1.317243522  -8.816814590   0.397320503
##  [701] -21.695061148 -15.889812593 -14.416696387 -16.598513215
##  [705]  -6.341708990  -0.118908763  11.375734516  19.607842427
##  [709] -10.327020868 -11.565401841   4.393454953   2.185250564
##  [713] -16.233975448   0.560680801 -15.082587957 -15.483528711
##  [717] -14.834464523  -9.205122207 -26.285852320 -23.680376121
##  [721]  -4.294363842  -1.557990759 -18.627650069  -3.913115153
##  [725]  -3.730659068 -15.222880116  -3.618294282  -2.753822317
##  [729] -11.440772628 -13.397235424  -4.523003703  -7.549813369
##  [733] -14.527880417   3.031223433  -9.246626192 -20.780031312
##  [737] -10.758296892 -13.896950026  -5.431463022   0.975748460
##  [741]  -3.714777981   7.594321844   0.844769252  -6.803442135
##  [745] -16.738261211 -16.249890674  -8.634574262 -12.784514936
##  [749] -15.577234059 -22.250270323 -14.831710944  -8.507249227
##  [753]  -8.141959054   5.371439624  10.080839453   3.858973039
##  [757] -23.018982004 -13.634286169   4.571586092  -8.107590427
##  [761]   5.441678616 -14.249376991 -15.021947044 -20.991364440
##  [765] -14.998673514 -14.742790666 -11.100267065 -17.476957704
##  [769]  -5.027965416  -3.310679898  -0.160480268   2.766751337
##  [773]  -5.884048424  -0.111518125 -15.661778916   1.598413979
##  [777] -29.631212380 -12.568655502  -4.411388788  -7.273127966
##  [781] -22.346786761  -3.865414557 -18.079201087 -18.090957908
##  [785] -17.652462730  -4.803763015 -16.108901872 -28.875176618
##  [789] -17.015683583 -15.329192663   4.492977869  -0.712394197
##  [793] -12.478187582   2.952659545  -4.335206701  -6.878966692
##  [797]  -2.327295303 -12.622784532 -24.653262613 -23.258389169
##  [801] -13.154918199  -9.484631569 -31.053959493 -10.865173646
##  [805] -13.117147493 -30.380364676 -12.479842791 -11.131472575
##  [809] -30.996596908 -24.461205556 -18.851633699  -8.105340829
##  [813] -28.063955257 -22.283753216   6.797456393 -10.969683135
##  [817]   3.995973688   0.580889135 -15.719270778 -26.079021952
##  [821]  -2.615609114  -1.977039484  -7.551373473 -10.698367831
##  [825] -19.937270146 -26.347671363  -4.015225276 -10.637451890
##  [829]  -7.070265556   6.954171349 -11.329755172 -16.540680072
##  [833]  -8.115372448 -20.782152746 -18.425334144 -13.205894949
##  [837] -14.242409127  -7.966475074 -11.129658135   0.383072209
##  [841]   2.236178469 -13.395152827   4.427206336  -6.779704350
```

```
##  [845] -18.991542210 -15.133230233 -10.927978860  -5.174350733
##  [849] -21.156155196 -26.707866735 -21.904822573   8.522719038
##  [853] -21.311731876  -6.816687467   0.815539105 -17.170413602
##  [857]  -0.853618109 -11.590455417  -0.751732797  -2.619574045
##  [861] -15.789057754 -16.737546131 -21.448359265 -18.855080520
##  [865] -10.405960141  -7.071103888  -7.185894158 -23.663350815
##  [869] -22.107079516 -10.309052517 -17.706166939   2.465582587
##  [873]  -3.617973821 -26.886003855 -16.846796472 -10.180121147
##  [877]  -5.957756963 -10.326966186 -19.880089437  -5.672031045
##  [881]  -9.063680822  -2.419558202   8.185493629  -7.056681046
##  [885] -19.077226225 -13.842553809  -1.620360598   1.897496474
##  [889]   3.909020502  -2.200422544  -7.300317576  -9.969937096
##  [893] -20.588765516 -13.997382909 -12.181981741 -18.791510672
##  [897]   0.135417661 -23.513126606  -7.977578265 -13.232231824
##  [901]   3.231466638 -18.818605739  -1.617864308  -4.339699032
##  [905]   7.600849035 -22.976364032   5.118867519 -18.813275734
##  [909]   2.578249331  -4.214397359 -10.866467668 -14.500489109
##  [913]  -7.418183451 -11.738335090  -7.612786966   0.837304858
##  [917]  -9.440133302 -13.410795360  -1.695069701   9.413527092
##  [921]  -6.948852426  -6.475274587  -3.013691065   1.130107589
##  [925]   0.577800059 -17.567250524 -10.864798393  -3.318035036
##  [929] -28.903068002 -11.110350299   9.559058741  -8.894551349
##  [933] -12.227931074 -15.013983342  -1.307215640  -7.592935810
##  [937] -16.300701911  -9.461978335  -8.866888182 -14.137297210
##  [941] -15.750081181 -21.785463134 -11.430095646 -15.846700159
##  [945] -12.619260718  -0.380431713 -28.644383092 -11.033935688
##  [949]   1.530172238  -4.365313879 -19.344015457  -3.019072566
##  [953] -11.972068707  -7.854741173   1.101245543  -2.698983696
##  [957]   1.201104215   9.551215517   3.736166506 -11.949585502
##  [961] -11.802323448 -24.016935768  -7.591835102 -14.642692192
##  [965]  -3.375688911   0.983435616 -19.060636469  10.063259965
##  [969] -12.756179786 -33.203383063 -19.342638184 -11.199894881
##  [973]  -9.114207065   0.299337316 -10.027588571  -6.949406749
##  [977] -13.691158847 -11.313819521 -20.599191442 -15.607724025
##  [981]  -6.784444714 -18.468190550   2.213437579 -12.743828373
##  [985]  -6.743079849  -3.583680174 -20.901238024  -2.896723000
##  [989] -12.599806546 -10.544521522   6.129900356  -0.006852154
##  [993]   2.750438351 -22.889311677 -15.968120326  18.329031256
##  [997] -10.530622722  -2.485486550   6.640413099  -9.892831148
```

Find the average of `v` and the standard error of `v`.

```
mean(v)
```

```
## [1] -9.536046
```

```
sd(v)
```

```
## [1] 9.636155
```

Find the 5%ile of `v` and use the `qnorm` function as part of a test to ensure it is correct based on probability theory.

```
v_0.05 =quantile(v, probs = 0.05)
test_v =qnorm(0.05, mean= mean(v), sd = sd(v), lower.tail = TRUE, log.p=FALSE)
#expect_equal(v_0.05, test_v )
```

Find the sample quantile corresponding to the value -7000 of `v` and use the `pnorm` function as part of a test to ensure it is correct based on probability theory.

```
inverse_quartile_v=ecdf(v)
inverse_quartile_v(-7000)
```

```
## [1] 0
```

```
test_in_v=pnorm(-7000, mean = -10, sd = 10, lower.tail = TRUE, log.p = FALSE)
expect_equal(inverse_quartile_v(-7000),test_in_v )
```

7. Create a list named `my_list` with keys "A", "B", ... where the entries are arrays of size 1, 2 x 2, 3 x 3 x 3, etc. Fill the array with the numbers 1, 2, 3, etc. Make 8 entries.

```
my_list=list()
for (i in 1:8){
  my_list[[i]]= array(1:i^i, dim = rep(i,i))
}
names(my_list) = c("A","B","C","D","E","F","G","H")
```

Test with the following uncomprehensive tests:

```
expect_equal(my_list$A, array(1,1))
expect_equal(my_list[[2]][, 1], 1 : 2)
expect_equal(dim(my_list[["H"]]), rep(8, 8))
```

Run the following code:

```
lapply(my_list, object.size)
```

```
## $A
## 208 bytes
##
## $B
## 216 bytes
##
## $C
## 336 bytes
##
## $D
## 1232 bytes
##
## $E
## 12728 bytes
##
## $F
## 186848 bytes
##
## $G
## 3294400 bytes
##
## $H
## 67109088 bytes
```

Use `?lapply` and `?object.size` to read about what these functions do. Then explain the output you see above. For the later arrays, does it make sense given the dimensions of the arrays?

Answer here in English.

The lapply(x, FUN) function takes in two inputs, a vector(atomic or list) or an expression object and an arbitary function FUN. The output is a list of the same length of x, with each element of it being the output of applying function FUN to an element of x. In the above code, the function is the object.size function, which provides an estimate of the memory that is being used to store an R object. For the list 'my_list', there is a list of the same length with every component being the estimated storage memory for the corresponding component in my_list. Given the dimensions of the later arrays, it makes sense to have such a memory storage. These latter arrays will contain over thousands of entries after the 4th array.

Now cleanup the namespace by deleting all stored objects and functions:

```
rm(list=ls())
```

Basic Binary Classification Modeling

8. Load the famous `iris` data frame into the namespace. Provide a summary of the columns and write a few descriptive sentences about the distributions using the code below and in English.

```
data(iris)
summary(iris)
```

```
##   Sepal.Length    Sepal.Width     Petal.Length    Petal.Width
##  Min.   :4.300   Min.   :2.000   Min.   :1.000   Min.   :0.100
##  1st Qu.:5.100   1st Qu.:2.800   1st Qu.:1.600   1st Qu.:0.300
##  Median :5.800   Median :3.000   Median :4.350   Median :1.300
##  Mean   :5.843   Mean   :3.057   Mean   :3.758   Mean   :1.199
##  3rd Qu.:6.400   3rd Qu.:3.300   3rd Qu.:5.100   3rd Qu.:1.800
##  Max.   :7.900   Max.   :4.400   Max.   :6.900   Max.   :2.500
##        Species
##  setosa    :50
##  versicolor:50
##  virginica :50
##
##
##
```

The above summary for the characteristics of a flower follow distributions that can vary amongst each other. The 4 leftmost traits are features that are continuous variables, for that reason it is useful to produce the Tukey 5 number summary. The rightmost column is a categorical feature and therefore would not benefit from the Tukey 5 number summary. Instead, it is useful to obtain the amount of flowers that fall into such a category.

The outcome metric is `Species`. This is what we will be trying to predict. However, we have only done binary classification in class (i.e. two classes). Thus the first order of business is to drop one class. Let's drop the level "virginica" from the data frame.

```
iris[,"Species"]=factor(iris[,"Species"],exclude = "virginica", ordered = TRUE)
```

Now create a vector `y` that is length the number of remaining rows in the data frame whose entries are 0 if "setosa" and 1 if "versicolor".

```
p=ifelse(as.factor(iris[,"Species"])=="virginica",NA,iris[,"Species"])
x=0
for(i in 1:length(p)){
  if(!is.na(p[i])){
    x=1+x
  }else{
    break
```

```
  }
}
x
```

```
## [1] 100
```

```
#this will obtain a number for the length of y. Note, I kept the levels ordered in the

#chunk of code above this one so I could immediately break once reaching the first 'NA'.

y_1=iris[1:x,"Species"]
y=ifelse(y_1=="versicolor",1,0)
y
```

```
##   [1] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
##  [36] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [71] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
```

9. Fit a threshold model to `y` using the feature `Sepal.Length`. Try to write your own code to do this. What is the estimated value of the threshold parameter? What is the total number of errors this model makes?

```
n=length(y)
num_errors_by_parameter=matrix(NA, nrow=n, ncol=2)
colnames(num_errors_by_parameter)=c("threshold_param","num_errors")
y_logical=y== 0
b=iris$Sepal.Length[1:n]
for(i in 1:n){
  threshold=iris$Sepal.Length[i]
  num_errors=sum((b<threshold)!=y_logical)
  num_errors_by_parameter[i,]= c(threshold,num_errors)
}
best_row=order(num_errors_by_parameter[,"num_errors"])[1]
num_errors_by_parameter[best_row,"threshold_param"]
```

```
## threshold_param
##             5.5
```

```
num_errors_by_parameter[best_row,"num_errors"]
```

```
## num_errors
##         11
```

Does this make sense given the following summaries:

```
summary(iris[iris$Species == "setosa", "Sepal.Length"])
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
##   4.300   4.800   5.000   5.006   5.200   5.800      50
```

```
summary(iris[iris$Species == "virginica", "Sepal.Length"])
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
##      NA      NA      NA     NaN      NA      NA      50
```

Write your answer here in English.

This makes sense given that the best estimated threshold parameter is bigger than the median for the sepal length of a setosa iris and smaller than the sepal length of the virginica iris. The sepal length of the versicolor iris is expected to be within that interval.

10. Fit a perceptron model explaining y using all four features. Try to write your own code to do this. Provide the estimated parameters (i.e. the five entries of the weight vector)? What is the total number of errors this model makes?

```
n=length(y)
MAX_ITER=1000
w_vec=rep(0,5)
S_l=iris$Sepal.Length[1:n]
S_w=iris$Sepal.Width[1:n]
P_l=iris$Petal.Length[1:n]
P_w=iris$Petal.Width[1:n]

X1 = as.matrix(cbind(1,S_l,S_w,P_l,P_w ))

for( iter in 1:MAX_ITER){
  for(i in 1:n){
    x_i=X1[1,]
    yhat_i=ifelse(sum(x_i*w_vec)>0,1,0)
    y_i=y[i]
    w_vec = w_vec + (y_i - yhat_i) * x_i
  }
}
w_vec
```

```
##     S_l S_w P_l P_w
## 1.0 5.1 3.5 1.4 0.2
```

```
yhat = ifelse(X1 %*% w_vec > 0, 1, 0)
total_errors=sum(y != yhat) / length(y)


total_errors
```

```
## [1] 0.5
```