Business Analytics – Homework 3

Due 9 am, Friday September 20

This assignment consists of 5 exercises the last one **being completely optional**. It has been designed to make sure you dominate coding loops and conditional If statements. As always, you will be asked to write short code snippets or code-chunks in R. You will be graded both on your code, and the written answers you provide. When evaluating the code, the grader will take on the role of a co-worker. Code will evaluated both in terms of how correct and how clear it is. By correctness, I mean that the code fulfills the requirements of the question. By clarity, I mean that the grader should be able to understand what your code does within 30 seconds of reading it. As discussed in class, this is aided by clear comments, good variable names, proper indentation, and short lines.

Problem 1: Code Reading

Explain the difference between the following two functions? Do they serve the same purpose?

```
eq_2nd_degree_V1=function(a,b,c){
  condition_sqr_positive = (b*b)-(4*a*c)>=0
  if(!condition_sqr_positive){ print("Xavi the SQRT of a negative number does not exist")  }
  if(condition_sqr_positive){
    if(a == 0)      print("Xavi you cannot divide by zero")

    if(a!=0){
      sol_1 = (-b - sqrt(b*b-(4*a*c)))/(2*a)
      sol_2 = (-b + sqrt(b*b-(4*a*c)))/(2*a)
    }
  }
  print(paste('solution one is equal to ',sol_1))
  print(paste('solution two is equal to ',sol_2))
}

eq_2nd_degree_V2=function(a,b,c){
  condition_sqr_positive = (b*b)-(4*a*c)>=0
  if(!condition_sqr_positive){    print("Xavi the SQRT of a negative number doesnot exist")}
  if(condition_sqr_positive){
    if(a == 0)      print("Xavi you cannot divide by zero")

    if(a!=0){
      sol_1 = (-b - sqrt(b*b-(4*a*c)))/(2*a)
      sol_2 = (-b + sqrt(b*b-(4*a*c)))/(2*a)
    }
  }
  solution = c(sol_1, sol_2)
  return(solution)
}
```

Problem 2: Fibonacci sequence

Write the function Fibonacci sequence that takes one integer and returns its Fibonacci counterpart.

```
fibonacci=function(N){
        # code here
        # code here
        # code here

}
```

- The function must only work when N is a positive integer.
- For more information about what a Fibonacci sequence please click here

Problem 3: Matrix Multiplication

Using only for-loops, write a function that **returns** the product of two matrices. A very nice tutorial of what matrix multiplication means can be found here and here. Make sure that the function only multiplies matrices that have compatible dimensions, that is, if the user inputs two matrices A and B with number of columns from A not matching number of rows of B, it should say something like: Wrong dimensions!!

```
matMult = function(A,B){
        # code here
        # code here

}
A = matrix(c(0,5,3,5,5,2),nrow=2,ncol=3)
B = matrix(c(3,3,4,4,-2,-2),nrow=3,ncol=2)
C = matMult(A,B)
print(C)
```

Problem 4: Poker Face

We will use the simulation methods seen in class to estimate the probability of getting a pair of cards with the same number when a 5 card poker hand is pulled out from a deck. The goal, however, is to make sure you are proficient combining lists and matrices with loops and functions. We go step by step.

1. Initializing the list `cards_Received`: Create the list `cards_Received` with length N=100000

2. Poker hand simulation:
    a. You can simulate a poker hand as a matrix that has only zeros or ones.
    b. I attach two examples for illustration. In `cardsReceived1`, below represents that I have been give the Ace of Spades, the 8 and the J of Clubs, the 8 of Hearts and 7 of diamonds. In `cardsReceived2` I have 3 figures of Clubs and one 8 and a 7 of Spades and of Diamonds respectively

```
cardsReceived1
#        Spades Clubs Hearts Diamonds
#  [1,]      1     0      0        0
#  [2,]      0     0      0        0
#  [3,]      0     0      0        0
#  [4,]      0     0      0        0
#  [5,]      0     0      0        0
#  [6,]      0     0      0        0
#  [7,]      0     0      0        1
#  [8,]      0     1      1        0
#  [9,]      0     0      0        0
# [10,]      0     0      0        0
# [11,]      0     1      0        0
# [12,]      0     0      0        0
# [13,]      0     0      0        0
```

```
cardsReceived2
#        Spades Clubs Hearts Diamonds
#  [1,]      0     0      0        0
#  [2,]      0     0      0        0
#  [3,]      0     0      0        0
#  [4,]      0     0      0        0
#  [5,]      0     0      0        0
#  [6,]      0     0      0        0
#  [7,]      0     0      0        1
#  [8,]      1     0      0        0
#  [9,]      0     0      0        0
# [10,]      0     0      0        0
# [11,]      0     1      0        0
# [12,]      0     1      0        0
# [13,]      0     1      0        0
```

3. Store in the list `cards_Received` N simulated poker hands

4. Count the number of times you get 2 cards of the same number and the divide it by the total number of card hands received (namely N)

Problem 5: Moving around

Consider the following problem I am currently facing. I have to visit cities A, B, C, D, E. My friend can drop me in **any** of them from for free but, from there on, I will have to walk. One option would be asking her to drop me in city A and then walk from A to B from B to C, from C to D and from D to E. If were to do that, I would have to walk 45 + 70 + 104 + 32 minutes.

- If instead of A,B,C,D,E I follow A,B,D,E,C how long should it take me?
- Can you tell me the shortest path that takes me to all the cities?

```
#         City A city B city C city D city E
# city A       0      45      89      78      24
# city B      45       0      70      51      18
# city C      89      70       0     104      44
# city D      78      51     104       0      32
# city E      24      18      44      32       0
```