Mid-term_1.R

pradyuth

Thu Mar 15 12:50:21 2018

```r
#Name: Pradyuth V
#Mid term exam 1

Mid_term <- function(){

  #Question 1.a
  #Total number of universities
  total_num_univ <- nrow(univ_ranking)

  #number of universities equal to public
  public_univ <- sum(univ_ranking$type == "public")

  #number of universities equal to private
  private_univ <- sum(univ_ranking$type == "private")

  #Question 1.b
  #Average endowment of schools with score greater than 5
  avg_endow_greater_than_5 <- mean(univ_ranking$score[which(univ_ranking$score > 5)])

  #Question 1.c
  #Name of the school with the highest tuition
  highest_tuition_school <- univ_ranking$name[which(max(univ_ranking$tuition))]

  #Name of the public school with highest tuition
  highest_pub_tuition_school <- univ_ranking$name[which(max(univ_ranking$tuition[univ_ranking$type == "

  #Question 1.d
  school_accept_rate_10_to_20 <- sum(univ_ranking$accept_rate >= 10 & univ_ranking$accept_rate <= 20)

  #Question 1.e
  #table(univ_ranking$type) gives a count of how many schools are public, and private


  #Question 2.a
  #1 is a numeric and it can be coerced into logical by as.logical function
  #T/F is a logical and it can be coerced into numeric by as.numeric function

  #Question 2.b
  #'&&' does operation for the first element of various vectors
  #'&' does operation for all the elements by considering one at a time
  #of each vector

  #c(0,1,5,0,1) & c(0,0,0,NA,NA) gives an output of F,F,F,F,NA

  #Question 2.c
  #When we create double variables, there is an additional small number added
  #to the double value. So comparing double values using == doesn't return True
```

```r
#value.
#At times, the shorter number recycles in order to match the length of larger
#number
#To fix this, we add a tolerance to the double value.
#a <- 0.03
#b <- 0.02
#tol <- 1e-5
#a - b == (0.01)

#We could use the function all.equal((a - b), 0.01) to compare both numbers

#Question 3.a
my_mat[seq(1, nrow(my_mat), by = 2), ]

#Question 3.b
matr <- function(matrix){
  for(i in seq(1, nrow(matrix))){
    for(j in seq(1, ncol(matrix))){
      if(j > i){
        matrix[i, j] <- 0
      }
    }
  }
}

#Question 3.c
matrix(c(1,2,3) > c(2,3), nrow = 2, ncol = 2)
#This will lead to an output of
# F T
# F F
#Warnings/ errors:
#Longer object is not a multiple of shorter vector for c(1,2,3) > c(2,3)
#Data given to the matrix is not sufficient to fill the matrix

#Question 4.a
my_vec_long <- function(my_vec_short){
  #Considering the starting points for the sequence
  start <- seq(1, length(my_vec_short) - 1, by = 2)

  #Considering the position points for the sequence
  position <- seq(2, length(my_vec_short), by = 2)
  j <- 2

  #Creating a new vector of size expected by the given input
  new_vec <- c(rep(0, as.integer(sum(my_vec_short[position]))))
  for(i in start){
    #Generating numbers as represented by start and position vector
    #Concatenating numbers with the new_vec
    new_vec <- c(new_vec, seq(my_vec_short[i], my_vec_short[i] + my_vec_short[j] - 1))
    j <- j + 2
  }
  return(new_vec[(sum(my_vec_short[position]) + 1) : (2 * sum(my_vec_short[position]))])
}
```

```r
#Question 4.b
original_form <- function(my_vec_longg){
  count <- 1
  j <- 2
  short_form <- rep(0, length(my_vec_longg))
  count_score <- rep(0, length(my_vec_longg))
  for(i in 1:length(my_vec_long)){
    if((my_vec_longg[i + 1] == my_vec_longg[i] + 1) & (i < 7)){
      count = count + 1
      count_score[j] <- count
      j = j + 1
    } else {
      count = 1
      j = j + 1
      count_score[j] <- count
    }
  }
  return(count_score)
}

#Question 4.c
#A vector can hold values of one particular type and working with them
#is easy. Lists hold various numbers of different types. Accessing elements, in
#vector is easier than a list.


#Question 5.a
ggplot(data = univ_ranking, aes(x = tuition, col = type)) +
  geom_hist()


#Question 5.b
ggplot(data = univ_ranking, aes(x = endowment, y = start_salary, col = ifelse(accept_rate > 50, 'red'
  geom_point()

#Question 5.c
#color = 'blue' within aes means that blue column would be chosen from the dataframe as the color.
#color = 'blue' outside aes means that color would be set to blue


#Question 6.a
lm(data = univ_ranking, score ~ . - name)

#Question 6.b
lm(data = univ_ranking, score ~ start_salary + start_salary^2 - 1)

#Question 6.c
#Underfitting is a phenomenon which speaks about a model not fitting the data correctly
#which results in incorrect prediction of new data

#Overfitting refers to a model which is trained really well. It takes into account of the
#noise and other trivial factors associated with the training data.
```

```
  #Cross Validation involves separating the data into test and training. Depending on the
  #value of K, the error rate is observed and the one with the smallest error rate is considered.
  #We select the value of K accordingly.

  #Knn method starts to underfit as we increase the value of K. Higher value of K represents the
  #larger number of neighbours which are surrounded. Larger neighbour values in training set,
  #correspond to a single value in the test data.

  #Question 7.a
  #rnorm(5,,10) generates 5 random variables with a mean of zero and a standard deviation of 10

  #Question 7.b
  #Lexical scoping refers to the variable initialisation in the function as well as globally.
  #If a variable if present both, inside the function, and globally, the function will take the
  #value of variable present globally.

  #Question 7.c
  #It generates a gaussian variable and compares it with zero. If it's less than zero, then it
  #equates to zero.
  #Alternate code:
  # x <- rnorm(1)
  # if(x < 0){x <- 0}

  #Question 7.d
  #length(my_df) = n
  #length(my_df[1]) = 1
  #length(my_df[[1]]) = m

}
```