**R interview questions**

一般R在面试中不会单独考，不会像engineer那样让你写一段code，实现什么算法，而是结合一个统计的问题或case study。

一般都先要**给定一个场景**，给定一个问题，要**先回答一个解决的思路**，到**具体某一步**的时候，可能会给你一个dataset或者很具体的task让你在R里实现一下。不会有特别复杂的R的syntax的interview，不用背语法，记住特别fancy的写法。大概**知道R中的功能**即可，考的更多的是**逻辑和基本的syntax**。

也有时候给你一个**统计的方法/算法**，比如k-means，算法是比较简单的，让你**解释一下这个算法并在R中写一遍**。算法本身要记得，知道这个算法怎么实现。写起来反而比较简单。

例题1：

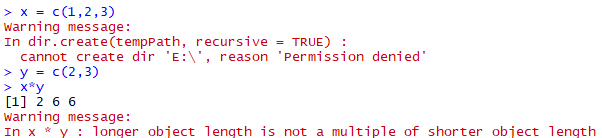
X <- c(1,2,3)

Y <- c(2,3)

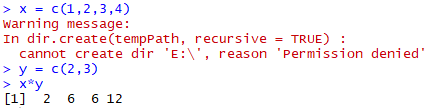
X\*Y? 2 6 6

在R语言中，不同长度的向量也是可以相加和相乘的，乘法的规则和加法类似

这一类的规则就是 x[1]+y[1]，x[2]+y[2]，x[3]+y[3]，**x[4]+y[1]**(因为y[3]就结束了，**进入了又一次循环**)，所得的向量长度为最长的那个向量的长度



**x\*y是point wise**的点乘，即一个元素和一个元素对应相乘，如果一个vector长度不够，回头再次使用该vector的第一个元素，依次后推（**自动循环较短的vector**）。



X = c(1,2,3,4,5)

Y = c(2,3)

X\*Y = 2,6,6,12,10

Matrix 相乘就不是point wise的，对应的乘号为 X**%\*%**Y

例题2：

How missing values and impossible values are represented in R language?

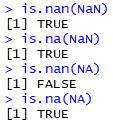
Misssing values-**NA**

Impossible values- **NaN** 注意大小写

还可能会问，用什么function来检查它是不是NA？

is.na() ~~and is.nan()~~

R is case sensitive, “error in is.NA(): could not find function ‘is.NA()’”



可能不会直接问你is.na()的function，会问：一个dataset中有missing value怎么办？Missing value能不能直接去掉？

When we meet missing values, we can not remove them directly, because some missing values also give us some insights and has great impact on our analysis results. We can use 0 or means to replace them.——my answer

When we meet missing values in our data set, first we need to find out **why we have missing values** here? Are they random? **If they are not random missing values**, systematically appears, then the **data quality may be bad**. We need to fix some problems /improve in data mining and data capturing process. **If we remove the missing values directly, this will lead to biased results**.

If the missing values looks random, if one feature has too many missing values, then this feature may be useless to our model so we can remove this column. If there is not too many missing values in one column, say just fewer than 30%, may be we can do some imputations,for example, using the avarage of the other values (in certain group) to replace the missing values (in this group). we can also use model based imputation- using other features without missing values to build a model to predict the values in this feature. If this is a categorical variable, then we can use a new level to label the missing values, may be we can call them “missing”.

例题3：python中的function在《python note》p17

Write a function in R language to replace the missing value in a numeric vector with the mean of that numeric vector.

impute <- function(vector) {

vector[is.na(vector)] <- mean(vector, na.rm = TRUE);

return(vector)

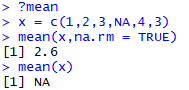
}

Mean中的一个option：

na.rm(**rm**-**r**emo**v**e)

a logical value indicating **whether NA values should be stripped before the computation proceeds**

这里设置为T就是说要先去掉所有的missing value来计算mean（不然直接mean求出的均值会把NA算进去，结果就会为NA），这里应该用这个不带missing value的mean（先把NA去掉后用其他值计算mean）来代替missing values。



函数写为：

imputation<-function(x){

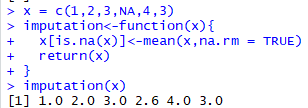
x[is.na(x)]<-mean(x,na.rm = TRUE)

return(x)

}

这里is.na(x) is a logical vector, we can not assign a value directly to a logical vector, but we can **use the logical vector to extract some values in x**(NA values here), and then **assign the mean to the values we extracted from x**.

\*\*\*用is.na创造的逻辑变量提取向量中的元素。\*\*\*——取出满足logical vector这个条件的x的值。[ ] 表示取值，自动取出logical为TRUE的那些值。——不需要做循环，遍历x中的NA来赋值



除了把NA替换成了2.6，还把其他整数变成了小数，why？——因为我们assign来的这个均值—2.6是一个小数，而x是个vector，**vector中所有元素的datatype都是一样的**，所以当有2.6进来时，**其他的integer也自动转换类型为**当前最高等级（**double**）。

Another vector example：

> y = c(1,NA,NA,NA,1,1,1,1)

> imputation(y)

1. 1 1 1 1 1 1 1 1

注意R中和Python中写函数的格式不同：

Python：

def function\_name\_1(x, y):

z = do\_something(x, y)

return(z)

Or

def function\_name\_2(x, y):

do\_something(x, y)

print()

plot()

R：

Your\_function\_name <- function(parameter1, parameter2..){

do something

return(your\_output)

}

例题4：

b <- 4

f <- function (a)

{

b <- 3 ——local variable，overwrites the original variable b = 4 **in f, 仅限于在f函数中overwrite**

b^3 + g (a) ——b^3=3^3 using local variable b, but since there is no definition for variable b **in g**, **it takes global variable b=4**

}

g <- function (a)

{

a\*b

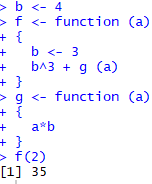
}

g(2)?

f(2)?

My ans: g(2) = 2 \* 4 = 8

f(2) = 3^3 + g(2) = 3^3 + ~~2\*3~~ = 27+~~6~~ = ~~33~~



3^3 + 2\*4 = 35

Local variable vs. Global variabl

Local variable——在一个function内部定义的variables

Global variabl——在functions外定义的variables

**在f中再次定义了b所以在f中用到b时，b=3；在g中没有定义过b，只能去调用global variable b，所以在g中用到b时b=4. 虽然g套在f中，但g的运算是在外面的，单独的，不能使用f函数的local variable。**

例题5：

What is the difference between data frame and a matrix in R?

<https://www.dezyre.com/article/100-data-science-in-r-interview-questions-and-answers-for-2017/187>

常见的data science interview questions in R

在R语言中，一个二维数据既可以用matrix来表示，也可以用data.frame来表示，那么它们的区别在哪里呢。通过多次实践发现，matrix是一个独立的数据类型，而data.frame是多个一维的数据，这就意味着，**data.frame里面允许不同类型的数据存在**（可以第一列是numeric，第二列是character，第三列是logic，etc.），可以使用str()函数查看具体信息，而**matrix里的所有数据都是同一种类型**。（要么整个都是numeric，要么整个都是character）

使用cbind()，rbind()，as.matirx()，matrix()生成的是matrix类型，而read.table()和data.frame()生成的是data.frame类型。

值得注意的是，**虽然可以给matrix 的不同列命名，但是$在matrix是不能使用的**。

例题6：

What functions will you use to see data summary in R?

summary( )

str( )

作业题2：

data(mtcars)#load R 自带的data set—mtcars

让你**创造一个新的列叫brand，把每个车的品牌放在这一列**。

我们看到每个车的**品牌加类型**是在第一列，作为indices（row names）

> data(mtcars)

> head(mtcars)

mpg cyl disp hp drat wt qsec vs am gear carb

Mazda RX4 21.0 6 160 110 3.90 2.620 16.46 0 1 4 4

Mazda RX4 Wag 21.0 6 160 110 3.90 2.875 17.02 0 1 4 4

Datsun 710 22.8 4 108 93 3.85 2.320 18.61 1 1 4 1

Hornet 4 Drive 21.4 6 258 110 3.08 3.215 19.44 1 0 3 1

Hornet Sportabout 18.7 8 360 175 3.15 3.440 17.02 0 0 3 2

Valiant 18.1 6 225 105 2.76 3.460 20.22 1 0 3 1

所以这里要用到 string split,需要取出第一列中的第一个单词，即brand。

相关函数：strsplit( ) 不用特别去记，具体要用时可以google“R split string by space”因为这里的string是用空格分隔的。

> rownames(mtcars)

[1] "Mazda RX4" "Mazda RX4 Wag" "Datsun 710"

[4] "Hornet 4 Drive" "Hornet Sportabout" "Valiant"

[7] "Duster 360" "Merc 240D" "Merc 230"

[10] "Merc 280" "Merc 280C" "Merc 450SE"

[13] "Merc 450SL" "Merc 450SLC" "Cadillac Fleetwood"

[16] "Lincoln Continental" "Chrysler Imperial" "Fiat 128"

[19] "Honda Civic" "Toyota Corolla" "Toyota Corona"

[22] "Dodge Challenger" "AMC Javelin" "Camaro Z28"

[25] "Pontiac Firebird" "Fiat X1-9" "Porsche 914-2"

[28] "Lotus Europa" "Ford Pantera L" "Ferrari Dino"

[31] "Maserati Bora" "Volvo 142E"

> x = rownames(mtcars)[1]

> x

1. "Mazda RX4"

> strsplit(x,' ')

**[[1]]**

1. "Mazda" "RX4"

给你一个list，存储按照空格分开后的substrings

对上面的结果再去第一个list中的第一个元素——就会返回第一个单词Mazda

> strsplit(x,' ')**[[1]][1]**

1. "Mazda"

快速的方法——sapply

**sapply(****strsplit(as.character(rownames(mtcars)), “ ”),”[[“,1)**

这里不用as.character也是可以的，因为rownames(mtcars)本身已经是character，但是有的时候怕是factor，所以习惯转换成chr一下

> str(rownames(mtcars))

chr [1:32] "Mazda RX4" "Mazda RX4 Wag" "Datsun 710" "Hornet 4 Drive" ...

拆解：首先把每一行都根据空格拆开。得到32个list，每个list中存放着拆开后的单词

> strsplit(rownames(mtcars), " ")

[[1]]

[1] "Mazda" "RX4"

[[2]]

[1] "Mazda" "RX4" "Wag"

[[3]]

[1] "Datsun" "710"

[[4]]

[1] "Hornet" "4" "Drive"

[[5]]

[1] "Hornet" "Sportabout"

[[6]]

[1] "Valiant"

[[7]]

[1] "Duster" "360"

[[8]]

[1] "Merc" "240D"

[[9]]

[1] "Merc" "230"

[[10]]

[1] "Merc" "280"

[[11]]

[1] "Merc" "280C"

[[12]]

[1] "Merc" "450SE"

[[13]]

[1] "Merc" "450SL"

[[14]]

[1] "Merc" "450SLC"

[[15]]

[1] "Cadillac" "Fleetwood"

[[16]]

[1] "Lincoln" "Continental"

[[17]]

[1] "Chrysler" "Imperial"

[[18]]

[1] "Fiat" "128"

[[19]]

[1] "Honda" "Civic"

[[20]]

[1] "Toyota" "Corolla"

[[21]]

[1] "Toyota" "Corona"

[[22]]

[1] "Dodge" "Challenger"

[[23]]

[1] "AMC" "Javelin"

[[24]]

[1] "Camaro" "Z28"

[[25]]

[1] "Pontiac" "Firebird"

[[26]]

[1] "Fiat" "X1-9"

[[27]]

[1] "Porsche" "914-2"

[[28]]

[1] "Lotus" "Europa"

[[29]]

[1] "Ford" "Pantera" "L"

[[30]]

[1] "Ferrari" "Dino"

[[31]]

[1] "Maserati" "Bora"

[[32]]

[1] "Volvo" "142E"

接下来，用sapply(拆成单词放在32个list中,”[[“,1) 取出每个list中的第一个元素，就是我们要的brand。

两个中括号**”[[”**的意思就是 要**从list中取出元素**。

> sapply(strsplit(as.character(rownames(mtcars)), " "),"[[",1)

[1] "Mazda" "Mazda" "Datsun" "Hornet" "Hornet" "Valiant" "Duster"

[8] "Merc" "Merc" "Merc" "Merc" "Merc" "Merc" "Merc"

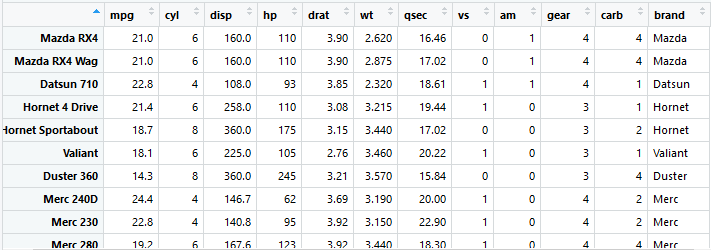
[15] "Cadillac" "Lincoln" "Chrysler" "Fiat" "Honda" "Toyota" "Toyota"

[22] "Dodge" "AMC" "Camaro" "Pontiac" "Fiat" "Porsche" "Lotus"

[29] "Ford" "Ferrari" "Maserati" "Volvo"

最后就是创建一个新的列叫brand然后把刚刚的赋给这个列。

mtcars$brand = sapply(strsplit(as.character(rownames(mtcars)), “ ”),”[[“,1)



可以直接敲View(mtcars)或者在environment->data中找这个数据，看创建的新的列长啥样。

作业3：

Here is a 8\*8 chessboard。You have a place start from the top left corner of the board and make movements following these rules:

Each time throw a dice, if it is

1. > move 1 step to the right,
2. >move 1 step to the bottom,
3. >move 1 step to the left,
4. >move 1 step to the top,

5 or 6 ->do not move

If you are about to move out of the border, move 2 steps to the opposite direction.

1. Create a matrix to represent the chessboard in R. Use 0/1 to represent if your chess has been to a given cell.
2. Write a function in R called move\_chess to calculate the position after next movement. It should take two parameters (x,y) as current position and return (x\_next,y\_next)
3. Use the move\_chess function and write a code to have your chess move on the board until it has been to all cells. Count how many movements did you take.

> #first create a 8\*8 matrix

> chess\_board = matrix(0, nrow = 8, ncol = 8)

> chess\_board

[,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8]

[1,] 0 0 0 0 0 0 0 0

[2,] 0 0 0 0 0 0 0 0

[3,] 0 0 0 0 0 0 0 0

[4,] 0 0 0 0 0 0 0 0

[5,] 0 0 0 0 0 0 0 0

[6,] 0 0 0 0 0 0 0 0

[7,] 0 0 0 0 0 0 0 0

[8,] 0 0 0 0 0 0 0 0

从左上角开始，已经去过[1,1]这个点了：

chess\_board[1,1]=1

扔骰子一次——产生一个随机数：

sample(1:6,1)

描述rules：

(x,y)为当前位置

if(sample(1:6,1)==1){

y\_next = y+1

}

elseif(sample(1:6,1)==2){

x\_next = x+1

}

elseif(sample(1:6,1)==3){

y\_next = y-1

}

elseif(sample(1:6,1)==4){

x\_next = x-1

}

else{

啥都不做

}

return(x\_next,y\_next)

写一个Loop让code一直在board上跑，停止条件——min(chess\_board)==1 break

或者写while：

moves = 0

while(min(chess\_board)<1){

move\_chess(x,y)

moves = moves + 1——How many movements did you take，每跑一轮就记一次

}

#1 答案：

chess\_board = matrix(0, nrow = 8, ncol = 8)

#2答案：

move\_chess<-function(x,y){

x\_next = x

y\_next = y

# throw a dice, get a random number

dice = sample(1:6, 1)

#move chess based on the rules

if(dice==1){

y\_next = y+1

}else if(dice==2){

x\_next = x+1

}

else if(dice==3){

y\_next = y-1

}

else if(dice==4){

x\_next = x-1

}

# if go out of boarder, return to the original point and move 2 steps to the opposite direction

if (x\_next < 1){

x\_next = x\_next + 1 + 2

} else if (x\_next > 8) {

x\_next = x\_next - 1 - 2

} else if (y\_next < 1){

y\_next = y\_next + 1 + 2

} else if (y\_next > 8) {

y\_next = y\_next - 1 - 2

}

return(x\_next,y\_next)

}

#3答案：

chess\_board = matrix(0, nrow = 8, ncol = 8)

chess\_board[1,1] = 1

x = 1

y = 1

moves = 0

#只要没走遍棋盘就一直走（跑上面写的函数按照rules移动棋子更新横纵坐标），所到之处标记为1，每走一步move+1（计步），直到棋盘最小值不再小于1，即全部值都为1，就停止

while(min(chess\_board)<1){

x = move\_chess(x,y)[1]

y = move\_chess(x,y)[2]

chess\_board[x,y] = 1

moves = moves + 1

}

moves