Lapply, order, by and other interesting things

No tags yet. + Add Tag

Sort replies by: Oldest first Newest first Most popular

Vladimir Kirilenko COMMUNITY TA · 18 days ago

As I see, already began to appear on specific issues related to data sorting and the ability to do all the job just a couple of the commands. Theoretically it can be done, but you almost certainly will face certain difficulties.Therefore, to avoid repetition of the same answers to the same questions I need to make some explanations.
But I won't be able to write such a brilliant post, as did our colleague Allan M. Due on previous course. So I'll just copy it here.

Ok, there is a lot in there for reasonably short post. :)

First where are you getting the idea that "it is not good practise to split a data frame and then use lapply"? That doesn't sound right to me.

This won't run

```
data.states <- lapply (split (data.rel, data.rel$State),order(as.numeric(data.rel[,3]),data.re
l[,1],na.last = NA,                                    decreasing = FALSE)
```

because this is not a function

```
order(as.numeric(data.rel[,3]),data.rel[,1],na.last = NA,                                    de
creasing = FALSE)
```

So lets see if we can get that idea to work. Explain what is going on. And show that it is essentially the same as by()

The help page for lapply says

```
lapply(X, FUN, ...)
```

FUN there is the same as the function in by()

So if you use by() you have to work out the function. And the function in by() and the function in

apply() are going to be the exact same function. So you aren't going to buy anything with by(). :)

You can use by(), but you still are going to have your problem of how to work out how to use order()

So lets look at a simple example of how to use your own function with both by() and lapply(). Here is a little made up example.

```
 mydf
   outcome Hospital state
1       20     hos1    MN
2        9     hos2    AR
3       48     hos3    AR
4       49     hos3    WA
5       26     hos2    AR
6       27     hos1    WA
7       41     hos2    WA
8       28     hos1    WA
9       21     hos1    AR
10      11     hos3    WA
```

We can split that by state

```
mysplit <- split(mydf,mydf$state)
> mysplit
$AR
  outcome Hospital state
2       9     hos2    AR
3      48     hos3    AR
5      26     hos2    AR
9      21     hos1    AR

$MN
  outcome Hospital state
1      20     hos1    MN

$WA
   outcome Hospital state
4       49     hos3    WA
6       27     hos1    WA
7       41     hos2    WA
8       28     hos1    WA
10      11     hos3    WA
```

Now we can use lapply to order them using an anonymous function

```
 myordered<-lapply(mysplit,function(x) {x[order(x$outcome),]})
> myordered
$AR
  outcome Hospital state
2       9     hos2    AR
9      21     hos1    AR
5      26     hos2    AR
3      48     hos3    AR

$MN
  outcome Hospital state
1      20     hos1    MN

$WA
   outcome Hospital state
10      11     hos3    WA
6       27     hos1    WA
8       28     hos1    WA
7       41     hos2    WA
4       49     hos3    WA
```

So what does this do

```
lapply(mysplit,function(x) {x[order(x$outcome),]})
```

mysplit gets passed to lapply one object/element (here a data frame) at a time - and we are going to assign that object to x (we could call it whatever we want).

Then we create an anonymous function that gets x and does something to it

So the first data frame is going to come in for AR. the function will order ARs data frame based on the outcome colum. we could have also written this as

```
myordered<-lapply(mysplit,function(x) {x[order(x[,1]),]})
```

because outcome is column 1 in my example.

Then that ordered data is returned, lapply continues its work and gets the second data frame (MN) and x becomes MN, MN is ordered. And so on. So we loop through each state's data frame. Ordering as we go.

If we want to use by() it, it is exactly the same conceptually. We have to make the function the same way. And it is the same code either way.

```
> by(mydf,mydf$state,function(x) {x[order(x[,1]),]})
mydf$state: AR
  outcome Hospital state
2       9     hos2    AR
9      21     hos1    AR
5      26     hos2    AR
3      48     hos3    AR
------------------------------------------------------------------------------------------------
mydf$state: MN
  outcome Hospital state
1      20     hos1    MN
------------------------------------------------------------------------------------------------
mydf$state: WA
   outcome Hospital state
10      11     hos3    WA
6       27     hos1    WA
8       28     hos1    WA
7       41     hos2    WA
4       49     hos3    WA
```

We could have used.

```
by(mydf,mydf$state,function(x) {x[order(x$outcome),]})
```

They are the same.

One thing about by(). The results of by are stored in an object of class by, which is a kind of list. lapply also produces a list

In the end, by() is conceptually the same as

```
> lapply(split(mydf,mydf$state),function(x) {x[order(x[,1]),]})
$AR
  outcome Hospital state
2       9     hos2    AR
9      21     hos1    AR
5      26     hos2    AR
3      48     hos3    AR

$MN
  outcome Hospital state
1      20     hos1    MN

$WA
   outcome Hospital state
10      11     hos3    WA
6       27     hos1    WA
```

```
8       28      hos1    WA
7       41      hos2    WA
4       49      hos3    WA
```

All thanks to him! VK

20 · flag

Dan Goldfield · 18 days ago

and now thanks to you.....

So when you split, R keeps each subset as a different element in a vector...this allows the lapply to loop through each subset at once?

0 · flag

+ Comment

Jayant Singh Signature Track · 17 days ago

Dear Vladimir,
First , i want to thank you for the useful information in this page. It helped me to get a solid start on assignment 3.

I have a basic question about factor variables. Suppose we have a factor variable, called "alphabets", which has 26 levels. My concern is how can we check that the character "aa" is one of the vaues or not. The "for loop" process sounds very rusty. Can you please suggest a faster option? thanks for the help.

0 · flag

Dan Goldfield · 17 days ago

although not in the lapply other posts used: isit="aa" test <- "zz" "ww" "uu" "az" "za" "ab"....."aa"...."mq".....

if(is.element(isit,test)){do something} or if(!is.element(isit,test)){do something} identical() or !identical (works if you are looping through test element by element. if(any(test==isit)){then do something} (searches through list for a match) match() (not sure about this, but I saw some forum traffic about it.

0 · flag


Leonard Greski Signature Track · 17 days ago

Hello Jayant. To answer your question we need to know whether the factor is a column in a dataframe, or the variable used to split a dataframe into a list.

If it is a column in a dataframe, you can use

```
result <- dataframe[dataframe$alphabets == "aa",]
```

to subset the data.

If you're trying to find the object in a list that had been split using alphabets as a factor, then to find the object where alphabets = "aa" you would use the following code:

```
if (myList$aa != NULL) {
   ## myList has a factor of AA for alphabets
} else {
   ## myList does not have a factor of AA
}
```

Len

0 · flag

 Vladimir Kirilenko COMMUNITY TA · 17 days ago

Maybe I don't exactly understand the question, but I think this is the one of the most efficient ways of search in the factors.

```
# prepare the data
> data = read.csv("outcome-of-care-measures.csv")
> temp = data[,1:6]
> pattern  <- 'ZU'
> class(temp$City)
[1] "factor"

# search pattern
> temp[grep(pattern,temp$City),]
     Provider.Number                                Hospital.Name
Address.1
1000          110190                       FLINT RIVER HOSPITAL 509 SUMTER STR
EET, BOX 770
2722          320060 ZUNI COMPREHENSIVE COMMUNITY HEALTH CENTER            ROU
TE 301 NORTH
     Address.2 Address.3      City
1000        NA        NA MONTEZUMA
2722        NA        NA      ZUNI
```

but not in the levels of the factors:

```
> temp[grep(pattern,levels(temp$City)),]
     Provider.Number                         Hospital.Name        Address.1 Add
ress.2
1700          181315 EPHRAIM MCDOWELL FORT LOGAN HOSPITAL 110 METKER TRAIL
NA
2926          340001   CAROLINAS MEDICAL CENTER-NORTHEAST  920 CHURCH ST N
NA
     Address.3     City
```

```
1700         NA STANFORD
2926         NA  CONCORD
```

0 · flag

+ Comment

Jayant Singh Signature Track · 16 days ago

Thanks for all the valuable suggestions, friends. It was very helpful.
Presently, I am stuck in a very intriguing situation in part 2(find the best hospital in a state) problem. I hope I can get some help here.

After splitting the data set into 54 data frames (arranged in a list), I am using the function;

new<- lapply(name_split,function(y) {y[order(y$Mortality),]})

to arrange the mortality rate (for the outcome) in increasing order for each state. The reason, I am stuck is that I saw that this function is not sorting the data for each type of outcome; heart attack, heart failure and pneumonia. I was able to get the correct answer for the first two output examples for outcome=heart attack. But when I use state=MD and outcome=heart failure, the above statement is giving me the rating sth like this:

```
1887  WESTERN MARYLAND REGIONAL MEDICAL CENTER    MD      12.1
1882        MEDSTAR MONTGOMERY MEDICAL CENTER    MD      12.6
1903         UPPER CHESAPEAKE MEDICAL CENTER     MD      12.6
1910              ATLANTIC GENERAL HOSPITAL      MD      13.3
1892               CARROLL HOSPITAL CENTER       MD      13.5
1881        GARRETT COUNTY MEMORIAL HOSPITAL     MD        14
1885           ANNE ARUNDEL MEDICAL CENTER       MD        14
1907         MEDSTAR GOOD SAMARITAN HOSPITAL     MD       7.4
1909              FORT WASHINGTON HOSPITAL       MD       8.6
1879    MEDSTAR FRANKLIN SQUARE MEDICAL CENTER    MD       8.8
```

As we can see, the rate is decreasing and then increasing, rather than increasing always. I have only shown a part of the result. Can you please suggest sth, what might be happening? the reason I am surprised is when I use heart attack for outcome, it works fine. Any help is greatly appreciated.

0 · flag

Dan Goldfield · 16 days ago

new<- lapply(name_split,function(y) {y[order(y$Mortality),]})

function(y) calls y. Y loops through each subset ordering by mortality rate correct?

I'm assuming your name_split is a data.frame split by state. Also, it looks like you are ordering my mortality only, you need a secondary order on hospital name to handle ties.

Is there a row index for each subset? I'm wrestling with how to extract the Nth row

append(new[n,1],new[n,2]) for each subset if there are now row indicies. I can do it in a for/loop but I've tried all day and can't swing it with an lapply.

Thanks,

↑ 1 ↓ · flag

 Shyamprasad Molugu · 13 days ago

just finished this part. do we need to use split and lapply for this part? i got it by subset and order operations. first did a subset to get the required columns, deleting NA's and then subset again for a particular state. sort the resultant subset to get the best hospital in a state.

↑ 4 ↓ · flag

Ruotong Yang Signature Track · 12 days ago

@Shyamprasad Molugu,

If do as you said, when sorting the resultant subset should I use the order function as above in the example the TA given?

↑ 0 ↓ · flag

 Shyamprasad Molugu · 12 days ago

Yes, its of the format --- x[order(coll 1, coll 2, ..),]

↑ 1 ↓ · flag

 Shyamprasad Molugu · 12 days ago

Yes, its of the format --- x[order(coll 1, coll 2, ..),]

↑ 0 ↓ · flag

 Rick Henderson Signature Track · 8 days ago

I have something like this. I have a complete data frame with just the hospital name, the state, and the specified outcome mortality rates, and I have it split by state so that it is now a list of dataframes. But I can't figure out how to get the output in the right format. I tried using the sorting techniques in the A3 tutorial but I couldn't get that to work either. Looks like I'm getting a mere 56% on this assignment and I wish I knew how much this would affect my final grade.

↑ 0 ↓ · flag

 Leonard Greski Signature Track · 8 days ago

If you have 56% on the final assignment, that means you lost 11 points from the total. With the swirl assignment you can obtain 5 points of extra credit beyond the total of 100 points for the course. Therefore, if you got perfect scores on all other work, the maximum score you could obtain is 105 - 11 = 94, or enough for a pass with distinction.

The entire grading rubric is listed in the syllabus https://class.coursera.org/rprog-030/wiki/syllabus.

regards,

Len

0 · flag

+ Comment

Jayant Singh Signature Track · 16 days ago

Hello Dan,
my main concern is that when I use outcome=Heart attack, it works fine. But for the other two outcomes,the function is not working properly.Can you please give some suggestion, what might be going wrong?

thanx

0 · flag

Leonard Greski Signature Track · 16 days ago

Make sure you're sorting the data file by outcome and then hospital name, to get the correct tiebreaker as described in the assignment instructions. Also, what is $Mortality? This reference should be dataframe[,outcome]. or dataframe$outcome, or the column id associated with outcome so the data is sorted by the outcome that was input as an argument to the function.

regards,

Len

1 · flag

+ Comment

Jayant Singh Signature Track · 16 days ago

Dear Leonard,
Thanks a lot for the message.
 Actually I just gave names to the columns of my data frame using names function;

names(Name)<-c("Hospital","State","Mortality")
I think we can name the columns as we want. Please let me know if I am wrong.And Mortality contains the data for the corresponding outcome(i.e.heart attack, heart failure or pneumonia) from the spreadsheet.
Then I did a splitting using following command;

name_split<-split(Name,as.factor(Name$State))

that works fine for me irrespective of what do we have for the outcome.

The main problem is when I use lapply to sort as per the Mortality rate, 3rd column of my data frame. When I enter outcome=heart attack the ordering is correct, but for the other two cases, the ordering is not working.

new_split<-lapply(name_split,function(y) {y[order(y$Mortality),]})

This is the place, where I am a bit surprised. After sorting as per the mortality rate, (and making sure it is correct), I will try to break the hospital name tie.But my problem is in the lapply step. Can you please suggest, what might be wrong?

thanks for helping me.

0 · flag

 Leonard Greski Signature Track · 16 days ago

Thanks for the clarification on how you named the $Mortality column.

Have you used as.numeric() to reset the values of $Mortality to numeric before sorting the dataframe? Since we needed to use colClasses = "character" on the read.csv() command, we need to convert the mortality column to numeric before sorting. You also should sort before split.

regards,

Len

0 · flag

+ Comment

Jayant Singh Signature Track · 16 days ago

Hello,
Thanks for the suggestion. Yes initially, all the columns are of class character.

As a follow up, I have a data frame; "name" (before splitting) which contains only non missing values and has 3 columns, hospital name, state, and mortality rate. I checked name$mortality, which is showing as factor variable. I am giving some values here;

11.4 15.2 11.3 13.6 13.8 12.5

As suggested, I did convert it to numeric, using as.numeric(name$mortality), then I got a numeric vector, with these values;
15 53 14 37 39 26

I have a very basic question. I don't understand, why do the values change so much? I would like to understand this difference. if you can help, it will be great. thanx

0 · flag

Gauthamraj · 12 days ago

Hello Jayant.
I don't know if you have solved this problem yet or not. I too encountered the same problem. If you use as.numeric(as.character(name$mortality)), the values will be converted as desired.
I hope this helps.

Regards,
Gautham

0 · flag

+ Comment

Irina Chinenova Signature Track · 16 days ago

Very helpful thread, almost got there.

BUT I am just struggling on how pass values ("heart attack", "heart failure", "pneumonia") to the function argument outcome.

I have created my own data frame with required columns and passed them the names of those values above:

```
mydata <- data.frame("hospital" = data[,2], "state" = data[,7], "heart attack" = data[,11],
                     "heart failure" = data[,17], "pneumonia" = data[,23])
```

I have also specified values or outcome in the function header best <- function (state, outcome = c("heart attack", "heart failure", "pneumonia"))

But when I am running this bit of code in lappy ..{x[order(x[,outcome],... I get the error:

Error in `[.data.frame`(x, , outcome) : undefined columns selected

Please help, what am I doing wrong here?

1 · flag

Leonard Greski Signature Track · 16 days ago

Hello Irina.

I used the outcome to retrieve the column number from the data frame, and then used the column number in the order code.

```
colOutcome <- which(colnames(theData) == outcome)
```

Other posters on the forum used if / else logic to set the column number for outcome, based on the outcome argument passed into the function, and named this column "outcome" rather than using a column name that included a space, as listed in https://class.coursera.org/rprog-030/forum/thread?thread_id=984#post-4127.

Information on dealing with spaces in R columns is located at: http://stackoverflow.com/questions/12744282/how-to-deal-with-spaces-in-column-names.

regards,

Len

↑ 2 ↓ · flag

Irina Chinenova Signature Track · 15 days ago

thanks, got it done through if/else approach

↑ 0 ↓ · flag

+ Comment

Irina Chinenova Signature Track · 16 days ago

Thank you, Len.

I tried if/else logic, but didn't get it to work. I get character(0) as a final result. But if I run individual commands with specific column numbers in console, it works just fine. So, it is all about how to define argument outcome.

I don't understand why in pollutanmean.R when we assigned values of nitrate and sulfate to pollutant argument, it all worked just fine. There was no need to define pollutant in the body of the function.

while in best.R, when I assign values to output in the head of the function, it doesn't work.

↑ 1 ↓ · flag

A post was deleted

+ Comment

Jayant Singh Signature Track · 15 days ago

Actually Irina, in case of pollutant mean, the data set had column names as sulfate and nitrate. so you did not have to specify the name in function body.that the reason. hope it helps.

Regards

0 · flag

Irina Chinenova Signature Track · 15 days ago

yup, makes sense. thanks!

0 · flag

+ Comment

Jayant Singh Signature Track · 15 days ago

Dear friends,
Can anyone suggest me what is the default value of "num" in last part of assignment 3? is it 54 or 0.
My output is matching the rest of example outputs, but I am not sure about the default value.

Any suggestion is appreciated.
thanx

0 · flag

Ivana Lipnerova · 15 days ago

Default value is "best, so it should be 1, if you have hospitals ranked from best to worst, from lowest death rates to highest death rates

0 · flag

Leonard Greski Signature Track · 15 days ago

...and this means you have to handle both character ("best", "worst") and numeric values for num in your code.

Len

0 · flag

+ Comment

Ivana Lipnerova · 15 days ago

BTW I've been reading this forum, because I'm stuck with something. There's one thing that strucks me:
Most of people seems to use lapply as described in first post in this thread. I wonder what logic is behind it? Why do you work with all the data, create enormous list (it has more then 600 Kb on my computer!), order it (another 600+ kb file) and only then ask for specific combination of state, outcome and rank?
I find it way easier and lets say more efficient to subset the whole dataset for just a state I'm working

with. This reduces amount of memory used and moreover I can look at my data and actually see how it looks like in a simple not-so-long-table.
Why, when there is no requirement of work with more than one state? I have to say that I'm working on part 2, but from reading about part 3, I still don't see need for work with whole huge dataset.

4 · flag

Donald F Coffin Signature Track · 15 days ago

Ivana,

I took the same approach you are proposing for all but part 4. I agree reducing the amount of data to only reflect the requested state reduces the amount of memory required, as well as improving the overall performance of the function.

0 · flag

 Vladimir Kirilenko COMMUNITY TA · 14 days ago

I agree with you, Ivana. Subsetting is a very efficient way to reduce data volume and the number of possible errors.

0 · flag

Michael Stelly Signature Track · 11 days ago

As an example, by subsetting for the state input parameter and necessary columns first, I can reduce the data space from 4706 obs with 46 variables to 40 obs with 5 variables.

/rantOn
Maybe I'm biased because of my background with other programming languages, but for me, R syntax is frustratingly archaic. I'm struggling to understand even the basic concepts of data manipulation due to the hoops one must jump through to simply target a particular data row/column.
/rantOff

0 · flag

+ Comment

Jayant Singh Signature Track · 15 days ago

Thank you Ivana and Leonard for the advice.

0 · flag

+ Comment

Dan Goldfield · 12 days ago

Finally after reading lots of posts and seeing lots of examples I got my first lapply to work. I've split my list and can use lapply for case="best" and case=n I've having trouble with case="worst"

I know I want to get the length of each sub listed state...and I can do this

```
  Len <- lapply(g_split,function(g_split) length(g_split$hospital))
```

Then I want to assign that length to elem so I can capture that row:

```
if (identical("worst",num)){elem <- Len}
```

Now I can capture the appropriate row of my list:

```
g_extract1 <- lapply(g_split,function(g_split) g_split[elem,1])
```

However it's not working because Len is list of ALL lengths for my split list instead of a single element relevant to the sub listed item. I don't know how to overcome this.

If this was in a for/loop, I could just pull the row index for the i in id I was in, but I'm pushing myself to try to solve it with lapply.

Any help is appreciated.

Update: Wahooo....got it to work. Never mind. It may seem silly, but I'm really proud of myself for figuring out how to use lapply instead of a for/loop. Still not totally efficient code, but that is for another time.

0 · flag

Anonymous · 12 days ago

What did you end up doing?

0 · flag

Dan Goldfield · 11 days ago

tail() instead of finding the length() and extracting the nth row.

0 · flag

+ Comment

Ruotong Yang Signature Track · 11 days ago

I just did the same as the top thread, but the result I got is confusing. I have no idea why the order of the state "AR" is not correct:

```
> mysplit <- split(mydf, mydf$state)
> mysplit
$AR
  outcome hospital state
2       9     hos2    AR
3      48     hos3    AR
```

```
5      26     hos2    AR
9      21     hos1    AR

$MN
  outcome hospital state
1      20     hos1    MN

$WA
   outcome hospital state
4       49     hos3    WA
6       27     hos1    WA
7       41     hos2    WA
8       28     hos1    WA
10      11     hos3    WA

> myordered <- lapply(mysplit,function(x) {x[order(x$outcome),]})
> myordered
$AR
  outcome hospital state
9      21     hos1    AR
5      26     hos2    AR
3      48     hos3    AR
2       9     hos2    AR

$MN
  outcome hospital state
1      20     hos1    MN

$WA
   outcome hospital state
10      11     hos3    WA
6       27     hos1    WA
8       28     hos1    WA
7       41     hos2    WA
4       49     hos3    WA
```

Can someone help me answer this?

↑ 0 ↓ · flag

 Shyamprasad Molugu · 11 days ago

Check the class of x in function(x). If it is a list then coerce it to a data frame by x<-as.data.frame(x) and then apply the order function

↑ 0 ↓ · flag

Dan Goldfield · 11 days ago

If you do all your ordering first, you don't have to do your order using lapply.

0 · flag

Michael Stelly Signature Track · 11 days ago

@Shyamprasad: mysplit fails with

```
mysplit <- as.data.frame(split(outcome,outcome$State))

Error in data.frame(AK = list(Provider.Number = c("020001", "020006",  :
  arguments imply differing number of rows: 17, 98, 77, 341, 72, 32, 8, 6, 180, 132,
1, 19, 109, 30, 179, 124, 118, 96, 114, 68, 45, 37, 134, 133, 108, 83, 54, 112, 36, 9
0, 26, 65, 40, 28, 185, 170, 126, 59, 175, 51, 12, 63, 48, 116, 370, 42, 87, 2, 15, 8
8, 125, 29
```

outcome is the original data frame defined as

```
    outcome <-
        read.csv(
            "data/outcome-of-care-measures.csv", colClasses = "character",stringsAsFa
ctors = F,na.strings = c("Not Available")
        )
```

So, how did you make this work? In the meantime, I'll try @Dan suggestion.

0 · flag

Shyamprasad Molugu · 11 days ago

split returns a list which you need for lapply function. dont convert mysplit to a data frame. pass mysplit to lapply and in the function(x) you can convert each individual element of mysplit to a data frame and then apply order function. In my code i found each individual element of mysplit to be a list, i converted it to a data frame before sorting it.
again it all depends on rest of the code. I was commenting for a possible reason for the above error.

0 · flag

Anonymous · 11 days ago

Why do you want your `split` result to be a data frame? You want it to be a *list of data frames* which is exactly the split output:

```
mysplit = list("AL" = df1, "AK" = df2, "AZ" = df3, ... )
```

The number of rows of each individual data frame might be different, as some states will have more records than others -- so that's why you get an error when attempting to glue them together.

The idea of using "split" was to produce a list that you can then work with using `lapply` (the "l" there standing for "list").

@Ruotong, I suspect that *none* of your states are technically ordering properly, you just didn't have an $2 \leq outcome \leq 9$ in those states. It looks like your outcome column is being ordered as a character vector, rather than a numeric vector -- so as characters, "1", "131", and "1.275" all come before "2", etc.

0 · flag

Michael Stelly Signature Track · 11 days ago

The syntax for the API is still a jumbled mess to me. So, I don't have any idea why I would want a data.frame vs. a list. @Anonymous: the incredulous tone of your comment implies that we should know better. Not all of us do. Perhaps that's why you posted anonymously. Nonetheless, I found a solution that makes sense to me and works. I avoided all this complexity by reducing the working data set first.

0 · flag

Anonymous · 11 days ago

Oh, I apologize! I *certainly* didn't mean to come off incredulous. Only attempting to steer you in the right direction by pointing out what questions you might consider while examining the code issues.

I'm glad you found a working method, nonetheless.

In the one discussed here, when you split the data frame, the idea is to obtain a list of individual data frames (one for each state) -- that way, you can then use `lapply(x, FUN)` where `x` is that list of data frames, and `FUN` is some function that is *applied* to each *list element* (i.e, each individual data frame).

This way, you do the same "some processing code" on the dataset of each state; namely, getting back the correct hospital name (or something of that nature).

I hope that makes sense!

0 · flag

Michael Stelly Signature Track · 11 days ago

I appreciate the reply. I only mentioned it because the Internet has no sense of humor, and my intent is often misinterpreted especially with a multilingual audience. No offense taken.

0 · flag

+ Comment

New post

To ensure a positive and productive discussion, please read our forum posting policies before posting.

B *I* Link <code> Pic Math Edit: Rich Preview

Make this post anonymous to other students

Subscribe to this thread at the same time

Add post