Rank all without loops

You are subscribed. Unsubscribe

rankall × + Add Tag

Sort replies by: Oldest first Newest first Most popular

Anonymous · 15 days ago

I was able to solve Rank All problem (in all parts) but I was just wondering if anybody did it without the use of loops? I did the Best and Rank Hospital without loops but with Rank All I resorted to loops.

Thanks in advance for quality response.

0 · flag

Daniel Nordlund · 15 days ago

If by "without the use of loops" you mean without for() loops, then the answer is yes Obviously, the "apply" functions (along with a lot of other functions) do looping "behind the scenes".

1 · flag

Anonymous · 14 days ago

Yes, what I mean is without the need to use for loops.
I know conceptually that I could use apply function or one of its family (which I did with my previous assignments) but I was just wary of traversing 2 levels deep on the loop without the need again of using a nested 'for' loops.

I am assuming that you did not use loops in your case on rank all? Or did you just assume that it is doable?

Because I have a hint that it is doable but could not execute it yet correctly.

Thanks for your response by the way.

0 · flag

 Leonard Greski Signature Track · 14 days ago

The trick to completing the assignment without for() loops is figuring out how create the output dataset in a way that it is not a list, since the requirement is to return a single data frame from the function, sorted by state. You also need to figure out how to assign the correct state abbreviations for the rows that are missing. It's definitely possible, however.

regards,

Len

1 · flag

Daniel Nordlund · 14 days ago

Yes, I programmed rankall without any explicit for loops. I also did not explicitly use any of the apply family of functions. Here is an outline of what I did:

1. read the data and immediately renamed the state, hospital name, and 3 outcome variables to match what the assignment called for.
2. subset the data to the three variables of interest ('state', 'hospital', and the outcome) and at the same time removed rows where outcome was NA. Then converted outcome to numeric.
3. ordered the data by 'state', outcome, and 'hospital', and then added a variable with the in state ranking using the functions order(), and ave() with seq_along().
4. picked out the rows of interest where the num parameter was equal to the within state rank.
5. finally, I merged a data frame of state abbreviations with the selected rows (from step 4) to ensure that each state was represented in the final data frame that was returned.

Excluding comments, it was a total of 11 lines of code. That includes some additional book-keeping to set up calls to functions, For example, three of the lines were if() statements to test the validity of the outcome parameter, and to determine the value of the num parameter to decide how to order the data and index the final data frame in steps 3, 4, and 5.

13 · flag

Anonymous · 14 days ago

Thank you Daniel for showing the pseudocode of your solution.

I'll try to apply what you have here and hopefully could make it work.

Thanks again.

0 · flag

 Keith Slaughter · 13 days ago

Daniel, I am curious how, with this approach, you handled the "worst" case with respect to your #4, given that "worst" could be a different rank value for each state. Thanks.

1 · flag

Daniel Nordlund · 13 days ago

I used the 'descending' parameter in the order() function.

```
if(num == 'worst') descend <- TRUE else descend <- FALSE
```

```
if(num %in% c('best','worst')) num <- 1
```

Then in the order function I set 'descending = descend'. So if num = 'worst' sort descending, otherwise sort ascending. Then in either the 'best' or the 'worst' selection, the row you want will come first. Therefore, in either case pick the first item (i.e. num <- 1).

↑ 2 ↓ · flag

Anonymous · 12 days ago

Daniel,

Thank you (and others here) once again for your tip. I finally made it work using **ave()**, **seq()** and **merge()**.

By the way when you say "descend" you probably refer to **decreasing = descend**?

But if we do that does it not also include the State and also the Hospital Name? Because if you do it would have different results. Unless if you just sort it by column target (e.g. pneumonia) and hospital name after splitting it earlier to States.

What I did is that I use if and put negative sign on the second column or the required column (e.g. pneumonia)

Example:

```
sorted_df <- if (tolower(num)=="worst")  x[order(x[,7],-x[,n],x[,2)]
      else x[order(x[,7],x[,n],x[,2)]
```

You could also use **decreasing = ifelse(tolower(num)=='worst',T,F)**

But overall thank you again for your tip.

Now, looking at the answers from below it looks like that R is really a prolific language. You could attack any problem from different angles or constructs and come up with the result, probably most than other programming languages I assume.

↑ 1 ↓ · flag

Daniel Nordlund · 12 days ago

Yes, state and hospital name were included: first by state to keep all the hospitals in a state together (I did not split the data up), second by outcome to make sure the the worst hospital

in the state was first in the state group, and last by hospital name to make sure that the tie-breaker rule was correctly implemented.

The tie-breaker was if two hospitals had the same outcome rate, pick the hospital that sorted first in ascending order. That means to get the correct worst hospital in case of a tie, I needed the hospital names to sorted in reverse order as well. That is, to get the worst, i needed to select the hospital that would have sorted last in ascending order, so for the worst case to sort first, I also needed the hospital names to be reverse sorted.

The states were then correctly re-ordered when I merged back the state names to get "potentially" missing states. If you didn't sort the hospital names in reverse order, then you would potentially get the wrong hospital if there was a tie for worst.

In the future, the rankall test should probably have at least one state where there was a tie for last place.

↑ 0 ↓ · flag

Anonymous · 12 days ago

Hm, that sounds confusing, Daniel -- is there a way to sort by *one* argument DESC, and by another ASC? There has to be, right? And it would avoid this.

The way I handled `num` was to take the first element if it was "best", the last element if it was "worst", and the `num`-th element in other numerical cases. For that, I used `head` and `tail` but presumably `length(...)` could have worked equally -- would that not have worked how you did it?

I'm sort of at a loss to where ave/seq is coming into all of this. Did you find this solution simpler than using the `apply` family?

↑ 1 ↓ · flag

Anonymous · 12 days ago

You can sort it in descending order by using the minus sign and therefore you could make a column ascending and another one descending.

For example in our assignment we want to sort by State (column 7) and column of interest (either 11, 17 or 23) and Hospital Names (column 2). So, if you want the State ascending and column of interest descending (let's say for the sake of example column 11) and Hospital Names ascending again. Then it would look like this:

outcome_data[order(x[,7], -x[,-11], x[,2]),]

On the **ave()** and **seq()** functions, it allows you to create a ranking for each state and therefore would be easier to filter out say rank 20 for all states. The ave() function allow you to group let's say per state and then use a function (seq) to create a sequence of number from 1 to how many hospitals on that state. By default ave use *mean* as the FUN so you need to change *FUN = seq*.

2 · flag

Daniel Nordlund · 11 days ago

I will try to clear up the confusion. Here is some sample data

```
> outcome_data <- structure(list(hospital = c("abc hosp", "bcd hosp", "mno hosp",
  "wxy hosp", "def hosp", "lmn hosp", "uvw hosp", "xyz hosp", "zzz hosp"
  ), state = c("WA", "WA", "WA", "WA", "WY", "WY", "WY", "WY",
  "WY"), rate = c(10L, 5L, 8L, 10L, 6L, 7L, 9L, 5L, 3L)), .Names = c("hospital",
  "state", "rate"), row.names = c(3L, 1L, 2L, 4L, 6L, 7L, 8L, 5L,
  9L), class = "data.frame")
>
> outcome_data
  hospital state rate
3 abc hosp    WA   10
1 bcd hosp    WA    5
2 mno hosp    WA    8
4 wxy hosp    WA   10
6 def hosp    WY    6
7 lmn hosp    WY    7
8 uvw hosp    WY    9
5 xyz hosp    WY    5
9 zzz hosp    WY    3
>
```

Now for num equal to 'best' (set num=1) or some specific numeric value, sort the data in increasing order by state, outcome rate, and hospital name

```
> #order increasing
> tmp <- outcome_data[order(outcome_data[,2],outcome_data[,3],outcome_data[,1]),]
> # assign within state ranks
> tmp$rank <- with(tmp, ave(state,state,FUN=seq_along))
> tmp
  hospital state rate rank
1 bcd hosp    WA    5    1
2 mno hosp    WA    8    2
3 abc hosp    WA   10    3
4 wxy hosp    WA   10    4
9 zzz hosp    WY    3    1
5 xyz hosp    WY    5    2
6 def hosp    WY    6    3
7 lmn hosp    WY    7    4
8 uvw hosp    WY    9    5
```

```
>
```

Now it can be seen in this simple example that the best hospitals in WA and WY are 'bcd hosp' and 'zzz hosp', respectively. The worst hospitals are 'wxy hosp' and 'usv hosp', respectively. From this dataframe, I can get n-th ranked hospital in each state with a simple

```
tmp[tmp[,'rank'] == num,]
```

For the case of worst hospital, I just do the sort in decreasing order for all variables. Now the ranking is from worst to best, and I don't need to worry about how many hospitals are in each state, I just take rank = 1.

```
> #order decreasing
> tmp <- outcome_data[order(outcome_data[,2],outcome_data[,3],outcome_data[,1],decrea
sing=TRUE),]
> tmp$rank <- with(tmp, ave(state,state,FUN=seq_along))
> tmp[tmp[,'rank'] == num,] ## where num=1
  hospital state rate rank
8 uvw hosp    WY    9    1
4 wxy hosp    WA   10    1
```

I get the same 'worst hospitals as seen above. However, if I negate just the rate variable to get the reverse sort, then the worst hospital doesn't match with what was found before.

```
> tmp <- outcome_data[order(outcome_data[,2],-outcome_data[,3],outcome_data[,1]),]
> tmp$rank <- with(tmp, ave(state,state,FUN=seq_along))
> tmp[tmp[,'rank'] == num,]
  hospital state rate rank
3 abc hosp    WA   10    1
8 uvw hosp    WY    9    1
>
```

Here, 'abc hosp' is incorrectly reported as worst in WA.

↑ 1 ↓ · flag

A post was deleted

Michael Stelly Signature Track · 10 days ago

From an earlier thread, there looks like a double negation that occurs for negative sorting

```
outcome_data[ order(x[,7], -x[,-11], x[,2]), ]
```

In your example, Daniel, I don't see that pattern. That may indicate a starting point to debug.

```
tmp <- outcome_data[order(outcome_data[,2],-outcome_data[,3],outcome_data[,1]),]
```

0 · flag

Daniel Nordlund · 10 days ago

No, that code will not work like the original poster is apparently expecting (or it just may be a typo). That syntax will remove column 11 from x[] and use all the other columns, which is not what is wanted here.

0 · flag

Tomasz Wojtas Signature Track · 9 days ago

What about NAs in your example? I used trying make rankhospital():

```
tmp <- data[order(data[,7], data[,outcome], data[,2]),]
```

but when I run the function all NAs of hospital names are gone (mabye that's why result 9 and 10 in commit() are incorrect). How to force function to unclude NA values?

0 · flag

Vladimir Kirilenko COMMUNITY TA · 9 days ago

```
?order
```

In help:

`na.last` for controlling the treatment of `NA`s. If `TRUE`, missing values in the data are put last; if `FALSE`, they are put first; if `NA`, they are removed (see 'Note'.)

0 · flag

Tomasz Wojtas Signature Track · 9 days ago

Adding parameter na.last with any value doesn't solve the issue... Let me put all my code here.

```
rankall <- function(outcome="heart attack", num="best")
{
```

```
        data=read.csv("outcome-of-care-measures.csv", colClasses="character")

        if(outcome=="heart attack") outcome <- 11
        if(outcome=="heart failure") outcome <- 17
        if(outcome=="pneumonia") outcome <- 23
        ##changing outcome according to particular reason of the death

        if(num=="best") num <- 1
        if (num=="worst")
        {
                num <- 1
                tmp <- data[order(data[,7], data[,outcome], data[,2],decreasin
g=TRUE, na.last=TRUE),]
        }
/CUT/
        tmp1
}
```

```
> head(rankall("heart attack", 20), 3)
                             hospital state
1      D W MCMILLAN MEMORIAL HOSPITAL    AL
2   ARKANSAS METHODIST MEDICAL CENTER    AR
3 JOHN C LINCOLN DEER VALLEY HOSPITAL    AZ
```

I expected the in the first row... What did I miss?

0 · flag

 Vladimir Kirilenko COMMUNITY TA · 9 days ago

Maybe this:
read.csv(....other options..., na.strings = "Not Available")
EDIT:
No, I checked your code with this option and it outputs wrong result.
Let me see...

And do not post full code, please.

0 · flag

A post was deleted

 Vladimir Kirilenko COMMUNITY TA · 9 days ago

I suppose, that the problem is in these lines:

```
tmp$rank <- as.numeric(with(tmp, ave(State, State, FUN=seq_along)))
```

```
tmp <- tmp[tmp[,'rank']==num,]
```

Your code generates NULL result instead of NA, if number of hospitals is less than the input, as I understood.

↑ 0 ↓ · flag

Tomasz Wojtas Signature Track · 9 days ago

Ok, sorry for posting full code - I thought that if it was not working there is nothing wrong with sharing it. But I will not do that again :)

Thanks for your help. I thing I need more time to understand that...

↑ 0 ↓ · flag

John Power · 7 days ago

Len, Vladimir, Daniel --> I was also having trouble with using apply instead of for loops and I think a concrete example is what I need.

```
csvdata <- read.csv("outcome-of-care-measures.csv", colClasses = "character")
statedata <- split(csvdata,csvdata$State)    #statedata is a list of data frames
(e.g. statedata$AK, statedata$AL, etc
#used a for loop to convert the 'outcome' column from char to numeric
  for (i in 1:length(statedata)) {
    statedata[[i]][[numoutcome]] <- suppressWarnings(as.numeric(statedata[[i]][[numou
tcome]]))
  }
```

The above for loop worked but it seems like it should be easy to use the apply family but I am stuck. **How can the above for loop be replaced with one of the apply family of functions?**

---John

↑ 0 ↓ · flag

John Power · 7 days ago

Daniel, I am trying to work through your example (it works perfectly) but I don't understand the command **tmp$rank <- with(tmp, ave(state,state,FUN=seq_along))**. even after trying to unpack it.

Step 1) It starts out ok when I run the **with** portion and see it returns a vector of characters that obviously gets assigned to tmp$rank in your code.

```
with(tmp, ave(state,state,FUN=seq_along))
[1] "1" "2" "3" "4" "1" "2" "3" "4" "5"
```

Step 2). Now I am confused when I run **ave** by itself it does not work.

```
ave(state,state,FUN=seq_along)
Error in interaction(...) : object 'state' not found
```

What is 'state'? It is not defined anywhere.

The documentation on "**ave**" that comes with RStudio does not help since there is no example even close to this. I've spent 30 minutes on google, stack overflow and cannot find any complete documentation on "**ave**" that explains your use of it.

Any explanation would be great.

---John

0 · flag

 Edmund Julian L. Ofilada · 7 days ago

Hi John,
i'm trying to learn this solution same as you are. State is the column of state abbreviations. I practiced using his approach on the airquality dataset. I changed the first state into the column of interest. In the rankall assignement, it is the column of either Pneumonia, heart failure or heart attack. In the airquality dataset, any column you choose to order and later on rank. The second state sort of set the environment which you want to rank, in this case 54 states. Each state will have all the hospitals in it ranked independent of other states. So when the function specifies the 20th ranked hospital,you get a dataframe containing all the 20th ranked hospital in every state, provided that there are 20 hospital with data in that particular state. Otherwise, it will be returned as NA. His approach is very interesting, specially the idea of reversing the order to rank the worst hospital number one. HOpe this helps.

0 · flag

Daniel Nordlund · 7 days ago

John,

the ave() is inside a with() function. This allows me to shortcut the variable references. If you want to call ave() by itself,
you need to specify where the variables are coming from, i.e.

```
ave(tmp[,'state'],tmp[,'state'],FUN=seq_along)
```

The first variable is the variable you want FUN to operate on, and the second variable is a grouping variable. That is, you want FUN to work with groups of rows. For this particular application, the first variable could be any variable in the data frame (as long as the data frame is properly sorted). The function is not really ranking the data (that is accomplished with the sort). The ave() call is just labelling the sort order within each state so you can use it to find the rank that you want.

If you were interested in using ave() to get the mean mortality rate within each state, then

the first variable would indeed matter. You would need to do something like

```
ave(tmp[,outcome],tmp[,'state'],FUN=mean)
```

This says you want the mean of the first variable, tmp[,outcome], but you want the mean calculated separately by state. Note: you are not limited to a single grouping variable.

0 · flag

 Edmund Julian L. Ofilada · 5 days ago

Thanks for that great explanation Daniel. In a way, ave, is like tapply, from what you said.

0 · flag

+ Comment

 Vladimir Kirilenko COMMUNITY TA · 15 days ago

Also I saw in one code using bunch of if/else conditions and in another using merge and aggregate.

1 · flag

Anonymous · 14 days ago

Vladimir,

I don't want to use a bunch of if..else but probably I will explore merge and aggregate.

Thank you for your reply by the way.

0 · flag

 Vladimir Kirilenko COMMUNITY TA · 14 days ago

I fully agree with you. If/else pack in 99% is a sign of an inexperienced programmer.

-6 · flag

Anonymous · 12 days ago

Vladimir,

Ther merge() function really help me to include States that have NAs. But I did not use the aggregate function on my case.

Thank you.

0 · flag

Anonymous · 12 days ago

> I fully agree with you. If/else pack in 99% is a sign of an inexperienced programmer.

What did you see specifically?

I mean, is there a better way to handle some of these things the code asks us to do? Namely, handling an argument that can be either numeric, "best", or "worst".

0 · flag

 Vladimir Kirilenko COMMUNITY TA · 12 days ago

It is not always possible to do the code without any conditions, but almost always they can be minimized.
As you know, you can use many approaches to check conditions.
For example, you can navigate through all of the options with the conditions. Or you can first make subsetting of the data you want, and then to process them by function arguments.
Also you can collect all the column names in the vector and then do a single check for the presence of input in this vector with %in%, or you can do multi-level checks on each input argument.
And so on and so forth...

1 · flag

Anonymous · 12 days ago

"*If/else pack in 99% is a sign of an inexperienced programmer.*"

Makes me laugh to see naive blanket statements like this.

2 · flag

 Vladimir Kirilenko COMMUNITY TA · 12 days ago

Ok, obviously, you've met plenty of programmers who write code with a chain of a dozen conditions despite the possibility of pre-fetch data.
I have little experience, such pro I have not seen.

-1 · flag

Anonymous · 11 days ago

Maybe it's a language issue. But your comment reads like 99% of programmers who use if/else are inexperienced. And nobody said anything about a "dozen" conditions. Personally, I think the most I ever used was about a half dozen or so calculating sliding scale insurance rates. I preferred nested if/else because it was more efficient. But back then number of operations was more important than storage or memory.

1 · flag

 Leonard Greski Signature Track · 11 days ago

Hello Anon. I interpreted Vladimir's comment in a similar vein to other expert developers who

caution against use of if / else if chains where the language supports a simpler mechanism to handle different cases, such as this Visual Basic example from Jeff Atwood at codinghorror.com: http://blog.codinghorror.com/the-tyranny-of-elseif/.

Also, in the classic programmer's book *Code Complete,* Steve McConnell makes a similar argument in Chapter 14: Using Conditionals (pp. 311 - 322).

...and yes, R does have a switch() function, as illustrated at http://stackoverflow.com/questions/7825501/switch-statement-usage.

I'll admit that I didn't bother to research the switch() function before submitting my answers to programming assignment 3. Why? Because it's OK to learn iteratively: make it work, make it right, make it fast. My program ran fast enough as originally written. Since it produced the correct results, I didn't spend a lot of additional time trying to squeeze every ounce of performance from the code.

Instead, I've been spending time on the bulletin boards trying to answer other people's questions. I prefer to help other people finish the programming assignments so they don't get discouraged and give up. I've learned a lot more about R these past 3.5 weeks by answering questions on the bulletin boards than I have by completing the assignments.

regards,

Len

↑ 5 ↓ · flag

Anonymous · 11 days ago

Len,

Admittedly, I did not use switch() also since I don't know it exist. I just use a single 'if' condition using %in% operator and then subset my data frame that contains both the outcome and the associated column number.

```
 df <- data.frame(out_name=c("heart attack","heart failure", "pneumonia"), out_co
l=c(11,17,23))
    if (!outcome %in% df[["out_name"]])
       stop("invalid outcome")
     col <- df[tolower(df[["out_name"]])==outcome,]["out_col"]
```

which could also be written as :

```
df <- data.frame(out_name=c("heart attack","heart failure", "pneumonia"), out_col=c(1
1,17,23))
    col <- df[tolower(df[["out_name"]])==outcome,]["out_col"]
    if (nrow(col)==0)
```

```
        stop("invalid outcome")
```

To tell you I have use many programming languages but I haven't seen a language (that I so far use) that use switch() like that as of R. It looks elegant and handy to use.

Thank you for sharing this.

0 · flag

ly · 11 days ago

Thanks, Len, that's very thorough. I can see where everyone is coming from when it comes to if/else *abuse*.

In both of his examples, Jeff Atwood favors the Visual Basic analog of `switch`. In the second example, though, all that's functionally missing is an "ELSE" case, right?...but I do agree that the SELECT CASE structure is easier to read and makes for better coding practice.

Let's see if I can understand this topic better with an example from my code. For this assignment, I used `switch` to handle the `outcome` argument. For `num`, I found it trickier. I revised my code in order to make it work, but I found that R treats numerical values unexpectedly, so that I had to switch on `as.character(num)`.

Original:

```
if(num == "best") {
    # Return first index.
    result <- head(e$Hospital.Name, 1)
} else if (num == "worst") {
    # Return last index.
    result <- tail(e$Hospital.Name, 1)
} else {
    # Return num-th index; NA if out of range.
    result <- e$Hospital.Name[num]
}
```

New:

```
switch(as.character(num),
         "best" = {
           result <- head(e$Hospital.Name, 1)
         },
         "worst" = {
```

```
            result <- tail(e$Hospital.Name, 1)
        },
        # Default case.
            result <- e$Hospital.Name[num]
)
```

Note that in the second case, `num` has been previously validated as either "best", "worst", or numeric. We weren't told to validate the num argument, but it seemed sensible to throw an error for invalid values. I could have handled the validation of `num` within the switch block, but it also made sense to me to throw that "stop" error *before* data processing, if possible.

This seems fine to me for handling an argument such as `num`, which can take valid values among different classes. Are there better ways? Was the first way (or both!) very poor? What could or should I have done different? (I won't be offended; after all, we are here to learn).

By the way, I agree wholeheartedly with your last two paragraphs about learning iteratively. Trying to diagnose other students' issues and more importantly, seeing how they are diagnosed by people such as Vladimir, yourself, and others, has been very instructive for me -- at least as much as the assignments themselves.

↑ 2 ↓ · flag

 Leonard Greski Signature Track · 11 days ago

Hello Ly. In the specific Visual Basic example I referenced, dt.DayOfWeek is always between 1 and 7, because dt is a date. Therefore, no CASE ELSE clause is required. Generally speaking, however, the coder should include a CASE ELSE clause. Also, the cases should be listed in the statement by descending frequency of use. For example, if 80% of the calls to the function will be "best", 10% to "worst" and another 10% to all other numbers, the "best" case should be the first clause in the switch() function.

I don't understand what you mean by "R treats numeric values unexpectedly." I found that using as.numeric() before processing the outcome data was required to get the right results. If you use as.numeric() to convert the num argument to a number after you've eliminated "best" and "worst" cases, it should behave fine as a number.

regards,

Len

↑ 0 ↓ · flag

ly · 11 days ago

Thanks! Appreciate the comment about descending frequency of use. That is a good tip.

Here is what I mean about numeric values. Granted, I may not be understanding this completely correctly. I hope I am being precise enough.

In `switch(EXPR, ...)` you have `EXPR` that can be either a "number or character string", but the behavior seems to be quite different depending on which. When `EXPR` is a number, `switch` no longer tests if that number is *equal* to any of the named `...` alternatives, it will

get the index among `...` specified by the value of the number.

For example, here is a series of identical switch statements, acting on different `EXPR`:

```
> switch("best", "best" = print("best"), "worst" = print("worst"), print("default"))
[1] "best"
```

As intended.

```
> switch(1, "best" = print("best"), "worst" = print("worst"), print("default"))
[1] "best"
```

Switch runs the *1st* alternative, regardless of its name.

```
> switch(3, "best" = print("best"), "worst" = print("worst"), print("default"))
[1] "default"
```

Switch runs the *3rd* alternative, which was *meant* to be the "CASE ELSE".

```
> switch(7, "best" = print("best"), "worst" = print("worst"), print("default"))
```

Switch runs nothing and returns NULL here because the alternatives list is only 3 elements long.

```
> switch("7", "best" = print("best"), "worst" = print("worst"), print("default"))
[1] "default"
```

Now Switch sees that "7" is neither "best" nor "worst", and goes with the "CASE ELSE" code.

I suppose this should have made sense. The `...` list, *if* it is a named list, can only be named with character names (and those can't be compared numerically).

So ultiamtely, I had to make sure I was doing `switch(as.character(num))` instead of `switch(num)`, to avoid unexpected behavior when `num` was numeric. A switch statement in R with a named list of alternatives must be performed on a character `EXPR`, and this was surprising to me. I expected numeric values of `num` to simply not match any of the named alternatives I provided.

Thinking about that more, it seems if `EXPR` is numeric, then you simply don't get a CASE ELSE. You'd have to check if the switch returned NULL outside of the switch statement. Right?

1 · flag

 Ayse Tezcan Signature Track · 9 days ago

and thank you @**Leonard Greski**. i have learned so much from these forum and making sooo many mistakes....

it has been very helpful.

1 · flag

+ Comment

 Al Warren · 14 days ago

In my dplyr version, I used summarize() with a function to select hospital name. Summarize also collapses groups to a single row. When you pass a column to a function in summarize, the function receives the column values as a vector. I just returned the hospital name based on num. I chose a function mainly for readability.

```
## At this point, we have a data frame with three columns and no NA values
## Column names were renamed to hospital, value, and state

checkState <- function(hospital) {
    ## for best return the first hospital
    ## for worst return the last hospital
    ## return hospital[num]
}

df <- df %>%
    ## Create the state groups
    group_by(state) %>%
    ## Sort by outcome value then hospital name
    arrange(value, hospital) %>%
    ## Collapse groups to a single row per group and select hospital
    summarize(hospital=checkState(hospital)) %>%
    ## Select columns for output
    select(hospital, state)

## return the data frame with row names
```

2 · flag

Guy Wells · 4 days ago

Neat, but I think you'll get the wrong hospital for "worst" cases where there are ties (as the hospitals are also in ascending order within value). By setting required rank to 1 for "best" and "worst" and inserting an if test in the arrange function in order to sort value descending if "worst", then you should get the correct hospital for all cases and simplify the code:

```
if (num %in% c("best", "worst")) {
        outcome.rank <- 1L
} else {
        outcome.rank <- as.integer(num)
}
```

```
:
:
arrange(if (!num == "worst") {value} else {desc(value)}, hospital)
summarize(hospital = hospital[outcome.rank])
:
:
```

↑ 0 ↓ · flag

Daniel Nordlund · 4 days ago

No, that is not right. The 'best' hospital is the hospital that sorts first when ordered ascending by rate and then by hospital name. The 'worst' hospital is the hospital that sorts last under those same conditions. If you want to get the 'worst' hospital to sort first, you need to order the the data in descending order by both rate and hospital name. See my discussion of this earlier in this thread.

↑ 0 ↓ · flag

Guy Wells · 3 days ago

That's how I initially interpreted the spec for handling ties in the "worst" case scenario but on reflection I thought that you'd still want to choose the hospital that would be appear first in ascending name order for the same tied "worst" 30 day mortality rate outcome (same basis as for "best"). It was also a little more challenging for me on my first attempt when using the order() function. Either approach can be applied to the modified dplyr code (i.e. without checkstate()) , but I guess clarification with the spec writer on this should be sought.

↑ 0 ↓ · flag

Daniel Nordlund · 3 days ago

For the instructions to be consistent, I don't think any further clarification is needed. Look at the sample data I provided (copied from above).

```
  hospital state rate rank
1 bcd hosp    WA    5    1
2 mno hosp    WA    8    2
3 abc hosp    WA   10    3
4 wxy hosp    WA   10    4
9 zzz hosp    WY    3    1
5 xyz hosp    WY    5    2
6 def hosp    WY    6    3
7 lmn hosp    WY    7    4
8 uvw hosp    WY    9    5
```

If you wanted to find which hospital was ranked 3rd, you would return 'abc hosp' for WA. If you wanted the hospital that ranked 4th, you would return 'wxy hosp' for WA. Which hospital do you think should be returned for WA if requested to return the 'worst' hospital? If you are

going to sort the rates in descending order to get the worst hospital, the only way to get consistent answers is to also sort hospital in descending order.

0 · flag

Guy Wells · 2 days ago

That's a fair point and valid approach and yes you would get the same hospital on choosing "worst" case for WA and if you explicitly requested the 4th ranked WA hospital which happens to be the last in the State's rankings. But your tie breaking method is not consistent for "best" and "worst" cases (i.e. for "best" you use ascending hospital name given equal rate, whilst for "worse" you use descending hospital name given equal rate).

I based my approach on the sample data provided in the **Handling ties** section of the assignment document:

```
> head(texas)
                       Hospital.Name Rate Rank
3935       FORT DUNCAN MEDICAL CENTER  8.1    1
4085  TOMBALL REGIONAL MEDICAL CENTER  8.5    2
4103 CYPRESS FAIRBANKS MEDICAL CENTER  8.7    3
3954           DETAR HOSPITAL NAVARRO  8.7    4
4010           METHODIST HOSPITAL,THE  8.8    5
3962  MISSION REGIONAL MEDICAL CENTER  8.8    6
```

Say we assume this is all the data for Texas (even though it's a head() summary), then the worst rate is 8.8 and I would choose "METHODIST HOSPITAL, THE" (but obviously I wouldn't get that one if I explicitly requested rank = 6), however this approach is consistent with choosing "best", for example if there was another hospital with rate 8.1 having name "ACME MEDICAL CENTER" say, then it would be chosen as best. Likewise in the assignment document example "because Cypress comes before Detar alphabetically, Cypress is ranked number 3 in this scheme and Detar is ranked number 4". So this approach yields a consistent method in handling ties.

Apologies if this post comes across as a bit laboured, but I wanted to convey my rationale for choosing this approach, as I said earlier I did approach it originally in the same manner as you. The scenario of tied worst rankings doesn't come up in the assignment as both ways yield correct answers for all parts - so maybe clarification is required.

0 · flag


Leonard Greski Signature Track · 19 hours ago

This is why for the "worst" scenario, I used nrow() to index the row to be returned, rather than implementing multiple sort approaches based on the num argument.

regards,

Len

0 · flag

Daniel Nordlund · 17 hours ago

That certainly works. There are other approaches that will do the same thing without sorting differently. If one attaches rank to the records, then using max(rank) will get you the worst. One could also use the tail() function.

But Guy is interpreting the instructions for breaking ties as "if there is a tie for any rank, return the record that sorts first." So for the sample data in the immediately preceding post, Guy would have returned

```
4010          METHODIST HOSPITAL,THE  8.8    5
```

Since there was a tie for worst, his method would return the one that sorts first. I simply pointed out that I didn't think that could be a possible interpretation because that would lead to an inconsistency with the example data where if you asked for rank=6, you get a different result than if you asked for 'worst'.

I suggested in an earlier post that in future classes, a tie for 'worst' in some state should be added. But as part of the overall learning experience, this points up the necessity to make sure one thoroughly understands what a client wants/needs before gathering, massaging, and analyzing data.

0 · flag

 Leonard Greski Signature Track · 17 hours ago

Agreed. I interpreted the instruction to mean that "worst" indicates the one that sorted last in the event of ties on the ranking, which would make it consistent with the rest of the instructions.

regards,

Len

0 · flag

+ Comment

Nakazawa Shigeki Signature Track · 14 days ago

I think I managed to make one with an apply family, which could be in a few lines eventually,
I haven't fully checked this, please tell me if there are errors.

First, I referred to this post:
https://class.coursera.org/rprog-030/forum/thread?thread_id=984
and made a list (x) which has lists by state, sorted by the mortality rates and hospital names.

```
head(x[[1]], 5)  # e.g. state: AK, outcome: heart atack

                      Hospital.Name State Hospital.30.Day.Death..Mortality..Rates.fro
m.Heart.Attack
99  PROVIDENCE ALASKA MEDICAL CENTER    AK
13.4
103         ALASKA REGIONAL HOSPITAL    AK
14.5
102      FAIRBANKS MEMORIAL HOSPITAL    AK
15.5
106     ALASKA NATIVE MEDICAL CENTER    AK
15.7
100   MAT-SU REGIONAL MEDICAL CENTER    AK
17.7
```

Then, used sapply to extract hospital names and change the output to a data frame (x1).
(This is the case of the best hospitals.)

```
x1 <- sapply(x, function(x) x[1,1])   #get the first hospital names in each list.
x1 <- as.data.frame(x1)               #convert the output to a data frame.
head(x1, 5)
                                 x1
AK  PROVIDENCE ALASKA MEDICAL CENTER
AL          CRESTWOOD MEDICAL CENTER
AR           ARKANSAS HEART HOSPITAL
AZ              MAYO CLINIC HOSPITAL
CA GLENDALE ADVENTIST MEDICAL CENTER
```

Similarly, I made another data frame of state (x2),

```
x2 <- sapply(x, function(x) x[1,2])
head(x2, 5)
   x2
AK AK
AL AL
AR AR
AZ AZ
CA CA
```

Combined both and changed labels,

```
x3 <- cbind(x1, x2)
names(x3) <- c("hospital", "state")
head(x3, 5)
                           hospital state
AK  PROVIDENCE ALASKA MEDICAL CENTER    AK
AL          CRESTWOOD MEDICAL CENTER    AL
AR           ARKANSAS HEART HOSPITAL    AR
```

```
AZ              MAYO CLINIC HOSPITAL    AZ
CA GLENDALE ADVENTIST MEDICAL CENTER    CA
```

Actually, the above, as.data.frame, --> cbind --> names, can be put together:

```
x3 <- data.frame(label1=x1, label2=x2 )
```

And the whole process can be combined with the sapply function.
Meantime, NAs are given as <NA>s in this case, which can be dealt with, I think (working now).

↑ 2 ↓ · flag

Anonymous · 14 days ago

Arigato Nakazawa.

I will try to incorporate your solution and see if I could make it work.

↑ 0 ↓ · flag

Matthew Pedone Signature Track · 13 days ago

I would also like to thank you for this post. I was hitting a wall trying to get something like what your "dat1 <- sapply" line does. For some reason, the "x[1, 1]" was eluding me, but once I had that, the rest of the code simply fell into place.

↑ 0 ↓ · flag

A post was deleted

+ Comment

Joseph Ramaswami · 13 days ago

I did it without using loops. I split the data and used sapply to create a matrix of the required values. I then had to rotate the matrix (because I ended up with a 2 row by 54 column matrix instead of the necessary 54 row by 2 column matrix) and return it as a data.frame with the added names.

↑ 1 ↓ · flag

+ Comment

 Eric Vaitl Signature Track · 13 days ago

I think it is possible to use lapply(), which creates a list, then recombine to a data frame without explicit for loops. From my submission:

```
lst<-lapply(split(df,df$state),nth,num)
# lst is a list of data frames. Put them back together as rows of one frame.
do.call(rbind.data.frame,lst)
```

0 · flag

+ Comment

 Joel Berman · 13 days ago

I tried a few ways similar to some of those in this thread to see how they would work. When I have time I may benchmark them and learn the graphics to do the benchmarks.

I decided on split, lapply and unlist so similar to Eric's above, but used

```
data.frame(
        hospital = unlist(result), state = names(result), row.names = names(result)
)
```

R certainly provides for many options. I think any program that is easy to understand and runs in a reasonable time is great.

1 · flag

Anonymous · 12 days ago

Eric and Joel,

Thanks for your answers.

This goes to show that R is a very prolific language and could come up with many solutions that would yield same result probably more than any language. That means we need to master different functions and constructs to come up with the best construct or solution possible to a problem.

0 · flag

+ Comment

Anonymous · 12 days ago

Finally joined the rank of completing rankall. Now it's about rewriting everything to make it more efficient. D:

2 · flag

+ Comment

Devon Ly Signature Track · 11 days ago

Well this one was can I say quite frankly a "ball buster".
My head hurts but in general this are steps I took
1. remove NAs out of the outcome column of the data frame
2. reordered the data frame by state, outcome ratio, hospital name (or negative outcome ratio if looking for the worst)
3. used tapply to split the ordered data frame by state (which create a split list) and used function of seq_along to do the numbering)
4. (after much pain) discovered the unlist function, which used on the above ranking, to recombine it back into a single continuous vector
5. created a new data frame with "unique" US states <- this is important for those pesky states with NA results
6. renamed the column of the unique states to be called "state" <- important for later on read the first line of ?merge description ubber carefully
7. subsetted my ordered data frame, using the unlisted rank list = num input value AND selecting the hospital name and state columns
8. renamed the columns in the subset data frame to "hospital" and "state" <- again read ?merge and see why the naming is important
9. merged my unique US states (from step 5) to my subset data frame (from step 8) and set all.x = TRUE <- that all.x is somewhat important, if in doubt read up on the help
10. for some reason my new merged data frame had state first and hospital second, so had to reorder the dataframe
and shazam!

*I've omitted the best and worst if statements, assuming you've got that handled from the previous exercise.

Have fun.. or in my case "not"

↑ 0 ↓ · flag

Anonymous · 11 days ago

What happened with pesky states with NA results? I didn't encounter this issue at all and I wonder what I am missing.

↑ 0 ↓ · flag

Devon Ly Signature Track · 11 days ago

maybe you did it differently, but if you look at the sample output in the PDF for the command

```
head(rankall("heart attack", 20), 10)
```

you'll see AK, DC & DE all have NA as the hospital.
going straight to subsetting in step 7 doesn't keep any states where there are no hospitals matching the criteria. Which is why I had to create another data frame with all states and merge later on.
The instructions do mention including states with no hospital but it's hard to find as its buried inside the paragraph.

0 · flag

Anonymous · 11 days ago

Ahhhhhhhh, I see now!

I guess I got around that without realizing by subsetting & splitting *first*, and then only handling NA removal within `lapply`. That way all the states are still there.

(Edit): Wait, that's not what I did. Hm.

0 · flag

Anonymous · 11 days ago

For my rankall, I tried to keep myself sane by taking as simple a pseudocode approach as I could:

1. validate arguments
2. subset columns of interest, removing NA values, and split by state -> list of state dataframes as result
3. call lapply on the list with an anonymous function that did everything
4. put the results together in the required format for return

So in this case, all the anonymous function did was:

1. order by "outcome, hospital" (make sure outcome ~ numeric, hospital ~ character!)
2. return either `head(d$hospital, 1)`, `tail(d$hospital, 1)`, or `d$hospital[num]` (the latter of which will return NA if that index is out of range, anyway)

Meaning what I get back from lapply is a list like this:

```
lapplyResult = list("AL" = hospital1, "AK" = hospital2, ...)
```

And so I needed to return two vectors: one, containing the contents of the list, and one containing the names (`c("AL", "AK", ...)`):

```
data.frame(hospital = as.character(lapplyResult), state = names(lapplyResult))
```

Just offering up my approach in case it is easier to follow and make sense of this assignment with :) I know there are other, maybe neater ways of doing it but they've mostly seemed harder to wrap my head around. They are just fine, though, especially if they work -- so good job, no matter what approach you took!

1 · flag

Anonymous · 11 days ago

Addendum:

I realized I was also removing NA values first, but I didn't run into any problems this way, which really probably was an accident. In

```
head(rankall("heart attack", 20), 10)
```

Every state still has hospitals, it was just that in a few cases such as DE that hospital was out of range. If a state had no hospitals with any results at all, it wouldn't have been included in the list, which is wrong. I suppose this could be remedied with a bit of work.

↑ 0 ↓ · flag

Juliette Armour-Wilson Signature Track · 11 days ago

thank you for this comment! I had something very similar but was only doing the ordering in my anonymous function, and then using a for loop to get the best/worst/num row. I am now trying your method of combining the second step into my anonymous function.
However, it keeps returning an error when I test the new lapplyresult. Would you mind clarifying your pseudocode for me?
If: `lapplyresult <- lapply( splitData, function(x) {} )`
do you first assign the ordered 'x' to a new variable, 'd', and then use an if statement to return either `head(d$hospital, 1)`, `tail(d$hospital, 1)`, or `d$hospital[num]` ? Or do you somehow combine this into one step?

Thank you!!

↑ 0 ↓ · flag

Anonymous · 11 days ago

`x$hospital` should work if that is what you've named that column. So what I've got looks something like this...

```
lapplyresult <- lapply(splitData,
    function(x) {
        # make sure x[[outcome]] is numeric
        # order x by x[[outcome]], x$hospital
        # depending on num, assign result to be either head, tail, or ...[num]
        return(result)
    })
```

So it's a rather long anonymous function (well, a few lines), but it made sense. Rather than a sequence of `if\else if` statements, I used `switch`:

```
switch(as.character(num),
    "best" = {
        result <- ...
    },
    "worst" = {
        result <- ...
    },
    # default case, neither "best" nor "worst"
    # num may be coerced numeric, but our function spec seems to assume num is always
```

```
valid
    result <- ...
)
```

It seems like it might be more correct to have `# remove NA values` in the anonymous function, too, but it didn't matter for purposes of the submit script. *Technically* a state may have had only NA values for a particular outcome, which would lead to that state not appearing in the list at all, which I think is wrong.

↑ 1 ↓ · flag

+ Comment

 Rick Henderson Signature Track · 9 days ago

My head is completely swimming after going through all this. I still can't see the benefit of splitting the data by state, and not sure how to use the apply family in this case. Maybe it's because the use of **anonymous functions** seems so wrong to me. If you are going to write an anonymous function, just write the function then call it.

↑ 0 ↓ · flag

+ Comment

 Edmund Julian L. Ofilada · 9 days ago

Hope somebody can reply to my question before the discussion threads close. I tried using the ave() as described in this thread for rankall. I'm stuck with this part of the code:

```
df_nona[, 3] <- as.numeric(df_nona[, 3])              ##Coerce column 3 as. numeric
      ordered_columns <- df_nona[order(df_nona[, 2], df_nona[, 3], df_nona[, 1]), ]
      ordered_columns$rank <- with(ordered_columns, ave(state, state, FUN = seq_along))
      if (num == "best")   {
          num == 1
      }
      else if (is.numeric(num) == TRUE) {
        num <- num
      }
      else if (num == "worst")  {
        num <- nrow
      }
      ranked_df <- ordered_columns[ordered_columns[, "rank"] == num, ]
```

I can't make R read my code for "best" and when i do it manually my results omit the NAs where the states don't have values for that rank. The instructions specifically said that the rows for states without values should be shown with NAs. I also haven't figured out how to write the code for the "worst" scenario. Thanks.

↑ 0 ↓ · flag

 Edper Castro Signature Track · 9 days ago

Edmund,

Here are my comments on your code:

1)
with(ordered_columns, ave(state, state, FUN = seq_along)

On the first 'state' above (inside ave) you should use instead the outcome column desired, namely, whether for Heart Attack which is 11, Heart Failure which is 17 and so on. Because that's where you want to rank the hospitals with, right? So, it would be:

with(ordered_columns, ave(ordered_columns[, target_outcome], ordered_columns$State, FUN = seq_along)

2) You don't need the one below in your if..else. So, remove that.

```
else if (is.numeric(num) == TRUE) {
      num <- num
    }
```

3) You want the nrow, for the 'worst' right? Which is the bottom-most part of the ranking. But you **num <- nrow** cannot do that. Either use tail or in my case I reverse the order/sorting ahead before it comes to ranking my hospitals.

4) Specify, the column that you want to display:

This one below:
ranked_df <- ordered_columns[ordered_columns[, "rank"] == num,]

could change to:

ranked_df <- ordered_columns[ordered_columns[, "rank"] == num, c(2,7)]

5) Don't forget the NAs

You say the NAs are omitted because it only have all the states that have a ranking on specific order say 20th. So, those missing States could be incorporated with NAs as values except for the State name of course using merge() function.

Hope this helps kabayan.

GOD bless.

↑ 0 ↓ · flag

+ Comment

 Edmund Julian L. Ofilada · 9 days ago

Thanks Edper!!! Still have one more question. How will you know which values will be missing in advance so you can use merge(). I haven't used merge() before.

0 · flag

 Edper Castro Signature Track · 9 days ago

See the documentation of merge. And you will see that you could either merge by row (i.e. all.x) or by column (all.y) or both (all). I use the all.y and the missing States will be incorporated in your new data frame.

0 · flag

 Edmund Julian L. Ofilada · 8 days ago

Thanks, I practiced using the airqualtiy dataset and got the results i wanted using ave(). I did look up the help for merge and will try bringing back the missing rows. Thanks for all your help Edper.

0 · flag

+ Comment

 Edmund Julian L. Ofilada · 9 days ago

Thanks also Mr Kirilenko

1 · flag

+ Comment

 Edmund Julian L. Ofilada · 8 days ago

I used the ave() on the airquality dataset and i now know how to use it. Interesting compliment to the split lapply approach. I'm trying now how to use merge() to put back the rows that have been skipped due to absence of values. Thanks everyone for all your posts. It was a great experience this time around. I finished the data scientist course early and jumped in the middle of the course for R programming. That part was a real nightmare. But this time, the second time around more than made up for the lost hair.

1 · flag

 Vladimir Kirilenko COMMUNITY TA · 8 days ago

I`m glad for you, Edmund!

0 · flag

+ Comment

 Edmund Julian L. Ofilada · 7 days ago

Mr. Kirilenko,
Thanks to you and others in this course, I have overcome the first instinct to ask help right away when I hit a blank wall. I was able to make the loopless rankall work in my computer after several hours of work, The only thing left is how to replace the skipped states where data was unavailable. I tried using all the arguments in Merge but it doesn't work for me. Can i ask a few more hints??? I practiced on the airquality data and this is my output:

```
    Ozone Solar.R Wind Temp Month Day rank
```

```
86    108     223  8.0   85     7  25   25
```

```
128    47      95  7.4   87     9   5   25
```

↑ 0 ↓ · flag

 Edmund Julian L. Ofilada · 7 days ago

I made a dataframe:

```
Ozone Solar.R Wind Temp Month Day rank Day.1
```

```
1    NA      NA   NA   NA     5  NA   25    NA
```

```
2    NA      NA   NA   NA     6  NA   25    NA
```

```
3    NA      NA   NA   NA     7  NA   25   NA
```

```
4    NA      NA   NA   NA     8  NA   25    NA
```

```
5    NA      NA   NA   NA     9  NA   25    NA
which I'm trying to merge with the output.  I figure, my error is in the way m
y data frame is contructed as I've tried all the arguments in merge.  Thanks a
gain for all your help
```

↑ 0 ↓ · flag

Steven Barberena Signature Track · 7 days ago

Hint for removing NA's, where colNumber is the column you want to remove the NA's:

dataX <- data[!is.na(data[,colNumber]),]

↑ 0 ↓ · flag

 Edper Castro Signature Track · 7 days ago

Edmund,

The key here is what column is common between the two and how are you going to turn it into unique values of States (the State values are from your original data frame and not the last result) and convert such into a data frame. You could actually also use **State.abb** unfortunately I found out there is no DC State in this global Environmental variable in R.

0 · flag

Steven Barberena Signature Track · 7 days ago

Another Hint: I recommend looking at the unique() function. You can apply this to the state column and get a list of all the states being used in the file.

https://stat.ethz.ch/R-manual/R-devel/library/base/html/unique.html

0 · flag

 Edmund Julian L. Ofilada · 7 days ago

Thanks edper, my output in the airquality dataset has values for the month 7 and 9. There are no values fro the months 5, 6, and 8. In the rankall assignment this missing values should also be returned as NAs. So i made a dataframe with same number of columns, with the column month having the elements 5,6,7,8,9 and the rank column having the elements 25,25,,25,25,25. The datframe with a lot of NA i was hoping will fill in the missing rows for month 5,6,and 8 but i couldn't make it merge.

0 · flag

 Edmund Julian L. Ofilada · 7 days ago

Steven,
in my other solution where I split the dataframe and later on used lapply, NA values were included in the output for states without data for a particular rank. By replacing the column of state with a new list of unique states fro the original data, the NAs in the column state was removed, but NAs were still present in the hospital name, which is what the assignment specified. In the loopless answer offered by Daniel, the states with the missing values on a particular rank were skipped. I need to bring back the NA values for the omitted states. The solution they offered was to use merge. But I have tried all the arguments and I couldn't make it work.

0 · flag

Anonymous · 7 days ago

@Edper - you can use concatenation the four codes for missing territories into a new vector and sort to get them in the right sequence.

Sorry, did not intend to post as Anon...

Len

0 · flag

Steven Barberena Signature Track · 7 days ago

Edmund,

In the data I was provided, all records had hospital names. Not one entry had "NA" as a hospital name. Only columns starting at #11 and above have NA's. You can use Excel to quickly review the CSV files and see what's there. The assignment never said to strip NA's from the hospital names, because there are none. It said to sort the hospital names in alphabetical order should there be several identical matches for the level of outcome to be displayed.

Another hint: Look at the order function.

https://stat.ethz.ch/R-manual/R-devel/library/base/html/order.html

0 · flag

Steven Barberena Signature Track · 7 days ago

Edmund,

I just realized you were referring to the "<NA>" in the return data for the rankall function. Sorry. My first post on how to remove NA could still be used with some minor modifications.

0 · flag

Daniel Nordlund · 7 days ago

Edmund,

what you could do with months (and what I did with states) is to get a vector of unique months from the original data, and then converted it to a 1-column data frame. The following code subsets airquality down to two rows, gets unique months, and merges them back into a final data frame. there is no need to create variables with NA values, it happens automatically in the merge.

```
newAQ <- with(airquality, airquality[Day==1 & !(Month %in% c(5,6,8)),])
months <- data.frame(Month=unique(airquality[,'Month']))
final <- merge(months, newAQ, by='Month', all.x=TRUE)
```

Hope this helps.

0 · flag

 Edmund Julian L. Ofilada · 5 days ago

I got it!!! After so many hours of trial. i got it to work. My mistake was providing too many columns in my data frame. I thought that by including the rank in both data frames plus a column of states in the case of rankall and in the case of airquality a column of months and rank, R would find it easier to merge the data frame to the output that i wanted. When i used a single column dataframe of states in the rankall and a single column of months in airquality, the result was what i wanted. More than 1 column produced an extra column with plus the column name of the column i needed had to be renamed. Thank you all for your input. It's a great motivation and starting point for the next course.

0 · flag

+ Comment

 Edmund Julian L. Ofilada · 5 days ago

Thanks Edper, Steven, Daniel and Mr Kirilenko for all your comments and to coursera for keeping this thread open till i got the right answer.

1 · 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