Lab-Debugging

May 22, 2021

DEBUGGING in R

0.1 Table of Contents

Estimated Time Needed: 15 min

What is debugging and error handling?

What do you get when you try to add **`"a" + 10`***? An error!*

```
[ ]: "a" + 10
```

And what happens to your code if an error occurs? It halts!

```
[ ]: for(i in 1:10){
         #for every number, i, in the sequence of 1,2,3:
         print(i + "a")
         }
```

These are very simple examples, and the sources of the errors are easy to spot. But when it's embedded in a large chunk of code with many parts, it can be difficult to identify *when*, *where*, and *why* an error has occurred. This process of identifying the source of the error and fixing it is called **debugging**.

Error Catching

If you know an error may occur, the best way to handle the error is to **`catch`** the error while it's happening, so it doesn't prevent the script from halting at the error.

No error:

```
[ ]: tryCatch(10 + 10)
```

Error:

```
[ ]: tryCatch("a" + 10) #Error
```

Error Catching with `tryCatch`:

tryCatch first *tries* to run the code, and if it works, it executes the code normally. **But if it results in an error**, you can define what to do instead.

```
[ ]: #If tryCatch detects it will cause an error, print a message instead. Overall,␣
     ↪no error is generated and the code continued to run successfully.

     tryCatch(10 + "a",
              error = function(e) print("Oops, something went wrong!") ) #No error
```

```
[ ]: #If error, return "10a" without an error

     x <- tryCatch(10 + "a", error = function(e) return("10a") ) #No error
     x
```

```
[ ]: tryCatch(
         for(i in 1:3){
             #for every number, i, in the sequence of 1,2,3:
             print(i + "a")
             }
         , error = function(e) print("Found error.") )
```

Warning Catching

Aside from **errors**, there are also **warnings**. Warnings do not halt code, but are displayed when something is perhaps not running the way a user expects.

```
[ ]: as.integer("A") #Converting "A" into an integer warns the user that the value␣
     ↪is converted to NA
```

If needed, you can also use **tryCatch** to catch the warnings as they occur, without producing the warning message:

```
[ ]: tryCatch(as.integer("A"), warning = function(e) print("Warning.") )
```

Scaling R with big data As you learn more about R, if you are interested in exploring platforms that can help you run analyses at scale, you might want to sign up for a free account on IBM Watson Studio, which allows you to run analyses in R with two Spark executors for free.

0.1.1 About the Author:

Hi! It's Kumar Gaurav, the author of this notebook. I hope you found R easy to learn! There's lots more to learn about R but you're well on your way. Feel free to connect with me if you have any questions.