Reading Data

Jayant

2/21/2020

The first two are **read.table** and **read.csv** and these are for reading tabular data.

The function **read lines** is for reading lines of a text file

The **source** function is important for reading R code

The **dget** function is also for reading R code files but it's for reading R objects that have been dparsed into text files.

The **low** and **unserialized** functions are for reading binary objects into R.

read.table function is the most commonly used function for reading data into R.

the first argument is pretty obvious, it's name of a file or the name of a connection

The header is a logical flag indicating whether the first line is a header line

The sep argument stands for separator. it's, it's a string that indicates how the columns are separated.

ColClasses is a character vector whose length is the same length as the number of columns the data set And the character vector indicates what, what is the class of each column the data set.

nrows is the number of rows in the data set

Comment.char is the character string that indicates what's the comments character (default is #)

Skip is the number of lines to skip from the beginning.

Last argument is strings as factors this defaults to true. And the idea is that it, the question is whether you want to encode character variables as factors. So by default. Anytime our read.table encounters a column of data that looks like it's a character variable, it will call, it will assume that what, what you mean to read in, is a factor variable. If you don't me, mean to read this in as a factor variable, then you can set strings as factors equal to false.

You can use read.table usually without specifying any of the other arguments besides the file name.

```
#   data <- read.table("foo.txt") Reads the file foo from the computer
```

It will figure out evrything else on its own.

"data" will be a data frame.

You can tell R all the arguments to make it runs faster and more efficiently

The read.csv function is identical to read.table, except for the key differences that the, the default separator for the read.csv function is the comma, whereas the default separator for read.table is the space.

read.csv is useful for reading csv files . eg from Microsoft excel

The other thing that read.csv specifies is that it always specified header to be equal to true.

Read.table help page read very imp , gives imp hints

Memory req for the table = no. of rows X no. of columns X 8 this is in bytes

the bytes value / (2^20bytes/MB) gives the value in MB , and that divided by 2^10 gives GB

We req 2x the memory to properly read this data