Mu Sigma

A Course in R

Text Manipulation, Date, Apply Functions

Do The Math

Chicago, IL
Bangalore, India
www.mu-sigma.com

May 9th , 2016

Agenda

1. **Text Manipulation**
2. Date Manipulation

Reference: www.gastonsanchez.com

Table of Contents-Text Manipulation

Creating Strings

- **character()** is the function that creates vector objects of type "character". It takes numeric values as arguments and creates a vector of that length, all elements are equal to “ “.

- We can create following type of strings
  - ‘a character string using single quotes’
  - "a character string using double quotes”

- To test if an object is of type "character" , we can use the function **is.character()**

- To convert non-character objects into character strings , we can use the function **as.character()**

- **paste()** is one of the most important functions that we can use to create and build strings
  - paste0 is equivalent to paste with collapsing

Scripts to try

- Make a vector “MyName” with 1st element as your name and the 2nd element as your surname
- Make a new string named “Action” containing “is learning R”
- **Paste** MyName and Action
- **Paste** 1st element of MyName, 2nd element of MyName and Action. Try this using **paste0()**
- Try the above script by putting different separators in the “sep” argument

Codes

```
is.character(Action)
is.character(MyName)
paste(MyName,Action)
paste(MyName[1],MyName[2],Action)
paste0(MyName[1],MyName[2],Action)

#Change separators
paste(MyName[1],MyName[2],sep=‘_')
paste(MyName[1],MyName[2],sep=‘ is weirdly ')
```

Table of Contents-Text Manipulation

Printing Characters

I. Printing values with **print()**
   - print() displays text in quoted form by default
   - If we want to print character strings with no quotes we can set the argument quote = FALSE

II. Unquoted characters with **noquote()**

III. Concatenate and print with **cat()**
   - Numeric and/or complex elements they are automatically converted to character strings
   - By default, the strings are concatenated with a space character as separator

IV. Encoding strings with **format()**
   - format() allows us to format an R object for pretty printing
   - Arguments-width, trim ,justify ("left","right", "centre", "none“),digits and scientific

V. C-style string formatting with **sprintf()**
   - returns a formatted string combining text and variable values.

VI. Converting objects to strings with **toString()**
   - toString() allows us to convert an R object to a character string.

Scripts to try

- Print MyName using the **print()** function then print it without quotes, then use **noquote()**
- Print a numeric and character values using the **cat()** function
- Make a data frame and use different options of *justify* argument in **format()** to print in different styles
- Change the number of digits to the right of the decimal using the *nsmall* argument in **format()**
- Use the *scientific* argument in **format()** to print the large values in scientific notation

- Create a numeric vector containing atleast 7 elements
- Convert it into character vector and print
- Convert it **toString()**. See the difference by observing the double quotes

Codes

- print(MyName)
- print(MyName,quote=F)
- noquote(MyName)

- zz <- data.frame("(row names)"= c("aaaaa", "b"), check.names = FALSE)
- format(zz)
- format(zz, justify = "left")
- format(zz, justify = "centre")

- format(13.7,nsmall=2)
- format(13.7,nsmall=4)
- format(13.5,digits=2)

- ## use of scientific
- format(2^31-1)
- format(2^31-1, scientific = TRUE)

- a<-c(1,2,3,4,5,6,7,8)
- as.character(a)
- toString(a)

Table of Contents-Text Manipulation

Basic text manipulations

I. Count number of characters with **nchar()**
   - nchar() counts the number of characters, while length() only gives the number of elements in a vector

II. Convert to lower case with **tolower()**

III. Convert to upper case with **toupper()**

IV. Upper or lower case conversion with **casefold()**
   - It is a wrapper for both tolower() and toupper()
   - By default, casefold() converts all characters to lower case

V. Character translation with **chartr()**
   - chartr() works is by replacing the characters in old by those indicated in new
   - Old and new must have the same number of characters
   - It can be used for multiple replacements at once

VI. Abbreviate strings with **abbreviate()**

VII. Replace substrings with **substr()**

Scripts to try

- Create a string which contains a sentence with capital and small characters, name it String1
- Create a vector named vector1 which contains all the words in String1
- Calculate the **number of characters** and the **length of both the objects**

- Convert the string to lower case using **tolower()**
- Convert the vector to upper case using **casefold()**

- Use the chartr to translate some characters in a string

- Abbreviate the vector1 to 2 characters

- Use substr() to keep only charaters from 2 to 5 in the vector1

Codes

- String1<-"THiS iS a VeRy WeiRd StrInG, PlEaSe MaKe It BeTteR"
- vector1<-c("THiS","iS","a","VeRy", "WeiRd StrInG", "PlEaSe MaKe","It BeTteR")

- nchar(String1)
- length(String1)
- nchar(vector1)
- length(vector1)

- tolower(String1)
- casefold(vector1,upper=T)

- x <- "MiXeD cAsE 123"
- chartr("iXs", "why", x)

- abbreviate(vector1,2)

- substr(vector1, 2, 5)

Table of Contents-Text Manipulation

Text manipulations with package *“stringr”*

Function	Description	Similar to
Str_c()	string concatenation	paste()
Str_length()	number of characters	nchar()
Str_sub()	extracts substrings	substring()
Str_dup()	duplicates characters	none
Str_trim()	removes leading and trailing whitespace	none
Str_pad()	pads a string	none
Str_wrap()	wraps a string paragraph	strwrap()
Str_trim()	trims a string	none

Note: Try the previous scripts with the stringr package

Table of Contents-Text Manipulation

Functions for Regular expressions

Function	Purpose	Characteristic
grep()	finding regex matches	which elements are matched (index or value)
grepl()	finding regex matches	which elements are matched (TRUE & FALSE)
regexpr()	finding regex matches	positions of the first match
gregexpr()	finding regex matches	positions of all matches
regexec()	finding regex matches	hybrid of regexpr() and gregexpr()
sub()	replacing regex matches	only first match is replaced
gsub()	replacing regex matches	all matches are replaced
strsplit()	splitting regex matches	split vector according to matches

Agenda

1. Text Manipulation
2. **Date Manipulation**

Table of Contents – Date Manipulation

Date handling in different computer systems

- To make computation easier, computers express dates as the number of seconds from a specific point in time which is taken as the zero point
- Microsoft excel works with seconds from January 1, 1900
- Unix (POSIX) follows counting the seconds from January 1, 1970
- SAS has a reference date of January 1, 1960
- R follows the convention of Unix time (January 1, 1970)
  - Dates are denoted as number of days from Jan 1, 1970
  - Time (Date time) is denoted as the number of seconds from Jan 1, 1970

Formats used in date manipulation

Format	Meaning	Example
%a	Weekday as locale’s abbreviated name.	Sun, Mon, ..., Sat (en_US); So, Mo, ..., Sa (de_DE)
%A	Weekday as locale’s full name.	Sunday, Monday, ..., Saturday
%d	Day of the month as a zero-padded decimal number.	01, 02, ..., 31
%b	Month as locale’s abbreviated name.	Jan, Feb, ..., Dec
%w	Weekday as a decimal number, where 0 is Sunday and 6 is Saturday.	0, 1, ..., 6
%B	Month as locale’s full name.	January, February, ..., December
%m	Month as a zero-padded decimal number.	01, 02, ..., 12
%y	Year without century as a zero-padded decimal number.	00, 01, ..., 99
%Y	Year with century as a decimal number.	1970, 1988, 2001, 2013
%H	Hour (24-hour clock) as a zero-padded decimal number.	00, 01, ..., 23
%I	Hour (12-hour clock) as a zero-padded decimal number.	01, 02, ..., 12
%p	Locale’s equivalent of either AM or PM.	AM, PM (en_US); am, pm (de_DE)
%M	Minute as a zero-padded decimal number.	00, 01, ..., 59
%S	Second as a zero-padded decimal number.	00, 01, ..., 59
%z	UTC offset in the form +HHMM or -HHMM (empty string if the the object is naive).	(empty), +0000, -0400, +1030
%Z	Time zone name (empty string if the object is naive).	(empty), UTC, EST, CST
%W	Week number of the year (Monday as the first day of the week) as a decimal number. All days in a new year preceding the first Monday are considered to be in week 0.	00, 01, ..., 53

References

- Text Manipulation
  - http://gastonsanchez.com/Handling_and_Processing_Strings_in_R.pdf

Text_Manipulation.R

- Date Manipulation
  - http://www.cyclismo.org/tutorial/R/time.html
  - http://www.aridhia.com/technical-tutorials/working-with-date-times-and-time-zones-in-r/