Post 2

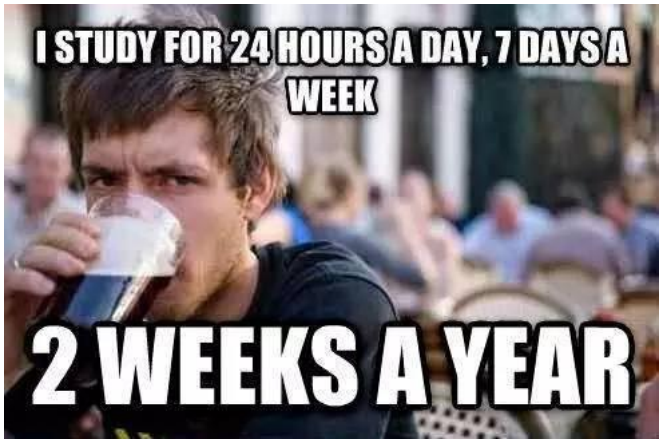
Yang Huang

December 3, 2017

Title: Funny R Packages: `cowsay`, `fortunes` and `fun`

Introduction and Motivation

Being stressed by finishing HW4 in one day, I decided to write something interesting and relaxing on my post2. Also, since the final week is coming, I guess everyone is pretty busy and does not have time and mood to read a boring academic post. I remember the last time when I graded my assigned peer's post1, I spent more than one hour just trying to understand his or her code. Although clearly, it was a great academic paper, I did not enjoy reading it at all. As a matter of fact, I can promise you guys that you would spend less than 10 minutes to finish reading my post.
My post is mainly about amusing string outputs and picture outputs involving using three R packages that I found most interesting and reproducible. Hope you all enjoy it!



Instruction Before Start

First, be sure to install packages you needed

```
# install.packages(c("cowsay", "fortunes", "fun"))
```

Then, load all the packages needed

```
# load packages
library(cowsay)
library(fortunes)
library(fun)
```

About `cowsay`

`cowsay` is a famous program that generates pre-made ASCII pictures of different animals with a message. However, the default animal in this package is cat probably because the contributors are cat people. And I think it would be better if the package is called `catsay`. Check out the example of the default animal below:

```
say('HELL YEAH!')
```

```
##
##  --------------
## HELL YEAH!
##  --------------
##     \
##       \
##         \
##             |\___/|
##           ==) ^Y^ (==
##             \  ^  /
##              )=*=(
##             /     \
##             |     |
##            /| | | |\
##            \| | |_|/\
##       jgs  //_// ___/
##                \_)
##
```

If you don't like cats, it's OK. You can always change the animal argument. So first, let's take a look at what animal arguments we have here:

```
sort(names(animals))
```

```
##  [1] "ant"          "anxiouscat"   "bat"          "bat2"
##  [5] "behindcat"    "bigcat"       "buffalo"      "cat"
##  [9] "chicken"      "clippy"       "cow"          "daemon"
## [13] "endlesshorse" "facecat"      "fish"         "frog"
## [17] "ghost"        "grumpycat"    "hypnotoad"    "longcat"
## [21] "longtailcat"  "monkey"       "mushroom"     "pig"
## [25] "poop"         "pumpkin"      "rabbit"       "shark"
## [29] "shortcat"     "signbunny"    "smallcat"     "snowman"
## [33] "spider"       "stretchycat"  "trilobite"    "turkey"
## [37] "yoda"
```

Oh, great. I will pick my favourite animal here as an example:

```
say("Did you just assume my gender?",
    by = "daemon")
```

```
##
##  -----
## Did you just assume my gender?
##  ------
##     \
##      \
##       \
##              ,        ,
##             /(        )`
##             \ \___   / |
##              /- _  `-/  '
##             (/\/ \ \   /\
##             / /   | `
##             O O   ) /    |
##             `-^--'`<     '
##            (_.)  _  )   /
##             `.___/`    /
##               `-----' /
## <----.     __ / __   \
## <----|====O)))==) \) /====
## <----'    `--' `.__,' \
##              |        |
##               \       /
##        ______( (_  / \______
##      ,'  ,-----'   |        \
##      `--{__________)        \/ [nosig]
```

About `fortune`

Every time I go to Panda Express, I enjoy the moment when I open the fortune cookie and read the words on it. `fortune` is an R package that generates collective fortunes from R community. Thus, I have a special crush on `fortune`.

Check out several examples below:

```
fortune() # generate a random fortune
```

```
##
## Happy families are all alike; every unhappy family is unhappy in its own
## way.
## Leo Tolstoy
##
## and every messy data is messy in its own way - it's easy to define the
## characteristics of a clean dataset (rows are observations, columns are
## variables, columns contain values of consistent types). If you start to
## look at real life data you'll see every way you can imagine data being
## messy (and many that you can't)!
##    -- Hadley Wickham (answering 'in what way messy data sets are messy')
##       R-help (January 2008)
```

```
fortune("dangerous") # search a specific fortune by character
```

```
##
## If you give people a linear model function you give them something
## dangerous.
##    -- John Fox
##       useR! 2004, Vienna (May 2004)
```

```
fortune(200) # search a specific fortune by number
```

```
##
## Bad English is the language of science.
##    -- Jan de Leeuw
##       JSS mailing list (February 2005)
```

By the way, you can use `cowsay` and `fortune` together

```
say(paste(fortune(), collapse = "\n"), by = "hypnotoad")
```

```
##
##
##  -----
## All Glory to the HYPNO TOAD!
##  ------
##     \          ,'``..._   ,'``.
##      \        :,--._:)\,:,._,.:
##       \       :`--,''   :`...';\
##                `.,'       `---'  `.
##               /                   :
##              /                     \
##            ,'                       :\.___,-.
##           `...,---'``````-...._     |:       \
##             (                 )   ;:    )   \  _,-.
##              `.              (   //          `'    \
##               :               `.//  )      )     , ;
##             ,-|`.            _,'/       )    ) ,' ,'
##            (  :`.`-..____..=:.-':     .     _,' ,'
##             `,'\ ``--....-)='    `._,  \  ,') _ '``._
##          _.-/ _ `.       (_)      /     )' ; / \ \`-.'
##         `--(   `-:`.     `' ___..'  _,-'   |/   `.)
##             `-. `.`.``-----``--,  .'
##               |/`.\`'        ,',); SSt
##                   `         (/  (/
##
```

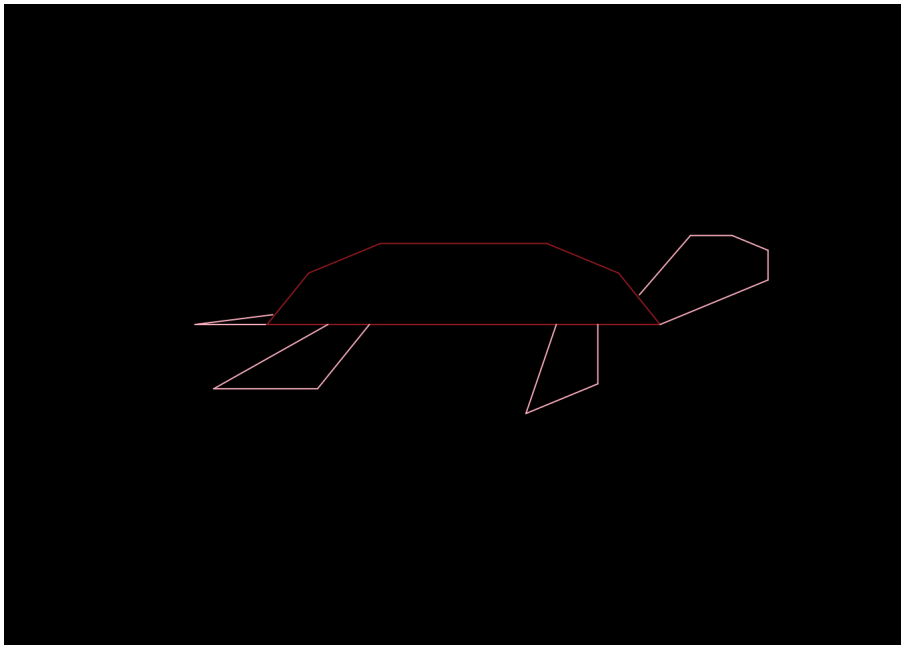
About `fun`

`fun` is an R package of a collection of R games and other funny stuff. But the R games are currently only available on Windows. So here I only use the demo `RealTurtle` to draw the picture of a turtle as an example:

```
demo(RealTurtle,package = "fun") # draw a turtle
```

```
##
##
##  demo(RealTurtle)
##  ---- ~~~~~~~~~~
##
## > ## A "Real" Turtle
## > ## code by Linlin Yan <linlin.yan@cos.name>
## > ## URL: http://cos.name/cn/topic/15876
## > turtle_x <- 0
##
## > turtle_y <- 0
##
## > turtle_direction <- 0
##
## > turtle_color <- "white"
##
## > turtle_drawing <- TRUE
##
## > turtle_init <- function(width = 100, height = 100,
## +     mar = rep(0, 4), bg = "black", ...) {
## +     par(mar = mar)
## +     par(bg = bg)
## +     plot(c(-width, width), c(-height, height), type = "n", xlab = "",
## +         ylab = "", axes = FALSE, ...)
## +     turtle_x <<- 0
## +     turtle_y <<- 0
## +     turtle_direction <<- 0
## + }
##
## > turtle_goto <- function(x, y) {
## +     if (turtle_drawing) {
## +         segments(turtle_x, turtle_y, x, y, col = turtle_color)
## +     }
## +     turtle_x <<- x
## +     turtle_y <<- y
## + }
##
## > turtle_pen_up <- function() {
## +     turtle_drawing <<- FALSE
## + }
```

```
## + }
##
## > turtle_pen_down <- function() {
## +     turtle_drawing <<- TRUE
## + }
##
## > turtle_forward <- function(distance) {
## +     x <- turtle_x + distance * cos(turtle_direction * pi/180)
## +     y <- turtle_y + distance * sin(turtle_direction * pi/180)
## +     turtle_goto(x, y)
## + }
##
## > turtle_turn_left <- function(degree) {
## +     turtle_direction <<- (turtle_direction + degree)%%360
## + }
##
## > turtle_turn_right <- function(degree) {
## +     turtle_direction <<- (turtle_direction - degree)%%360
## + }
##
## > turtle_set_color <- function(col) {
## +     turtle_color <<- col
## + }
##
## > draw_turtle <- function() {
## +     turtle_init()
## +     turtle_set_color("brown")
## +     turtle_forward(50)
## +     turtle_turn_left(120)
## +     turtle_forward(20)
## +     turtle_turn_left(30)
## +     turtle_forward(20)
## +     turtle_turn_left(30)
## +     turtle_forward(40)
## +     turtle_turn_left(30)
## +     turtle_forward(20)
## +     turtle_turn_left(30)
## +     turtle_forward(20)
## +     turtle_turn_left(120)
## +     turtle_forward(50)
## +     turtle_set_color("pink")
## +     turtle_pen_up()
## +     turtle_goto(-20, 0)
## +     turtle_pen_down()
## +     turtle_turn_right(120)
## +     turtle_forward(25)
## +     turtle_turn_right(60)
## +     turtle_forward(25)
## +     turtle_goto(-30, 0)
## +     turtle_turn_left(180)
## +     turtle_pen_up()
## +     turtle_goto(35, 0)
## +     turtle_pen_down()
## +     turtle_turn_right(90)
## +     turtle_forward(20)
## +     turtle_turn_right(60)
## +     turtle_forward(20)
## +     turtle_goto(25, 0)
## +     turtle_turn_left(150)
## +     turtle_pen_up()
## +     turtle_goto(50, 0)
## +     turtle_pen_down()
## +     turtle_turn_left(30)
## +     turtle_forward(30)
## +     turtle_turn_left(60)
## +     turtle_forward(10)
## +     turtle_turn_left(60)
## +     turtle_forward(10)
## +     turtle_turn_left(30)
## +     turtle_forward(10)
## +     turtle_goto(45, 10)
## +     turtle_pen_up()
## +     turtle_goto(-45, 0)
## +     turtle_pen_down()
## +     turtle_forward(17)
## +     turtle_turn_right(170)
## +     turtle_forward(19)
## + }
##
## > draw_turtle()
```



If you are interested in exploring R games such as Mine sweeper, Five in a row and sliding puzzles in this package, you can check other demos using the code below:

```
demo(package = "fun")
```

Take Home Message

Thank you to everyone who takes time to read my post, does not give up and reach here. Here are some key points you don't want to miss at the end of the post:

- GitHub is an amazing place that one can share source code with others all around the world
- There are more interesting R packages to explore. If you have time, be sure to search on GitHub. It definitely worths the time.
- Good luck on your finals. Go Bears!

Reference

Research from

1. https://en.wikipedia.org/wiki/Cowsay
2. https://github.com/sckott/cowsay
3. https://cran.rstudio.com/web/packages/fun/fun.pdf
4. https://github.com/yihui/fun
5. https://cran.r-project.org/web/packages/cowsay/cowsay.pdf

Memes from

1. UCBMFET https://www.facebook.com/groups/1717731545171536/
2. http://xueba17.com/Upload/pstatp/2017-05-04/1dca0005dc9af3832309.jpg