Contents

1 R programming Week 4 2

1 R programming Week 4

The `grep()` function

What does the `grep()` function do when called with its default arguments?

It returns the indices for strings in a character that match a given regular expression.

`grep(value = TRUE)` returns a character vector containing the selected elements of x (after coercion, preserving names but no other attributes).

The `grepl()` function

`grepl` returns a logical vector (match or not for each element of x).

Regular Expressions

```
^s(.*?)r
```

Generic Functions

In the R system of classes and methods, what is a generic function?

S4 Methods

What function is used to obtain the function body for an S4 method function?

Not `getS3method()`.

What does the setOldClass function do?

Question 1

What is produced at the end of this snippet of `R` code?

```
set.seed(1)
rpois(5, 2)
```

(i) It is impossible to tell because the result is random

(ii) A vector with the numbers 1, 4, 1, 1, 5

(iii) A vector with the numbers 3.3, 2.5, 0.5, 1.1, 1.7

(iv) A vector with the numbers 1, 1, 2, 4, 1

Question 2

What `R` function can be used to generate standard Normal random variables?

(i) `dnorm`

(ii) `qnorm`

(iii) `rnorm`

(iv) `pnorm`

Standard Normal Distribution

Standardization Formula (used to compute Z-score)

$$z_o = \frac{x_o - \mu_x}{\sigma_x}$$

$$Z \sim \mathcal{N}(\mu = 1, \sigma^2 = 1^2)$$

```
?dnorm
```

Same help file for each command

Question 3

When simulating data, why is using the `set.seed()` function important?

(i) It ensures that the sequence of random numbers starts in a specific place and is therefore reproducible.

(ii) It can be used to generate non-uniform random numbers.

(iii) It ensures that the sequence of random numbers is truly random.

(iv) It ensures that the random numbers generated are within specified boundaries.

- Pseudo-random Numbers
- Mersenne Twister
- (Talk on this at Python Ireland 2014)

Question 4

Which function can be used to evaluate the **inverse cumulative distribution function** for the Poisson distribution?

(i) `rpois`

(ii) `ppois`

(iii) `qpois`

(iv) `dpois`

- Remark - another name for this function is the **quantile function**.

Question 5

What does the following code do?

```
set.seed(10)
n=10
x <- rbinom(n, 10, 0.5)
e <- rnorm(n, 0, 20)
y <- 0.5 + 2 * x + e
```

(i) Generate uniformly distributed random data

(ii) Generate data from a Normal linear model

(iii) Generate random exponentially distributed data

(iv) Generate data from a Poisson generalized linear model

```
set.seed(10)
n=10000
x <- rbinom(n, 10, 0.5)
e <- rnorm(n, 0, 20)
y <- 0.5 + 2 * x + e
hist(y)
```

- Normal Approximation of the Binomial Distribution

Question 6

What `R` function can be used to generate Binomial random variables?

(i) `dbinom`

(ii) `qbinom`

(iii) `rbinom`

(iv) `pbinom`

Question 7

What aspect of the `R` runtime does the profiler keep track of when an `R` expression is evaluated?

(i) the function call stack

(ii) the global environment

(iii) the working directory

(iv) the package search list

Question 8

Consider the following `R` code

```
library(datasets)
Rprof()
fit <- lm(y ~ x1 + x2)
Rprof(NULL)
```

(Assume that y, x1, and x2 are present in the workspace.)

Without running the code, what percentage of the run time is spent in the '`lm`' function, based on the '`by.total`' method of normalization shown in '`summaryRprof()`'?

(i) 100%

(ii) 23%

(iii) 50%

(iv) It is not possible to tell

Question 9

When using '`system.time()`', what is the user time?

(i) It is a measure of network latency

(ii) It is the time spent by the CPU waiting for other tasks to finish

(iii) It is the "wall-clock" time it takes to evaluate an expression

(iv) It is the time spent by the CPU evaluating an expression

Question 10

If a computer has more than one available processor and `R` is able to take advantage of that, then which of the following is true when using '`system.time()`'?

(i) elapsed time is 0

(ii) user time is always smaller than elapsed time

(iii) user time is 0

(iv) elapsed time may be smaller than user time