Chapter 4 - R Notation

Aodhagan

1/6/2020

load the data deck from website

```
rm(list = ls())

# partition the URL to make it more readable
url_remote <- "https://gist.githubusercontent.com/"
path_github <- "garrettgman/9629323/raw/ee5dfc039fd581cb467cc69c226ea2524913c3d8/"
filename <- "deck.csv"

# assign full URL to variable file_path
file_path <- paste0(url_remote, path_github, filename)

# read in the file from website
library(data.table)
deck <- fread(file_path, header = TRUE)
```

selecting values

you can extract the value or set of values from a data frame using indexing with [,]

```
deck[1, 1] # first row, first column
```

```
##    face
## 1: king
```

six methods of indexing

1. Positive integers

e.g.

```
deck[1, 2]
```

```
##      suit
## 1: spades
```

use vectors to select more than one item

```
deck[1, c(1, 2, 3)]
```

```
##    face   suit value
## 1: king spades    13
```

note that the methods used for indexing data frames can also be used for other R objects such as Vectors, matrices or arrays

note indexing in R begins at 1

note if two or more columns are selected in the index a data frame is returned. If only one column is selected a vector is returned.

```
deck[1:3, 2]
```

```
##      suit
## 1: spades
## 2: spades
## 3: spades
```

if you want to select one column and to have a data frame returned use the argument **drop=FALSE**

```
df <- deck[1:2, 1, drop=FALSE]
is.data.frame(df)
```

```
## [1] TRUE
```

2. negative integers

negative integers to the exact opposite of positive integers when indexing i.e. every element except the elements in the negative index will be returned

```
# only the first and second rows will be indexed
deck[-(3:52), 1:3]
```

```
##     face   suit value
## 1:  king spades    13
## 2: queen spades    12
```

negative integers are more efficient when you want a subset the majority of data frames rows or columns

3. zeroes

R will return nothing from a dimension when you use zero as an index

```
deck[0, 0] # data frame with zero columns and zero rows
```

```
## Null data.table (0 rows and 0 cols)
```

4. Blank spaces

blank space extracts every value in a dimension

```
deck[1, ] # extracts the entire first row of data frame
```

```
##    face   suit value
## 1: king spades    13
```

5. Logical values

you can index a R object using vectors of logical values

```
deck[1, c(TRUE, TRUE, FALSE)]
```

```
##    face   suit
## 1: king spades
```

every index where the vector states "TRUE" will return the corresponding index in the data frame

6. names

you can specify elements in a data frame by name as long as the name attributes exist

```
deck[1, c("face", "suit", "value")]
```

```
##    face   suit value
## 1: king spades    13
```

exercise 1

```
deal <- function(cards){
  cards[1, ]
}
deal(deck)
```

```
##    face   suit value
## 1: king spades    13
```

shuffle the deck

Begin by extracting every row in the data frame

```
deck2 <- deck[1:52, ]
head(deck2)
```

```
##     face   suit value
## 1:  king spades    13
## 2: queen spades    12
## 3:  jack spades    11
## 4:   ten spades    10
## 5:  nine spades     9
## 6: eight spades     8
```

note that above the order hasn't changed at all. We can get R to extract the rows in a different order

```
deck3 <- deck[c(2, 1, 3:52), ]
head(deck3)
```

```
##     face   suit value
## 1: queen spades    12
## 2:  king spades    13
## 3:  jack spades    11
## 4:   ten spades    10
## 5:  nine spades     9
## 6: eight spades     8
```

how do we apply the same principle to get a random order?use the randomising function sample()

```
random <- sample(1:52, size = 52)
deck4 <- deck[random, ]
head(deck4)
```

```
##     face     suit value
## 1:   ten    clubs    10
## 2:   ten   spades    10
```

```
## 3:   two    clubs     2
## 4: three   spades     3
## 5:   six   spades     6
## 6: three diamonds     3
```

exercise - create a shuffle function

```
shuffle <- function(cards){
  random <- sample(1:52, size = 52)
  cards[random, ]
}
```

now we can shuffle cards between each deal

```
deal(deck)
```

```
##    face   suit value
## 1: king spades    13
```

```
deck2 <- shuffle(deck)
deal(deck2)
```

```
##     face     suit value
## 1: queen diamonds    12
```

dollar signs and double brackets