Data Transformation

Table of Contents

# Using pipes

## Example of using intermediate objects

This is a way to proceed to several steps using intermediate objects. However, the code tends to be lengthy and the incrementation of each object has to be carefully done.

#Overwriting the data  
 foo\_foo <- as.data.frame(c(1,2,3))  
 foo\_foo\_1 <- cbind(foo\_foo,c(2,3,4))  
 foo\_foo\_2 <- cbind(foo\_foo\_1,c(3,4,5))  
 foo\_foo\_3 <- cbind(foo\_foo\_2,c(4,5,6))  
 colnames(foo\_foo\_3)=c("a","b","c","d")

## Overwriting the data

Another possibility is to overwrite the object at each step.

#Overwriting the data  
 foo\_foo <- data.frame(c(1,2,3))  
 foo\_foo <- cbind(foo\_foo,c(2,3,4))  
 foo\_foo <- cbind(foo\_foo,c(3,4,5))  
 foo\_foo <- cbind(foo\_foo,c(4,5,6))  
 colnames(foo\_foo)=c("a","b","c","d")

## Function composition

This method does not require any intermediate object but it may be difficult to read since it should be read from the last instruction.

#Method using intermediate objects  
 foo\_foo=as.data.frame(foo\_foo)  
 foo\_foo\_1=select(foo\_foo,b)  
 foo\_foo\_2=summarise(foo\_foo\_1,mean=mean(b))  
   
#Method using function composition (encapsulation)  
 summarise(select(as.data.frame(foo\_foo),b),mean=mean(b))

## mean  
## 1 3

## Using pipes

Pipe are called using the symbol %>% (shortcut: Ctrl+Shift+M) and create an intermediate object "." which is discarded at last.

#Using pipes  
 foo\_foo %>%   
 as.data.frame() %>%   
 select(b) %>%  
 summarise(mean=mean(b))

## mean  
## 1 3

#How do pipe work?  
mypipe=function(.){  
 .=as.data.frame(.)  
 .=select(.,b)  
 summarise(.,mean=mean(b))  
   
}  
  
mypipe(foo\_foo)

## mean  
## 1 3

## Caution when using pipe

* Pipes do not work with assign() and environment() functions.
* Better to limit the number of operations
* Not adapted to multiple inputs

# Other operators than pipes

## The %$% operator

Useful for splitting variables from a dataframe

#Splitting variables from a dataframe  
mtcars %$%  
 cor(disp, mpg)

## [1] -0.8475514

## The %<>% operator

Useful for storing the output into the original dataframe

mtcars %<>% transform(cyl = cyl \* 2)