Manipulate data

Dr Wayne Stewart

Latest

Contents

Introduction

There are different ways to wrangle data. We need to do this becasue we want usable summaries that answer the questions we have.

Please see `Into2R` R package for more details.

Base R

Base R refers to the standard base package that we all obtain when R is first downloaded and installed. If you want to see the functions contained in this package use `help(package = "base")`.

We will learn how to manipulate a data frame using base R functions first and then after we have gained a basic facility we will start to use specialist packages like `dplyr` and `data.table`.

Reading in data

Most times we will read in a `csv` file. To do this we will use `read.csv()`.

If you have the `csv` file in your working directory and you know its name then go ahead and read it into R's workspace:

```
ddt <- read.csv("DDT.csv",
               header = TRUE)
```

In the case of the data set `DDT.csv`, this is already available in the `Inro2R` package.

You can make it available by invoking the following:

```
library(Intro2R)
data(ddt)
head(ddt)
```

```
##   RIVER MILE  SPECIES LENGTH WEIGHT DDT
## 1   FCM    5 CCATFISH   42.5    732  10
## 2   FCM    5 CCATFISH   44.0    795  16
## 3   FCM    5 CCATFISH   41.5    547  23
## 4   FCM    5 CCATFISH   39.0    465  21
## 5   FCM    5 CCATFISH   50.5   1252  50
## 6   FCM    5 CCATFISH   52.0   1255 150
```

Now that you have the data in the workspace

Lets start to wrangle the data. This data is in `standard form` – that is each row records measurements on an experimental unit. In this case each row carries a multivariate measurement of a fish. The first row gives:

```
head(ddt,1)
```

```
##   RIVER MILE  SPECIES LENGTH WEIGHT DDT
## 1   FCM    5 CCATFISH   42.5    732  10
```

That is the first fish was caught on the FCM river, at the 5 mile mark, its species was recorded as catfish and length 42.5 cm, weight 732 gms and has 10 ppm DDT in its flesh.

Subsetting

The following questions will be answered using indexing/subsetting methods [, [[and $.

If you want more information on these basic functions then by all means look at the documentation and further more advanced procedures:

- ?`[`
- https://adv-r.hadley.nz/subsetting.html

Please note that subsetting operators function differently according to the type of vector (e.g. lists, matrices, and data frames).

For now we will concentrate on data frames since this object type will be commonly the way that data will appear in R. However, understanding other applications of the operators/methods/functions on different vectors will help in comprehending what [does.

Preserve dimensionality using drop

When manipulating data frames and matrices it will often be important to maintain the same dimensionality.

ddt[,"WEIGHT"] will cease to be a 2 dimensional object. But will instead be a vector.

```
wt_vector <- ddt[,"WEIGHT"]
head(wt_vector)
```

```
## [1]  732  795  547  465 1252 1255
```

```
dim(wt_vector)
```

```
## NULL
```

```
wt_df <- ddt[,"WEIGHT", drop = FALSE]
head(wt_df)
```

```
##   WEIGHT
## 1    732
## 2    795
## 3    547
## 4    465
## 5   1252
## 6   1255
```

```
dim(wt_df)
```

```
## [1] 144   1
```

Investigate [and [[

These have some interesting effects on lists, [will subset to a list, [[will give the component in the list and take the class of the component object.

In the case of a data frame the single square bracket gives a data frame while a double [[gives the atomic vector. See below

```
v <- ddt[3]
head(v)
```

```
##    SPECIES
## 1 CCATFISH
## 2 CCATFISH
## 3 CCATFISH
## 4 CCATFISH
## 5 CCATFISH
## 6 CCATFISH
```

```
ddt[[3]]
```

```
##   [1] CCATFISH  CCATFISH  CCATFISH  CCATFISH  CCATFISH  CCATFISH  CCATFISH
##   [8] CCATFISH  CCATFISH  CCATFISH  CCATFISH  CCATFISH  CCATFISH  CCATFISH
##  [15] CCATFISH  CCATFISH  CCATFISH  CCATFISH  CCATFISH  CCATFISH  CCATFISH
##  [22] CCATFISH  CCATFISH  CCATFISH  CCATFISH  CCATFISH  CCATFISH  CCATFISH
##  [29] CCATFISH  CCATFISH  SMBUFFALO SMBUFFALO SMBUFFALO SMBUFFALO SMBUFFALO
##  [36] SMBUFFALO CCATFISH  CCATFISH  CCATFISH  CCATFISH  CCATFISH  CCATFISH
##  [43] LMBASS    LMBASS    LMBASS    LMBASS    LMBASS    LMBASS    CCATFISH
##  [50] CCATFISH  CCATFISH  CCATFISH  CCATFISH  CCATFISH  SMBUFFALO SMBUFFALO
##  [57] SMBUFFALO SMBUFFALO SMBUFFALO SMBUFFALO CCATFISH  CCATFISH  CCATFISH
##  [64] CCATFISH  CCATFISH  CCATFISH  CCATFISH  CCATFISH  CCATFISH  CCATFISH
##  [71] CCATFISH  CCATFISH  SMBUFFALO SMBUFFALO SMBUFFALO SMBUFFALO SMBUFFALO
##  [78] SMBUFFALO CCATFISH  CCATFISH  CCATFISH  CCATFISH  CCATFISH  CCATFISH
##  [85] CCATFISH  CCATFISH  CCATFISH  CCATFISH  CCATFISH  CCATFISH  SMBUFFALO
##  [92] SMBUFFALO SMBUFFALO SMBUFFALO SMBUFFALO SMBUFFALO CCATFISH  CCATFISH
##  [99] CCATFISH  CCATFISH  CCATFISH  CCATFISH  CCATFISH  CCATFISH  CCATFISH
## [106] CCATFISH  CCATFISH  CCATFISH  SMBUFFALO SMBUFFALO SMBUFFALO SMBUFFALO
## [113] SMBUFFALO SMBUFFALO CCATFISH  CCATFISH  CCATFISH  CCATFISH  CCATFISH
## [120] CCATFISH  CCATFISH  CCATFISH  CCATFISH  CCATFISH  CCATFISH  CCATFISH
## [127] SMBUFFALO SMBUFFALO CCATFISH  CCATFISH  CCATFISH  CCATFISH  CCATFISH
## [134] CCATFISH  SMBUFFALO SMBUFFALO SMBUFFALO SMBUFFALO LMBASS    LMBASS
## [141] LMBASS    LMBASS    LMBASS    LMBASS
## Levels: CCATFISH LMBASS SMBUFFALO
```

Questions

We will now apply our knowledge of subsetting and assignment to answer some questions.

Find the mean weight of all the fish in the sample

```
mean(ddt$WEIGHT)
```

```
## [1] 1049.715
```

Or

```
with(ddt, mean(WEIGHT))
```

```
## [1] 1049.715
```

Find the mean weight of catfish

```
mean(ddt[ddt$SPECIES=="CCATFISH","WEIGHT"])
```

```
## [1] 987.2917
```

Or

```
cw <- subset(ddt, SPECIES == "CCATFISH", "WEIGHT")
mean(cw[,1])
```

```
## [1] 987.2917
```

Find the number of fish over 1000 gms

```
tab <- table(ddt[ddt$WEIGHT > 1000, "SPECIES"])
sum(tab)
```

```
## [1] 72
```

Or

```
dim(ddt[ddt$WEIGHT > 1000,])[1]
```

```
## [1] 72
```

Use `dplyr`

The `dplyr` package is very useful for wrangling data. There are essentially 5 verbs that you should master to help in this process.

- `arrange`
- `select`
- `filter`

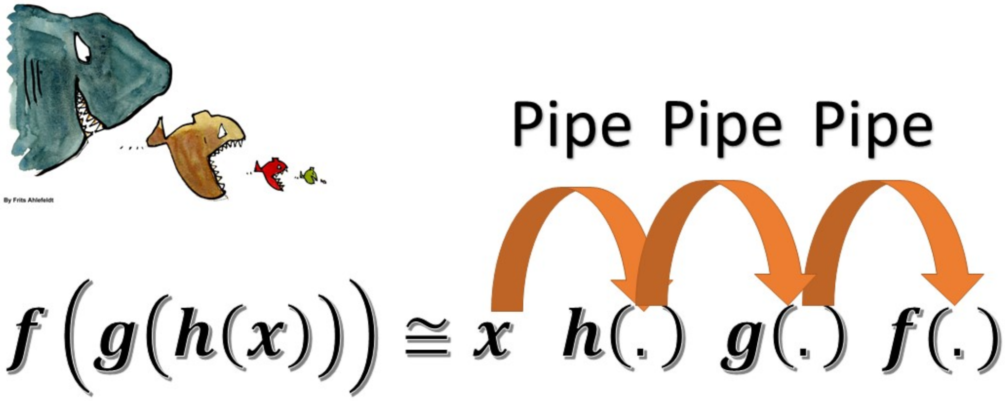
- `mutate`
- `summarize`

For a good online introduction to these verbs go https://teachingr.com/content/the-5-verbs-of-dplyr/the-5-verbs-of-dplyr-solutions.html.

The package relies on a different approach to programming using behind the scenes meta-programming and `%>%` piping.

One difference is the untangling of composite functions:

$$f(g(h(x))) = x \ \% > \% \ h(.) \ \% > \% \ g(.) \ \% > \% \ f(.)$$



Example of piping

x in this case is the data frame `ddt` we will pipe this to the function `filter` – this will filter the rows according to the conditionals placed in the arguments - all rows where the SPECIES is SMBUFFALO and WEIGHT is bigger than 2000, the data frame that is made is then piped to `arrange` where it is sorted by the LENGTH variable (small to large) which is then piped to the function `select` which keeps only the columns SPECIES, LENGTH and DDT.

See below:

```
library(dplyr ,warn.conflicts = FALSE)
ddt %>% filter(SPECIES == "SMBUFFALO" & WEIGHT > 2000) %>% arrange(LENGTH) %>% select(SPECIES, LENGTH,
```

```
##     SPECIES LENGTH DDT
## 1 SMBUFFALO   48.0 6.8
## 2 SMBUFFALO   48.5 2.8
## 3 SMBUFFALO   52.0 3.0
```

Of course we could further process the output by piping into other functions.

Questions

Find the mean weight of all the fish in the sample

```r
#mean(ddt$WEIGHT)
# using dplyr package
ddt  %>% summarize(mean = mean(WEIGHT))
```

```
##       mean
## 1 1049.715
```

Find the mean weight of catfish

```r
#mean(ddt[ddt$SPECIES=="CCATFISH","WEIGHT"])

ddt %>% filter( SPECIES == "CCATFISH") %>% summarize(mean = mean(WEIGHT))
```

```
##       mean
## 1 987.2917
```

Find the number of fish over 1000 gms

```r
#tab <- table(ddt[ddt$WEIGHT > 1000, "SPECIES"])
#sum(tab)
# using dplyr

ddt %>% filter( WEIGHT > 1000) %>% summarize(n = n())
```

```
##    n
## 1 72
```

Use `data.table` package

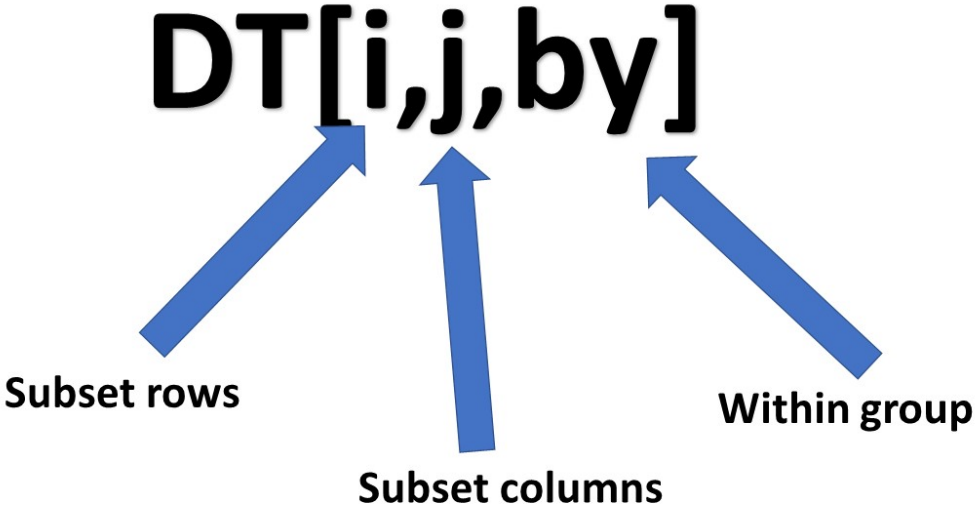
The dplyr package has some intersting verbs and is very nicely useable with the `%>%` pipe function.

When the data is large the system can slow or even run out of memory. When this happens processing stops and the application will “fall over”.

The `data.table` is fast and will use more than one thread of the cpu depending on the number available on your machine. It is very effiient and will only do the computations it “has to”. In some cases it will not make a copy of the data but use “reference” semantics or “copy in place”.

It has become an important packge in R and I want you to be conversent with the basics of it so that in the future you will easily perfect your understanding.

This is the basic syntax:



You can learn more by consulting:

- help(package = "data.table")
- https://www.listendata.com/2016/10/r-data-table.html

Lets dive in.

First we will need to convert the `ddt` data frame into a data table.

```
library(data.table, warn.conflicts = FALSE)
dt <- as.data.table(ddt)
class(dt)
```

```
## [1] "data.table" "data.frame"
```

We will now make some subsets:

All fish that are SMBUFFALO and have a WEIGHT larger than 2000,

```
dt[SPECIES == "SMBUFFALO" & WEIGHT > 2000,]
```

```
##    RIVER MILE   SPECIES LENGTH WEIGHT DDT
## 1:   TRM  280 SMBUFFALO   52.0   2302 3.0
## 2:   TRM  280 SMBUFFALO   48.0   2006 6.8
## 3:   TRM  310 SMBUFFALO   48.5   2061 2.8
```

Same as above BUT we only want the LENGTH variable:

```
dt[SPECIES == "SMBUFFALO" & WEIGHT > 2000, LENGTH]
```

```
## [1] 52.0 48.0 48.5
```

Suppose we need to summarize measures for each species.

```
dt[, list(mean_weight = mean(WEIGHT)), by = SPECIES]
```

```
##      SPECIES mean_weight
## 1:  CCATFISH    987.2917
## 2: SMBUFFALO   1356.4167
## 3:    LMBASS    629.0000
```

The `data.table` package uses helper notation, we can write the previous by using `.` instead of `list`:

```
dt[, .(mean_weight = mean(WEIGHT)), by = SPECIES]
```

```
##      SPECIES mean_weight
## 1:  CCATFISH    987.2917
## 2: SMBUFFALO   1356.4167
## 3:    LMBASS    629.0000
```

Questions

Find the mean weight of all the fish in the sample

```
#mean(ddt$WEIGHT)
# using dplyr package
#ddt  %>% summarize(mean = mean(WEIGHT))

dt[,.(mean = mean(WEIGHT))]
```

```
##        mean
## 1: 1049.715
```

Find the mean weight of catfish

```
#mean(ddt[ddt$SPECIES=="CCATFISH","WEIGHT"])

#ddt %>% filter( SPECIES == "CCATFISH") %>% summarize(mean = mean(WEIGHT))

dt[SPECIES == "CCATFISH", .(mean = mean(WEIGHT))]
```

```
##        mean
## 1: 987.2917
```

Find the number of fish over 1000 gms

```
#tab <- table(ddt[ddt$WEIGHT > 1000, "SPECIES"])
#sum(tab)
# using dplyr

#ddt %>% filter( WEIGHT > 1000) %>% summarize(n = n())

dt[WEIGHT > 1000, .N]
```

```
## [1] 72
```