Dplyr Bible

Kevin da Silva Castanheira

8/21/2018

## Packages 101

dplyr and tidyr are R libraries which fanicilitate data manipulation.

If you do not have these libraries installed, you can use the function install.packages to install them.

install.packages("dplyr")  
install.packages("tidyr")

If you don’t know if you have these packages installed, you can check by using the function installed.packages(), which returns a matrix of all the packages oyu have installed, and indexing to the row of the package you want.

head(installed.packages(), 2)

## Package   
## abind "abind"   
## acepack "acepack"  
## LibPath   
## abind "/Library/Frameworks/R.framework/Versions/3.5/Resources/library"  
## acepack "/Library/Frameworks/R.framework/Versions/3.5/Resources/library"  
## Version Priority Depends Imports LinkingTo  
## abind "1.4-5" NA "R (>= 1.5.0)" "methods, utils" NA   
## acepack "1.4.1" NA NA NA NA   
## Suggests Enhances License License\_is\_FOSS  
## abind NA NA "LGPL (>= 2)" NA   
## acepack "testthat" NA "MIT + file LICENSE" NA   
## License\_restricts\_use OS\_type MD5sum NeedsCompilation Built   
## abind NA NA NA "no" "3.5.0"  
## acepack NA NA NA "yes" "3.5.0"

If it returns something you have that package.

installed.packages()["dplyr",]

## Package   
## "dplyr"   
## LibPath   
## "/Library/Frameworks/R.framework/Versions/3.5/Resources/library"   
## Version   
## "0.7.6"   
## Priority   
## NA   
## Depends   
## "R (>= 3.1.2)"   
## Imports   
## "assertthat (>= 0.2.0), bindrcpp (>= 0.2.0.9000), glue (>=\n1.1.1), magrittr (>= 1.5), methods, pkgconfig (>= 2.0.1), R6\n(>= 2.2.2), Rcpp (>= 0.12.15), rlang (>= 0.2.0), tibble (>=\n1.3.1), tidyselect (>= 0.2.3), utils"   
## LinkingTo   
## "BH (>= 1.58.0-1), bindrcpp (>= 0.2.0.9000), plogr (>=\n0.1.10), Rcpp (>= 0.12.15)"   
## Suggests   
## "bit64 (>= 0.9.7), callr, covr (>= 3.0.1), DBI (>= 0.7.14),\ndbplyr (>= 1.2.0), dtplyr (>= 0.0.2), ggplot2 (>= 2.2.1), hms\n(>= 0.4.1), knitr (>= 1.19), Lahman (>= 3.0-1), lubridate,\nMASS, mgcv (>= 1.8.23), microbenchmark (>= 1.4.4), nycflights13\n(>= 0.2.2), rmarkdown (>= 1.8), RMySQL (>= 0.10.13),\nRPostgreSQL (>= 0.6.2), RSQLite (>= 2.0), testthat (>= 2.0.0),\nwithr (>= 2.1.1)"   
## Enhances   
## NA   
## License   
## "MIT + file LICENSE"   
## License\_is\_FOSS   
## NA   
## License\_restricts\_use   
## NA   
## OS\_type   
## NA   
## MD5sum   
## NA   
## NeedsCompilation   
## "yes"   
## Built   
## "3.5.1"

You can load the library by using the function library.

library(tidyr)  
library(dplyr)

## Warning: package 'dplyr' was built under R version 3.5.1

##   
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':  
##   
## filter, lag

## The following objects are masked from 'package:base':  
##   
## intersect, setdiff, setequal, union

And you can see the loaded packages using the sessionInfo() function. The packages loaded will be under “other attached packages”.

sessionInfo()

## R version 3.5.0 (2018-04-23)  
## Platform: x86\_64-apple-darwin15.6.0 (64-bit)  
## Running under: OS X El Capitan 10.11.6  
##   
## Matrix products: default  
## BLAS: /Library/Frameworks/R.framework/Versions/3.5/Resources/lib/libRblas.0.dylib  
## LAPACK: /Library/Frameworks/R.framework/Versions/3.5/Resources/lib/libRlapack.dylib  
##   
## locale:  
## [1] en\_CA.UTF-8/en\_CA.UTF-8/en\_CA.UTF-8/C/en\_CA.UTF-8/en\_CA.UTF-8  
##   
## attached base packages:  
## [1] stats graphics grDevices utils datasets methods base   
##   
## other attached packages:  
## [1] dplyr\_0.7.6 tidyr\_0.8.1  
##   
## loaded via a namespace (and not attached):  
## [1] Rcpp\_0.12.16 knitr\_1.20 bindr\_0.1.1 magrittr\_1.5   
## [5] tidyselect\_0.2.4 R6\_2.2.2 rlang\_0.2.0 stringr\_1.3.1   
## [9] tools\_3.5.0 htmltools\_0.3.6 yaml\_2.1.19 rprojroot\_1.3-2   
## [13] digest\_0.6.15 assertthat\_0.2.0 tibble\_1.4.2 bindrcpp\_0.2.2   
## [17] purrr\_0.2.4 glue\_1.2.0 evaluate\_0.10.1 rmarkdown\_1.9   
## [21] stringi\_1.2.2 compiler\_3.5.0 pillar\_1.2.2 backports\_1.1.2   
## [25] pkgconfig\_2.0.1

Alternatively, if you dont want to load the entire library you can always simply use a function from that library provided you know which library it comes from.

Simply put the name of the library proceeded by “::” and the name of the fucntion. For example: dplyr::summarize()

## DPLYR and TIDYR

Dplyr and tidyr are two pacages availbe in R which are very powerful tools when it comes to data manipulation.

The data I will be using as an example was scraped online. It contains information on a bunch of publication in psychology and neuroscience.

Let’s take a look at this data frame. It has 18 columns and 47,425 rows. it is in long format (each row is a publication).

summary(df)

## AU AF TI   
## Length:200 Length:200 Length:200   
## Class :character Class :character Class :character   
## Mode :character Mode :character Mode :character   
##   
##   
##   
## SO LA DE   
## Length:200 Length:200 Length:200   
## Class :character Class :character Class :character   
## Mode :character Mode :character Mode :character   
##   
##   
##   
## ID Z9 U1 U2   
## Length:200 Min. : 1.00 Min. : 0.000 Min. : 0.0   
## Class :character 1st Qu.: 5.00 1st Qu.: 0.000 1st Qu.: 9.0   
## Mode :character Median : 9.00 Median : 1.000 Median : 15.5   
## Mean :12.74 Mean : 1.045 Mean : 18.8   
## 3rd Qu.:15.00 3rd Qu.: 2.000 3rd Qu.: 23.0   
## Max. :92.00 Max. :11.000 Max. :134.0   
## PU PY DI WC   
## Length:200 Min. :2013 Length:200 Length:200   
## Class :character 1st Qu.:2013 Class :character Class :character   
## Mode :character Median :2013 Mode :character Mode :character   
## Mean :2013   
## 3rd Qu.:2013   
## Max. :2013   
## ANIMAL FMRI\_AB JOUR IF   
## Mode :logical Mode :logical Length:200 Min. :1.964   
## FALSE:200 FALSE:198 Class :character 1st Qu.:2.235   
## TRUE :2 Mode :character Median :3.425   
## Mean :3.312   
## 3rd Qu.:4.035   
## Max. :4.974

### PIPIING

The %>% symbol is used with dplyr to “pipe” ourput of one function to another. It’s like a relay race where the function on the left side of the %>% passes its output to the function on the right. Usually, the first element on the left is a data frame you’d like to manipulate.

df %>% head(2)

## AU  
## 1 Jarso, S; Li, MW; Faria, A; Davis, C; Leigh, R; Sebastian, R; Tsapkini, K; Mori, S; Hillis, AE  
## 2 Gordon, JK; Cheimariou, S  
## AF  
## 1 Jarso, Samson; Li, Muwei; Faria, Andreia; Davis, Cameron; Leigh, Richard; Sebastian, Rajani; Tsapkini, Kyrana; Mori, Susumu; Hillis, Argye E.  
## 2 Gordon, Jean K.; Cheimariou, Spyridoula  
## TI  
## 1 Distinct mechanisms and timing of language recovery after stroke  
## 2 Semantic interference in a randomized naming task: Effects of age, order, and category  
## SO LA  
## 1 COGNITIVE NEUROPSYCHOLOGY English  
## 2 COGNITIVE NEUROPSYCHOLOGY English  
## DE  
## 1   
## 2 Aphasia; Functional magnetic resonance imaging; Stroke; Recovery; Language  
## ID  
## 1   
## 2 POSITRON-EMISSION-TOMOGRAPHY; TRANSCRANIAL MAGNETIC STIMULATION; POSTSTROKE APHASIA; CORTICAL HYPOPERFUSION; HEMODYNAMIC-RESPONSE; SUBACUTE STROKE; PET ACTIVATION; RECOVERY; LANGUAGE; BRAIN  
## Z9 U1 U2 PU PY  
## 1 20 0 14 ROUTLEDGE JOURNALS, TAYLOR & FRANCIS LTD 2013  
## 2 2 0 18 ROUTLEDGE JOURNALS, TAYLOR & FRANCIS LTD 2013  
## DI WC ANIMAL  
## 1 10.1080/02643294.2014.881334 Psychology; Psychology, Experimental FALSE  
## 2 10.1080/02643294.2013.875467 Psychology; Psychology, Experimental FALSE  
## FMRI\_AB JOUR IF  
## 1 FALSE COGNITIVE\_NEUROPSYC 1.964  
## 2 FALSE COGNITIVE\_NEUROPSYC 1.964

You can chain pipe statements so that the output continuously gets passed to another function. Here I’m selecint the first 20 rows and passing only those rows to the function summary.

df %>% head(20) %>% summary()

## AU AF TI   
## Length:20 Length:20 Length:20   
## Class :character Class :character Class :character   
## Mode :character Mode :character Mode :character   
##   
##   
##   
## SO LA DE   
## Length:20 Length:20 Length:20   
## Class :character Class :character Class :character   
## Mode :character Mode :character Mode :character   
##   
##   
##   
## ID Z9 U1 U2   
## Length:20 Min. : 1.00 Min. :0 Min. : 4.00   
## Class :character 1st Qu.: 2.00 1st Qu.:0 1st Qu.:13.75   
## Mode :character Median : 6.50 Median :0 Median :17.00   
## Mean : 7.95 Mean :1 Mean :18.70   
## 3rd Qu.:11.25 3rd Qu.:1 3rd Qu.:26.25   
## Max. :22.00 Max. :5 Max. :37.00   
## PU PY DI WC   
## Length:20 Min. :2013 Length:20 Length:20   
## Class :character 1st Qu.:2013 Class :character Class :character   
## Mode :character Median :2013 Mode :character Mode :character   
## Mean :2013   
## 3rd Qu.:2013   
## Max. :2013   
## ANIMAL FMRI\_AB JOUR IF   
## Mode :logical Mode :logical Length:20 Min. :1.964   
## FALSE:20 FALSE:20 Class :character 1st Qu.:1.964   
## Mode :character Median :3.371   
## Mean :2.808   
## 3rd Qu.:3.371   
## Max. :3.371

### SELECTING COLUMNS

This data frame has too many columns, a lot of which we do not need. To filter out the unnecessary columns we can use the function select. Just provide the fucntion with a vector of the names of the columns you want to remove.

REMEMBER TO ALWAYS ASSIGN THE OUTPUT TO A VARIABLE IF YOU WANT TO KEEP IT!

#The following two statements are equivalent  
df = select(df, c("AF", "TI", "LA", "Z9", "U1", "U2", "PY", "WC", "ANIMAL", "FMRI\_AB", "DI"))  
  
df = df %>% select(c("AF", "TI", "LA", "Z9", "U1", "U2", "PY", "WC", "ANIMAL", "FMRI\_AB", "DI"))  
  
# same as basic R command  
df = df[,c("AF", "TI", "Z9", "U1", "U2", "PY", "WC", "ANIMAL", "FMRI\_AB", "DI", "LA")]

We can also use a shorthand since the columns we want to keep are all one after the other, we can specify that we want ALL the columns between AF and LA.

df = df %>% select(AF:LA)

#### SELECTING ROWS

You can also filter out rows using the filter function. You need to supply the filter function with a logical statement (a true or false statement).

####Logical Statements Here are some examples of logical statements. Note that the ! sign means NOT.

3 > 2

## [1] TRUE

!(3>2)

## [1] FALSE

"help"=="Help"

## [1] FALSE

4 <=6

## [1] TRUE

Here we’re telling the function filter we do not want any rows which have no data for the column Z9. This is done by using the function is.na() which returns a TRUE if the supplied value is empty or FALSE if it contains an element. Since we want to KEEP the rows with data we would use !is.na()

df = df %>% filter(!is.na(Z9))

You can combine logical statements using the & operator. This will evaluate to TRUE if and only if BOTH values are TRUE. Here we’re selecting the rows which are both published after 1999 and with greater than 0 citations.

#### AND ###  
df = df %>% filter( PY > 1999 & Z9 > 0)

Yuu can use the OR operator | which will evaluate to TRUE if one of the inputs is TRUE.

### OR ###  
df = df %>% filter(LA == "English" | LA == "Portuguese")

If you want to combine sveral statements you have to use parentheses.

df = df %>% filter( (PY > 1999 & PY < 2018 ) & (LA == "English" | LA == "Portuguese"))

Instead of using a lot of | operators you can also use the %in% operator. This will check is a value is contained within a vector. Here, instead of checking if the Language is one of four values (using four OR operators) we check if Language is in a vactor of languages I want to keep in my data set.

df = df %>% filter(LA %in% c("English", "Portuguese", "French", "German"))

You can also remove duplicate rows by using the function distinct. This function will only keep one entry per value of a column. Here we’re using the DOIs to remove any duplicates. The argument “.keep\_all” tells this function to keep all the other columns in th data frame.

df = df %>% distinct(DI, .keep\_all = TRUE)

### Group\_by & Summarize

One fo the most useful tools in dplyr is the ability to group and summarize your data.

Dplyr lets you group your data by using the function group\_by; this function will make groups based on a categorical variable. The function summarize will allow you to apply fucntions to your groups.

Here we are grouping by the logical variable FMRI\_AB which denotes a publication which either does or does not use fMRI. We are also calculating a mean and sd by group.

df %>% group\_by(FMRI\_AB) %>% summarize(mean = mean(Z9), sd=sd(Z9))

## # A tibble: 2 x 3  
## FMRI\_AB mean sd  
## <lgl> <dbl> <dbl>  
## 1 FALSE 12.8 13.2   
## 2 TRUE 10.5 0.707

We can also use more than one grouping variable. Here we’re splitting the data by year and by whether or not the publication used fMRI.

df %>% filter(!FMRI\_AB) %>% group\_by(PY,FMRI\_AB) %>% summarize(n = length(Z9))

## # A tibble: 1 x 3  
## # Groups: PY [?]  
## PY FMRI\_AB n  
## <int> <lgl> <int>  
## 1 2013 FALSE 198

###Mutate Dplyr also allows you to make new columns out of old columns by using the function mutate.

df = df %>% mutate(Years\_Since\_Pub = 2018 - PY)  
  
df %>% select(Years\_Since\_Pub) %>% head(10)

## Years\_Since\_Pub  
## 1 5  
## 2 5  
## 3 5  
## 4 5  
## 5 5  
## 6 5  
## 7 5  
## 8 5  
## 9 5  
## 10 5

###Arrange

You can alos use dplyr to sort the data frame by a certain column in either ascending (default) or descending order.

df %>% arrange(PY) %>% select(PY) %>% head(10)

## PY  
## 1 2013  
## 2 2013  
## 3 2013  
## 4 2013  
## 5 2013  
## 6 2013  
## 7 2013  
## 8 2013  
## 9 2013  
## 10 2013

df %>% arrange(desc(PY)) %>% select(PY) %>% head(10)

## PY  
## 1 2013  
## 2 2013  
## 3 2013  
## 4 2013  
## 5 2013  
## 6 2013  
## 7 2013  
## 8 2013  
## 9 2013  
## 10 2013

df = df %>% arrange(desc(PY))

###Long and Wide format

You can use TIDYR to transform your data from long to wide format. Generally speaking, you want your data in long format for analyses and for graphing. However, a lot of the time data is collected in wide format.

Here we have the impact factor for a bunch of journals, over several years. Each row represents a year and each column represents a journal.

impactFactor\_wide %>% head(2)

## Year ACTA\_PSYC ADDICTIVE\_BEHAV AMER\_JOUR\_OF\_DRUG\_AND\_ALC\_ABUSE  
## 1 2016 2.031 2.944 2.161  
## 2 2015 1.816 2.795 1.828  
## AUTISM\_RESEARCH BEHAVIOR\_RESEARCH\_METHODS BILINGUALISM\_LANG\_AND\_COG  
## 1 3.765 3.623 3.01  
## 2 3.048 3.048 2.33  
## BIOLOGICAL\_PSYCHOLOGY BRAIN\_AND\_COG BRAIN\_AND\_LANG COGNITION  
## 1 3.070 2.432 2.439 3.414  
## 2 3.234 2.399 3.038 3.411  
## COGNITIVE\_NEUROPSYC CONSCIOUSNESS\_AND\_COGNITION  
## 1 1.146 2.144  
## 2 1.444 2.182  
## CURRENT\_DIRECTIONS\_IN\_PSYCHOLOGICAL\_SCIENCE DEPRESSION\_AND\_ANXIETY  
## 1 5.255 4.971  
## 2 5.545 5.004  
## DEVELOPMENTAL\_NEUROPSYCHOLOGY DEVELOPMENTAL\_SCIENCE EMOTION  
## 1 1.813 4.604 3.251  
## 2 1.947 3.982 3.082  
## FRONTIERS\_IN\_HUMAN\_NEUROSCIENCE FRONTIERS\_IN\_PSYC  
## 1 3.209 2.321  
## 2 3.634 2.463  
## INTERNATIONAL\_JOURNAL\_OF\_PSYCHOPHYSIOLOGY JOURNAL\_OF\_ABNORMAL\_PSYCHOLOGY  
## 1 2.582 4.133  
## 2 2.596 5.538  
## JOURNAL\_OF\_AUTISM\_AND\_DEVELOPMENTAL\_DISORDERS  
## 1 3.321  
## 2 3.493  
## JOURNAL\_OF\_CHILD\_PSYCHOLOGY\_AND\_PSYCHIATRY  
## 1 6.226  
## 2 6.615  
## JOURNAL\_OF\_CLINICAL\_AND\_EXPERIMENTAL\_NEUROPSYCHOLOGY  
## 1 1.839  
## 2 1.693  
## JOURNAL\_OF\_COGNITIVE\_NEUROSCIENCE  
## 1 3.108  
## 2 3.559  
## JOURNAL\_OF\_EXPERIMENTAL\_PSYCHOLOGY\_GENERAL  
## 1 4.42  
## 2 4.07  
## JOURNAL\_OF\_EXPERIMENTAL\_PSYCHOLOGY\_LEARNING\_MEMORY\_AND\_COGNITION  
## 1 2.667  
## 2 2.776  
## JOURNAL\_OF\_NEUROLINGUISTICS JOURNAL\_OF\_PSYCHOPHYSIOLOGY  
## 1 1.403 0.683  
## 2 1.296 1.167  
## JOURNAL\_OF\_THE\_AMERICAN\_ACADEMY\_OF\_CHILD\_AND\_ADOLESCENT\_PSYCHIATRY  
## 1 6.442  
## 2 7.182  
## JOURNAL\_OF\_THE\_INTERNATIONAL\_NEUROPSYCHOLOGICAL\_SOCIETY LATERALITY  
## 1 2.181 1.267  
## 2 2.633 1.239  
## LEARNING\_AND\_MEMORY NEUROBIOLOGY\_OF\_LEARNING\_AND\_MEMORY NEUROCASE  
## 1 2.894 3.543 0.988  
## 2 2.906 3.439 1.225  
## NEUROPSYCHOBIOLOGY NEUROPSYCHOLOGIA NEUROPSYCHOLOGY PERCEPTION  
## 1 1.491 3.197 3.286 1.087  
## 2 1.763 2.989 2.879 0.917  
## PERSPECTIVES\_ON\_PSYCHOLOGICAL\_SCIENCE PHYSIOLOGY\_AND\_BEHAVIOR PSYC\_SCI  
## 1 7.359 2.341 5.667  
## 2 7.658 2.461 5.476  
## PSYCHOLOGICAL\_MEDICINE PSYCHOLOGICAL\_RESEARCH\_PSYCHOLOGISCHE\_FORSCHUNG  
## 1 5.230 3.119  
## 2 5.491 2.681  
## PSYCHOPHYSIOLOGY PSYCHOSOMATIC\_MEDICINE  
## 1 2.668 3.863  
## 2 3.074 3.638  
## QUARTERLY\_JOURNAL\_OF\_EXPERIMENTAL\_PSYCHOLOGY  
## 1 2.129  
## 2 2.130  
## SCANDINAVIAN\_JOURNAL\_OF\_PSYCHOLOGY SOCIAL\_COG\_AND\_AFF\_NEURO SOCIAL\_NEURO  
## 1 1.284 3.937 2.255  
## 2 1.364 5.101 2.778

What we want is for each row to represent a year and a journal (i.e. a specific data point). To do this we can use the function gather which take as arguments the data frame you want to transform, the name of the column you want to store the old column names, the name of the column you want to store the values, and the columns to transform.

impactFactor\_long = gather(impactFactor\_wide, journal, impactFactor, ACTA\_PSYC:SOCIAL\_NEURO,  
 factor\_key=FALSE)

You can also go from long to wide by using the function spread. this one takes the data frame you want to transform, the column with the new column names and the column with the values.

# going from long to wide  
impactFactor\_wide2 = spread(impactFactor\_long, journal, impactFactor)

###Advanced Column Selection

You can select columns a variety of ways. Here are a few examples: Columns which contain something

#Looks for BIO in the Column  
impactFactor\_wide %>% select(contains("BIO")) %>% head(1)

## BIOLOGICAL\_PSYCHOLOGY NEUROBIOLOGY\_OF\_LEARNING\_AND\_MEMORY  
## 1 3.07 3.543  
## NEUROPSYCHOBIOLOGY  
## 1 1.491

Columns which end with something

#Column name ends with PSYCHOLOGY  
impactFactor\_wide %>% select(ends\_with("PSYCHOLOGY")) %>% head(1)

## BIOLOGICAL\_PSYCHOLOGY DEVELOPMENTAL\_NEUROPSYCHOLOGY  
## 1 3.07 1.813  
## JOURNAL\_OF\_ABNORMAL\_PSYCHOLOGY  
## 1 4.133  
## JOURNAL\_OF\_CLINICAL\_AND\_EXPERIMENTAL\_NEUROPSYCHOLOGY NEUROPSYCHOLOGY  
## 1 1.839 3.286  
## QUARTERLY\_JOURNAL\_OF\_EXPERIMENTAL\_PSYCHOLOGY  
## 1 2.129  
## SCANDINAVIAN\_JOURNAL\_OF\_PSYCHOLOGY  
## 1 1.284

Columns which start with something

#Column name starts with FRONTIERS  
impactFactor\_wide %>% select(starts\_with("FRONTIERS")) %>% head(1)

## FRONTIERS\_IN\_HUMAN\_NEUROSCIENCE FRONTIERS\_IN\_PSYC  
## 1 3.209 2.321

Columns from a list

#Column name IS EITHER EMOTION OR COGNITION  
impactFactor\_wide %>% select(one\_of(c("EMOTION", "COGNITION"))) %>% head(1)

## EMOTION COGNITION  
## 1 3.251 3.414

Or selecting all columns EXCEPT a certain few

# NEGATIVE SIGN MEANS NOT THAT COLUMN  
impactFactor\_wide %>% select(-starts\_with("FRONTIERS")) %>% head(1)

## Year ACTA\_PSYC ADDICTIVE\_BEHAV AMER\_JOUR\_OF\_DRUG\_AND\_ALC\_ABUSE  
## 1 2016 2.031 2.944 2.161  
## AUTISM\_RESEARCH BEHAVIOR\_RESEARCH\_METHODS BILINGUALISM\_LANG\_AND\_COG  
## 1 3.765 3.623 3.01  
## BIOLOGICAL\_PSYCHOLOGY BRAIN\_AND\_COG BRAIN\_AND\_LANG COGNITION  
## 1 3.07 2.432 2.439 3.414  
## COGNITIVE\_NEUROPSYC CONSCIOUSNESS\_AND\_COGNITION  
## 1 1.146 2.144  
## CURRENT\_DIRECTIONS\_IN\_PSYCHOLOGICAL\_SCIENCE DEPRESSION\_AND\_ANXIETY  
## 1 5.255 4.971  
## DEVELOPMENTAL\_NEUROPSYCHOLOGY DEVELOPMENTAL\_SCIENCE EMOTION  
## 1 1.813 4.604 3.251  
## INTERNATIONAL\_JOURNAL\_OF\_PSYCHOPHYSIOLOGY JOURNAL\_OF\_ABNORMAL\_PSYCHOLOGY  
## 1 2.582 4.133  
## JOURNAL\_OF\_AUTISM\_AND\_DEVELOPMENTAL\_DISORDERS  
## 1 3.321  
## JOURNAL\_OF\_CHILD\_PSYCHOLOGY\_AND\_PSYCHIATRY  
## 1 6.226  
## JOURNAL\_OF\_CLINICAL\_AND\_EXPERIMENTAL\_NEUROPSYCHOLOGY  
## 1 1.839  
## JOURNAL\_OF\_COGNITIVE\_NEUROSCIENCE  
## 1 3.108  
## JOURNAL\_OF\_EXPERIMENTAL\_PSYCHOLOGY\_GENERAL  
## 1 4.42  
## JOURNAL\_OF\_EXPERIMENTAL\_PSYCHOLOGY\_LEARNING\_MEMORY\_AND\_COGNITION  
## 1 2.667  
## JOURNAL\_OF\_NEUROLINGUISTICS JOURNAL\_OF\_PSYCHOPHYSIOLOGY  
## 1 1.403 0.683  
## JOURNAL\_OF\_THE\_AMERICAN\_ACADEMY\_OF\_CHILD\_AND\_ADOLESCENT\_PSYCHIATRY  
## 1 6.442  
## JOURNAL\_OF\_THE\_INTERNATIONAL\_NEUROPSYCHOLOGICAL\_SOCIETY LATERALITY  
## 1 2.181 1.267  
## LEARNING\_AND\_MEMORY NEUROBIOLOGY\_OF\_LEARNING\_AND\_MEMORY NEUROCASE  
## 1 2.894 3.543 0.988  
## NEUROPSYCHOBIOLOGY NEUROPSYCHOLOGIA NEUROPSYCHOLOGY PERCEPTION  
## 1 1.491 3.197 3.286 1.087  
## PERSPECTIVES\_ON\_PSYCHOLOGICAL\_SCIENCE PHYSIOLOGY\_AND\_BEHAVIOR PSYC\_SCI  
## 1 7.359 2.341 5.667  
## PSYCHOLOGICAL\_MEDICINE PSYCHOLOGICAL\_RESEARCH\_PSYCHOLOGISCHE\_FORSCHUNG  
## 1 5.23 3.119  
## PSYCHOPHYSIOLOGY PSYCHOSOMATIC\_MEDICINE  
## 1 2.668 3.863  
## QUARTERLY\_JOURNAL\_OF\_EXPERIMENTAL\_PSYCHOLOGY  
## 1 2.129  
## SCANDINAVIAN\_JOURNAL\_OF\_PSYCHOLOGY SOCIAL\_COG\_AND\_AFF\_NEURO SOCIAL\_NEURO  
## 1 1.284 3.937 2.255

Finally, you can also use regular expressions to specify which columns you want.

Regular expressions are a way of specifying a pattern you want to match. Usually these patterns involve some combination of numbers and letters. Here O only gave two simple examples

# USES REGULAR EXPRESSIONS TO MATCH COLUMN NAMES  
  
#ANYTHING IN [ ] PRECEDING ? WILL BE MATCHED 0 OR 1 TIMES  
impactFactor\_wide %>% select(matches("COGN[ITIVE]?[ITION]?")) %>% head(1)

## COGNITION COGNITIVE\_NEUROPSYC CONSCIOUSNESS\_AND\_COGNITION  
## 1 3.414 1.146 2.144  
## JOURNAL\_OF\_COGNITIVE\_NEUROSCIENCE  
## 1 3.108  
## JOURNAL\_OF\_EXPERIMENTAL\_PSYCHOLOGY\_LEARNING\_MEMORY\_AND\_COGNITION  
## 1 2.667

#ANYTHING IN [ ] PRECEDING \* WILL BE MATCHED 0 OR INFINITE TIMES  
impactFactor\_wide %>% select(matches("COGN[A-Z]\*")) %>% head(1)

## COGNITION COGNITIVE\_NEUROPSYC CONSCIOUSNESS\_AND\_COGNITION  
## 1 3.414 1.146 2.144  
## JOURNAL\_OF\_COGNITIVE\_NEUROSCIENCE  
## 1 3.108  
## JOURNAL\_OF\_EXPERIMENTAL\_PSYCHOLOGY\_LEARNING\_MEMORY\_AND\_COGNITION  
## 1 2.667

#ANYTHING IN [ ] PRECEDING {n} WILL BE MATCHED exactly n times  
impactFactor\_wide %>% select(matches("COGN[A-Z]{5}")) %>% head(1)

## COGNITION COGNITIVE\_NEUROPSYC CONSCIOUSNESS\_AND\_COGNITION  
## 1 3.414 1.146 2.144  
## JOURNAL\_OF\_COGNITIVE\_NEUROSCIENCE  
## 1 3.108  
## JOURNAL\_OF\_EXPERIMENTAL\_PSYCHOLOGY\_LEARNING\_MEMORY\_AND\_COGNITION  
## 1 2.667

Similarly, you can use grep (a regular expression matcher to filter out rows).

df %>% filter(grepl("Otto, A. Ross", AF)) %>% select(c(AF))

## AF  
## 1 Blanco, Nathaniel J.; Otto, A. Ross; Maddox, W. Todd; Beevers, Christopher G.; Love, Bradley C.

##### JOINING DATAFRAMES

Finally, we’ll talk about how to merge data frames. Let us first load in three data frames.

We’ll be using two data frames of impact factor for two different journals. Notice how there are a few missing values in each data frame and that there are some duplictaes (in both data frames).

EMOTION

## Year Journal.Impact.Factor  
## 1 2016 3.251  
## 2 2015 3.082  
## 3 2012 3.269  
## 4 2011 3.875  
## 5 2008 3.073  
## 6 2007 3.088  
## 7 2006 3.053

FRONTIERS\_PSYC

## Year Journal.Impact.Factor  
## 1 2016 2.321  
## 2 2015 2.463  
## 3 2014 2.560  
## 4 2013 2.843

Left join, adds the right data frame to the left data frame based on the values the right has in common with the left one BUT IT WILL NOT ADD VALUES FROM THE RIGHT ONE NOT CONTAINED IN THE LEFT.

left\_join(EMOTION, FRONTIERS\_PSYC, "Year")

## Year Journal.Impact.Factor.x Journal.Impact.Factor.y  
## 1 2016 3.251 2.321  
## 2 2015 3.082 2.463  
## 3 2012 3.269 NA  
## 4 2011 3.875 NA  
## 5 2008 3.073 NA  
## 6 2007 3.088 NA  
## 7 2006 3.053 NA

Same as left join except here the left data frame is added to the right data frame.

right\_join(EMOTION, FRONTIERS\_PSYC, "Year")

## Year Journal.Impact.Factor.x Journal.Impact.Factor.y  
## 1 2016 3.251 2.321  
## 2 2015 3.082 2.463  
## 3 2014 NA 2.560  
## 4 2013 NA 2.843

Inner Join, joins the data frame based on common values to BOTH.

inner\_join(EMOTION, FRONTIERS\_PSYC, "Year")

## Year Journal.Impact.Factor.x Journal.Impact.Factor.y  
## 1 2016 3.251 2.321  
## 2 2015 3.082 2.463

Full join will completely join the two data frames.

full\_join(EMOTION, FRONTIERS\_PSYC, "Year")

## Year Journal.Impact.Factor.x Journal.Impact.Factor.y  
## 1 2016 3.251 2.321  
## 2 2015 3.082 2.463  
## 3 2012 3.269 NA  
## 4 2011 3.875 NA  
## 5 2008 3.073 NA  
## 6 2007 3.088 NA  
## 7 2006 3.053 NA  
## 8 2014 NA 2.560  
## 9 2013 NA 2.843

Semi join will only return the dataframe based on the values in the first data frame contained in the second data frame.

semi\_join(EMOTION, FRONTIERS\_PSYC, "Year")

## Year Journal.Impact.Factor  
## 1 2016 3.251  
## 2 2015 3.082

Anti join will return a data frame based on the values of the first data frame that DONT have a match in the second data frame.

anti\_join(EMOTION, FRONTIERS\_PSYC, "Year")

## Year Journal.Impact.Factor  
## 1 2012 3.269  
## 2 2011 3.875  
## 3 2008 3.073  
## 4 2007 3.088  
## 5 2006 3.053

EMOTION2

## Year Journal.Impact.Factor  
## 1 2016 3.251  
## 2 2015 3.082  
## 3 2014 3.383  
## 4 2013 3.371  
## 5 2012 3.269  
## 6 2011 3.875  
## 7 2010 3.027  
## 8 2009 3.726  
## 9 2008 3.073  
## 10 2007 3.088  
## 11 2006 3.053

EMOTION

## Year Journal.Impact.Factor  
## 1 2016 3.251  
## 2 2015 3.082  
## 3 2012 3.269  
## 4 2011 3.875  
## 5 2008 3.073  
## 6 2007 3.088  
## 7 2006 3.053

Intercet will return the values of the first data frame contained in the second one.

intersect(EMOTION, EMOTION2)

## Year Journal.Impact.Factor  
## 1 2016 3.251  
## 2 2015 3.082  
## 3 2012 3.269  
## 4 2011 3.875  
## 5 2008 3.073  
## 6 2007 3.088  
## 7 2006 3.053

Union will join the two data frames but collapsing the columns.

union(EMOTION, EMOTION2)

## Year Journal.Impact.Factor  
## 1 2009 3.726  
## 2 2010 3.027  
## 3 2013 3.371  
## 4 2014 3.383  
## 5 2006 3.053  
## 6 2007 3.088  
## 7 2008 3.073  
## 8 2011 3.875  
## 9 2012 3.269  
## 10 2015 3.082  
## 11 2016 3.251

setdiff will return the rows in the first data frame not contained in the second data frame.

setdiff(EMOTION2, EMOTION)

## Year Journal.Impact.Factor  
## 1 2014 3.383  
## 2 2013 3.371  
## 3 2010 3.027  
## 4 2009 3.726