dplyr and pipes

Install the tidyverse package set

Most usefull packages for modern R code : https://www.tidyverse.org/packages/ For those who learn to code R before 2015: http://kbroman.org/hipsteR/ Installs ggplot2, dplyr, tidyr, readr, purr, tibble and much more: 'colorspace', 'mnormt', 'bindr', 'RColorBrewer', 'dichromat', 'munsell', 'labeling', 'viridisLite', 'rematch', 'plyr', 'psych', 'reshape2', 'assertthat', 'bindrcpp', 'glue', 'pkgconfig', 'rlang', 'R6', 'Rcpp', 'BH', 'plogr', 'digest', 'gtable', 'scales', 'lazyeval', 'mime', 'curl', 'openssl', 'cellranger', 'stringi', 'selectr', 'tidyselect', 'broom', 'dplyr', 'forcats', 'ggplot2', 'haven', 'httr', 'hms', 'jsonlite', 'lubridate', 'magrittr', 'modelr', 'purrr', 'readr', 'readxl', 'stringr', 'tibble', 'rvest', 'tidyr', 'xml2'

```r
# install.packages("tidyverse")
library(tidyverse)
library(magrittr)

# other useful depedencies
library(tableone)
library(psych)
library(fBasics)


# Sometimes function mask each other !
# tib %>% filter(Sex == "Male", Eye=="Brown") # produce error
filter = dplyr::filter
# tib %>% filter(Sex == "Male", Eye=="Brown") # now its ok


# vignette("dplyr")
```

Load some datesets

```r
#install.packages("datasets")
# install.packages("mlbench")

# library(help= "mlbench") (Glass, BreastCancer, BostonHousin, Ionosphere, PimaIndiansDiabetes, Sonar)
library(datasets)
library(mlbench)
data("BostonHousing")
head(BostonHousing)
```

```
##      crim zn indus chas   nox    rm  age    dis rad tax ptratio      b
## 1 0.00632 18  2.31    0 0.538 6.575 65.2 4.0900   1 296    15.3 396.90
## 2 0.02731  0  7.07    0 0.469 6.421 78.9 4.9671   2 242    17.8 396.90
## 3 0.02729  0  7.07    0 0.469 7.185 61.1 4.9671   2 242    17.8 392.83
## 4 0.03237  0  2.18    0 0.458 6.998 45.8 6.0622   3 222    18.7 394.63
## 5 0.06905  0  2.18    0 0.458 7.147 54.2 6.0622   3 222    18.7 396.90
## 6 0.02985  0  2.18    0 0.458 6.430 58.7 6.0622   3 222    18.7 394.12
##   lstat medv
## 1  4.98 24.0
## 2  9.14 21.6
## 3  4.03 34.7
## 4  2.94 33.4
## 5  5.33 36.2
## 6  5.21 28.7
```

```
data("HairEyeColor")
HairEyeColor
```

```
## , , Sex = Male
## 
##        Eye
## Hair    Brown Blue Hazel Green
##   Black    32   11    10     3
##   Brown    53   50    25    15
##   Red      10   10     7     7
##   Blond     3   30     5     8
## 
## , , Sex = Female
## 
##        Eye
## Hair    Brown Blue Hazel Green
##   Black    36    9     5     2
##   Brown    66   34    29    14
##   Red      16    7     7     7
##   Blond     4   64     5     8
```

```
data("mtcars")
head(mtcars)
```

```
##                    mpg cyl disp  hp drat    wt  qsec vs am gear carb
## Mazda RX4         21.0   6  160 110 3.90 2.620 16.46  0  1    4    4
## Mazda RX4 Wag     21.0   6  160 110 3.90 2.875 17.02  0  1    4    4
## Datsun 710        22.8   4  108  93 3.85 2.320 18.61  1  1    4    1
## Hornet 4 Drive    21.4   6  258 110 3.08 3.215 19.44  1  0    3    1
## Hornet Sportabout 18.7   8  360 175 3.15 3.440 17.02  0  0    3    2
## Valiant           18.1   6  225 105 2.76 3.460 20.22  1  0    3    1
```

```
data("swiss")
head(swiss)
```

```
##              Fertility Agriculture Examination Education Catholic
## Courtelary        80.2        17.0          15        12     9.96
## Delemont          83.1        45.1           6         9    84.84
## Franches-Mnt      92.5        39.7           5         5    93.40
## Moutier           85.8        36.5          12         7    33.77
## Neuveville        76.9        43.5          17        15     5.16
## Porrentruy        76.1        35.3           9         7    90.57
##              Infant.Mortality
## Courtelary               22.2
## Delemont                 22.2
## Franches-Mnt             20.2
## Moutier                  20.3
## Neuveville               20.6
## Porrentruy               26.6
```

```r
data("BreastCancer")
head(BreastCancer)
```

```
##        Id Cl.thickness Cell.size Cell.shape Marg.adhesion Epith.c.size
## 1 1000025            5         1          1             1            2
## 2 1002945            5         4          4             5            7
## 3 1015425            3         1          1             1            2
## 4 1016277            6         8          8             1            3
## 5 1017023            4         1          1             3            2
## 6 1017122            8        10         10             8            7
##   Bare.nuclei Bl.cromatin Normal.nucleoli Mitoses     Class
## 1           1           3               1       1    benign
## 2          10           3               2       1    benign
## 3           2           3               1       1    benign
## 4           4           3               7       1    benign
## 5           1           3               1       1    benign
## 6          10           9               7       1 malignant
```

Manipulate data with dplyr and the pipe operator

Add a pipe operator : CTRL + SHIFT + M

```r
tib = as.tibble(HairEyeColor)

# Filtering and selecting
tib %>% filter(Sex == "Male", Eye=="Brown")
```

```
## # A tibble: 4 x 4
##    Hair   Eye   Sex     n
##   <chr> <chr> <chr> <dbl>
## 1 Black Brown  Male    32
## 2 Brown Brown  Male    53
## 3   Red Brown  Male    10
## 4 Blond Brown  Male     3
```

```r
tib %>% select(contains('S'))
```

```
## # A tibble: 32 x 1
##      Sex
##    <chr>
##  1  Male
##  2  Male
##  3  Male
##  4  Male
##  5  Male
##  6  Male
##  7  Male
##  8  Male
##  9  Male
## 10  Male
## # ... with 22 more rows
```

```r
tib %>% select(Eye:n)
```

```
## # A tibble: 32 x 3
##      Eye   Sex     n
##    <chr> <chr> <dbl>
##  1 Brown  Male    32
##  2 Brown  Male    53
##  3 Brown  Male    10
##  4 Brown  Male     3
##  5  Blue  Male    11
##  6  Blue  Male    50
##  7  Blue  Male    10
##  8  Blue  Male    30
##  9 Hazel  Male    10
## 10 Hazel  Male    25
## # ... with 22 more rows
```

```
mtcars %>%
  group_by(cyl, am) %>%
  select(mpg, cyl, wt, am) %>%
  summarise(avgmpg = mean(mpg), avgwt = mean(wt)) %>%
  filter(avgmpg > 20)
```

```
## # A tibble: 3 x 4
## # Groups:   cyl [2]
##     cyl    am   avgmpg   avgwt
##   <dbl> <dbl>    <dbl>   <dbl>
## 1     4     0 22.90000 2.93500
## 2     4     1 28.07500 2.04225
## 3     6     1 20.56667 2.75500
```

```
# cw %>% group_by(sid) %>% summarise(accuracy = mean(correct)) %>% arrange(desc(accuracy))

# select(iris, contains("."))
# Select columns whose name contains a character string.
# select(iris, ends_with("Length"))
# Select columns whose name ends with a character string.
# select(iris, everything()) Select every column.
# select(iris, matches(".t."))
# Select columns whose name matches a regular expression.
# select(iris, num_range("x", 1:5)) Select columns named x1, x2, x3, x4, x5.
# select(iris, one_of(c("Species", "Genus")))
# Select columns whose names are in a group of names.
# select(iris, starts_with("Sepal"))
# Select columns whose name starts with a character string.
# select(iris, Sepal.Length:Petal.Width)
# Select all columns between Sepal.Length and Petal.Width (inclusive).
# select(iris, -Species)
# Select all columns except Species.
```

Assignment

```
df <- mpg %>% filter(cty > 20, cyl == 4) %>% select(1:2)

mpg %>% filter(cty > 20, cyl == 4) %>% select(1:2) -> df
```

```
df %<>% filter(manufacturer != "honda")

df %>% table %>% as.tibble %>% filter(n!=0)
```

```
## # A tibble: 12 x 3
##    manufacturer        model     n
##           <chr>        <chr> <int>
##  1         audi           a4     2
##  2       nissan       altima     3
##  3       toyota        camry     4
##  4       toyota camry solara     4
##  5       toyota      corolla     5
##  6   volkswagen          gti     3
##  7       subaru  impreza awd     1
##  8   volkswagen        jetta     4
##  9    chevrolet       malibu     1
## 10   volkswagen   new beetle     3
## 11   volkswagen       passat     2
## 12      hyundai       sonata     2
```

```
all_letters <- c(letters, LETTERS) %>%
    sort %>%
    write.csv(file = "all_letters.csv")
```

Using functional sequences

```
library(magrittr) # needed to include the pipe operators
library(lubridate) # in tidyverse
```

```
##
## Attaching package: 'lubridate'

## The following object is masked from 'package:base':
##
##     date
```

```
read_year <- . %>% as.character %>% as.Date %>% year

# Creating a dataset
df <- data.frame(now = "2015-11-11", before = "2012-01-01")
#          now     before
# 1 2015-11-11 2012-01-01

# Example 1: applying `read_year` to a single character-vector
df$now %>% read_year
```

```
## [1] 2015
```

```
# [1] 2015

# Example 2: applying `read_year` to all columns of `df`
df %>% lapply(read_year) %>% as.data.frame  # implicit `lapply(df, read_year)
```

```
##    now before
## 1 2015   2012
```

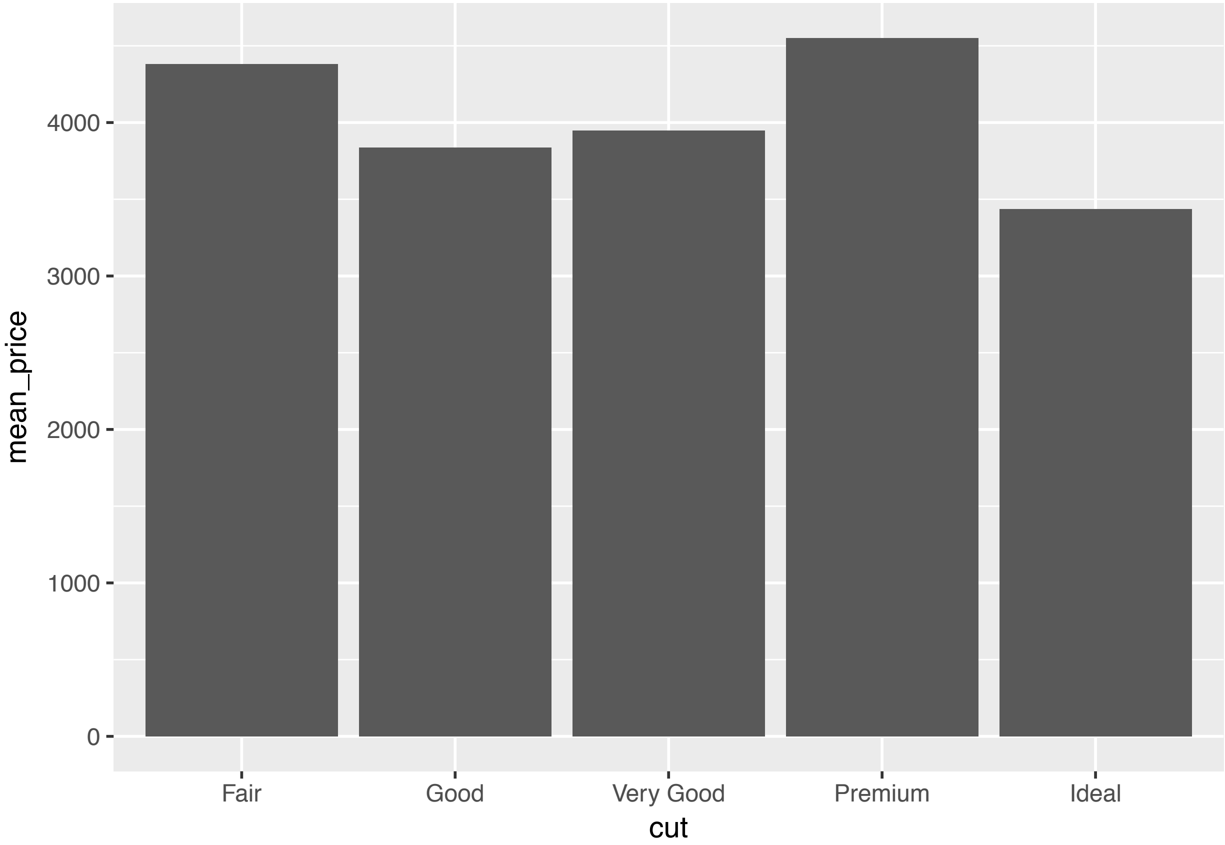
```
#    now before
# 1 2015   2012

# Example 3: same as above using `mutate_all`
library(dplyr)
df %>% mutate_all(funs(read_year))
```

```
##    now before
## 1 2015   2012
```

```
# if an older version of dplyr use `mutate_each`
#    now before
# 1 2015   2012
```

Plots

```
library(ggplot2)
diamonds %>%
    filter(depth > 60) %>%
    group_by(cut) %>%
    summarize(mean_price = mean(price)) %>%
    ggplot(aes(x = cut, y = mean_price)) +
        geom_bar(stat = "identity")
```



Basic renaming

```
require(pander)
```

```
## Loading required package: pander
```

```
# example of variable renaming
breastcancer = BreastCancer %>% rename_all(tolower)
cat("toLower", pandoc.list(names(breastcancer)))
```

- id
- cl.thickness
- cell.size
- cell.shape
- marg.adhesion
- epith.c.size
- bare.nuclei
- bl.cromatin
- normal.nucleoli
- mitoses
- class

toLower

```
# you could also use base functions
breastcancer %<>% setNames(tolower(gsub("\\.", "_", names(.))))
cat("setNames(tolower(gsub(.)))", pandoc.list(names(breastcancer)))
```

- id
- cl_thickness
- cell_size
- cell_shape
- marg_adhesion
- epith_c_size
- bare_nuclei
- bl_cromatin
- normal_nucleoli
- mitoses
- class

setNames(tolower(gsub(.)))

Reshaping

Base R function for reshaping datasets is reshape Some very usefull parameters

```
# Base R function
class(Indometh)
```

```
## [1] "nfnGroupedData" "nfGroupedData"  "groupedData"    "data.frame"
```

```
wide <- reshape(Indometh, v.names = "conc", idvar = "Subject",
                timevar = "time", direction = "wide")

df <- data.frame(id = rep(1:4, rep(2,4)),
                 visit = rep(c("Before","After"), 4),
                 x = rnorm(4), y = runif(4))
reshape(df, timevar = "visit", idvar = "id", direction = "wide")
```

```
##   id   x.Before  y.Before   x.After   y.After
## 1  1  0.1094677 0.1292292 -1.710644 0.2408263
## 3  2 -0.9349554 0.6174639 -1.466715 0.3039000
## 5  3  0.1094677 0.1292292 -1.710644 0.2408263
## 7  4 -0.9349554 0.6174639 -1.466715 0.3039000
```

```
reshape(df, timevar = "visit", idvar = "id", direction = "wide", v.names = "x")
```

```
## Warning in reshapeWide(data, idvar = idvar, timevar = timevar, varying =
## varying, : some constant variables (y) are really varying
```

```
##   id         y   x.Before   x.After
## 1  1 0.1292292  0.1094677 -1.710644
## 3  2 0.6174639 -0.9349554 -1.466715
## 5  3 0.1292292  0.1094677 -1.710644
## 7  4 0.6174639 -0.9349554 -1.466715
```

```
df3 <- data.frame(id = 1:4,
                  age = c(40,50,60,50),
                  dose1 = c(1,2,1,2),
                  dose2 = c(2,1,2,1),
                  dose4 = c(3,3,3,3))

reshape(df3, direction = "long", varying = 3:5, sep = "")
```

```
##     id age time dose
## 1.1  1  40    1    1
## 2.1  2  50    1    2
## 3.1  3  60    1    1
## 4.1  4  50    1    2
## 1.2  1  40    2    2
## 2.2  2  50    2    1
## 3.2  3  60    2    2
## 4.2  4  50    2    1
## 1.4  1  40    4    3
## 2.4  2  50    4    3
## 3.4  3  60    4    3
## 4.4  4  50    4    3
```

Use TidyR / Reshape2

Best organization for datasets

Rules:

- Each variable has its own column
- Each observation in its own row
- Each value is placed in its own cell

Functions gather, spread, separate, and unite are the building block of the tidyR library Here parameters are very simple so you have to have a good understanding of how you want to reshape your data.

Sometimes what you really want to do is create new variables: for that you need all the smaller functions to make it longer first, unite variables, then spread it out so that each variable has its own column.

```
stocks = data.frame(
  time = as.Date('2009-01-01') + 0:9,
  X = rnorm(10, 0, 1),
  Y = rnorm(10, 0, 2),
  Z = rnorm(10, 0, 4)
)
head(stocks)
```

```
##         time          X         Y          Z
## 1 2009-01-01 -0.6619441 -2.835567  3.5165349
## 2 2009-01-02 -2.0694765 -1.776735 -0.2858571
## 3 2009-01-03 -0.9062100  3.194403  1.0223352
## 4 2009-01-04  0.3172697 -2.198807  8.7843088
## 5 2009-01-05 -1.3213492  1.613739 -2.1011186
## 6 2009-01-06 -1.5920188 -3.006474 -1.3597501
```

```
# Gather all the selected variables in two columns stock and price
# you can use the minus sign to exclude columns from the selection
stocksm = stocks %>% gather(stock, price, -time)
stocksm %>% head
```

```
##         time stock      price
## 1 2009-01-01     X -0.6619441
## 2 2009-01-02     X -2.0694765
## 3 2009-01-03     X -0.9062100
## 4 2009-01-04     X  0.3172697
## 5 2009-01-05     X -1.3213492
## 6 2009-01-06     X -1.5920188
```

```
# You can invert what you just did using spread
stocksm %>% spread(stock, price) %>% head
```

```
##         time          X         Y          Z
## 1 2009-01-01 -0.6619441 -2.835567  3.5165349
## 2 2009-01-02 -2.0694765 -1.776735 -0.2858571
## 3 2009-01-03 -0.9062100  3.194403  1.0223352
## 4 2009-01-04  0.3172697 -2.198807  8.7843088
## 5 2009-01-05 -1.3213492  1.613739 -2.1011186
## 6 2009-01-06 -1.5920188 -3.006474 -1.3597501
```

```
# But you can also spread based on another variable
stocksm %>% spread(time, price) %>% head
```

```
##   stock 2009-01-01 2009-01-02 2009-01-03 2009-01-04 2009-01-05 2009-01-06
## 1     X -0.6619441 -2.0694765  -0.906210  0.3172697  -1.321349  -1.592019
## 2     Y -2.8355666 -1.7767351   3.194403 -2.1988068   1.613739  -3.006474
## 3     Z  3.5165349 -0.2858571   1.022335  8.7843088  -2.101119  -1.359750
##   2009-01-07 2009-01-08 2009-01-09 2009-01-10
## 1 -1.2628544  -2.304841  0.4334867  -0.173297
## 2 -0.1993526  -2.125841 -1.0827912  -1.958514
## 3  1.2422260  -2.814044  5.3319974   1.382366
```

```
# Use 'convert = TRUE' to produce variables of mixed type
df <- data.frame(id = rep(c(1, 2), each = 3),
                 var = c("Sepal.Length", "Species", "Species_num"),
```

```r
                 value = c(5.1, "setosa", 1, 7.0, "versicolor", 2))
df
```

```
##   id          var      value
## 1  1 Sepal.Length        5.1
## 2  1      Species     setosa
## 3  1  Species_num          1
## 4  2 Sepal.Length          7
## 5  2      Species versicolor
## 6  2  Species_num          2
```

```r
df %>% spread(var, value)
```

```
##   id Sepal.Length    Species Species_num
## 1  1          5.1     setosa           1
## 2  2            7 versicolor           2
```

```r
df %>% spread(var, value, convert = TRUE)
```

```
##   id Sepal.Length    Species Species_num
## 1  1          5.1     setosa           1
## 2  2          7.0 versicolor           2
```

```r
# Unite several columns into one
storms %>% unite(date, day:year, sep = "-")
```

```
## # A tibble: 10,010 x 11
##      name      date  hour   lat  long              status category  wind
##  * <chr>     <chr> <dbl> <dbl> <dbl>               <chr>    <ord> <int>
##   1   Amy 27-6-1975     0  27.5 -79.0 tropical depression       -1    25
##   2   Amy 27-6-1975     6  28.5 -79.0 tropical depression       -1    25
##   3   Amy 27-6-1975    12  29.5 -79.0 tropical depression       -1    25
##   4   Amy 27-6-1975    18  30.5 -79.0 tropical depression       -1    25
##   5   Amy 28-6-1975     0  31.5 -78.8 tropical depression       -1    25
##   6   Amy 28-6-1975     6  32.4 -78.7 tropical depression       -1    25
##   7   Amy 28-6-1975    12  33.3 -78.0 tropical depression       -1    25
##   8   Amy 28-6-1975    18  34.0 -77.0 tropical depression       -1    30
##   9   Amy 29-6-1975     0  34.4 -75.8      tropical storm        0    35
##  10   Amy 29-6-1975     6  34.0 -74.8      tropical storm        0    40
## # ... with 10,000 more rows, and 3 more variables: pressure <int>,
## #   ts_diameter <dbl>, hu_diameter <dbl>
```

```r
# Separate values into new column based on pattern
example <- tibble(`N [ears]` = c("173", "60", "54 [96]", "168 [328]", "906 [1685]"),
                  `% Otorrhea` = c("58.61%", "13.30%", "11.11%", "52.38%", "14.79% [10.45%]"))

example %>%
    separate(`N [ears]`, into = c("N_patients", "N_ears"), sep = "\\s\\[", fill = "right") %>%
    separate(`% Otorrhea`, into = c("pct_patients", "pct_ears"), sep = "\\s\\[", fill = "right") %>%
    mutate_each(funs(parse_number))
```

```
## `mutate_each()` is deprecated.
## Use `mutate_all()`, `mutate_at()` or `mutate_if()` instead.
## To map `funs` over all variables, use `mutate_all()`
```

```
## # A tibble: 5 x 4
##   N_patients N_ears pct_patients pct_ears
##        <dbl>  <dbl>        <dbl>    <dbl>
## 1        173     NA        58.61       NA
## 2         60     NA        13.30       NA
## 3         54     96        11.11       NA
## 4        168    328        52.38       NA
## 5        906   1685        14.79    10.45
```

```r
# Exercice
# We will try to do what the reshape function did for us on a
df <- data.frame(id = rep(1:4, rep(2,4)),
                 visit = rep(c("Before","After"), 4),
                 x = rnorm(4), y = runif(4))
reshape(df, timevar = "visit", idvar = "id", direction = "wide")
```

```
##   id    x.Before  y.Before    x.After    y.After
## 1  1 0.002239927 0.9390014 -0.7418142 0.92362157
## 3  2 1.977745827 0.2196017 -0.2763168 0.03209948
## 5  3 0.002239927 0.9390014 -0.7418142 0.92362157
## 7  4 1.977745827 0.2196017 -0.2763168 0.03209948
```

```r
# SOLUTION

 # first gather variales into two columns called variable and value
df %<>% gather(variable, value, -(c("id", "visit"))) %>% print
```

```
##    id  visit variable        value
## 1   1 Before        x  0.002239927
## 2   1  After        x -0.741814188
## 3   2 Before        x  1.977745827
## 4   2  After        x -0.276316788
## 5   3 Before        x  0.002239927
## 6   3  After        x -0.741814188
## 7   4 Before        x  1.977745827
## 8   4  After        x -0.276316788
## 9   1 Before        y  0.939001360
## 10  1  After        y  0.923621567
## 11  2 Before        y  0.219601650
## 12  2  After        y  0.032099484
## 13  3 Before        y  0.939001360
## 14  3  After        y  0.923621567
## 15  4 Before        y  0.219601650
## 16  4  After        y  0.032099484
```

```r
# Then create the new variables X before X after and Y before Y after by uniting the two columns
df %<>% unite(variable, variable, visit, sep = "_") %>% print
```

```
##    id variable        value
## 1   1 x_Before  0.002239927
## 2   1  x_After -0.741814188
## 3   2 x_Before  1.977745827
## 4   2  x_After -0.276316788
## 5   3 x_Before  0.002239927
## 6   3  x_After -0.741814188
## 7   4 x_Before  1.977745827
## 8   4  x_After -0.276316788
## 9   1 y_Before  0.939001360
## 10  1  y_After  0.923621567
## 11  2 y_Before  0.219601650
## 12  2  y_After  0.032099484
## 13  3 y_Before  0.939001360
## 14  3  y_After  0.923621567
## 15  4 y_Before  0.219601650
## 16  4  y_After  0.032099484
```

```
# Finally spread into one column per variable
df %<>% spread(variable, value) %>% print
```

```
##   id    x_After    x_Before    y_After  y_Before
## 1  1 -0.7418142 0.002239927 0.92362157 0.9390014
## 2  2 -0.2763168 1.977745827 0.03209948 0.2196017
## 3  3 -0.7418142 0.002239927 0.92362157 0.9390014
## 4  4 -0.2763168 1.977745827 0.03209948 0.2196017
```

melt and cast are the two main functions of reshape2

```
# convert matrix to data.frame
HairEyeColor %>% reshape2::melt() -> tib
tib = as.tibble(HairEyeColor) # !! Does not transforms character columns as factor
```

Basic stats

```
# Linear regression
# Logistic regression
```

Pandoc

```
require(fBasics)
mpg %>% mutate_all(as.numeric) %>% basicStats %>% t %>% pandoc.table
```

```
## Warning in evalq(as.numeric(manufacturer), <environment>): NAs introduced
## by coercion

## Warning in evalq(as.numeric(model), <environment>): NAs introduced by
## coercion

## Warning in evalq(as.numeric(trans), <environment>): NAs introduced by
## coercion

## Warning in evalq(as.numeric(drv), <environment>): NAs introduced by
## coercion

## Warning in evalq(as.numeric(fl), <environment>): NAs introduced by coercion

## Warning in evalq(as.numeric(class), <environment>): NAs introduced by
## coercion

## Warning in min(X): no non-missing arguments to min; returning Inf

## Warning in max(X): no non-missing arguments to max; returning -Inf

## Warning in min(X): no non-missing arguments to min; returning Inf
## Warning in max(X): no non-missing arguments to max; returning -Inf

## Warning in min(X): no non-missing arguments to min; returning Inf

## Warning in max(X): no non-missing arguments to max; returning -Inf

## Warning in min(X): no non-missing arguments to min; returning Inf

## Warning in max(X): no non-missing arguments to max; returning -Inf

## Warning in min(X): no non-missing arguments to min; returning Inf

## Warning in max(X): no non-missing arguments to max; returning -Inf
```

Table 1: Table continues below

	nobs	NAs	Minimum	Maximum	1. Quartile	3. Quartile
manufacturer	234	234	Inf	-Inf	NA	NA
model	234	234	Inf	-Inf	NA	NA
displ	234	0	1.6	7	2.4	4.6
year	234	0	1999	2008	1999	2008
cyl	234	0	4	8	4	8
trans	234	234	Inf	-Inf	NA	NA
drv	234	131	4	4	4	4
cty	234	0	9	35	14	19
hwy	234	0	12	44	18	27
fl	234	234	Inf	-Inf	NA	NA
class	234	234	Inf	-Inf	NA	NA

Table 2: Table continues below

	Mean	Median	Sum	SE Mean	LCL Mean	UCL Mean
manufacturer	NA	NA	0	NA	NA	NA
model	NA	NA	0	NA	NA	NA
displ	3.472	3.3	812.4	0.08446	3.305	3.638
year	2004	2004	468819	0.2948	2003	2004
cyl	5.889	6	1378	0.1053	5.681	6.096
trans	NA	NA	0	NA	NA	NA
drv	4	4	412	0	4	4
cty	16.86	17	3945	0.2782	16.31	17.41
hwy	23.44	24	5485	0.3893	22.67	24.21
fl	NA	NA	0	NA	NA	NA
class	NA	NA	0	NA	NA	NA

	Variance	Stdev	Skewness	Kurtosis
manufacturer	NA	NA	NA	NA
model	NA	NA	NA	NA
displ	1.669	1.292	0.4386	-0.9106
year	20.34	4.51	0	-2.009
cyl	2.597	1.612	0.1123	-1.464
trans	NA	NA	NA	NA
drv	0	0	NA	NA
cty	18.11	4.256	0.7864	1.431
hwy	35.46	5.955	0.3645	0.1369

	Variance	Stdev	Skewness	Kurtosis
fl	NA	NA	NA	NA
class	NA	NA	NA	NA

df2latex

A couple of xx2latex function allows to transform data into a latex format that can help produce clean graphics. To show the graphics or in general to use the output of a code chunk as if you wrote the RMarkdown code directly, use the chunk option results="asis"

```
require(fBasics)
require(psych)

# some utilities
to.latex.table = function (data, ...) {
    data %>%
        df2latex(silent = T, file="tempcorrtest.txt", ...) %>%
        gsub(pattern = "\\begin{tabular}", replacement = "\\resizebox{\\textwidth}{!}{\\begin{tabular}"
        gsub(pattern = "\\end{tabular}", replacement = "\\end{tabular}}", ., fixed = T) %>%
        paste0(collapse = " ")
}

corr.test.2latex = function (data, variables = NULL, method="spearman", ...) {
    if (is.null(variables)) { variables = names(data)}

    rs = corr.test(data %>% select(variables), method=method) #find the probabilities of the correlatio
    cp =corr.p(r=rs$r, n = NROW(data))

    cpp = rs$r
    cpp[cp$p>0.05] = 3

    colnames(cpp) = paste0(1:NROW(cpp), ".")
    rownames(cpp) = paste(colnames(cpp), gsub("_", " ", rownames(cpp), perl=T))

    cpp[upper.tri(cpp)] = NA
    cpp %>%
        df2latex(silent = T, file="tempcorrtest.txt", ...) %>%
        gsub(pattern = "3.00",replacement = ".", ., fixed = T) %>%
        gsub(pattern = "\\begin{tabular}", replacement = "\\resizebox{\\textwidth}{!}{\\begin{tabular}"
        gsub(pattern = "\\end{tabular}", replacement = "\\end{tabular}}", ., fixed = T) %>%
        paste0(collapse = " ")

}

shapiro.test.p = function (x) { shapiro.test(x)$p.value }
shapiro.test.star = function (x) {if (shapiro.test(x)$p.value<0.05) { stars.pval(shapiro.test(x)$p.valu

cat(mpg %>% mutate_all(as.numeric) %>% basicStats %>% t %>% to.latex.table)
```

```
cat(breastcancer %>% select(-id) %>% mutate_all(as.numeric) %>% corr.test.2latex)
```

Table 4: df2latex

A table from the psych package in R

Variable	nobs	NAs	Minmm	Maxmm	1.Qrt	3.Qrt	Mean	Medin	Sum	SEMen	LCLMn	UCLMn	Varnc	Stdev	Skwns	Krtss
manufacturer	234.00	234.00	Inf	-Inf			NaN		0.00						NaN	NaN
model	234.00	234.00	Inf	-Inf			NaN		0.00						NaN	NaN
displ	234.00	0.00	1.60	7.00	2.40	4.60	3.47	3.30	812.40	0.08	3.31	3.64	1.67	1.29	0.44	-0.91
year	234.00	0.00	1999.00	2008.00	1999.00	2008.00	2003.50	2003.50	468819.00	0.29	2002.92	2004.08	20.34	4.51	0.00	-2.01
cyl	234.00	0.00	4.00	8.00	4.00	8.00	5.89	6.00	1378.00	0.11	5.68	6.10	2.60	1.61	0.11	-1.46
trans	234.00	234.00	Inf	-Inf			NaN		0.00						NaN	NaN
drv	234.00	131.00	4.00	4.00	4.00	4.00	4.00	4.00	412.00	0.00	4.00	4.00	0.00	0.00	NaN	NaN
cty	234.00	0.00	9.00	35.00	14.00	19.00	16.86	17.00	3945.00	0.28	16.31	17.41	18.11	4.26	0.79	1.43
hwy	234.00	0.00	12.00	44.00	18.00	27.00	23.44	24.00	5485.00	0.39	22.67	24.21	35.46	5.95	0.36	0.14
fl	234.00	234.00	Inf	-Inf			NaN		0.00						NaN	NaN
class	234.00	234.00	Inf	-Inf			NaN		0.00						NaN	NaN

Table 5: df2latex

A table from the psych package in R

Variable	1.	2.	3.	4.	5.	6.	7.	8.	9.	10.
1. cl thickness	1.00									
2. cell size	0.67	1.00								
3. cell shape	0.66	0.89	1.00							
4. marg adhesion	0.54	0.74	0.71	1.00						
5. epith c size	0.58	0.79	0.76	0.67	1.00					
6. bare nuclei	0.59	0.77	0.75	0.70	0.69	1.00				
7. bl cromatin	0.54	0.72	0.69	0.62	0.64	0.68	1.00			
8. normal nucleoli	0.57	0.76	0.73	0.63	0.71	0.66	0.66	1.00		
9. mitoses	0.42	0.51	0.47	0.45	0.48	0.47	0.39	0.50	1.00	
10. class	0.68	0.86	0.84	0.73	0.76	0.84	0.74	0.74	0.53	1.00

Tables

A useful package to create tables is tableone. install.packages("tableone")

```
library(tableone)
generate_table = function(data, columns = NULL, strata_g = NULL) {
  if (is.null(columns)) { columns = names(data) }
  columns = unique(columns)
  catVar = NULL
  sapply(columns, function(name) {
    if (class(data[[name]])%in%"factor") {
      catVar <<- c(catVar, name)
    }
  })

  data_filtered = data %>% select(unique(c(columns,strata_g)))

  if (is.null(strata_g)) {
    table_g <- CreateTableOne(vars = columns, data = data_filtered, factorVars = catVar, includeNA = T)
  } else {
    table_g <- CreateTableOne(columns, data_filtered, catVar, strata = strata_g, includeNA = T)
  }


  mat_g <- print(table_g, exact = "stage", quote = FALSE, noSpaces = TRUE, printToggle = FALSE)

  return(mat_g)
}
```

```
breastcancer %<>% mutate_each(funs(as.numeric), -id, -class)
```

```
## `mutate_each()` is deprecated.
## Use `mutate_all()`, `mutate_at()` or `mutate_if()` instead.
## To map `funs` over a selection of variables, use `mutate_at()`
```

```
breastcancer %>% select(-id) %>% generate_table(strata_g = "class")
```

```
##                               Stratified by class
##                                benign        malignant     p        test
##   n                            "458"         "241"         ""       ""
##   cl_thickness (mean (sd))     "2.96 (1.67)" "7.20 (2.43)" "<0.001" ""
##   cell_size (mean (sd))        "1.33 (0.91)" "6.57 (2.72)" "<0.001" ""
##   cell_shape (mean (sd))       "1.44 (1.00)" "6.56 (2.56)" "<0.001" ""
##   marg_adhesion (mean (sd))    "1.36 (1.00)" "5.55 (3.21)" "<0.001" ""
##   epith_c_size (mean (sd))     "2.12 (0.92)" "5.30 (2.45)" "<0.001" ""
##   bare_nuclei (mean (sd))      "1.35 (1.18)" "7.63 (3.12)" "<0.001" ""
##   bl_cromatin (mean (sd))      "2.10 (1.08)" "5.98 (2.27)" "<0.001" ""
##   normal_nucleoli (mean (sd)) "1.29 (1.06)" "5.86 (3.35)" "<0.001" ""
##   mitoses (mean (sd))          "1.06 (0.50)" "2.53 (2.39)" "<0.001" ""
##   class = malignant (%)        "0 (0.0)"     "241 (100.0)" "<0.001" ""
```

Recode

http://dplyr.tidyverse.org/reference/recode.html

```
x <- sample(c("a", "b", "c"), 10, replace = TRUE)
recode(x, a = "Apple")
```

```
##  [1] "b"     "b"     "b"     "c"     "Apple" "c"     "c"     "b"
##  [9] "Apple" "b"
```

TidyR

```
# nested dataframes
# data %>% group_by(x) %>% nest()
```

BROOM

http://varianceexplained.org/r/broom-intro/

Survival

https://github.com/pavopax/gists/blob/master/survival-in-R.md

Purrr