CM535 Data science development

Week 2 : Data preparation and data cleaning

Benjamin Lacroix
b.m.e.lacroix@rgu.ac.uk

February 4, 2020

Introduction

Data preparation and data cleaning in R

- Loading data
- Type conversion
- Consistent data
- Data transformation
- Feature selection

Data type in R

Data types in R:

- numeric: Numeric data (approximations of the real numbers, R)
- integer: Integer data (whole numbers, Z)
- factor: Categorical data (simple classifications, like gender)
- ordered: Ordinal data (ordered classifications, like educational level)
- character: Character data (strings)
- logical: TRUE or FALSE values

```
class(c(1, 2))

## [1] "numeric"

class(c("abc", "bca"))

## [1] "character"
```

Special values in R: NA

NA: *Not Available*. NA is a placeholder for a missing value. Most operation work with NA but will return NA

```
c(NA, 1, 2)
```

```
## [1] NA  1  2
```

```
sum(c(NA, 1, 2))
```

```
## [1] NA
```

Special values in R: NULL

NULL: An empty set, it has no length and no class.

```
c(NULL, 1, 2)
```

```
## [1] 1 2
```

```
sum(c(NULL, 1, 2))
```

```
## [1] 3
```

Special values in R: Inf

Inf: stands for infinity. It is a numeric.

```
3/0
```

```
## [1] Inf
```

```
Inf+Inf
```

```
## [1] Inf
```

```
sort(c(Inf,3, 10,-5,-Inf))
```

```
## [1] -Inf   -5    3   10  Inf
```

Special values in R: Nan

NaN: Not a Number. It is the result of a calculation where the result is unknown.

```
1+NaN

## [1] NaN

sqrt(-2)

## Warning in sqrt(-2):  NaNs produced

## [1] NaN
```

Data structures

Collection which can only hold ony type/class of variables

- Vectors: one dimensional
- Matrices: two dimensional
- Arrays: 3+ dimensional

```
class(c(1,"2",TRUE))
```

```
## [1] "character"
```

```
class(matrix(1:16,nrow = 4, ncol = 4))
```

```
## [1] "matrix"
```

Data structures

Lists: an R object which can contain elements of different types

```
person <- list(age = 3, children = c("John", "Kate"), unemployed = FALSE)
person

## $age
## [1] 3
##
## $children
## [1] "John" "Kate"
##
## $unemployed
## [1] FALSE

person$children

## [1] "John" "Kate"

person[[2]]

## [1] "John" "Kate"

person[[3]]

## [1] FALSE
```

Data structures

Data frames: tabular data object. Unlike a matrix in data frame each column can contain different modes of data.

```
stIds <- c(1,2,3,4,5)
stNames<- c("John","Mark","Dave","Kate","Anna")
stGrades<-c("Excellent","Good","Bad","Very Bad","Bad")


df <- data.frame(student_id=stIds,student_name=stNames,student_grades=stGrades)
df

##   student_id student_name student_grades
## 1          1         John      Excellent
## 2          2         Mark           Good
## 3          3         Dave            Bad
## 4          4         Kate       Very Bad
## 5          5         Anna            Bad
```

Loading data in R

In practice, you'll be brought to import data from external files, typically a 'comma seperated value' (.csv) file

adult.csv

```
age,workclass,fnlwgt,education,education-num,marital-status,occupation,relationship,race,sex,capital-gain,capital-loss,hours-per-week,native-country,label
39,State-gov,77516,Bachelors,13,Never-married,Adm-clerical,Not-in-family,White,Male,2174,0,40,United-States,<=50K
50,Self-emp-not-inc,83311,Bachelors,13,Married-civ-spouse,Exec-managerial,Husband,White,Male,0,0,13,United-States,<=50K
38,Private,215646,HS-grad,9,Divorced,Handlers-cleaners,Not-in-family,White,Male,0,0,40,United-States,<=50K
53,Private,234721,11th,7,Married-civ-spouse,Handlers-cleaners,Husband,Black,Male,0,0,40,United-States,<=50K
28,Private,338409,Bachelors,13,Married-civ-spouse,Prof-specialty,Wife,Black,Female,0,0,40,Cuba,<=50K
37,Private,284582,Masters,14,Married-civ-spouse,Exec-managerial,Wife,White,Female,0,0,40,United-States,<=50K
49,Private,160187,9th,5,Married-spouse-absent,Other-service,Not-in-family,Black,Female,0,0,16,Jamaica,<=50K
52,Self-emp-not-inc,209642,HS-grad,9,Married-civ-spouse,Exec-managerial,Husband,White,Male,0,0,45,United-States,>50K
31,Private,45781,Masters,14,Never-married,Prof-specialty,Not-in-family,White,Female,14084,0,50,United-States,>50K
42,Private,159449,Bachelors,13,Married-civ-spouse,Exec-managerial,Husband,White,Male,5178,0,40,United-States,>50K
37,Private,280464,Some-college,10,Married-civ-spouse,Exec-managerial,Husband,Black,Male,0,0,80,United-States,>50K
30,State-gov,141297,Bachelors,13,Married-civ-spouse,Prof-specialty,Husband,Asian-Pac-Islander,Male,0,0,40,India,>50K
23,Private,122272,Bachelors,13,Never-married,Adm-clerical,Own-child,White,Female,0,0,30,United-States,<=50K
32,Private,205019,Assoc-acdm,12,Never-married,Sales,Not-in-family,Black,Male,0,0,50,United-States,<=50K
40,Private,121772,Assoc-voc,11,Married-civ-spouse,Craft-repair,Husband,Asian-Pac-Islander,Male,0,0,40,?,>50K
34,Private,245487,7th-8th,4,Married-civ-spouse,Transport-moving,Husband,Amer-Indian-Eskimo,Male,0,0,45,Mexico,<=50K
25,Self-emp-not-inc,176756,HS-grad,9,Never-married,Farming-fishing,Own-child,White,Male,0,0,35,United-States,<=50K
32,Private,186824,HS-grad,9,Never-married,Machine-op-inspct,Unmarried,White,Male,0,0,40,United-States,<=50K
38,Private,28887,11th,7,Married-civ-spouse,Sales,Husband,White,Male,0,0,50,United-States,<=50K
43,Self-emp-not-inc,292175,Masters,14,Divorced,Exec-managerial,Unmarried,White,Female,0,0,45,United-States,>50K
40,Private,193524,Doctorate,16,Married-civ-spouse,Prof-specialty,Husband,White,Male,0,0,60,United-States,>50K
54,Private,302146,HS-grad,9,Separated,Other-service,Unmarried,Black,Female,0,0,20,United-States,<=50K
35,Federal-gov,76845,9th,5,Married-civ-spouse,Farming-fishing,Husband,Black,Male,0,0,40,United-States,<=50K
43,Private,117037,11th,7,Married-civ-spouse,Transport-moving,Husband,White,Male,0,2042,40,United-States,<=50K
59,Private,109015,HS-grad,9,Divorced,Tech-support,Unmarried,White,Female,0,0,40,United-States,<=50K
56,Local-gov,216851,Bachelors,13,Married-civ-spouse,Tech-support,Husband,White,Male,0,0,40,United-States,>50K
19,Private,168294,HS-grad,9,Never-married,Craft-repair,Own-child,White,Male,0,0,40,United-States,<=50K
54,?,180211,Some-college,10,Married-civ-spouse,?,Husband,Asian-Pac-Islander,Male,0,0,60,South,>50K
39,Private,367260,HS-grad,9,Divorced,Exec-managerial,Not-in-family,White,Male,0,0,80,United-States,<=50K
49,Private,193366,HS-grad,9,Married-civ-spouse,Craft-repair,Husband,White,Male,0,0,40,United-States,<=50K
23,Local-gov,190709,Assoc-acdm,12,Never-married,Protective-serv,Not-in-family,White,Male,0,0,52,United-States,<=50K
20,Private,266015,Some-college,10,Never-married,Sales,Own-child,Black,Male,0,0,44,United-States,<=50K
45,Private,386940,Bachelors,13,Divorced,Exec-managerial,Own-child,White,Male,0,1408,40,United-States,<=50K
30,Federal-gov,59951,Some-college,10,Married-civ-spouse,Adm-clerical,Own-child,White,Male,0,0,40,United-States,<=50K
22,State-gov,311512,Some-college,10,Married-civ-spouse,Other-service,Husband,Black,Male,0,0,15,United-States,<=50K
48,Private,242406,11th,7,Never-married,Machine-op-inspct,Unmarried,White,Male,0,0,40,Puerto-Rico,<=50K
21,Private,197200,Some-college,10,Never-married,Machine-op-inspct,Own-child,White,Male,0,0,40,United-States,<=50K
19,Private,544091,HS-grad,9,Married-AF-spouse,Adm-clerical,Wife,White,Female,0,0,25,United-States,<=50K
31,Private,84154,Some-college,10,Married-civ-spouse,Sales,Husband,White,Male,0,0,38,?,>50K
```

- *read.table()* function is the most flexible function. And the following functions actually use *read.table()* with some fixed parameters
- *read.csv()*: comma separated values with period as decimal separator.
- *read.csv2()*: semicolon separated values with comma as decimal separator.

Loading data in R

```
adultData <- read.csv("adult.csv")
class(adultData)

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

Argument	Description
header	Does the first line contain column names?
col.names	character vector with column names
na.string	Which strings should be considered NA?
colClasses	character vector with the types of columns.
stringsAsFactors	If TRUE, converts all character vectors into factor vectors.
sep	Field separator

First look at your data

Once you have loaded your data it is important to have a first look at it using

- head(): shows the first few rows of your data frame
- summary(): provides basic statistics on each column
- str(): displays the internal structure of your data

First look at your data: head

```
head(adultData)
```

```
##   age         workclass fnlwgt education education.num      marital.status
## 1  39         State-gov  77516 Bachelors            13       Never-married
## 2  50 Self-emp-not-inc  83311 Bachelors            13  Married-civ-spouse
## 3  38           Private 215646   HS-grad             9            Divorced
## 4  53           Private 234721      11th             7  Married-civ-spouse
## 5  28           Private 338409 Bachelors            13  Married-civ-spouse
## 6  37           Private 284582   Masters            14  Married-civ-spouse
##           occupation  relationship  race    sex capital.gain capital.loss
## 1      Adm-clerical Not-in-family White   Male         2174            0
## 2   Exec-managerial       Husband White   Male            0            0
## 3 Handlers-cleaners Not-in-family White   Male            0            0
## 4 Handlers-cleaners       Husband Black   Male            0            0
## 5    Prof-specialty          Wife Black Female            0            0
## 6   Exec-managerial          Wife White Female            0            0
##   hours.per.week native.country label
## 1             40  United-States <=50K
## 2             13  United-States <=50K
## 3             40  United-States <=50K
## 4             40  United-States <=50K
## 5             40           Cuba <=50K
## 6             40  United-States <=50K
```

First look at your data: str

```
str(adultData)

## 'data.frame': 32561 obs. of  15 variables:
##  $ age           : int  39 50 38 53 28 37 49 52 31 42 ...
##  $ workclass     : Factor w/ 9 levels "?","Federal-gov",..: 8 7 5 5 5 5 5 7 5 5 
##  $ fnlwgt        : int  77516 83311 215646 234721 338409 284582 160187 209642 457
##  $ education     : Factor w/ 16 levels "10th","11th",..: 10 10 12 2 10 13 7 12 13
##  $ education.num : int  13 13 9 7 13 14 5 9 14 13 ...
##  $ marital.status: Factor w/ 7 levels "Divorced","Married-AF-spouse",..: 5 3 1 3 
##  $ occupation    : Factor w/ 15 levels "?","Adm-clerical",..: 2 5 7 7 11 5 9 5 11
##  $ relationship  : Factor w/ 6 levels "Husband","Not-in-family",..: 2 1 2 1 6 6 2
##  $ race          : Factor w/ 5 levels "Amer-Indian-Eskimo",..: 5 5 5 3 3 5 3 5 5 
##  $ sex           : Factor w/ 2 levels "Female","Male": 2 2 2 2 1 1 1 2 1 2 ...
##  $ capital.gain  : int  2174 0 0 0 0 0 0 0 14084 5178 ...
##  $ capital.loss  : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ hours.per.week: int  40 13 40 40 40 40 16 45 50 40 ...
##  $ native.country: Factor w/ 42 levels "?","Cambodia",..: 40 40 40 40 6 40 24 40
##  $ label         : Factor w/ 2 levels "<=50K",">50K": 1 1 1 1 1 1 1 2 2 2 ...
```

First look at your data: summary

```
summary(adultData)
```

```
##       age                    workclass          fnlwgt
##  Min.   :17.00   Private          :22696   Min.   :  12285
##  1st Qu.:28.00   Self-emp-not-inc: 2541   1st Qu.: 117827
##  Median :37.00   Local-gov        : 2093   Median : 178356
##  Mean   :38.58   ?                : 1836   Mean   : 189778
##  3rd Qu.:48.00   State-gov        : 1298   3rd Qu.: 237051
##  Max.   :90.00   Self-emp-inc     : 1116   Max.   :1484705
##                  (Other)          :  981
##         education     education.num                  marital.status
##  HS-grad     :10501   Min.   : 1.00   Divorced                  : 4443
##  Some-college: 7291   1st Qu.: 9.00   Married-AF-spouse         :   23
##  Bachelors   : 5355   Median :10.00   Married-civ-spouse        :14976
##  Masters     : 1723   Mean   :10.08   Married-spouse-absent     :  418
##  Assoc-voc   : 1382   3rd Qu.:12.00   Never-married             :10683
##  11th        : 1175   Max.   :16.00   Separated                 : 1025
##  (Other)     : 5134                   Widowed                   :  993
##            occupation           relationship                    race
##  Prof-specialty :4140   Husband       :13193   Amer-Indian-Eskimo:  311
##  Craft-repair   :4099   Not-in-family : 8305   Asian-Pac-Islander: 1039
##  Exec-managerial:4066   Other-relative:  981   Black             : 3124
##  Adm-clerical   :3770   Own-child     : 5068   Other             :  271
##  Sales          :3650   Unmarried     : 3446   White             :27816
```

First look at your data

```
adultData <- read.csv("adult.csv", na.string = "?")
summary(adultData)
```

```
##       age                    workclass          fnlwgt
##  Min.   :17.00   Private          :22696   Min.   :  12285
##  1st Qu.:28.00   Self-emp-not-inc: 2541   1st Qu.: 117827
##  Median :37.00   Local-gov        : 2093   Median : 178356
##  Mean   :38.58   State-gov        : 1298   Mean   : 189778
##  3rd Qu.:48.00   Self-emp-inc     : 1116   3rd Qu.: 237051
##  Max.   :90.00   (Other)          :  981   Max.   :1484705
##                  NA's             : 1836
##         education     education.num                   marital.status
##  HS-grad     :10501   Min.   : 1.00   Divorced                  : 4443
##  Some-college: 7291   1st Qu.: 9.00   Married-AF-spouse         :   23
##  Bachelors   : 5355   Median :10.00   Married-civ-spouse        :14976
##  Masters     : 1723   Mean   :10.08   Married-spouse-absent     :  418
##  Assoc-voc   : 1382   3rd Qu.:12.00   Never-married             :10683
##  11th        : 1175   Max.   :16.00   Separated                 : 1025
##  (Other)     : 5134                   Widowed                   :  993
##            occupation              relationship                    race
##  Prof-specialty : 4140   Husband       :13193   Amer-Indian-Eskimo:  311
##  Craft-repair   : 4099   Not-in-family : 8305   Asian-Pac-Islander: 1039
##  Exec-managerial: 4066   Other-relative:  981   Black             : 3124
##  Adm-clerical   : 3770   Own-child     : 5068   Other             :  271
```

Technically correct data

Ensuring that your data is in the right type.

```
str(df)

## 'data.frame': 5 obs. of  3 variables:
##  $ student_id    : num  1 2 3 4 5
##  $ student_name  : Factor w/ 5 levels "Anna","Dave",..: 3 5 2 4 1
##  $ student_grades: Factor w/ 4 levels "Bad","Excellent",..: 2 3 1 4 1

newStudent <- c(6,"Nath","Average")
rbind(df,newStudent)

## Warning in `[<-.factor`(`*tmp*`, ri, value = "Nath"):  invalid factor level,
NA generated
## Warning in `[<-.factor`(`*tmp*`, ri, value = "Nath"):  invalid factor level,
NA generated

##   student_id student_name student_grades
## 1          1         John      Excellent
## 2          2         Mark           Good
## 3          3         Dave            Bad
## 4          4         Kate       Very Bad
## 5          5         Anna            Bad
## 6          6         <NA>           <NA>
```

Technically correct data

```
# change grades type in an ordered factor
df$student_grades <- factor(df$student_grades,
          levels = c("Very Bad", "Bad", "Average", "Good", "Excellent"), ordered = T
# change IDs and names to characters
df$student_id <- as.character(df$student_id)
df$student_name <- as.character(df$student_name)
str(df)

## 'data.frame': 5 obs. of  3 variables:
##  $ student_id    : chr  "1" "2" "3" "4" ...
##  $ student_name  : chr  "John" "Mark" "Dave" "Kate" ...
##  $ student_grades: Ord.factor w/ 5 levels "Very Bad"<"Bad"<..: 5 4 2 1 2
```

Dates

Dates and times in a dataset will come in two form

- a string of character such as "2011-03-27 01:30:00"
- a timestamp that corresponds to the number of seconds since the 'Unix Epoch'. That is the time 00:00:00 UTC on 1 January 1970.

```
dts <- c("2005-10-21 18:47:22","2005-12-24 16:39:58","2005-10-28 07:30:05")
as.POSIXct(dts, format="%Y-%m-%d %H:%M:%S", tz = "GMT")

## [1] "2005-10-21 18:47:22 GMT" "2005-12-24 16:39:58 GMT"
## [3] "2005-10-28 07:30:05 GMT"

dts <- c(1127056501,1104295502,1129233601,1113547501,1119826801)
dts <- as.POSIXct("January 1, 1970", format = "%B %d, %Y", tz="GMT")+dts
dts

## [1] "2005-09-18 15:15:01 GMT" "2004-12-29 04:45:02 GMT"
## [3] "2005-10-13 20:00:01 GMT" "2005-04-15 06:45:01 GMT"
## [5] "2005-06-26 23:00:01 GMT"

#get details from dates
format(dts,"%H")

## [1] "15" "04" "20" "06" "23"
```

Manipulating Dates

Extracting dates information

```
format(dts,"%H")

## [1] "15" "04" "20" "06" "23"

format(dts,"%Y")

## [1] "2005" "2004" "2005" "2005" "2005"

format(dts,"%m")

## [1] "09" "12" "10" "04" "06"
```

Differences between dates

```
difftime(dts[1],dts[2], units = "mins")

## Time difference of 379350 mins

difftime(dts[1],dts[2], units = "days")

## Time difference of 263.4375 days
```

Consistent data

Consistent data = data fit for statistical purposes.
Where:

- Missing data
- Special values
- Errors
- Outliers

Are removed or corrected

Missing data

- Represented by NA in R
- The most common issue you can find in data
- Most operations with data containing NAs will simply return NA
- But some function include the option 'na.rm' to ignore them

```
a <- c(1,2,NA,3)
mean(a)

## [1] NA

mean(a, na.rm = TRUE)

## [1] 2
```

Identifying missing data

```
dfWithNA <- df
dfWithNA[2,3] <- NA
dfWithNA[5,c(1,3)] <- NA
dfWithNA
```

```
##   student_id student_name student_grades
## 1          1         John      Excellent
## 2          2         Mark           <NA>
## 3          3         Dave            Bad
## 4          4         Kate       Very Bad
## 5       <NA>         Anna           <NA>
```

```
complete.cases(dfWithNA)
```

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

```
# Number of NA in each column
apply(dfWithNA, 2,function(x){return(sum(is.na(x))) } )
```

```
##     student_id   student_name student_grades
##              1              0              2
```

Dealing with missing data: Removal

The easiest way to deal with missing data is to simply remove the instances containing them from your data set

```
dfWithoutNA <- na.omit(dfWithNA)
dfWithoutNA <- dfWithNA[complete.cases(dfWithNA),]
dfWithoutNA
```

```
##   student_id student_name student_grades
## 1          1         John      Excellent
## 3          3         Dave            Bad
## 4          4         Kate       Very Bad
```

Dealing with missing data: Imputation

Estimating or deriving values for fields where data is missing.

- Using the mean value of the feature

```
x <- c(NA,2,3,4,5,4,NA,5,1,NA)
x

##  [1] NA  2  3  4  5  4 NA  5  1 NA

x[is.na(x)] <- mean(x, na.rm = TRUE)
x

##  [1] 3.428571 2.000000 3.000000 4.000000 5.000000 4.000000 3.428571
##  [8] 5.000000 1.000000 3.428571
```

- A random value from that feature

```
x <- c(NA,"a","b","c","d",NA,"z",NA)
x

## [1] NA  "a" "b" "c" "d" NA  "z" NA

xNotNA <- x[!is.na(x)]
x[is.na(x)] <- xNotNA[sample(1:length(xNotNA), size = sum(is.na(x)))]
x

## [1] "z" "a" "b" "c" "d" "a" "z" "b"
```

Dealing with missing data: Imputation

Advanced methods:

- Regression methods
- K-Nearest Neighbour

R packages for imputation

- Hmisc
- mi
- mice
- VIM

Outliers

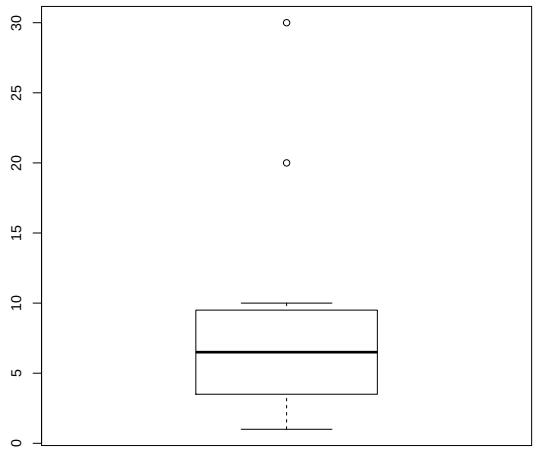
Outliers: observations which appear inconsistent with that set of data.

- Difficult to differentiate incorrect from extraordinary values
- May require domain expert input
- Sources:
  - Instrument failure
  - Human error (data entry, labelling error...)

```
x <- c(1:10, 20, 30)
boxplot.stats(x)$out

## [1] 20 30

boxplot(x)
```



Data consistency

Countless number of cases where you will need to transform some column:

- Correct values: age>0, distances>0...
- Correct aggrgation of values:

```
houseChores <- data.frame(person = c("Jack", "Jane", "John"),
                          cleaning = c(2,0,4),
                          cooking = c(0,0,5),
                          laundry = c(0,6,0),
                          total = c(2,6,0) )
houseChores
```

```
##   person cleaning cooking laundry total
## 1   Jack        2       0       0     2
## 2   Jane        0       0       6     6
## 3   John        4       5       0     0
```

```
houseChores$total == houseChores$cleaning +
                     houseChores$cooking +
                     houseChores$laundry
```

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

Data consistency

```
sizes <- c("1.5 m", "122 cm", "20 inch")
sizes <- as.data.frame(matrix(unlist(strsplit(sizes, " ")), nrow = 3, byrow = TRUE))
colnames(sizes) <- c("measure","unit")
str(sizes)

## 'data.frame': 3 obs. of  2 variables:
##  $ measure: Factor w/ 3 levels "1.5","122","20": 1 2 3
##  $ unit   : Factor w/ 3 levels "cm","inch","m": 3 1 2

sizes$measure <- as.numeric(as.character(sizes$measure))
sizes

##   measure unit
## 1     1.5    m
## 2   122.0   cm
## 3    20.0 inch

sizes$measure[sizes$unit == "m"] <- sizes$measure[sizes$unit == "m"]*100
sizes$measure[sizes$unit == "inch"] <- sizes$measure[sizes$unit == "m"]*2.54
```

Data transformation

Converting data from numerical to categorical:

```
df$gpa <- runif(1:nrow(df),20,90)
df$student_grades <- ifelse(df$gpa>=80,"Excellent",
                            ifelse(df$gpa>=70,"Very Good",
                                   ifelse(df$gpa>=50,"Good",
                                          ifelse(df$gpa>=40,"Bad","Very Bad"))))
df
```

```
##   student_id student_name student_grades      gpa
## 1          1         John      Excellent 80.92944
## 2          2         Mark           Good 50.87671
## 3          3         Dave            Bad 43.91822
## 4          4         Kate       Very Bad 27.47128
## 5          5         Anna       Very Bad 27.97652
```

Data transformation

Making new features from old:

```
measurements <- data.frame(size = c(1.50, 1.95, 1.75, 1.62 ),
                           weight = c(52, 90, 120, 50))
measurements$BMI <- measurements$weight/(measurements$size^2)
measurements
```

```
##   size weight      BMI
## 1 1.50     52 23.11111
## 2 1.95     90 23.66864
## 3 1.75    120 39.18367
## 4 1.62     50 19.05197
```

Feature selection

- Use only the relevant features
- Discard irrelevant features

```
#removing constant features
persons <- data.frame(size = c(1.50, 1.95, 1.75, 1.62, 1.62 ),
                      weight = c(52, 90, 120, 50, 50),
                      age = c(40,40,40,40,40),
                       gender = c("M","M","M","M","M"),
                     body.mass = c(52, 90, 120, 50, 50))
# get for each columns the number of
apply(persons, 2, function(x){return(length(unique(x)))})

##      size    weight       age    gender body.mass
##         4         4         1         1         4

#remove columns that have only one unique value
persons <- persons[, apply(persons, 2,
                           function(x){return(length(unique(x)))}) != 1]
# remove duplicate rows
persons <- persons[duplicated(persons),]
#remove duplicate columns
persons <- persons[,duplicated(t(persons))]
```

Up next

Today's lab in N533

- Data preparation and data cleaning in R
- Loading data
- Detecting NA
- Data transformation

Next week:

- Exploratory data analysis