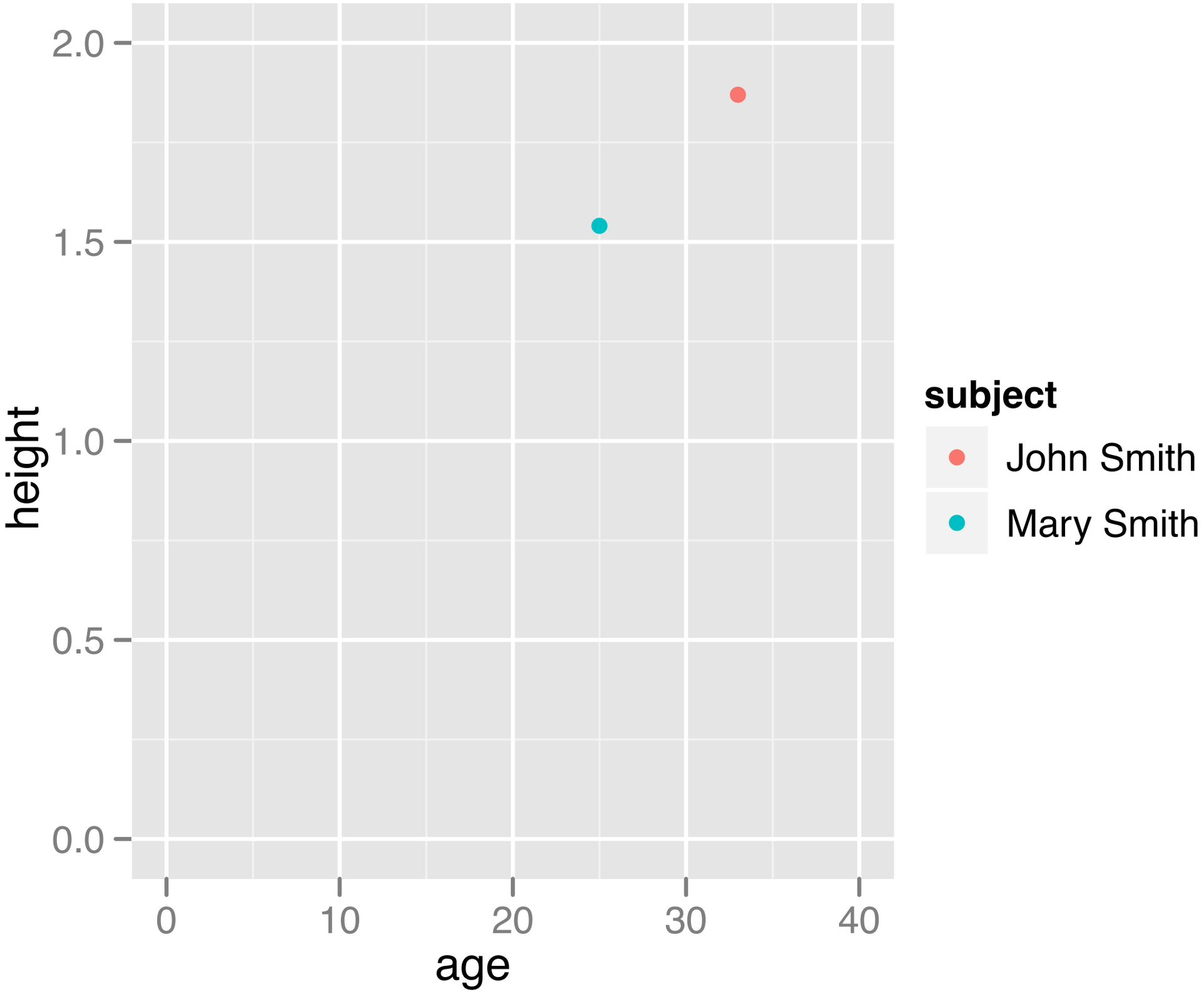
# Stat405

## Tables

Hadley Wickham

Today we will use the reshape2 and xtable packages, and the movies.csv.bz2 dataset.

```
install.packages(c("reshape2", "xtable"))
```


height
2.0
1.5
1.0
0.5
0.0
0
10
20
30
40
age
subject
John Smith
Mary Smith

| | subject | age | height |
|---|---|---|---|
| 1 | John Smith | 33.00 | 1.87 |
| 2 | Mary Smith | 25.00 | 1.54 |

1. Arranging tables (reshape2)
2. Exporting tables from R (xtables)
3. Advanced styling (booktabs)

# Making tables
## (organizing data)

# Recall

Every entry in a data set is either a variable name or a value.

| Variable Name | Value |
|---|---|
| Gender | female |
| Age | 33 |
| Genre | comedy |

Matters for analyzing data, but not for displaying it.

```
# example
library(reshape2)
smiths
```

| subject | age | weight | height |
|---|---|---|---|
| John Smith | 33 | 90 | 1.87 |
| Mary Smith | | | 1.54 |

| subject | variable | value |
|---|---|---|
| John Smith | age | 33 |
| John Smith | weight | 90 |
| John Smith | height | 1.87 |
| Mary Smith | height | 1.54 |

| variable | John Smith | Mary Smith |
|---|---|---|
| age | 33 | |
| weight | 90 | |
| height | 1.87 | 1.54 |

What is the difference between these data frames?

| subject | age | weight | height |
|---|---|---|---|
| John Smith | 33 | 90 | 1.87 |
| Mary Smith | | | 1.54 |

| subject | variable | value |
|---|---|---|
| John Smith | age | 33 |
| John Smith | weight | 90 |
| John Smith | height | 1.87 |
| Mary Smith | height | 1.54 |

| variable | John Smith | Mary Smith |
|---|---|---|
| age | 33 | |
| weight | 90 | |
| height | 1.87 | 1.54 |

Same measurements, different arrangements

| subject | age | weight | height |
|---|---|---|---|
| John Smith | 33 | 90 | 1.87 |
| Mary Smith | | | 1.54 |

| subject | variable | value |
|---|---|---|
| John Smith | age | 33 |
| John Smith | weight | 90 |
| John Smith | height | 1.87 |
| Mary Smith | height | 1.54 |

Same measurements, different arrangements

| variable | John Smith | Mary Smith |
|---|---|---|
| age | 33 | |
| weight | 90 | |
| height | 1.87 | 1.54 |

Values in column names

# Question

How do we rearrange data to be in the format we want?

# melt and dcast

Liquify the data set with melt(), pour it into a new shap with dcast(). Both from reshape package.

The d is for data frame.

We've already studied melt()

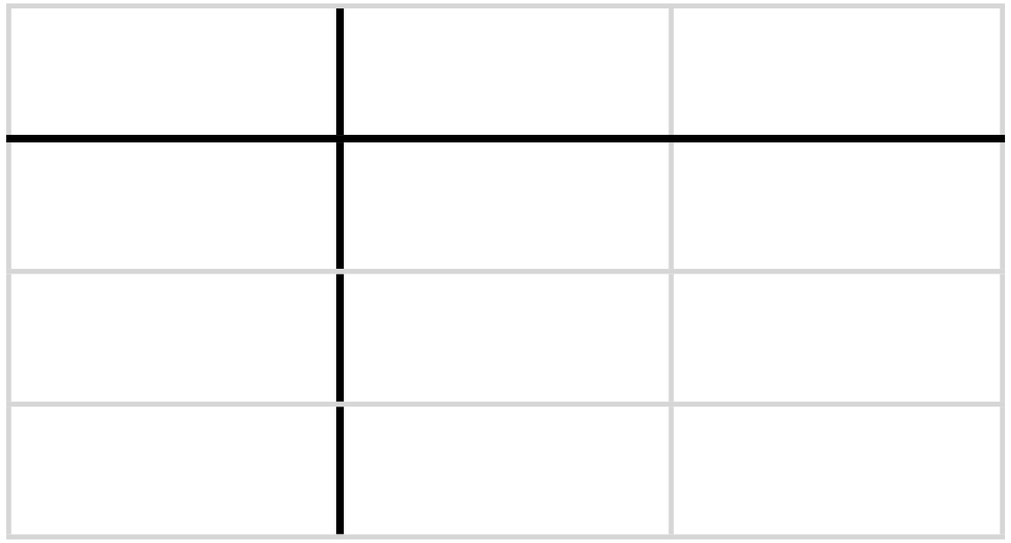
# dcast

Transforms a tidy data frame into a new arrangement. Note: data frame must contain a column named “value.”*

```
m.smiths <- melt(smiths[ , -2])
m.smiths # now tidy
dcast(m.smiths, subject ~ variable)
```

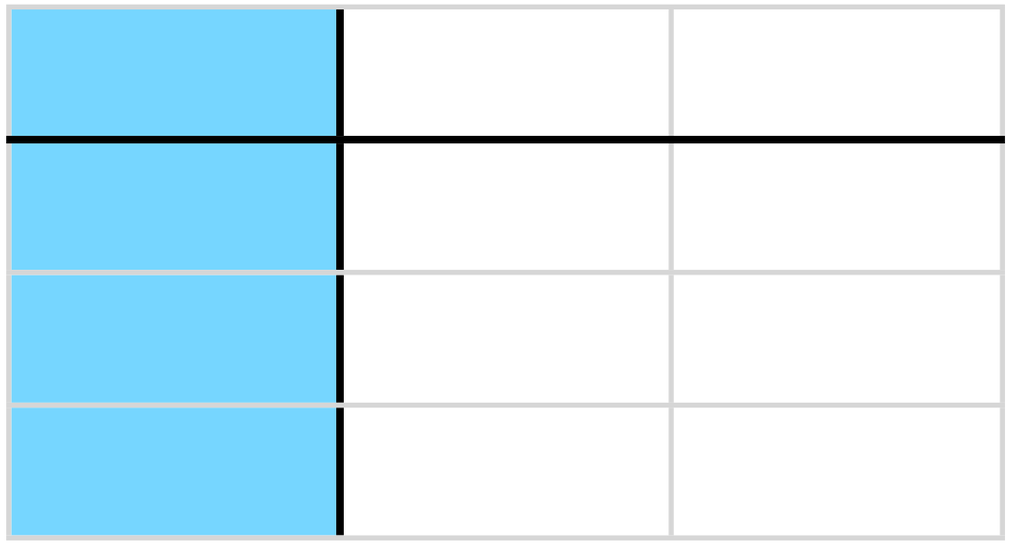
# General strategy

Think of dcast() as making a table



rectangular dataset

```
dcast(m.smiths, subject ~ variable)
```



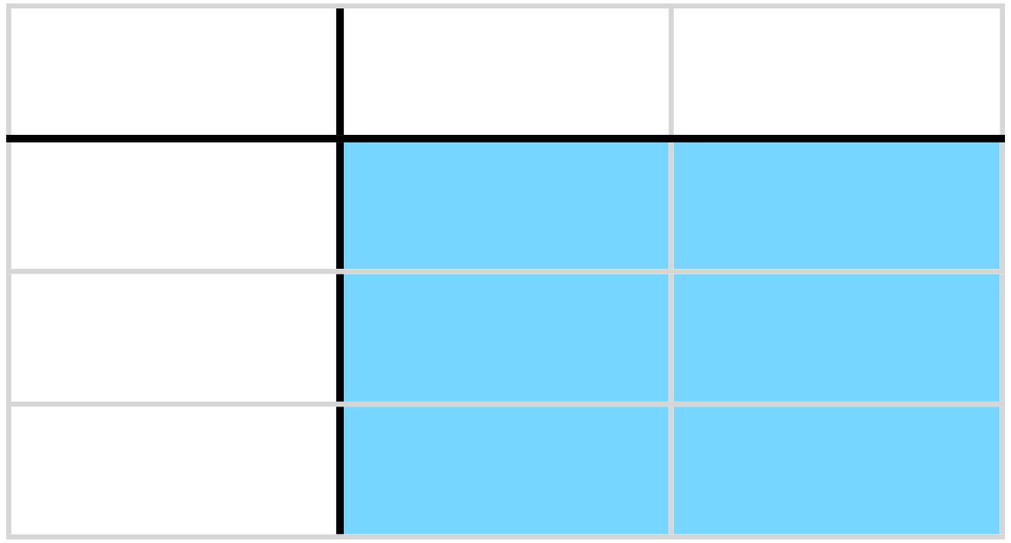
what variable to put in left hand column

```
dcast(m.smiths, subject ~ variable)
```

| | | |
|---|---|---|
| | | |
| | | |
| | | |

What variables to put along the top

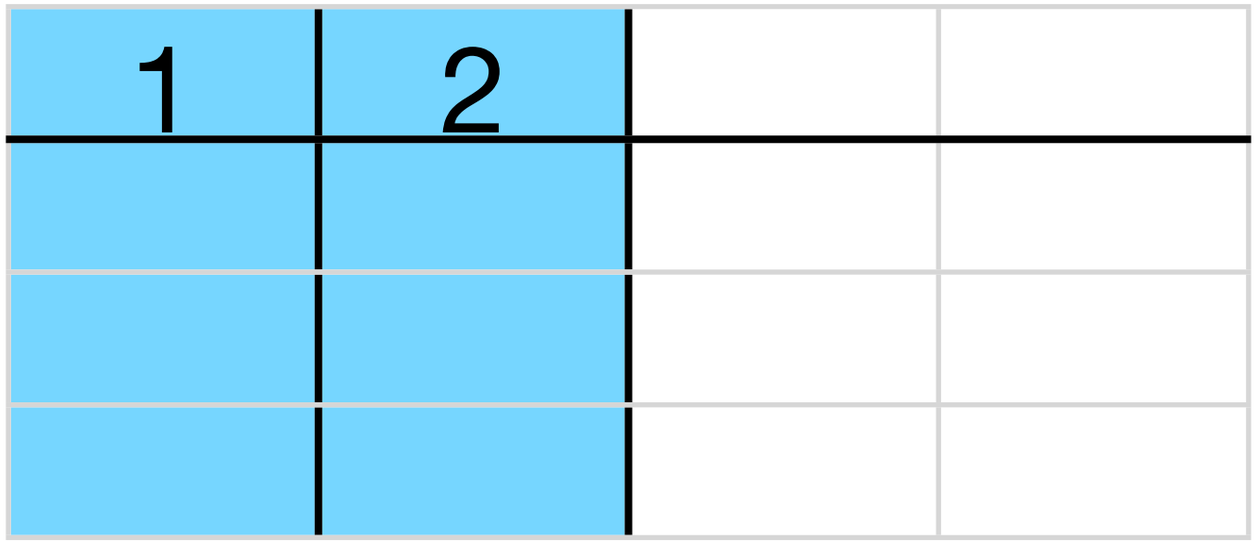
```
dcast(m.smiths, subject ~ variable)
```



Whatever was in the “value” column will get spread across the cells of the table

```
dcast(m.smiths, subject ~ variable)
```

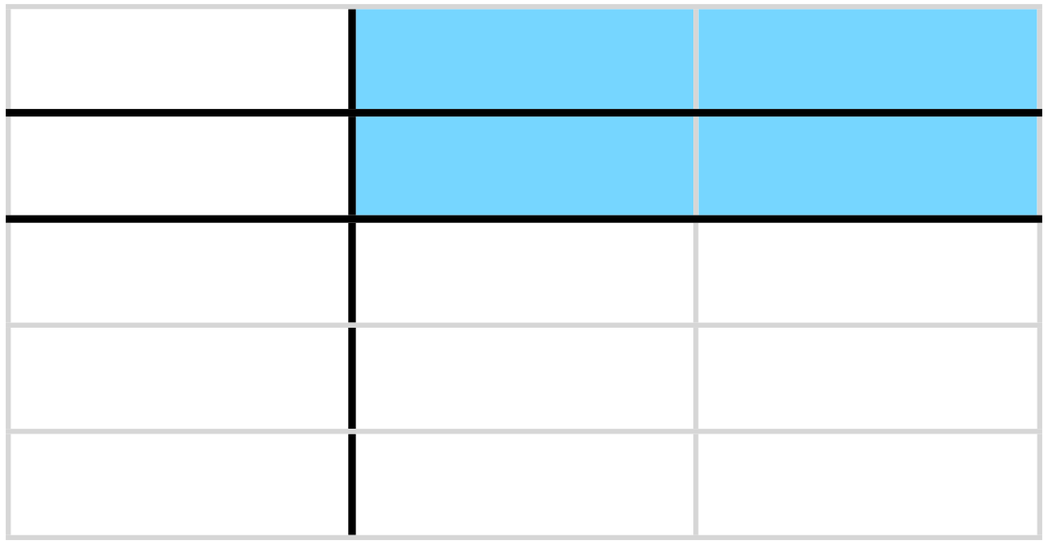
```
m.smiths$gender <- rep(c("male", "female"), 3)
```




left hand column 1

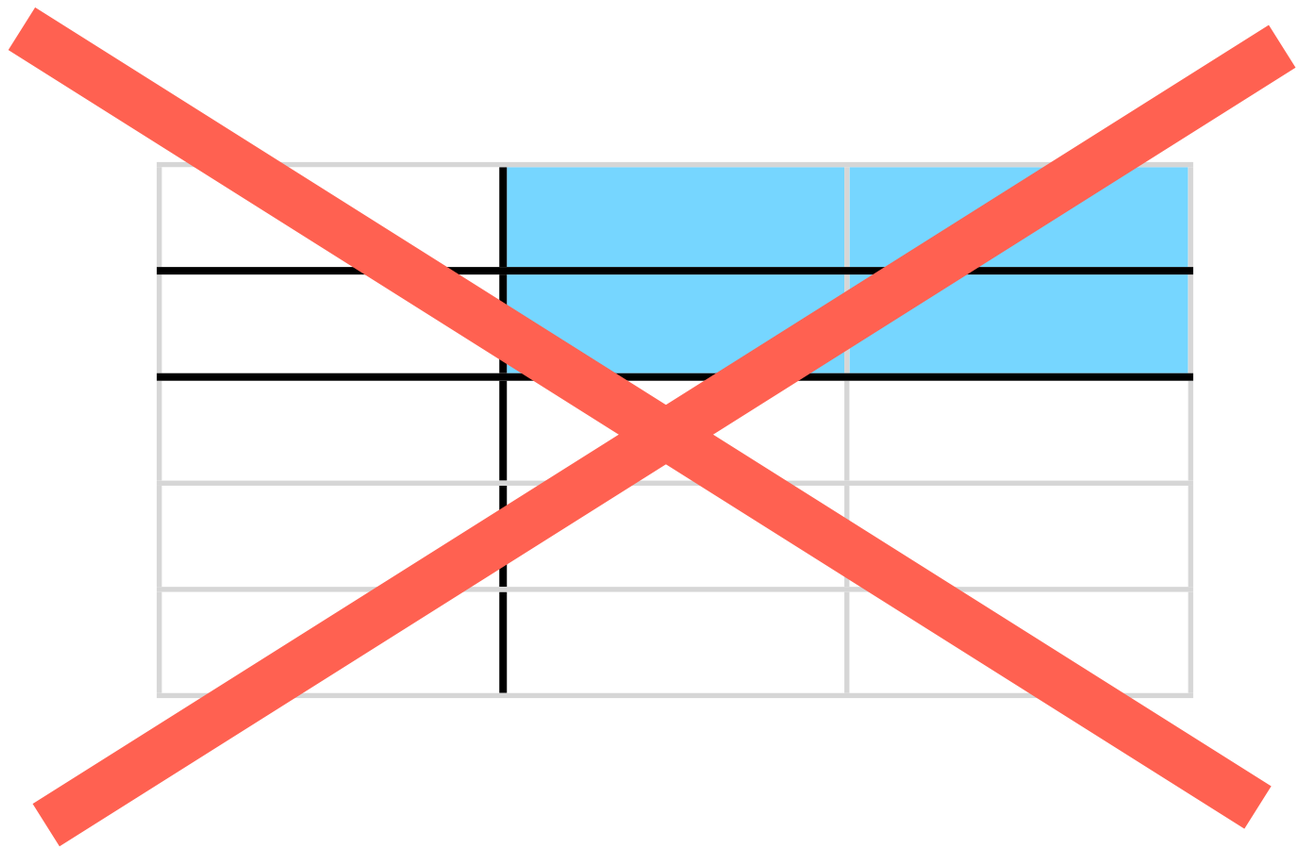
left hand column 2

```
dcast(m.smiths, subject + gender ~ variable)
```



?

```
dcast(m.smiths, subject ~ gender + variable)
```



?

```
dcast(m.smiths, subject ~ gender + variable)
```

| | | | | |
|---|---|---|---|---|
| | | | | |
| | | | | |
| | | | | |

still only one header row: names get combined, each combination gets a column

```
dcast(m.smith, subject ~ gender + variable)
```

# Your turn

Use melt(), dcast() and the smiths data set to create the following tables

| | variable | John Smith | Mary Smith |
|---|---|---|---|
| 1 | age | 33.00 | NA |
| 2 | weight | 90.00 | NA |
| 3 | height | 1.87 | 1.54 |

| | subject | variable | 1.54 | 1.87 |
|---|---|---|---|---|
| 1 | John Smith | time | NA | 1 |
| 2 | John Smith | age | NA | 33 |
| 3 | John Smith | weight | NA | 90 |
| 4 | Mary Smith | time | 1 | NA |
| 5 | Mary Smith | age | NA | NA |
| 6 | Mary Smith | weight | NA | NA |

```
dcast(m.smiths, variable ~ subject)

m.smiths2 <- melt(smiths, id = c("subject",
  "height"))
dcast(m.smiths2, subject + variable ~ height)
```

```
dcast(m.smiths, variable ~ subject)

m.smiths2 <- melt(smiths, id = c("subject",
  "height"))
dcast(m.smiths2, subject + variable ~ height)
```

How is m.smiths2 different than m.smiths?

# Your turn

Use m.smiths2 to make the table below.
What happens?

| | height | John Smith | Mary Smith |
|---|---|---|---|
| 1 | 1.54 | ? | ? |
| 2 | 1.87 | ? | ? |

# Aggregating

Q: Often reshaping a data set will lead to two values being put into the same cell. How do we combine them?

For example,

| game | player | variable | value |
|---|---|---|---|
| 1 | Yao Ming | shot attempts | 16 |
| 2 | Yao Ming | shot attempts | 12 |
| 2 | Jordan Hill | shot attempts | 9 |

| player | shot attempts |
|---|---|
| Jordan Hill | 9 |
| Yao Ming | ? |

# Aggregating

A: However you want.

R function to use to combine the values

```
dcast(data, left ~ top, agg.method)
```

| player | shot attempts |
|---|---|
| Jordan Hill | 9 |
| Yao Ming | **28** |

```
df <- data.frame(game = c(1,2,2),
   player = c("Yao Ming", "Yao Ming",
   "Jordan Hill"), variable = c("shot
attempts",
   "shot attempts", "shot attempts"),
      value = c(16, 12, 9))

dcast(df, player ~ variable, sum)
```

# Useful aggregating functions

| | |
|---|---|
| length | (default) number of times the combination appeared. This creates counts |
| sum | total number of all values in that cell |
| mean | average number of all values in that cell |

* remember to use na.rm = T

# Movies data

Ratings for 115,000 films collected by the internet movie data base at www.imdb.com

```
movies <- read.csv("movies2.csv.bz2",
stringsAsFactors = F)
```

# Your turn

Use the movie data set to create a table that shows the average rating, length, and budget of movies made in each year

(hint 1: begin by subsetting down to just the variables you are interested in)

(hint 2: years in rows)

```
# reducing to variables of interest
movies <- movies[ , c(2:5)]

# putting rating, length, and budget into a
single
  column to cast from
movies <- melt(movies, id = "year")

dcast(movies, year ~ variable, mean, na.rm = T)
```

# Margins

Its often useful to see the total for each row and column. These totals are known as margins or marginal distributions.

| | Art Degree | Science Degree | Total |
|---|---|---|---|
| Boys | 25 | 50 | 75 |
| Girls | 55 | 20 | 75 |
| Total | 80 | 70 | 150 |

# Margins

To add a margin column:

```
dcast(data, left ~ top, margins = "name of last
variable  on left side of ~")
```

To add a margin row:

```
dcast(data, left ~ top, margins = "name of last
variable on right side of ~")
```

To add both:

```
dcast(data, left ~ top, margins = c("name1",
"name2"))
```

# Exporting tables from R

# Problem

Data analysis done in R

Tables published in latex

# Problem


Data analysis done in R
you type up tables by hand
Tables published in latex

# Solution

```
library(xtable)

# xtable converts objects in R to code to
# their equivalent latex code
```

```
head(smiths)
print(xtable(smiths))
print(xtable(smiths), floating = FALSE)
```

```
     subject  time   age  weight  height
1 John Smith     1    33      90    1.87
2 Mary Smith     1    NA      NA    1.54
```

| | subject | time | age | weight | height |
|---|---|---|---|---|---|
| 1 | John Smith | 1 | 33.00 | 90.00 | 1.87 |
| 2 | Mary Smith | 1 | | | 1.54 |

# Even better

Add `usepackage(booktabs)` at the top of your latex file and then use:

```
xtable <- function(x, file = "", ...){
 table <- xtable::xtable(x, ...)
 print(table, floating = F, hline.after = NULL,
   add.to.row = list(pos = list(-1,0, nrow(x)),
   command = c('\\toprule\n ','\\midrule\n  ','\
\bottomrule\n')),
   include.rownames = FALSE, file = file
 )
}
```

| | subject | time | age | weight | height |
|---|---|---|---|---|---|
| 1 | John Smith | 1 | 33.00 | 90.00 | 1.87 |
| 2 | Mary Smith | 1 | | | 1.54 |

vs.

| subject | time | age | weight | height |
|---|---|---|---|---|
| John Smith | 1 | 33.00 | 90.00 | 1.87 |
| Mary Smith | 1 | | | 1.54 |

# Column Alignment

```
xtable(smiths, align = "|cc|cccc|")
```

| | subject | time | age | weight | height |
|---|---|---|---|---|---|
| 1 | John Smith | 1 | 33.00 | 90.00 | 1.87 |
| 2 | Mary Smith | 1 | | | 1.54 |

Remember that variable names like "`year_2010`" won't print out in latex. You'll have to change them to "`year\_2010`" first.

For more examples of xtable, visit:

cran.r-project.org/web/packages/xtable/vignettes/xtableGallery.pd

# Your turn

Create latex code for your movies table table. Give the table an appropriate caption and alignment.

# Advanced table styling

# 6 rules for pretty tables

1. Never use vertical lines
2. Never use double lines
3. Put units in the column heading, not in body of table
4. Always precede a decimal point by a digit
5. Never use “ditto” signs to repeat a previous value
6. Use signficant digits consistently within columns

From http://www.ctan.org/tex-archive/macros/latex/contrib/booktabs/booktabs.pdf

# When to use a table

Use a table instead of a graphic if:

a) there is only a small amount of data, or

b) precision is important

# Significant digits

Pick an appropriate amount of significant digits (at most 4 or 5)

Use `signif( )` to round data to that amount

# Align decimals

Ensure that decimal points line up so that differences in order of magnitude are easy to spot.

# Include captions

Always include a caption.

Its difficult to spot patterns in tables if you don’t know what to look for

Captions should explain what data the figure shows and highlight the important finding.

# Ordering

Unless your table is ordered chronologically (i.e, by year), order rows and columns by size, (small values to large or vice versa).

This makes patterns easier to see.