Analysing

Introduction to Data Science with R

www.therbootcamp.com

@therbootcamp

October 2018

Where you're at...

1 - Loaded packages (like `tidyverse`) with `library()`

2 - Loaded external files as a new dataframe

3 - Explore dataframes

4 - Calculate descriptive statistics on specific columns

5 - Wrangle

```
- Change column names
- Add new columns
- Filter
- Join multiple dataframes
- Change data format (wide v. long)
```

What's next?... **Analysing!**

```r
# Load libraries

library(tidyverse)

# Read external file

baslers <- read_csv(file = "data/baslers.txt")

# Explore data

View(baslers)   # Open in new window
dim(baslers)    # Show number of rows and columns
names(baslers)  # Show names

# Calculate descriptives on named colums

mean(baslers$age)   # What is the mean age?
table(baslers$sex)  # How many of each sex?

# Wrangle

baselers <- baselers %>%
  rename(age_y = age,            # New names
         salary = income) %>%
  mutate(age_m = age * 12) %>% # Create new column
  filter(sex == "male")        # filter rows...
```

What is analysing?

Create Groups

Group data by certain variables

- For all males (`sex == "male"`)
- For all people in placebo conditoin (`condition == "placebo"`)

Calculate summaries

- Count number of cases
- Calculate mean of age (`mean(age)`)
- Calculate number of events (`sum(events)`)

Bonus: Statistical Analyses

- Simple hypothesis tests (t-test, correlation test)
- Generalised linear model (regression, ANOVA)

Raw data (First 5 out of 1,000 rows)

id	sex	education	income	happiness
1	male	SEK_III	6300	5
2	male	obligatory_school	10900	7
3	female	SEK_III	5100	7
4	male	SEK_III	4200	7
5	male	SEK_III	4000	5

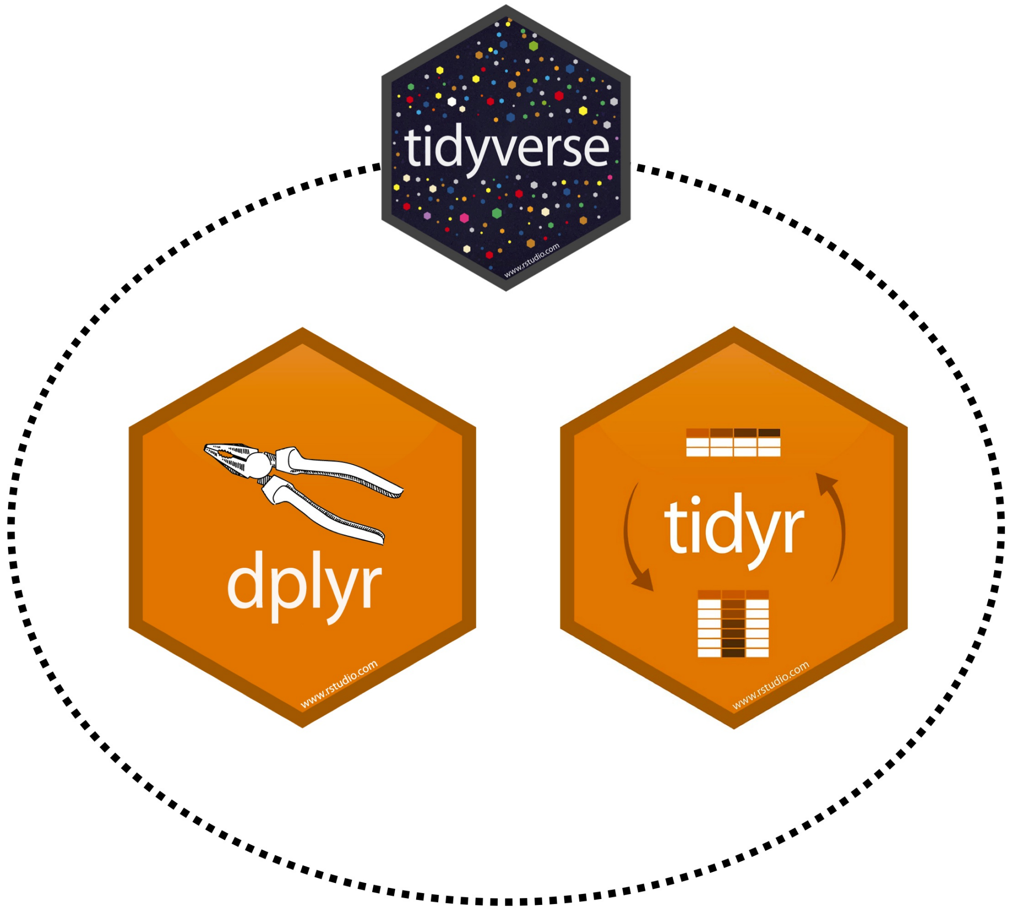
Aggregated data

education	sex	N	Inc_mean	Hap_mean
apprenticeship	female	2168	7663.0	6.9
apprenticeship	male	1818	7388.9	6.9
obligatory_school	female	714	7746.1	6.9
obligatory_school	male	525	7293.7	6.8
SEK_II	female	469	7385.0	6.9
SEK_II	male	272	7254.7	6.9

dplyr

To calculate grouped summary analyses, we will use `dplyr` (again!)

```
# Load packages individually
# install.packages('dplyr')
library(dplyr)
# Or just use the tidyverse!
# install.packages('tidyverse')
library(tidyverse)
```



The Pipe! %>%

dplyr makes extensive use of a new operator called the "Pipe" %>%

Read the "Pipe" %>% as "And Then..."

```
# Start with data
data %>% # AND THEN...

DO_SOMETHING %>% # AND THEN...

DO_SOMETHING %>% # AND THEN...

DO_SOMETHING %>% # AND THEN...
```

This is not a pipe (but %>% is!)

summarise()

Use summarise() to create new columns of **summary statistics**

```
df %>%
  summarise(
    NAME = SUMMARY_FUN(A),
    NAME = SUMMARY_FUN(B)
  )
```

Summary functions

Function	Purpose
n()	Number of cases in each group
mean(), median(), max(), min() sum()	Summary stats

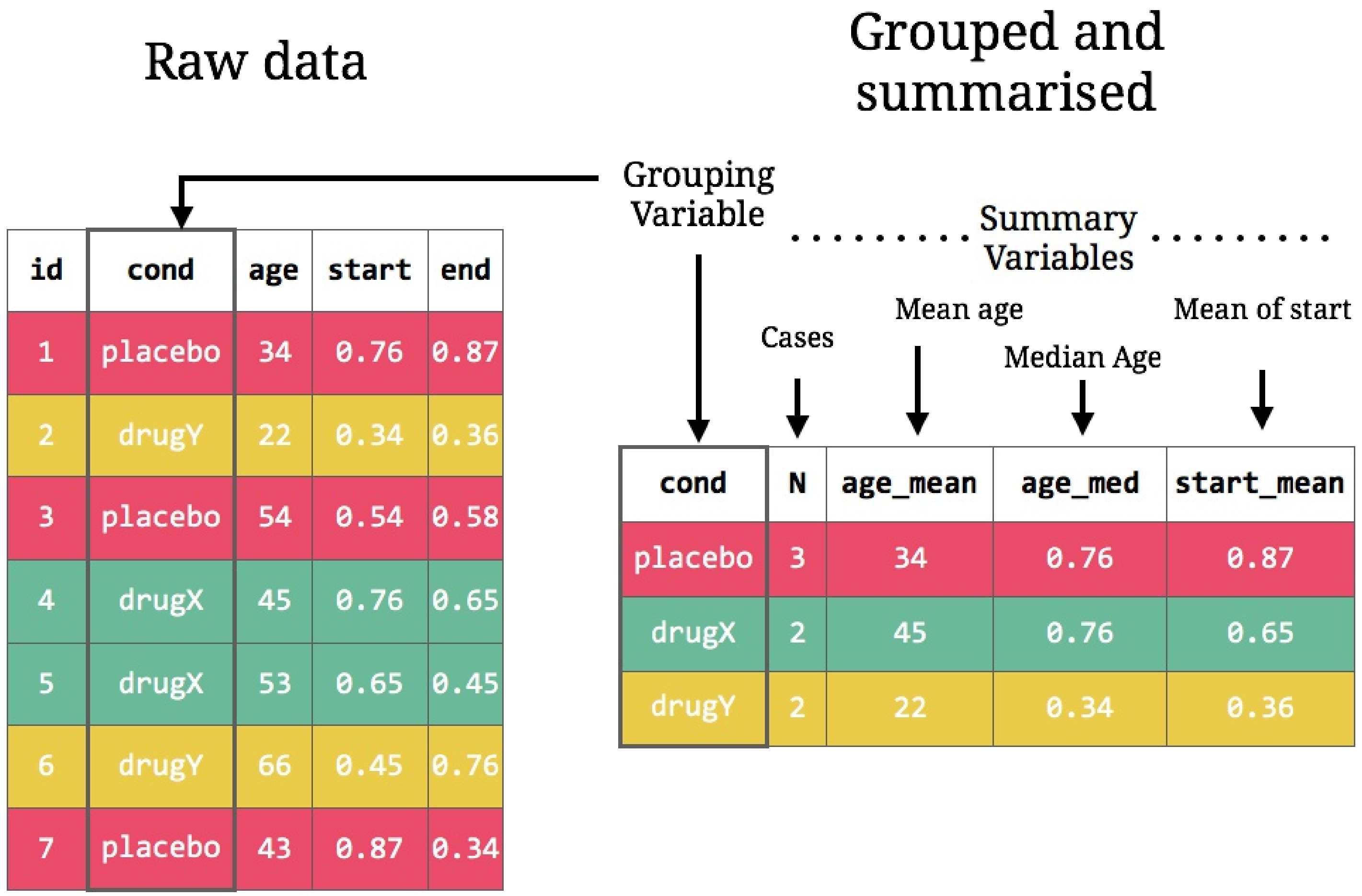
```
# Calculate summary statistics
baselers %>%
  summarise(
    N = n(),
    age_mean = mean(age),
    height_median = median(height),
    children_max = max(children, na.rm = TRUE)
  )
```

```
## # A tibble: 1 x 4
##       N age_mean height_median children_max
##   <int>    <dbl>         <dbl>        <dbl>
## 1 10000     44.6          171.            6
```

The result of summarise() will always be a tibble!

Important You can only include summary functions that return a single value (i.e.; can't use table())

Grouped Aggregation



Raw data

id	cond	age	start	end
1	placebo	34	0.76	0.87
2	drugY	22	0.34	0.36
3	placebo	54	0.54	0.58
4	drugX	45	0.76	0.65
5	drugX	53	0.65	0.45
6	drugY	66	0.45	0.76
7	placebo	43	0.87	0.34

Grouped and summarised

cond	N	age_mean	age_med	start_mean
placebo	3	34	0.76	0.87
drugX	2	45	0.76	0.65
drugY	2	22	0.34	0.36

group_by(), summarise()

Use group_by() to **group data** according to one or more columns

After grouping data, use summarise() to **calculate summary statistics** across groups of data

Statistical functions

Function	Purpose
n()	Number of cases in each group
mean(), median(), max(), min() sum()	Summary stats

```
# Group data by arm, and calculate many
#  summary statistics
baselers %>%
  group_by(sex) %>%
  summarise(
    N = n(),
    age_mean = mean(age),
    height_median = median(height),
    children_max = max(children)
  )
```

```
## # A tibble: 2 x 5
##   sex        N age_mean height_median children_max
##   <chr>  <int>    <dbl>         <dbl>        <dbl>
## 1 female  5000     45.4           164            6
## 2 male    5000     43.8           178.           6
```

Combine wrangling with analysing

You can easily combine multiple wrangling (filtering, slicing, renaming) and analysing operations at once!

Just use the pipe %>%

```
baselers %>%
  filter(sex == "male" & children > 0) %>%  # male parents only
  group_by(confession) %>%
  summarise(
    N = n(),
    age_mean = mean(age),
    income_median = median(income, na.rm = TRUE)
  )
```

```
## # A tibble: 6 x 4
##   confession                N age_mean income_median
##   <chr>                 <int>    <dbl>         <dbl>
## 1 catholic               1401     44.0          7100
## 2 confessionless         1125     43.8          7100
## 3 evangelical-reformed    925     43.9          7200
## 4 muslim                  155     41.5          6800
## 5 other                   247     44.0          6900
## 6 <NA>                    703     43.5          7000
```

Quiz 1

Here is part of the baselers dataframe

```
baselers %>%
  select(sex, fasnacht, age, income) %>%
  slice(1:5)
```

```
## # A tibble: 5 x 4
##   sex    fasnacht   age income
##   <chr>  <chr>    <dbl>  <dbl>
## 1 male   no          44   6300
## 2 male   no          65  10900
## 3 female no          31   5100
## 4 male   no          27   4200
## 5 male   no          24   4000
```

How do I calculate the following table?

```
## # A tibble: 2 x 4
##   fasnacht     N age_mean income_mean
##   <chr>    <int>    <dbl>       <dbl>
## 1 no        9706     44.6       7527.
## 2 yes        294     45.3       7692.
```

Quiz 1

Here is part of the baselers dataframe

```
baselers %>%
  select(sex, fasnacht, age, income) %>%
  slice(1:5)
```

```
## # A tibble: 5 x 4
##   sex    fasnacht   age income
##   <chr>  <chr>    <dbl>  <dbl>
## 1 male   no          44   6300
## 2 male   no          65  10900
## 3 female no          31   5100
## 4 male   no          27   4200
## 5 male   no          24   4000
```

How do I calculate the following table?

```
baselers %>%
  group_by(fasnacht) %>%
  summarise(
    N = n(),
    age_mean = mean(age),
    income_mean = mean(income, na.rm = TRUE)
  )
```

```
## # A tibble: 2 x 4
##   fasnacht     N age_mean income_mean
##   <chr>    <int>    <dbl>       <dbl>
## 1 no        9706     44.6       7527.
## 2 yes        294     45.3       7692.
```

Quiz 2

Here is part of the baselers dataframe

```
baselers %>%
  select(sex, fasnacht, age, income) %>%
  slice(1:5)
```

```
## # A tibble: 5 x 4
##   sex    fasnacht   age income
##   <chr>  <chr>    <dbl>  <dbl>
## 1 male   no          44   6300
## 2 male   no          65  10900
## 3 female no          31   5100
## 4 male   no          27   4200
## 5 male   no          24   4000
```

How do I calculate the following table?

```
## # A tibble: 4 x 5
## # Groups:   fasnacht [?]
##   fasnacht sex        N age_mean income_mean
##   <chr>    <chr>  <int>    <dbl>       <dbl>
## 1 no       female  4886     45.4       7646.
## 2 no       male    4820     43.8       7407.
## 3 yes      female   114     46.4       7829.
## 4 yes      male     180     44.6       7602
```

Quiz 2

Here is part of the baselers dataframe

```
baselers %>%
  select(sex, fasnacht, age, income) %>%
  slice(1:5)
```

```
## # A tibble: 5 x 4
##   sex    fasnacht   age income
##   <chr>  <chr>    <dbl>  <dbl>
## 1 male   no          44   6300
## 2 male   no          65  10900
## 3 female no          31   5100
## 4 male   no          27   4200
## 5 male   no          24   4000
```

How do I calculate the following table?

```
baselers %>%
  group_by(fasnacht, sex) %>%
  summarise(
    N = n(),
    age_mean = mean(age),
    income_mean = mean(income, na.rm = TRUE)
  )
```

```
## # A tibble: 4 x 5
## # Groups:   fasnacht [?]
##   fasnacht sex        N age_mean income_mean
##   <chr>    <chr>  <int>    <dbl>       <dbl>
## 1 no       female  4886     45.4       7646.
## 2 no       male    4820     43.8       7407.
## 3 yes      female   114     46.4       7829.
## 4 yes      male     180     44.6       7602
```

Quiz 3

Here is part of the baselers dataframe

```
baselers %>%
  select(sex, fasnacht, age, income) %>%
  slice(1:5)
```

```
## # A tibble: 5 x 4
##   sex    fasnacht   age income
##   <chr>  <chr>    <dbl>  <dbl>
## 1 male   no          44   6300
## 2 male   no          65  10900
## 3 female no          31   5100
## 4 male   no          27   4200
## 5 male   no          24   4000
```

How do I calculate the following table?

```
## # A tibble: 2 x 5
## # Groups:   fasnacht [?]
##   fasnacht sex       N age_mean income_mean
##   <chr>    <chr> <int>    <dbl>       <dbl>
## 1 no       male   4820     43.8       7407.
## 2 yes      male    180     44.6       7602
```

Quiz 3

Here is part of the baselers dataframe

```
baselers %>%
  select(sex, fasnacht, age, income) %>%
  slice(1:5)
```

```
## # A tibble: 5 x 4
##   sex    fasnacht   age income
##   <chr>  <chr>    <dbl>  <dbl>
## 1 male   no          44   6300
## 2 male   no          65  10900
## 3 female no          31   5100
## 4 male   no          27   4200
## 5 male   no          24   4000
```

How do I calculate the following table?

```
baselers %>%
  filter(sex == "male") %>%     # male patients only
  group_by(fasnacht, sex) %>%
  summarise(
    N = n(),
    age_mean = mean(age),
    income_mean = mean(income, na.rm = TRUE)
  )
```

```
## # A tibble: 2 x 5
## # Groups:   fasnacht [?]
##   fasnacht sex       N age_mean income_mean
##   <chr>    <chr> <int>    <dbl>       <dbl>
## 1 no       male   4820     43.8       7407.
## 2 yes      male    180     44.6       7602
```

Quiz 4

Here is part of the baselers dataframe

```
baselers %>%
  select(sex, fasnacht, age, income) %>%
  slice(1:5)
```

```
## # A tibble: 5 x 4
##   sex    fasnacht   age income
##   <chr>  <chr>    <dbl>  <dbl>
## 1 male   no          44   6300
## 2 male   no          65  10900
## 3 female no          31   5100
## 4 male   no          27   4200
## 5 male   no          24   4000
```

How do I calculate the following table?

```
## # A tibble: 4 x 3
##   education              N income_mean
##   <chr>              <int>       <dbl>
## 1 SEK_III             4034       7555.
## 2 obligatory_school   1239       7551.
## 3 apprenticeship      3986       7538.
## 4 SEK_II               741       7338.
```

Quiz 4

Here is part of the baselers dataframe

```
baselers %>%
  select(sex, fasnacht, age, income) %>%
  slice(1:5)
```

```
## # A tibble: 5 x 4
##   sex    fasnacht   age income
##   <chr>  <chr>    <dbl>  <dbl>
## 1 male   no          44   6300
## 2 male   no          65  10900
## 3 female no          31   5100
## 4 male   no          27   4200
## 5 male   no          24   4000
```

How do I calculate the following table?

```
baselers %>%
  group_by(education) %>%
  summarise(
    N = n(),
    income_mean = mean(income, na.rm = TRUE)
  ) %>%
  arrange(desc(income_mean))
```

```
## # A tibble: 4 x 3
##   education            N income_mean
##   <chr>            <int>       <dbl>
## 1 SEK_III           4034       7555.
## 2 obligatory_school 1239       7551.
## 3 apprenticeship    3986       7538.
## 4 SEK_II             741       7338.
```

What have we not covered yet? Statistics!

Statistical functions (almost) always require two key arguments

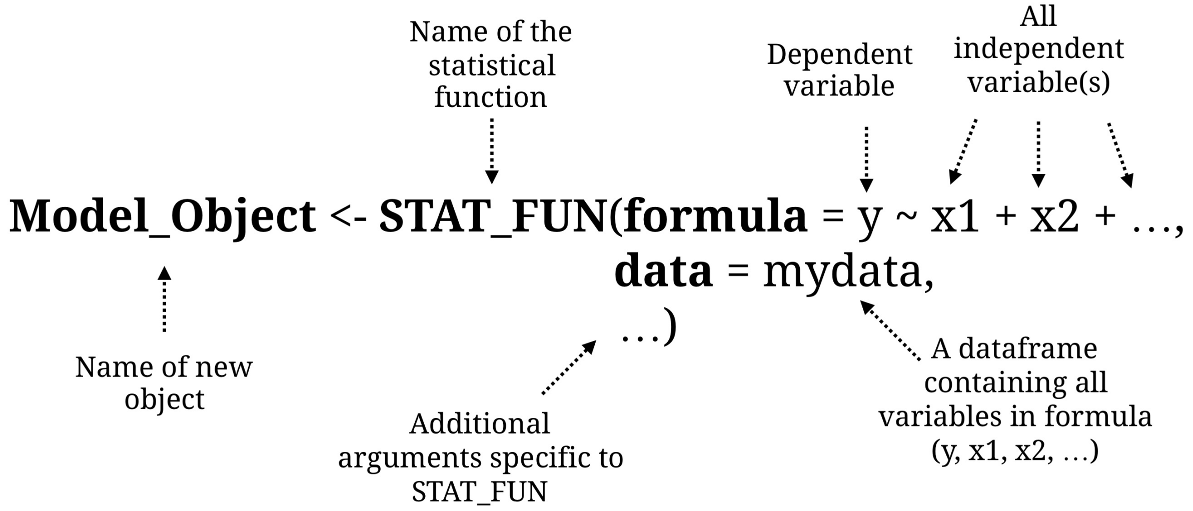
`data`	A dataframe
`formula`	A formula specifying variables in the model

A **formula** specifies a **dependent** variable (y) as a function of one or more **independent** variables (x1, x2, …) in the form:

formula = y ~ x1 + x2 +…

How to create a statistical object:

```
# Example: Create regression object (my_glm)
my_glm <- glm(formula = income ~ age + height,
              data = baselers)
```



Name of the statistical function

Dependent variable

All independent variable(s)

Model_Object <- **STAT_FUN**(**formula** = y ~ x1 + x2 + …,
data = mydata,
…)

Name of new object

Additional arguments specific to STAT_FUN

A dataframe containing all variables in formula (y, x1, x2, …)

Simple hypothesis tests

All of the basic **one and two sample hypothesis tests** are included in the `stats` package.

These tests take either a **formula** for the argument `formula`, or **individual vectors** for the arguments `x`, and `y`

Hypothesis Test	R Function
t-test	`t.test()`
Correlation Test	`cor.test()`
Chi-Square Test	`chisq.test()`

t-test with `t.test()`

```r
# 2-sample t-test
t.test(formula = income ~ sex,
       data = baselers)
```

```
##
##     Welch Two Sample t-test
##
## data:  income by sex
## t = 4, df = 8500, p-value = 6e-05
## alternative hypothesis: true difference in means is not
## 95 percent confidence interval:
##  120.6 352.2
## sample estimates:
## mean in group female   mean in group male
##                 7650                 7414
```

Regression with glm(), lm()

How to **create a regression model** predicting, e.g., how much money people spend on food as a function of income?

Part of the baselers dataframe:

food	income	happiness
610	6300	5
1550	10900	7
720	5100	7
680	4200	7
260	4000	5

Generalized regression with glm()

```
# food (y) on income (x1) and happiness (x2)
food_glm <- glm(formula = food ~ income + happiness,
                data = baselers)

# Print food_glm
food_glm
```

```
##
## Call:  glm(formula = food ~ income + happiness, data = baselers)
##
## Coefficients:
## (Intercept)       income    happiness
##    -302.089        0.101       52.205
##
## Degrees of Freedom: 8509 Total (i.e. Null);  8507 Residual
##   (1490 observations deleted due to missingness)
## Null Deviance:       1.27e+09
## Residual Deviance: 6.06e+08     AIC: 119000
```

Exploring statistical objects

Explore statistical objects using **generic** functions such as `print()`, `summary()`, `predict()` and `plot()`.

Generic functions different things depending on the **class label** of the object.

```
# Create statistical object
obj <- STAT_FUN(formula = ...,
                data = ...)

names(obj)        # Elements
print(obj)        # Print
summary(obj)      # Summary
plot(obj)         # Plotting
predict(obj, ..) # Predict
```

```
# Create a glm object
my_glm <- glm(formula = income ~ happiness + age,
              data = baselers)


summary(my_glm)
```

```
## 
## Call:
## glm(formula = income ~ happiness + age, data = baselers)
## 
## Deviance Residuals: 
##    Min       1Q   Median       3Q      Max  
##  -4045     -835        3      814     4899  
## 
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)    
## (Intercept) 1575.497     94.363   16.70  < 2e-16 ***
## happiness   -100.431     12.520   -8.02  1.2e-15 ***
## age          149.312      0.815  183.31  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

tidy()

The `tidy()` function from the `broom` package **converts** the most important results of many statistical object like "glm" to a **data frame**.

```
# install and load broom
install.packages('broom')
library(broom)
```

```
# Original printout
my_glm
```

```
## 
## Call:  glm(formula = income ~ happiness + age, data = baselers
## 
## Coefficients:
## (Intercept)    happiness          age  
##        1575         -100          149  
## 
## Degrees of Freedom: 8509 Total (i.e. Null);  8507 Residual
##   (1490 observations deleted due to missingness)
## Null Deviance:       6.33e+10 
## Residual Deviance: 1.28e+10    AIC: 145000
```

```
# Tidy printout
tidy(my_glm)
```

```
## # A tibble: 3 x 5
##   term        estimate std.error statistic  p.value
##   <chr>          <dbl>     <dbl>     <dbl>    <dbl>
## 1 (Intercept)    1575.    94.4       16.7  1.33e-61
## 2 happiness      -100.    12.5       -8.02 1.18e-15
## 3 age             149.     0.815    183.   0.
```

Summary

1 - To calculate summary statistics across all rows, use `summarise()`.

2 - To calculate grouped summary statistics, use `group_by()` and then `summarise()`.

3 - "Keep the pipe `%>%` going" to continue working with your data frame.

4 - You can always do wrangling operations (`filter()`, `rename()`) before (or after!) aggregating.

5 - Statistical functions (like `glm()`, `t.test()`) require `data` and `formula` arguments

```
# Assign result to baslers_agg

baslers_agg <- baselers %>%

  # Change column names with rename()
  rename(age_years = age,
         weight_kg = weight)  %>% # PIPE!

  # Select specific rows with filter()
  filter(age_years < 40)  %>% # PIPE!

  # Create new columns witb mutate()
  mutate(debt_ratio = debt / income)  %>% # PIPE!

  # Group data with group_by()
  group_by(sex) %>% # PIPE!

  # Calculate summary statistics with summarise()
  summarise(income_mean = mean(income),
            debt_mean = mean(debt),
            dr_mean = mean(dr))
```

Practical

Link to practical