06 - Linear models

Data Science with R · Summer 2021

Uli Niemann · Knowledge Management & Discovery Lab

https://brain.cs.uni-magdeburg.de/kmd/DataSciR/

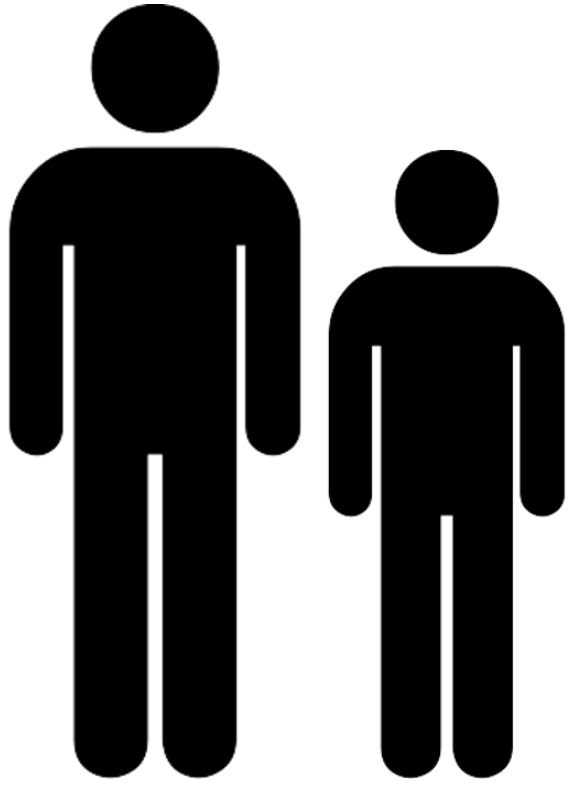
Photo courtesy of Ulrich Arendt

Bivariate relationships

Example data: **father-son dataset** (1078 Measurements of the height of a father and his son.)

```
library(tidyverse)
father_son <- UsingR::father.son %>%
  as_tibble() %>%
  mutate(across(everything(), ~ . * 2.54)) # convert from inch to cm
father_son
```

```
## # A tibble: 1,078 x 2
##    fheight sheight
##      <dbl>   <dbl>
##  1    165.    152.
##  2    161.    161.
##  3    165.    161.
##  4    167.    159.
##  5    155.    163.
##  6    160.    163.
##  7    166.    163.
##  8    164.    163.
##  9    168.    164.
## 10    170.    163.
## # ... with 1,068 more rows
```



How can we summarize the relationship between the two variables `fheight` and `sheight`?

Summary statistics

We could describe the relationship between `fheight` and `sheight` with **mean** and **standard deviation**:

```
father_son %>%
  summarize(
    avg_height_father = mean(fheight),
    sd_height_father = sd(fheight),
    avg_height_son = mean(sheight),
    sd_height_son = sd(sheight)
  )
```

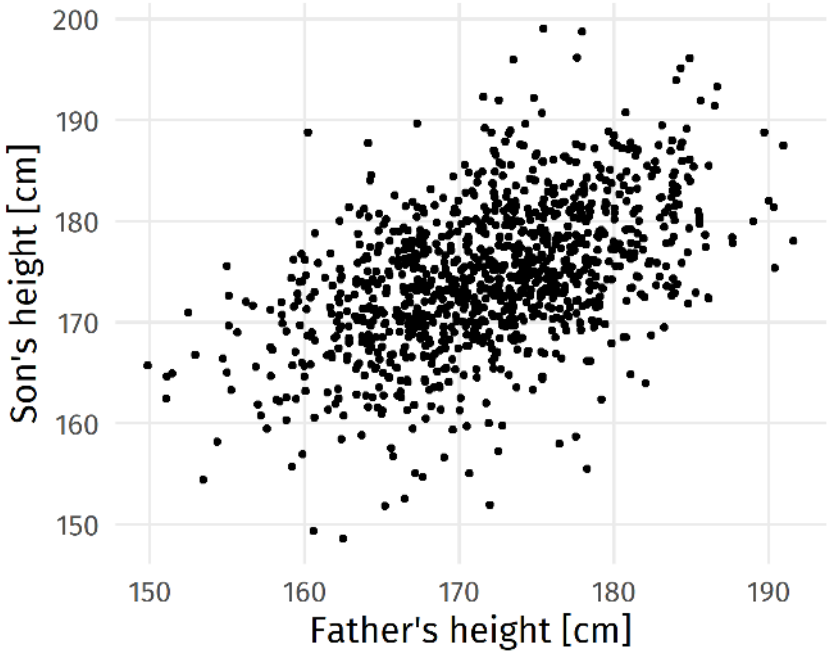
```
## # A tibble: 1 x 4
##   avg_height_father sd_height_father avg_height_son sd_height_son
##               <dbl>            <dbl>          <dbl>         <dbl>
## 1              172.             6.97           174.          7.15
```

Are mean and standard deviation sufficient?

Visualizing bivariate relationships with...

...scatterplots ...boxplots

```
ggplot(father_son, aes(x = fheight, y = sheight)) +
  geom_point() +
  labs(x = "Father's height [cm]", y = "Son's height [cm]")
```

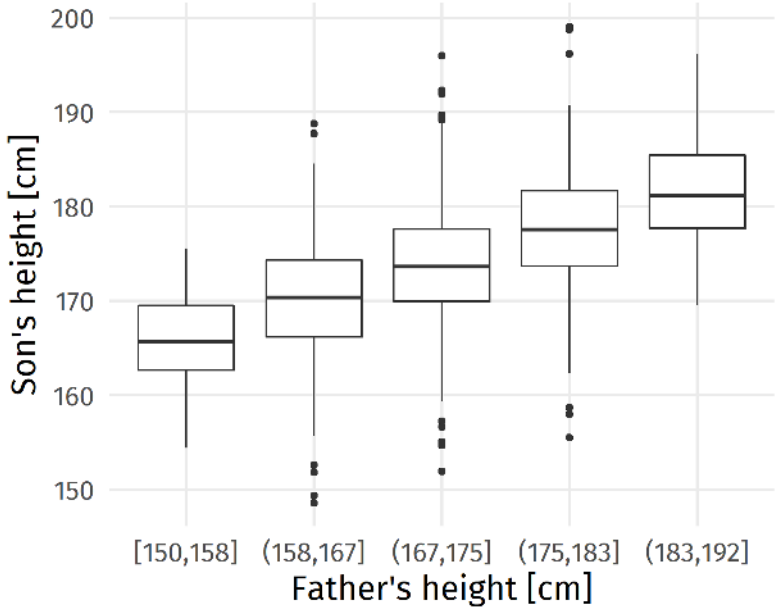


Visualizing bivariate relationships with...

...scatterplots | ...boxplots

`ggplot2::cut_interval()`: convert numerical vector into factor by discretizing the values into `n` groups of equal range

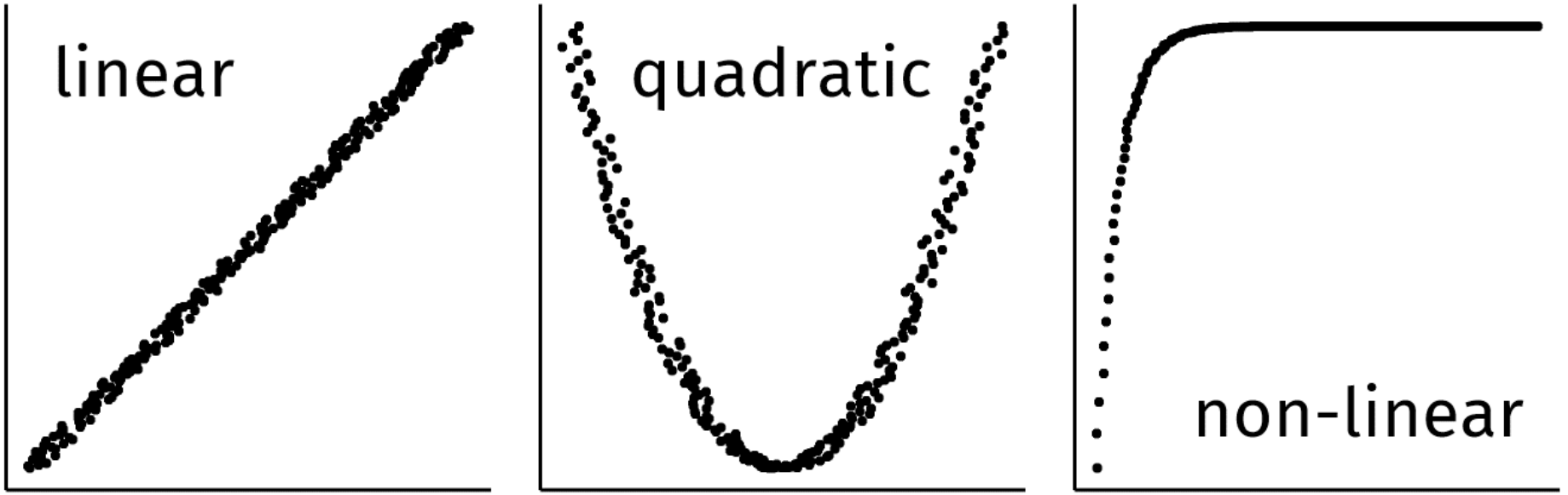
```
father_son %>%
  mutate(fheight = cut_interval(fheight, n = 5)) %>%
  ggplot(aes(x = fheight, y = sheight)) +
  geom_boxplot() +
  labs(x = "Father's height [cm]",
       y = "Son's height [cm]")
```



We observe that generally taller fathers have taller sons.

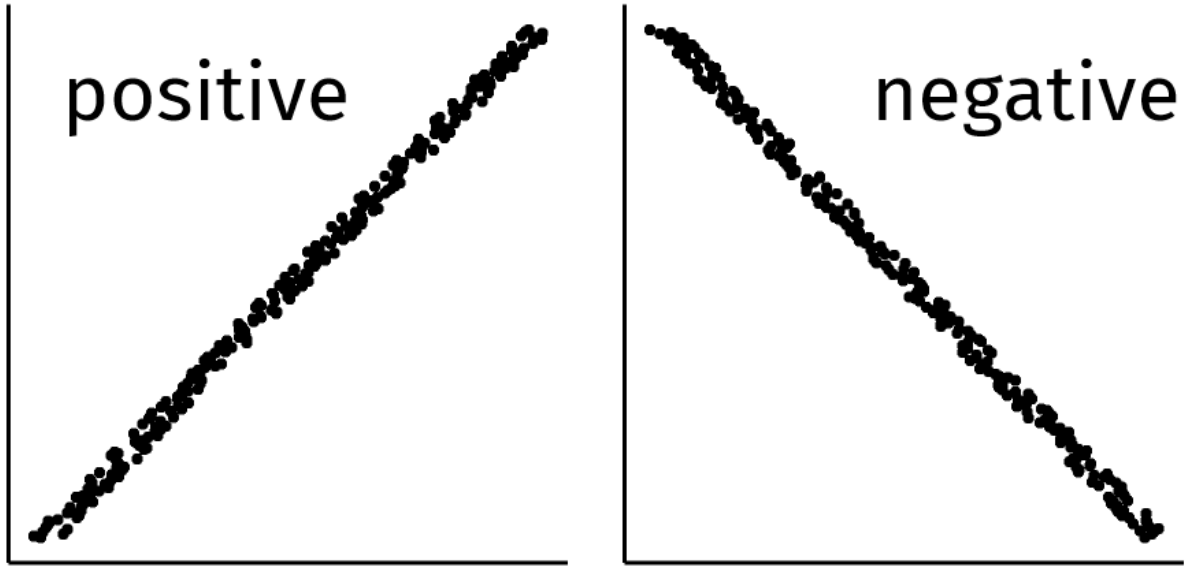
Characterizing bivariate relationships

Linearity | Direction | Strength | Outliers



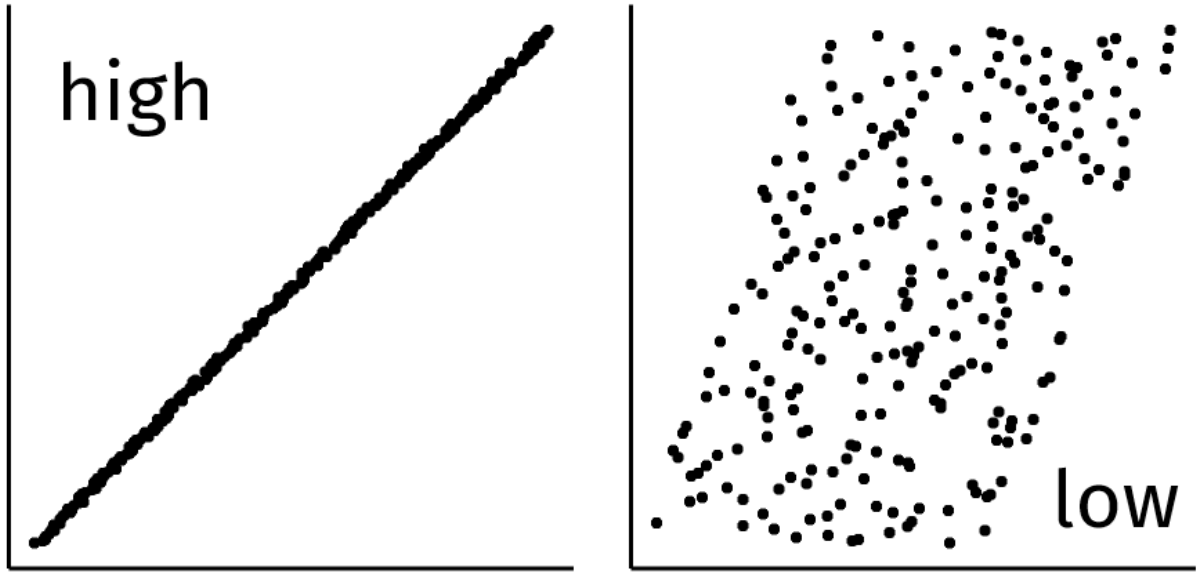
Characterizing bivariate relationships

Linearity | Direction | Strength | Outliers



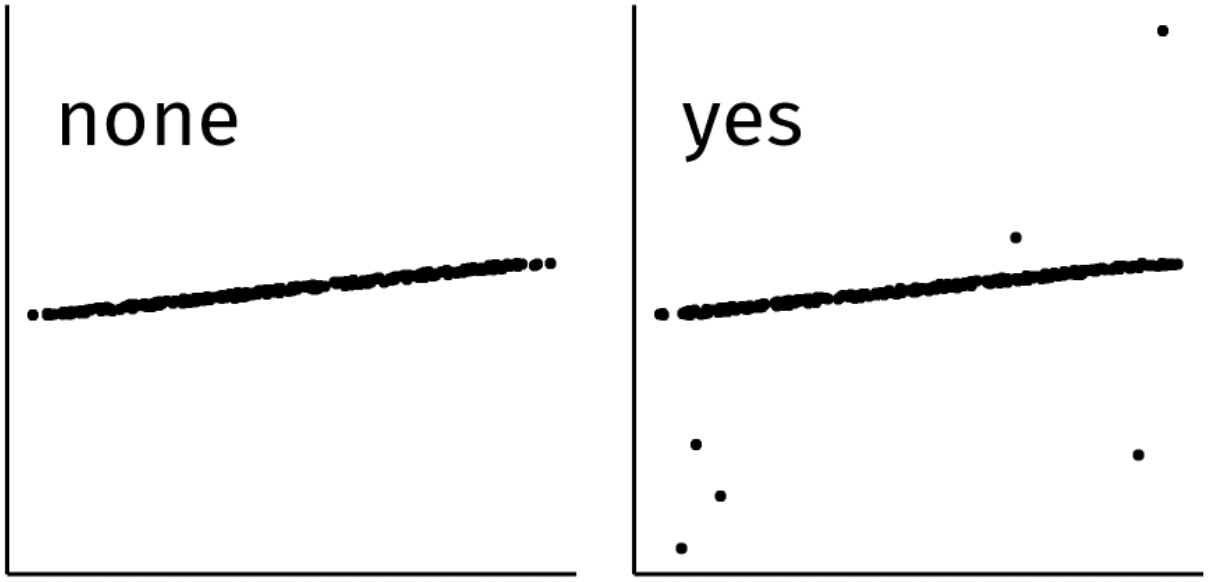
Characterizing bivariate relationships

Linearity | Direction | **Strength** | Outliers



Characterizing bivariate relationships

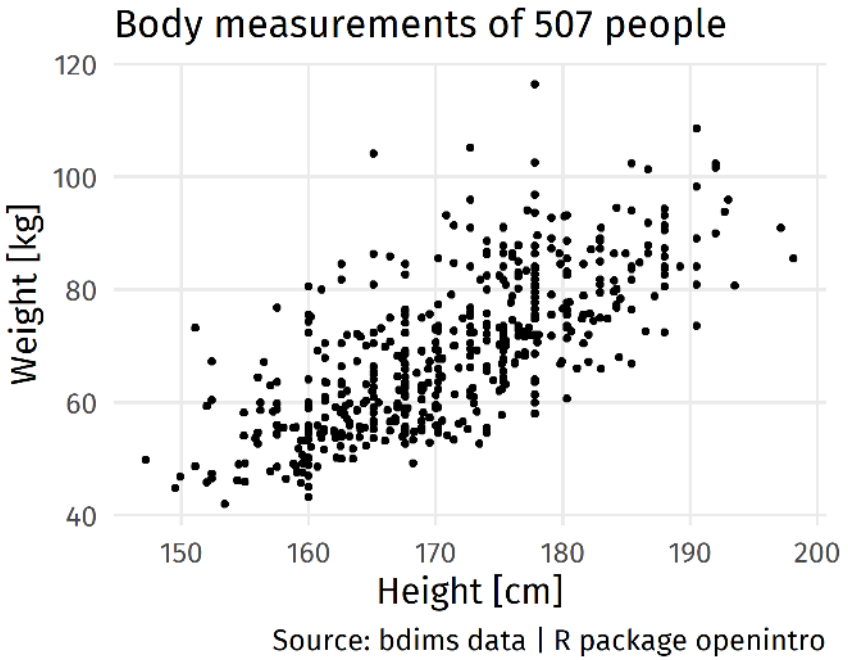
Linearity | Direction | Strength | Outliers



Examples of bivariate relationships examples

bdims | mammals | mammals (without outliers) | Boston | smoking

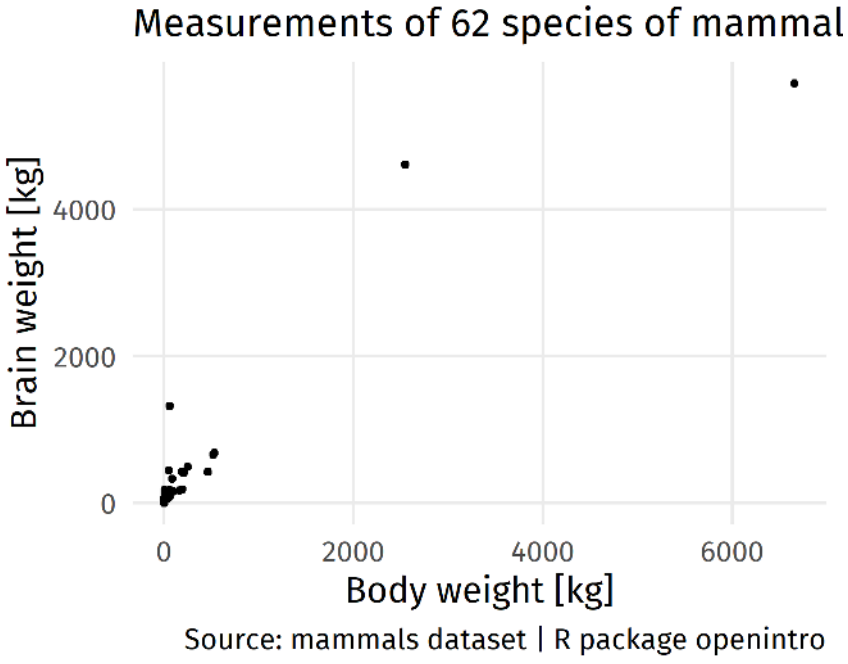
```
library(openintro)
ggplot(bdims, aes(hgt, wgt)) +
  geom_point() +
  labs(x = "Height [cm]", y = "Weight [kg]",
       title = "Body measurements of 507 people",
       caption = "Source: bdims data | R package openintro")
```



Examples of bivariate relationships examples

bdims | **mammals** | mammals (without outliers) | Boston | smoking

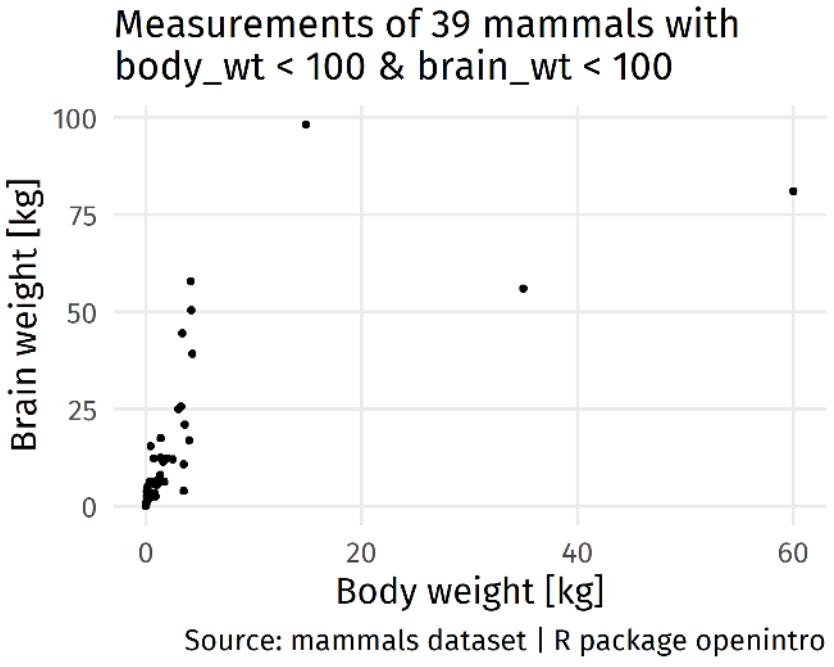
```
ggplot(mammals, aes(x = body_wt, y = brain_wt)) +
  geom_point() +
  labs(x = "Body weight [kg]", y = "Brain weight [kg]",
       title = "Measurements of 62 species of mammals",
       caption = "Source: mammals dataset | R package openintro")
```



Examples of bivariate relationships examples

bdims | mammals | **mammals (without outliers)** | Boston | smoking

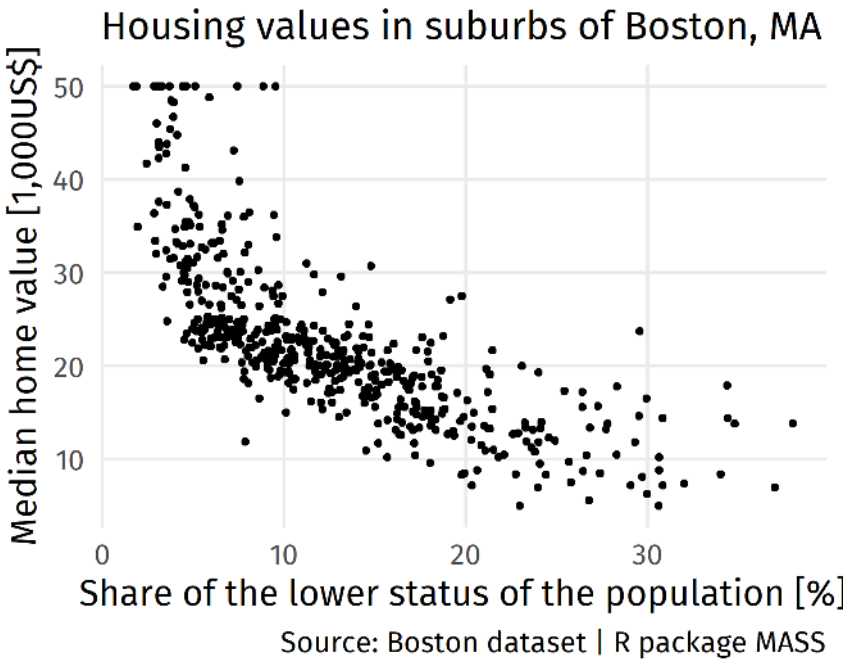
```
mammals %>%
  filter(body_wt < 100, brain_wt < 100) %>%
  ggplot(aes(x = body_wt, y = brain_wt)) +
  geom_point() +
  labs(x = "Body weight [kg]", y = "Brain weight [kg]",
       title = "Measurements of 39 mammals with\nbody_wt < 100 & brain_wt < 100",
       caption = "Source: mammals dataset | R package openintro")
```



Examples of bivariate relationships examples

bdims | mammals | mammals (without outliers) | **Boston** | smoking

```
ggplot(MASS::Boston, aes(x = lstat, y = medv)) +
  geom_point() +
  # geom_smooth(formula = y ~ x + I(x^2)) +
  labs(x = "Share of the lower status of the population [%]",
       y = "Median home value [1,000US$]",
       title = "Housing values in suburbs of Boston, MA",
       caption = "Source: Boston dataset | R package MASS")
```

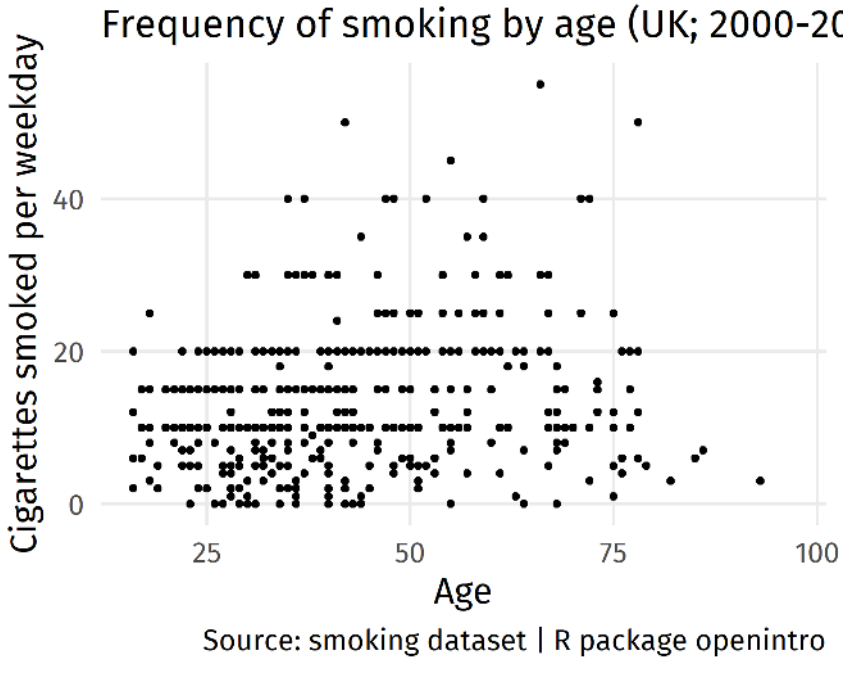


Examples of bivariate relationships examples

bdims | mammals | mammals (without outliers) | Boston | **smoking**

```
ggplot(smoking, aes(x = age, y = amt_weekdays)) +
  geom_point() +
  labs(x = "Age", y = "Cigarettes smoked per weekday",
       title = "Frequency of smoking by age (UK; 2000-2009)",
       caption = "Source: smoking dataset | R package openintro")
```

```
## Warning: Removed 1270 rows containing missing values (geom_point).
```



Correlation

(Pearson) Correlation coefficient ρ: quantification of the **strength** and **direction** of a **linear bivariate** relationship.

$$\rho = \frac{\text{cov}_{x,y}}{s_x s_y} = \frac{\sum_{i=1}^{N}(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\frac{\sum_{i=1}^{N}(x_i - \bar{x})^2}{N-1}}\sqrt{\frac{\sum_{i=1}^{N}(y_i - \bar{y})^2}{N-1}}}$$

with

- $\bar{x}, \bar{y}$: sample mean of x, y
- s_x, s_y: standard deviation of x, y
- $\text{cov}_{x,y}$: covariance of x and y
- N: number of observations

Interpretation

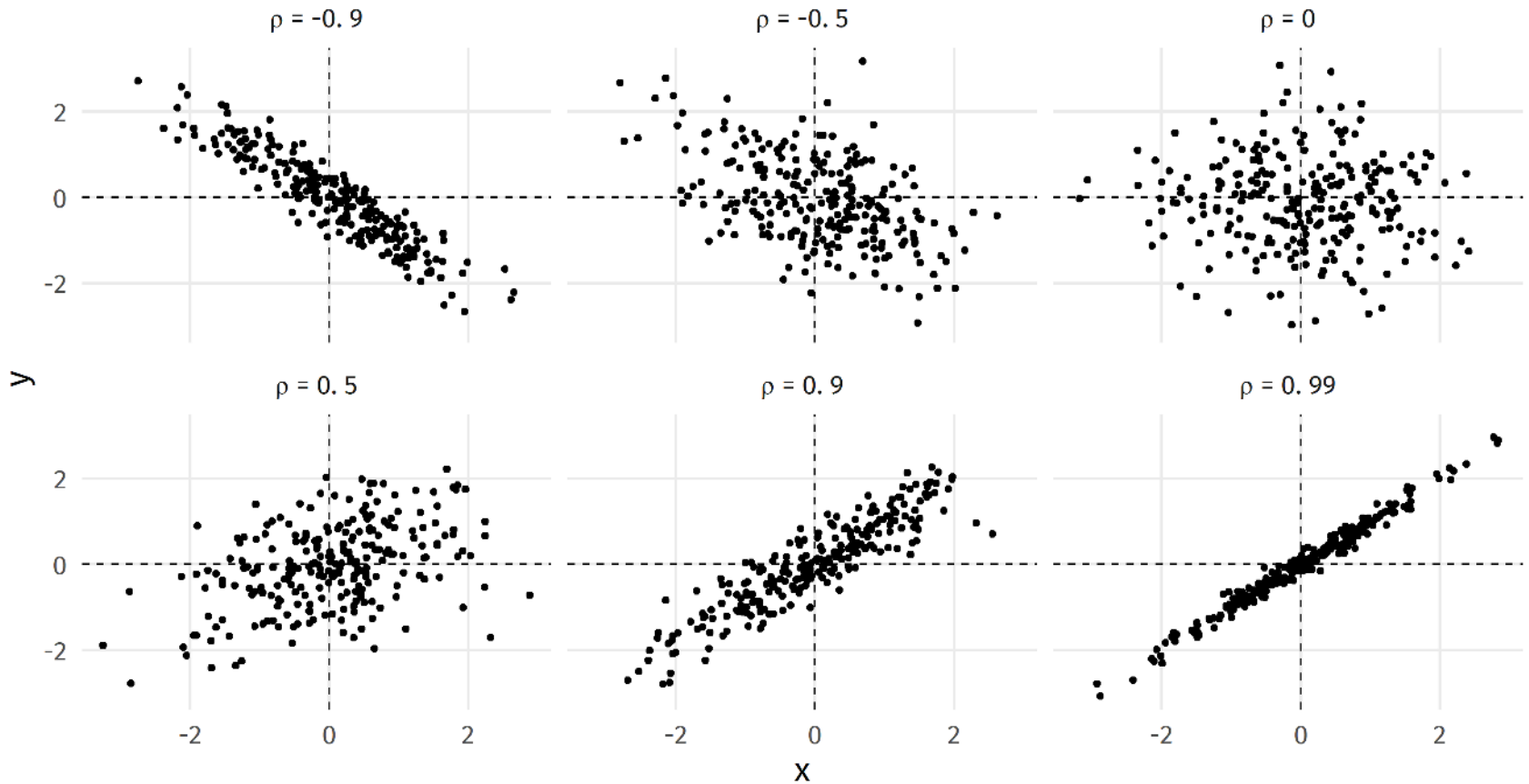
	Direction	Description
$\rho > 0$	**positive linear relationship**	*Both variables are always above the mean or below the mean together.*
$\rho < 0$	**negative linear relationship**	*The variables move in opposite directions.*

	Strength (rules of thumb)
$\rho \approx 0$	**no linear correlation** (the variables are independent)
$\text{abs}(\rho) \approx 0.1$	**weak relationship**
$\text{abs}(\rho) \approx 0.3$	**moderate relationship**
$\text{abs}(\rho) \approx 0.5$	**strong relationship**
$\text{abs}(\rho) \approx 1$	**perfect linear correlation** (the variables are dependent)

Correlation examples

Six examples of bivariate distributions with different correlation coefficients:

Plot | Code



Correlation examples

Six examples of bivariate distributions with different correlation coefficients:

Plot | Code

```
library(purrr)
n <- 250 # Sample size
cors <- c(-0.9, -0.5, 0, 0.5, 0.9, 0.99) # correlation coefficients
dat <- map_dfr(cors, function(r) {
  # Draw random sample from bivariate normal distribution with
  # mean=0, sd=1 and covar=r
  mat_mvrnorm <- MASS::mvrnorm(n, c(0,0), matrix(c(1,r,r,1),2,2))
  tibble(r = r, x = mat_mvrnorm[, 1], y = mat_mvrnorm[, 2])
})
ggplot(dat, aes(x,y)) +
  facet_wrap(~r, labeller = as_labeller(function(x){
    latex2exp::TeX(paste0("$\\rho$ = ", x), output = "text")
  }, default = label_parsed)) +
  geom_point() +
  geom_vline(xintercept = 0, linetype = 2) +
  geom_hline(yintercept = 0, linetype = 2)
```

Computing correlation coefficients with `R`

We can calculate the correlation coefficient in `R` using `stats::cor()`:

```
cor(x, y = NULL, use = "everything", method = "pearson")
```

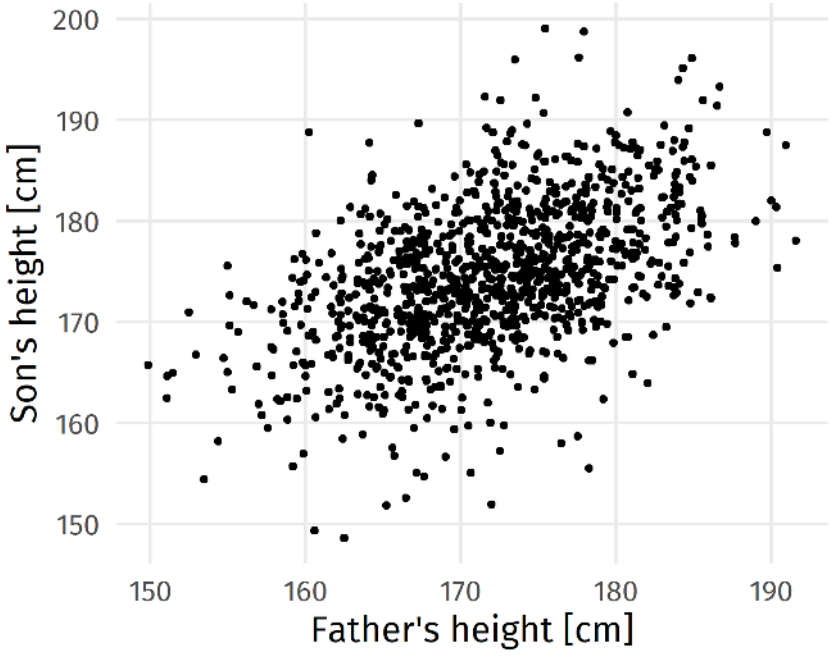
- `x`: numeric variable/ data frame/ matrix
- `y`: another numeric variable (optional)
- `use`: handling of missing values
  - `"everything"` returns `NA` if any of the two variables has missing values
  - `"all.obs"` returns an error if there are any missing values in the data
  - `"complete.obs"` computes correlations only from cases without missing values
  - `"pairwise.complete.obs"` computes correlations between pairs of variables only from cases without missing values for those two variables
- `method`:
  - `"pearson"`: parametric, use for normally distributed data
  - `"spearman"`: non-parametric, use for non-normally distributed data
  - `"kendall"`: non-parametric, use for *smallish* non-normally distributed data

Correlation: father-son data

Pearson correlation coefficient for father-son data:

```
rho <-
  cor(father_son$fheight, father_son$sheight)
rho
```

```
## [1] 0.5013383
```



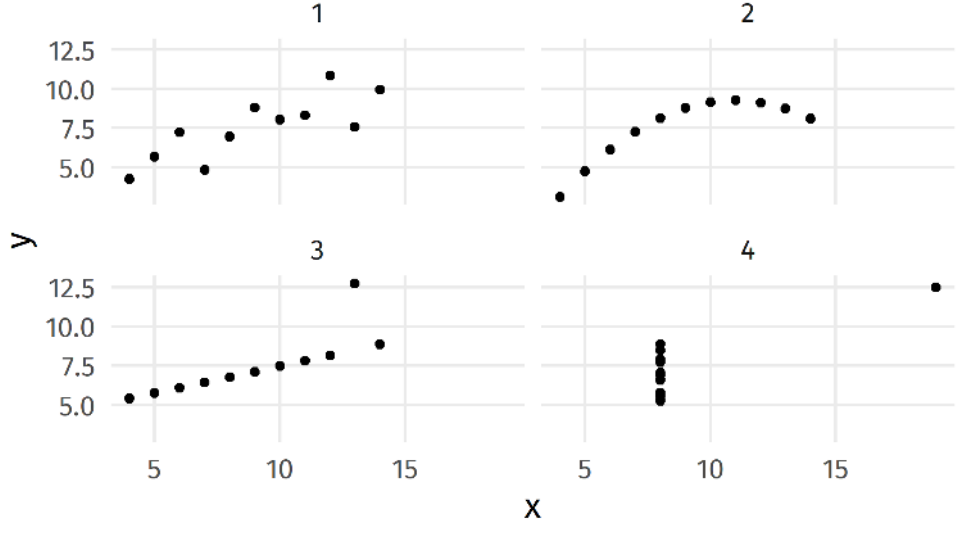
Use correlation for linear relationships only

⚠ Correlation is not always an appropriate summary of the relationship between a pair of variables.

This is illustrated by the famous Anscombe's quartet which comprises four two-dimensional datasets with a correlation of $\rho = 0.82$ between the input features:

x1	x2	x3	x4	y1	y2	y3	y4
10	10	10	8	8.04	9.14	7.46	6.58
8	8	8	8	6.95	8.14	6.77	5.76
13	13	13	8	7.58	8.74	12.74	7.71
9	9	9	8	8.81	8.77	7.11	8.84
11	11	11	8	8.33	9.26	7.81	8.47
14	14	14	8	9.96	8.10	8.84	7.04
6	6	6	8	7.24	6.13	6.08	5.25
4	4	4	19	4.26	3.10	5.39	12.50
12	12	12	8	10.84	9.13	8.15	5.56
7	7	7	8	4.82	7.26	6.42	7.91
5	5	5	8	5.68	4.74	5.73	6.89

```
## # A tibble: 4 x 6
##   group x_mean  x_sd y_mean  y_sd   rho
##   <chr>  <dbl> <dbl>  <dbl> <dbl> <dbl>
## 1 1          9  3.32   7.50  2.03 0.816
## 2 2          9  3.32   7.50  2.03 0.816
## 3 3          9  3.32   7.5   2.03 0.816
## 4 4          9  3.32   7.50  2.03 0.817
```



Spurious correlations

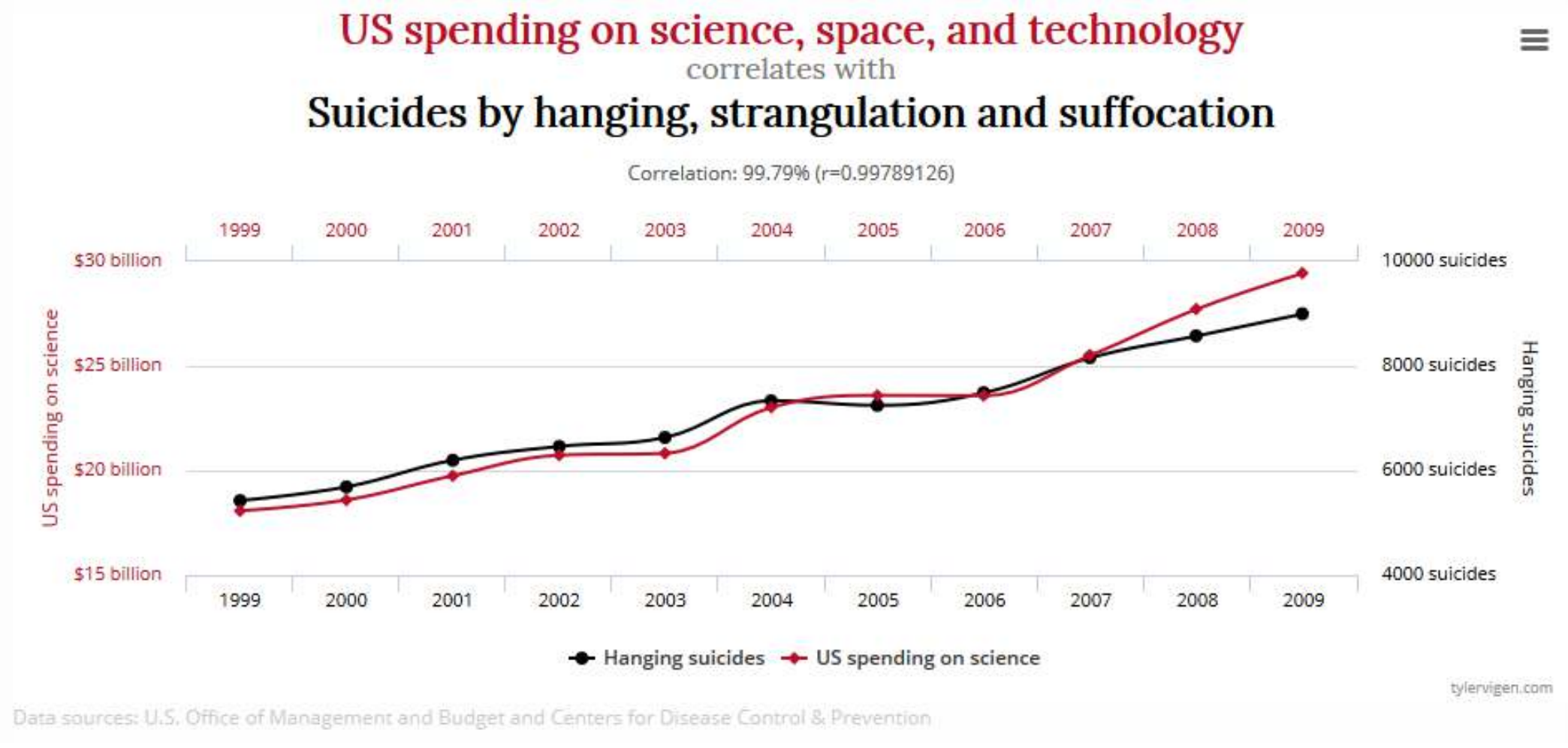
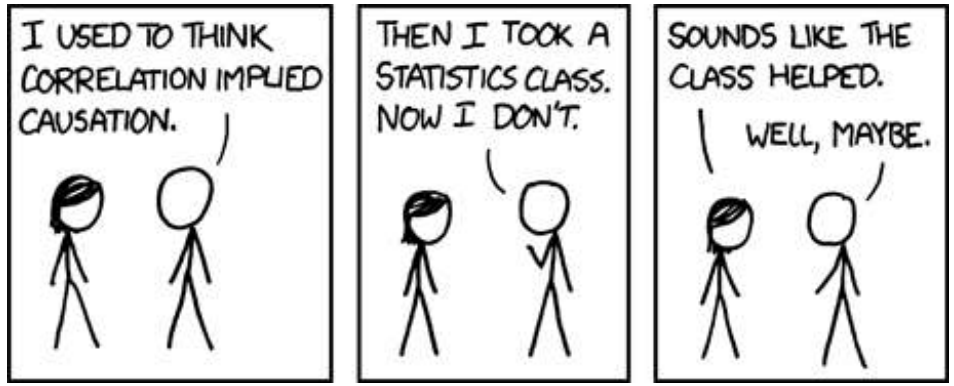


Figure source: http://www.tylervigen.com/spurious-correlations. Last accessed 17.02.2021.

Correlation vs. causation



⚠ Correlation does not imply causation.

Possible reasons:

Confounding/ third-variable problem: a measured or unmeasured variable affects the results.
Example: *Elite college degree → adult earnings* (confounder: parental socioeconomic status)

Direction of causality: correlation coefficients do not imply the direction of causation.
Example: *Chocolate consumption is correlated with acne and vice versa. Chocolate contains ingredients such as fats or sugar causing acne, so chocolate consumption leads to acne, but not the other way around.*

Figure source: https://xkcd.com/552/

Simple Linear Regression

Goal: **predict** a **quantitative response** y on the basis of a single **predictor** x

Terminology:

- y: response / dependent variable
- x: predictor / independent or explanatory variable

General statistical model

The value of the **response** is **modeled** as a **function of the predictor** (plus some additional noise).

$$y = f(x) + \epsilon, \qquad \epsilon \sim N(0, \sigma_\epsilon)$$

Example: $y = f(x) = 2x + 3$ is a function with input x (predictor) and output y (response). If x is 4, y is 11, then $y = 2 \times 4 + 3 = 11$

Simple linear regression

Simple linear regression: predict a quantitative response Y on the basis of a **single predictor variable** X

Assumption: model fit is linear, i.e., the data can be summarized with a straight line.

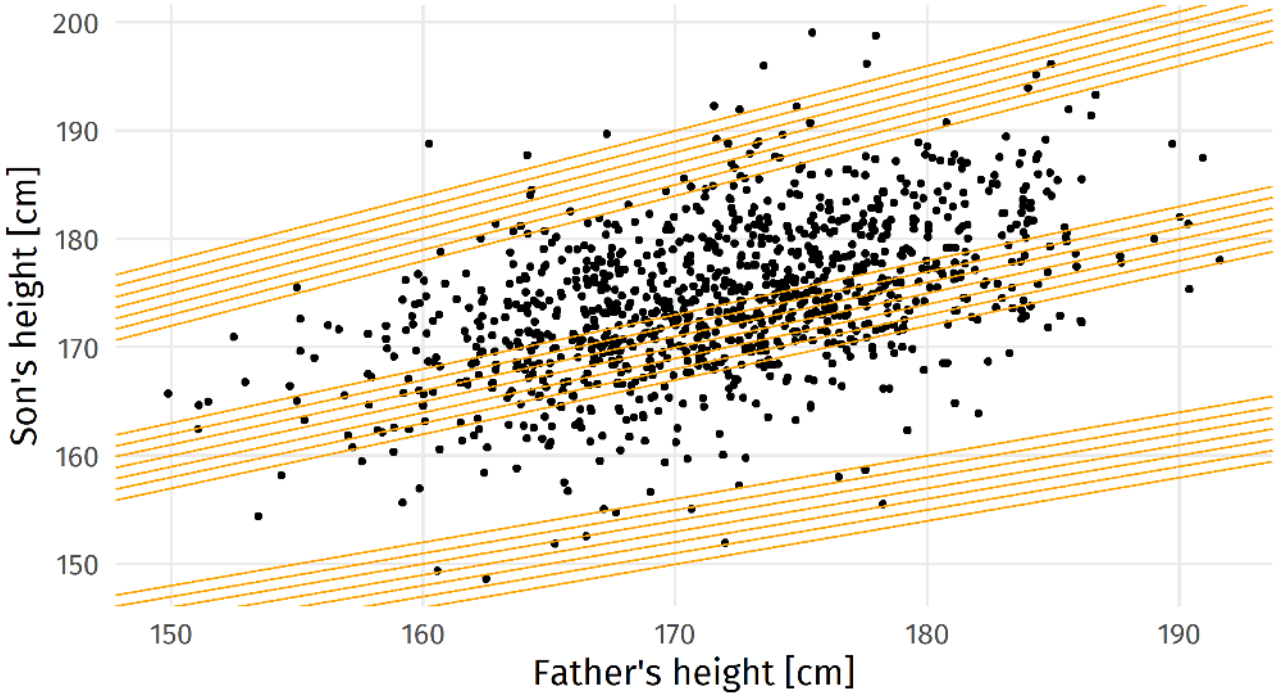
$$f(x) = \beta_0 + \beta_1 x$$

$\rightarrow$ two unknown **parameters**/**coefficients**: **intercept** β_0 and **slope** β_1

Best fit line

Line candidates | Best fit | Best fit without measure of uncertainty

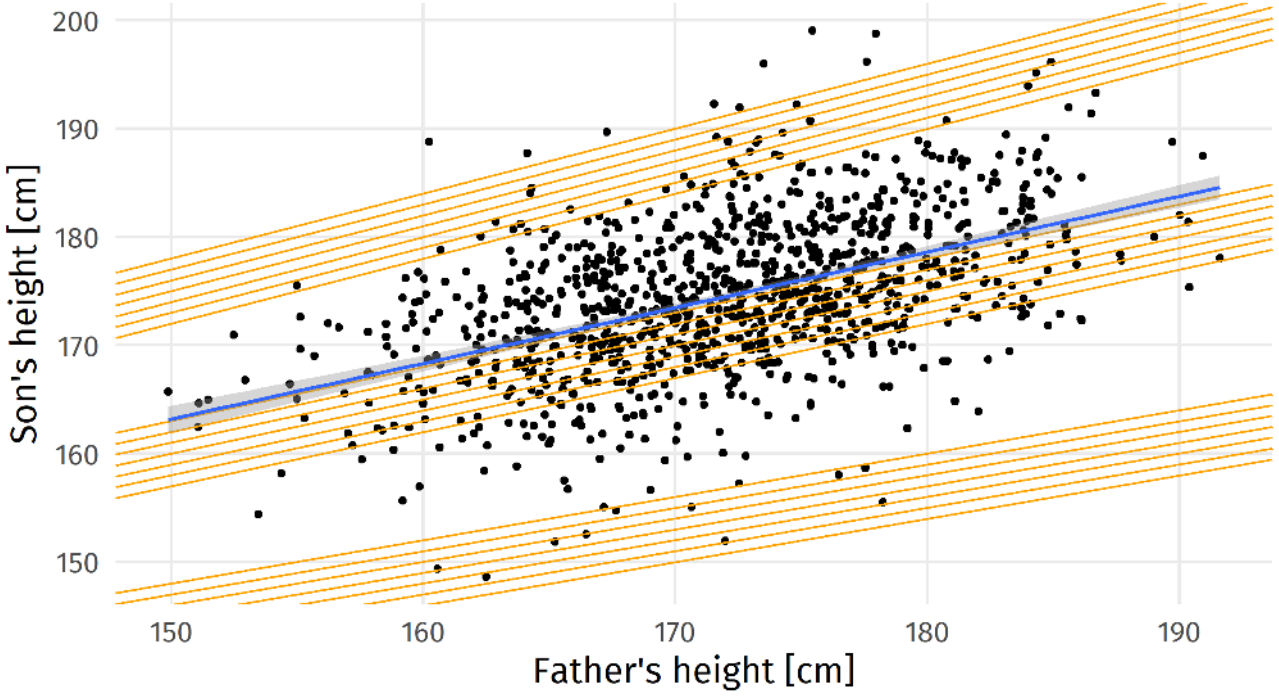
```
# Manually create some regression lines
df_reg_lines <- expand_grid(intercept = 82:88, slope = seq(0.4, 0.6, 0.1))
ggplot(data = father_son, aes(x = fheight, y = sheight)) +
  geom_point() +
  geom_abline(data = df_reg_lines, aes(intercept = intercept, slope = slope), color = "orange") +
  labs(x = "Father's height [cm]", y = "Son's height [cm]")
```



Best fit line

Line candidates | **Best fit** | Best fit without measure of uncertainty

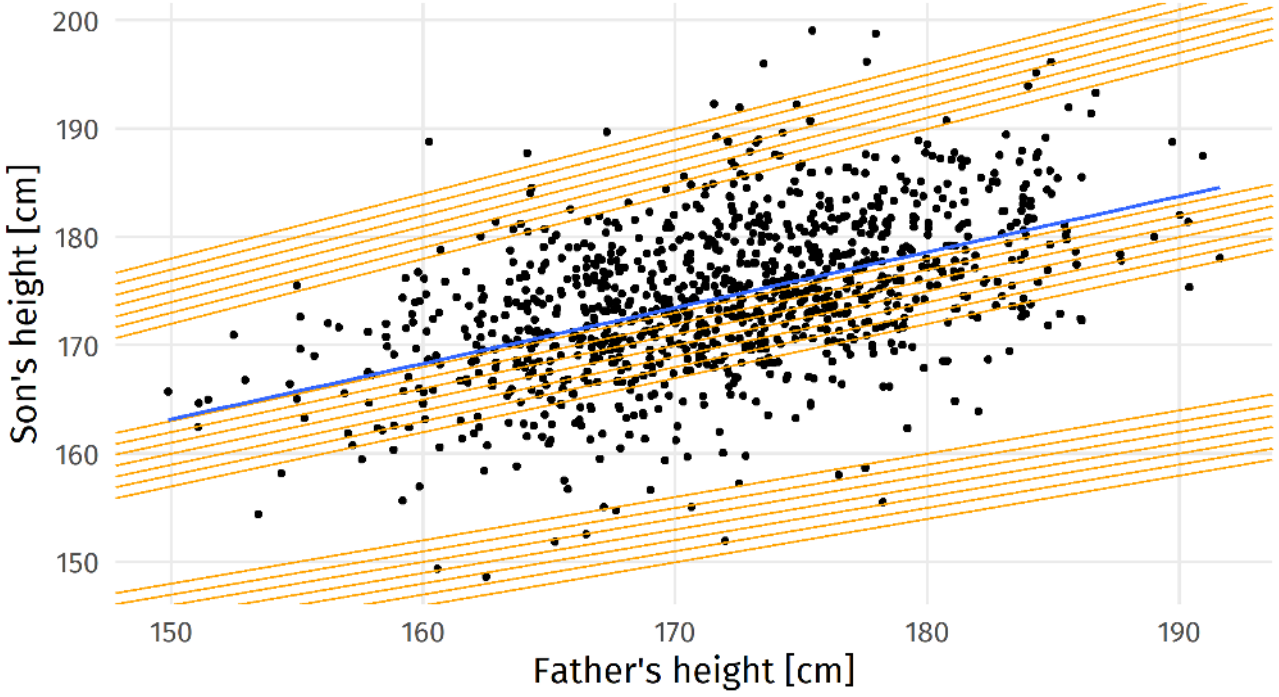
```
ggplot(data = father_son, aes(x = fheight, y = sheight)) +
  geom_point() +
  geom_abline(data = df_reg_lines, aes(intercept = intercept, slope = slope), color = "orange") +
  geom_smooth(method = "lm", size = 1) +
  labs(x = "Father's height [cm]", y = "Son's height [cm]")
```



Best fit line

Line candidates | Best fit | **Best fit without measure of uncertainty**

```
ggplot(data = father_son, aes(x = fheight, y = sheight)) +
  geom_point() +
  geom_abline(data = df_reg_lines, aes(intercept = intercept, slope = slope), color = "orange") +
  geom_smooth(method = "lm", size = 1,
    se = FALSE) +
  labs(x = "Father's height [cm]", y = "Son's height [cm]")
```

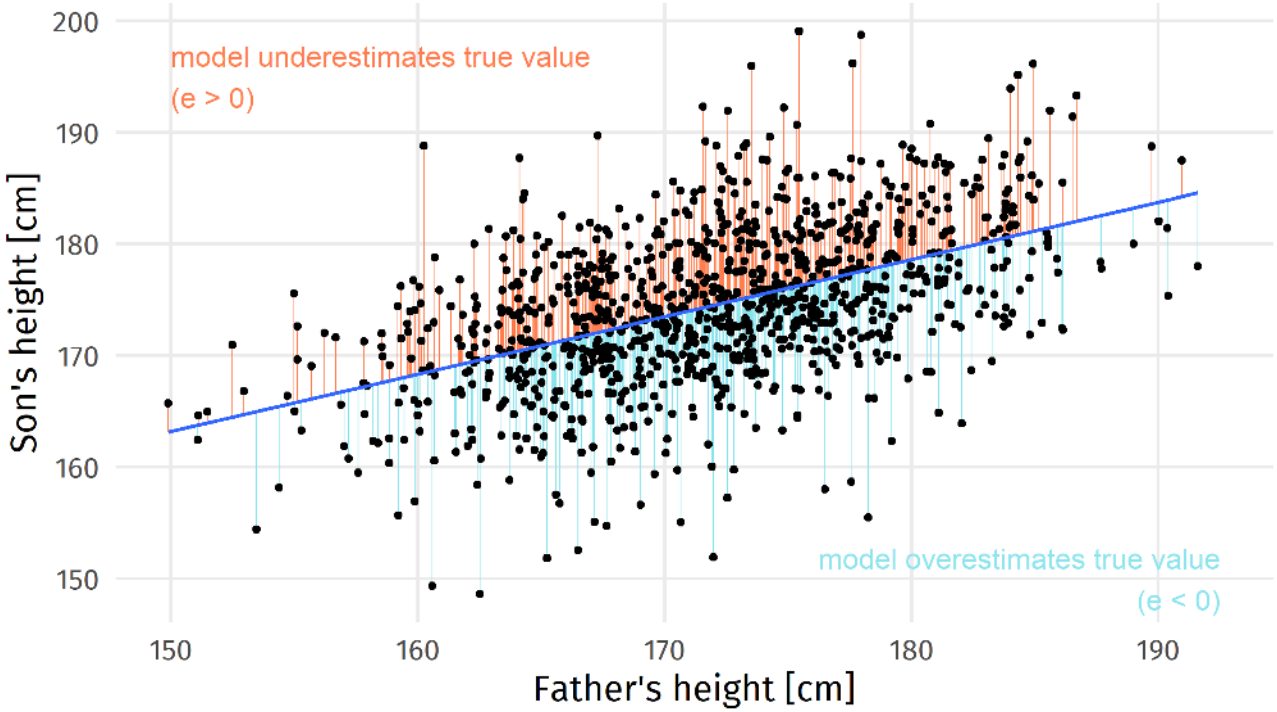


Residuals

Fitted values: $\hat{Y} = \hat{\beta}_0 + \hat{\beta}_1 X$ $\longrightarrow$ **Residuals**: $e = Y - \hat{Y}$

A residual quantifies the model error for an individual observation by calculating the difference between the actual response value and the predicted response value.

Plot | Code



Residuals

Fitted values: $\hat{Y} = \hat{\beta}_0 + \hat{\beta}_1 X$ $\longrightarrow$ **Residuals**: $e = Y - \hat{Y}$

A residual quantifies the model error for an individual observation by calculating the difference between the actual response value and the predicted response value.

Plot | Code

```
model <- linear_reg() %>%
  set_engine("lm") %>%
  fit(sheight ~ fheight, data = father_son)
father_son %>%
  mutate(y_hat = model$fit$coefficients[1] + fheight * model$fit$coefficients[2]) %>%
  mutate(res_cat = ifelse(y_hat > sheight, "overest", "underest")) %>%
  ggplot(aes(fheight, sheight)) +
  geom_segment(aes(xend = fheight, yend = y_hat, color = res_cat), show.legend = FALSE, size = .25) +
  geom_point() +
  annotate("text", x = 150, y = 195, label = "model underestimates true value\n(e > 0)", color = "coral"
  annotate("text", x = 192.5, y = 150, label = "model overestimates true value\n(e < 0)", color = "cadet
  scale_color_manual(values = c("cadetblue2", "coral")) +
  geom_smooth(method = "lm", se = FALSE, size = 1) +
  labs(x = "Father's height [cm]", y = "Son's height [cm]")
```

Least squares fitting procedure

Goal: find the values that minimize the distance of the fitted model to the data.

$\rightarrow$ **Least squares** (LS) equation for N observations of pairs (x_i, y_i):

$$\min_{\beta_0,\beta_1} RSS = \min_{\beta_0,\beta_1} \sum_{i=1}^{N} e_i^2 = \min_{\beta_0,\beta_1} \sum_{i=1}^{N} \left(y_i - \left(\hat{\beta}_0 + \hat{\beta}_1 x_i\right)\right)^2$$

- Parameter values of the minimum are referred to as the **least squares estimates (LSE)** $\hat{\beta}$
- The loss function is also known as **residual sum of squares** RSS

Relationship between correlation and regression

Regression line explains that for every standard deviation σ_x increase above the average $\bar{x}$, y grows ρ standard deviations σ_y above the average $\bar{y}$:

$$\left(\frac{y_i - \bar{y}}{\sigma_y}\right) = \rho \left(\frac{x_i - \bar{x}}{\sigma_x}\right)$$

Example for our father-son dataset, given a father height of x_i = 160 cm:

$$\left(\frac{\hat{y}_i - 174.46}{7.15}\right) = 0.5013 \cdot \left(\frac{160 - 171.93}{6.97}\right)$$

$$\hat{y}_i = 0.5013 \cdot \left(\frac{160 - 171.93}{6.97}\right) \cdot 7.15 + 174.46$$

$$\hat{y}_i = 168.33$$

TODO: tidymodels overview

Linear regression in `R`

Specify model | Set engine | Fit model and estimate parameters

```
library(tidymodels)
linear_reg()
```

```
## Linear Regression Model Specification (regression)
```

TODO: tidymodels overview

Linear regression in `R`

Specify model | Set engine | Fit model and estimate parameters

```
linear_reg() %>%
  set_engine("lm")
```

```
## Linear Regression Model Specification (regression)
##
## Computational engine: lm
```

TODO: tidymodels overview

Linear regression in `R`

Specify model | Set engine | Fit model and estimate parameters

```
linear_reg() %>%
  set_engine("lm") %>%
  fit(formula = sheight ~ fheight, data = father_son) -> model
model
```

```
## parsnip model object
##
## Fit time:  0ms
##
## Call:
## stats::lm(formula = sheight ~ fheight, data = data)
##
## Coefficients:
## (Intercept)      fheight
##     86.0720       0.5141
```

We specify the regression equation using **formula syntax**:

`response ~ predictor1 + predictor2 + ...` (Note that we don't use quotation marks here.)

Model output

```
## parsnip model object
##
## Fit time:  0ms
##
## Call:
## stats::lm(formula = sheight ~ fheight, data = data)
##
## Coefficients:
## (Intercept)      fheight
##     86.0720       0.5141
```

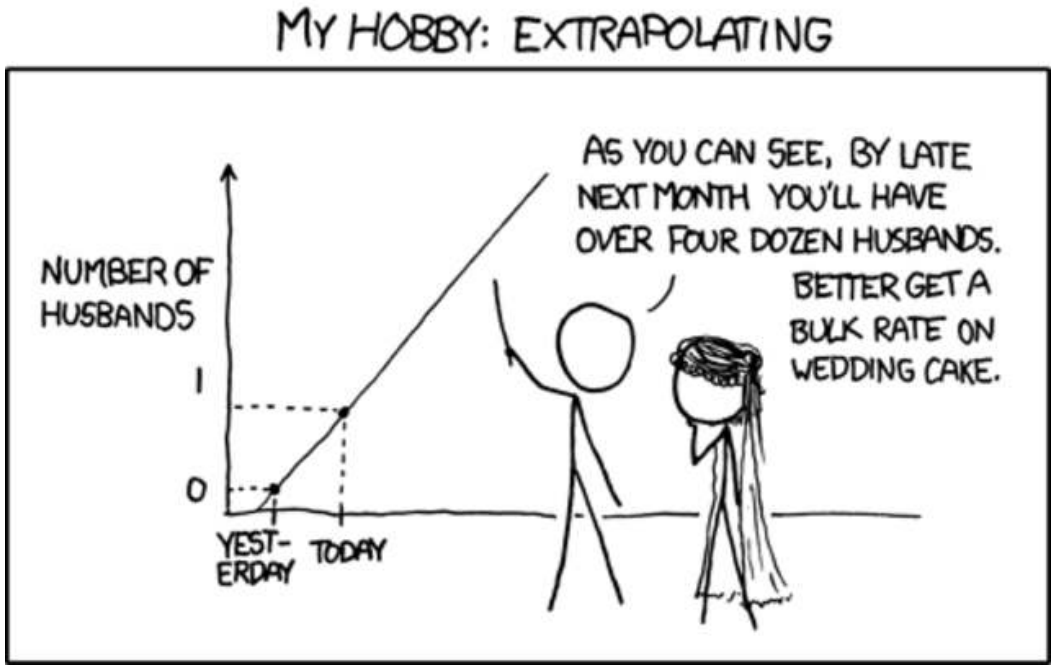
$$\widehat{sheight}_i = 86.0720 + 0.5141 \cdot fheight_i$$

Interpretation:

- **Slope**: For each additional cm the father is taller, the son is expected to be taller, on average, by 0.5141 cm.
- **Intercept**: A father who is 0 cm tall is expected to have a son that is, on average, 86.0720 cm tall.

For simple linear regression, the relationship between a regression line's slope β_1 and the Pearson correlation coefficient ρ between the explanatory variable and the response is: $\beta_1 = \rho \cdot \frac{s_y}{s_x}$.

Extrapolation



⚠ Be careful when interpreting linear model predictions for observations beyond the range of the data it was built on.

Figure source: https://xkcd.com/605/. Last accessed 17.02.2021.

Assessing model diagnostics with the `broom` package

The `broom` package provides functions to convert model diagnostics into *tidy* data frames.

- `tidy()` returns for each coefficient a row with its value, the standard error, t-statistic and p-value.
- `augment()` adds columns to the original dataset from the model fit, e.g. residuals and influence statistics.
- `glance()` returns a row with summary statistics, e.g. R^2, degrees of freedom, AIC.

```
library(broom) # included in tidymodels
```

See `vignette("broom")` for more information.

Tidy model output

```
linear_reg() %>%
  set_engine("lm") %>%
  fit(formula = sheight ~ fheight, data = father_son) %>%
  tidy()
```

```
## # A tibble: 2 x 5
##   term        estimate std.error statistic  p.value
##   <chr>          <dbl>     <dbl>     <dbl>    <dbl>
## 1 (Intercept)   86.1      4.65        18.5 1.60e-66
## 2 fheight        0.514    0.0270      19.0 1.12e-69
```

$$\widehat{sheight}_i = 86.0720 + 0.5141 \cdot fheight_i$$

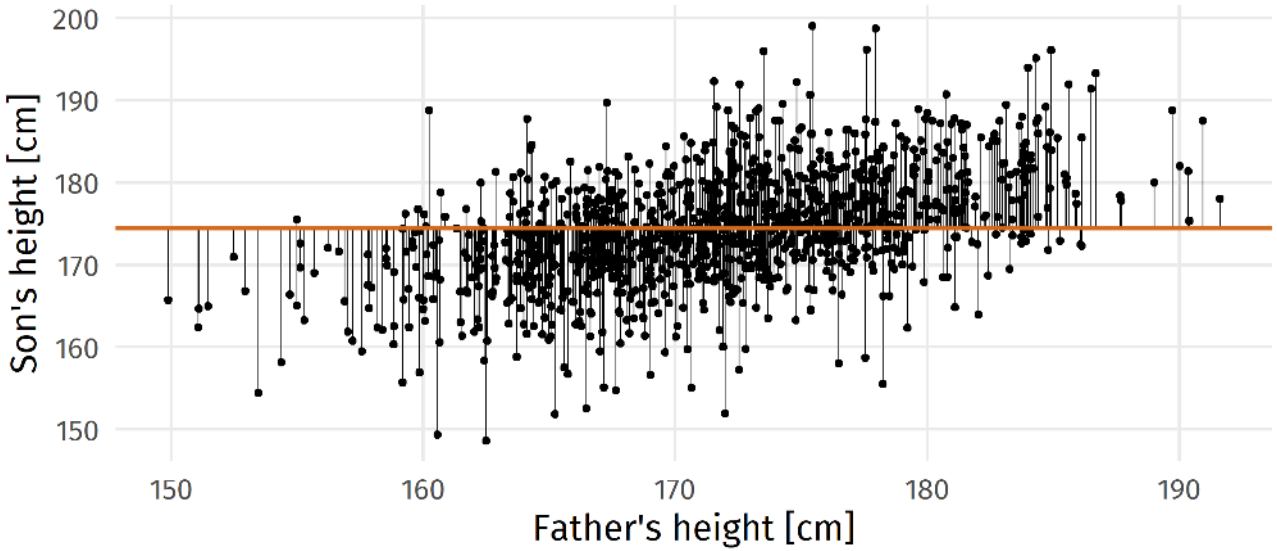
Assessing the model fit

TSS | RSS | MSS | R^2

A simple **baseline model** is the mean of the response.

```
(y_hat <- mean(father_son$sheight))
```

```
## [1] 174.4575
```

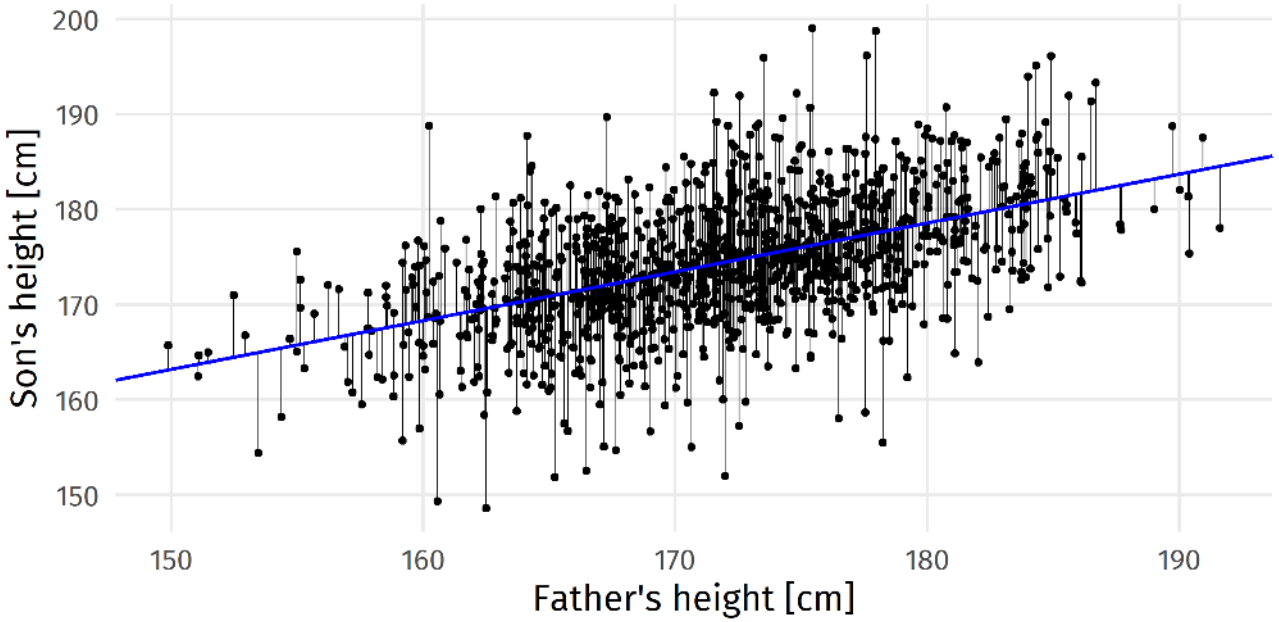


The sum of squared differences between the observed values and the values predicted by the mean is called **total sum of squares** TSS.

Assessing the model fit

TSS | RSS | MSS | R^2

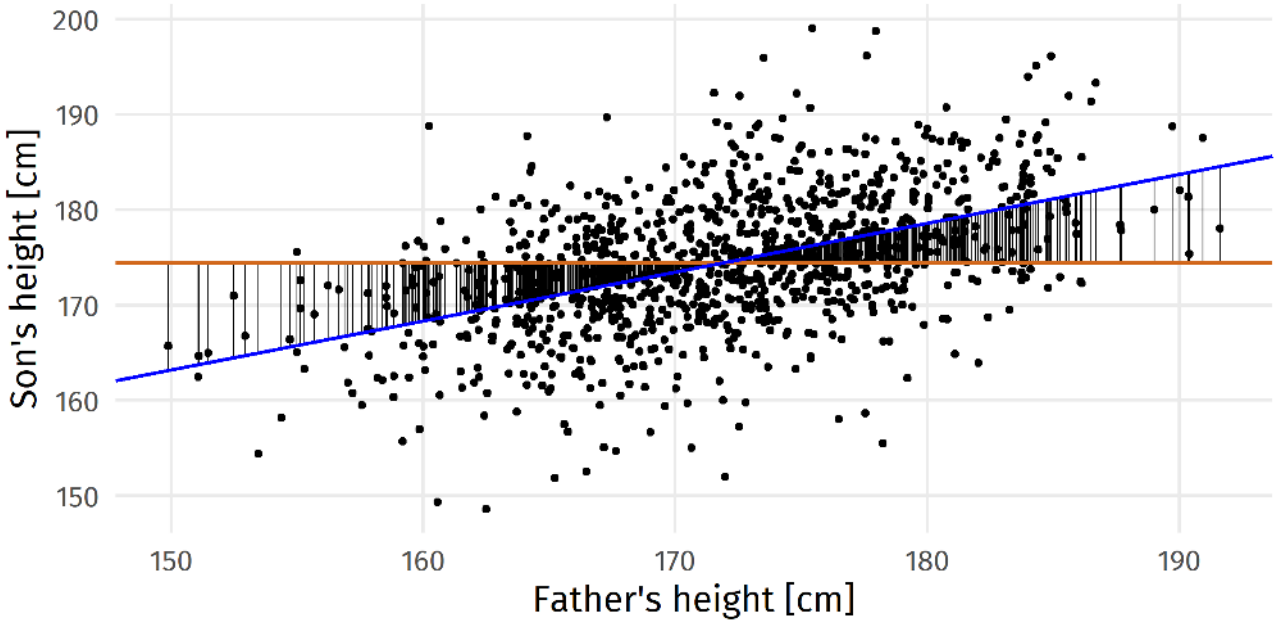
The sum of squared differences between the observed values and the values predicted by the linear regression model is called **residual sum of squares** RSS.



Assessing the model fit

TSS RSS MSS R^2

The improvement in prediction between the regression model and the mean is shown in the **model sum of squares** MSS. It is the sum of squared differences between the predicted values of the regression model and the mean. The value of MSS is large when the regression model is very different to the mean prediction.



Assessing the model fit

TSS RSS MSS R^2

The proportion of improvement due to the regression model is **the coefficient of determination** R^2. It measures the **share of variability in the response that is captured by the regression model**.

$$R^2 = \frac{MSS}{TSS} = 1 - \frac{RSS}{TSS}$$

```
glance(model)
```

```
## # A tibble: 1 x 12
##   r.squared adj.r.squared sigma statistic  p.value    df logLik   AIC   BIC deviance
##       <dbl>         <dbl> <dbl>     <dbl>    <dbl> <dbl>  <dbl> <dbl> <dbl>    <dbl>
## 1     0.251         0.251  6.19      361. 1.12e-69     1 -3494. 6993. 7008.   41213.
## # ... with 2 more variables: df.residual <int>, nobs <int>
```

R^2 = 0.25 → Only 25% of the variation in son height can be explained by father height alone.

It can be concluded that there are other factors that influence the height of sons (possibly mother height, genetic markers, etc.).

Multiple linear regression

General model:

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \ldots + \beta_i X_p + \epsilon$$

Predicted response equation:

$$\hat{Y} = \hat{\beta}_0 + \hat{\beta}_1 X_1 + \hat{\beta}_2 X_2 + \ldots + \hat{\beta}_i X_p$$

Ames Iowa Housing Dataset

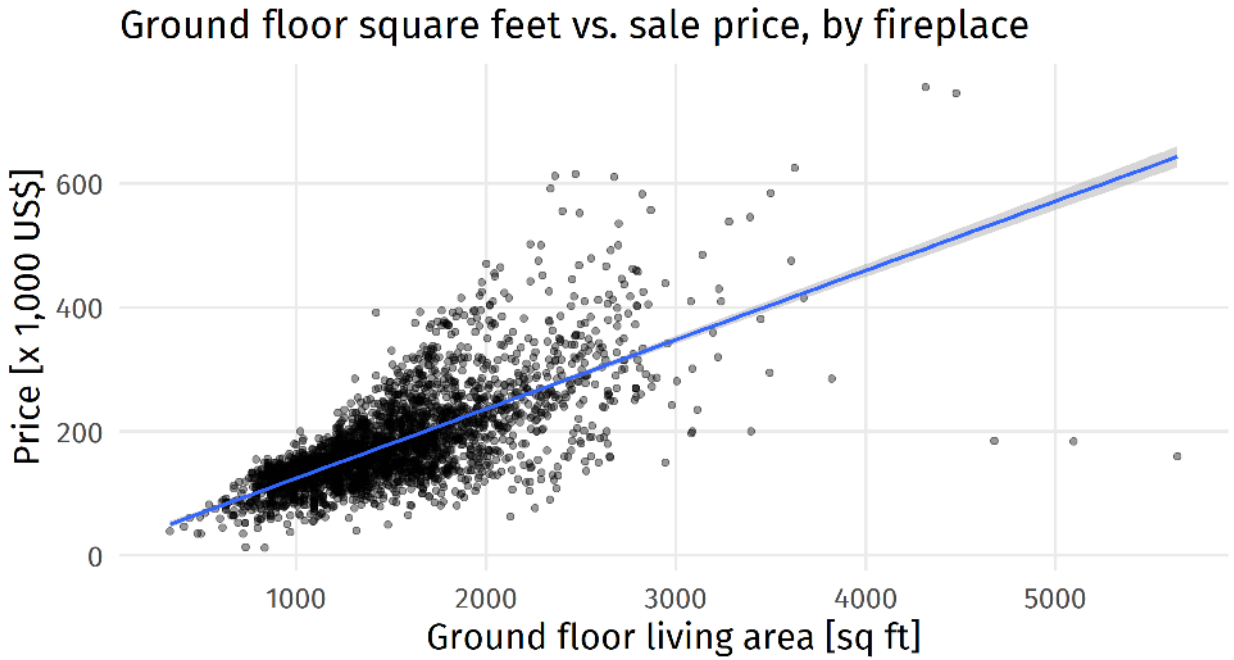
> *"Data set contains information from the Ames Assessor's Office used in computing assessed values for individual residential properties sold in Ames, IA from 2006 to 2010." — Dataset documentation*

De Cock, Dean. "Ames, Iowa: Alternative to the Boston housing data as an end of semester regression project." Journal of Statistics Education 19.3 (2011). URL

```
library(AmesHousing)
ames <- make_ames() %>% # prepares dataset
  select(-matches("Qu")) # remove quality columns
glimpse(ames)
```

```
## Rows: 2,930
## Columns: 74
## $ MS_SubClass       <fct> One_Story_1946_and_Newer_All_Styles, One_~
## $ MS_Zoning         <fct> Residential_Low_Density, Residential_High~
## $ Lot_Frontage      <dbl> 141, 80, 81, 93, 74, 78, 41, 43, 39, 60, ~
## $ Lot_Area          <int> 31770, 11622, 14267, 11160, 13830, 9978, ~
## $ Street            <fct> Pave, Pave, Pave, Pave, Pave, Pave, Pave,~
## $ Alley             <fct> No_Alley_Access, No_Alley_Access, No_Alle~
## $ Lot_Shape         <fct> Slightly_Irregular, Regular, Slightly_Irr~
## $ Land_Contour      <fct> Lvl, Lvl, Lvl, Lvl, Lvl, Lvl, Lvl, HLS, L~
## $ Utilities         <fct> AllPub, AllPub, AllPub, AllPub, AllPub, A~
## $ Lot_Config        <fct> Corner, Inside, Corner, Corner, Inside, I~
## $ Land_Slope        <fct> Gtl, Gtl, Gtl, Gtl, Gtl, Gtl, Gtl, Gtl, G~
## $ Neighborhood      <fct> North_Ames, North_Ames, North_Ames, North~
## $ Condition_1       <fct> Norm, Feedr, Norm, Norm, Norm, Norm, Norm~
## $ Condition_2       <fct> Norm, Norm, Norm, Norm, Norm, Norm, Norm,~
## $ Bldg_Type         <fct> OneFam, OneFam, OneFam, OneFam, OneFam, O~
## $ House_Style       <fct> One_Story, One_Story, One_Story, One_Stor~
## $ Overall_Cond      <fct> Average, Above_Average, Above_Average, Av~
## $ Year_Built        <int> 1960, 1961, 1958, 1968, 1997, 1998, 2001,~
## $ Year_Remod_Add    <int> 1960, 1961, 1958, 1968, 1998, 1998, 2001,~
## $ Roof_Style        <fct> Hip, Gable, Hip, Hip, Gable, Gable, Gable~
## $ Roof_Matl         <fct> CompShg, CompShg, CompShg, CompShg, CompS~
## $ Exterior_1st      <fct> BrkFace, VinylSd, Wd Sdng, BrkFace, Vinyl~
```

Simple linear regression



The plot displays the relationship between ground floor living area and sale price of properties in Ames, Iowa.

Is ground floor living area alone *sufficient* to estimate a property's sale price?

Categorical predictor with two levels

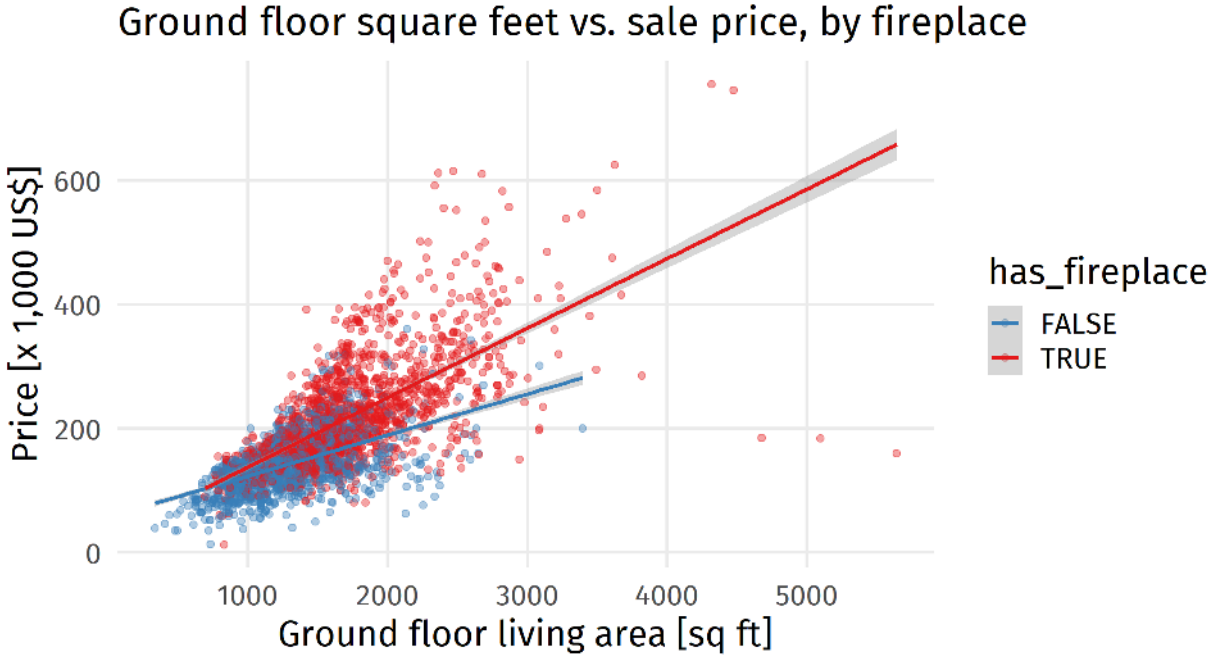
```
ames <- ames %>%
  mutate(has_fireplace =
           as.factor(Fireplaces > 0))
ames %>%
  select(Sale_Price, Fireplaces, has_fireplace) %>%
  print(n = 15)
```

```
## # A tibble: 2,930 x 3
##    Sale_Price Fireplaces has_fireplace
##         <int>      <int> <fct>
##  1     215000          2 TRUE
##  2     105000          0 FALSE
##  3     172000          0 FALSE
##  4     244000          2 TRUE
##  5     189900          1 TRUE
##  6     195500          1 TRUE
##  7     213500          0 FALSE
##  8     191500          0 FALSE
##  9     236500          1 TRUE
## 10     189000          1 TRUE
## 11     175900          1 TRUE
## 12     185000          0 FALSE
## 13     180400          1 TRUE
## 14     171500          1 TRUE
## 15     212000          0 FALSE
## # ... with 2,915 more rows
```

- `has_fireplace = FALSE`: property does not have a fireplace
- `has_fireplace = TRUE`: property has one or more fireplaces

How, if at all, does the relationship between ground floor living area and price of Ames properties vary by whether or not they have a fireplace?

Plot | Code



How, if at all, does the relationship between ground floor living area and price of Ames properties vary by whether or not they have a fireplace?

Plot | Code

```
ggplot(data = ames, aes(x = Gr_Liv_Area, y = Sale_Price/1000, color
  geom_point(alpha = 0.4) +
  geom_smooth(method = "lm") +
  scale_color_brewer(palette = "Set1", direction = -1) +
  labs(x = "Ground floor living area [sq ft]", y = "Price [x 1,000
       title = "Ground floor square feet vs. sale price, by firepla
```

Multiple linear regressions

```
linear_reg() %>%
  set_engine("lm") %>%
  fit(formula = Sale_Price ~ Gr_Liv_Area + has_fireplace, data = ames) %>%
  tidy()
```

```
## # A tibble: 3 x 5
##   term              estimate std.error statistic   p.value
##   <chr>                <dbl>     <dbl>     <dbl>     <dbl>
## 1 (Intercept)        17970.    3166.        5.68 1.52e-  8
## 2 Gr_Liv_Area           97.0      2.22     43.7  5.60e-322
## 3 has_fireplaceTRUE  33714.    2243.       15.0  3.03e- 49
```

Interpretation

- Slope of `Gr_Liv_Area`: *all else held constant*, for each additional square feet a property's ground floor living area is larger, we would expect the sale price to be higher, on average, by ca. 97 US$.
- Slope of `has_fireplaceTRUE`: *all else held constant*, a property with a fire place is, on average, by 33,714 US$ more expensive.
- Intercept: a property with 0 sq ft ground floor living area and without fire places is expected to cost ca. 17,970 US$.

Incorporating model complexity

Recall the coefficient of determination: $R^2 = \frac{MSS}{TSS}$

This measure has an issue: R^2 increases with each additional variable.

Occam's razor or the principle of parsimony

Given two models of equal quality, the simpler model is preferred over the more complex model.

→ Add a variable to the model *only* if its addition increases model quality (predictive power) by a *substantial degree*

Adjusted R^2

The **Adjusted** R^2 contains a term penalizing the addition of each further variable to the model:

$$\begin{aligned}\text{Adjusted } R^2 &= R^2 \cdot \frac{n-1}{n-p-1} \\ &= \frac{MSS}{TSS} \cdot \frac{n-1}{n-p-1}\end{aligned}$$

- n: number of observations
- p: number of predictors / explanatory variables

Adjusted R^2

One vs. two predictors | Counterexample plot | Counterexample R^2 vs. adj. R^2

```
linear_reg() %>% set_engine("lm") %>%
  fit(formula = Sale_Price ~ Gr_Liv_Area, data = ames) %>% glance()
```

```
## # A tibble: 1 x 12
##   r.squared adj.r.squared  sigma statistic p.value    df  logLik    AIC    BIC deviance
##       <dbl>         <dbl>  <dbl>     <dbl>   <dbl> <dbl>   <dbl>  <dbl>  <dbl>    <dbl>
## 1     0.500         0.499 56524.     2923.       0     1 -36218. 72442. 72460.  9.35e12
## # ... with 2 more variables: df.residual <int>, nobs <int>
```

```
linear_reg() %>% set_engine("lm") %>%
  fit(formula = Sale_Price ~ Gr_Liv_Area + has_fireplace, data = ames) %>% glance()
```

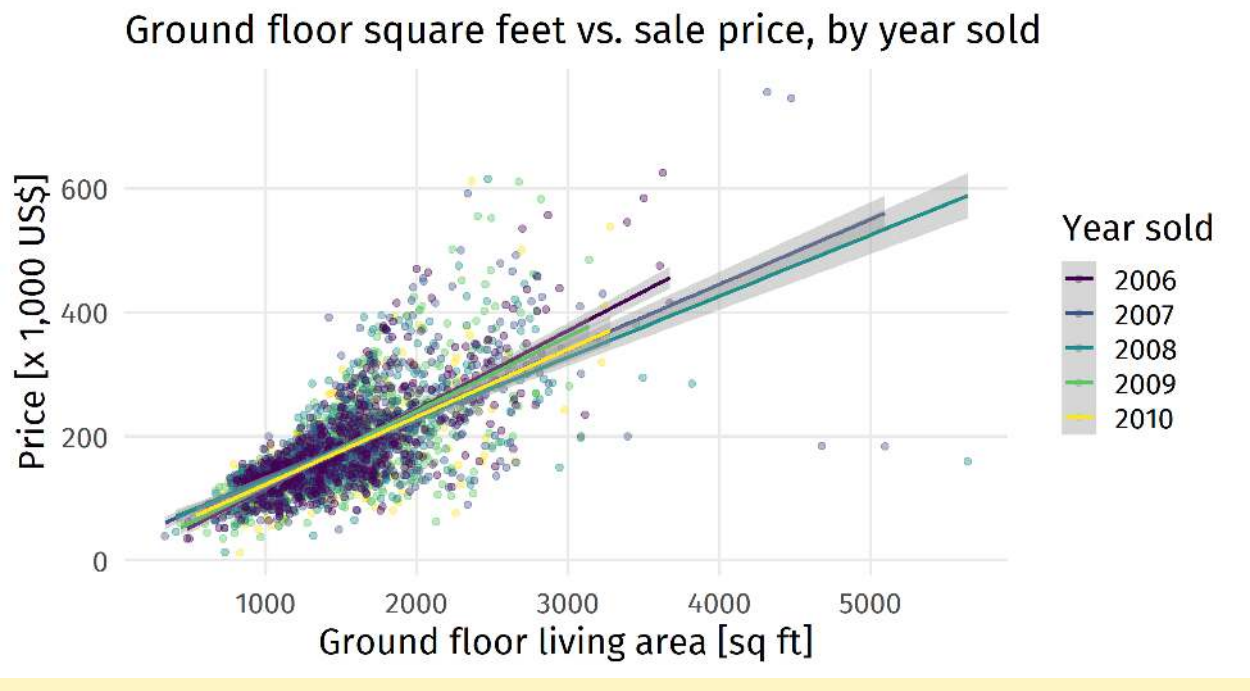
```
## # A tibble: 1 x 12
##   r.squared adj.r.squared  sigma statistic p.value    df  logLik    AIC    BIC deviance
##       <dbl>         <dbl>  <dbl>     <dbl>   <dbl> <dbl>   <dbl>  <dbl>  <dbl>    <dbl>
## 1     0.535         0.535 54471.     1687.       0     2 -36109. 72226. 72250.  8.68e12
## # ... with 2 more variables: df.residual <int>, nobs <int>
```

The addition of `has_fireplace` leads to a better model, both in terms of R^2 and adjusted R^2.

Adjusted R^2

One vs. two predictors | **Counterexample plot** | Counterexample R^2 vs. adj. R^2

Does the addition of `Year_Sold` to the model lead to a better fit?



Adjusted R^2

One vs. two predictors | Counterexample plot | **Counterexample R^2 vs. adj. R^2**

```
linear_reg() %>% set_engine("lm") %>%
  fit(formula = Sale_Price ~ Gr_Liv_Area, data = ames) %>% glance()
```

```
## # A tibble: 1 x 12
##   r.squared adj.r.squared  sigma statistic p.value    df  logLik    AIC    BIC  deviance
##       <dbl>         <dbl>  <dbl>     <dbl>   <dbl> <dbl>   <dbl>  <dbl>  <dbl>     <dbl>
## 1   0.49954       0.49937 56524.    2922.6       0     1 -36218. 72442. 72460. 9.3549e12
## # ... with 2 more variables: df.residual <int>, nobs <int>
```

```
linear_reg() %>% set_engine("lm") %>%
  fit(formula = Sale_Price ~ Gr_Liv_Area + Year_Sold, data = ames) %>% glance()
```

```
## # A tibble: 1 x 12
##   r.squared adj.r.squared  sigma statistic p.value    df  logLik    AIC    BIC  deviance
##       <dbl>         <dbl>  <dbl>     <dbl>   <dbl> <dbl>   <dbl>  <dbl>  <dbl>     <dbl>
## 1   0.49968       0.49934 56526.    1461.6       0     2 -36217. 72443. 72467. 9.3523e12
## # ... with 2 more variables: df.residual <int>, nobs <int>
```

The addition of `Year_Sold` leads to a model with slightly better R^2 (0.4997 > 0.4995). However, since the adjusted R^2 value is lower (0.49934 < 0.49937), the simpler model should be preferred.
Adjusted R^2 doesn't increase because `Year_Sold` does not provide enough new information, or, in other words, is nearly unrelated.

Main vs. interaction effects

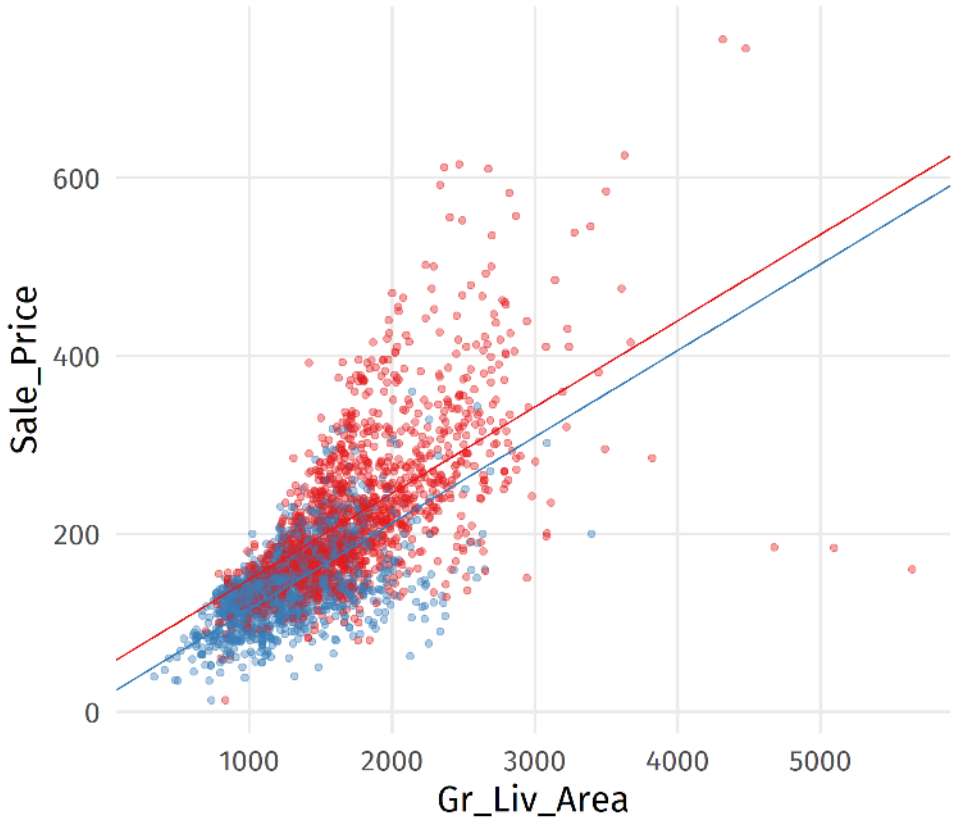
Suppose we want to predict property prices from their living area and presence of a fire place.

Which of the following models is more appropriate?

1. **Main effects:** The rate at which price changes as living area increases is the same for properties with fire places and properties without fire places.
2. **Interaction effects:** The rate at which price changes as living area increases differs between properties with fire places and properties without fire places.

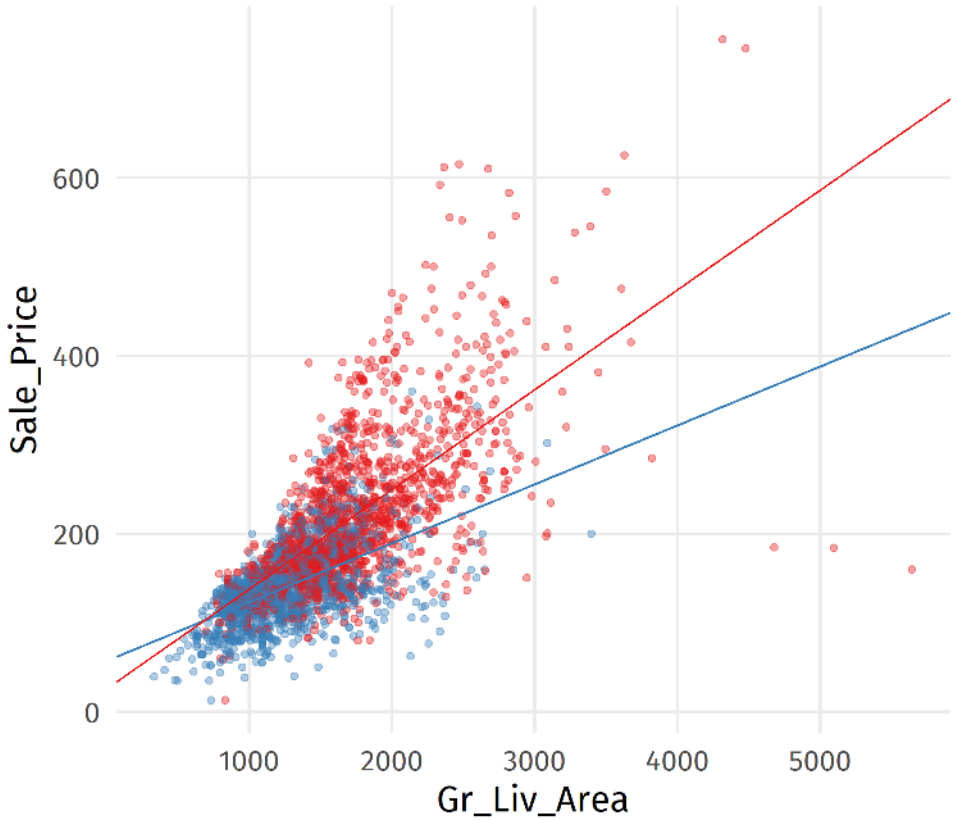
Main effects, parallel slopes

Gr_Liv_Area = Sale_price + has_fireplace



Interaction effects, not parallel slopes

Gr_Liv_Area = Sale_price + has_fireplace + Sale_price * has_fireplace



has_fireplace FALSE TRUE

Model with main effects

```
m1 <- linear_reg() %>%
  set_engine("lm") %>%
  fit(formula = Sale_Price ~ Gr_Liv_Area + has_fireplace, data = ames)
m1
```

```
## parsnip model object
##
## Fit time:  0ms
##
## Call:
## stats::lm(formula = Sale_Price ~ Gr_Liv_Area + has_fireplace,
##     data = data)
##
## Coefficients:
##       (Intercept)        Gr_Liv_Area  has_fireplaceTRUE
##             17970                 97              33714
```

$$\widehat{Sale\_Price} = 17,970 + 97 \cdot Gr\_Liv\_Area + 33,714 \cdot has\_fireplace$$

Model with interaction effects

```
(m2 <- linear_reg() %>% set_engine("lm") %>%
  fit(formula = Sale_Price ~ Gr_Liv_Area * has_fireplace, data = ames))
```

```
## parsnip model object
##
## Fit time:  0ms
##
## Call:
## stats::lm(formula = Sale_Price ~ Gr_Liv_Area * has_fireplace,
##     data = data)
##
## Coefficients:
##                   (Intercept)                    Gr_Liv_Area
##                      57113.38                         66.19
##              has_fireplaceTRUE  Gr_Liv_Area:has_fireplaceTRUE
##                     -31309.47                         45.90
```

$$\widehat{Sale\_Price} = 57,113 \quad +66 \cdot Gr\_Liv\_Area - 31,309 \cdot has\_fireplace \quad +46 \cdot Gr\_Liv\_Area \cdot has\_fireplace$$

$$\begin{aligned} \widehat{Sale\_Price}(has\_fireplace = 1) &= 57,113 \quad +66 \cdot Gr\_Liv\_Area - 31,309 \cdot 1 \quad +46 \cdot Gr\_Liv\_Area \cdot 1 \\ &= 25,804 \quad +112 \cdot Gr\_Liv\_Area \end{aligned}$$

$$\begin{aligned} \widehat{Sale\_Price}(has\_fireplace = 0) &= 57,113 \quad +66 \cdot Gr\_Liv\_Area - 31,309 \cdot 0 \quad +46 \cdot Gr\_Liv\_Area \cdot 0 \\ &= 57,113 \quad +66 \cdot Gr\_Liv\_Area \end{aligned}$$

Model selection

```
glance(m1)
```

```
## # A tibble: 1 x 12
##   r.squared adj.r.squared  sigma statistic p.value    df  logLik    AIC    BIC deviance
##       <dbl>         <dbl>  <dbl>     <dbl>   <dbl> <dbl>   <dbl>  <dbl>  <dbl>    <dbl>
## 1     0.535         0.535 54471.     1687.       0     2 -36109. 72226. 72250.  8.68e12
## # ... with 2 more variables: df.residual <int>, nobs <int>
```

```
glance(m2)
```

```
## # A tibble: 1 x 12
##   r.squared adj.r.squared  sigma statistic p.value    df  logLik    AIC    BIC deviance
##       <dbl>         <dbl>  <dbl>     <dbl>   <dbl> <dbl>   <dbl>  <dbl>  <dbl>    <dbl>
## 1     0.550         0.550 53594.     1194.       0     3 -36061. 72132. 72162.  8.40e12
## # ... with 2 more variables: df.residual <int>, nobs <int>
```

Including the interaction effect of `Gr_Liv_Area` and `has_fireplace` yields a better model.

```
glance(m2)$adj.r.squared > glance(m1)$adj.r.squared
```

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

Models with categorical predictors

Dwelling type | Dummy variables | Relationsh. between living area, dwelling type & sale price | Plot

```
janitor::tabyl(ames, Bldg_Type)
```

```
##  Bldg_Type    n    percent
##     OneFam 2425 0.82764505
##   TwoFmCon   62 0.02116041
##     Duplex  109 0.03720137
##      Twnhs  101 0.03447099
##     TwnhsE  233 0.07952218
```

The explanatory variable `Bldg_Type` is a factor with five levels.

Multinominal predictors, i.e., variables with more than two categories, will be automatically encoded as **dummy variables**.

Models with categorical predictors

	Dummy variables			
Bldg_Type	**TwoFmCon**	**Duplex**	**Twnhs**	**TwnhsE**
OneFam	0	0	0	0
TwoFmCon	1	0	0	0
Duplex	0	1	0	0
Twnhs	0	0	1	0
TwnhsE	0	0	0	1

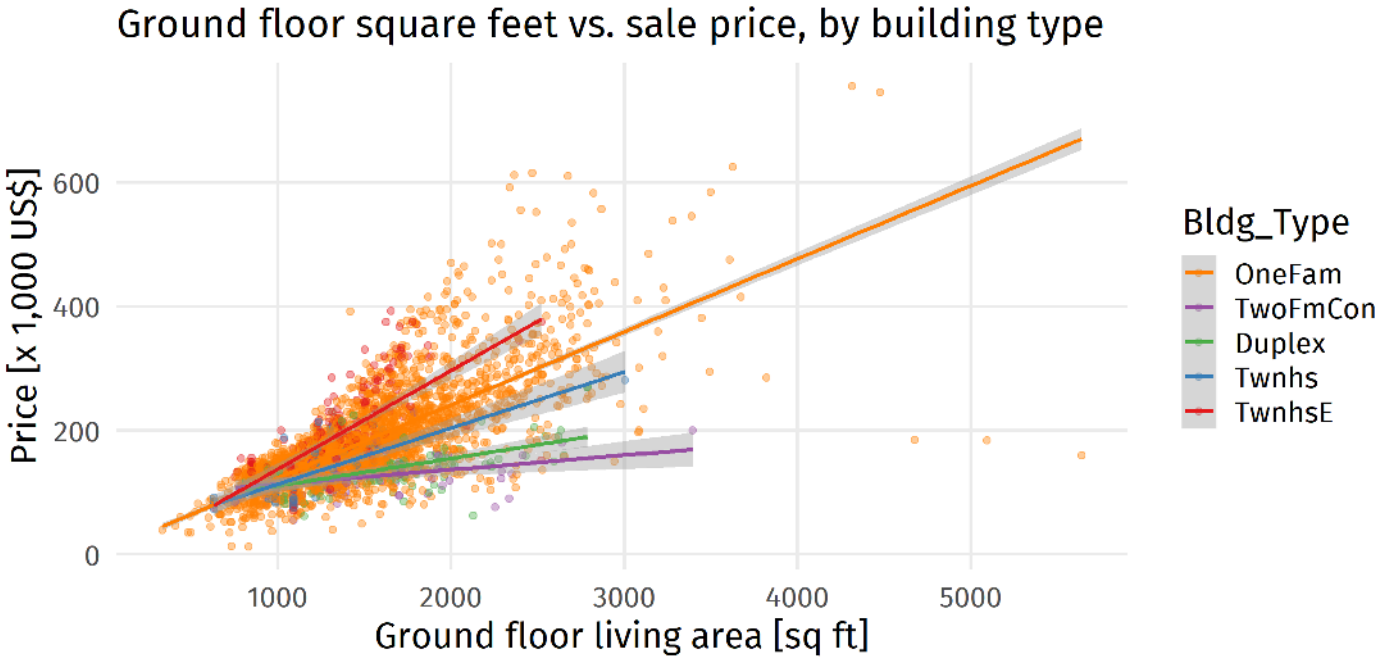
Note that only four dummy variables are created. The building type of properties with *baseline category* `OneFam` can be inferred from the other dummy variables.

Models with categorical predictors


```
linear_reg() %>%
  set_engine("lm") %>%
  fit(Sale_Price ~ Gr_Liv_Area + Bldg_Type, data = ames) %>%
  tidy()
```

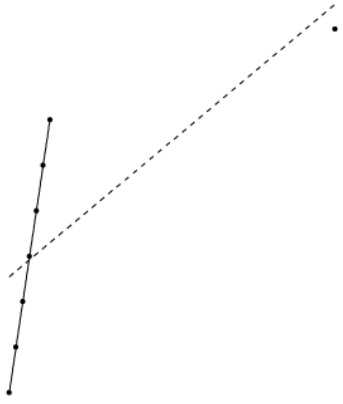
```
## # A tibble: 6 x 5
##   term              estimate std.error statistic  p.value
##   <chr>                <dbl>     <dbl>     <dbl>    <dbl>
## 1 (Intercept)         11935.   3223.        3.70 2.17e- 4
## 2 Gr_Liv_Area           114.      2.00     57.0  0.
## 3 Bldg_TypeTwoFmCon  -59300.   6951.       -8.53 2.29e-17
## 4 Bldg_TypeDuplex    -60746.   5299.      -11.5  8.58e-30
## 5 Bldg_TypeTwnhs     -18059.   5515.       -3.27 1.07e- 3
## 6 Bldg_TypeTwnhsE     27477.   3723.        7.38 2.06e-13
```

Models with categorical predictors



Further observation-level diagnostics

- For which properties does the model have difficulty estimating the sale price?
- Which properties are **outliers** with respect to the sample distribution?
- Which properties have a high **influence** on the model?



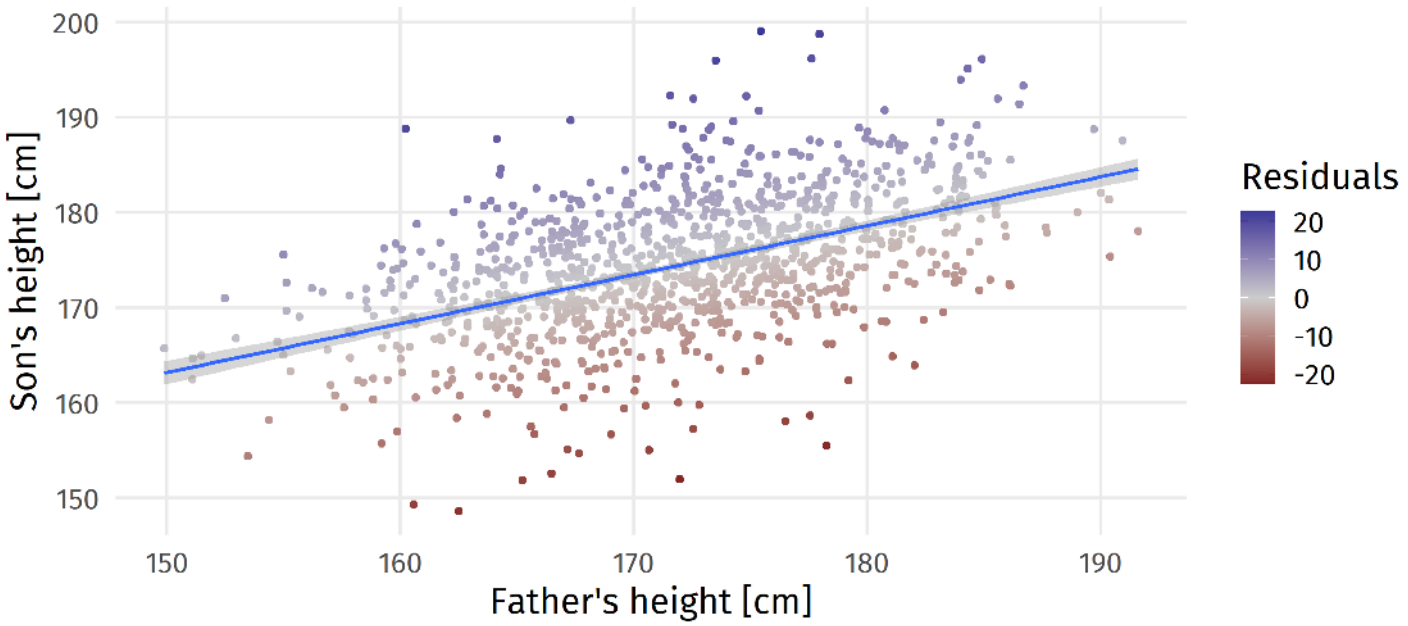
```
(augmented_mod <- augment(model$fit))
```

```
## # A tibble: 1,078 x 8
##    sheight fheight .fitted .resid     .hat .sigma  .cooksd .std.resid
##      <dbl>   <dbl>   <dbl>  <dbl>    <dbl>  <dbl>    <dbl>      <dbl>
##  1    152.    165.    171. -19.2  0.00179    6.16 0.00860      -3.10
##  2    161.    161.    169.  -8.10 0.00335    6.19 0.00289      -1.31
##  3    161.    165.    171. -10.0  0.00185    6.18 0.00242      -1.62
##  4    159.    167.    172. -12.4  0.00139    6.18 0.00281      -2.01
##  5    163.    155.    166.  -2.63 0.00621    6.19 0.000568     -0.426
##  6    163.    160.    168.  -5.19 0.00361    6.19 0.00128      -0.840
##  7    163.    166.    171.  -8.66 0.00159    6.19 0.00156      -1.40
##  8    163.    164.    171.  -8.04 0.00201    6.19 0.00170      -1.30
##  9    164.    168.    172.  -8.22 0.00125    6.19 0.00111      -1.33
## 10    163.    170.    174. -11.0  0.000991   6.18 0.00157      -1.78
## # ... with 1,068 more rows
```

Properties with high residuals

Which son heights are difficult to predict?

Plot | Code



Residuals are measured in the same units as the response. To obtain **standard residuals**, we divide the residuals by their standard deviation.

Properties with high residuals

Which son heights are difficult to predict?

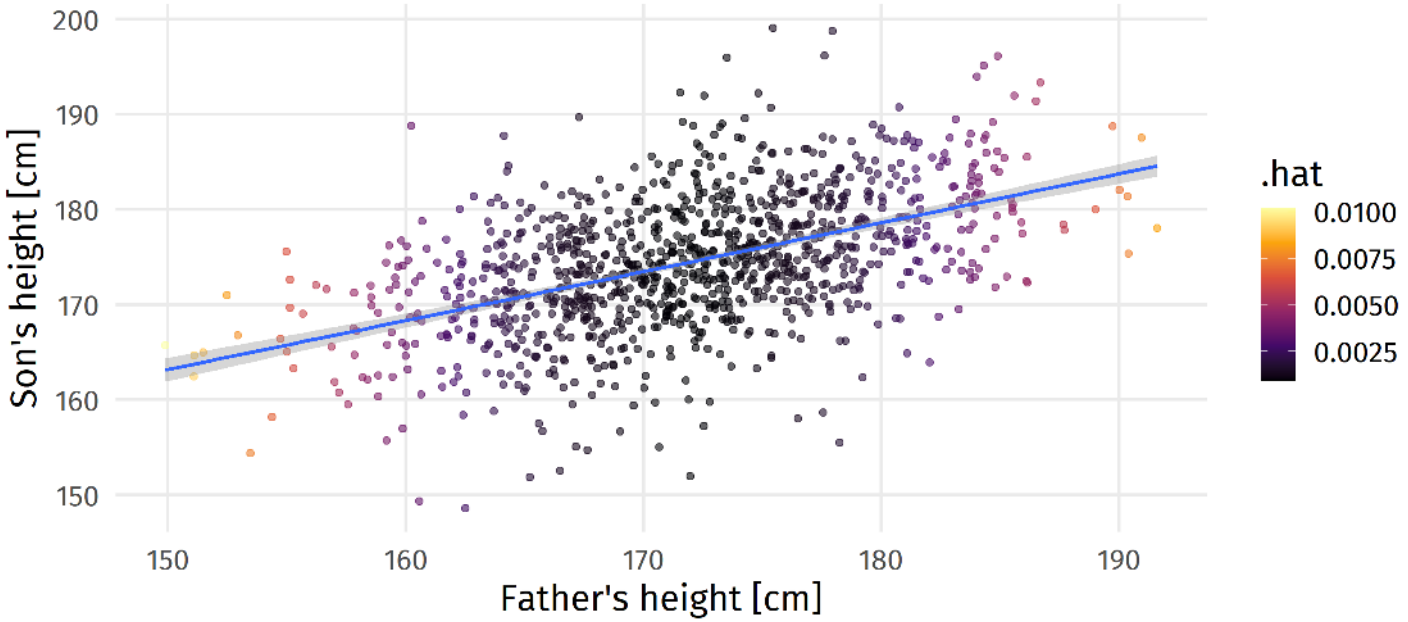
Plot | Code

```
ggplot(augmented_mod, aes(x = fheight, y = sheight)) +
  geom_point(aes(color = .resid)) +
  geom_smooth(method = "lm") +
  scale_color_gradient2(mid = "gray80") +
  labs(x = "Father's height [cm]", y = "Son's height [cm]", color = "Residuals")
```

High leverage properties

Leverage (or **hat values**) measures the outlierness of an observation based on the difference to the general distribution of the sample.

Plot | Code



The range of hat values is between 0 (very close to other observations) and 1 (far away from other observations).

High leverage properties

Leverage (or **hat values**) measures the outlierness of an observation based on the difference to the general distribution of the sample.

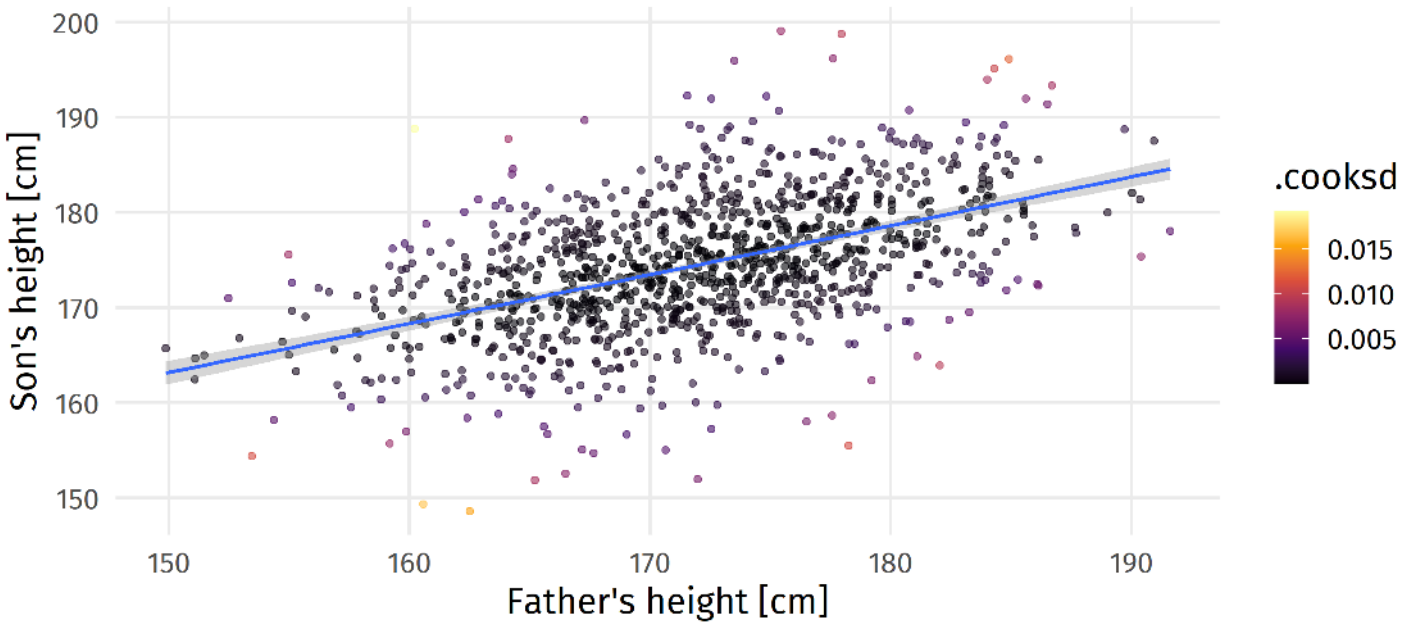
Plot | Code

```
ggplot(augmented_mod, aes(fheight, sheight)) +
  geom_point(aes(color = .hat), alpha = 0.6) +
  geom_smooth(method = "lm") +
  scale_color_viridis_c(option = "inferno") +
  labs(x = "Father's height [cm]", y = "Son's height [cm]")
```

"Influential" properties

Cook's distance measures the effect of removing a given observation on the change of the model.

Plot | Code



As a rule of thumb, Cook's distance values greater than 1 indicate observations with a high influence on the model.

In general, for observations with high Cook distance, one must weigh whether (a) they represent natural variation and should therefore be retained, or (b) they are outliers that are better removed before fitting

"Influential" properties

Cook's distance measures the effect of removing a given observation on the change of the model.

Plot | Code

```
ggplot(augmented_mod, aes(fheight, sheight)) +
  geom_point(aes(color = .cooksd), alpha = 0.6) +
  geom_smooth(method = "lm") +
  scale_color_viridis_c(option = "inferno") +
  labs(x = "Father's height [cm]", y = "Son's height [cm]")
```

Assumptions of linear regression

The list of assumptions for linear regression models includes:

- **linearity**, i.e., "straight-line-relationship" between predictors and response
- **homoscedasticity**, i.e., constant variance of residual terms
- no perfect **collinearity** (perfect linear relationship between two or more predictors)
- predictors are uncorrelated with variables that are not included in the model
- **outliers**, **high-leverage** points

Considerations for high-dimensional datasets 1/2

Two popular methods for datasets with a high number of predictor variables are **Ridge** regression and **LASSO** regression. They introduce a **shrinkage** term in their optimization functions.

Standard least squares estimate minimizes:

$$RSS = \sum_{i=1}^{n} \left(y_i - \left(\hat{\beta}_0 + \hat{\beta}_1 x_i \right) \right)^2$$

Ridge regression: penalizes high $\hat{\beta}$ values

$$RSS = \sum_{i=1}^{n} \left(y_i - \left(\hat{\beta}_0 + \hat{\beta}_1 x_i \right) \right)^2 + \lambda \sum_{j=1}^{p} \beta_j^2 = RSS + \lambda \sum_{j=1}^{p} \beta_j^2$$

→ implemented in `glmnet::glmnet()`: set `alpha=0`

Advantage: **more stable than least squares** when the number of predictors p is almost as large as the number of observations n

Considerations for high-dimensional datasets 2/2

Ridge regression: penalizes high $\hat{\beta}$ values

$$RSS = \sum_{i=1}^{n} \left(y_i - \left(\hat{\beta}_0 + \hat{\beta}_1 x_i \right) \right)^2 + \lambda \sum_{j=1}^{p} \beta_j^2 = RSS + \lambda \sum_{j=1}^{p} \beta_j^2$$

LASSO regression: penalizes high $\hat{\beta}$ values

$$RSS = \sum_{i=1}^{n} \left(y_i - \left(\hat{\beta}_0 + \hat{\beta}_1 x_i \right) \right)^2 + \lambda \sum_{j=1}^{p} |\beta_j| = RSS + \lambda \sum_{j=1}^{p} |\beta_j|$$

→ implemented in `glmnet::glmnet()`: set `alpha=1`

Advantage: forces some coefficients to be **exactly equal to 0** which means that the model is easier to interpret

Other regression models in `R`

- **Generalized linear models** (`stats::glm()`): allow for discrete responses, e.g. logistic regression
- **Generalized additive models** (`mgcv::gam()`): allow for non-linear transformation of predictors
- **Robust linear models** (`MASS::rlm()`): distance function reduces sensitivity to outliers

Session info

```
##  setting  value
##  version  R version 4.0.4 (2021-02-15)
##  os       Windows 10 x64
##  system   x86_64, mingw32
##  ui       RTerm
##  language (EN)
##  collate  English_United States.1252
##  ctype    English_United States.1252
##  tz       Europe/Berlin
##  date     2021-03-29
```

package	version	date	source
airports	0.1.0	2020-06-29	CRAN (R 4.0.3)
AmesHousing	0.0.4	2020-06-23	CRAN (R 4.0.3)
broom	0.7.5	2021-02-19	CRAN (R 4.0.3)
cherryblossom	0.1.0	2020-06-25	CRAN (R 4.0.3)
dials	0.0.9	2020-09-16	CRAN (R 4.0.3)
dplyr	1.0.5	2021-03-05	CRAN (R 4.0.4)
forcats	0.5.1	2021-01-27	CRAN (R 4.0.3)
ggplot2	3.3.3	2020-12-30	CRAN (R 4.0.3)
infer	0.5.4	2021-01-13	CRAN (R 4.0.3)
kableExtra	1.3.4	2021-02-20	CRAN (R 4.0.3)
modeldata	0.1.0	2020-10-22	CRAN (R 4.0.3)
openintro	2.0.0	2020-07-03	CRAN (R 4.0.3)
parsnip	0.1.5	2021-01-19	CRAN (R 4.0.3)

package	version	date	source
purrr	0.3.4	2020-04-17	CRAN (R 4.0.2)
readr	1.4.0	2020-10-05	CRAN (R 4.0.3)
recipes	0.1.15	2020-11-11	CRAN (R 4.0.3)
rsample	0.0.9	2021-02-17	CRAN (R 4.0.4)
scales	1.1.1	2020-05-11	CRAN (R 4.0.2)
stringr	1.4.0	2019-02-10	CRAN (R 4.0.2)
tibble	3.1.0	2021-02-25	CRAN (R 4.0.3)
tidymodels	0.1.2	2020-11-22	CRAN (R 4.0.3)
tidyr	1.1.3	2021-03-03	CRAN (R 4.0.4)
tidyverse	1.3.0	2019-11-21	CRAN (R 4.0.2)
tune	0.1.2	2020-11-17	CRAN (R 4.0.3)
usdata	0.1.0	2020-06-30	CRAN (R 4.0.3)
workflows	0.2.2	2021-03-10	CRAN (R 4.0.4)


Thank you! Questions?
Photo courtesy of Stefan Berger