Linear Regression

Alexandre Sauze

2023-12-03

Section 1: Introduction to Regression

Install the required packages: - Lahman - tidyverse - dslabs

```
library(Lahman)
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.4     v readr     2.1.4
## v forcats   1.0.0     v stringr   1.5.1
## v ggplot2   3.4.4     v tibble    3.2.1
## v lubridate 1.9.3     v tidyr     1.3.0
## v purrr     1.0.2
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become error
```

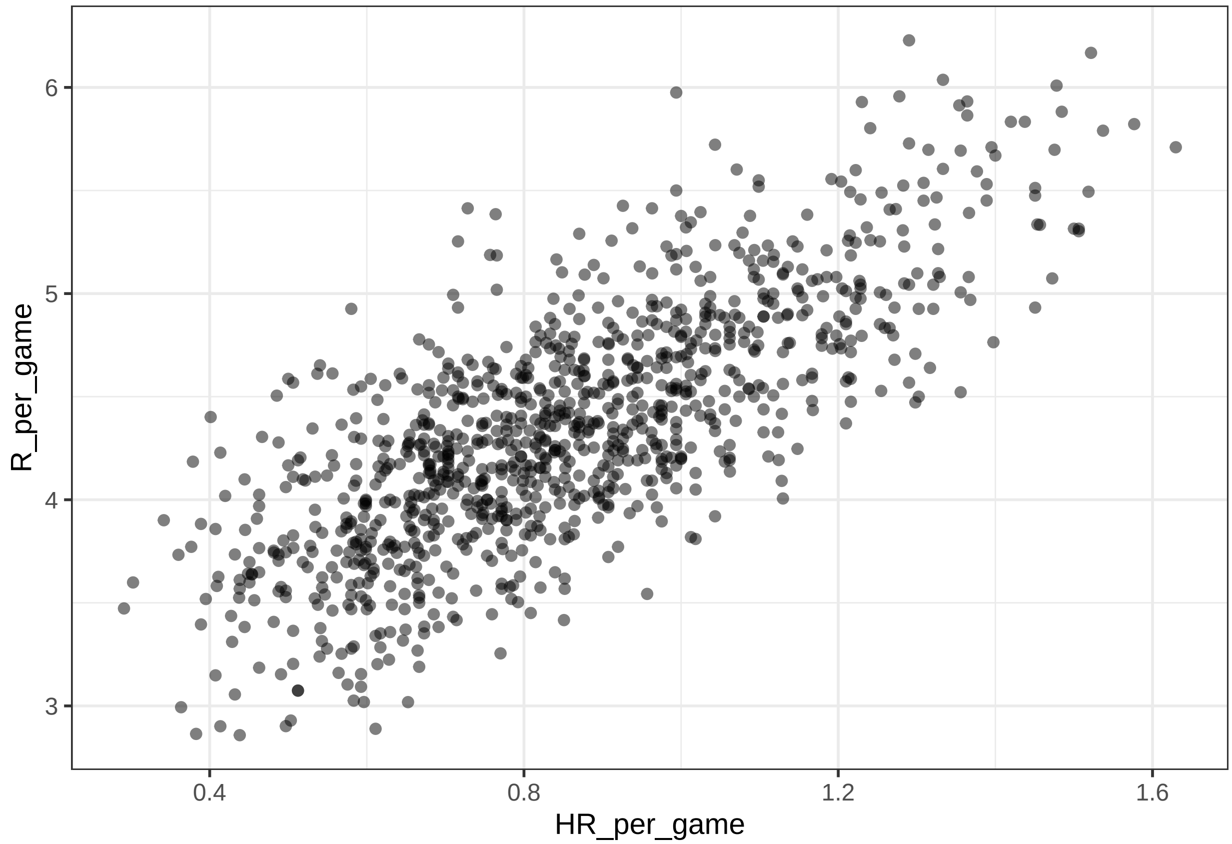
```
library(dslabs)
ds_theme_set()
```

1.1: Baseball as a Motivating Example

Bases on Balls or Stolen Bases?

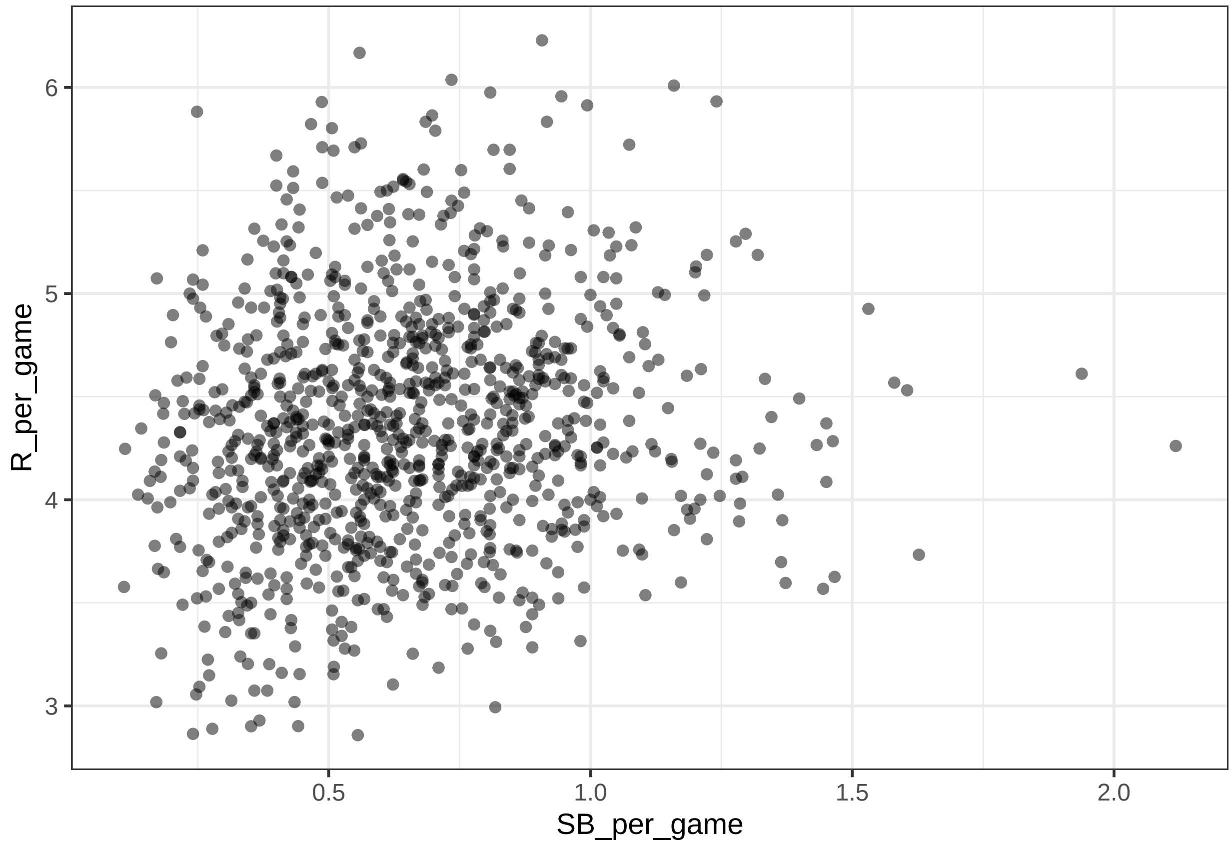
Code: Scatterplot of the relationship between HRs and wins

```
Teams %>% filter(yearID %in% 1961:2001) %>%
  mutate(HR_per_game = HR / G, R_per_game = R / G) %>%
  ggplot(aes(HR_per_game, R_per_game)) +
  geom_point(alpha = 0.5)
```



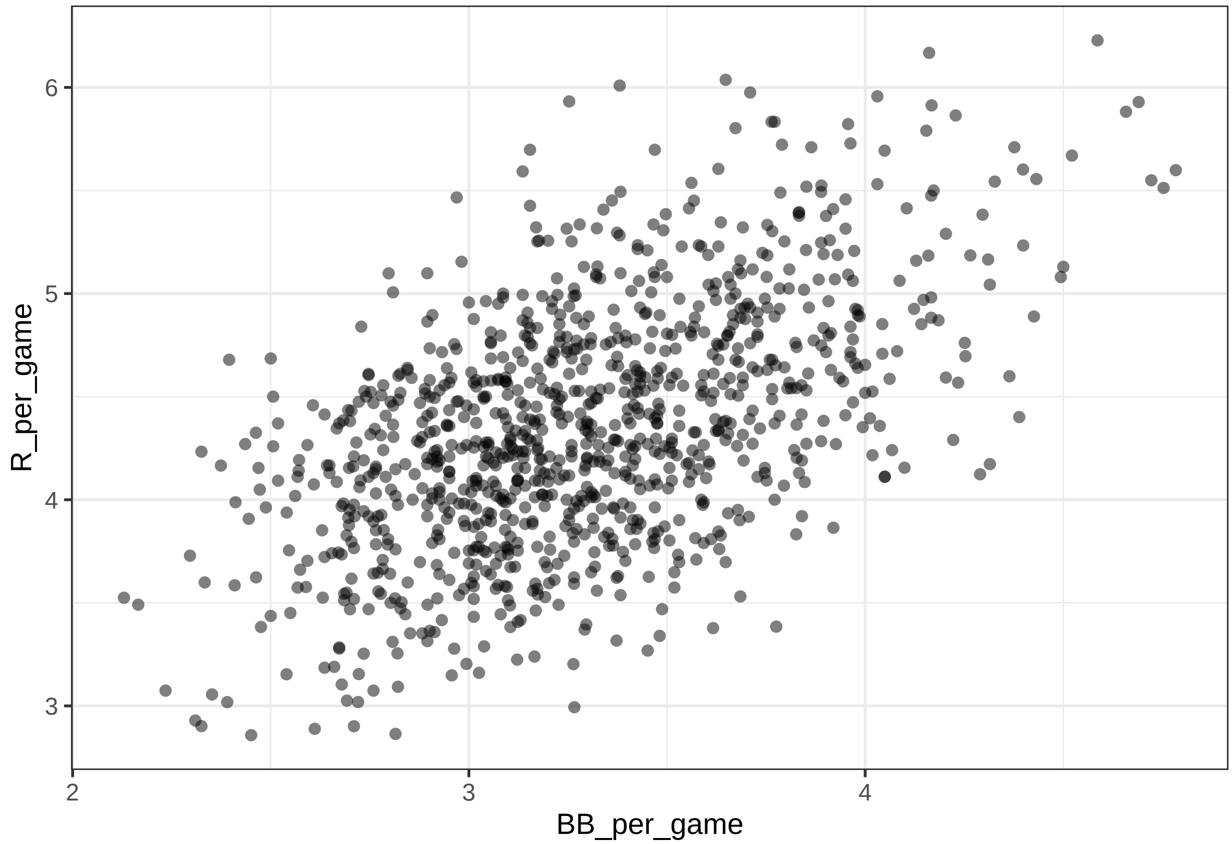
Code: Scatterplot of the relationship between stolen bases and wins

```
Teams %>% filter(yearID %in% 1961:2001) %>%
    mutate(SB_per_game = SB / G, R_per_game = R / G) %>%
    ggplot(aes(SB_per_game, R_per_game)) +
    geom_point(alpha = 0.5)
```



Code: Scatterplot of the relationship between bases on balls and runs

```
Teams %>% filter(yearID %in% 1961:2001) %>%
    mutate(BB_per_game = BB / G, R_per_game = R / G) %>%
    ggplot(aes(BB_per_game, R_per_game)) +
    geom_point(alpha = 0.5)
```



Assessment: Baseball as a Motivating Example

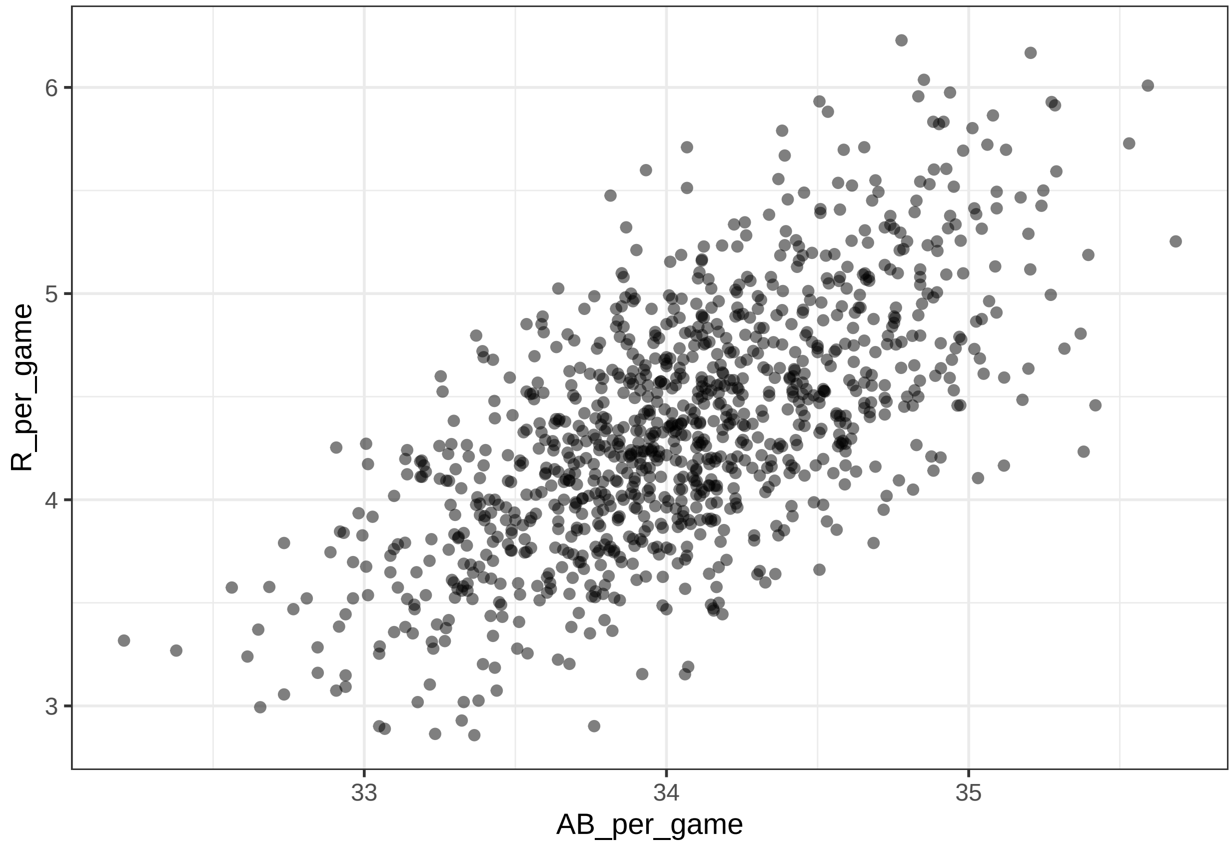
Question 1 **Q.** What is the application of statistics and data science to baseball called? **A.** Sabermetrics

Question 2 **Q.** Which of the following outcomes is not included in the batting average? **A.** A base on balls

Question 3 **Q.** Why do we consider team statistics as well as individual player statistics? **A.** The success of any individual player also depends on the strength of their team.

Question 4 **Q.** You want to know whether teams with more at-bats per game have more runs per game. What R code below correctly makes a scatter plot for this relationship? **A.**

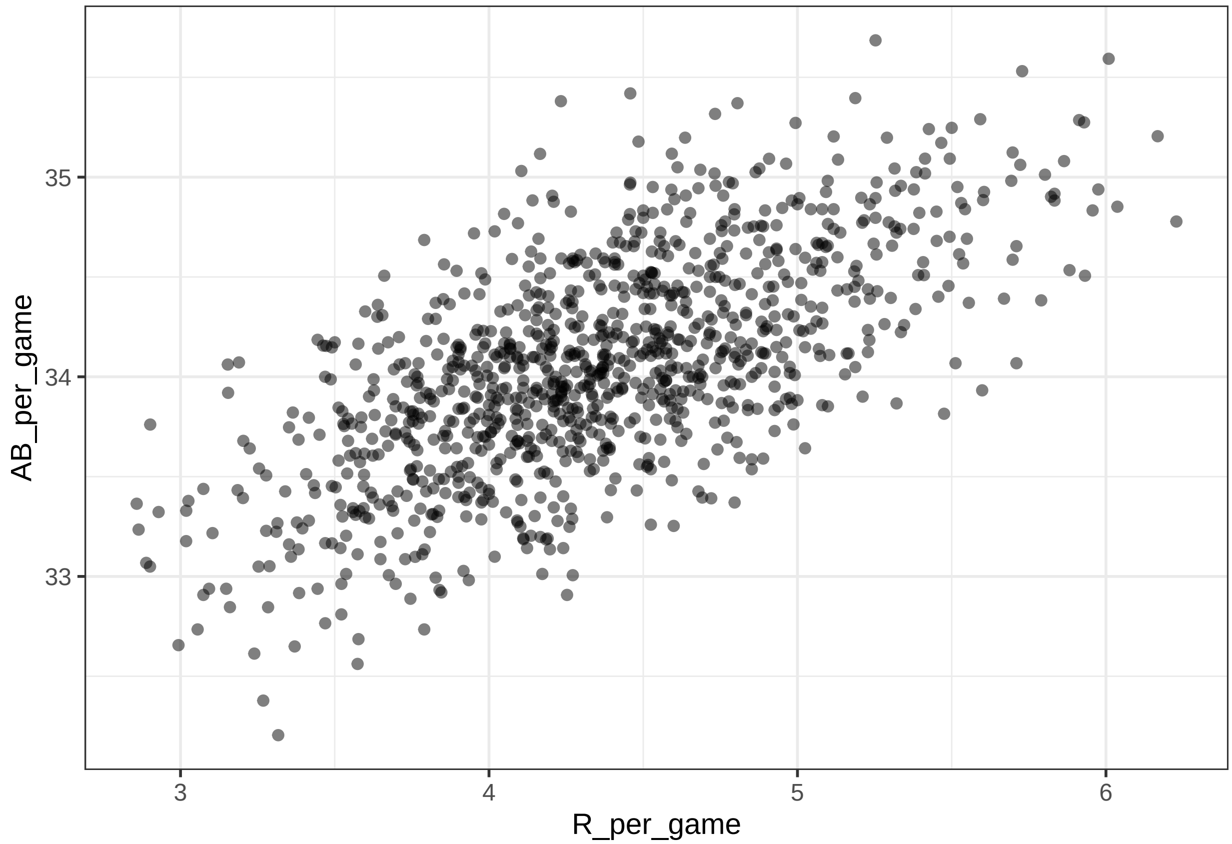
```
Teams %>% filter(yearID %in% 1961:2001 ) %>%
    mutate(AB_per_game = AB/G, R_per_game = R/G) %>%
    ggplot(aes(AB_per_game, R_per_game)) +
    geom_point(alpha = 0.5)
```



Question 5 **Q.** What does the variable "SOA" stand for in the Teams table? **A.** strikeouts by pitchers

Question 6 **Q.** Load the Lahman library. Filter the Teams data frame to include years from 1961 to 2001. Make a scatterplot of runs per game versus at bats (AB) per game. Which of the following is true? **A.**

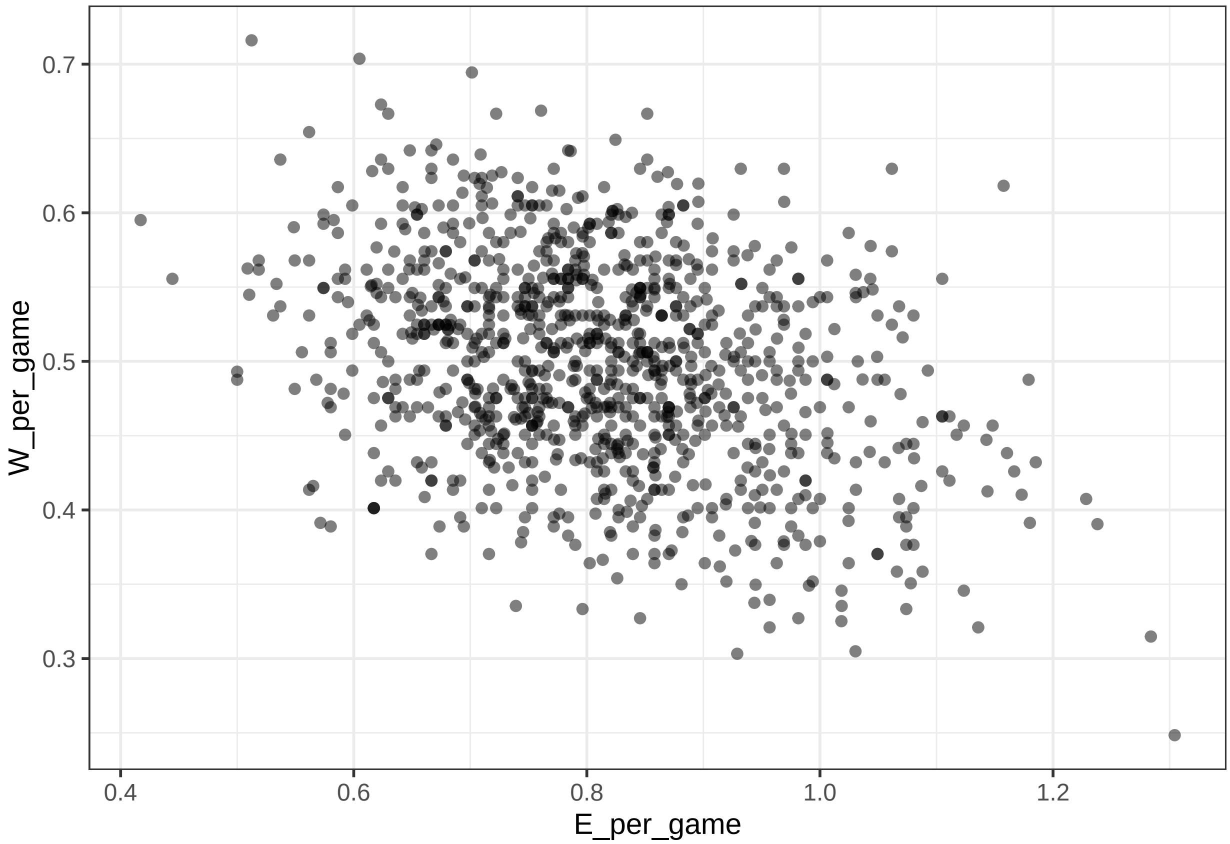
```
Teams %>% filter(yearID %in% 1961:2001) %>%
  mutate(R_per_game = R/G, AB_per_game = AB/G) %>%
  ggplot(aes(R_per_game, AB_per_game)) +
  geom_point(alpha = 0.5)
```



As the number of at bats per game increases, the number of runs per game tends to increase.

Question 7 **Q.** Use the filtered Teams data frame from Question 6. Make a scatterplot of win rate (number of wins per game) versus number of fielding errors (E) per game. Which of the following is true? **A.**

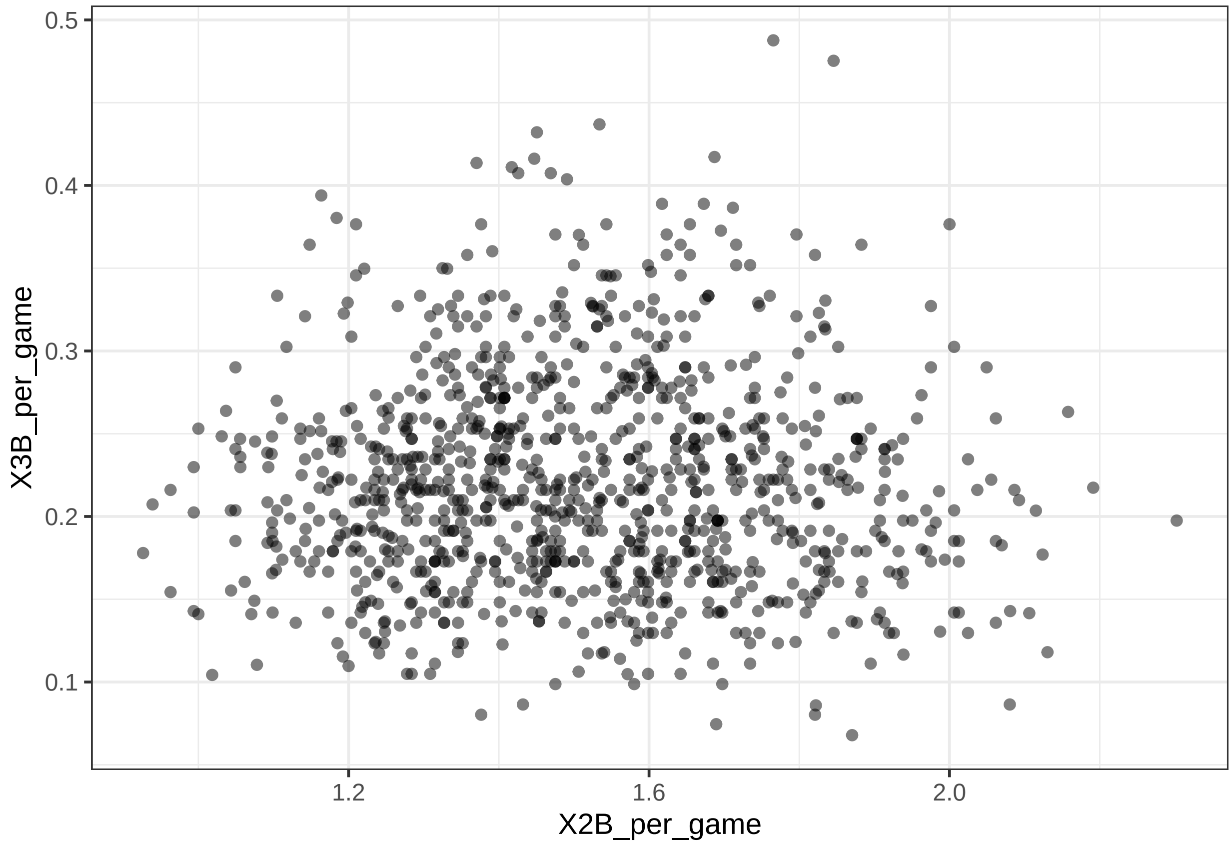
```
Teams %>% filter(yearID %in% 1961:2001) %>%
  mutate(W_per_game = W/G, E_per_game = E/G) %>%
  ggplot(aes(E_per_game, W_per_game)) +
  geom_point(alpha = 0.5)
```



As the number of errors per game increases, the win rate tends to decrease.

Question 8 **Q.** Use the filtered Teams data frame from Question 6. Make a scatterplot of triples (X3B) per game versus doubles (X2B) per game. Which of the following is true? **A.**

```
Teams %>% filter(yearID %in% 1961:2001) %>%
  mutate(X3B_per_game = X3B/G, X2B_per_game = X2B/G) %>%
  ggplot(aes(X2B_per_game, X3B_per_game)) +
  geom_point(alpha = 0.5)
```



There is no clear relationship between doubles per game and triples per game.

1.2: Correlation

Correlation

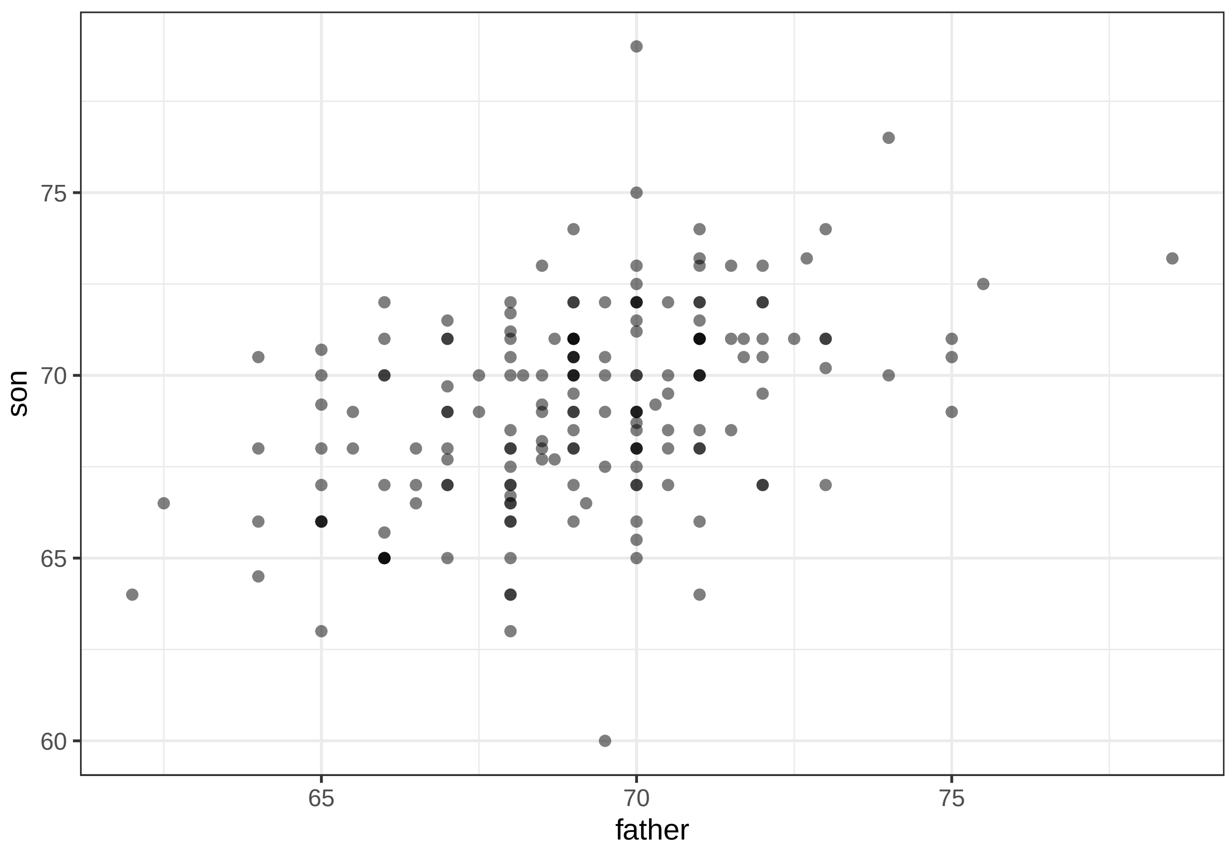
- Galton tried to predict sons' heights based on fathers' heights.
- The mean and standard errors are insufficient for describing an important characteristic of the data: the trend that the taller the father, the taller the son.
- The correlation coefficient is an informative summary of how two variables move together that can be used to predict one variable using the other.

```
# create the dataset
library(tidyverse)
library(HistData)
data("GaltonFamilies")
set.seed(1983)
galton_heights <- GaltonFamilies %>%
  filter(gender == "male") %>%
  group_by(family) %>%
  sample_n(1) %>%
  ungroup() %>%
  select(father, childHeight) %>%
  rename(son = childHeight)

# means and standard deviations
```

```r
galton_heights %>%
    summarize(mean(father), sd(father), mean(son), sd(son))
```

```
## # A tibble: 1 x 4
##   `mean(father)` `sd(father)` `mean(son)` `sd(son)`
##            <dbl>        <dbl>       <dbl>     <dbl>
## 1           69.1         2.55        69.2      2.71
```

```r
# scatterplot of father and son heights
galton_heights %>%
    ggplot(aes(father, son)) +
    geom_point(alpha = 0.5)
```



Correlation Coefficient

$\rho = \frac{1}{n}\sum_{i=1}^{n}\left(\frac{x_i-\mu_x}{\sigma_x}\right)\left(\frac{x_i-\mu_y}{\sigma_y}\right)$

- The correlation coefficient is defined for a list of pairs $(x_1, y_2), ..., (x_n, y_n)$ as the sum of the product of the standarized values: $\left(\frac{x_i-\mu_x}{\sigma_x}\right)\left(\frac{x_i-\mu_x}{\sigma_x}\right)$ for each observation i. The product term is positive when both the standardized x and y are positive or when they are both negative, and the product term is negative when the standardized x and y have different signs (one is positive and one is negative).
- The greek letter ρ is typically used to denote the correlation.

- The correlation coefficient essentially conveys how two variables move together. ρ is always between -1 and 1.

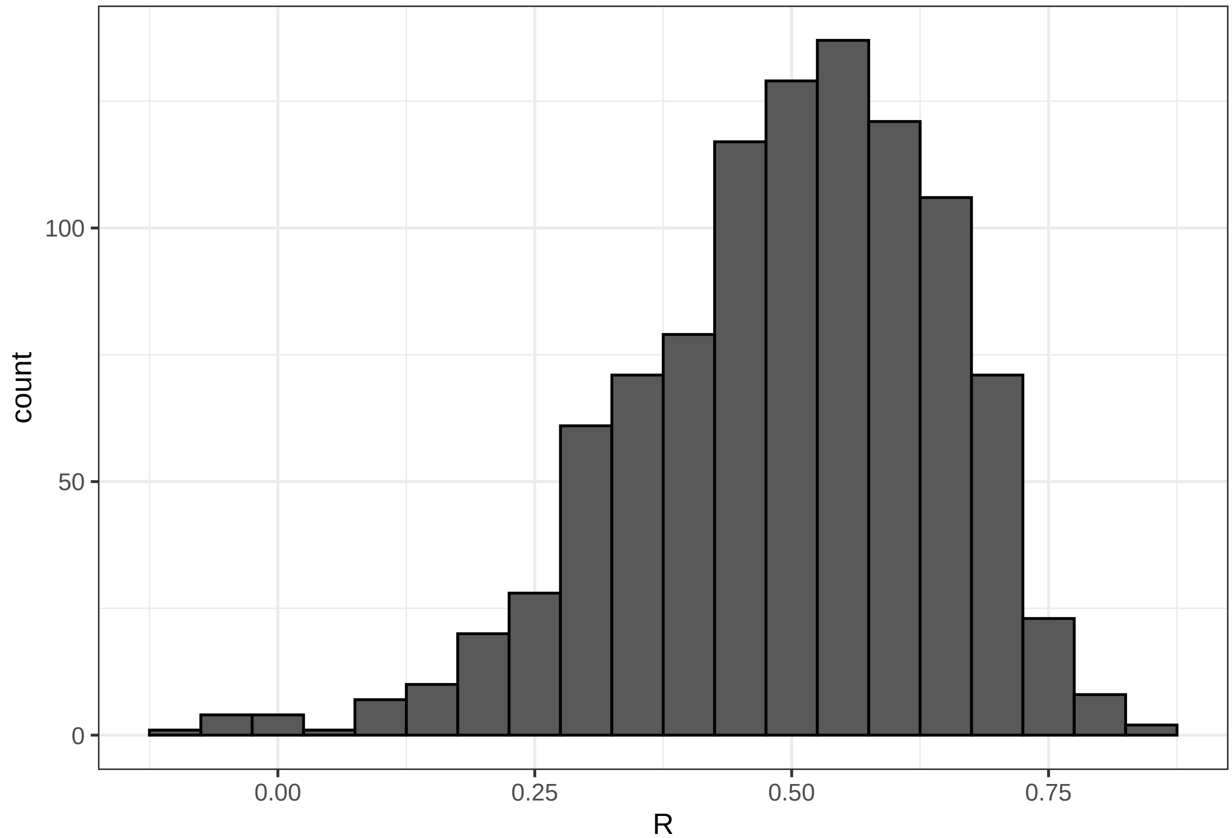
```
galton_heights <- GaltonFamilies %>%
  filter(childNum == 1 & gender == "male") %>%
  select(father,childHeight) %>%
  rename(son = childHeight)

galton_heights %>% summarize(cor(father, son))
```

```
##   cor(father, son)
## 1        0.5007248
```

Sample Correlation is a Random Variable

- The correlation that we compute and use as a summary is a random variable.
- When interpreting correlations, it is important to remember that correlations derived from samples are estimates containing uncertainty.
- Because the sample correlation is an average of independent draws, the central limit theorem applies.

```
# compute sample correlation
my_sample <- slice_sample(galton_heights, n = 25, replace = TRUE)

R <- my_sample %>% summarize(cor(father, son))

# Monte Carlo simulation to show distribution of sample correlation
B <- 1000
N <- 25
R <- replicate(B, {
  slice_sample(galton_heights, n = N, replace = TRUE) %>%
    summarize(r=cor(father, son)) %>% .$r
})
data.frame(R) %>% ggplot(aes(R)) + geom_histogram(binwidth = 0.05, color = "black")
```



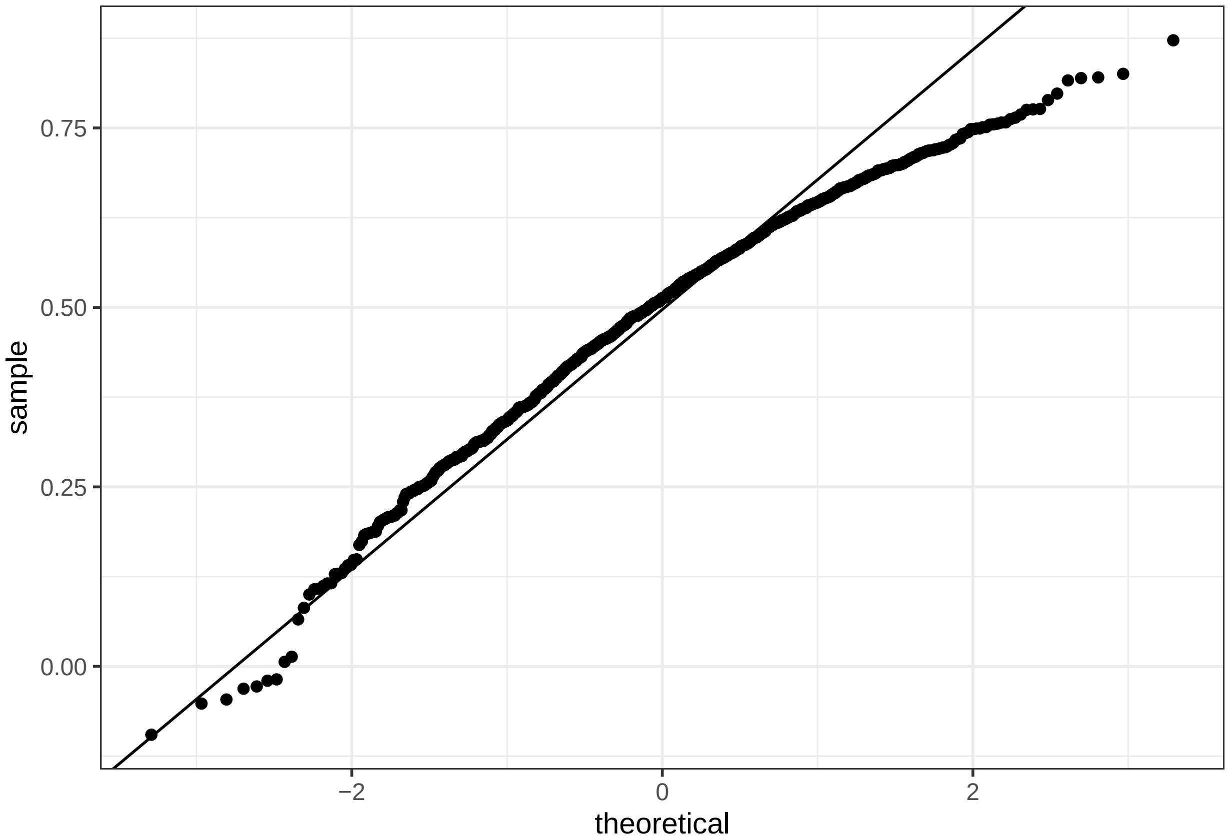

```r
# expected value is the population correlation
mean(R)
```

```
## [1] 0.4970997
```

```r
# standard error is high relative to its size
sd(R)
```

```
## [1] 0.1512451
```

```r
# QQ-plot to evaluate whether N is large enough
data.frame(R) %>%
    ggplot(aes(sample = R)) +
    stat_qq() +
    geom_abline(intercept = mean(R), slope = sqrt((1-mean(R)^2)/(N-2)))
```



Assessment: Correlation

Question 4 **Q.** Instead of running a Monte Carlo simulation with a sample size of 25 from the 179 father-son pairs described in the videos, imagine we now run the simulation with a sample size of 50. Note: You do not need to run any code to determine the answer to this exercise. Would you expect the mean of the sample correlation to increase, decrease, or stay approximately the same? **A.** Stay approximately the same

Question 5 **Q.** Instead of running a Monte Carlo simulation with a sample size of 25 from the 179 father-son pairs described in the videos, imagine we now run the simulation with a sample size of 50. Note: You do not need to run any code to determine the answer to this exercise. Would you expect the standard deviation of the sample correlation to increase, decrease, or stay approximately the same? **A.** Decrease

Question 7 **Q.** Load the Lahman library. Filter the Teams data frame to include years from 1961 to 2001. What is the correlation coefficient between number of runs per game and number of at bats per game? **A.**

```
Teams %>% filter(yearID %in% 1961:2001) %>%
  summarize(r=cor(R/G, AB/G))
```

```
##           r
## 1 0.6580976
```

Question 8 **Q.** What is the correlation coefficient between win rate (number of wins per game) and number of errors per game? **A.**

```
Teams %>% filter(yearID %in% 1961:2001) %>%
  summarize(r=cor(W/G, E/G))
```

```
##            r
## 1 -0.3396947
```

Question 9 **Q.** What is the correlation coefficient between doubles (X2B) per game and triples (X3B) per game? **A.**

```
Teams %>% filter(yearID %in% 1961:2001) %>%
  summarize(r=cor(X2B/G, X3B/G))
```

```
##             r
## 1 -0.01157404
```

1.3: Stratification and Variance Explained

Stratification

- The general idea of conditional expectation is that we stratify a population into groups and compute summaries in each group.
- A practical way to improve the estimates of the conditional expectations is to define strata of with similar values of x.
- If there is perfect correlation, the regression line predicts an increase that is the same number of SDs for both variables. If there is 0 correlation, then we don't use x at all for the prediction and simply predict the average μ_y. For values between 0 and 1, the prediction is somewhere in between. If the correlation is negative, we predict a reduction instead of an increase.

Intercept is zero and slope is ρ when the variables are standardized
Recall that, after standardization of a given variable, the mean of the variable will be equal to 0 and the standard deviation will be equal to 1. That is, after standardization, we have $\mu_x = 0$, $\mu_y = 0$, $\sigma_x = 1$, and $\sigma_y = 1$. Now, notice that the formula for the slope is given by:
$m = \rho\frac{\sigma_y}{\sigma_x}$
and the intercept is given by:
$b = \mu_y - m\mu_x$
Now, if we substitute the mean and the standard deviation of the standarized x and y variable, we arrive at slope:
$m = \rho \times \frac{1}{1}$
which simplifies to:
$m = \rho$
Now, if we substitute this slope into the formula for the receipt, we arrive at:
$b = 0 - \rho \times 0$
which simplifies to:
$b = 0 - 0$
or $b = 0$. Thus, we have shown that the interceipt is zero and slope is ρ once the variables are standarized.

```
# number of fathers with height 72 or 72.5 inches
sum(galton_heights$father == 72)
```

```
## [1] 8
```

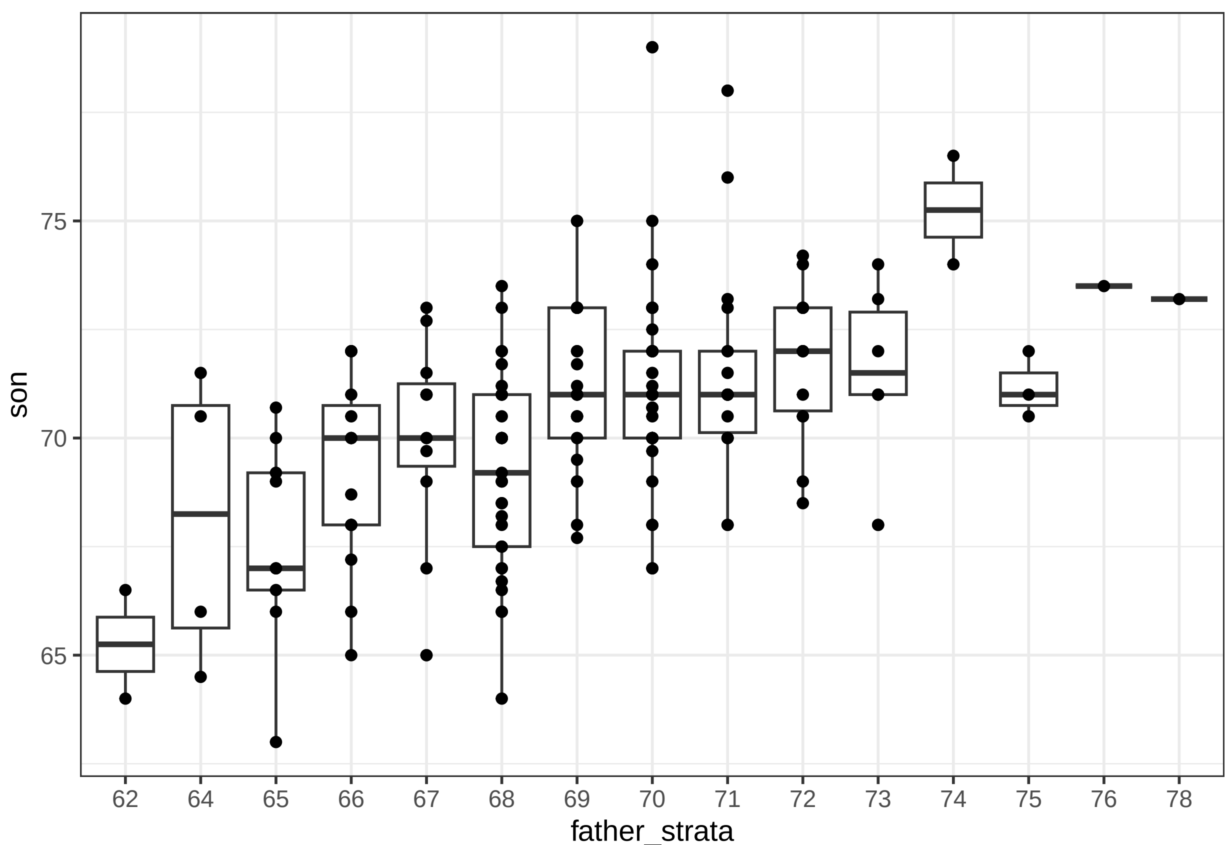
```r
sum(galton_heights$father == 72.5)
```

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

```r
# predicted height of a son with a 72 inch tall father
conditional_avg <- galton_heights %>%
    filter(round(father) == 72) %>%
    summarize(avg = mean(son)) %>%
    pull(avg)
conditional_avg
```

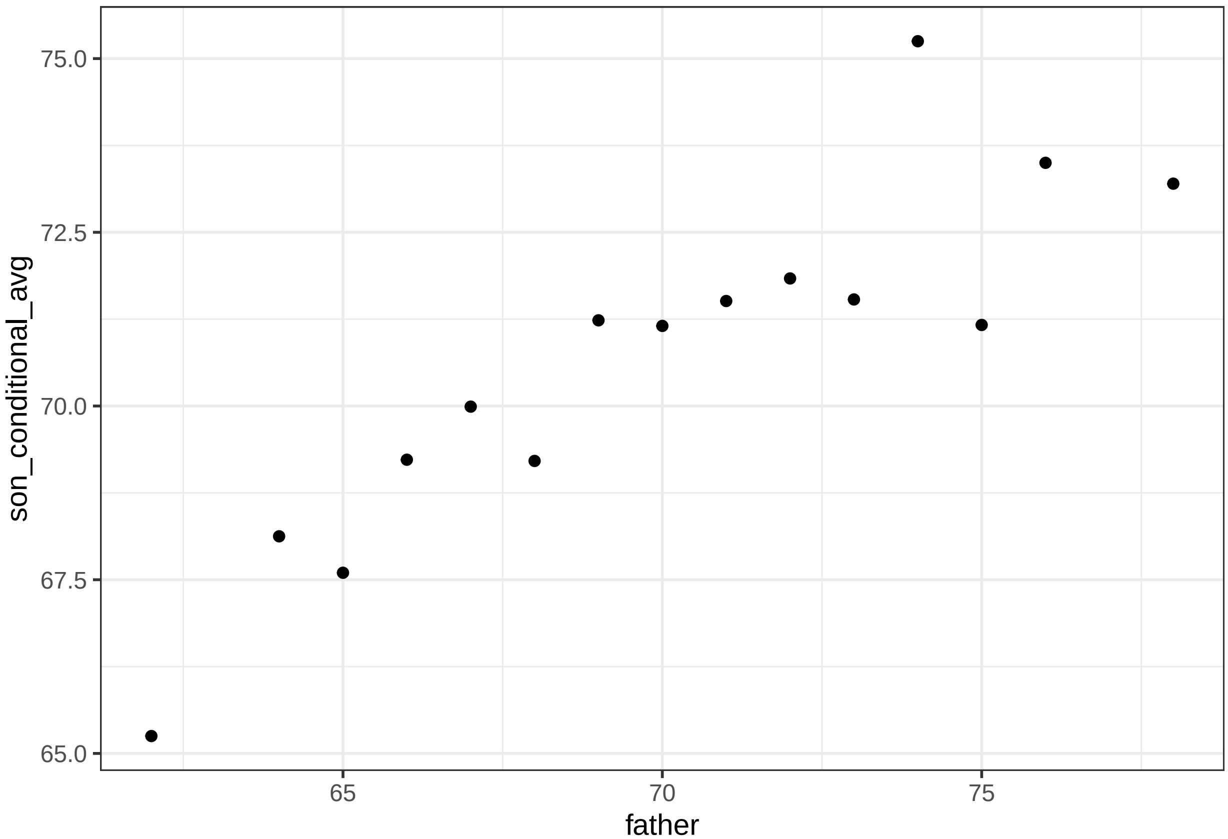
```
## [1] 71.83571
```

```r
# stratify fathers' heights to make a boxplot of son heights
galton_heights %>% mutate(father_strata = factor(round(father))) %>%
    ggplot(aes(father_strata, son)) +
    geom_boxplot() +
    geom_point()
```

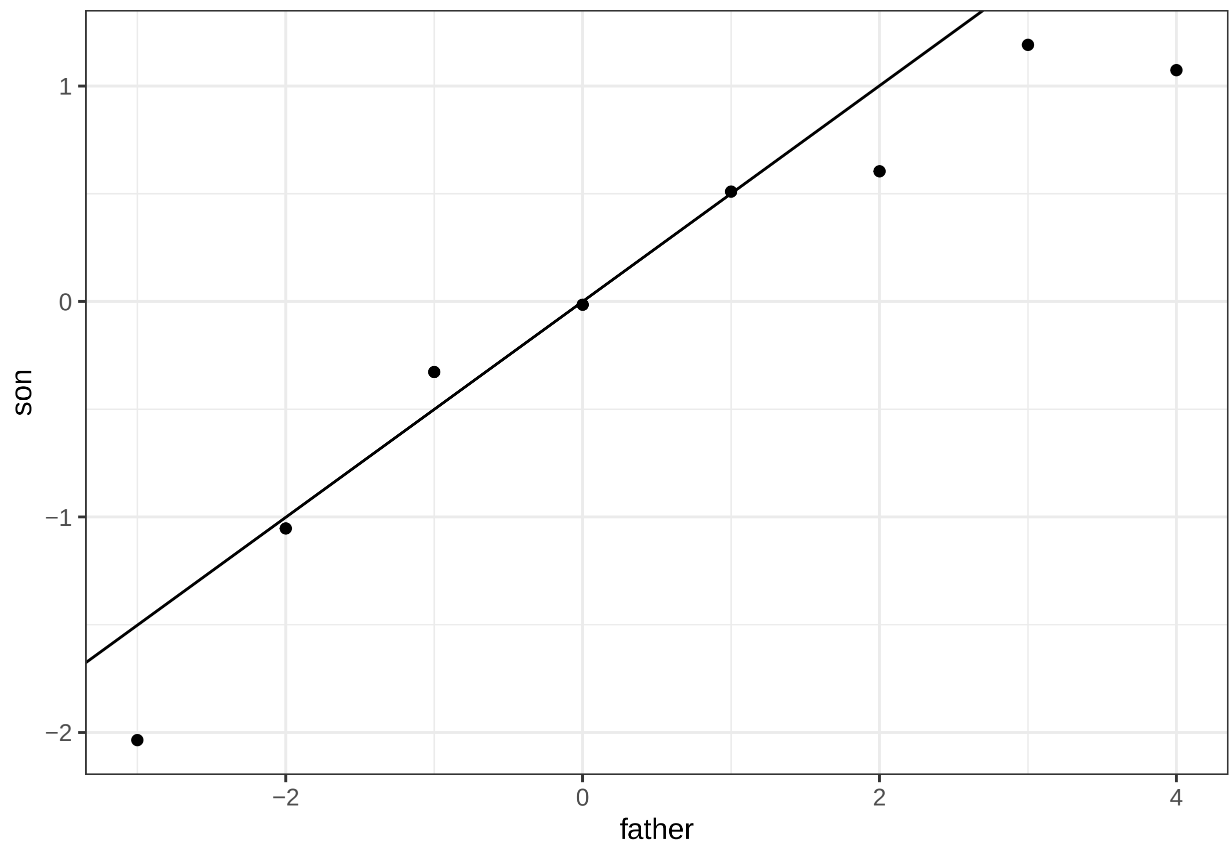



```r
# center of each boxplot
galton_heights %>%
    mutate(father = round(father)) %>%
    group_by(father) %>%
```

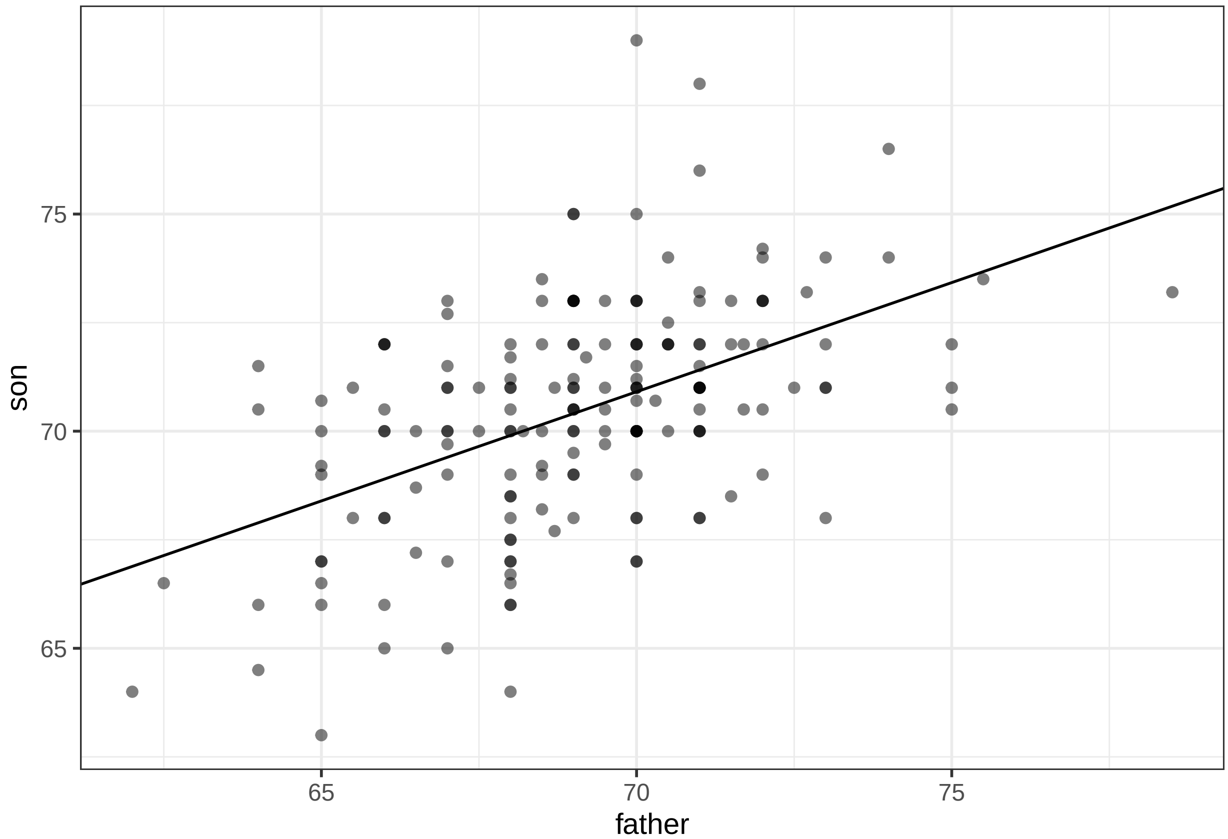
```r
  summarize(son_conditional_avg = mean(son)) %>%
  ggplot(aes(father, son_conditional_avg)) +
  geom_point()
```



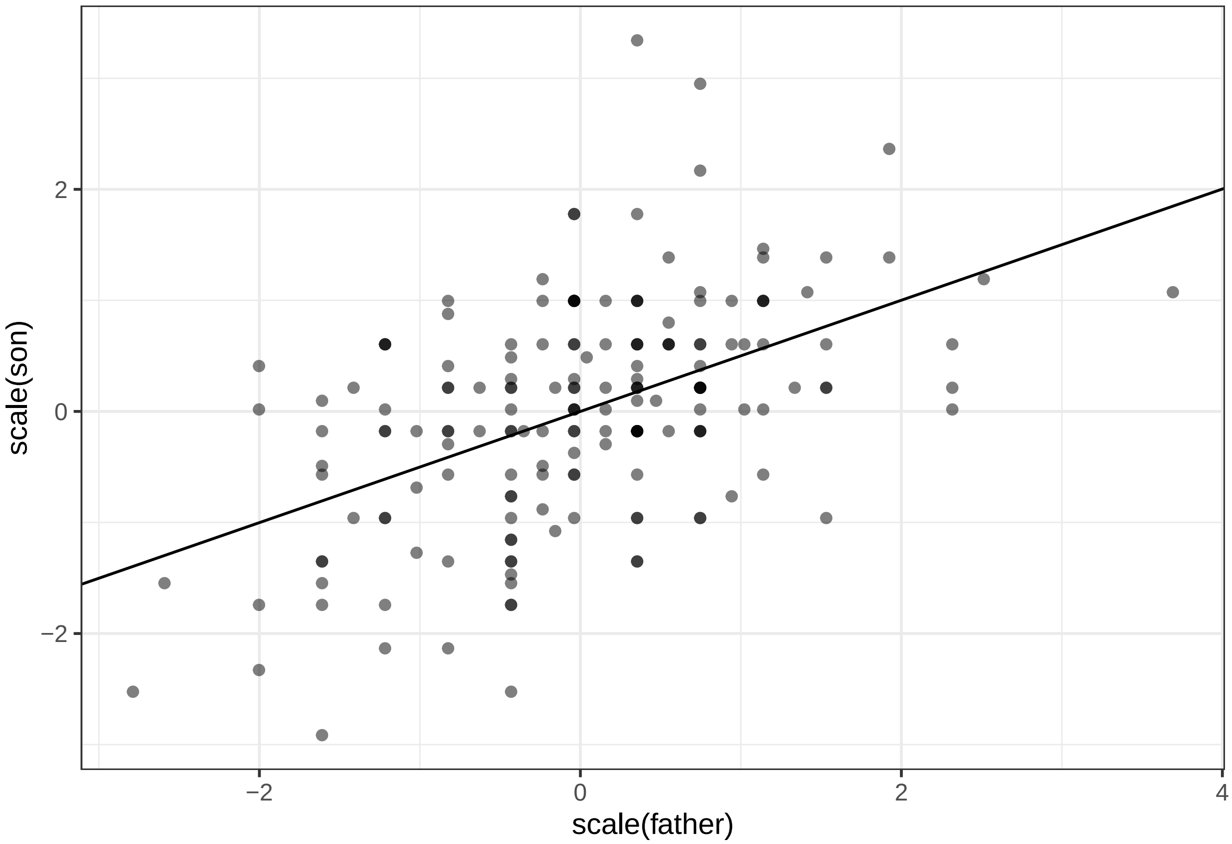

```r
# add regression line to standardized data
r <- galton_heights %>% summarize(r = cor(father, son)) %>% pull(r)

galton_heights %>%
  mutate(father = scale(father), son = scale(son)) %>%
  mutate(father = round(father)) %>%
  group_by(father) %>%
  summarize(son = mean(son)) %>%
  ggplot(aes(father, son)) +
  geom_point() +
  geom_abline(intercept = 0, slope = r)
```




```r
# add regression line to original data
mu_x <- mean(galton_heights$father)
mu_y <- mean(galton_heights$son)
s_x <- sd(galton_heights$father)
s_y <- sd(galton_heights$son)
r <- cor(galton_heights$father, galton_heights$son)
m <-  r * s_y / s_x
b <- mu_y - m*mu_x

galton_heights %>%
  ggplot(aes(father, son)) +
  geom_point(alpha = 0.5) +
  geom_abline(intercept = b, slope = m )
```



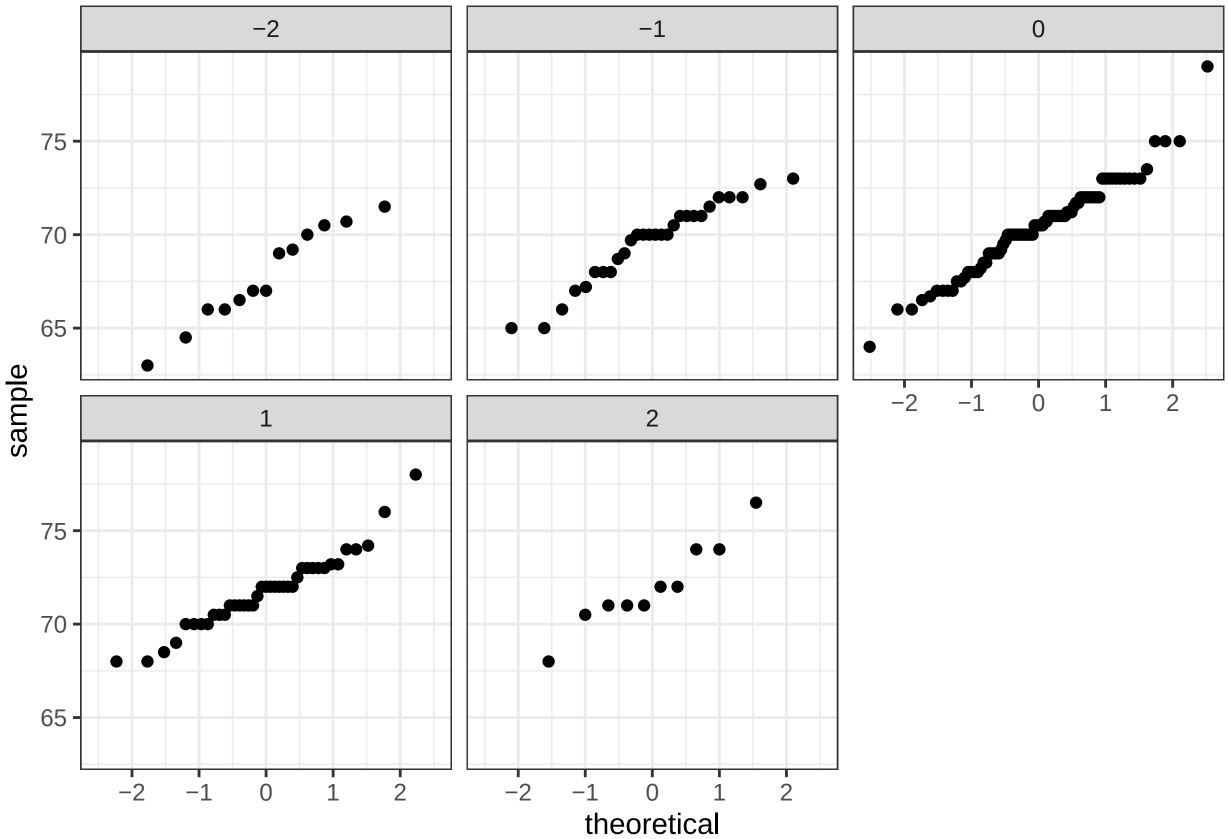

```r
# plot in standard units and see that intercept is 0 and slope is rho
galton_heights %>%
  ggplot(aes(scale(father), scale(son))) +
  geom_point(alpha = 0.5) +
  geom_abline(intercept = 0, slope = r)
```



Bivariate Normal Distribution

- When a pair of random variables are approximated by the bivariate normal distribution, scatterplots look like ovals. They can be thin (high correlation) or circle-shaped (no correlation).
- When two variables follow a bivariate normal distribution, computing the regression line is equivalent to computing conditional expectations.
- We can obtain a much more stable estimate of the conditional expectation by finding the regression line and using it to make predictions.
  #### Key equations
  Conditional distribution
  $f_{Y|X=x}$ is the conditional distribution and $E(Y|X=x)$ is the conditional expected value.
  Expected value (X is random variable and x is a fixed value that we pick)
  $E(Y|X=x) = \mu_Y + \rho \frac{X-\mu_X}{\sigma_X} \sigma_y$
  Same as the regression line
  $\frac{E(Y|X=x)-\mu_Y}{\sigma_Y} = \rho \frac{x-\mu_X}{\sigma_X}$

```
galton_heights %>%
  mutate(z_father = round((father - mean(father))/sd(father))) %>%
  filter(z_father %in% -2:2) %>%
  ggplot() +
  stat_qq(aes(sample=son)) +
  facet_wrap(~z_father)
```



Variance Explained

- Conditioning on a random variable X can help to reduce variance of response variable Y.
- The standard deviation of the conditional distribution is $SD(Y|X=x) = \sigma_y \sqrt{1-\rho^2}$, which is smaller tha the standard deviation without conditioning σ_y
- The variance is the square of the 'sd' so $\sigma_y^2 \left(1-\rho^2\right)$.
- In the statement "X explains such and such percent of the variability," the percent value refers to the variance. The variance decreases by ρ^2 percent.
- The "variance explained" statement only makes sense when the data is approximated by a bivariate normal distribution.

There are Two Regression Lines

```
# compute a regression line to predict the son's height from the father's height
mu_x <- mean(galton_heights$father)
mu_y <- mean(galton_heights$son)
s_x <- sd(galton_heights$father)
s_y <- sd(galton_heights$son)
r <- cor(galton_heights$father, galton_heights$son)
m <-  r * s_y / s_x
b <- mu_y - m*mu_x

# compute a regression line to predict the father's height from the son's height
```

```
m <-  r * s_x / s_y
b <- mu_x - m*mu_y
```

Assessment: Stratification and Variance Explained, Part 2

In the second part of this assessment, you'll analyze a set of mother and daughter heights, also from GaltonFamilies.

Define female_heights, a set of mother and daughter heights sampled from GaltonFamilies, as follows:

```
set.seed(1989) #if you are using R 3.5 or earlier
set.seed(1989, sample.kind="Rounding") #if you are using R 3.6 or later
```

```
## Warning in set.seed(1989, sample.kind = "Rounding"): non-uniform 'Rounding'
## sampler used
```

```
library(HistData)
data("GaltonFamilies")

female_heights <- GaltonFamilies%>%
    filter(gender == "female") %>%
    group_by(family) %>%
    sample_n(1) %>%
    ungroup() %>%
    select(mother, childHeight) %>%
    rename(daughter = childHeight)
```

Question 8 **Q.** Calculate the mean and standard deviation of mothers' heights, the mean and standard deviation of daughters' heights, and the correlaton coefficient between mother and daughter heights. **A.** Mean of mothers' heights:

```
mum <- mean(female_heights$mother)
```

Standard deviation of mothers' heights:

```
sm <- sd(female_heights$mother)
```

Mean of daughters' heights:

```
mud <- mean(female_heights$daughter)
```

Standard deviation of daughters' heights:

```
sda <- sd(female_heights$daughter)
```

Correlation coefficient:

```r
rho <- cor(female_heights$mother, female_heights$daughter)
```

Question 9 **Q.** Calculate the slope and intercept of the regression line predicting daughters' heights given mothers' heights. Given an increase in mother's height by 1 inch, how many inches is the daughter's height expected to change?

Slope of regression line predicting daughters' height from mothers' heights

```r
m <- rho * sda / sm
```

Intercept of regression line predicting daughters' height from mothers' heights

```r
b <- mud - m*mum
```

Change in daughter's height in inches given a 1 inch increase in the mother's height

```r
rho * sda/sm
```

```
## [1] 0.3393856
```

Question 10 **Q.** What percent of the variability in daughter heights is explained by the mother's height? **A.**

```r
rho^2 *100 #We multiply 100 for percentage
```

```
## [1] 10.53132
```

Question 11 **Q.** A mother has a height of 60 inches. Using the regression formula, what is the conditional expected value of her daughter's height given the mother's height? **A.**

```r
Xx <- 60
mean(mud + rho * ((Xx-mum)/(sm)) * sda)
```

```
## [1] 62.88015
```

Section 2: Linear Models

```r
library(tidyverse)
library(Lahman)
library(HistData)
library(gridExtra)
```

```
##
## Attaching package: 'gridExtra'

## The following object is masked from 'package:dplyr':
##
##     combine
```

2.1: Introduction to Linear Models

Confounding: Are BBs More Predictive?

Remember: Association is not causation Regression can help us account for confounding.

```
# find regression line for predicting runs from BBs (not shown in video)
get_slope <- function(x, y) cor(x, y) * sd(y) / sd(x)

bb_slope <- Teams %>%
  filter(yearID %in% 1961:2001 ) %>%
  mutate(BB_per_game = BB/G, R_per_game = R/G) %>%
  summarize(slope = get_slope(BB_per_game, R_per_game))

bb_slope
```

```
##       slope
## 1 0.7353288
```

```
# compute regression line for predicting runs from singles (not shown in video)
singles_slope <- Teams %>%
  filter(yearID %in% 1961:2001 ) %>%
  mutate(Singles_per_game = (H-HR-X2B-X3B)/G, R_per_game = R/G) %>%
  summarize(slope = get_slope(Singles_per_game, R_per_game))

singles_slope
```

```
##       slope
## 1 0.4494253
```

```
# calculate correlation between HR, BB, and singles
Teams %>%
  filter(yearID %in% 1961:2001 ) %>%
  mutate(Singles = (H-HR-X2B-X3B)/G, BB = BB/G, HR = HR/G) %>%
  summarize(cor(BB, HR), cor(Singles, HR), cor(BB,Singles))
```

```
##   cor(BB, HR) cor(Singles, HR) cor(BB, Singles)
## 1   0.4039313       -0.1737435      -0.05603822
```

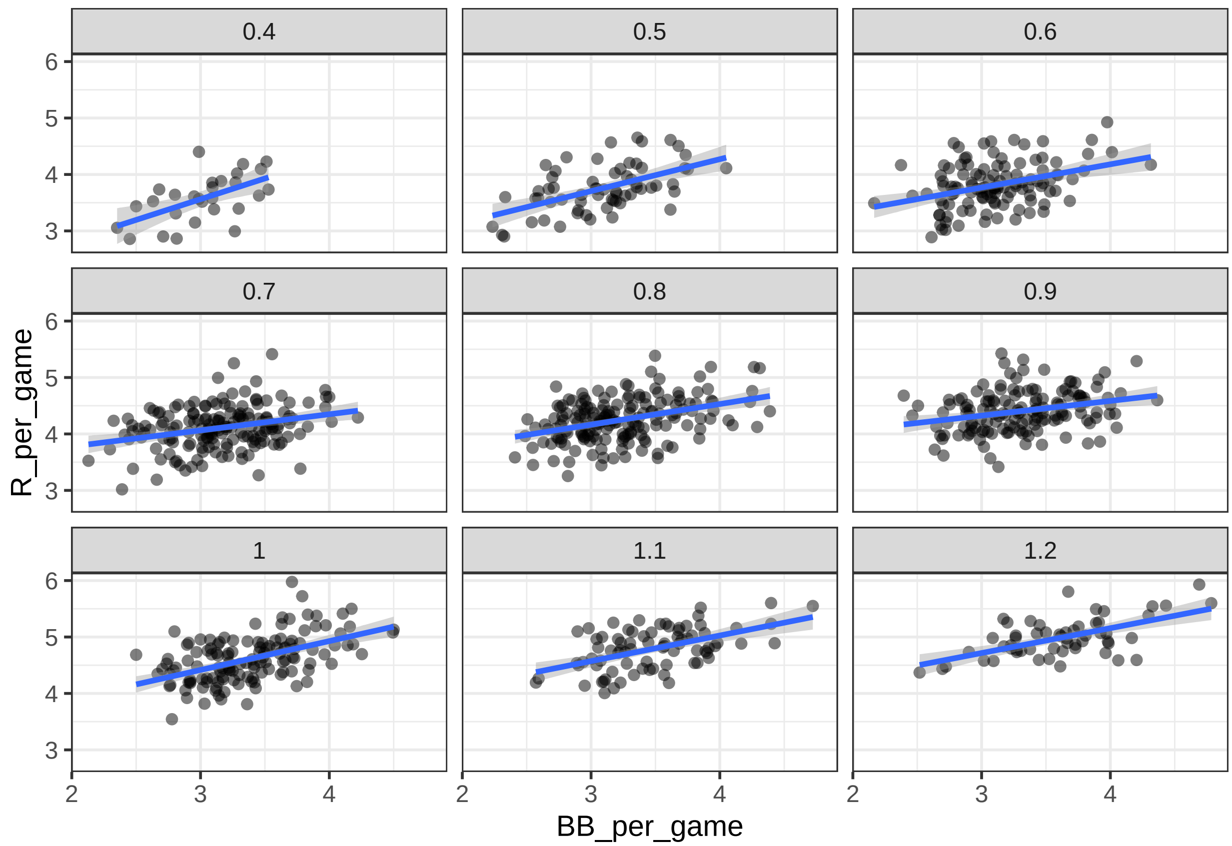
Stratification and Multivariate Regression

- A first approach to check confounding is to keep HRs fixed at a certain value and then examine the relationship between BB and runs.
- The slopes of BB after stratifying on HR are reduced, but they are not 0, which indicates that BB are helpful for producing runs, just not as much as previously thought.

```
# stratify HR per game to nearest 10, filter out strata with few points
dat <- Teams %>% filter(yearID %in% 1961:2001) %>%
  mutate(HR_strata = round(HR/G, 1),
         BB_per_game = BB / G,
         R_per_game = R / G) %>%
```

```
  filter(HR_strata >= 0.4 & HR_strata <=1.2)

# scatterplot for each HR stratum
dat %>%
  ggplot(aes(BB_per_game, R_per_game)) +
  geom_point(alpha = 0.5) +
  geom_smooth(method = "lm") +
  facet_wrap( ~ HR_strata)
```

```
## `geom_smooth()` using formula = 'y ~ x'
```



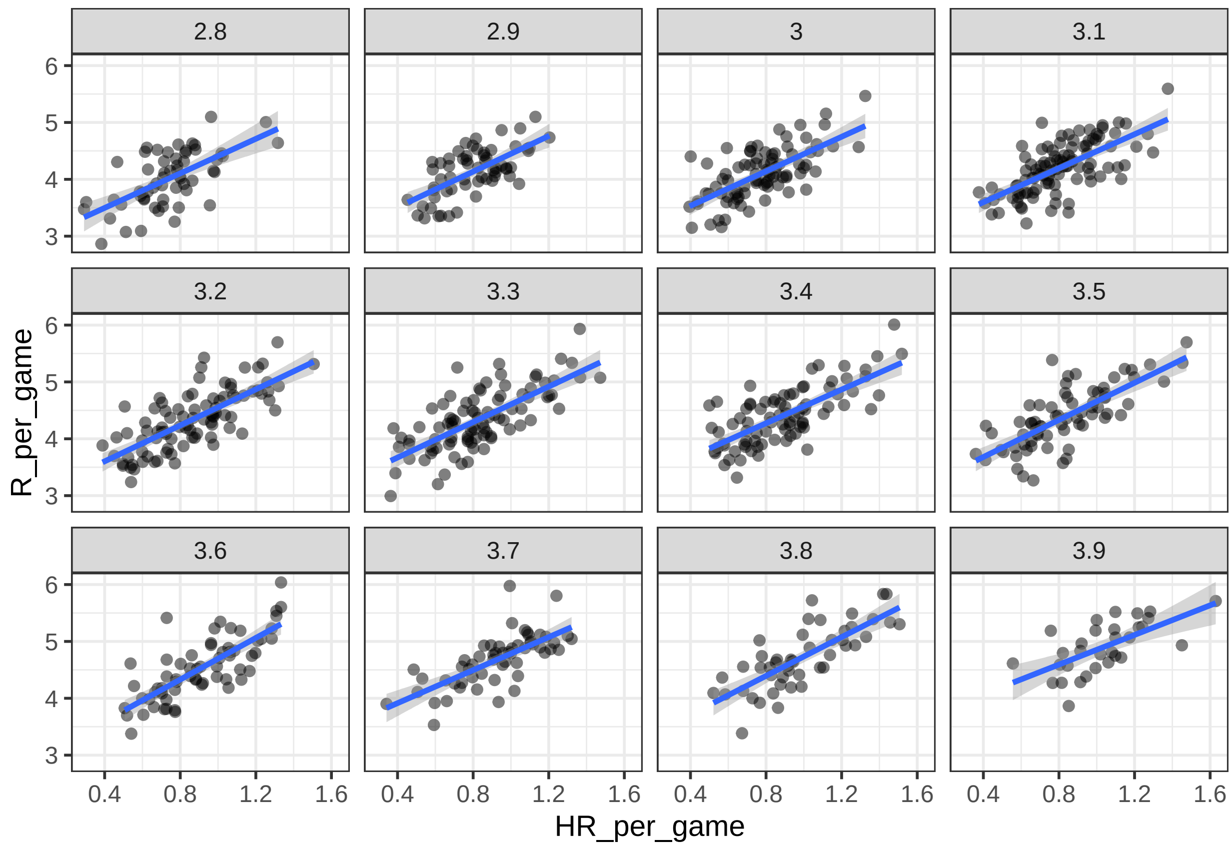

```
# calculate slope of regression line after stratifying by HR
dat %>%
  group_by(HR_strata) %>%
  summarize(slope = cor(BB_per_game, R_per_game)*sd(R_per_game)/sd(BB_per_game))
```

```
## # A tibble: 9 x 2
##   HR_strata slope
##       <dbl> <dbl>
## 1       0.4 0.734
## 2       0.5 0.566
## 3       0.6 0.412
## 4       0.7 0.285
## 5       0.8 0.365
```

```
## 6        0.9 0.261
## 7        1   0.512
## 8        1.1 0.454
## 9        1.2 0.440
```

```r
# stratify by BB
dat <- Teams %>% filter(yearID %in% 1961:2001) %>%
  mutate(BB_strata = round(BB/G, 1),
         HR_per_game = HR / G,
         R_per_game = R / G) %>%
  filter(BB_strata >= 2.8 & BB_strata <=3.9)

# scatterplot for each BB stratum
dat %>% ggplot(aes(HR_per_game, R_per_game)) +
  geom_point(alpha = 0.5) +
  geom_smooth(method = "lm") +
  facet_wrap( ~ BB_strata)
```

```
## `geom_smooth()` using formula = 'y ~ x'
```




```r
# slope of regression line after stratifying by BB
dat %>%
  group_by(BB_strata) %>%
  summarize(slope = cor(HR_per_game, R_per_game)*sd(R_per_game)/sd(HR_per_game))
```

```
## # A tibble: 12 x 2
##    BB_strata slope
##        <dbl> <dbl>
##  1       2.8  1.52
##  2       2.9  1.57
##  3       3    1.52
##  4       3.1  1.49
##  5       3.2  1.58
##  6       3.3  1.56
##  7       3.4  1.48
##  8       3.5  1.63
##  9       3.6  1.83
## 10       3.7  1.45
## 11       3.8  1.70
## 12       3.9  1.30
```

Linear Models

- "Linear" here does not refer to lines, but rather to the fact that the conditional expectation is a linear combination of known quantities.
- In Galton's model, we assume Y (son's height) is a linear combination of a constant and X (father's height) plus random noise. We further assume that ϵ_i are independent from each other, have expected value 0 and the standard deviation σ which does not depend on i.
- Note that if we further assume that ϵ is normally distributed, then the model is exactly the same one we derived earlier by assuming bivariate normal data.
- We can substract the mean from X to make β_0 more interpretable.

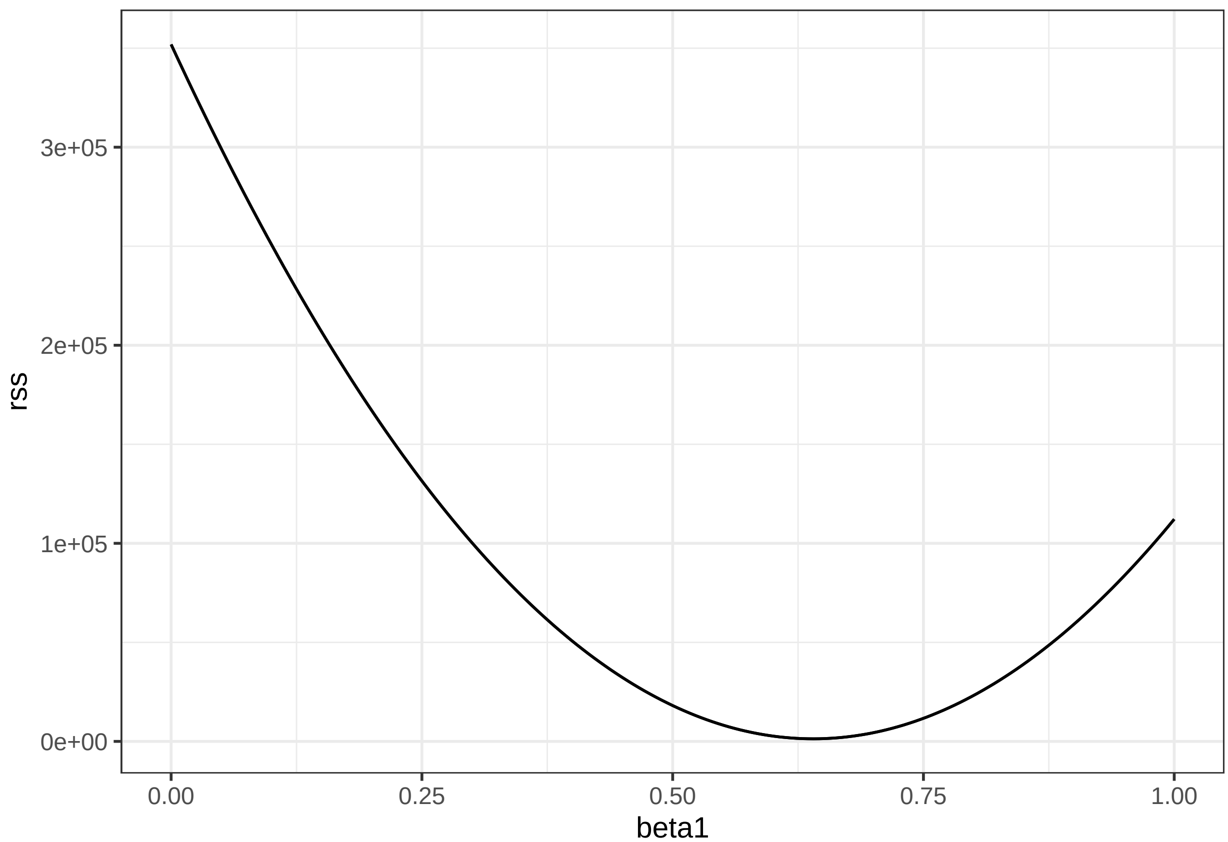
2.2: Least Squares Estimates

Least Squares Estimates (LSE)

- For regression, we aim to find the coefficient values that minimize the distance of the fitted model to the data.
- Residual sum of squares (RSS) measures the distance between the true value and the predicted value given by the regression line. The values that minimize the RSS are called the least squares estimates (LSE).
- We can use partial derivatives to get the values for β_0 and β_1 in Galton's data.

```
# compute RSS for any pair of beta0 and beta1 in Galton's data
library(HistData)
data("GaltonFamilies")
set.seed(1983)
galton_heights <- GaltonFamilies %>%
  filter(gender == "male") %>%
  group_by(family) %>%
  sample_n(1) %>%
  ungroup() %>%
  select(father, childHeight) %>%
  rename(son = childHeight)
rss <- function(beta0, beta1){
    resid <- galton_heights$son - (beta0+beta1*galton_heights$father)
    return(sum(resid^2))
```

```
}

# plot RSS as a function of beta1 when beta0=25
beta1 = seq(0, 1, len=nrow(galton_heights))
results <- data.frame(beta1 = beta1,
                      rss = sapply(beta1, rss, beta0 = 25))

results %>% ggplot(aes(beta1, rss)) + geom_line() +
  geom_line(aes(beta1, rss))
```



The lm Function

- When calling the lm() function, the variable that we want to predict is put to the left of the ~ symbol, and the variables that we use to predict is put to the right of the ~ symbol. The intercept is added automatically.
- LSEs are random variables as they are derived from samples.

```
# fit regression line to predict son's height from father's height
fit <- lm(son ~ father, data = galton_heights)
fit
```

```
##
## Call:
## lm(formula = son ~ father, data = galton_heights)
```

```
## 
## Coefficients:
## (Intercept)       father  
##     38.7646       0.4411
```

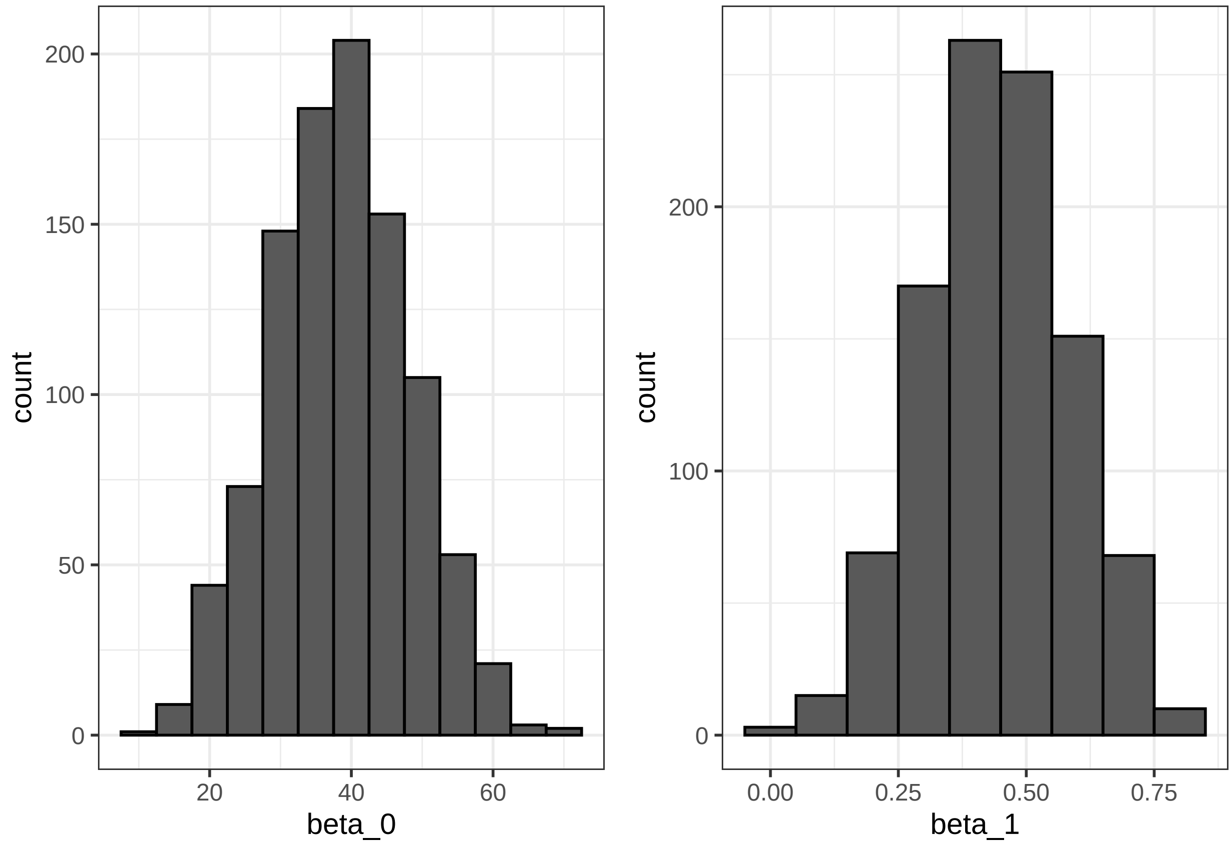
```r
# summary statistics
summary(fit)
```

```
## 
## Call:
## lm(formula = son ~ father, data = galton_heights)
## 
## Residuals:
##     Min      1Q  Median      3Q     Max 
## -9.4228 -1.7022  0.0333  1.5670  9.3567 
## 
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)    
## (Intercept) 38.76457    5.41093   7.164 2.03e-11 ***
## father       0.44112    0.07825   5.637 6.72e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## 
## Residual standard error: 2.659 on 177 degrees of freedom
## Multiple R-squared:  0.1522, Adjusted R-squared:  0.1474 
## F-statistic: 31.78 on 1 and 177 DF,  p-value: 6.719e-08
```

LSE are Random Variables

- β_0 and β_1 appear to be normally distributed because the central limit theorem plays a role.
- The t-statistic depends on the assumption that ϵ follows a normal distribution.

```r
# Monte Carlo simulation
B <- 1000
N <- 50
lse <- replicate(B, {
  sample_n(galton_heights, N, replace = TRUE) %>%
    lm(son ~ father, data = .) %>%
    .$coef
})
lse <- data.frame(beta_0 = lse[1,], beta_1 = lse[2,])

# Plot the distribution of beta_0 and beta_1
library(gridExtra)
p1 <- lse %>% ggplot(aes(beta_0)) + geom_histogram(binwidth = 5, color = "black")
p2 <- lse %>% ggplot(aes(beta_1)) + geom_histogram(binwidth = 0.1, color = "black")
grid.arrange(p1, p2, ncol = 2)
```




```
# summary statistics
sample_n(galton_heights, N, replace = TRUE) %>%
  lm(son ~ father, data = .) %>%
  summary %>%
  .$coef
```

```
##               Estimate Std. Error  t value     Pr(>|t|)
## (Intercept) 34.4729422  8.6021831 4.007464 0.0002129225
## father       0.4990193  0.1240572 4.022493 0.0002030210
```

```
lse %>% summarize(se_0 = sd(beta_0), se_1 = sd(beta_1))
```

```
##       se_0      se_1
## 1 9.683973 0.1411404
```

Advanced Note on LSE

Although interpretation is not straight-forward, it is also useful to know that the LSE can be strongly correlated, which can be seen using this code:

```
lse %>% summarize(cor(beta_0, beta_1))
```

```
##   cor(beta_0, beta_1)
## 1          -0.9993386
```

However, the correlation depends on how the predictors are defined or transformed. Here we standardize the father heights, which changes x_i to $x_i - \bar{x}$.

```
B <- 1000
N <- 50
lse <- replicate(B, {
      sample_n(galton_heights, N, replace = TRUE) %>%
      mutate(father = father - mean(father)) %>%
      lm(son ~ father, data = .) %>% .$coef
})
```

Observe what happens to the correlation in this case:

```
cor(lse[1,], lse[2,])
```

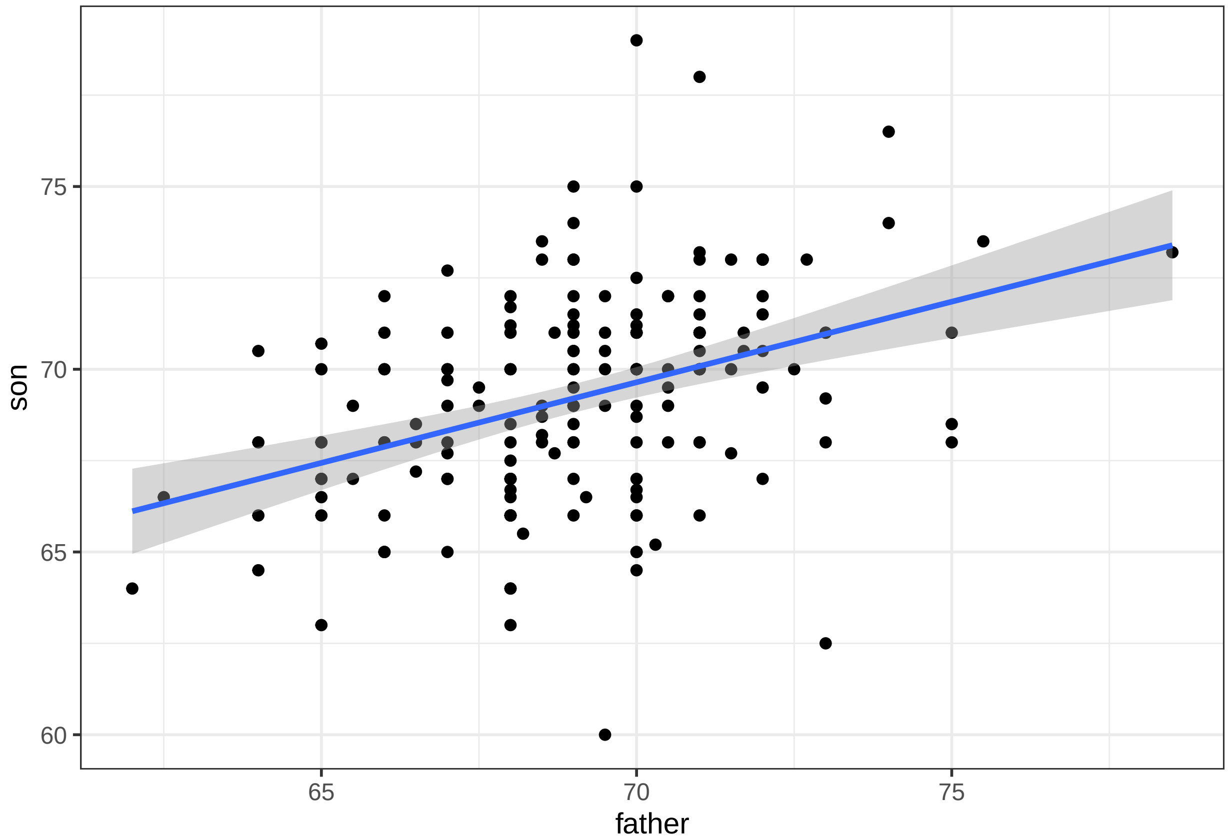
```
## [1] 0.1100929
```

Predicted Variables are Random Variables

- The predicted value is often denoted as $\hat{Y}$, which is a random variable. Mathematical theory tells us what the standard error of the predicted value is.
- The predict() function in R can give us predictions directly.

```
# plot predictions and confidence intervals
galton_heights %>% ggplot(aes(father, son)) +
  geom_point() +
  geom_smooth(method = "lm")
```

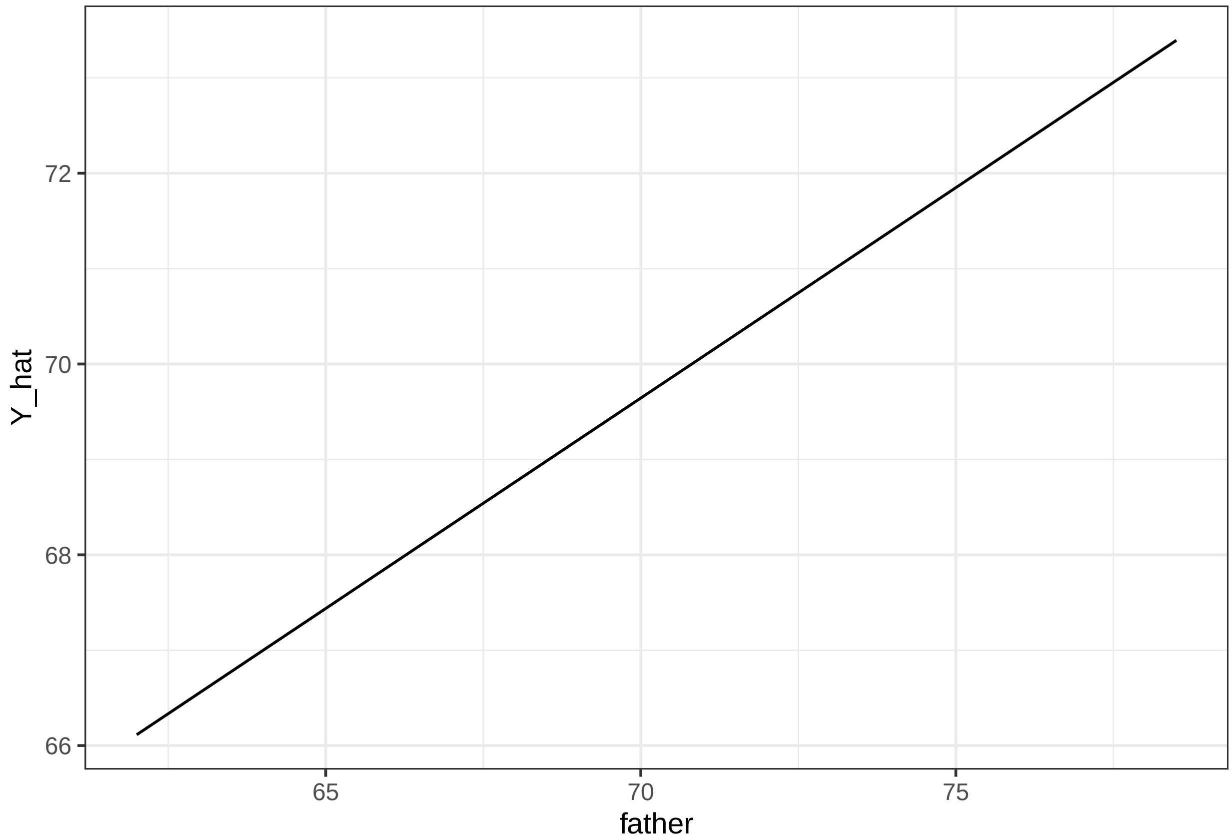
```
## `geom_smooth()` using formula = 'y ~ x'
```




```r
# predict Y directly
fit <- galton_heights %>% lm(son ~ father, data = .)
Y_hat <- predict(fit, se.fit = TRUE)
names(Y_hat)
```

```
## [1] "fit"            "se.fit"         "df"             "residual.scale"
```

```r
# plot best fit line
galton_heights %>%
  mutate(Y_hat = predict(lm(son ~ father, data=.))) %>%
  ggplot(aes(father, Y_hat))+
  geom_line()
```



Assessment: Least Squares Estimates, part 1

Question 3 **Q.** Load the Lahman library and filter the Teams data frame to the years 1961-2001. Mutate the dataset to create variables for bases on balls per game, runs per game, and home runs per game, then run a linear model in R predicting the number of runs per game based on both the number of bases on balls per game and the number of home runs per game. What is the coefficient for bases on balls per game? **A.**

```
library(Lahman)
fit <- Teams %>% filter(yearID %in% 1961:2001) %>%
  mutate(BBG = BB/G, RG = R/G, HRG = HR/G)
lm(RG ~ BBG + HRG, data = fit)
```

```
##
## Call:
## lm(formula = RG ~ BBG + HRG, data = fit)
##
## Coefficients:
## (Intercept)          BBG          HRG
##      1.7443       0.3874       1.5612
```

```
## Or

library(Lahman)
library(broom)
```

```
Teams_small <- Teams %>% filter(yearID %in% 1961:2001)
Teams_small %>% mutate(R_per_game = R/G, BB_per_game = BB/G, HR_per_game = HR/G) %>% do(tidy(lm(R_per_ga
```

```
## # A tibble: 3 x 5
##   term        estimate std.error statistic   p.value
##   <chr>          <dbl>     <dbl>     <dbl>     <dbl>
## 1 (Intercept)    1.74     0.0824      21.2 7.62e- 83
## 2 BB_per_game    0.387    0.0270      14.3 1.20e- 42
## 3 HR_per_game    1.56     0.0490      31.9 1.78e-155
```

Assessment: Least Squares Estimates, part 2

```
set.seed(1989) #if you are using R 3.5 or earlier
set.seed(1989, sample.kind="Rounding") #if you are using R 3.6 or later
```

```
## Warning in set.seed(1989, sample.kind = "Rounding"): non-uniform 'Rounding'
## sampler used
```

```
library(HistData)
data("GaltonFamilies")
options(digits = 3)    # report 3 significant digits

female_heights <- GaltonFamilies %>%
    filter(gender == "female") %>%
    group_by(family) %>%
    sample_n(1) %>%
    ungroup() %>%
    select(mother, childHeight) %>%
    rename(daughter = childHeight)
```

Question 7 **Q.** Fit a linear regression model predicting the mothers' heights using daughters' heights. What is the slope of the model? **A.**

```
lm(mother ~ daughter, data = female_heights)
```

```
##
## Call:
## lm(formula = mother ~ daughter, data = female_heights)
##
## Coefficients:
## (Intercept)     daughter
##       44.18         0.31
```

Question 8 **Q.** Predict mothers' heights using the model from Question 7 and the predict() function. What is the predicted height of the first mother in the dataset? **A.**

```r
model <- lm(mother ~ daughter, data = female_heights)

predictions <- predict(model, interval = c("confidence"), level = 0.95)

data <- as_tibble(predictions) %>% bind_cols(daughter = female_heights$daughter)

head(data)
```

```
## # A tibble: 6 x 4
##     fit   lwr   upr daughter
##   <dbl> <dbl> <dbl>    <dbl>
## 1  65.6  64.9  66.3     69
## 2  64.5  64.1  64.9     65.5
## 3  65.3  64.7  65.9     68
## 4  64.2  63.9  64.5     64.5
## 5  64.8  64.4  65.3     66.5
## 6  65.7  65.0  66.5     69.5
```

```r
head(female_heights)
```

```
## # A tibble: 6 x 2
##   mother daughter
##    <dbl>    <dbl>
## 1   67        69
## 2   66.5      65.5
## 3   64        68
## 4   64        64.5
## 5   58.5      66.5
## 6   68        69.5
```

```r
## Or
predict(model)
```

```
##    1    2    3    4    5    6    7    8    9   10   11   12   13   14   15   16
## 65.6 64.5 65.3 64.2 64.8 65.7 66.1 66.1 64.7 64.5 65.0 64.3 63.4 64.9 64.2 64.8
##   17   18   19   20   21   22   23   24   25   26   27   28   29   30   31   32
## 64.2 63.6 65.6 65.3 65.0 64.7 64.5 64.7 64.3 63.7 63.7 64.9 64.0 64.3 63.9 65.3
##   33   34   35   36   37   38   39   40   41   42   43   44   45   46   47   48
## 64.6 65.0 65.0 64.8 63.7 65.0 64.7 64.3 64.4 63.4 64.3 65.0 64.7 63.4 65.1 64.3
##   49   50   51   52   53   54   55   56   57   58   59   60   61   62   63   64
## 63.7 63.9 63.3 62.8 64.0 64.2 63.4 65.6 65.3 64.7 65.1 64.7 65.3 64.2 63.6 64.3
##   65   66   67   68   69   70   71   72   73   74   75   76   77   78   79   80
## 65.0 64.3 64.2 65.0 64.5 63.2 64.3 62.8 64.3 65.6 63.9 63.9 64.2 65.1 63.9 63.4
##   81   82   83   84   85   86   87   88   89   90   91   92   93   94   95   96
## 63.7 63.9 62.8 63.7 64.3 63.7 64.7 64.5 64.0 64.3 63.6 63.1 64.0 63.7 64.2 64.8
##   97   98   99  100  101  102  103  104  105  106  107  108  109  110  111  112
## 64.3 62.8 64.3 64.5 63.9 64.7 63.7 63.9 63.6 63.7 63.4 63.6 64.7 63.7 63.9 65.3
##  113  114  115  116  117  118  119  120  121  122  123  124  125  126  127  128
## 63.3 63.3 65.0 63.8 64.0 64.4 65.3 62.8 64.0 63.9 63.4 63.6 63.4 64.3 64.7 64.5
##  129  130  131  132  133  134  135  136  137  138  139  140  141  142  143  144
## 64.2 63.4 63.9 64.2 63.3 63.7 62.8 62.8 64.3 63.9 64.5 64.3 64.5 63.7 63.4 63.7
##  145  146  147  148  149  150  151  152  153  154  155  156  157  158  159  160
```

```
## 63.9 63.7 64.7 63.4 62.8 64.5 64.7 62.0 64.7 64.2 64.3 63.6 62.8 63.1 64.3 63.7
##  161  162  163  164  165  166  167  168  169  170  171  172  173  174  175  176
## 63.7 64.3 63.1 64.5 64.2 64.0 63.7 63.7 63.9 63.4 63.1 62.8 62.8 63.9 63.4 61.9
```

Question 9 Filter players who appeared more than 100 times in the 2002 season.

```
library(Lahman)
bat_02 <- Batting %>% filter(yearID == 2002) %>%
    mutate(pa = AB + BB, singles = (H - X2B - X3B - HR)/pa, bb = BB/pa) %>%
    filter(pa >= 100) %>%
    select(playerID, singles, bb)
```

Q. Now compute a similar table but with rates computed over 1999-2001. Keep only rows from 1999-2001 where players have 100 or more plate appearances, calculate each player's single rate and BB rate per stint (where each row is one stint - a player can have multiple stints within a season), then calculate the average single rate (mean_singles) and average BB rate (mean_bb) per player over the three year period. How many players had a single rate mean_singles of greater than 0.2 per plate appearance over 1999-2001? **A.**

```
bat_99_01 <- Batting %>% filter(yearID %in% 1999:2001) %>%
  mutate(pa = AB + BB, singles = (H - X2B - X3B - HR)/pa, bb = BB/pa) %>%
  filter(pa >= 100) %>%
  select(playerID, singles, bb)

mean_bat_99_01 <- Batting %>% filter(yearID %in% 1999:2001) %>%
  mutate(pa = AB + BB, singles = (H - X2B - X3B - HR)/pa, bb = BB/pa) %>%
  filter(pa >= 100) %>%
  group_by(playerID) %>%
  summarize(mean_singles = mean(singles), mean_bb = mean(bb))

sum(mean_bat_99_01$mean_singles > 0.2)
```

```
## [1] 46
```

```
sum(mean_bat_99_01$mean_bb > 0.2)
```

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

Question 10 **Q.** Use inner_join() to combine the bat_02 table with the table of 1999-2001 rate averages you created in the previous question. What is the correlation between 2002 singles rates and 1999-2001 average singles rates? **A.**

```
bat <- inner_join(bat_02, mean_bat_99_01, join_by(playerID))
cor(bat$singles, bat$mean_singles)
```

```
## [1] 0.551
```

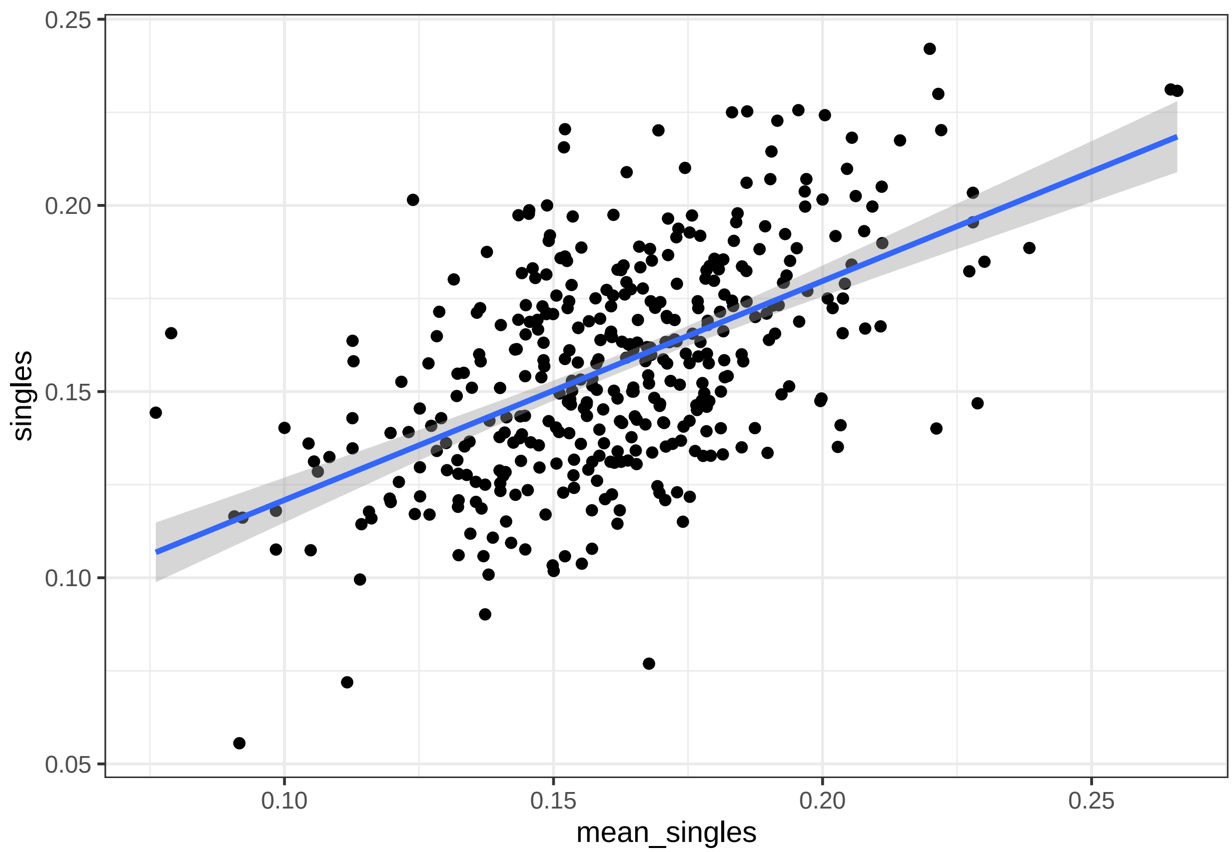
```
cor(bat$bb, bat$mean_bb)
```

```
## [1] 0.717
```

Question 11 **Q.** Make scatterplots of mean_singles versus singles and mean_bb versus bb. Are either of these distributions bivariate normal? **A.**

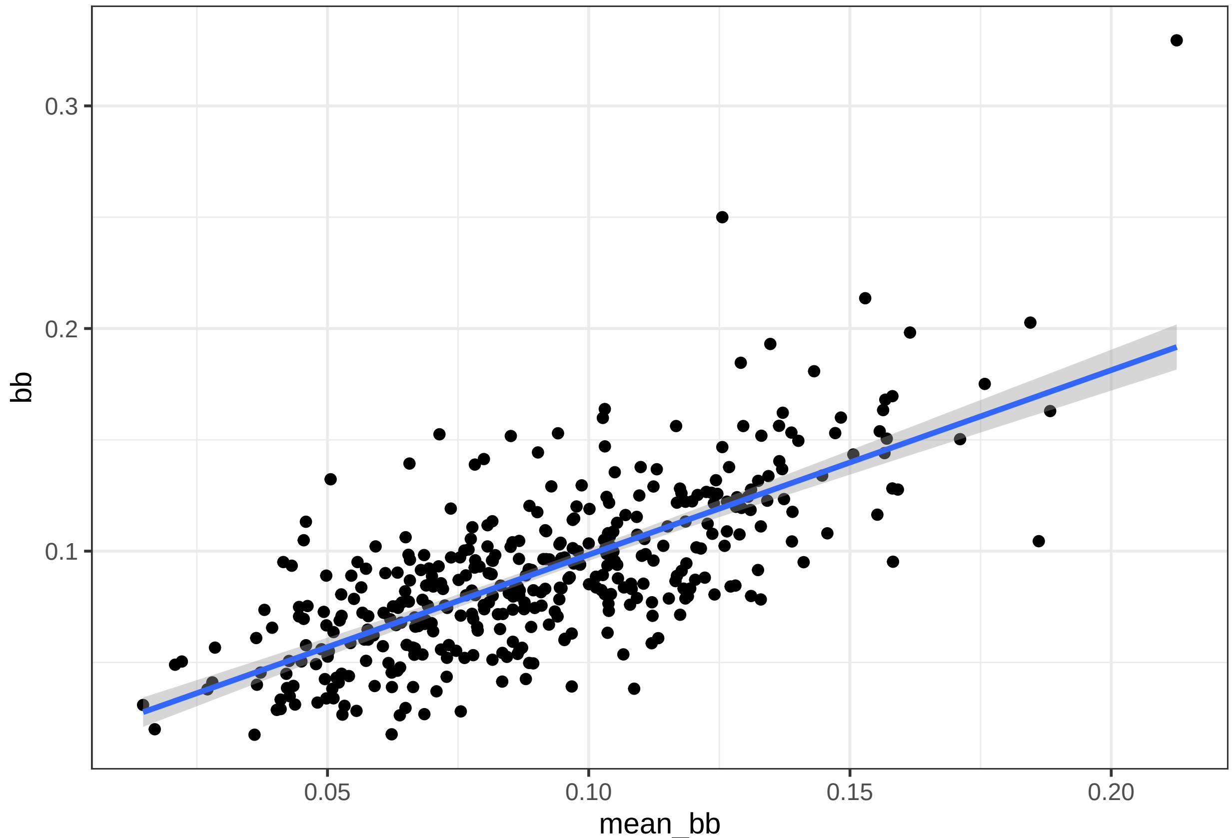
```
bat %>% ggplot(aes(x = mean_singles, y = singles)) +
  geom_point() +
  geom_smooth(method = 'lm')
```

```
## `geom_smooth()` using formula = 'y ~ x'
```




```
bat %>% ggplot(aes(x = mean_bb, y = bb)) +
  geom_point() +
  geom_smooth(method = 'lm')
```

```
## `geom_smooth()` using formula = 'y ~ x'
```



Question 12 **Q.** Fit a linear model to predict 2002 singles given 1999-2001 mean_singles. What is the coefficient of mean_singles, the slope of the fit? **A.**

```
lm(singles ~ mean_singles, data = bat)
```

```
##
## Call:
## lm(formula = singles ~ mean_singles, data = bat)
##
## Coefficients:
##  (Intercept)  mean_singles
##       0.0621        0.5881
```

Q. Fit a linear model to predict 2002 bb given 1999-2001 mean_bb. What is the coefficient of mean_bb, the slope of the fit? **A.**

```
lm(bb ~ mean_bb, data = bat)
```

```
##
## Call:
## lm(formula = bb ~ mean_bb, data = bat)
##
## Coefficients:
## (Intercept)      mean_bb
##      0.0155       0.8290
```

2.3: Advanced dplyr: summarize with functions and broom

Advanced dplyr: summarize with functions and broom

```
# stratify by HR
dat <- Teams %>% filter(yearID %in% 1961:2001) %>%
  mutate(HR = round(HR/G, 1),
         BB = BB/G,
         R = R/G) %>%
  select(HR, BB, R) %>%
  filter(HR >= 0.4 & HR<=1.2)

# calculate slope of regression lines to predict runs by BB in different HR strata
dat %>%
  group_by(HR) %>%
  summarize(slope = cor(BB,R)*sd(R)/sd(BB))
```

```
## # A tibble: 9 x 2
##      HR slope
##   <dbl> <dbl>
## 1   0.4 0.734
## 2   0.5 0.566
## 3   0.6 0.412
## 4   0.7 0.285
## 5   0.8 0.365
## 6   0.9 0.261
## 7   1   0.512
## 8   1.1 0.454
## 9   1.2 0.440
```

```
# use lm to get estimated slopes - lm does not work with grouped tibbles
dat %>%
  group_by(HR) %>%
  lm(R ~ BB, data = .) %>%
  .$coef
```

```
## (Intercept)          BB
##       2.198       0.638
```

```
# include the lm inside a summarize and it will work
dat %>%
  group_by(HR) %>%
  summarize(slope = lm(R ~ BB)$coef[2])
```

```
## # A tibble: 9 x 2
##      HR slope
##   <dbl> <dbl>
## 1   0.4 0.734
## 2   0.5 0.566
## 3   0.6 0.412
## 4   0.7 0.285
```

```
## 5   0.8 0.365
## 6   0.9 0.261
## 7   1   0.512
## 8   1.1 0.454
## 9   1.2 0.440
```

```
# tidy function from broom returns estimates in and information in a data frame
library(broom)
fit <- lm(R ~ BB, data = dat)
tidy(fit)
```

```
## # A tibble: 2 x 5
##   term        estimate std.error statistic  p.value
##   <chr>          <dbl>     <dbl>     <dbl>    <dbl>
## 1 (Intercept)    2.20     0.113       19.4 1.12e-70
## 2 BB             0.638    0.0344      18.5 1.35e-65
```

```
# add confidence intervals
tidy(fit, conf.int = TRUE)
```

```
## # A tibble: 2 x 7
##   term        estimate std.error statistic  p.value conf.low conf.high
##   <chr>          <dbl>     <dbl>     <dbl>    <dbl>    <dbl>     <dbl>
## 1 (Intercept)    2.20     0.113       19.4 1.12e-70    1.98      2.42
## 2 BB             0.638    0.0344      18.5 1.35e-65    0.570     0.705
```

```
# combine with group_by and summarize to get the table we want
dat %>%
  group_by(HR) %>%
  summarize(tidy(lm(R ~ BB), conf.int = TRUE))
```

```
## Warning: Returning more (or less) than 1 row per `summarise()` group was deprecated in
## dplyr 1.1.0.
## i Please use `reframe()` instead.
## i When switching from `summarise()` to `reframe()`, remember that `reframe()`
##   always returns an ungrouped data frame and adjust accordingly.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.

## `summarise()` has grouped output by 'HR'. You can override using the `.groups`
## argument.

## # A tibble: 18 x 8
## # Groups:   HR [9]
##       HR term        estimate std.error statistic  p.value conf.low conf.high
##    <dbl> <chr>          <dbl>     <dbl>     <dbl>    <dbl>    <dbl>     <dbl>
##  1   0.4 (Intercept)    1.36      0.631      2.16 4.05e- 2   0.0631     2.66
##  2   0.4 BB             0.734     0.208      3.54 1.54e- 3   0.308      1.16
##  3   0.5 (Intercept)    2.01      0.344      5.84 2.07e- 7   1.32       2.69
##  4   0.5 BB             0.566     0.110      5.14 3.02e- 6   0.346      0.786
##  5   0.6 (Intercept)    2.53      0.305      8.32 2.43e-13   1.93       3.14
```

```
## 6   0.6 BB            0.412   0.0974     4.23 4.80e- 5  0.219     0.605
## 7   0.7 (Intercept)   3.21    0.225     14.3  1.49e-30  2.76      3.65
## 8   0.7 BB            0.285   0.0705     4.05 7.93e- 5  0.146     0.425
## 9   0.8 (Intercept)   3.07    0.213     14.4  5.40e-31  2.65      3.49
## 10  0.8 BB            0.365   0.0653     5.59 9.13e- 8  0.236     0.494
## 11  0.9 (Intercept)   3.54    0.251     14.1  8.77e-29  3.05      4.04
## 12  0.9 BB            0.261   0.0751     3.47 6.85e- 4  0.112     0.409
## 13  1   (Intercept)   2.88    0.256     11.3  6.62e-21  2.37      3.39
## 14  1   BB            0.512   0.0751     6.81 3.28e-10  0.363     0.660
## 15  1.1 (Intercept)   3.21    0.300     10.7  6.46e-17  2.61      3.81
## 16  1.1 BB            0.454   0.0855     5.31 1.03e- 6  0.284     0.624
## 17  1.2 (Intercept)   3.40    0.291     11.7  2.33e-16  2.81      3.98
## 18  1.2 BB            0.440   0.0801     5.50 1.07e- 6  0.280     0.601
```

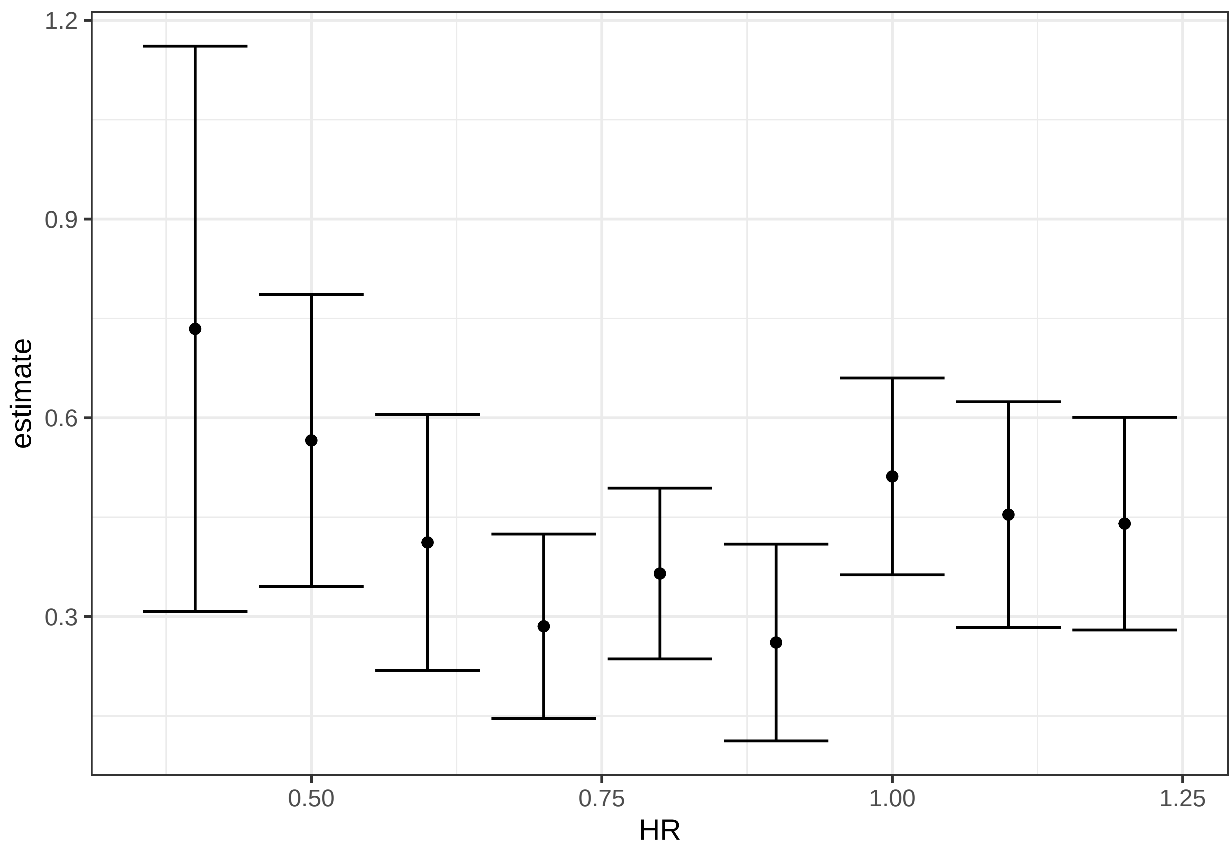
```r
# it's a data frame so we can filter and select the rows and columns we want
dat %>%
  group_by(HR) %>%
  summarize(tidy(lm(R ~ BB), conf.int = TRUE)) %>%
  filter(term == "BB") %>%
  select(HR, estimate, conf.low, conf.high)
```

```
## Warning: Returning more (or less) than 1 row per `summarise()` group was deprecated in
## dplyr 1.1.0.
## i Please use `reframe()` instead.
## i When switching from `summarise()` to `reframe()`, remember that `reframe()`
##   always returns an ungrouped data frame and adjust accordingly.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.

## `summarise()` has grouped output by 'HR'. You can override using the `.groups`
## argument.

## # A tibble: 9 x 4
## # Groups:   HR [9]
##      HR estimate conf.low conf.high
##   <dbl>    <dbl>    <dbl>     <dbl>
## 1   0.4    0.734    0.308     1.16
## 2   0.5    0.566    0.346     0.786
## 3   0.6    0.412    0.219     0.605
## 4   0.7    0.285    0.146     0.425
## 5   0.8    0.365    0.236     0.494
## 6   0.9    0.261    0.112     0.409
## 7   1      0.512    0.363     0.660
## 8   1.1    0.454    0.284     0.624
## 9   1.2    0.440    0.280     0.601
```

```r
# visualize the table with ggplot
dat %>%
  group_by(HR) %>%
  summarize(tidy(lm(R ~ BB), conf.int = TRUE)) %>%
  filter(term == "BB") %>%
  select(HR, estimate, conf.low, conf.high) %>%
  ggplot(aes(HR, y = estimate, ymin = conf.low, ymax = conf.high)) +
```

```r
  geom_errorbar() +
  geom_point()
```

```
## Warning: Returning more (or less) than 1 row per `summarise()` group was deprecated in
## dplyr 1.1.0.
## i Please use `reframe()` instead.
## i When switching from `summarise()` to `reframe()`, remember that `reframe()`
##   always returns an ungrouped data frame and adjust accordingly.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.

## `summarise()` has grouped output by 'HR'. You can override using the `.groups`
## argument.
```




```r
# EXTRA CODE TO DEMONSTRATE THE USE OF across()
# Compare the output of the 3 options below:
dat %>%
  group_by(HR) %>%
  summarize(tidy(lm(R ~ BB, data = .), conf.int = TRUE))
```

```
## Warning: Returning more (or less) than 1 row per `summarise()` group was deprecated in
## dplyr 1.1.0.
## i Please use `reframe()` instead.
## i When switching from `summarise()` to `reframe()`, remember that `reframe()`
```

```
##   always returns an ungrouped data frame and adjust accordingly.
## Call 'lifecycle::last_lifecycle_warnings()' to see where this warning was
## generated.

## 'summarise()' has grouped output by 'HR'. You can override using the '.groups'
## argument.

## # A tibble: 18 x 8
## # Groups:   HR [9]
##        HR term        estimate std.error statistic  p.value conf.low conf.high
##     <dbl> <chr>          <dbl>     <dbl>     <dbl>    <dbl>    <dbl>     <dbl>
##  1   0.4 (Intercept)    2.20     0.113        19.4 1.12e-70    1.98      2.42
##  2   0.4 BB             0.638    0.0344       18.5 1.35e-65    0.570     0.705
##  3   0.5 (Intercept)    2.20     0.113        19.4 1.12e-70    1.98      2.42
##  4   0.5 BB             0.638    0.0344       18.5 1.35e-65    0.570     0.705
##  5   0.6 (Intercept)    2.20     0.113        19.4 1.12e-70    1.98      2.42
##  6   0.6 BB             0.638    0.0344       18.5 1.35e-65    0.570     0.705
##  7   0.7 (Intercept)    2.20     0.113        19.4 1.12e-70    1.98      2.42
##  8   0.7 BB             0.638    0.0344       18.5 1.35e-65    0.570     0.705
##  9   0.8 (Intercept)    2.20     0.113        19.4 1.12e-70    1.98      2.42
## 10   0.8 BB             0.638    0.0344       18.5 1.35e-65    0.570     0.705
## 11   0.9 (Intercept)    2.20     0.113        19.4 1.12e-70    1.98      2.42
## 12   0.9 BB             0.638    0.0344       18.5 1.35e-65    0.570     0.705
## 13   1   (Intercept)    2.20     0.113        19.4 1.12e-70    1.98      2.42
## 14   1   BB             0.638    0.0344       18.5 1.35e-65    0.570     0.705
## 15   1.1 (Intercept)    2.20     0.113        19.4 1.12e-70    1.98      2.42
## 16   1.1 BB             0.638    0.0344       18.5 1.35e-65    0.570     0.705
## 17   1.2 (Intercept)    2.20     0.113        19.4 1.12e-70    1.98      2.42
## 18   1.2 BB             0.638    0.0344       18.5 1.35e-65    0.570     0.705
```

```r
  # Incorrect, will provide identical estimates for all groups

dat %>%
  group_by(HR) %>%
  summarize(tidy(lm(R ~ BB, data = across()), conf.int = TRUE))
```

```
## Warning: There was 1 warning in 'summarize()'.
## i In argument: 'tidy(lm(R ~ BB, data = across()), conf.int = TRUE)'.
## i In group 1: 'HR = 0.4'.
## Caused by warning:
## ! Using 'across()' without supplying '.cols' was deprecated in dplyr 1.1.0.
## i Please supply '.cols' instead.

## Warning: Returning more (or less) than 1 row per 'summarise()' group was deprecated in
## dplyr 1.1.0.
## i Please use 'reframe()' instead.
## i When switching from 'summarise()' to 'reframe()', remember that 'reframe()'
##   always returns an ungrouped data frame and adjust accordingly.
## Call 'lifecycle::last_lifecycle_warnings()' to see where this warning was
## generated.

## 'summarise()' has grouped output by 'HR'. You can override using the '.groups'
## argument.
```

```
## # A tibble: 18 x 8
## # Groups:   HR [9]
##       HR term        estimate std.error statistic  p.value conf.low conf.high
##    <dbl> <chr>          <dbl>     <dbl>     <dbl>    <dbl>    <dbl>     <dbl>
##  1   0.4 (Intercept)    1.36     0.631       2.16 4.05e- 2   0.0631     2.66
##  2   0.4 BB             0.734    0.208       3.54 1.54e- 3   0.308      1.16
##  3   0.5 (Intercept)    2.01     0.344       5.84 2.07e- 7   1.32       2.69
##  4   0.5 BB             0.566    0.110       5.14 3.02e- 6   0.346      0.786
##  5   0.6 (Intercept)    2.53     0.305       8.32 2.43e-13   1.93       3.14
##  6   0.6 BB             0.412    0.0974      4.23 4.80e- 5   0.219      0.605
##  7   0.7 (Intercept)    3.21     0.225      14.3  1.49e-30   2.76       3.65
##  8   0.7 BB             0.285    0.0705      4.05 7.93e- 5   0.146      0.425
##  9   0.8 (Intercept)    3.07     0.213      14.4  5.40e-31   2.65       3.49
## 10   0.8 BB             0.365    0.0653      5.59 9.13e- 8   0.236      0.494
## 11   0.9 (Intercept)    3.54     0.251      14.1  8.77e-29   3.05       4.04
## 12   0.9 BB             0.261    0.0751      3.47 6.85e- 4   0.112      0.409
## 13   1   (Intercept)    2.88     0.256      11.3  6.62e-21   2.37       3.39
## 14   1   BB             0.512    0.0751      6.81 3.28e-10   0.363      0.660
## 15   1.1 (Intercept)    3.21     0.300      10.7  6.46e-17   2.61       3.81
## 16   1.1 BB             0.454    0.0855      5.31 1.03e- 6   0.284      0.624
## 17   1.2 (Intercept)    3.40     0.291      11.7  2.33e-16   2.81       3.98
## 18   1.2 BB             0.440    0.0801      5.50 1.07e- 6   0.280      0.601
```

```r
  # Correct option 1, provides distinct estimates for all groups

dat %>%
  group_by(HR) %>%
  summarize(tidy(lm(R ~ BB), conf.int = TRUE))
```

```
## Warning: Returning more (or less) than 1 row per `summarise()` group was deprecated in
## dplyr 1.1.0.
## i Please use `reframe()` instead.
## i When switching from `summarise()` to `reframe()`, remember that `reframe()`
##   always returns an ungrouped data frame and adjust accordingly.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.

## `summarise()` has grouped output by 'HR'. You can override using the `.groups`
## argument.

## # A tibble: 18 x 8
## # Groups:   HR [9]
##       HR term        estimate std.error statistic  p.value conf.low conf.high
##    <dbl> <chr>          <dbl>     <dbl>     <dbl>    <dbl>    <dbl>     <dbl>
##  1   0.4 (Intercept)    1.36     0.631       2.16 4.05e- 2   0.0631     2.66
##  2   0.4 BB             0.734    0.208       3.54 1.54e- 3   0.308      1.16
##  3   0.5 (Intercept)    2.01     0.344       5.84 2.07e- 7   1.32       2.69
##  4   0.5 BB             0.566    0.110       5.14 3.02e- 6   0.346      0.786
##  5   0.6 (Intercept)    2.53     0.305       8.32 2.43e-13   1.93       3.14
##  6   0.6 BB             0.412    0.0974      4.23 4.80e- 5   0.219      0.605
##  7   0.7 (Intercept)    3.21     0.225      14.3  1.49e-30   2.76       3.65
##  8   0.7 BB             0.285    0.0705      4.05 7.93e- 5   0.146      0.425
##  9   0.8 (Intercept)    3.07     0.213      14.4  5.40e-31   2.65       3.49
```

```
## 10   0.8 BB            0.365    0.0653      5.59 9.13e- 8   0.236      0.494
## 11   0.9 (Intercept)   3.54     0.251      14.1  8.77e-29   3.05       4.04
## 12   0.9 BB            0.261    0.0751      3.47 6.85e- 4   0.112      0.409
## 13   1   (Intercept)   2.88     0.256      11.3  6.62e-21   2.37       3.39
## 14   1   BB            0.512    0.0751      6.81 3.28e-10   0.363      0.660
## 15   1.1 (Intercept)   3.21     0.300      10.7  6.46e-17   2.61       3.81
## 16   1.1 BB            0.454    0.0855      5.31 1.03e- 6   0.284      0.624
## 17   1.2 (Intercept)   3.40     0.291      11.7  2.33e-16   2.81       3.98
## 18   1.2 BB            0.440    0.0801      5.50 1.07e- 6   0.280      0.601
```

```
  # Correct option 2, provides distinct estimates for all groups
```

Assessment: Advanced dplyr, part 2

We have investigated the relationship between fathers' heights and sons' heights. But what about other parent-child relationships? Does one parent's height have a stronger association with child height? How does the child's gender affect this relationship in heights? Are any differences that we observe statistically significant?

The galton dataset is a sample of one male and one female child from each family in the GaltonFamilies dataset. The pair column denotes whether the pair is father and daughter, father and son, mother and daughter, or mother and son.

Create the galton dataset using the code below:

```
library(tidyverse)
library(HistData)
data("GaltonFamilies")
# set.seed(1) # if you are using R 3.5 or earlier
set.seed(1, sample.kind = "Rounding") # if you are using R 3.6 or later
```

```
## Warning in set.seed(1, sample.kind = "Rounding"): non-uniform 'Rounding'
## sampler used
```

```
galton <- GaltonFamilies %>%
    group_by(family, gender) %>%
    sample_n(1) %>%
    ungroup() %>%
    gather(parent, parentHeight, father:mother) %>%
    mutate(child = ifelse(gender == "female", "daughter", "son")) %>%
    unite(pair, c("parent", "child"))

galton
```

```
## # A tibble: 710 x 8
##    family midparentHeight children childNum gender childHeight pair
##    <fct>            <dbl>    <int>    <int> <fct>        <dbl> <chr>
##  1 001               75.4        4        2 female        69.2 father_daughter
##  2 001               75.4        4        1 male          73.2 father_son
##  3 002               73.7        4        4 female        65.5 father_daughter
##  4 002               73.7        4        2 male          72.5 father_son
```

```
##   5 003                 72.1        2         2 female        68   father_daughter
##   6 003                 72.1        2         1 male          71   father_son
##   7 004                 72.1        5         5 female        63   father_daughter
##   8 004                 72.1        5         2 male          68.5 father_son
##   9 005                 69.1        6         5 female        62.5 father_daughter
##  10 005                 69.1        6         1 male          72   father_son
## # i 700 more rows
## # i 1 more variable: parentHeight <dbl>
```

Question 8 Group by pair and summarize the number of observations in each group.
How many father-daughter pairs are in the dataset?
How many mother-son pairs are in the dataset?

```
galton %>% group_by(pair) %>%
  summarise(n = n())
```

```
## # A tibble: 4 x 2
##   pair                n
##   <chr>           <int>
## 1 father_daughter   176
## 2 father_son        179
## 3 mother_daughter   176
## 4 mother_son        179
```

Question 9 Calculate the correlation coefficients for fathers and daughters, fathers and sons, mothers and daughters and mothers and sons. Which pair has the strongest correlation in heights?

```
galton %>% group_by(pair) %>% summarise(cor(parentHeight, childHeight))
```

```
## # A tibble: 4 x 2
##   pair            `cor(parentHeight, childHeight)`
##   <chr>                                      <dbl>
## 1 father_daughter                            0.401
## 2 father_son                                 0.430
## 3 mother_daughter                            0.383
## 4 mother_son                                 0.343
```

```
galton
```

```
## # A tibble: 710 x 8
##    family midparentHeight children childNum gender childHeight pair
##    <fct>            <dbl>    <int>    <int> <fct>        <dbl> <chr>
##  1 001               75.4        4        2 female        69.2 father_daughter
##  2 001               75.4        4        1 male          73.2 father_son
##  3 002               73.7        4        4 female        65.5 father_daughter
##  4 002               73.7        4        2 male          72.5 father_son
##  5 003               72.1        2        2 female        68   father_daughter
##  6 003               72.1        2        1 male          71   father_son
##  7 004               72.1        5        5 female        63   father_daughter
##  8 004               72.1        5        2 male          68.5 father_son
##  9 005               69.1        6        5 female        62.5 father_daughter
```

```
## 10 005                69.1        6        1 male          72    father_son
## # i 700 more rows
## # i 1 more variable: parentHeight <dbl>
```

Question 10a What is the estimate of the father-daughter coefficient?

```r
library(broom)
galton %>%
  filter(pair == 'father_daughter') %>%
  do(tidy(lm(childHeight ~ parentHeight, data = .), conf.int = TRUE))
```

```
## # A tibble: 2 x 7
##   term         estimate std.error statistic  p.value conf.low conf.high
##   <chr>           <dbl>     <dbl>     <dbl>    <dbl>    <dbl>     <dbl>
## 1 (Intercept)    40.1      4.16        9.65 6.50e-18   31.9      48.3
## 2 parentHeight    0.345    0.0599      5.77 3.56e- 8    0.227     0.464
```

For every 1-inch increase in mother's height, how many inches does the typical son's height increase?

```r
galton %>%
  filter(pair == 'mother_son') %>%
  do(tidy(lm(childHeight ~ parentHeight, data = .), conf.int = TRUE))
```

```
## # A tibble: 2 x 7
##   term         estimate std.error statistic  p.value conf.low conf.high
##   <chr>           <dbl>     <dbl>     <dbl>    <dbl>    <dbl>     <dbl>
## 1 (Intercept)    44.9      5.02        8.94 4.96e-16   35.0      54.8
## 2 parentHeight    0.381    0.0784      4.86 2.59e- 6    0.226     0.535
```

Question 10b Which sets of parent-child heights are significantly correlated at a p-value cut off of 0.05?

```r
galton %>% group_by(pair) %>%
  do(tidy(lm(childHeight ~ parentHeight, data = .), conf.int = TRUE)) %>%
  filter(term == 'parentHeight')
```

```
## # A tibble: 4 x 8
## # Groups:   pair [4]
##   pair            term   estimate std.error statistic p.value conf.low conf.high
##   <chr>           <chr>     <dbl>     <dbl>     <dbl>   <dbl>    <dbl>     <dbl>
## 1 father_daughter paren~    0.345    0.0599      5.77 3.56e-8    0.227     0.464
## 2 father_son      paren~    0.443    0.0700      6.33 1.94e-9    0.305     0.581
## 3 mother_daughter paren~    0.394    0.0720      5.47 1.56e-7    0.252     0.536
## 4 mother_son      paren~    0.381    0.0784      4.86 2.59e-6    0.226     0.535
```

```r
# All p values are < 0.05
```

2.4: Regression and Baseball

Building a Better Offensive Metric for Baseball

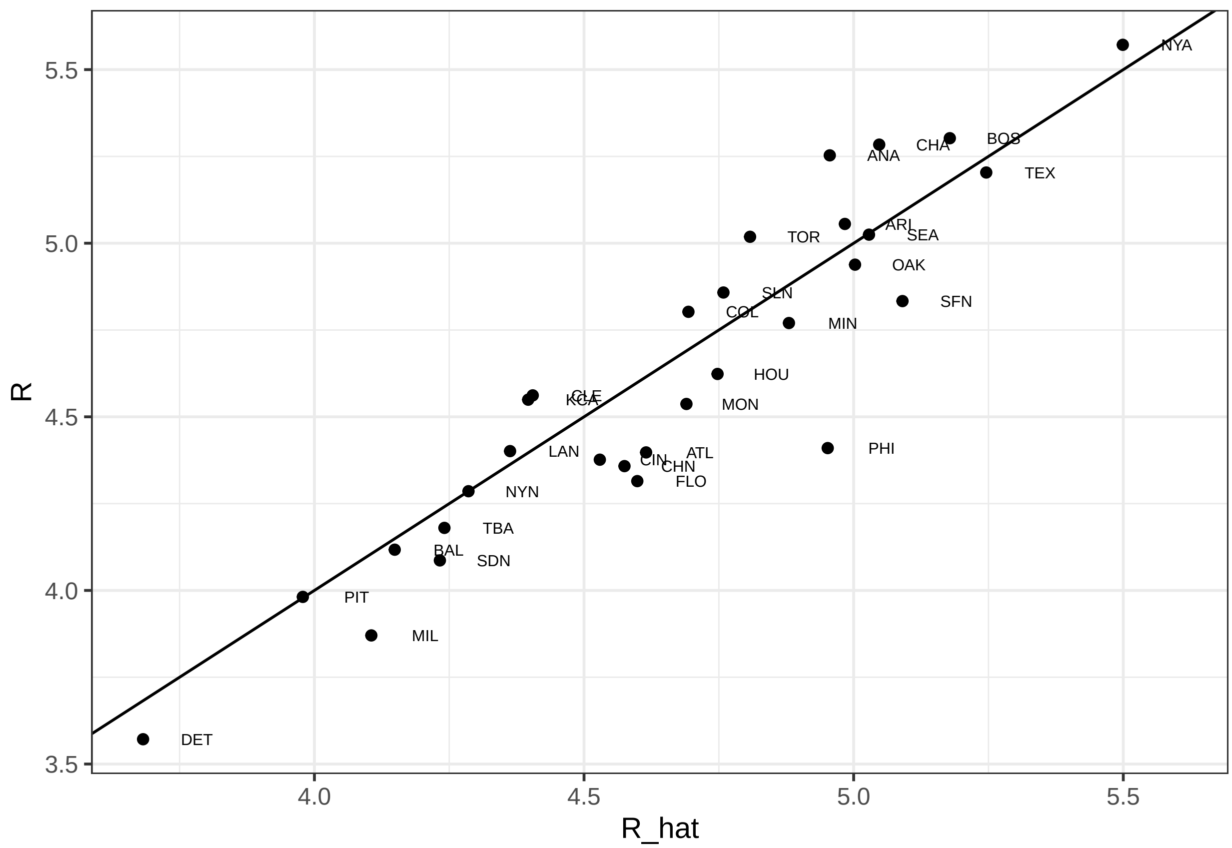
```r
# linear regression with two variables
library(Lahman)
fit <- Teams %>%
  filter(yearID %in% 1961:2001) %>%
  mutate(BB = BB/G, HR = HR/G,  R = R/G) %>%
  lm(R ~ BB + HR, data = .)
  tidy(fit, conf.int = TRUE)
```

```
## # A tibble: 3 x 7
##   term        estimate std.error statistic   p.value conf.low conf.high
##   <chr>          <dbl>     <dbl>     <dbl>     <dbl>    <dbl>     <dbl>
## 1 (Intercept)    1.74     0.0824      21.2 7.62e- 83    1.58      1.91
## 2 BB             0.387    0.0270      14.3 1.20e- 42    0.334     0.440
## 3 HR             1.56     0.0490      31.9 1.78e-155    1.47      1.66
```

```r
# regression with BB, singles, doubles, triples, HR
fit <- Teams %>%
  filter(yearID %in% 1961:2001) %>%
  mutate(BB = BB / G,
         singles = (H - X2B - X3B - HR) / G,
         doubles = X2B / G,
         triples = X3B / G,
         HR = HR / G,
         R = R / G) %>%
  lm(R ~ BB + singles + doubles + triples + HR, data = .)
coefs <- tidy(fit, conf.int = TRUE)
coefs
```

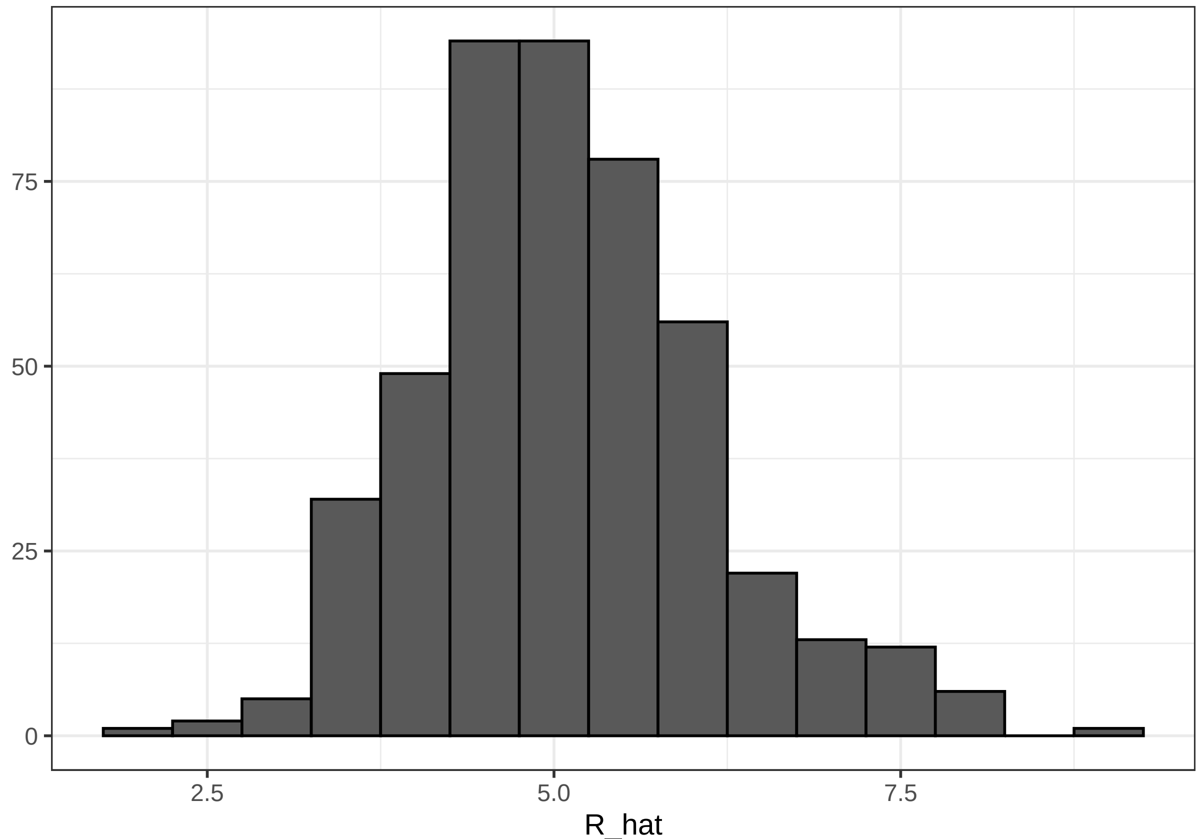
```
## # A tibble: 6 x 7
##   term        estimate std.error statistic   p.value conf.low conf.high
##   <chr>          <dbl>     <dbl>     <dbl>     <dbl>    <dbl>     <dbl>
## 1 (Intercept)   -2.77     0.0862     -32.1 4.76e-157   -2.94     -2.60
## 2 BB             0.371    0.0117      31.6 1.87e-153    0.348     0.394
## 3 singles        0.519    0.0127      40.8 8.67e-217    0.494     0.544
## 4 doubles        0.771    0.0226      34.1 8.44e-171    0.727     0.816
## 5 triples        1.24     0.0768      16.1 2.12e- 52    1.09      1.39
## 6 HR             1.44     0.0243      59.3 0            1.40      1.49
```

```r
# predict number of runs for each team in 2002 and plot
Teams %>%
  filter(yearID %in% 2002) %>%
  mutate(BB = BB/G,
         singles = (H-X2B-X3B-HR)/G,
         doubles = X2B/G,
         triples =X3B/G,
         HR=HR/G,
         R=R/G)  %>%
  mutate(R_hat = predict(fit, newdata = .)) %>%
  ggplot(aes(R_hat, R, label = teamID)) +
  geom_point() +
  geom_text(nudge_x=0.1, cex = 2) +
  geom_abline()
```




```
# average number of team plate appearances per game
pa_per_game <- Batting %>% filter(yearID == 2002) %>%
  group_by(teamID) %>%
  summarize(pa_per_game = sum(AB+BB)/max(G)) %>%
  pull(pa_per_game) %>%
  mean

# compute per-plate-appearance rates for players available in 2002 using previous data
players <- Batting %>% filter(yearID %in% 1999:2001) %>%
  group_by(playerID) %>%
  mutate(PA = BB + AB) %>%
  summarize(G = sum(PA)/pa_per_game,
    BB = sum(BB)/G,
    singles = sum(H-X2B-X3B-HR)/G,
    doubles = sum(X2B)/G,
    triples = sum(X3B)/G,
    HR = sum(HR)/G,
    AVG = sum(H)/sum(AB),
    PA = sum(PA)) %>%
  filter(PA >= 300) %>%
  select(-G) %>%
  mutate(R_hat = predict(fit, newdata = .))

# plot player-specific predicted runs
qplot(R_hat, data = players, geom = "histogram", binwidth = 0.5, color = I("black"))
```

```
## Warning: `qplot()` was deprecated in ggplot2 3.4.0.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```




```r
# add 2002 salary of each player
players <- Salaries %>%
  filter(yearID == 2002) %>%
  select(playerID, salary) %>%
  right_join(players, by="playerID")

# add defensive position
position_names <- c("G_p","G_c","G_1b","G_2b","G_3b","G_ss","G_lf","G_cf","G_rf")
tmp_tab <- Appearances %>%
  filter(yearID == 2002) %>%
  group_by(playerID) %>%
  summarize_at(position_names, sum) %>%
  ungroup()
pos <- tmp_tab %>%
  select(position_names) %>%
  apply(., 1, which.max)
```

```
## Warning: Using an external vector in selections was deprecated in tidyselect 1.1.0.
## i Please use `all_of()` or `any_of()` instead.
##   # Was:
```

```
##   data %>% select(position_names)
##
##   # Now:
##   data %>% select(all_of(position_names))
##
## See <https://tidyselect.r-lib.org/reference/faq-external-vector.html>.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```

```
players <- data_frame(playerID = tmp_tab$playerID, POS = position_names[pos]) %>%
  mutate(POS = str_to_upper(str_remove(POS, "G_"))) %>%
  filter(POS != "P") %>%
  right_join(players, by="playerID") %>%
  filter(!is.na(POS)  & !is.na(salary))
```

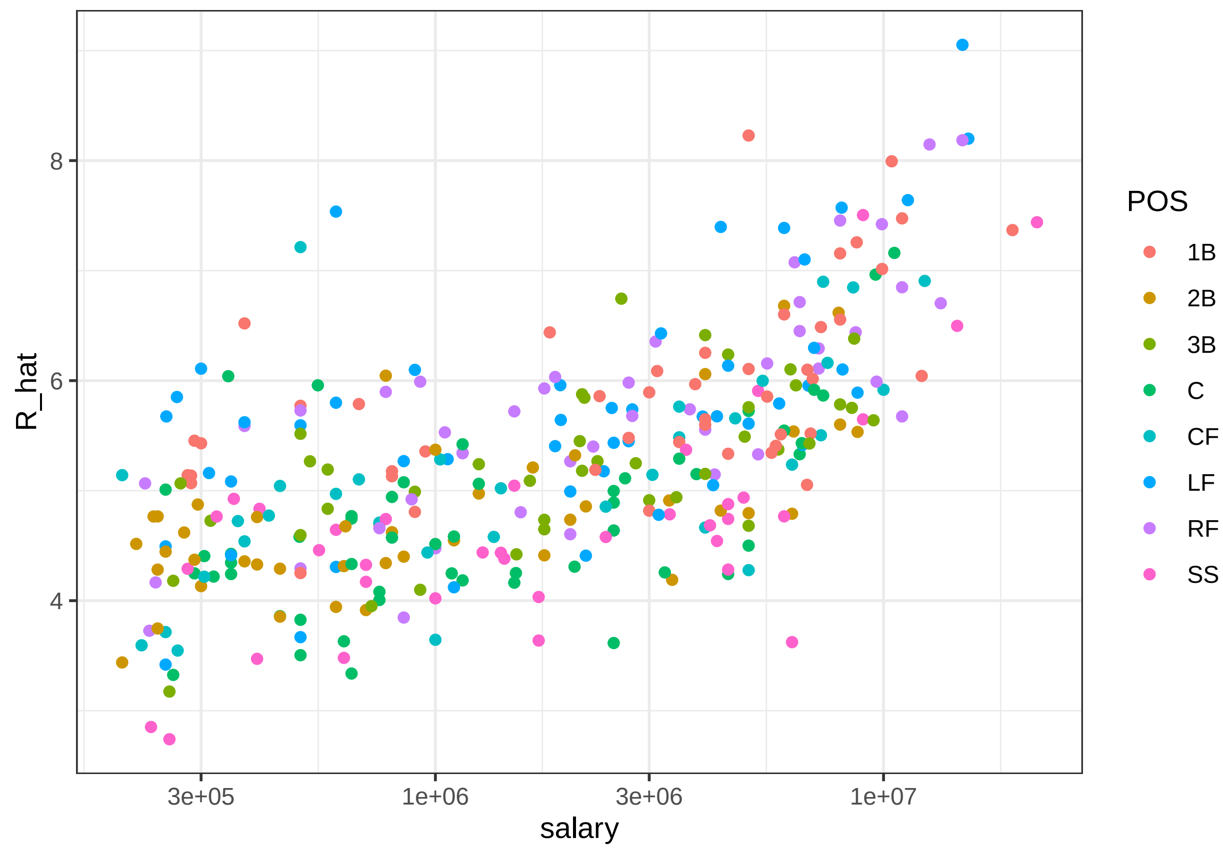
```
## Warning: `data_frame()` was deprecated in tibble 1.1.0.
## i Please use `tibble()` instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```

```
# add players' first and last names
# NOTE: In old versions of the Lahman library, the "People" dataset was called "Master"
# The following code may need to be modified if you have not recently updated the Lahman library.
players <- People %>%
  select(playerID, nameFirst, nameLast, debut) %>%
  mutate(debut = as.Date(debut)) %>%
  right_join(players, by="playerID")

# top 10 players
players %>% select(nameFirst, nameLast, POS, salary, R_hat) %>%
  arrange(desc(R_hat)) %>%
  top_n(10)
```

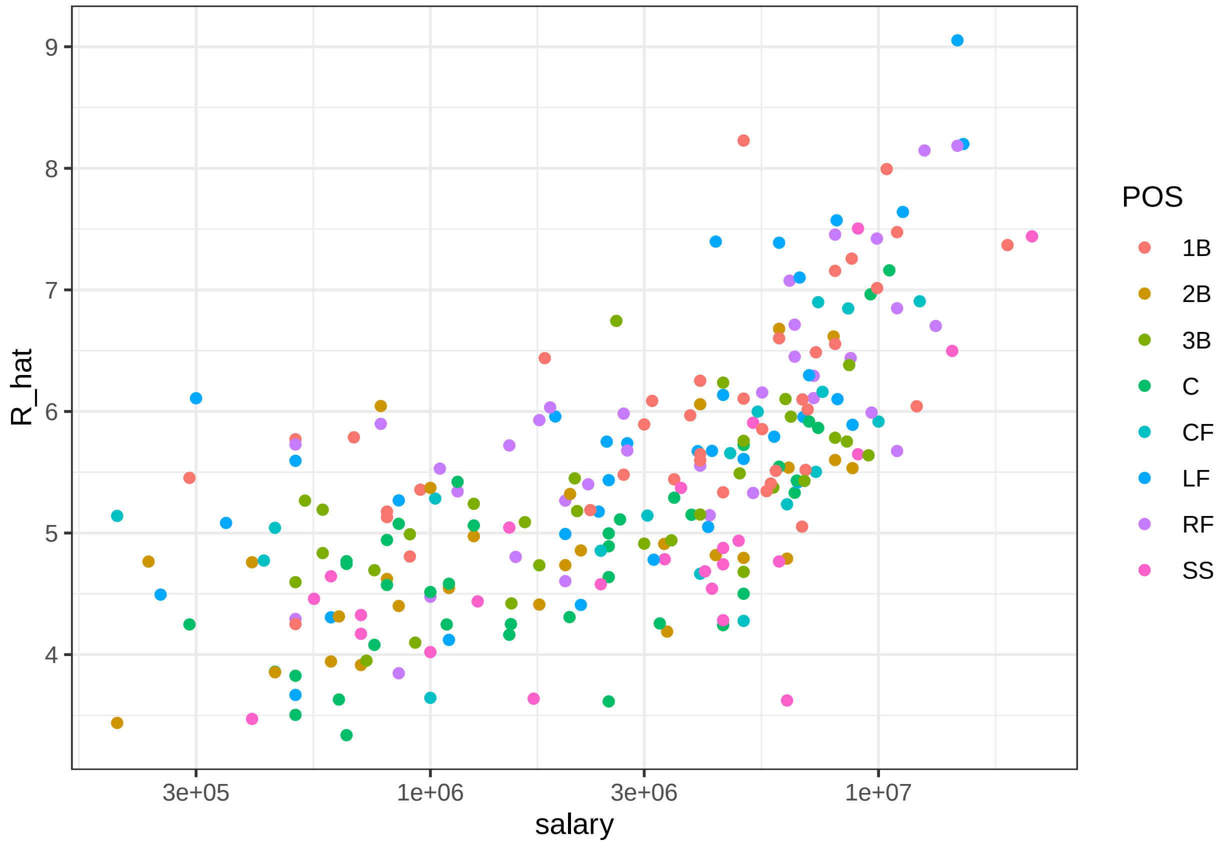
```
## Selecting by R_hat
```

```
##    nameFirst    nameLast POS   salary R_hat
## 1      Barry       Bonds  LF 15000000  9.05
## 2       Todd      Helton  1B  5000000  8.23
## 3      Manny     Ramirez  LF 15462727  8.20
## 4      Sammy        Sosa  RF 15000000  8.19
## 5      Larry      Walker  RF 12666667  8.15
## 6      Jason      Giambi  1B 10428571  7.99
## 7    Chipper       Jones  LF 11333333  7.64
## 8      Brian       Giles  LF  8063003  7.57
## 9     Albert      Pujols  LF   600000  7.54
## 10     Nomar Garciaparra  SS  9000000  7.51
```

```
# players with a higher metric have higher salaries
players %>% ggplot(aes(salary, R_hat, color = POS)) +
  geom_point() +
  scale_x_log10()
```




```
# remake plot without players that debuted after 1998
library(lubridate)
players %>% filter(year(debut) < 1998) %>%
 ggplot(aes(salary, R_hat, color = POS)) +
  geom_point() +
  scale_x_log10()
```



Building a Better Offensive Metric for Baseball: Linear Programming

A way to actually pick the players for the team can be done using what computer scientists call linear programming. Although we don't go into this topic in detail in this course, we include the code anyway:

```
library(reshape2)
```

```
##
## Attaching package: 'reshape2'

## The following object is masked from 'package:tidyr':
##
##     smiths
```

```
library(lpSolve)

players <- players %>% filter(debut <= "1997-01-01" & debut > "1988-01-01")
constraint_matrix <- acast(players, POS ~ playerID, fun.aggregate = length, value.var = 'R_hat')
npos <- nrow(constraint_matrix)
constraint_matrix <- rbind(constraint_matrix, salary = players$salary)
constraint_dir <- c(rep("==", npos), "<=")
constraint_limit <- c(rep(1, npos), 50*10^6)
lp_solution <- lp("max", players$R_hat,
                  constraint_matrix, constraint_dir, constraint_limit,
                  all.int = TRUE)
```

This algorithm chooses these 9 players: (We actually get 8, maybe the code is wrong or the data was modified.)

```
our_team <- players %>%
  filter(lp_solution$solution == 1) %>%
  arrange(desc(R_hat))
our_team %>% select(nameFirst, nameLast, POS, salary, R_hat)
```

```
##   nameFirst    nameLast POS   salary R_hat
## 1     Larry      Walker  RF 12666667  8.15
## 2     Nomar Garciaparra  SS  9000000  7.51
## 3      Luis    Gonzalez  LF  4333333  7.40
## 4      Mike      Piazza   C 10571429  7.16
## 5       Jim     Edmonds  CF  7333333  6.90
## 6      Phil       Nevin  3B  2600000  6.75
## 7      Greg    Colbrunn  1B  1800000  6.44
## 8     Terry    Shumpert  2B   775000  6.04
```

```
my_scale <- function(x) (x - median(x))/mad(x)
players %>% mutate(BB = my_scale(BB),
                   singles = my_scale(singles),
                   doubles = my_scale(doubles),
                   triples = my_scale(triples),
                   HR = my_scale(HR),
                   AVG = my_scale(AVG),
                   R_hat = my_scale(R_hat)) %>%
    filter(playerID %in% our_team$playerID) %>%
    select(nameFirst, nameLast, BB, singles, doubles, triples, HR, AVG, R_hat) %>%
    arrange(desc(R_hat))
```

```
##   nameFirst    nameLast      BB singles doubles triples     HR   AVG R_hat
## 1     Larry      Walker  1.0605  0.6554   0.922   1.562  1.566 2.835 2.904
## 2     Nomar Garciaparra  0.0274  1.6371   3.118   0.336  0.625 3.197 2.244
## 3      Luis    Gonzalez  0.7046  0.0000   1.413   0.537  1.355 1.829 2.133
## 4      Mike      Piazza  0.3129 -0.0547  -0.242  -1.274  2.035 1.252 1.891
## 5       Jim     Edmonds  1.8074 -1.1409   0.674  -0.674  1.264 0.579 1.621
## 6      Phil       Nevin  0.4909 -0.6479   0.764  -1.098  1.548 0.728 1.463
## 7      Greg    Colbrunn  0.2703  0.6546   0.784   0.585  0.475 1.375 1.148
## 8     Terry    Shumpert -0.1576  0.1221   1.326   3.908 -0.123 0.859 0.744
```

Regression Fallacy

Regression can bring about errors in reasoning, especially when interpreting individual observations. The example showed in the video demonstrates that the "sophomore slump" observed in the data is caused by regressing to the mean.
The code to create a table with player ID, their names, and their most played position:

```
library(Lahman)
playerInfo <- Fielding %>%
    group_by(playerID) %>%
    arrange(desc(G)) %>%
    slice(1) %>%
```

```r
    ungroup %>%
    left_join(People, by="playerID") %>%
    select(playerID, nameFirst, nameLast, POS)
```

The code to create a table with only the ROY award winners and add their batting statistics:

```r
ROY <- AwardsPlayers %>%
    filter(awardID == "Rookie of the Year") %>%
    left_join(playerInfo, by="playerID") %>%
    rename(rookie_year = yearID) %>%
    right_join(Batting, by="playerID") %>%
    mutate(AVG = H/AB) %>%
    filter(POS != "P")
```

The code to keep only the rookie and sophomore seasons and remove players who did not play sophomore seasons:

```r
ROY <- ROY %>%
    filter(yearID == rookie_year | yearID == rookie_year+1) %>%
    group_by(playerID) %>%
    mutate(rookie = ifelse(yearID == min(yearID), "rookie", "sophomore")) %>%
    filter(n() == 2) %>%
    ungroup %>%
    select(playerID, rookie_year, rookie, nameFirst, nameLast, AVG)
```

The code to use the spread function to have one column for the rookie and sophomore years batting averages:

```r
ROY <- ROY %>% spread(rookie, AVG) %>% arrange(desc(rookie))
ROY
```

```
## # A tibble: 108 x 6
##    playerID  rookie_year nameFirst nameLast rookie sophomore
##    <chr>           <int> <chr>     <chr>     <dbl>     <dbl>
##  1 mccovwi01        1959 Willie    McCovey   0.354     0.238
##  2 suzukic01        2001 Ichiro    Suzuki    0.350     0.321
##  3 bumbral01        1973 Al        Bumbry    0.337     0.233
##  4 lynnfr01         1975 Fred      Lynn      0.331     0.314
##  5 pujolal01        2001 Albert    Pujols    0.329     0.314
##  6 troutmi01        2012 Mike      Trout     0.326     0.323
##  7 braunry02        2007 Ryan      Braun     0.324     0.285
##  8 olivato01        1964 Tony      Oliva     0.323     0.321
##  9 hargrmi01        1974 Mike      Hargrove  0.323     0.303
## 10 darkal01         1948 Al        Dark      0.322     0.276
## # i 98 more rows
```

The code to calculate the proportion of players who have a lower batting average their sophomore year:

```r
mean(ROY$sophomore - ROY$rookie <= 0)
```

```
## [1] 0.704
```

The code to do the similar analysis on all players that played the 2013 and 2014 seasons and batted more than 130 times (minimum to win Rookie of the Year):

```r
two_years <- Batting %>%
    filter(yearID %in% 2013:2014) %>%
    group_by(playerID, yearID) %>%
    filter(sum(AB) >= 130) %>%
    summarize(AVG = sum(H)/sum(AB)) %>%
    ungroup %>%
    spread(yearID, AVG) %>%
    filter(!is.na(`2013`) & !is.na(`2014`)) %>%
    left_join(playerInfo, by="playerID") %>%
    filter(POS!="P") %>%
    select(-POS) %>%
    arrange(desc(`2013`)) %>%
    select(nameFirst, nameLast, `2013`, `2014`)
```

```
## `summarise()` has grouped output by 'playerID'. You can override using the
## `.groups` argument.
```

```r
two_years
```

```
## # A tibble: 312 x 4
##    nameFirst nameLast `2013` `2014`
##    <chr>     <chr>     <dbl>  <dbl>
##  1 Miguel    Cabrera   0.348  0.313
##  2 Hanley    Ramirez   0.345  0.283
##  3 Michael   Cuddyer   0.331  0.332
##  4 Scooter   Gennett   0.324  0.289
##  5 Joe       Mauer     0.324  0.277
##  6 Mike      Trout     0.323  0.287
##  7 Chris     Johnson   0.321  0.263
##  8 Freddie   Freeman   0.319  0.288
##  9 Yasiel    Puig      0.319  0.296
## 10 Yadier    Molina    0.319  0.282
## # i 302 more rows
```

The code to see what happens to the worst performers of 2013:

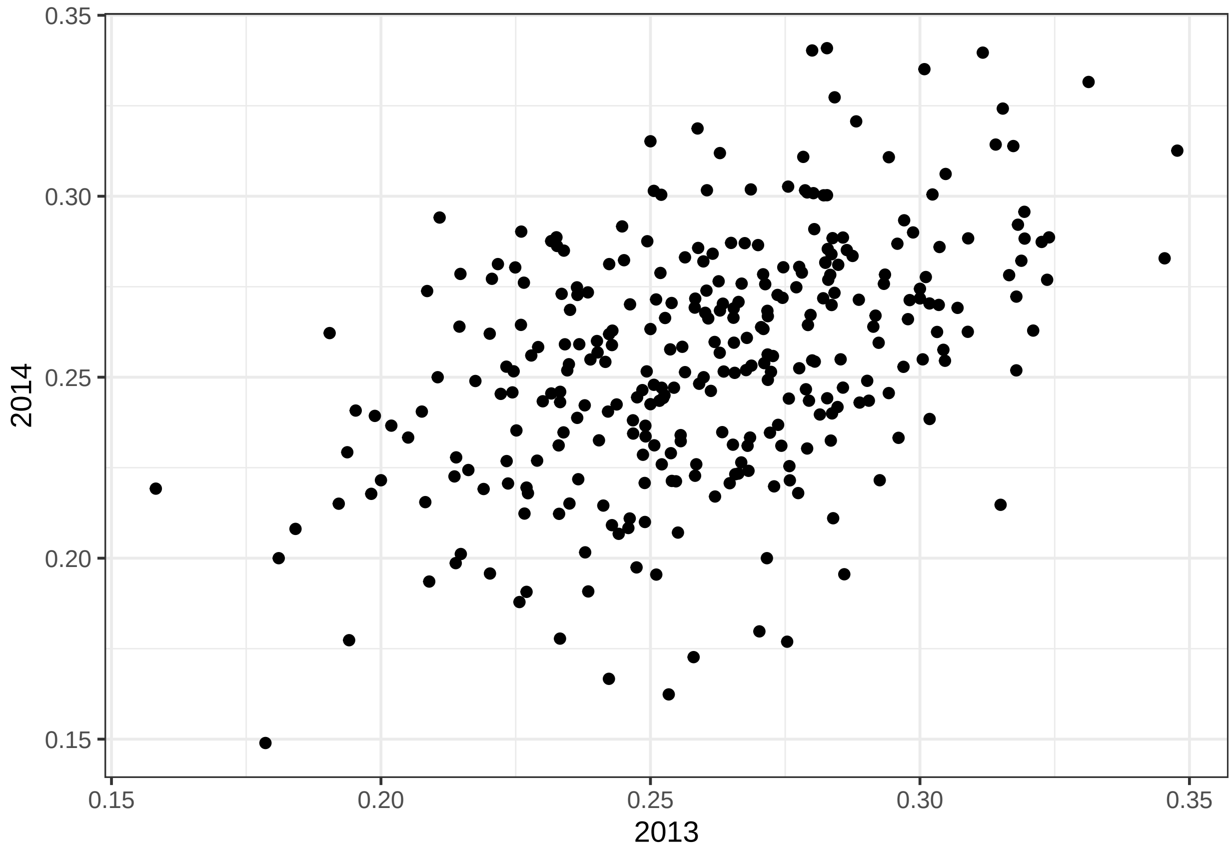
```r
arrange(two_years, `2013`)
```

```
## # A tibble: 312 x 4
##    nameFirst nameLast  `2013` `2014`
##    <chr>     <chr>      <dbl>  <dbl>
##  1 Danny     Espinosa   0.158  0.219
##  2 Dan       Uggla      0.179  0.149
##  3 Jeff      Mathis     0.181  0.2
##  4 B. J.     Upton      0.184  0.208
##  5 Adam      Rosales    0.190  0.262
##  6 Aaron     Hicks      0.192  0.215
##  7 Chris     Colabello  0.194  0.229
##  8 J. P.     Arencibia  0.194  0.177
```

```
##   9 Tyler      Flowers     0.195  0.241
## 10 Ryan       Hanigan     0.198  0.218
## # i 302 more rows
```

The code to see the correlation for performance in two separate years:

```
qplot(`2013`, `2014`, data = two_years)
```




```
summarize(two_years, cor(`2013`,`2014`))
```

```
## # A tibble: 1 x 1
##   `cor(\`2013\`, \`2014\`)`
##                       <dbl>
## 1                     0.460
```

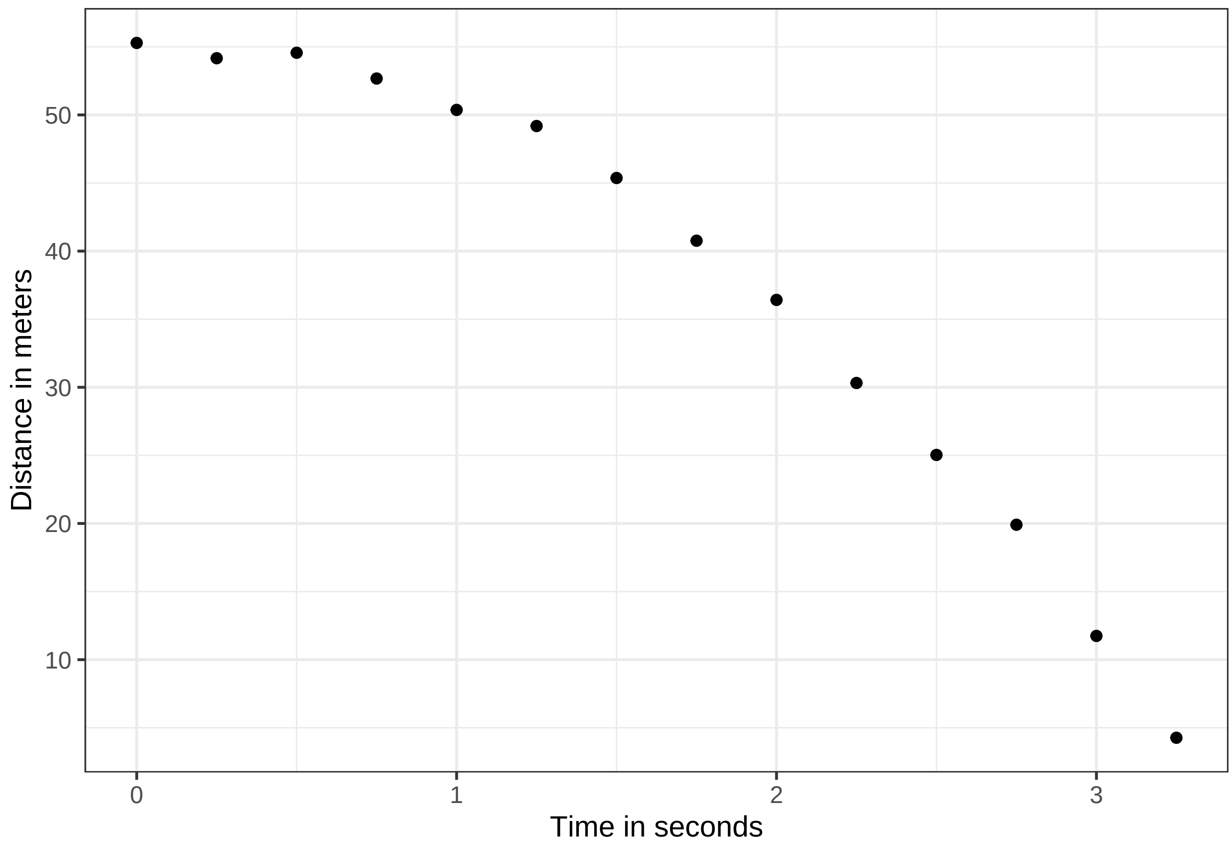
Measurement Error Models

Up to now, all our linear regression examples have been applied to two or more random variables. We assume the pairs are bivariate normal and use this to motivate a linear model.
Another use for linear regression is with measurement error models, where it is common to have a non-random covariate (such as time). Randomness is introduced from measurement error rather than sampling or natural variability.
The code to use dslabs function rfalling_object to generate simulations of dropping balls:

```r
library(dslabs)
falling_object <- rfalling_object()
```

The code to draw the trajectory of the ball:

```r
falling_object %>%
    ggplot(aes(time, observed_distance)) +
    geom_point() +
    ylab("Distance in meters") +
    xlab("Time in seconds")
```



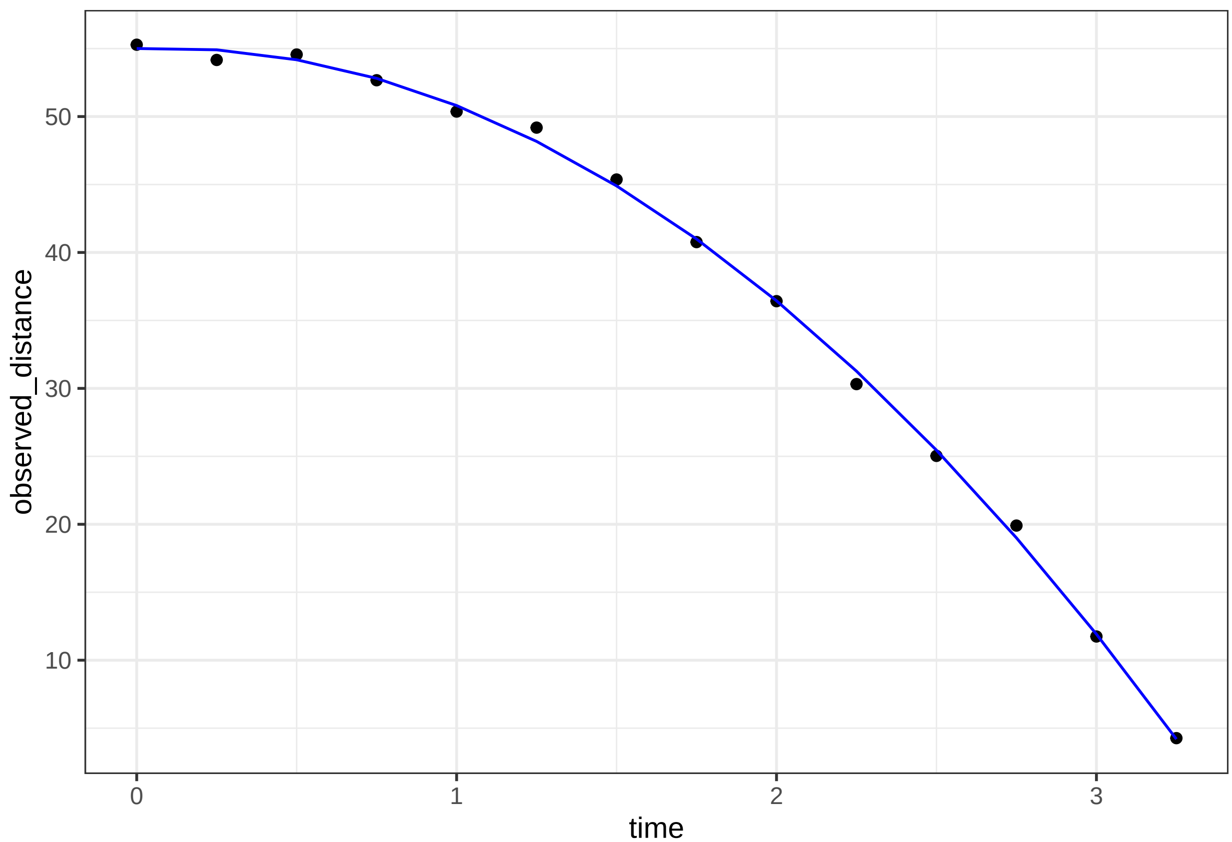
The code to use the lm() function to estimate the coefficients:

```r
library(broom)
fit <- falling_object %>%
    mutate(time_sq = time^2) %>%
    lm(observed_distance~time+time_sq, data=.)

tidy(fit)
```

```
## # A tibble: 3 x 5
##   term        estimate std.error statistic  p.value
##   <chr>          <dbl>     <dbl>     <dbl>    <dbl>
## 1 (Intercept)   55.0       0.434     127.   9.15e-19
## 2 time           0.888     0.620       1.43 1.80e- 1
## 3 time_sq       -5.08      0.184     -27.7  1.61e-11
```

The code to check if the estimated parabola fits the data:

```
augment(fit) %>%
    ggplot() +
    geom_point(aes(time, observed_distance)) +
    geom_line(aes(time, .fitted), col = "blue")
```



The code to see the summary statistic of the regression:

```
tidy(fit, conf.int = TRUE)
```

```
## # A tibble: 3 x 7
##   term        estimate std.error statistic  p.value conf.low conf.high
##   <chr>          <dbl>     <dbl>     <dbl>    <dbl>    <dbl>     <dbl>
## 1 (Intercept)   55.0       0.434    127.   9.15e-19   54.1       56.0
## 2 time           0.888     0.620      1.43 1.80e- 1   -0.476      2.25
## 3 time_sq       -5.08      0.184    -27.7  1.61e-11   -5.49      -4.68
```

Assessment: Regression and Baseball, part 2

Use the Teams data frame from the Lahman package. Fit a multivariate linear regression model to obtain the effects of BB and HR on Runs (R) in 1971. Use the tidy() function in the broom package to obtain the results in a data frame.

```r
library(Lahman)
library(broom)
fit <- Teams %>% filter(yearID == 1971) %>% lm(R ~ BB + HR, data = .) %>% tidy()
fit
```

```
## # A tibble: 3 x 5
##   term        estimate std.error statistic p.value
##   <chr>          <dbl>     <dbl>     <dbl>   <dbl>
## 1 (Intercept)  257.      112.         2.31 0.0314
## 2 BB             0.414     0.210      1.97 0.0625
## 3 HR             1.30      0.431      3.01 0.00673
```

Question 9a What is the estimate for the effect of BB on runs?

```r
fit %>% filter(term == 'BB') %>% select(term, estimate)
```

```
## # A tibble: 1 x 2
##   term  estimate
##   <chr>    <dbl>
## 1 BB       0.414
```

What is the estimate for the effect of HR on runs?

```r
fit %>% filter(term == 'HR') %>% select(term, estimate)
```

```
## # A tibble: 1 x 2
##   term  estimate
##   <chr>    <dbl>
## 1 HR        1.30
```

Question 9b Interpret the p-values for the estimates using a cutoff of 0.05 and considering the year 1971 as a sample to make inference on the population of all baseball games across years.

Which of the following is the correct interpretation?

```r
fit %>% filter(term %in% c('BB','HR')) %>% select(term, p.value)
```

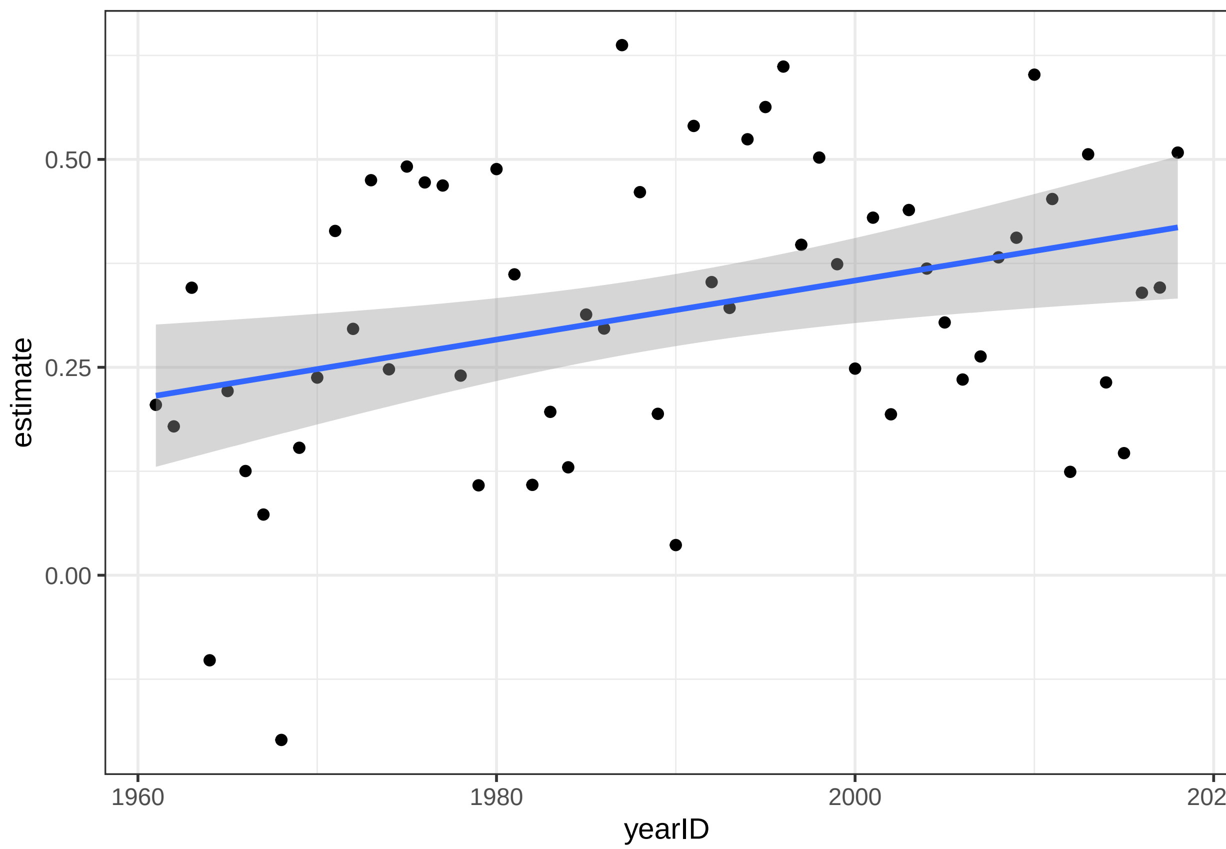
```
## # A tibble: 2 x 2
##   term  p.value
##   <chr>   <dbl>
## 1 BB    0.0625
## 2 HR    0.00673
```

Question 10 Repeat the above exercise to find the effects of BB and HR on runs (R) for every year from 1961 to 2018 using do() and the broom package.

Make a scatterplot of the estimate for the effect of BB on runs over time and add a trend line with confidence intervals.

```
fit <- Teams %>% filter(yearID %in% 1961:2018) %>%
  group_by(yearID) %>%
  do(tidy(lm(R ~ BB + HR, data = .))) %>%
  ungroup()
fit %>%
  filter(term == 'BB') %>%
  ggplot(aes(yearID, estimate)) +
  geom_point() +
  geom_smooth(method = 'lm')
```

```
## `geom_smooth()` using formula = 'y ~ x'
```



Question 11 Fit a linear model on the results from Question 10 to determine the effect of year on the impact of BB. That is, determine how the estimated coefficients of BB from the models in Question 10 can be predicted by the year (recall that we grouped the data by year before fitting the models, so we have different estimated coefficients for each year).

```
fit <- fit %>% filter(term == 'BB') %>% select(yearID,estimate) %>%
  do(tidy(lm(estimate ~ yearID, data = .)))
```

For each additional year, by what value does the impact of BB on runs change?

```
fit %>% filter(term == 'yearID') %>% pull(estimate)
```

```
## [1] 0.00355
```

What is the p-value for this effect?

```
fit %>% filter(term == 'yearID') %>% pull(p.value)
```

```
## [1] 0.00807
```

Assessment: Linear Models (Verified Learners only)

This assessment has 6 multi-part questions that will all use the setup below.
Game attendance in baseball varies partly as a function of how well a team is playing.
Load the Lahman library. The Teams data frame contains an attendance column. This is the total attendance for the season. To calculate average attendance, divide by the number of games played, as follows:

```
library(tidyverse)
library(broom)
library(Lahman)
Teams_small <- Teams %>%
    filter(yearID %in% 1961:2001) %>%
    mutate(avg_attendance = attendance/G)
```

Question 1a Use runs (R) per game to predict average attendance.
For every 1 run scored per game, average attendance increases by how much?

```
Teams_small %>% mutate(RG = R/G) %>%
  do(tidy(lm(avg_attendance ~ RG, data = .))) %>%
  filter(term == 'RG') %>% pull(estimate)
```

```
## [1] 4117
```

Use home runs (HR) per game to predict average attendance.
For every 1 home run hit per game, average attendance increases by how much?

```
Teams_small %>% mutate(HRG = HR/G) %>%
  do(tidy(lm(avg_attendance ~ HRG, data = .))) %>%
  filter(term == 'HRG') %>% pull(estimate)
```

```
## [1] 8113
```

Question 1b Use number of wins to predict average attendance; do not normalize for number of games.
For every game won in a season, how much does average attendance increase?

```
Teams_small %>% do(tidy(lm(avg_attendance ~ W, data = .))) %>%
  filter(term == 'W') %>% pull(estimate)
```

```
## [1] 121
```

Suppose a team won zero games in a season.
Predict the average attendance.

```
Teams_small %>% do(tidy(lm(avg_attendance ~ W, data = .))) %>%
  filter(term == '(Intercept)') %>% pull(estimate)
```

```
## [1] 1129
```

Question 1c Use year to predict average attendance.
How much does average attendance increase each year?

```
Teams_small %>% do(tidy(lm(avg_attendance ~ yearID, data = .)))
```

```
## # A tibble: 2 x 5
##   term        estimate std.error statistic  p.value
##   <chr>          <dbl>     <dbl>     <dbl>    <dbl>
## 1 (Intercept) -473937.   20632.      -23.0 1.63e-94
## 2 yearID          244.      10.4      23.5 5.90e-98
```

Question 2 Game wins, runs per game and home runs per game are positively correlated with attendance. We saw in the course material that runs per game and home runs per game are correlated with each other. Are wins and runs per game or wins and home runs per game correlated? Use the Teams_small data once again.
What is the correlation coefficient for runs per game and wins?

```
Teams_small %>% mutate(RG = R/G) %>% summarise(cor(RG,W))
```

```
##   cor(RG, W)
## 1      0.412
```

What is the correlation coefficient for home runs per game and wins?

```
Teams_small %>% mutate(HRG = HR/G) %>% summarise(cor(HRG,W))
```

```
##   cor(HRG, W)
## 1       0.274
```

Question 3 Stratify Teams_small by wins: divide number of wins by 10 and then round to the nearest integer. Filter to keep only strata 5 through 10. (The other strata have fewer than 20 data points, too few for our analyses).

```
WTS <- Teams_small %>% mutate(W = round(W/10), RG = R/G, HRG = HR/G) %>% group_by(W) %>% filter(W %in%
```

Question 3a How many observations are in the 8 win strata?

```r
WTS %>% filter(W == 8) %>% nrow()
```

```
## [1] 338
```

Question 3b Calculate the slope of the regression line predicting average attendance given runs per game for each of the win strata.
Which win stratum has the largest regression line slope?

```r
WTS %>% do(tidy(lm(avg_attendance ~ RG, data = .))) %>% filter(term == 'RG')
```

```
## # A tibble: 6 x 6
## # Groups:   W [6]
##       W term  estimate std.error statistic  p.value
##   <dbl> <chr>    <dbl>     <dbl>     <dbl>    <dbl>
## 1     5 RG       4362.     1112.      3.92 4.20e- 4
## 2     6 RG       4343.      903.      4.81 5.05e- 6
## 3     7 RG       3888.      464.      8.38 1.08e-14
## 4     8 RG       3128.      380.      8.23 4.06e-15
## 5     9 RG       3701.      607.      6.09 4.75e- 9
## 6    10 RG       3107.      827.      3.76 2.80e- 4
```

Calculate the slope of the regression line predicting average attendance given HR per game for each of the win strata.
Which win stratum has the largest regression line slope?

```r
WTS %>% do(tidy(lm(avg_attendance ~ HRG, data = .))) %>% filter(term == 'HRG')
```

```
## # A tibble: 6 x 6
## # Groups:   W [6]
##       W term  estimate std.error statistic  p.value
##   <dbl> <chr>    <dbl>     <dbl>     <dbl>    <dbl>
## 1     5 HRG     10192.     3423.      2.98 5.41e- 3
## 2     6 HRG      7032.     2444.      2.88 4.85e- 3
## 3     7 HRG      8931.     1126.      7.93 1.70e-13
## 4     8 HRG      6301.      886.      7.11 7.05e-12
## 5     9 HRG      5863.     1279.      4.58 7.58e- 6
## 6    10 HRG      4917.     1976.      2.49 1.44e- 2
```

Question 4 Fit a multivariate regression determining the effects of runs per game, home runs per game, wins, and year on average attendance. Use the original Teams_small wins column, not the win strata from question 3.
What is the estimate of the effect of runs per game, home runs per game and wins on average attendance?

```r
Teams_small <- Teams_small %>% mutate(RG = R/G, HRG = HR/G)
Teams_small %>% do(tidy(lm(avg_attendance ~ RG + HRG + W + yearID, data = .)))
```

```
## # A tibble: 5 x 5
##   term         estimate std.error statistic  p.value
##   <chr>           <dbl>     <dbl>     <dbl>    <dbl>
## 1 (Intercept) -456674.   21815.     -20.9   3.00e-81
```

```
## 2 RG              322.     331.       0.972 3.31e- 1
## 3 HRG            1798.     690.       2.61  9.24e- 3
## 4 W               117.       9.88    11.8   2.79e-30
## 5 yearID          230.      11.2     20.6   7.10e-79
```

Question 5 Use the multivariate regression model from Question 4. Suppose a team averaged 5 runs per game, 1.2 home runs per game, and won 80 games in a season. Use the predict() function to generate predictions for this team.
What would this team's average attendance be in 2002?

```r
fit <- Teams_small %>%
  lm(avg_attendance ~ RG + HRG + W + yearID, data = .)

predict(fit, data.frame(RG = 5, HRG = 1.2, W = 80, yearID = 2002))
```

```
##     1
## 16149
```

What would this team's average attendance be in 1960?

```r
predict(fit, data.frame(RG = 5, HRG = 1.2, W = 80, yearID = 1960))
```

```
##    1
## 6505
```

Question 6 Use your model from Question 4 to predict average attendance for teams in 2002 in the original Teams data frame.
What is the correlation between the predicted attendance and actual attendance?

```r
newdata <- Teams %>%
    filter(yearID == 2002) %>%
    mutate(avg_attendance = attendance/G,
           RG = R/G,
           HRG = HR/G)
preds <- predict(fit, newdata)
cor(preds, newdata$avg_attendance)
```

```
## [1] 0.519
```

Section 3: Confounding

```r
library(tidyverse)
library(dplyr)
```

Correlation is Not Causation

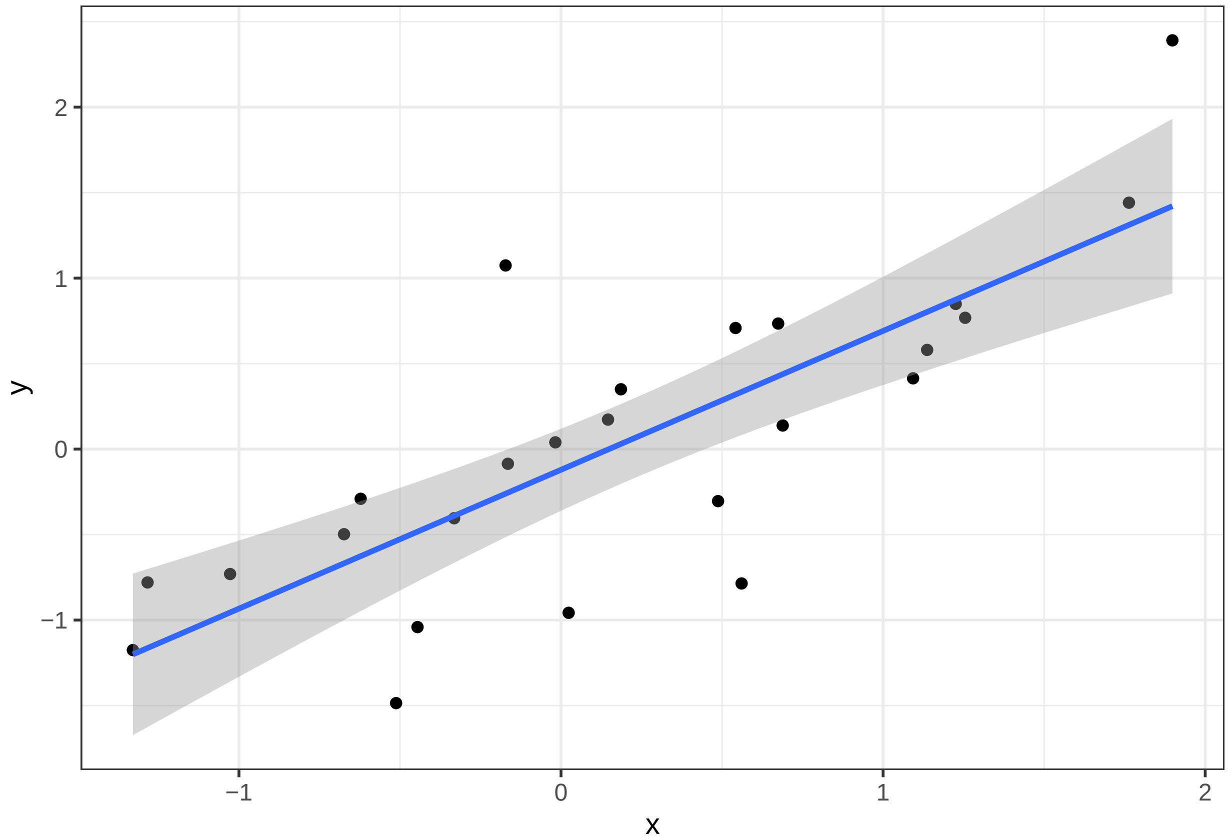
Association/correlation is not causation.
p-hacking is a topic of much discussion because it is a problem in scientific publications. Because publishers tend to reward statistically significant results over negative results, there is an incentive to report significant results.

```
# generate the Monte Carlo simulation
N <- 25
g <- 1000000
sim_data <- tibble(group = rep(1:g, each = N), x = rnorm(N * g), y = rnorm(N * g))

# calculate correlation between X,Y for each group
res <- sim_data %>%
  group_by(group) %>%
  summarize(r = cor(x, y)) %>%
  arrange(desc(r))
res
```

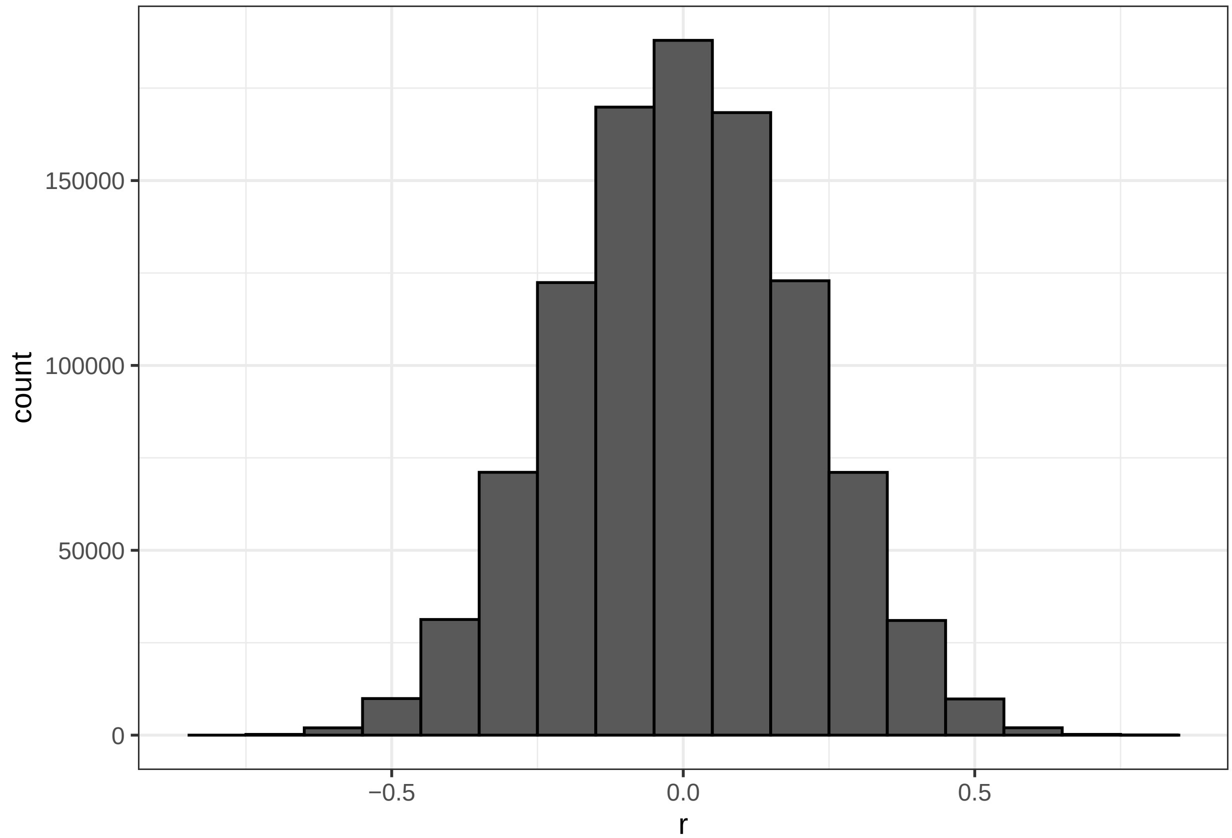
```
## # A tibble: 1,000,000 x 2
##     group     r
##     <int> <dbl>
##  1 840003 0.794
##  2 767028 0.791
##  3 971856 0.776
##  4 212248 0.768
##  5  60200 0.761
##  6  27045 0.756
##  7 114409 0.756
##  8 755537 0.754
##  9 422986 0.753
## 10 789165 0.747
## # i 999,990 more rows
```

```
# plot points from the group with maximum correlation
sim_data %>% filter(group == res$group[which.max(res$r)]) %>%
  ggplot(aes(x, y)) +
  geom_point() +
  geom_smooth(method = "lm")
```

```
## `geom_smooth()` using formula = 'y ~ x'
```




```r
# histogram of correlation in Monte Carlo simulations
res %>% ggplot(aes(x=r)) + geom_histogram(binwidth = 0.1, color = "black")
```




```r
# linear regression on group with maximum correlation
library(broom)
sim_data %>%
  filter(group == res$group[which.max(res$r)]) %>%
  summarize(tidy(lm(y ~ x)))
```

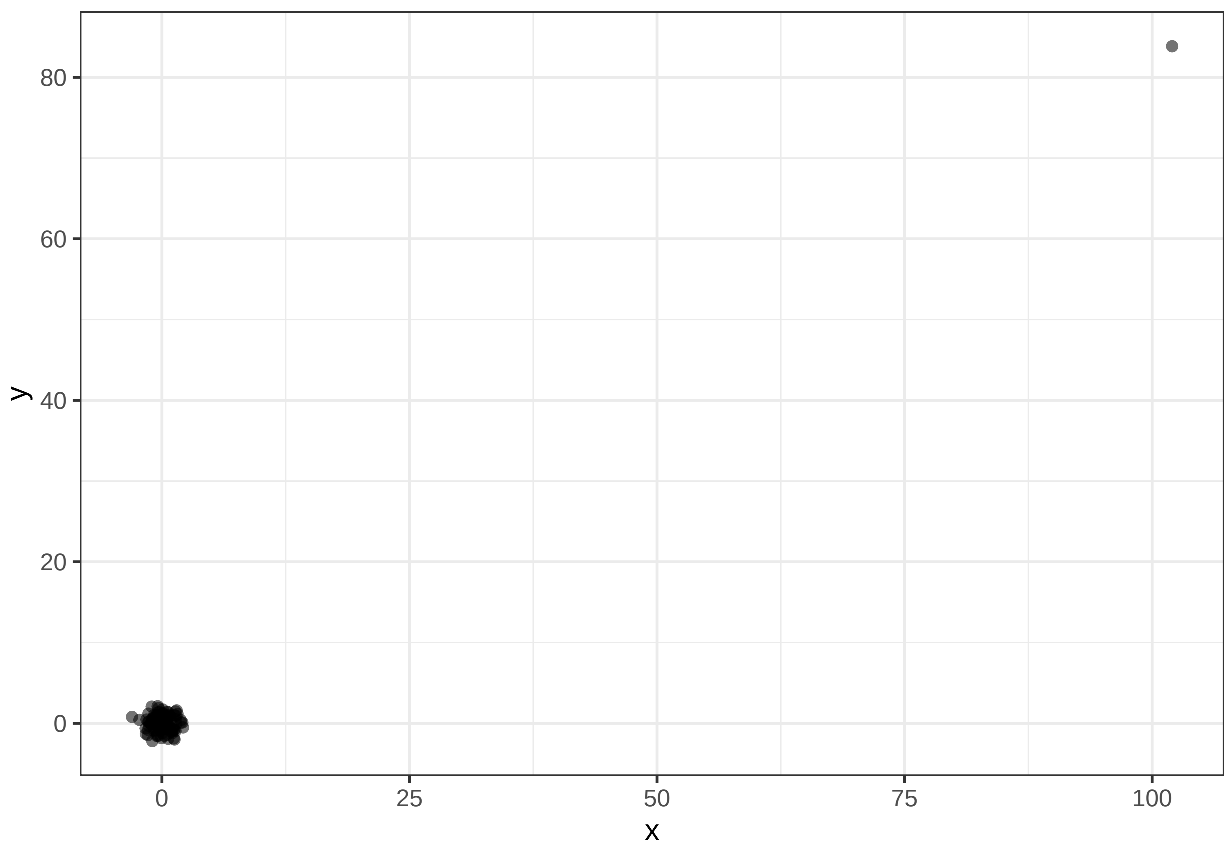
```
## Warning: Returning more (or less) than 1 row per `summarise()` group was deprecated in
## dplyr 1.1.0.
## i Please use `reframe()` instead.
## i When switching from `summarise()` to `reframe()`, remember that `reframe()`
##   always returns an ungrouped data frame and adjust accordingly.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```

```
## # A tibble: 2 x 5
##   term        estimate std.error statistic    p.value
##   <chr>          <dbl>     <dbl>     <dbl>      <dbl>
## 1 (Intercept)   -0.121     0.116     -1.04 0.309
## 2 x              0.812     0.130      6.27 0.00000213
```

Outliers

Correlations can be caused by outliers.
The Spearman correlation is calculated based on the ranks of data.

```r
# simulate independent X, Y and standardize all except entry 23
# note that you may get different values than those shown in the video depending on R version
set.seed(1985)
x <- rnorm(100,100,1)
y <- rnorm(100,84,1)
x[-23] <- scale(x[-23])
y[-23] <- scale(y[-23])

# plot shows the outlier
qplot(x, y, alpha = 0.5) + theme(legend.position = 'none')
```



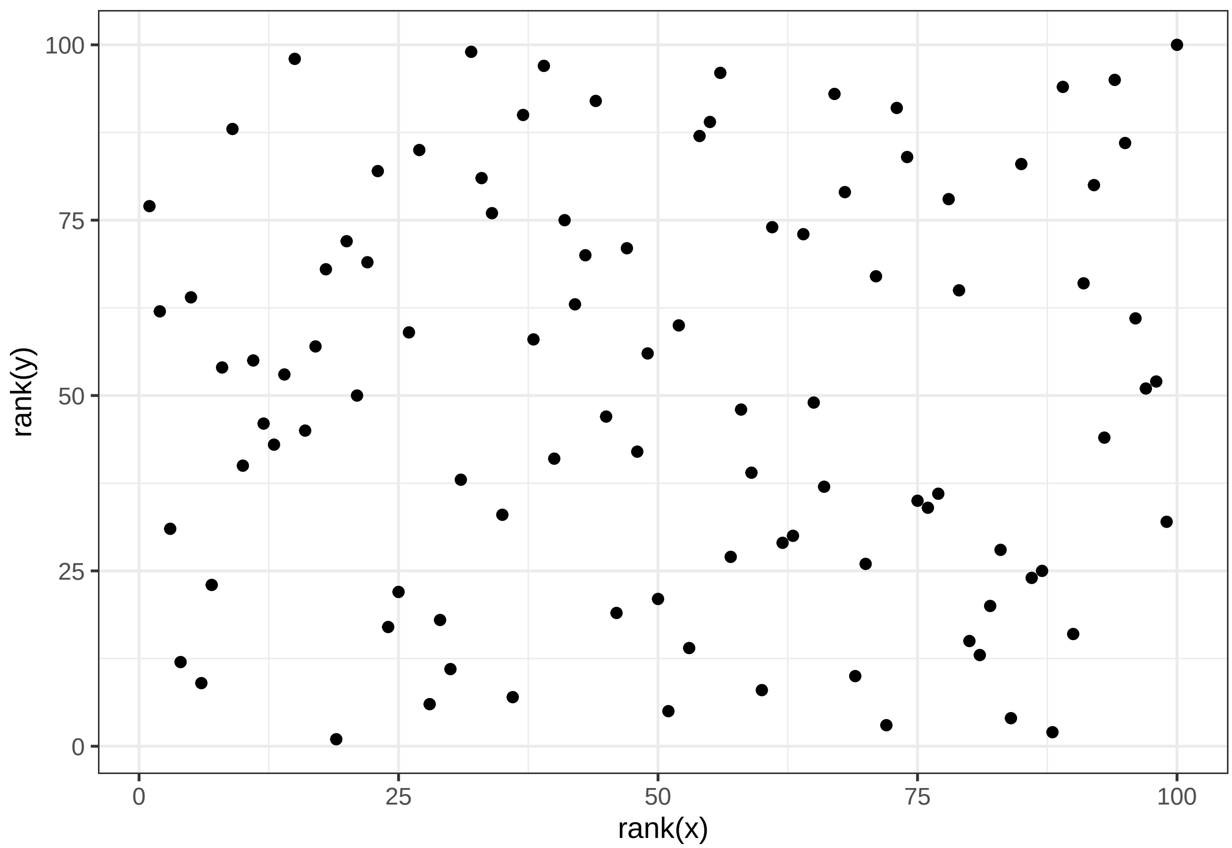

```r
# outlier makes it appear there is correlation
cor(x,y) # correlation with outlier
```

```
## [1] 0.988
```

```r
cor(x[-23], y[-23]) # correlation without outlier
```

```
## [1] -0.0442
```

```r
# use rank instead
qplot(rank(x), rank(y))
```




```
cor(rank(x), rank(y))
```

```
## [1] 0.00251
```

```
# Spearman correlation with cor function
cor(x, y, method = "spearman")
```

```
## [1] 0.00251
```

Reversing Cause and Effect

Another way association can be confused with causation is when the cause and effect are reversed.
As discussed in the video, in the Galton data, when father and son were reversed in the regression, the model was technically correct. The estimates and p-values were obtained correctly as well. What was incorrect was the interpretation of the model.

```
# cause and effect reversal using son heights to predict father heights
library(HistData)
data("GaltonFamilies")
set.seed(1983)
galton_heights <- GaltonFamilies %>%
  filter(gender == "male") %>%
  group_by(family) %>%
  sample_n(1) %>%
```

```r
  ungroup() %>%
  select(father, childHeight) %>%
  rename(son = childHeight)

galton_heights %>% summarize(tidy(lm(father ~ son)))
```

```
## Warning: Returning more (or less) than 1 row per `summarise()` group was deprecated in
## dplyr 1.1.0.
## i Please use `reframe()` instead.
## i When switching from `summarise()` to `reframe()`, remember that `reframe()`
##   always returns an ungrouped data frame and adjust accordingly.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.

## # A tibble: 2 x 5
##   term        estimate std.error statistic  p.value
##   <chr>          <dbl>     <dbl>     <dbl>    <dbl>
## 1 (Intercept)   45.2      4.24       10.7  8.44e-21
## 2 son            0.345    0.0612      5.64 6.72e- 8
```

Confounders

If X and Y are correlated, we call Z a confounder if changes in Z causes changes in both X and Y.

```r
# UC-Berkeley admission data
library(dslabs)
data(admissions)
admissions
```

```
##    major gender admitted applicants
## 1      A    men       62        825
## 2      B    men       63        560
## 3      C    men       37        325
## 4      D    men       33        417
## 5      E    men       28        191
## 6      F    men        6        373
## 7      A  women       82        108
## 8      B  women       68         25
## 9      C  women       34        593
## 10     D  women       35        375
## 11     E  women       24        393
## 12     F  women        7        341
```

```r
# percent men and women accepted
admissions %>% group_by(gender) %>%
  summarize(percentage =
              round(sum(admitted*applicants)/sum(applicants),1))
```

```
## # A tibble: 2 x 2
##   gender percentage
##   <chr>       <dbl>
## 1 men          44.5
## 2 women        30.3
```

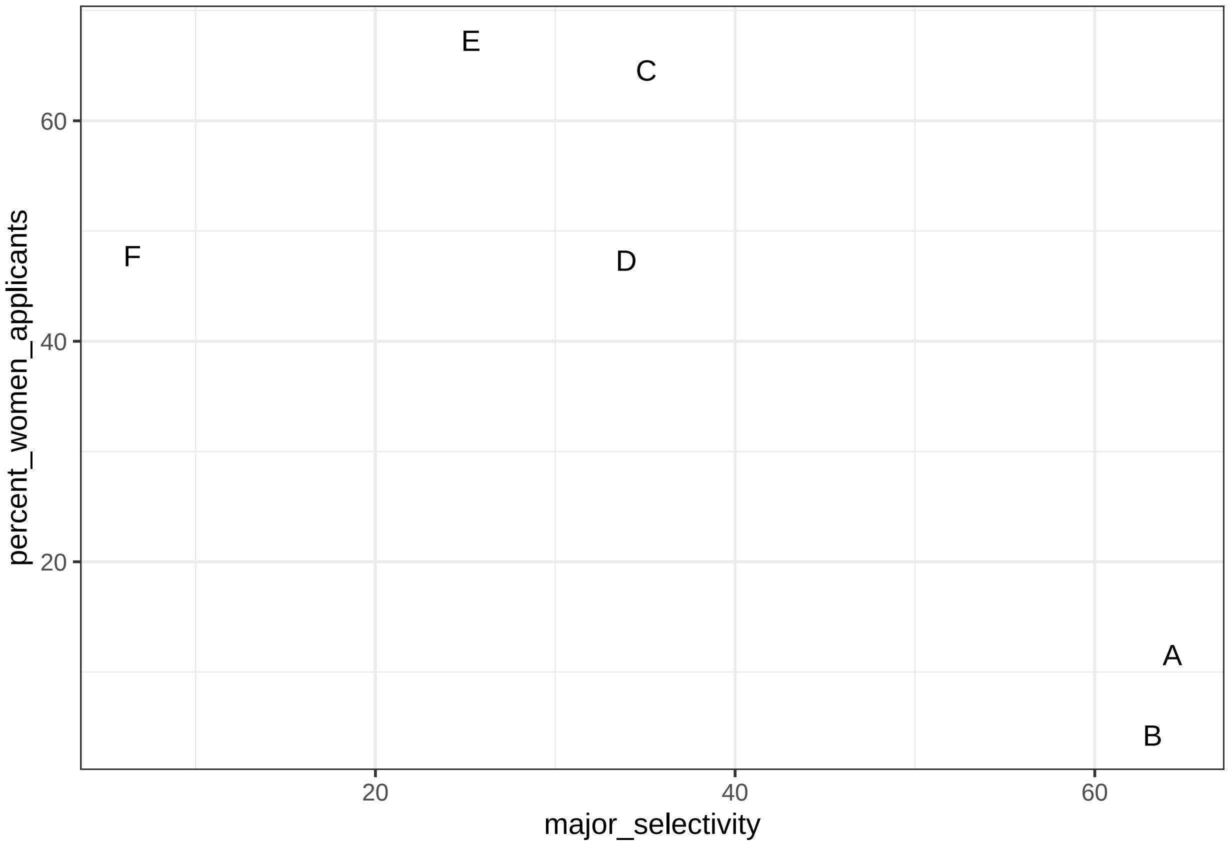
```r
# test whether gender and admission are independent
admissions %>% group_by(gender) %>%
  summarize(total_admitted = round(sum(admitted / 100 * applicants)),
            not_admitted = sum(applicants) - sum(total_admitted)) %>%
  select(-gender) %>%
  summarize(tidy(chisq.test(.)))
```

```
## # A tibble: 1 x 4
##   statistic  p.value parameter method
##       <dbl>    <dbl>     <int> <chr>
## 1      91.6 1.06e-21         1 Pearson's Chi-squared test with Yates' continuit~
```

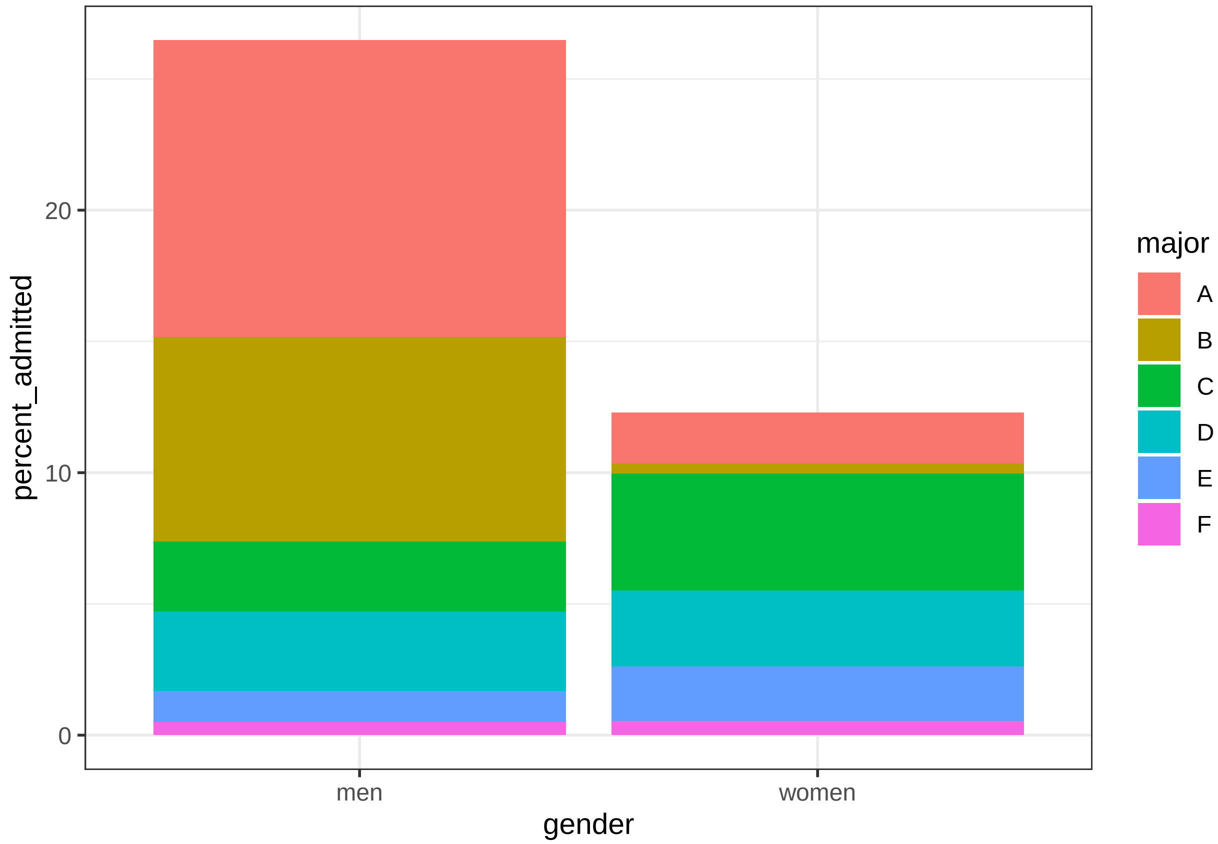
```r
# percent admissions by major
admissions %>% select(major, gender, admitted) %>%
  pivot_wider(names_from = gender, values_from = admitted) %>%
  mutate(women_minus_men = women - men)
```

```
## # A tibble: 6 x 4
##   major   men women women_minus_men
##   <chr> <dbl> <dbl>           <dbl>
## 1 A        62    82              20
## 2 B        63    68               5
## 3 C        37    34              -3
## 4 D        33    35               2
## 5 E        28    24              -4
## 6 F         6     7               1
```

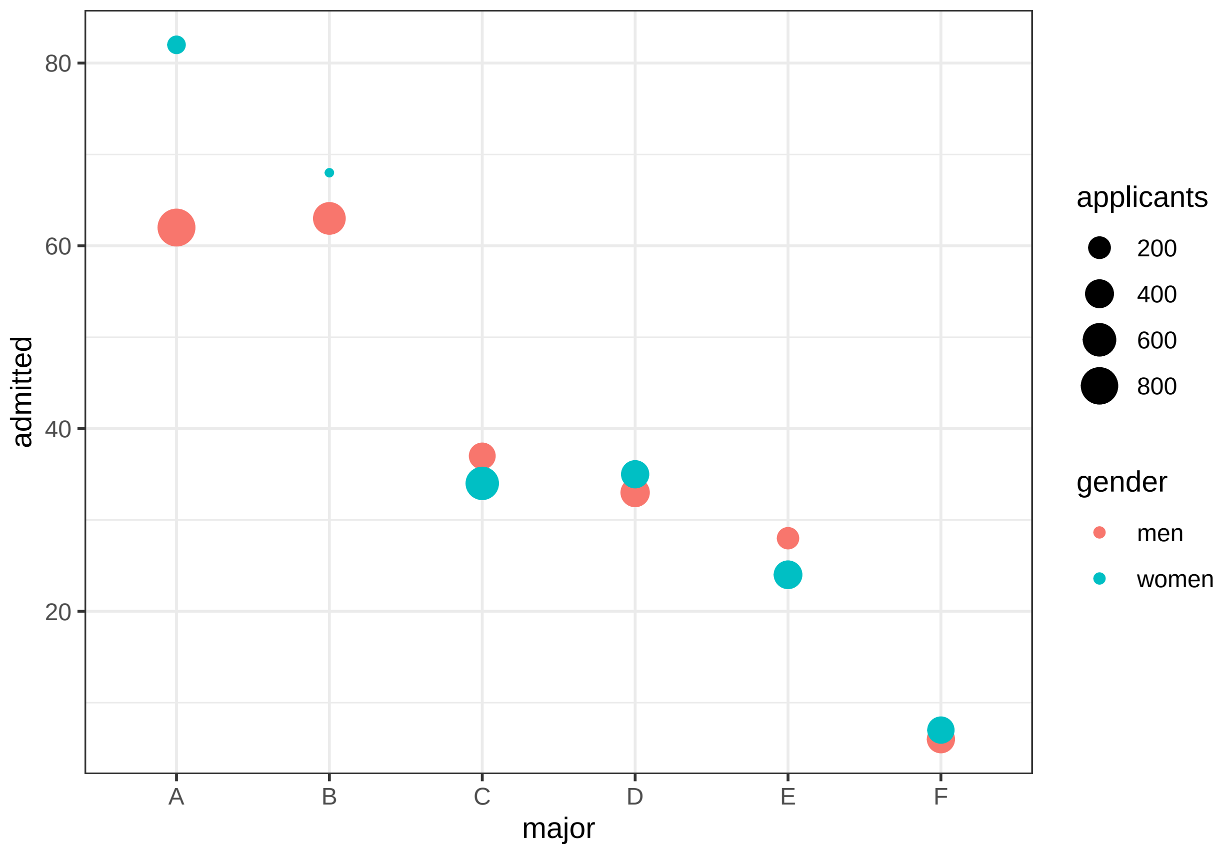
```r
# plot total percent admitted to major versus percent women applicants
admissions %>%
  group_by(major) %>%
  summarize(major_selectivity = sum(admitted * applicants) / sum(applicants),
            percent_women_applicants = sum(applicants * (gender=="women")) /
                                             sum(applicants) * 100) %>%
  ggplot(aes(major_selectivity, percent_women_applicants, label = major)) +
  geom_text()
```




```r
# plot percent of applicants accepted by gender
admissions %>%
  mutate(percent_admitted = admitted*applicants/sum(applicants)) %>%
  ggplot(aes(gender, y = percent_admitted, fill = major)) +
  geom_bar(stat = "identity", position = "stack")
```




```r
# plot admissions stratified by major
admissions %>%
  ggplot(aes(major, admitted, col = gender, size = applicants)) +
  geom_point()
```




```
# average difference by major
admissions %>%  group_by(gender) %>% summarize(average = mean(admitted))
```

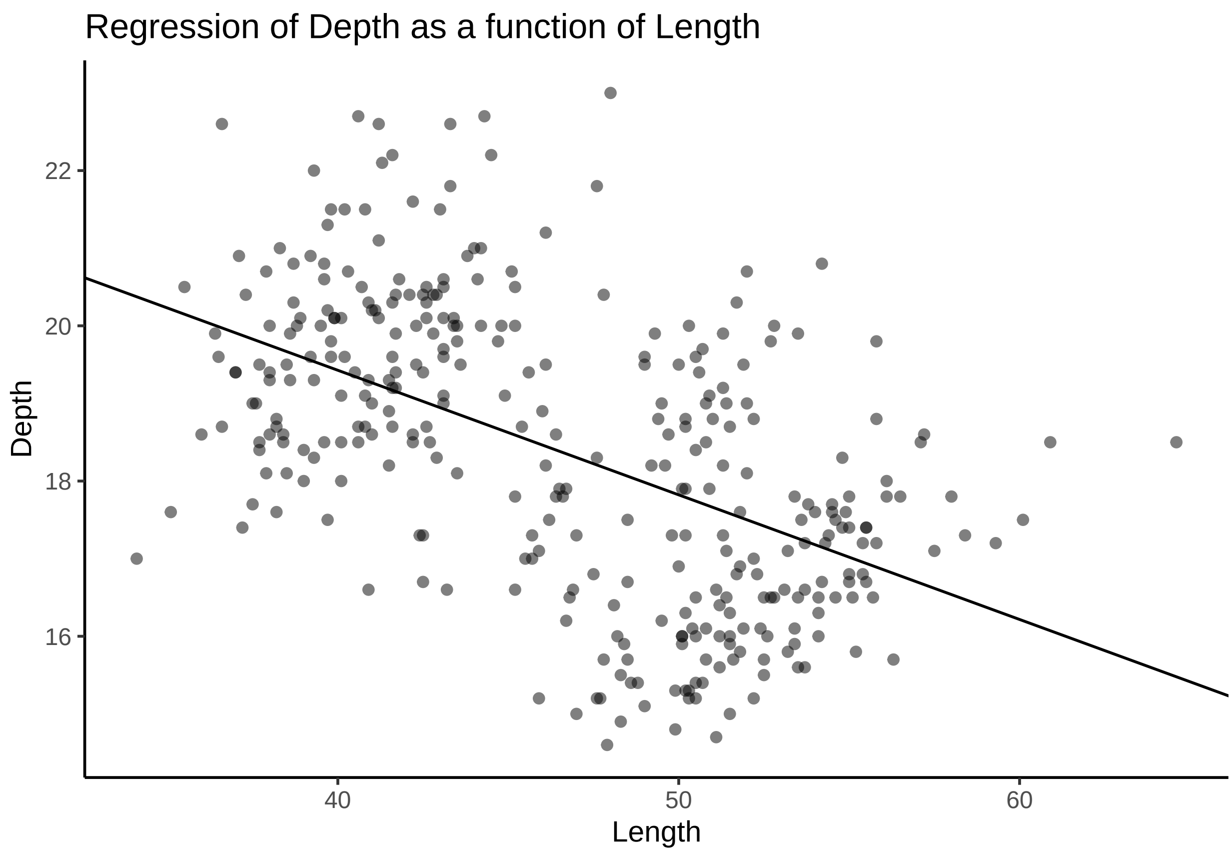
```
## # A tibble: 2 x 2
##   gender average
##   <chr>    <dbl>
## 1 men       38.2
## 2 women     41.7
```

Simpson's Paradox

Simpson's Paradox happens when we see the sign of the correlation flip when comparing the entire dataset with specific strata.

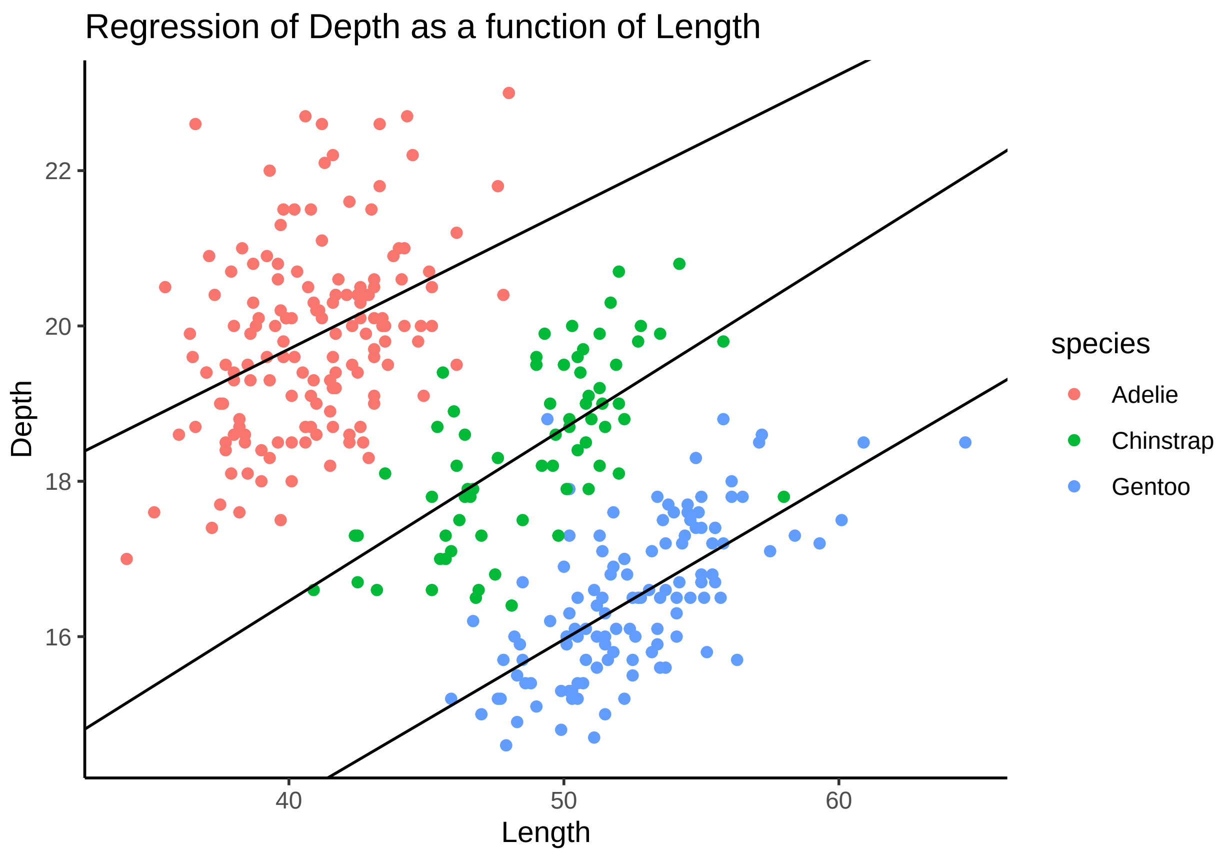
```
## Trying to recreate the plots on the explanation of the Sympson's paradox
library(tidyverse)
library(palmerpenguins)

penguin_df <- palmerpenguins::penguins %>% na.omit()
penguin_df <- penguin_df %>%
  mutate(bill_length_mm = ifelse(species == 'Gentoo', bill_length_mm + 5, bill_length_mm), bill_depth_mm
  mutate(bill_length_mm = ifelse(species == 'Adelie', bill_length_mm + 2, bill_length_mm), bill_depth_mm

chin <- penguin_df %>%
```

```
  filter(species == "Chinstrap")
adelie <- penguin_df %>%
  filter(species == "Adelie")
gentoo <- penguin_df %>%
  filter(species == "Gentoo")

lm_peng <- lm(bill_depth_mm ~ bill_length_mm, data=penguin_df)
lm_chin <- lm(bill_depth_mm ~ bill_length_mm, data=chin)
lm_adelie <- lm(bill_depth_mm ~ bill_length_mm, data=adelie)
lm_gentoo <- lm(bill_depth_mm ~ bill_length_mm, data=gentoo)

penguin_df %>%
  ggplot(aes(x = bill_length_mm, y = bill_depth_mm)) +
  geom_point(alpha = 0.5) +
  geom_abline(slope = lm_peng$coefficients[[2]], intercept = lm_peng$coefficients[[1]], color="black") +
  labs(x="Length", y="Depth", title="Regression of Depth as a function of Length") +
  theme_classic()
```




```
penguin_df %>%
  ggplot(aes(x = bill_length_mm, y = bill_depth_mm, color = species)) +
  geom_point() +
  geom_abline(slope = lm_chin$coefficients[[2]], intercept = lm_chin$coefficients[[1]], color="black") +
  geom_abline(slope = lm_adelie$coefficients[[2]], intercept = lm_adelie$coefficients[[1]], color="blac
  geom_abline(slope = lm_gentoo$coefficients[[2]], intercept = lm_gentoo$coefficients[[1]], color="blac
  labs(x="Length", y="Depth", title="Regression of Depth as a function of Length") +
```

```r
theme_classic()
```



Assessment: Confounding (Verified Learners only)

For this set of exercises, we examine the data from a 2014 PNAS paper that analyzed success rates from funding agencies in the Netherlands and concluded:

"our results reveal gender bias favoring male applicants over female applicants in the prioritization of their"quality of researcher" (but not "quality of proposal") evaluations and success rates, as well as in the language used in instructional and evaluation materials."

A response was published a few months later titled No evidence that gender contributes to personal research funding success in The Netherlands: A reaction to Van der Lee and Ellemers, which concluded:

However, the overall gender effect borders on statistical significance, despite the large sample. Moreover, their conclusion could be a prime example of Simpson's paradox; if a higher percentage of women apply for grants in more competitive scientific disciplines (i.e., with low application success rates for both men and women), then an analysis across all disciplines could incorrectly show "evidence" of gender inequality.

Who is right here: the original paper or the response? Here, you will examine the data and come to your own conclusion.

The main evidence for the conclusion of the original paper comes down to a comparison of the percentages. The information we need was originally in Table S1 in the paper, which we include in **dslabs**:

```
library(dslabs)
data("research_funding_rates")
research_funding_rates
```

```
##           discipline applications_total applications_men applications_women
## 1   Chemical sciences                122               83                 39
## 2   Physical sciences                174              135                 39
## 3             Physics                 76               67                  9
## 4          Humanities                396              230                166
## 5  Technical sciences                251              189                 62
## 6   Interdisciplinary                183              105                 78
## 7 Earth/life sciences                282              156                126
## 8     Social sciences                834              425                409
## 9    Medical sciences                505              245                260
##   awards_total awards_men awards_women success_rates_total success_rates_men
## 1           32         22           10                26.2              26.5
## 2           35         26            9                20.1              19.3
## 3           20         18            2                26.3              26.9
## 4           65         33           32                16.4              14.3
## 5           43         30           13                17.1              15.9
## 6           29         12           17                15.8              11.4
## 7           56         38           18                19.9              24.4
## 8          112         65           47                13.4              15.3
## 9           75         46           29                14.9              18.8
##   success_rates_women
## 1                25.6
## 2                23.1
## 3                22.2
## 4                19.3
## 5                21.0
## 6                21.8
## 7                14.3
## 8                11.5
## 9                11.2
```

Question 1

Construct a two-by-two table of gender (men/women) by award status (awarded/not) using the total numbers across all disciplines.

```
award_gender <- research_funding_rates %>%
  select(applications_men, applications_women, awards_men, awards_women) %>%
  summarize_all(funs(sum)) %>%
  summarize(yes_men = awards_men,
            no_men = applications_men - awards_men,
            yes_women = awards_women,
            no_women = applications_women - awards_women) %>%
  gather %>%
  separate(key, c("awarded", "gender")) %>%
  spread(gender, value)
```

```
## Warning: ‘funs()‘ was deprecated in dplyr 0.8.0.
## i Please use a list of either functions or lambdas:
##
## # Simple named list: list(mean = mean, median = median)
##
## # Auto named with ‘tibble::lst()‘: tibble::lst(mean, median)
##
## # Using lambdas list(~ mean(., trim = .2), ~ median(., na.rm = TRUE))
## Call ‘lifecycle::last_lifecycle_warnings()‘ to see where this warning was
## generated.
```

```
award_gender %>% filter(awarded == 'no') %>% pull(men)
```

What is the number of men not awarded?

```
## [1] 1345
```

```
award_gender %>% filter(awarded == 'no') %>% pull(women)
```

What is the number of women not awarded?

```
## [1] 1011
```

Question 2

Use the two-by-two table from Question 1 to compute the percentages of men awarded versus women awarded.

```
award_gender_per <- award_gender %>%
  mutate(men = round(men/sum(men)*100, 1), women = round(women/sum(women)*100, 1))
```

```
award_gender_per %>% filter(awarded == 'yes') %>% pull(men)
```

What is the percentage of men awarded

```
## [1] 17.7
```

```
award_gender_per %>% filter(awarded == 'yes') %>% pull(women)
```

What is the percentage of men awarded

```
## [1] 14.9
```

Question 3

Run a chi-squared test on the two-by-two table to determine whether the difference in the two funding awarded rates is significant. (You can use tidy() to turn the output of chisq.test() into a data frame as well.) What is the p-value of the difference in funding awarded rate?

```
award_gender %>% select(-awarded) %>% chisq.test() %>% tidy()
```

```
## # A tibble: 1 x 4
##   statistic p.value parameter method
##       <dbl>   <dbl>     <int> <chr>
## 1      3.81  0.0509         1 Pearson's Chi-squared test with Yates' continuity~
```

Question 4

There may be an association between gender and funding. But can we infer causation here? Is gender bias causing this observed difference? The response to the original paper claims that what we see here is similar to the UC Berkeley admissions example. Specifically they state that this "could be a prime example of Simpson's paradox; if a higher percentage of women apply for grants in more competitive scientific disciplines, then an analysis across all disciplines could incorrectly show 'evidence' of gender inequality."
To settle this dispute, use this dataset with number of applications, awards, and success rate for each gender:

```
dat <- research_funding_rates %>%
      mutate(discipline = reorder(discipline, success_rates_total)) %>%
      rename(success_total = success_rates_total,
             success_men = success_rates_men,
             success_women = success_rates_women) %>%
      pivot_longer(-discipline) %>%
      separate(name, c("type", "gender")) %>%
      pivot_wider(names_from = type, values_from = value) %>%
      filter(gender != "total")
dat
```

```
## # A tibble: 18 x 5
##    discipline          gender applications awards success
##    <fct>               <chr>         <dbl>  <dbl>   <dbl>
##  1 Chemical sciences   men              83     22    26.5
##  2 Chemical sciences   women            39     10    25.6
##  3 Physical sciences   men             135     26    19.3
##  4 Physical sciences   women            39      9    23.1
##  5 Physics             men              67     18    26.9
##  6 Physics             women             9      2    22.2
##  7 Humanities          men             230     33    14.3
##  8 Humanities          women           166     32    19.3
##  9 Technical sciences  men             189     30    15.9
## 10 Technical sciences  women            62     13    21
## 11 Interdisciplinary   men             105     12    11.4
## 12 Interdisciplinary   women            78     17    21.8
## 13 Earth/life sciences men             156     38    24.4
## 14 Earth/life sciences women           126     18    14.3
## 15 Social sciences     men             425     65    15.3
## 16 Social sciences     women           409     47    11.5
```

```
## 17 Medical sciences    men              245      46     18.8
## 18 Medical sciences    women            260      29     11.2
```

To check if this is a case of Simpson's paradox, plot the success rates versus disciplines, which have been ordered by overall success, with colors to denote the genders and size to denote the number of applications.

```
dat %>% group_by(gender) %>%
  ggplot(aes(discipline, success, color = gender, size = applications)) +
  theme(axis.text.x = element_text(angle = 90, hjust = 1)) +
  geom_point()
```