Template

Oleg Sechovcov

2025-01-04

R Shortcuts

- `Ctrl + Enter` - Run current line or selection
- `Ctrl + Shift + Enter` - Run current chunk
- `Ctrl + Alt + I` - Insert chunk
- `Ctrl + Alt + C` - Comment/Uncomment line or selection
- `Ctrl + Shift + M` - Insert pipe operator `%>%`

R Libraries

```
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.4     v readr     2.1.5
## v forcats   1.0.0     v stringr   1.5.1
## v ggplot2   3.5.1     v tibble    3.2.1
## v lubridate 1.9.3     v tidyr     1.3.1
## 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 errors
```

```
library(ggtext) # for text on plot
```

```
## Warning: pakke 'ggtext' blev bygget under R version 4.4.2
```

```
library(forcats) # for fct_reorder
library(png) # for image as background
library(grid)# for image as background
library(jpeg)
```

R Data

R Data insert

```
#data <- read.csv("data.csv") # read data from csv
```

Read data

```
head(iris, 5) # show first 5 rows
```

```
##   Sepal.Length Sepal.Width Petal.Length Petal.Width Species
## 1          5.1         3.5          1.4         0.2  setosa
## 2          4.9         3.0          1.4         0.2  setosa
## 3          4.7         3.2          1.3         0.2  setosa
## 4          4.6         3.1          1.5         0.2  setosa
## 5          5.0         3.6          1.4         0.2  setosa
```

```
nrow(iris) # number of rows
```

```
## [1] 150
```

```
summary(iris) # summary of data
```

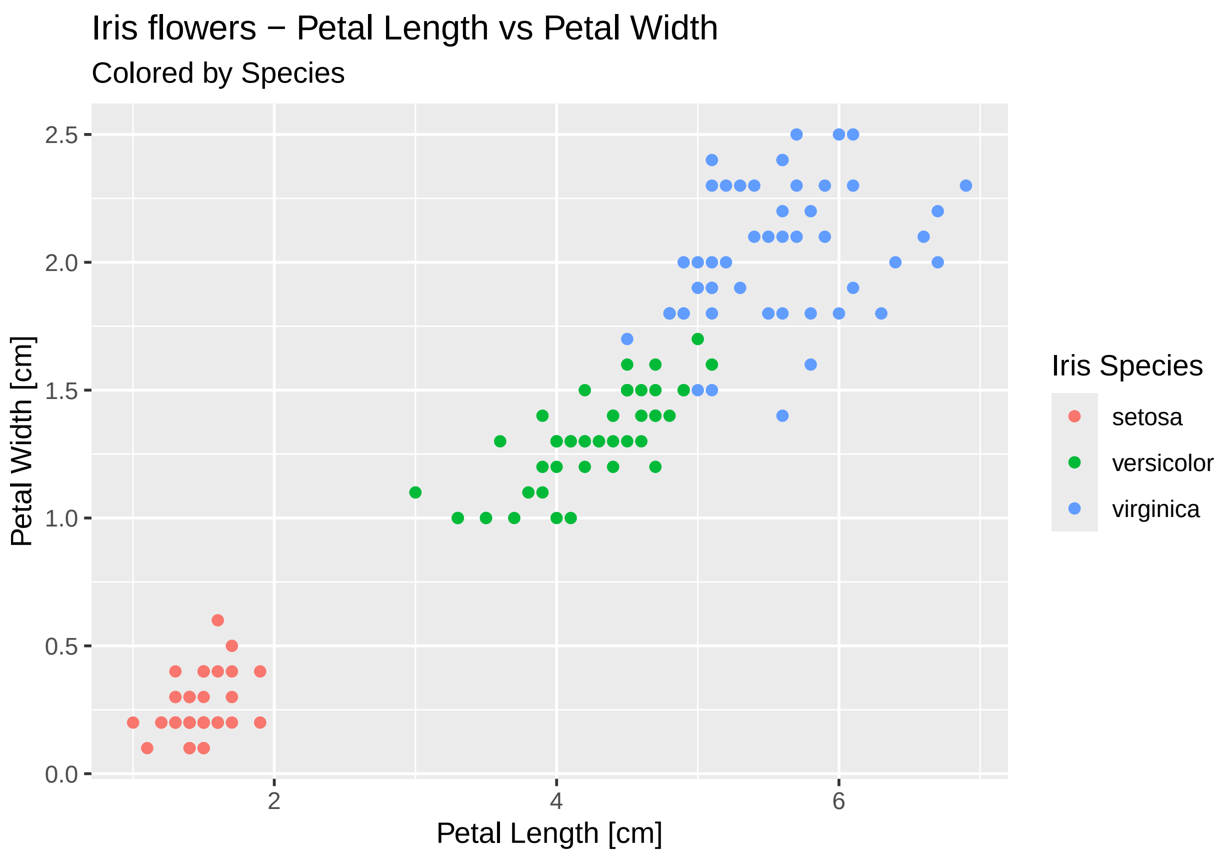
```
##   Sepal.Length    Sepal.Width     Petal.Length    Petal.Width   
##  Min.   :4.300   Min.   :2.000   Min.   :1.000   Min.   :0.100  
##  1st Qu.:5.100   1st Qu.:2.800   1st Qu.:1.600   1st Qu.:0.300  
##  Median :5.800   Median :3.000   Median :4.350   Median :1.300  
##  Mean   :5.843   Mean   :3.057   Mean   :3.758   Mean   :1.199  
##  3rd Qu.:6.400   3rd Qu.:3.300   3rd Qu.:5.100   3rd Qu.:1.800  
##  Max.   :7.900   Max.   :4.400   Max.   :6.900   Max.   :2.500  
##        Species  
##  setosa    :50  
##  versicolor:50  
##  virginica :50  
##                 
##                 
##                 
```

Plots

Scatter plot

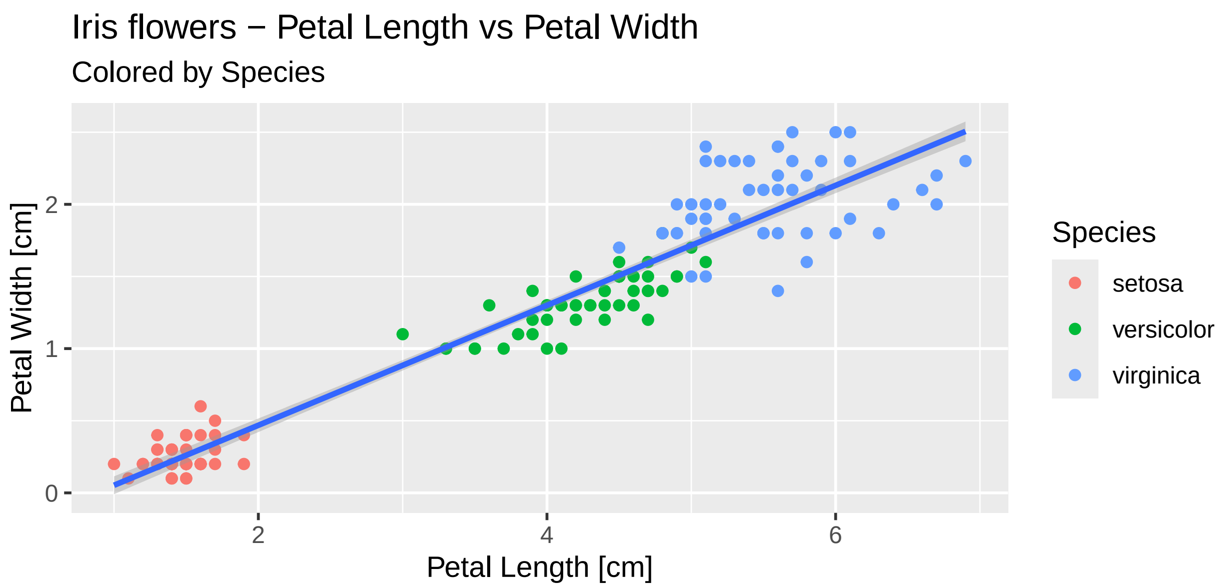
```
ggplot(data = iris) +
  geom_point(mapping = aes(x = Petal.Length, y = Petal.Width, color = Species)) +
  labs(x = 'Petal Length [cm]',
```

```r
       y = 'Petal Width [cm]',
       color = 'Iris Species',
       title = 'Iris flowers - Petal Length vs Petal Width',
       subtitle = 'Colored by Species')
```



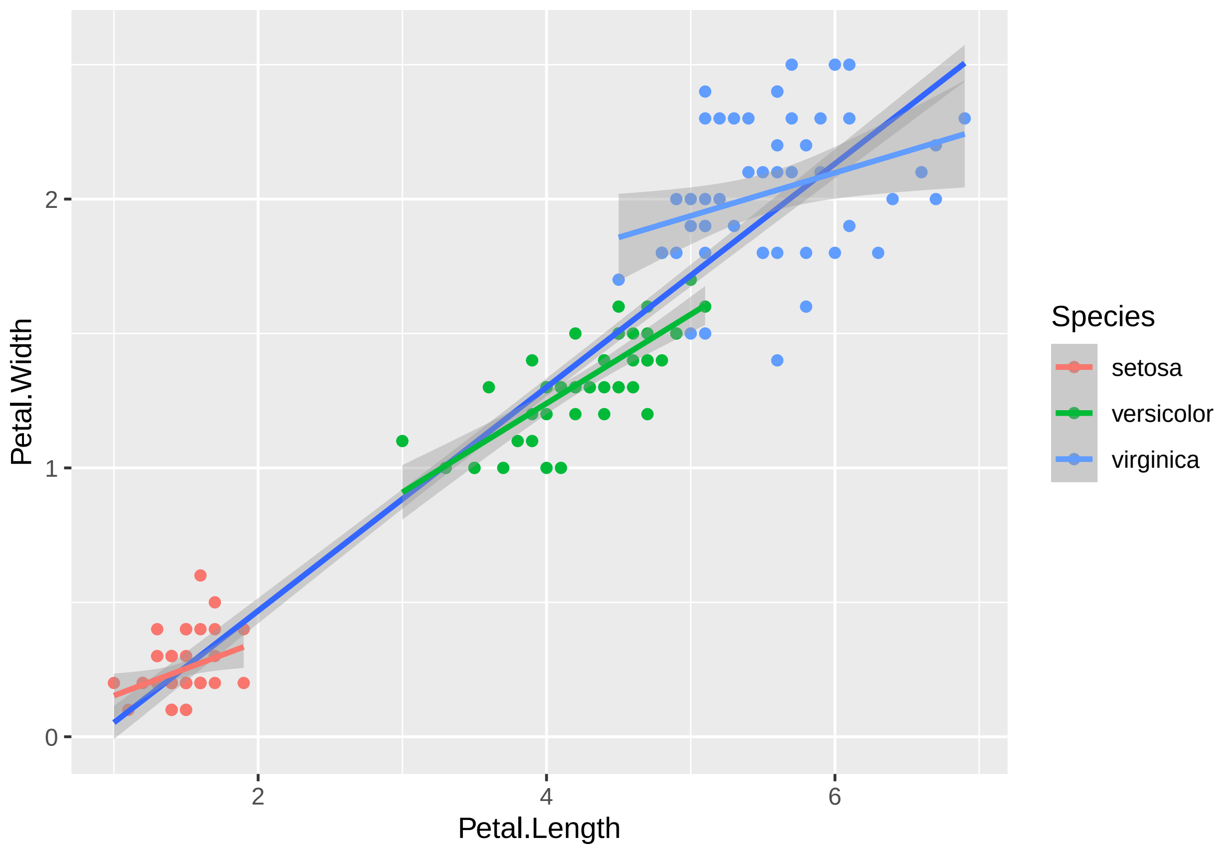

```r
ggplot(data = iris) +
  geom_point(mapping = aes(x = Petal.Length, y = Petal.Width,
                           #shape = Species,
                           #size = Species
                           color = Species)) +
  labs(x = 'Petal Length [cm]',
       y = 'Petal Width [cm]',
       size = 'Iris Species',
       title = 'Iris flowers - Petal Length vs Petal Width',
       subtitle = 'Colored by Species') +
  coord_fixed(ratio = 1) +
  geom_smooth(mapping = aes(x = Petal.Length, y = Petal.Width), method = lm)
```

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



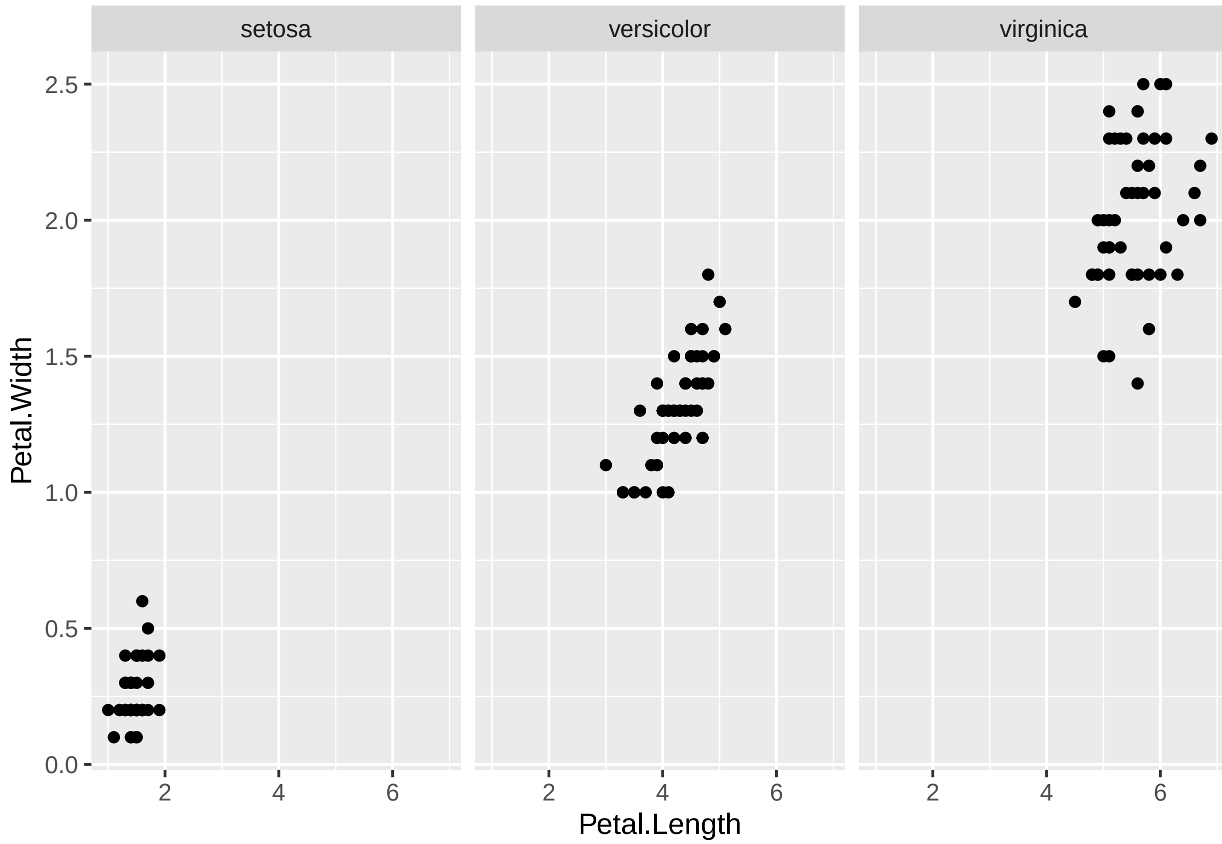

```
ggplot(data = iris) +
  geom_point(mapping = aes(x = Petal.Length, y = Petal.Width,
                           color = Species)) +
  geom_smooth(mapping = aes(x = Petal.Length, y = Petal.Width), method = lm)+
  geom_smooth(mapping = aes(x = Petal.Length, y = Petal.Width, color = Species), method = lm)
```

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

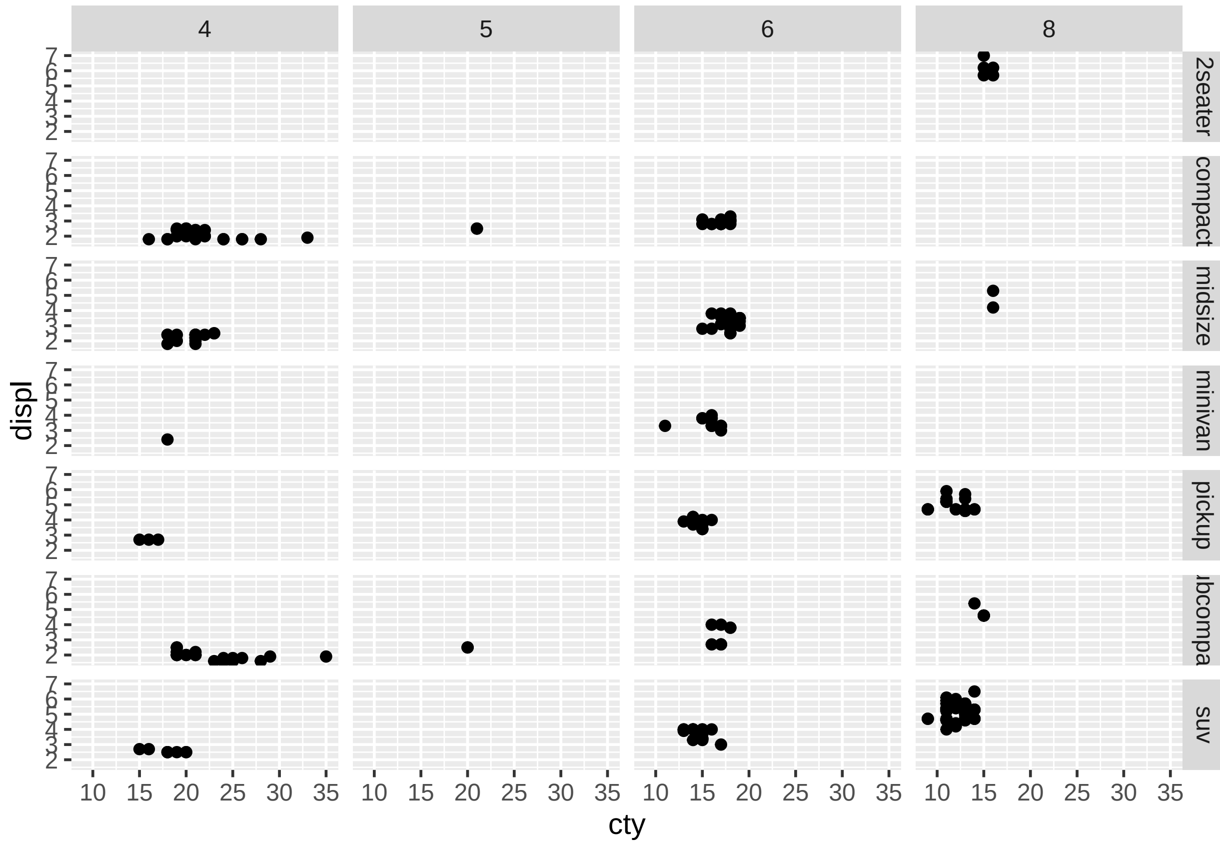


Facet wrap and facet grid

```r
ggplot(data = iris) +
  geom_point(mapping = aes(x = Petal.Length, y = Petal.Width)) +
  facet_wrap(~Species)
```

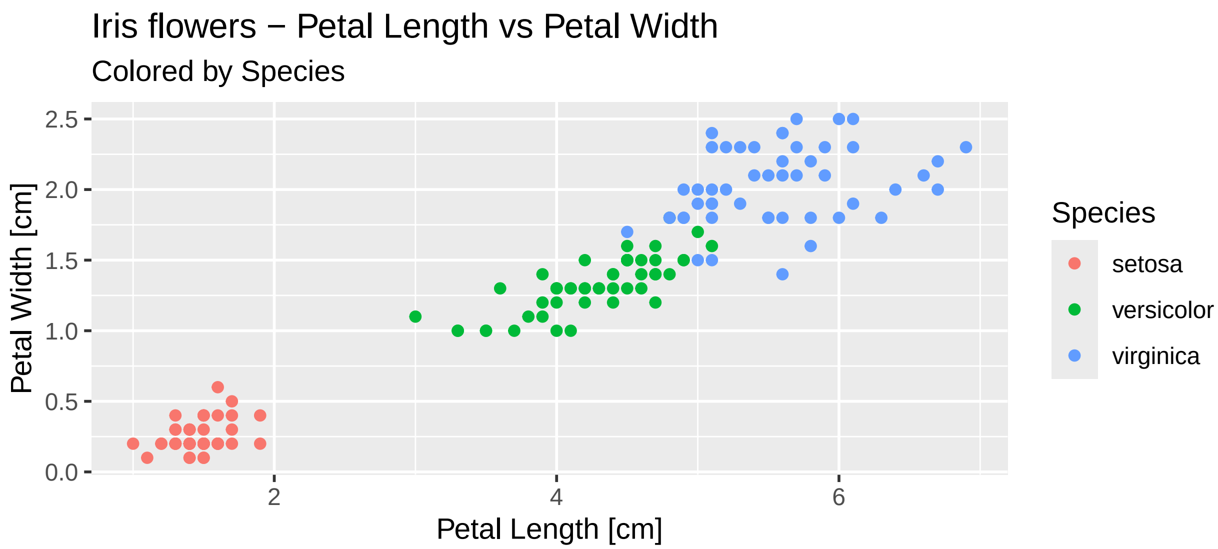



```
ggplot(data = mpg) +
  geom_point(mapping = aes(x = cty, y = displ)) +
  facet_grid(class~factor(cyl))
```



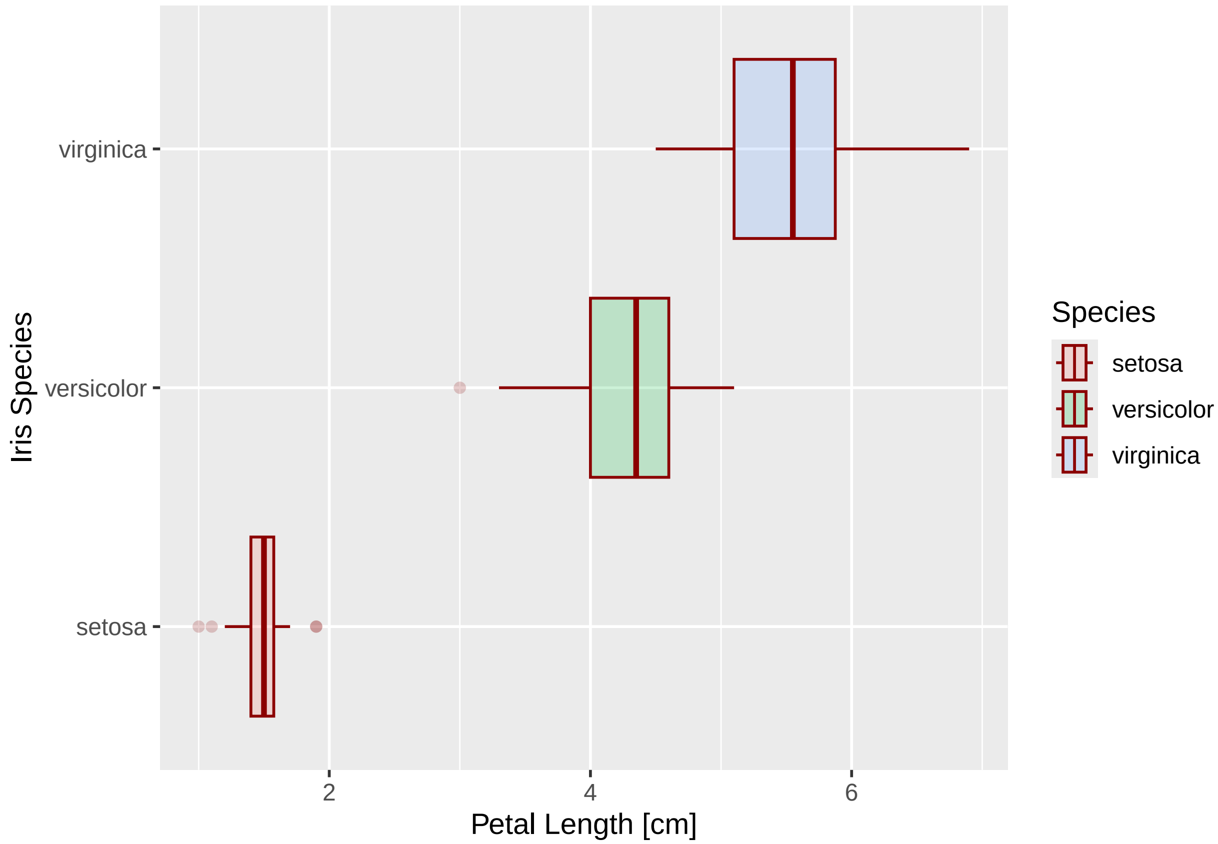
Naming

```
ggplot(iris) +
  geom_point(aes(Petal.Length, Petal.Width,
                             #shape = Species,
                             #size = Species
                             color = Species)) +
  labs(x = 'Petal Length [cm]',
       y = 'Petal Width [cm]',
       size = 'Iris Species',
       title = 'Iris flowers - Petal Length vs Petal Width',
       subtitle = 'Colored by Species') +
  coord_fixed(ratio = 1)
```

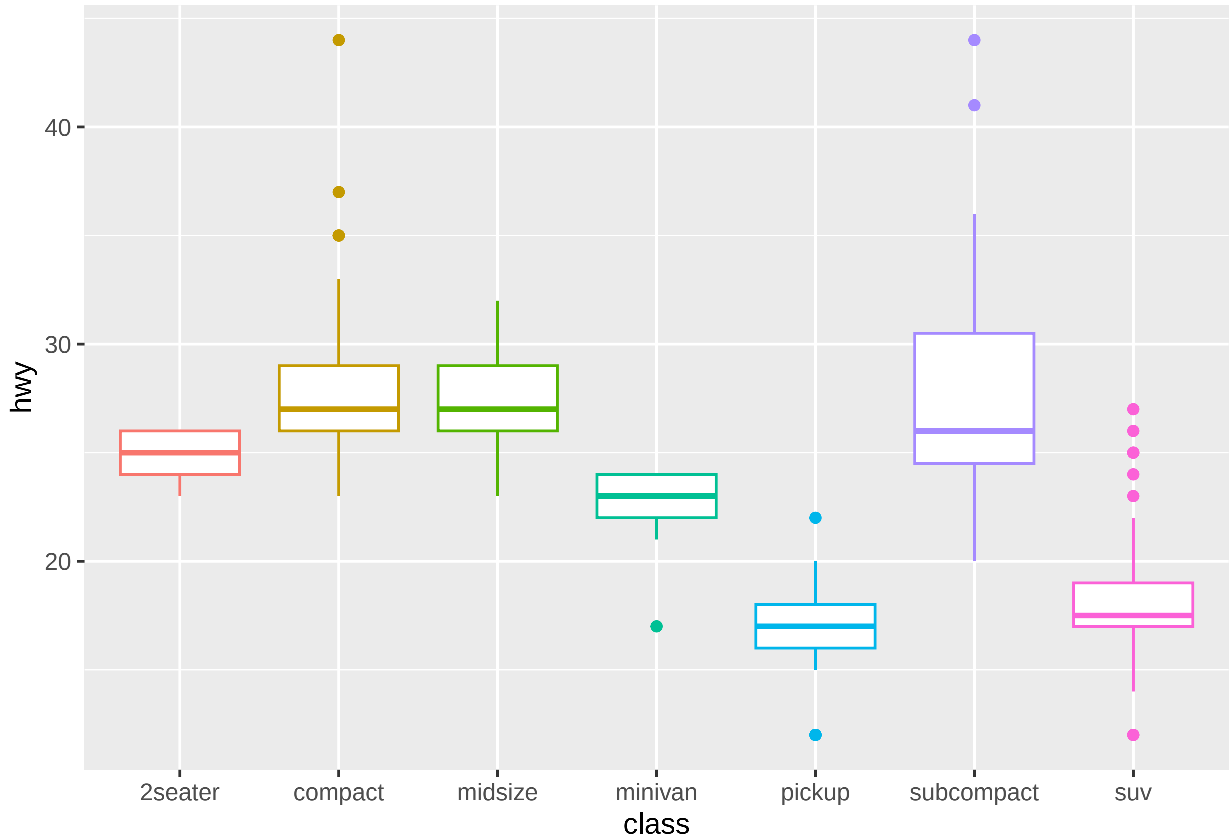


Box Plots

```
ggplot(iris) +
  geom_boxplot(mapping = aes(y = Species,
                             x = Petal.Length,
                             fill = Species),
               alpha = 0.2,
               color = 'darkred') +
  labs(x = 'Petal Length [cm]',
       y = 'Iris Species')
```

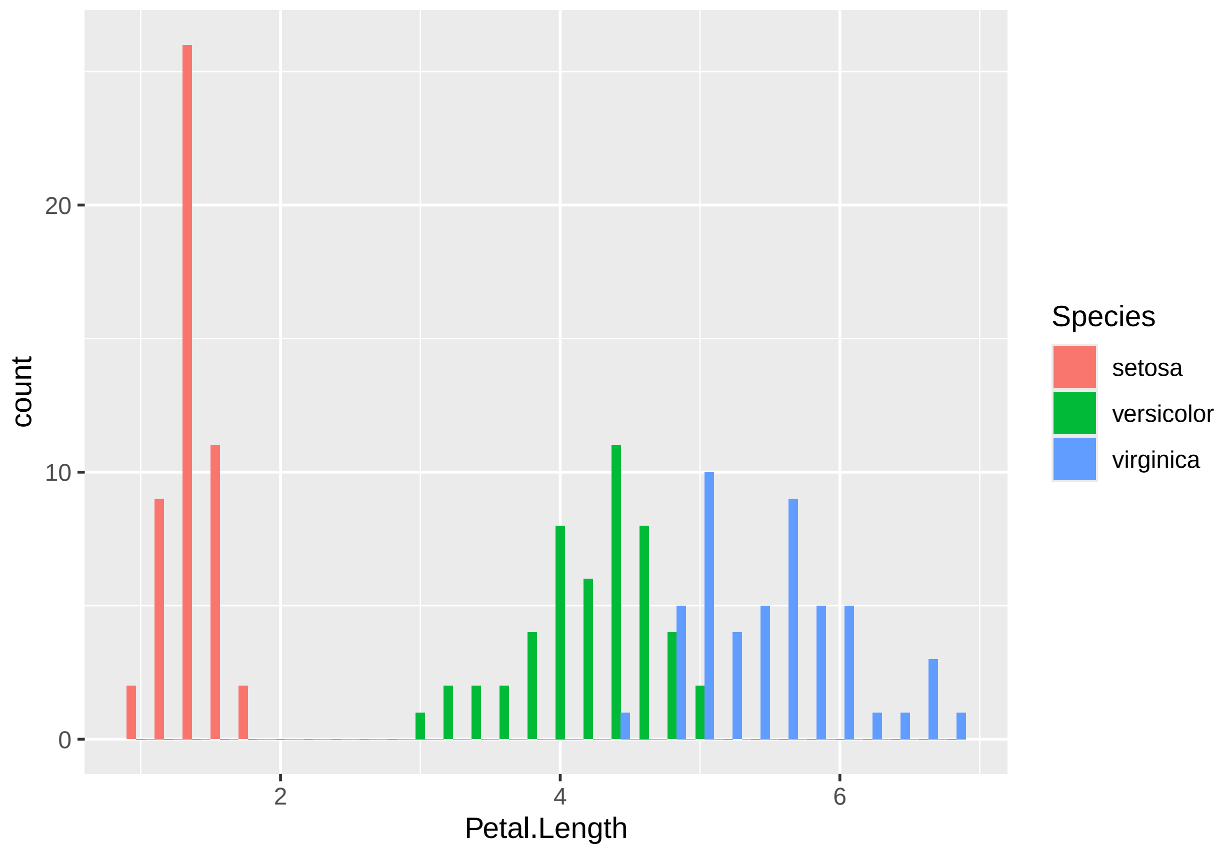



```
ggplot(mpg) +
  geom_boxplot(aes(x = class, y = hwy, color = class), show.legend = FALSE)
```



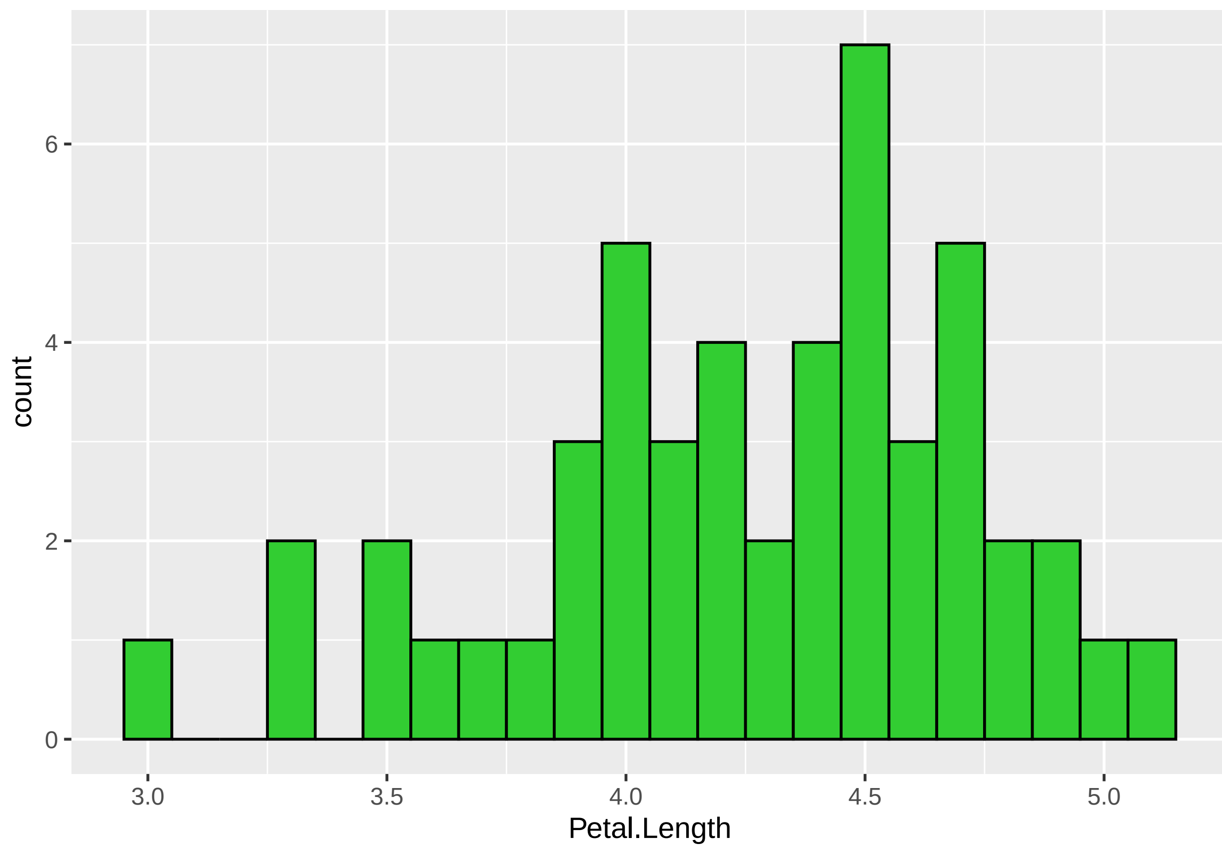
Histogram and Density Plot

```
ggplot(iris) +
  geom_histogram(aes(x = Petal.Length, fill = Species), binwidth = 0.2, position = 'dodge')
```

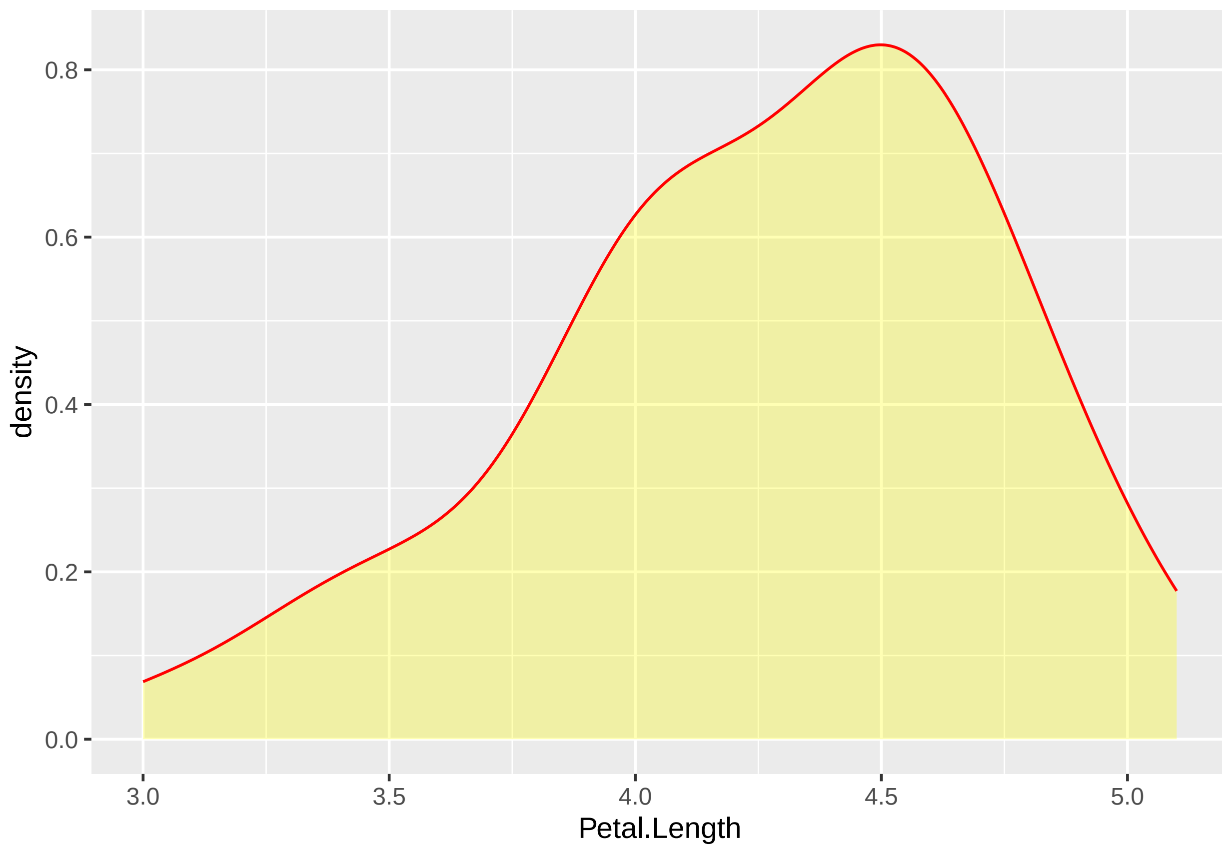



```r
versicolor <- filter(iris, Species == 'versicolor')
```

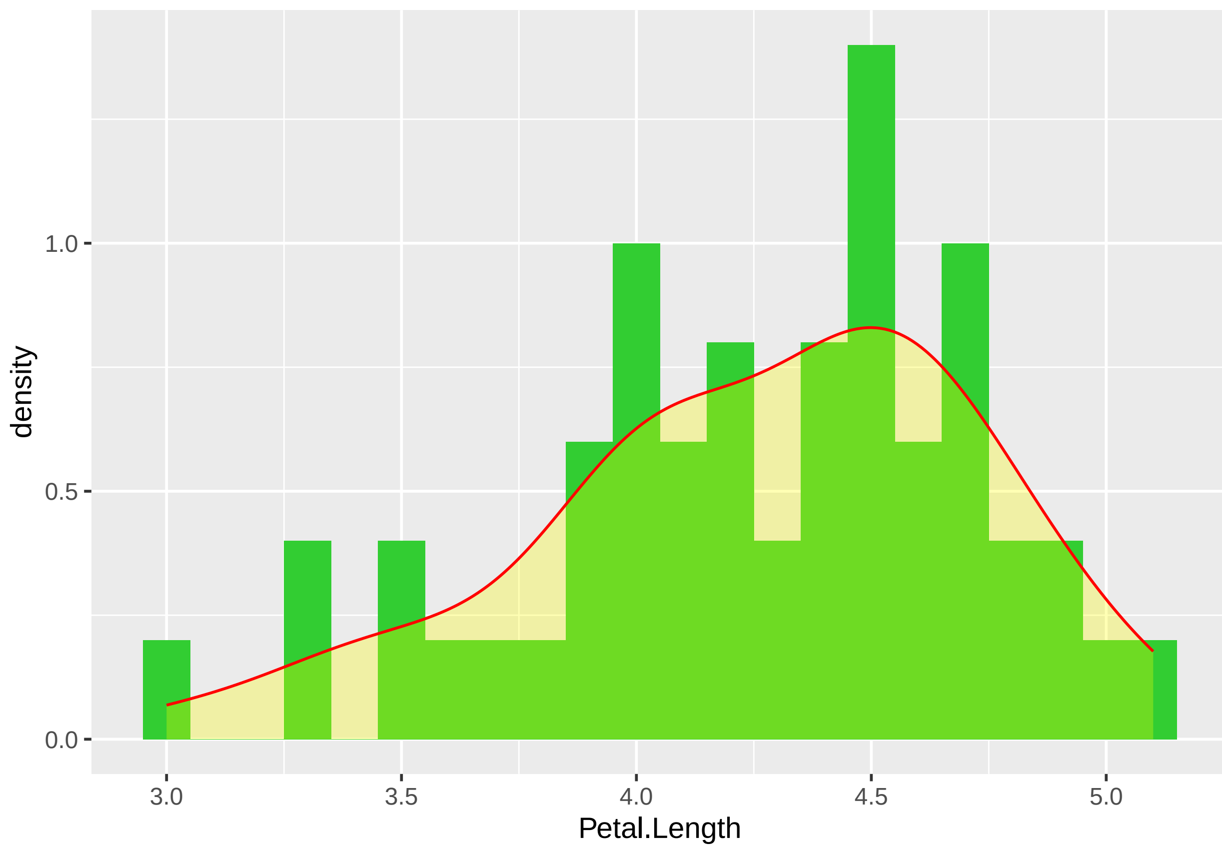
```r
ggplot(versicolor) +
  geom_histogram(aes(x = Petal.Length), color = 'black', fill ='limegreen', binwidth = 0.1)
```



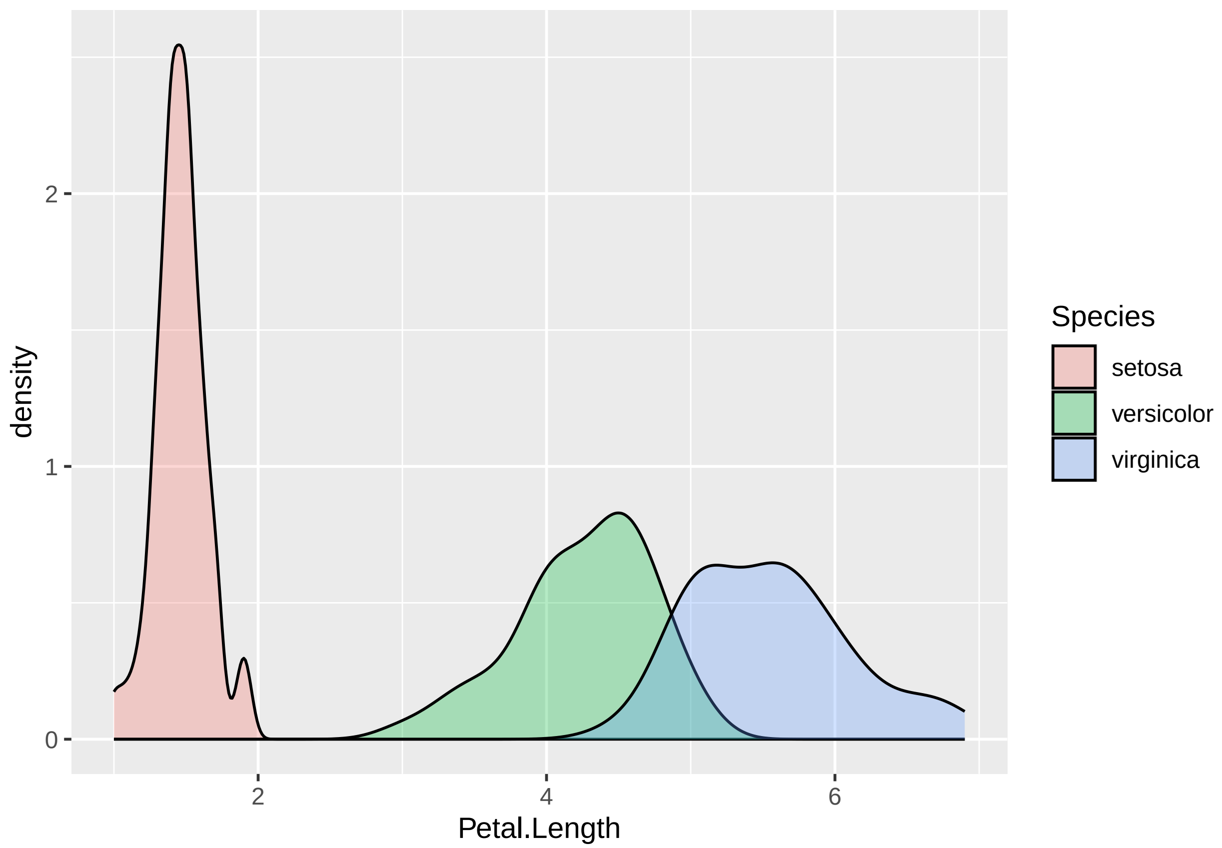

```
ggplot(versicolor) +
  geom_density(aes(x = Petal.Length), fill = 'yellow',
               alpha = 0.3,
               color = 'red')
```




```r
ggplot(versicolor) +
  geom_histogram(aes(x = Petal.Length, y = after_stat(density)),
                 fill = 'limegreen', binwidth = 0.1)+
  geom_density(aes(x = Petal.Length), color = 'red', fill = 'yellow', alpha = 0.3)
```

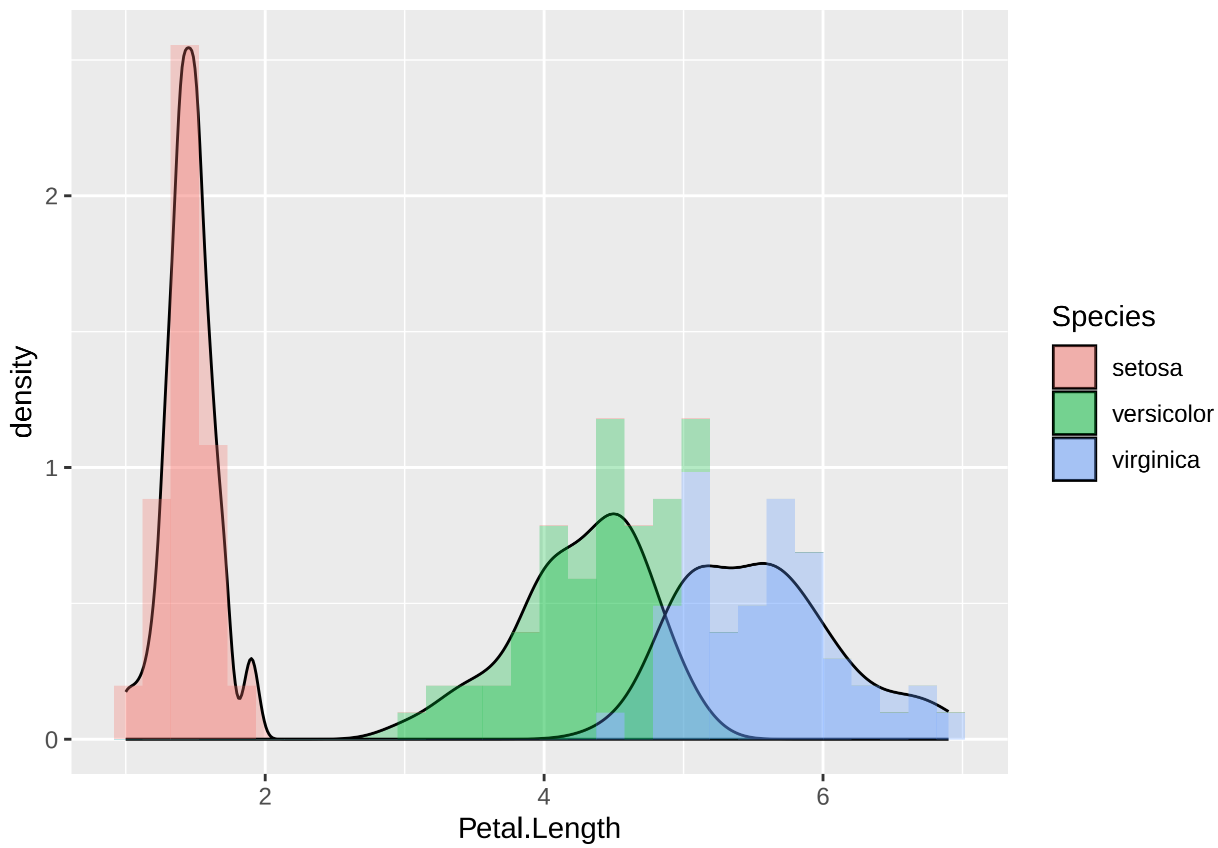



```r
ggplot(iris) +
  geom_density(aes(x = Petal.Length, fill = Species), alpha = 0.3)
```



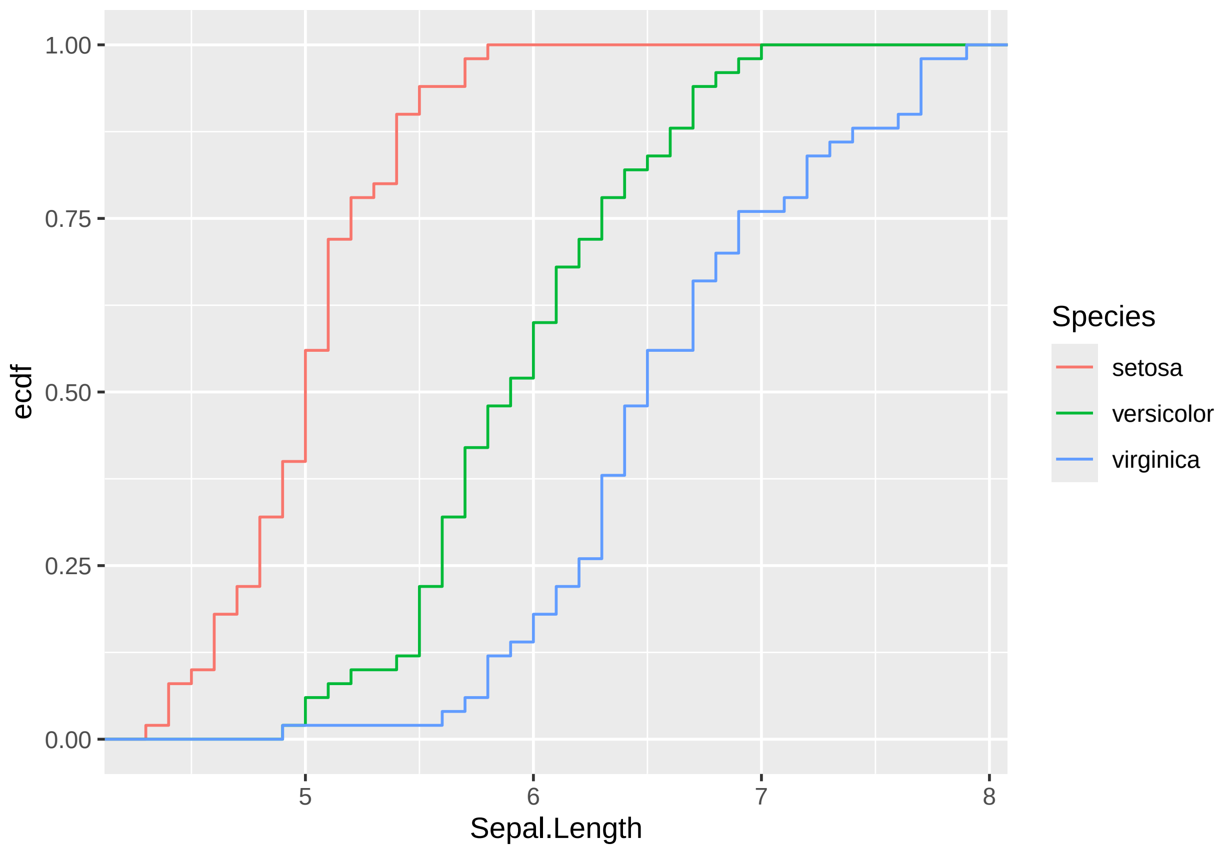

```r
ggplot(iris) +
  geom_density(aes(x = Petal.Length, fill = Species), alpha = 0.3) +
  geom_histogram(aes(x = Petal.Length, y = after_stat(density), fill = Species),
                 alpha = 0.3)
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



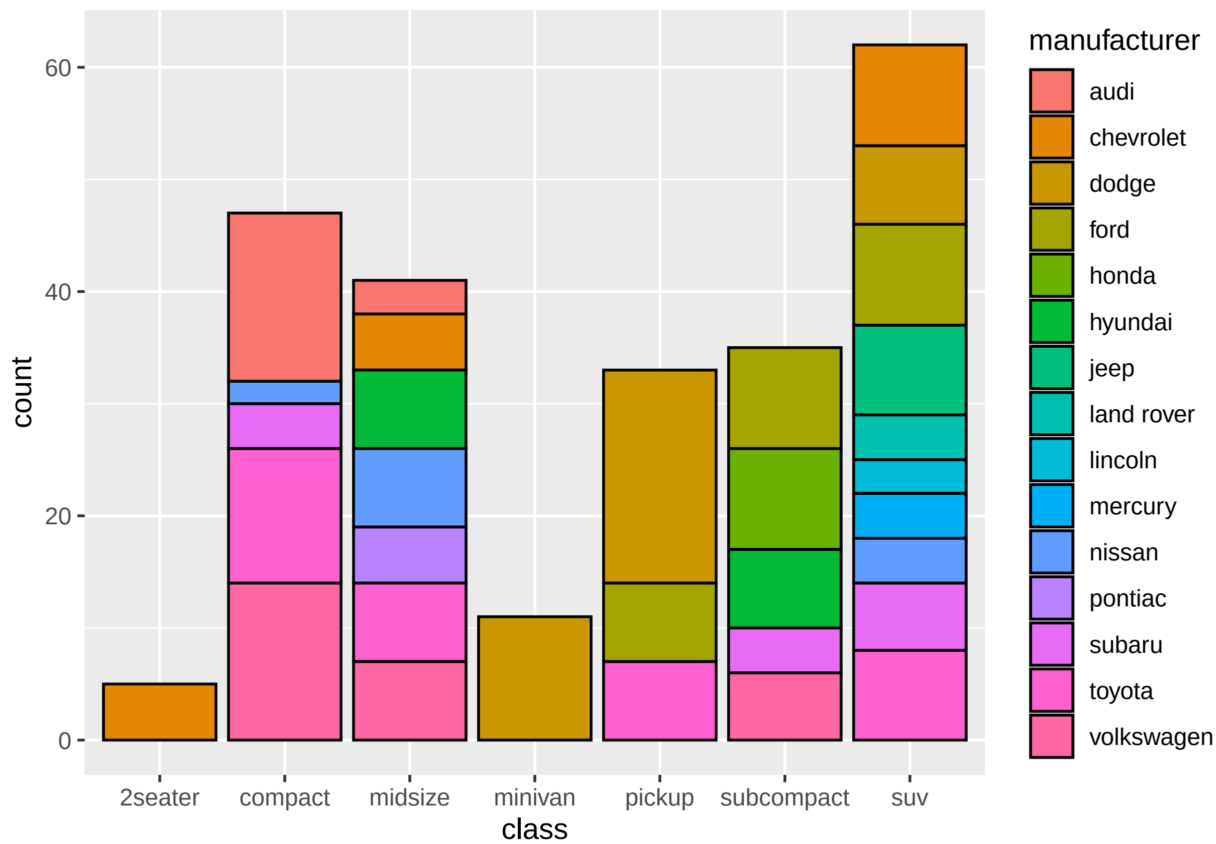
Empirical cumulative distribution function - stat_ecdf

```
ggplot(iris) +
  stat_ecdf(mapping = aes(x = Sepal.Length, color = Species))
```



Barplots

```
ggplot(mpg) +
  geom_bar(mapping = aes(x = class, fill = manufacturer), color = 'black')
```



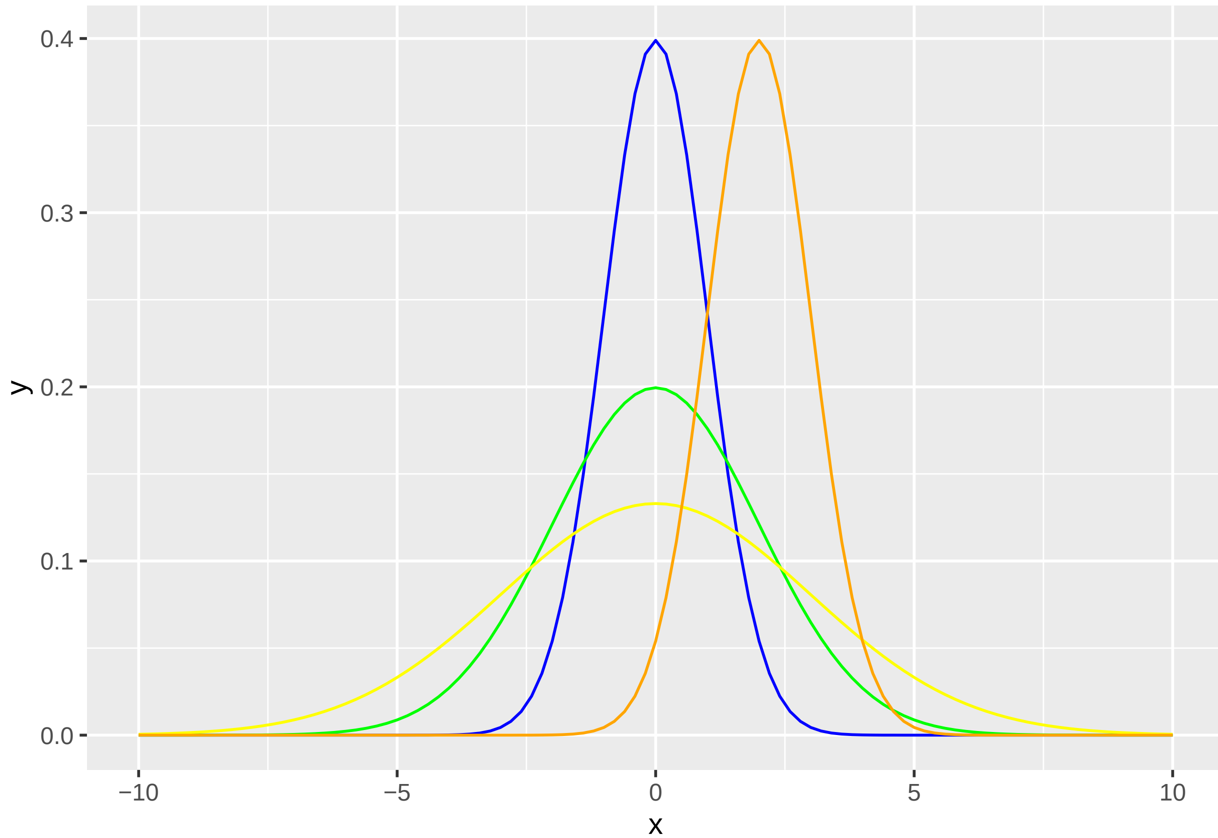
inference

Normal distribution

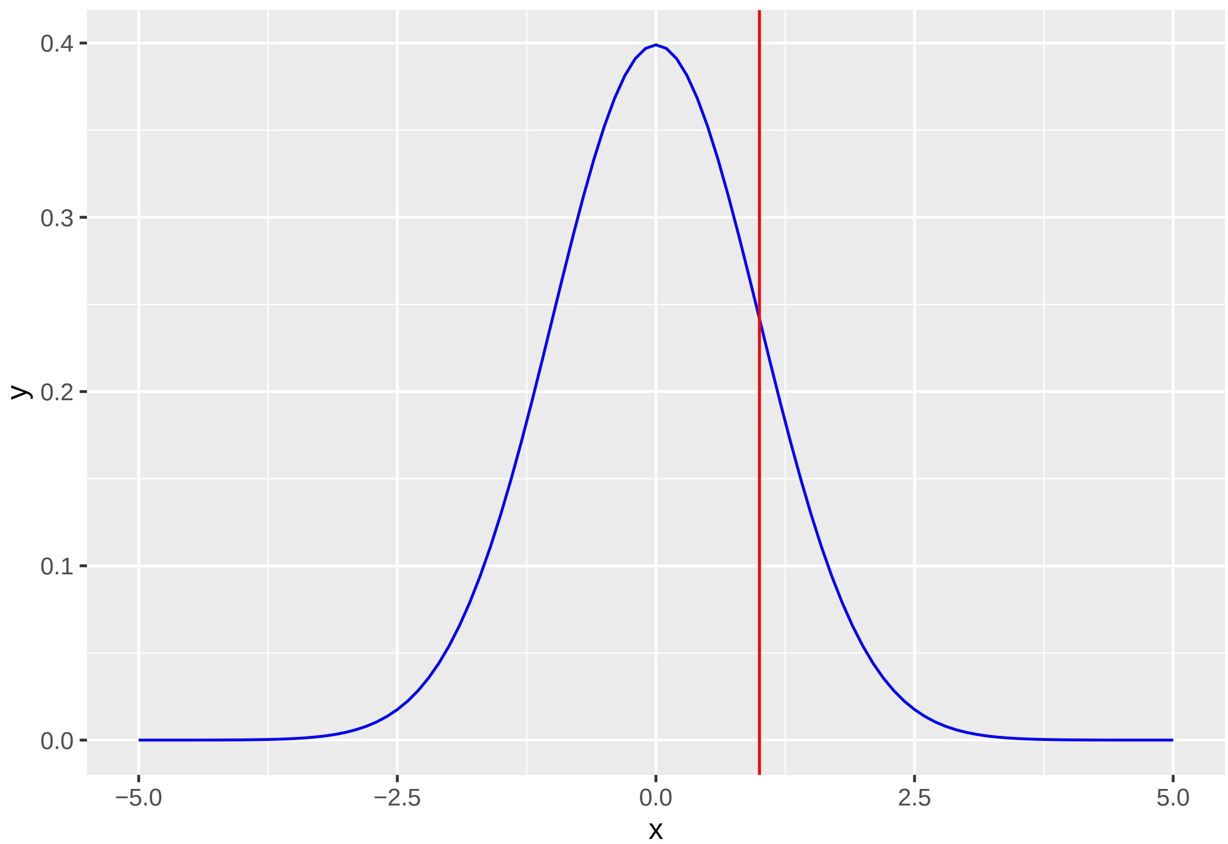
dnorm(x, mean, sd) - distribution function pnorm(x, mean, sd) - probability, qnorm(probability, mean, sd) - given probability what is the value rnorm(how_many, mean, sd) - generate data

default values: mean = 0, sd = 1

```
ggplot(data.frame(x = seq(-10, 10, length = 100)), aes(x = x))+
  stat_function(fun = dnorm, args = list(mean = 0, sd = 1), color = 'blue') +
  stat_function(fun = dnorm, args = list(mean = 0, sd = 2), color = 'green') +
  stat_function(fun = dnorm, args = list(mean = 0, sd = 3), color = 'yellow') +
  stat_function(fun = dnorm, args = list(mean = 2, sd = 1), color = 'orange')
```

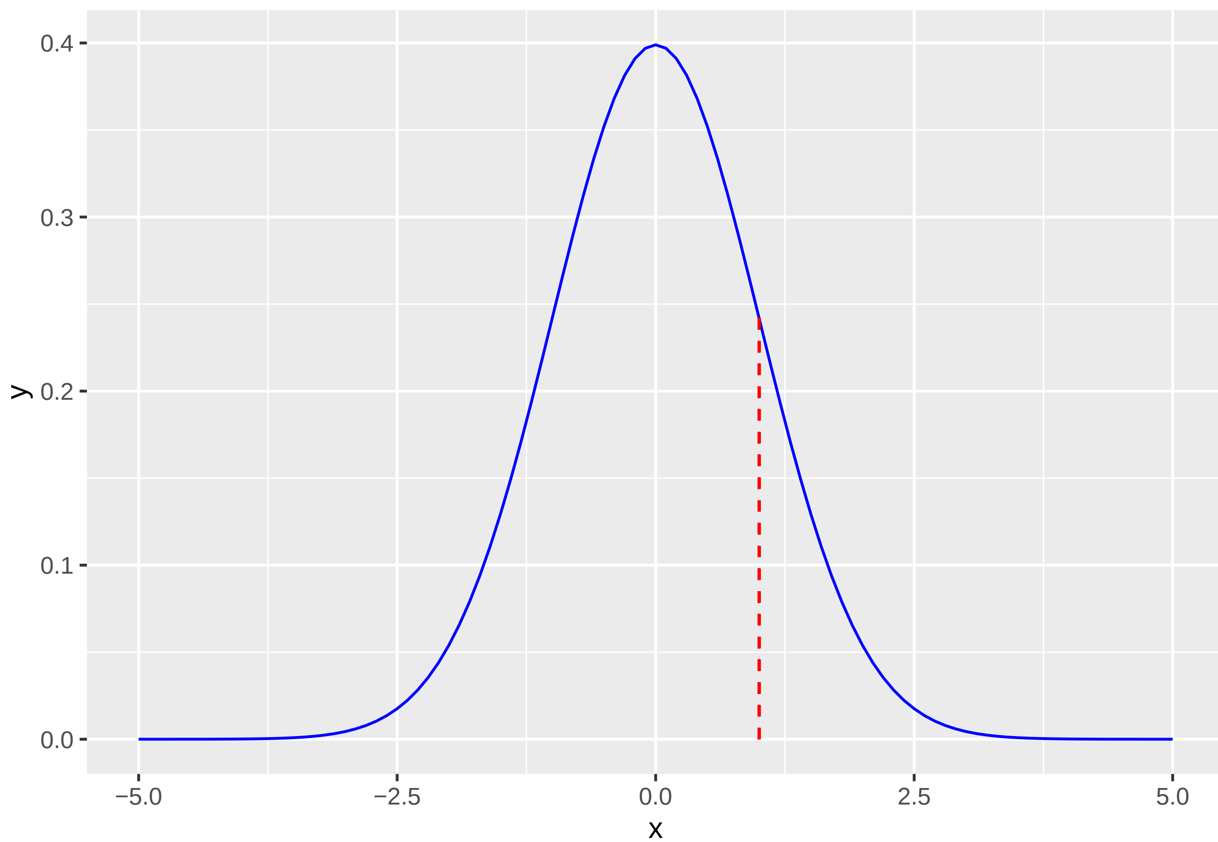



```
ggplot(data.frame(x = seq(-5, 5, length = 100)), aes(x = x))+
  stat_function(fun = dnorm, args = list(mean = 0, sd = 1), color = 'blue') +
  geom_vline(xintercept = 1, color ='red')
```




```
ggplot(data.frame(x = seq(-5, 5, length = 100)), aes(x = x))+
  stat_function(fun = dnorm, args = list(mean = 0, sd = 1), color = 'blue') +
  geom_segment(aes(x = 1, y = 0,
               xend = 1, yend = dnorm(1)), color ='red', linetype = 'dashed')
```

```
## Warning in geom_segment(aes(x = 1, y = 0, xend = 1, yend = dnorm(1)), color = "red", : All aesthetics
## i Please consider using ‘annotate()‘ or provide this layer with data containing
##   a single row.
```



ECDF

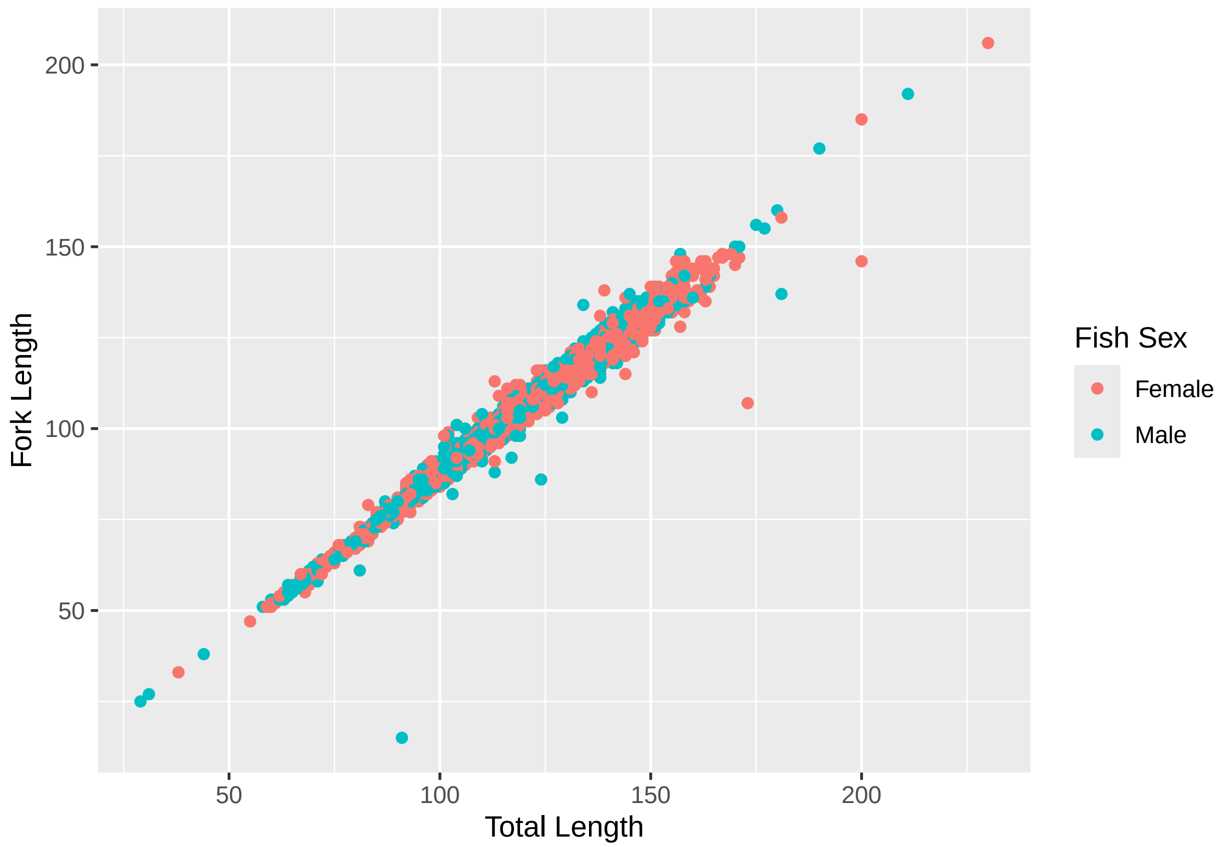
```r
sharks <- readr::read_csv('sharks.csv')
```

```
## Rows: 2510 Columns: 4
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (1): Fish Sex
## dbl (3): Calendar Year, Total Length, Fork Length
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

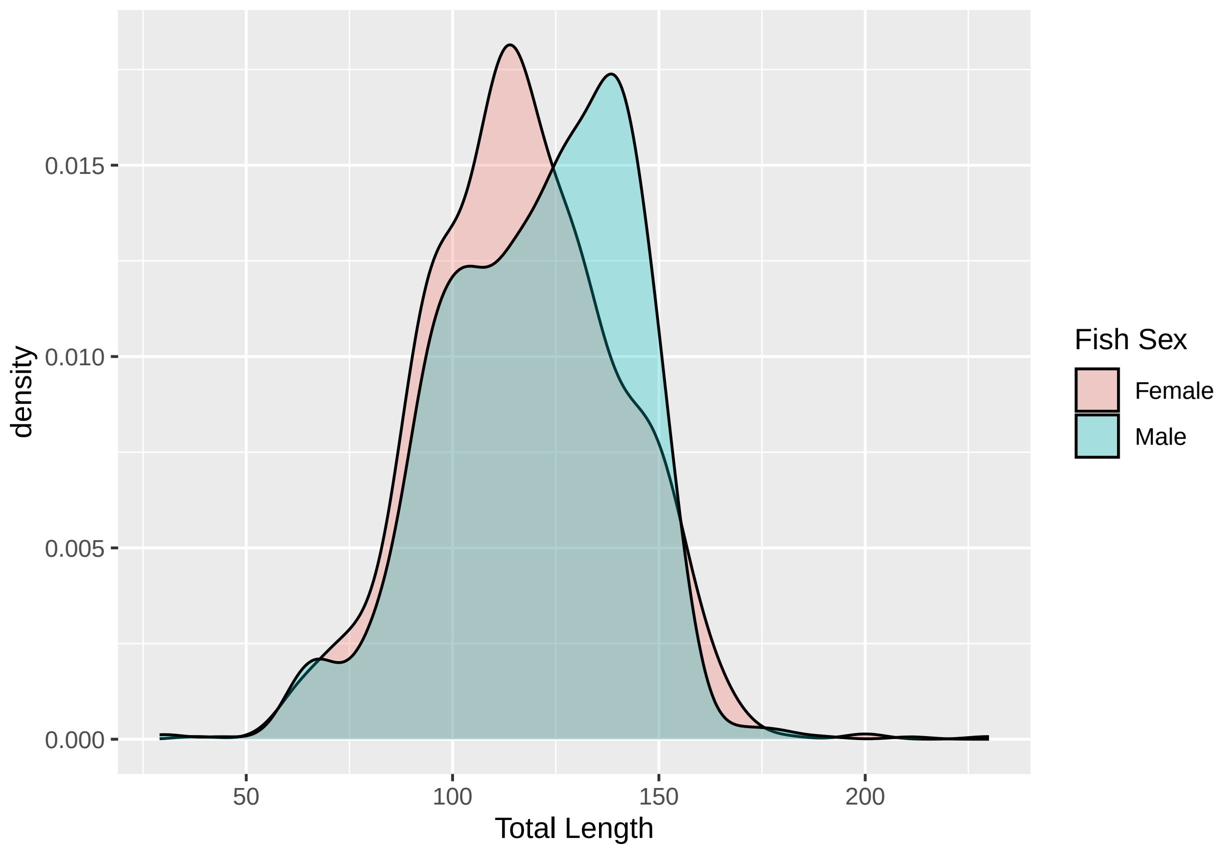
```r
head(sharks, 5)
```

```
## # A tibble: 5 x 4
##   `Calendar Year` `Fish Sex` `Total Length` `Fork Length`
##             <dbl> <chr>               <dbl>         <dbl>
## 1            2007 Male                  106            94
## 2            2007 Female                102            92
## 3            2007 Female                 87            75
## 4            2007 Female                133           116
## 5            2007 Female                 84            71
```

```r
ggplot(sharks) +
  geom_point(aes(x = `Total Length`, y = `Fork Length`, color = `Fish Sex`))
```



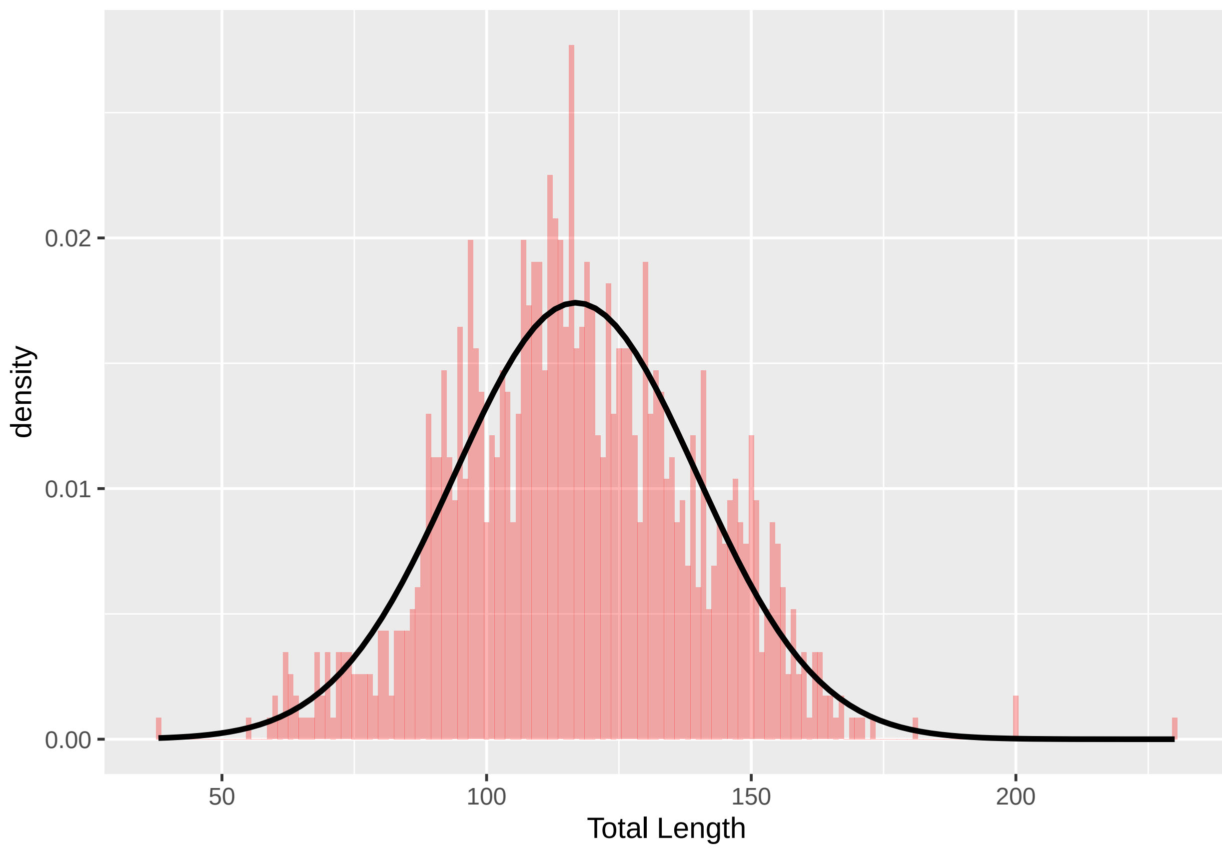

```r
ggplot(sharks) +
  geom_density(aes(x = `Total Length`, fill = `Fish Sex`), alpha = 0.3)
```



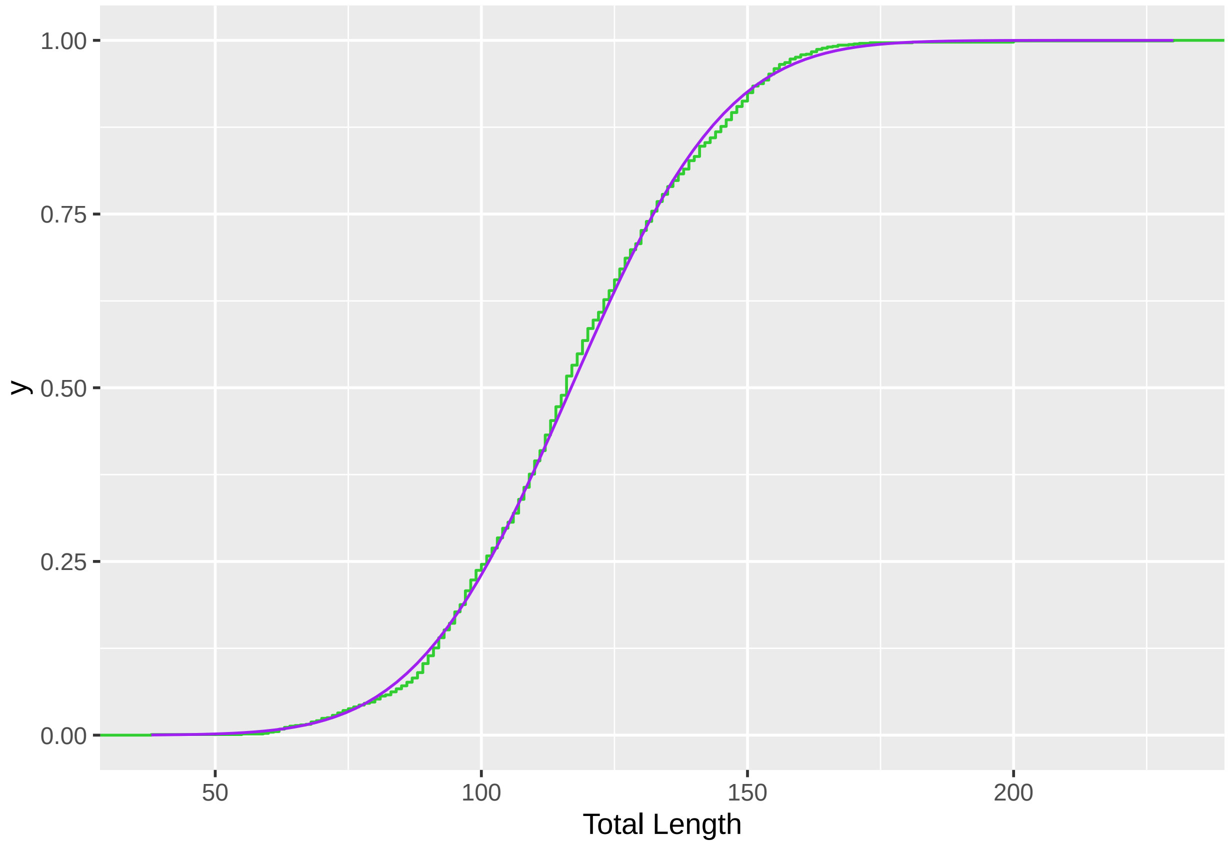

```
females <- filter(sharks, `Fish Sex` == 'Female')
```

```
mean_fs <- mean(females$`Total Length`)
sd_fs <- sd(females$`Total Length`)
ggplot(females) +
  geom_histogram(aes(x = `Total Length`, y = after_stat(density)),
                 fill = 'red', alpha = 0.3, binwidth = 1) +
  stat_function(fun = dnorm, args = list(mean = mean_fs, sd = sd_fs), size = 1)
```

```
## Warning: Using ‘size‘ aesthetic for lines was deprecated in ggplot2 3.4.0.
## i Please use ‘linewidth‘ instead.
## This warning is displayed once every 8 hours.
## Call ‘lifecycle::last_lifecycle_warnings()‘ to see where this warning was
## generated.
```

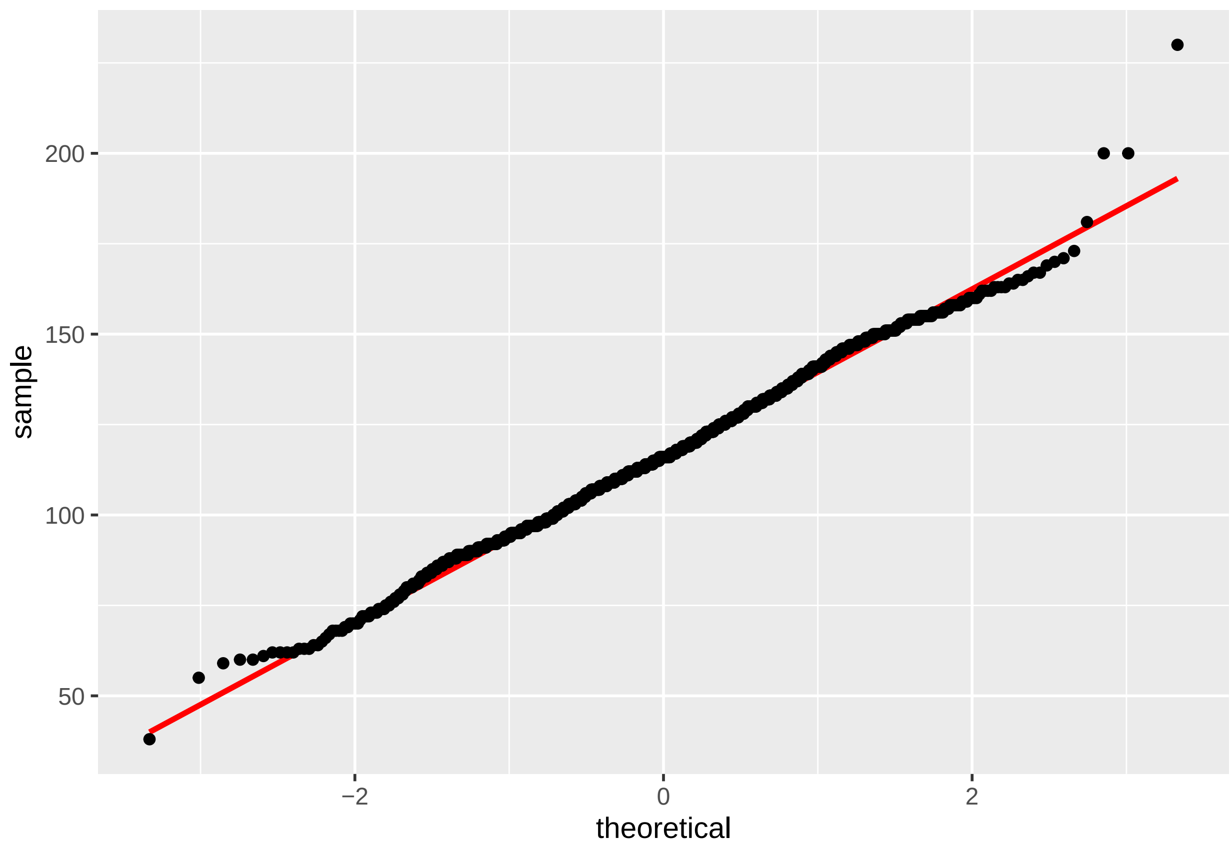



```
ggplot(females) +
  stat_ecdf(aes(x = `Total Length`), color = 'limegreen') +
  geom_line(stat = 'function', fun = pnorm, args = list(mean = mean_fs, sd = sd_fs),
            color = 'purple')
```



Q-Q plot

```
ggplot(females) +
  stat_qq_line(aes(sample = `Total Length`), color = 'red', size = 1) +
  stat_qq(aes(sample = `Total Length`))
```



Z-scores

150 inches long female shark.

```
(z_score <- (150 - mean_fs)/sd_fs)
```

```
## [1] 1.446787
```

```
pnorm(z_score, mean = 0, sd = 1)
```

```
## [1] 0.9260217
```

```
pnorm(z_score)
```

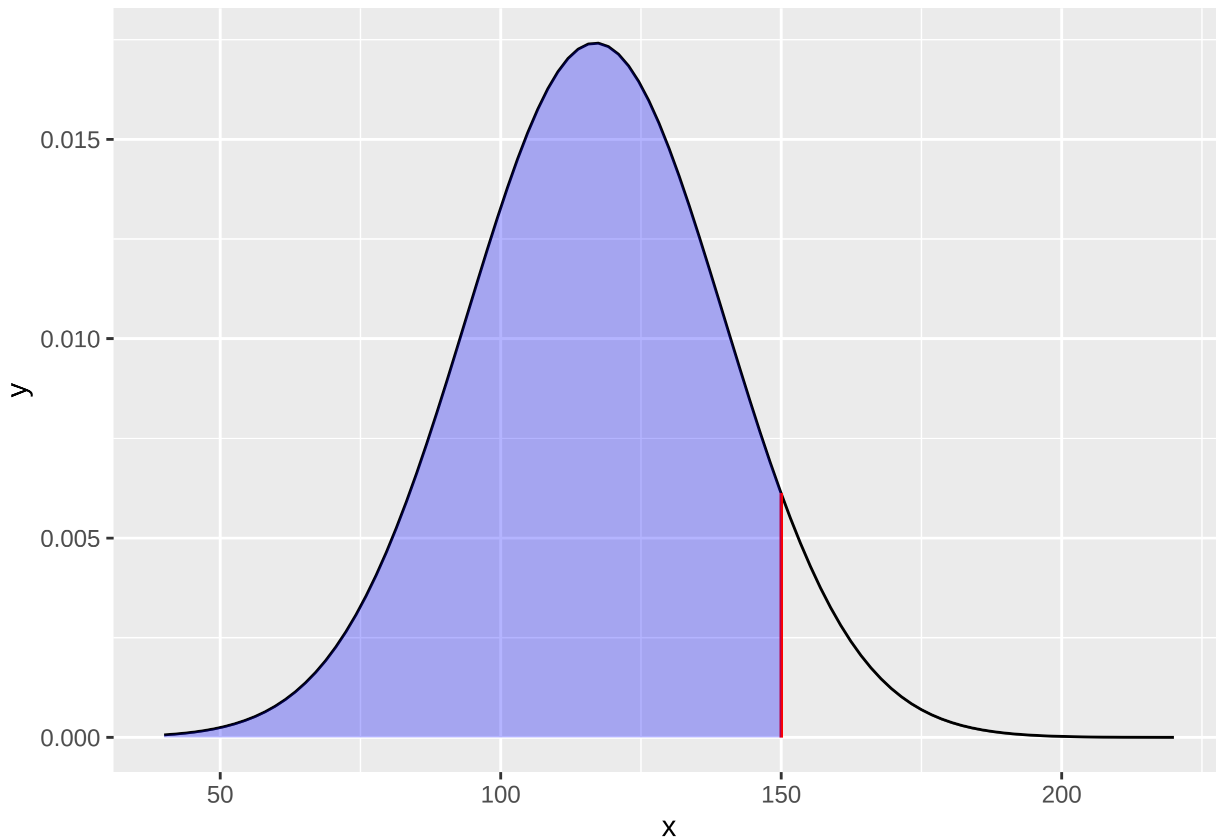
```
## [1] 0.9260217
```

```
pnorm(150, mean = mean_fs, sd = sd_fs)
```

```
## [1] 0.9260217
```

```r
ggplot(data.frame(x = seq(40, 220, length = 500)), aes(x=x)) +
  stat_function(fun = dnorm, args = list(mean = mean_fs, sd = sd_fs)) +
  geom_segment(aes(x = 150, y = 0,
                   xend = 150, yend = dnorm(150, mean = mean_fs, sd = sd_fs)), color = 'red') +
  geom_area(stat = 'function', fun = dnorm, args = list(mean = mean_fs, sd = sd_fs),
            fill = 'blue', xlim = c (40, 150), alpha = 0.3)
```

```
## Warning in geom_segment(aes(x = 150, y = 0, xend = 150, yend = dnorm(150, : All aesthetics have leng
## i Please consider using ‘annotate()‘ or provide this layer with data containing
##   a single row.
```



SE

Mean female shark Total Length. 1155 cases, 40-220 inches.

```r
mean_fs
```

```
## [1] 116.8606
```

Central Limit Theorem: * Samples are independent * Sample size is bigger than 30 * Population distribution is not strongly skewed

YES!

$$SE = \frac{\sigma}{\sqrt{n}}$$

```r
(SE <- sd_fs/sqrt(nrow(females)))
```

```
## [1] 0.6739831
```

95% confidence interval.

1.96 qnorm()

```r
mean_fs - 1.96 *SE
```

```
## [1] 115.5396
```

```r
mean_fs + qnorm(0.025)*SE
```

```
## [1] 115.5396
```

```r
mean_fs + 1.96*SE
```

```
## [1] 118.1816
```

```r
mean_fs + qnorm(0.975)*SE
```

```
## [1] 118.1816
```

We are 95% confident that population mean female shark total length is in between 115.54 inch and 118.18 inch.

99% confidence interval

```r
mean_fs - 2.58 *SE
```

```
## [1] 115.1217
```

```r
mean_fs + qnorm(0.005)*SE
```

```
## [1] 115.1245
```

```r
mean_fs + 2.58*SE
```

```
## [1] 118.5995
```

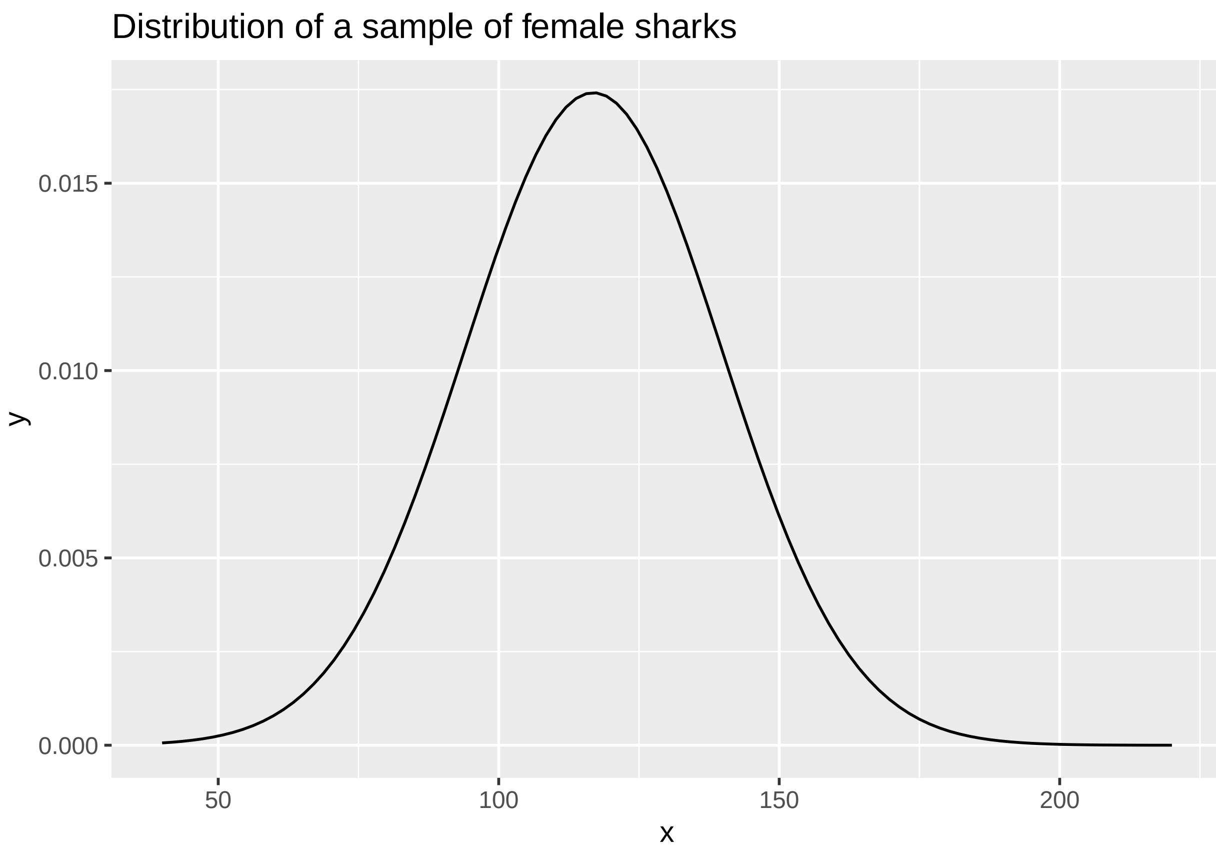
```r
mean_fs + qnorm(0.995)*SE
```

```
## [1] 118.5967
```

We are 99% confident that population mean total length of a female shark is between 115.12 inch and 118.6 inch.

Distribution of a sample - one sample (ONE!)

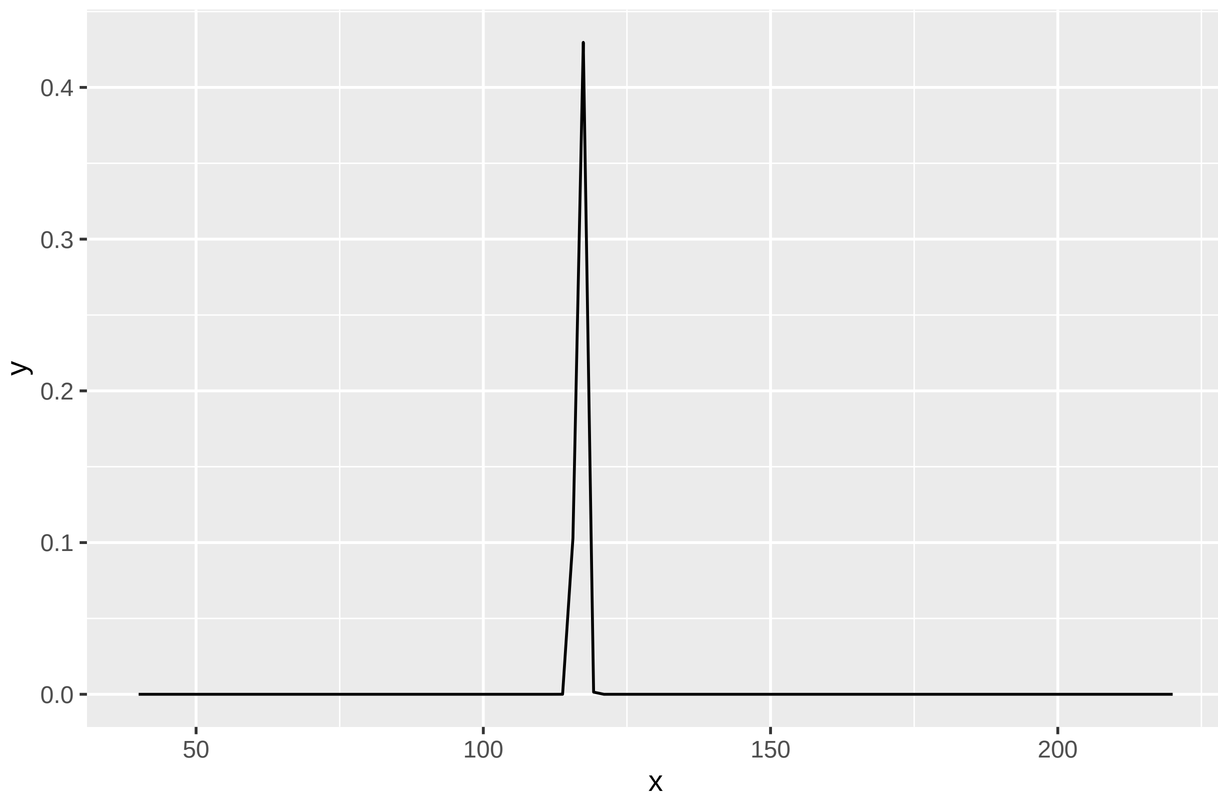
```
ggplot(data.frame(x = seq(40, 220, length = 500)), aes(x = x)) +
  stat_function(fun = dnorm, args = list(mean = mean_fs, sd = sd_fs))+
labs(title = 'Distribution of a sample of female sharks ')
```



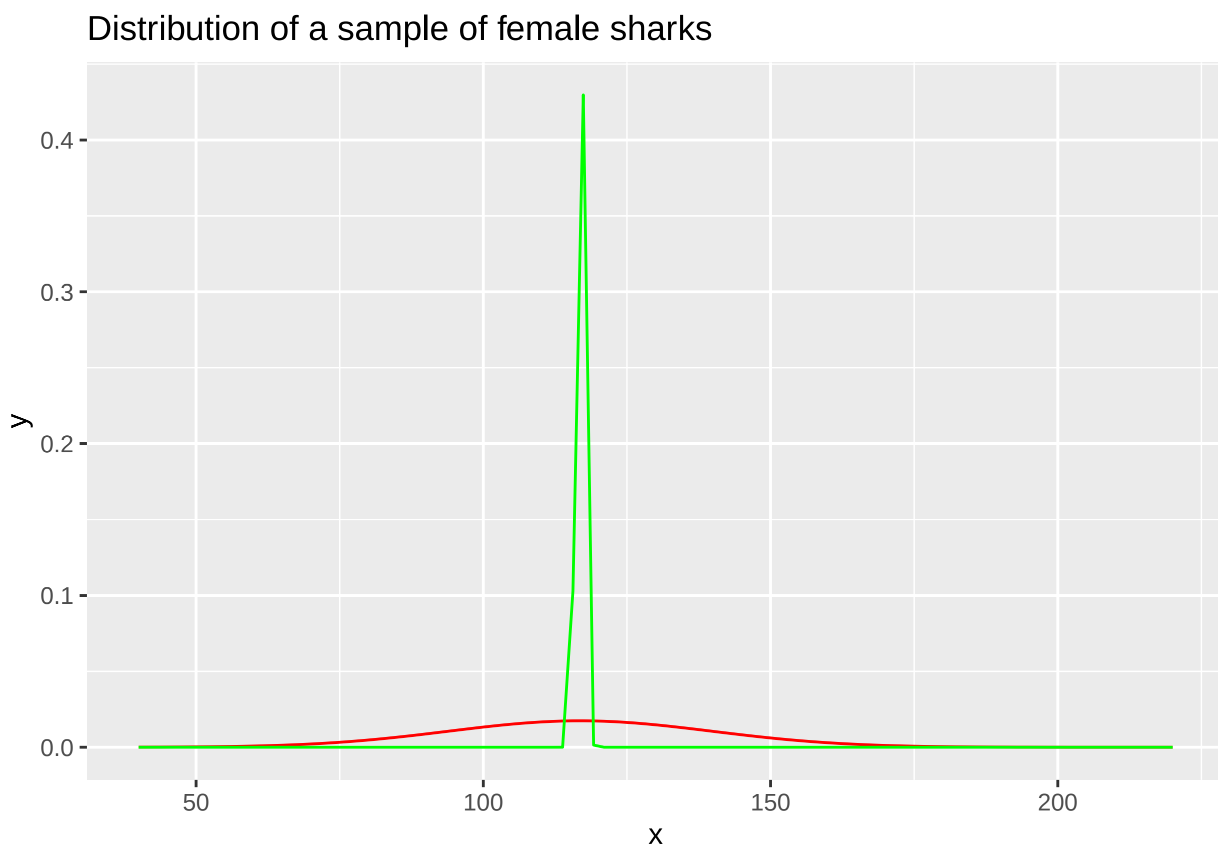
Sampling Distribution - multiple samples.

```
ggplot(data.frame(x = seq(40, 220, length = 500)), aes(x = x)) +
  stat_function(fun = dnorm, args = list(mean = mean_fs, sd = SE))+
labs(title = 'Distribution of a sample of female sharks ')
```

Distribution of a sample of female sharks




```r
ggplot(data.frame(x = seq(40, 220, length = 500)), aes(x = x)) +
  stat_function(fun = dnorm, args = list(mean = mean_fs, sd = sd_fs), color = 'red')+
  stat_function(fun = dnorm, args = list(mean = mean_fs, sd = SE), color = 'green')+
  labs(title = 'Distribution of a sample of female sharks ')
```



Tests

One sample proportion test

$$SE = \sqrt{\frac{p \cdot (1-p)}{n}}$$

0) Check CLT conditions. 1) Setup hypotheses 2) Assume threshold for values * $\alpha = 0.05$ is common 3) Calculate the results * point estimate: $\hat{p}$ * standard error: $SE = \sqrt{\frac{p \cdot (1-p)}{n}}$ * p-value: $2 \cdot P(Z > |z|)$ 4) Draw a conclusion

H_0: 50% of NBA players is over 200 cm tall: $p = 0.5$

H_a: More than 50% of NBA players is over 200 cm tall: $p > 0.5$

```
alpha <- 0.05
```

```
nba <- readr::read_csv('nba_ht_wt.csv')
```

```
## Rows: 505 Columns: 5
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (2): Player, Position
## dbl (3): Height, Weight, Age
```

```
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
nba <- nba %>% mutate(Height = Height* 2.54)
```

```
n_success <- nba %>% filter(Height > 200) %>% nrow()
```

```
nba %>% filter(Height <= 200) %>% nrow()
```

```
## [1] 203
```

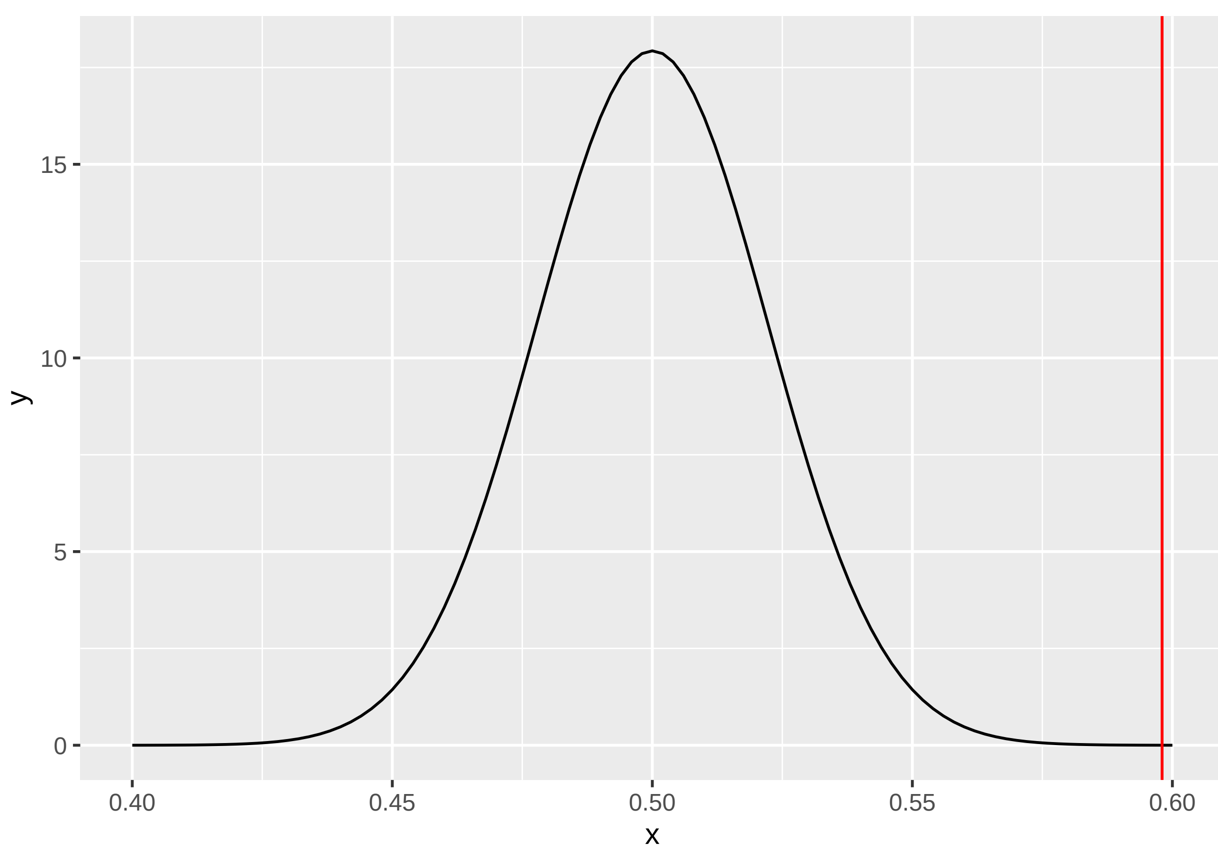
```
n_sample <- nrow(nba)
(point_estimate_nba <-  n_success / n_sample)
```

```
## [1] 0.5980198
```

```
(SE_nba <- sqrt(0.5 * 0.5/n_sample))
```

```
## [1] 0.02224971
```

```
ggplot(data.frame(x = seq(0.4,0.6, length = 100)), aes(x=x)) +
  stat_function(fun = dnorm, args = list(mean = 0.5, sd = SE_nba)) +
  geom_vline(xintercept = point_estimate_nba, color = "red")
```



```
(p_value_nba <- (1-pnorm(point_estimate_nba, mean = 0.5, sd = SE_nba)))
```

```
## [1] 5.278415e-06
```

We reject null hypothesis, because p-value is less than 0.05.

Difference of proportions test

0) Check CLT conditions.
1) Setup hypotheses
2) Assume threshold for values

- $\alpha = 0.05$ is common

3) Calculate the results

- point estimate: $\hat{p}$
- standard error: $SE = \sqrt{\frac{p \cdot (1-p)}{n}}$
- p-value: $2 \cdot P(Z > |z|)$

4) Draw a conclusion

A clothes producer is looking for a new supplier of zipppers.Two factories are frontrunners. The producer wants to decide based on one days production results.

First factory produces 23 935 zippers, out of which 132 were faulty. Second factory produces 22 312 zippers, out of which 111 were faulty. We want to check if it's safe to assume that they have the same production of faulty zippers, and due to that we can choosed based squerly on price

H_0: Proportion of faulty zippers is the same in both factories: $p_1 = p_2$

H_a: Proportion of faulty zippers is different in both factories: $p_1 \neq p_2$

Pooled proportion: $\hat{p} = \frac{x_1 + x_2}{n_1 + n_2}$

```
(p_pooled <- (132 + 111) / (23935 + 22312))
```

```
## [1] 0.005254395
```

```
p_pooled * 23935
```

```
## [1] 125.7639
```

```
p_pooled * 22312
```

```
## [1] 117.2361
```

```
(1- p_pooled) * 23935
```

```
## [1] 23809.24
```

```r
(1- p_pooled) * 22312
```

```
## [1] 22194.76
```

```r
alpha <- 0.05
```

```r
proportion1 <- 132 / 23935
proportion2 <- 111 / 22312
(point_estimate_zippers <- proportion1 - proportion2)
```

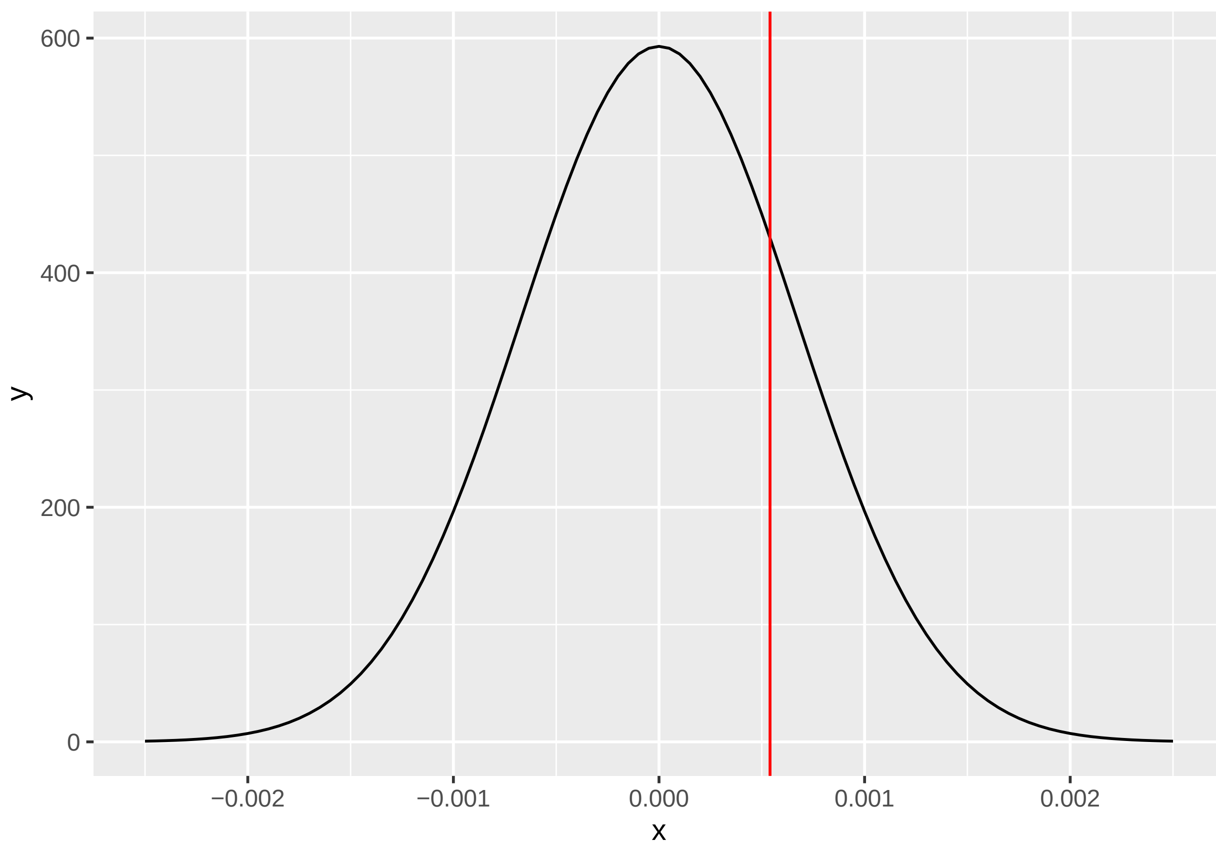
```
## [1] 0.0005400349
```

$$SE = \sqrt{\hat{p} \cdot (1 - \hat{p}) \cdot \left(\frac{1}{n_1} + \frac{1}{n_2}\right)}$$

```r
(SE_zippers <- sqrt(p_pooled * (1-p_pooled) * (1/23935 + 1/22312)))
```

```
## [1] 0.0006727802
```

```r
ggplot(data.frame(x = seq(-0.0025,0.0025, length = 100)), aes(x=x)) +
  stat_function(fun = dnorm, args = list(mean = 0, sd = SE_zippers)) +
  geom_vline(xintercept = point_estimate_zippers, color = "red")
```



```r
(p_value_zippers <- 2 * (1 - pnorm(point_estimate_zippers, mean = 0, sd = SE_zippers)))
```

```
## [1] 0.4221531
```

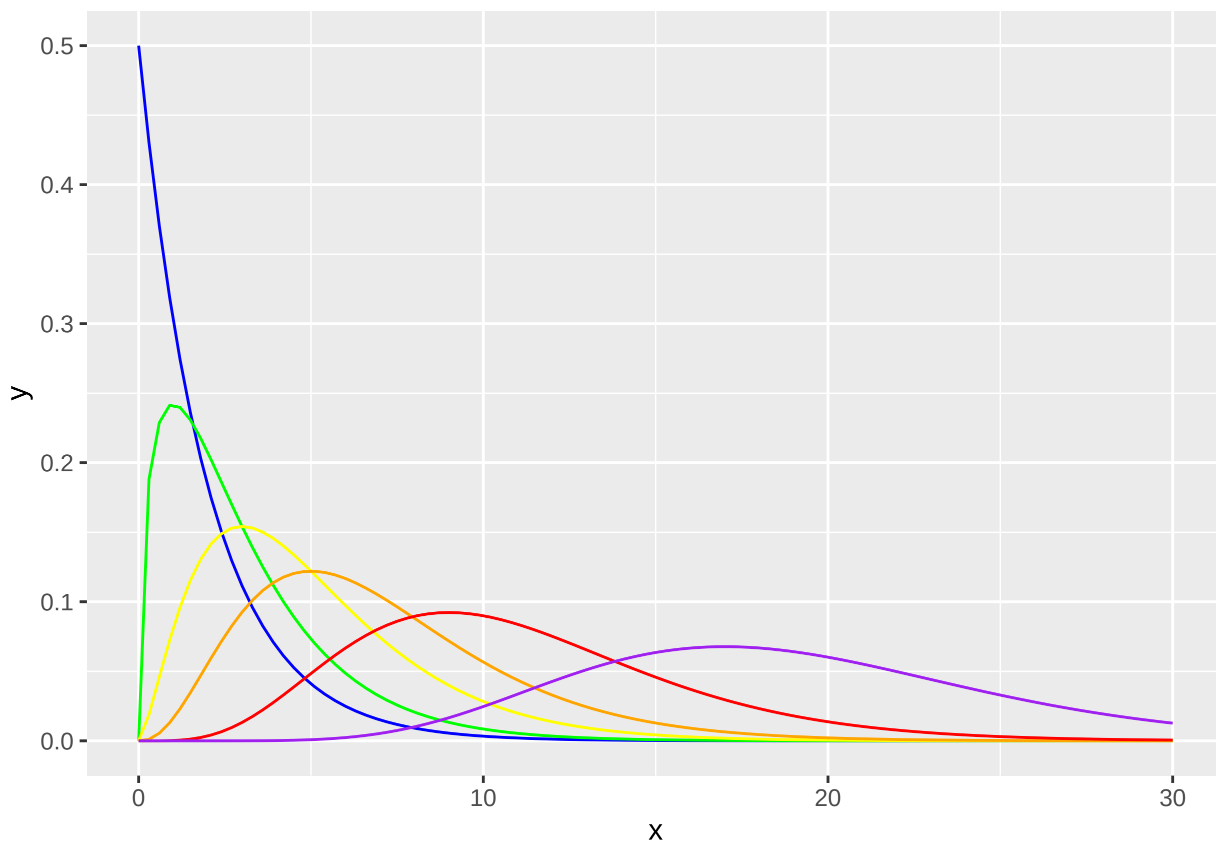
```r
p_value_zippers > alpha
```

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

We accept Null hypothesis, because p-value is greater than 0.05. There is no evidence that the proportion of faulty zippers is different in both factories.

χ^2 distribution

```r
ggplot(data.frame(x=seq(0,30,length=100)), aes(x=x))+
  stat_function(fun = dchisq, args = list(df = 2), color = 'blue')+
  stat_function(fun = dchisq, args = list(df = 3), color = 'green')+
  stat_function(fun = dchisq, args = list(df = 5), color = 'yellow')+
    stat_function(fun = dchisq, args = list(df = 7), color = 'orange')+
      stat_function(fun = dchisq, args = list(df = 11), color = 'red')+
        stat_function(fun = dchisq, args = list(df = 19), color = 'purple')
```



χ^2 goodness of fit test

0) Check conditions.
1) Setup hypotheses
2) Setup threshold

- $\alpha = 0.05$ is common

3) Calculate chi^2 statistic
4) Calculate p-value
5) Draw a conclusion

Sample of 669 Haribo Gummy Bears

```
(gummy_bears <- c(83, 142, 100, 103, 104, 137))
```

```
## [1]  83 142 100 103 104 137
```

H_0: The distribution of flavors is equal

H_a: The distribution of flavors is not equal

```
alpha <- 0.05
```

```
(expected <- sum(gummy_bears) / length(gummy_bears))
```

```
## [1] 111.5
```

$$Z_n = \sum_{i=1}^{k} \frac{(O_i - E_i)^2}{E_i}$$

$$X^2 = \sum_{i=1}^{k} \frac{(O_i - E_i)^2}{E_i}$$

```
(gummy_Z <- (gummy_bears - expected) / sqrt(expected))
```

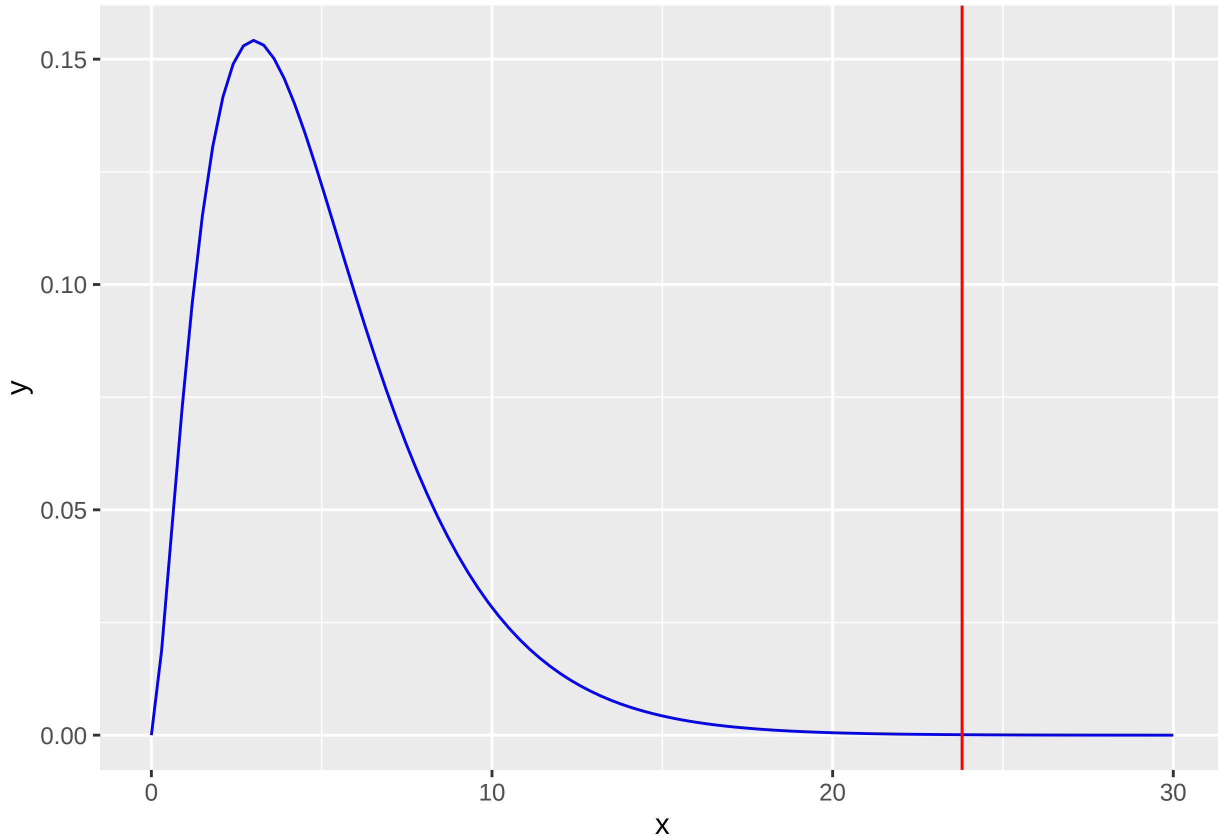
```
## [1] -2.6990282  2.8884337 -1.0890816 -0.8049733 -0.7102706  2.4149200
```

```
(chi_score <- sum(gummy_Z^2))
```

```
## [1] 23.79821
```

degrees of freedom: df = 5

```
ggplot(data.frame(x=seq(0,30,length=100)), aes(x=x))+
  stat_function(fun = dchisq, args = list(df = 5), color = 'blue')+
  geom_vline(xintercept = chi_score, color = "red")
```




```r
(p_value <- 1 - pchisq(chi_score, df = 5))
```

```
## [1] 0.0002373797
```

```r
p_value > alpha
```

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

We reject null hypothesis, because p-value is less than 0.05. There is evidence that the distribution of flavors is not equal.

Just run the test with no calculations

```r
chisq.test(x= gummy_bears, p = rep(c(expected), length(gummy_bears)), rescale.p = TRUE)
```

```
##
##  Chi-squared test for given probabilities
##
## data:  gummy_bears
## X-squared = 23.798, df = 5, p-value = 0.0002374
```

χ^2 test for Independence

```
(movies_pizza <- tribble(
  ~Movie, ~Pepperoni, ~Mushrooms, ~Kebab,
  "Frozen 2", 20, 10, 5,
  "Joker", 15, 12, 15,
  "Someone Great", 8, 13,2
))
```

```
## # A tibble: 3 x 4
##   Movie         Pepperoni Mushrooms Kebab
##   <chr>             <dbl>     <dbl> <dbl>
## 1 Frozen 2             20        10     5
## 2 Joker                15        12    15
## 3 Someone Great         8        13     2
```

0) Check conditions.
1) Setup hypotheses
2) Get the observed count for each category
3) Calculate expected count for each category
4) Calculate test statistic
5) Calculate p-value
6) Draw a conclusion

H_0: There is no association between movie and pizza topping

H_a: There is an association between movie and pizza topping

```
p_total <- sum(movies_pizza$Pepperoni)
m_total <- sum(movies_pizza$Mushrooms)
k_total <- sum(movies_pizza$Kebab)
total <- p_total + m_total + k_total
```

```
(movies_pizza <- movies_pizza %>%
        mutate(P_exp = (Pepperoni + Mushrooms + Kebab) * p_total/total)  %>%
        mutate(M_exp = (Pepperoni + Mushrooms + Kebab) * m_total/total) %>%
        mutate (K_exp = (Pepperoni + Mushrooms + Kebab) * k_total/total))
```

```
## # A tibble: 3 x 7
##   Movie         Pepperoni Mushrooms Kebab P_exp M_exp K_exp
##   <chr>             <dbl>     <dbl> <dbl> <dbl> <dbl> <dbl>
## 1 Frozen 2             20        10     5 15.0  12.2   7.7
## 2 Joker                15        12    15 18.1  14.7   9.24
## 3 Someone Great         8        13     2  9.89  8.05  5.06
```

```
(movies_pizza <- movies_pizza %>%
  mutate(P_Z = ((Pepperoni - P_exp) / sqrt(P_exp))^2) %>%
  mutate(M_Z = ((Mushrooms - M_exp) / sqrt(M_exp))^2) %>%
  mutate(K_Z = ((Kebab - K_exp) / sqrt(K_exp))^2))
```

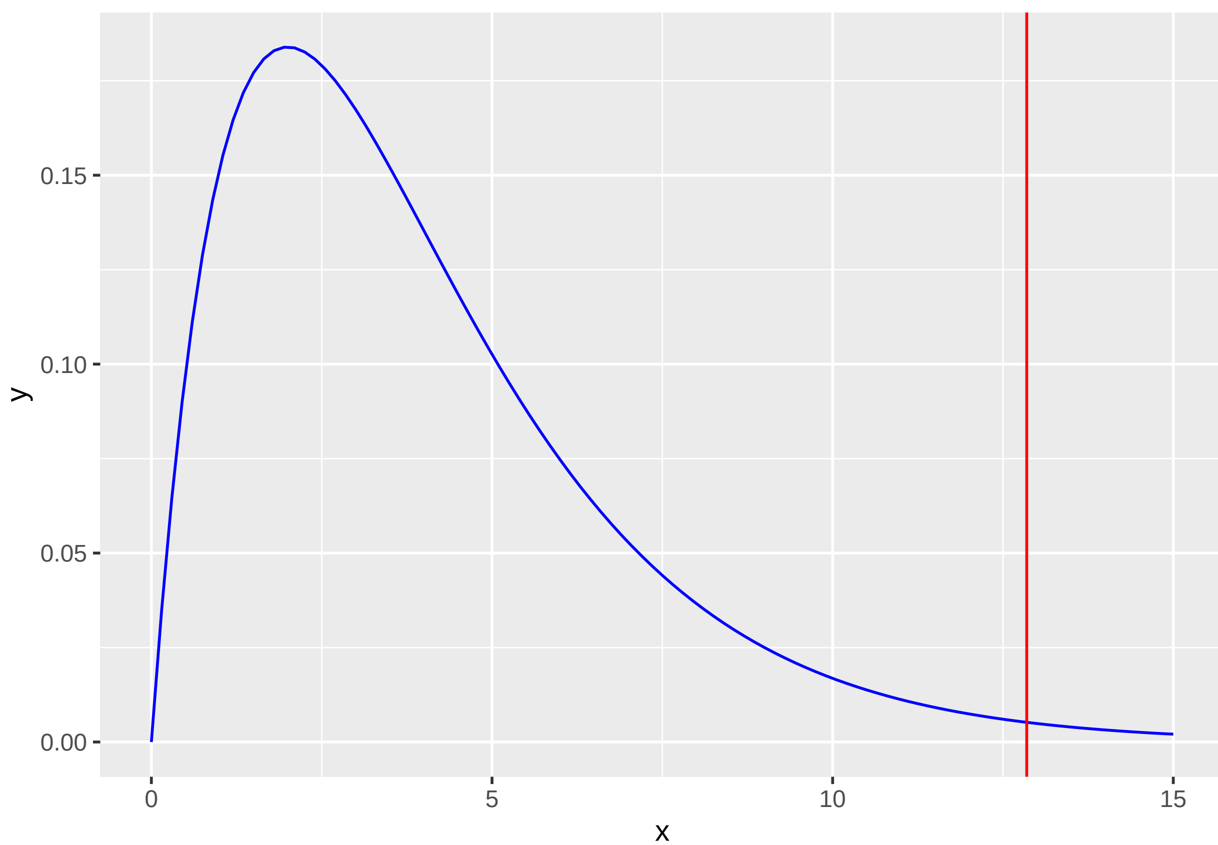
```
## # A tibble: 3 x 10
##   Movie          Pepperoni Mushrooms Kebab P_exp M_exp K_exp   P_Z   M_Z   K_Z
##   <chr>              <dbl>     <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
## 1 Frozen 2              20        10     5 15.0  12.2   7.7  1.63  0.413 0.947
## 2 Joker                 15        12    15 18.1  14.7   9.24 0.518 0.496 3.59
## 3 Someone Great          8        13     2  9.89  8.05  5.06 0.361 3.04  1.85
```

```r
(chi_score_mp <- sum(movies_pizza$P_Z) + sum(movies_pizza$M_Z) + sum(movies_pizza$K_Z))
```

```
## [1] 12.84862
```

df $= 4$

```r
ggplot(data.frame(x=seq(0,15,length=100)), aes(x=x))+
  stat_function(fun = dchisq, args = list(df = 4), color = 'blue')+
  geom_vline(xintercept = chi_score_mp, color = "red")
```




```r
alpha <- 0.05
```

```r
(p_value_mp <- 1 - pchisq(chi_score_mp, df = 4))
```

```
## [1] 0.01203966
```

```r
p_value_mp > alpha
```

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

We reject null hypothesis, because p-value is less than 0.05. There is evidence that there is an association between movie and pizza topping.

Just run the test with no calculations

```r
(movies_pizza <- tribble(
  ~Movie, ~Pepperoni, ~Mushrooms, ~Kebab,
  "Frozen 2", 20, 10, 5,
  "Joker", 15, 12, 15,
  "Someone Great", 8, 13,2
))
```

```
## # A tibble: 3 x 4
##   Movie         Pepperoni Mushrooms Kebab
##   <chr>             <dbl>     <dbl> <dbl>
## 1 Frozen 2             20        10     5
## 2 Joker                15        12    15
## 3 Someone Great         8        13     2
```

```r
movies_pizza %>% select(-Movie) %>% chisq.test()
```

```
##
##  Pearson's Chi-squared test
##
## data:  .
## X-squared = 12.849, df = 4, p-value = 0.01204
```

data Transformation

filter

```r
kiwi <- readr::read_csv('kiwi.csv')
```

```
## Rows: 700 Columns: 5
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (3): Species_code, Gender, Location
## dbl (2): Weight(kg), Height(cm)
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```r
kiwi %>% filter(Species_code == 'GS', Gender == 'M', `Weight(kg)` > 2.2)
```

```
## # A tibble: 54 x 5
##    Species_code Gender `Weight(kg)` `Height(cm)` Location
##    <chr>        <chr>         <dbl>        <dbl> <chr>
##  1 GS           M              2.46         45.9 CW
##  2 GS           M              2.33         46.3 EC
##  3 GS           M              2.29         44.4 EC
##  4 GS           M              2.63         46.7 EC
##  5 GS           M              2.68         47   EC
##  6 GS           M              2.54         46.4 EC
##  7 GS           M              2.61         45.7 CW
##  8 GS           M              2.51         48   CW
##  9 GS           M              2.83         43.9 EC
## 10 GS           M              2.92         44.1 EC
## # i 44 more rows
```

```r
kiwi %>% filter(Species_code == 'GS') %>%
  filter(Gender == 'M') %>%
  filter(`Weight(kg)` > 2.2)
```

```
## # A tibble: 54 x 5
##    Species_code Gender `Weight(kg)` `Height(cm)` Location
##    <chr>        <chr>         <dbl>        <dbl> <chr>
##  1 GS           M              2.46         45.9 CW
##  2 GS           M              2.33         46.3 EC
##  3 GS           M              2.29         44.4 EC
##  4 GS           M              2.63         46.7 EC
##  5 GS           M              2.68         47   EC
##  6 GS           M              2.54         46.4 EC
##  7 GS           M              2.61         45.7 CW
##  8 GS           M              2.51         48   CW
##  9 GS           M              2.83         43.9 EC
## 10 GS           M              2.92         44.1 EC
## # i 44 more rows
```

```r
kiwi %>% filter(Species_code != 'GS')
```

```
## # A tibble: 537 x 5
##    Species_code Gender `Weight(kg)` `Height(cm)` Location
##    <chr>        <chr>         <dbl>        <dbl> <chr>
##  1 Tok          M              2.05         36.5 StI
##  2 Tok          F              2.40         40.3 SF
##  3 NIBr         M              1.81         36.1 E
##  4 NIBr         F              2.89         41.4 W
##  5 NIBr         M              2.05         38.1 E
##  6 Tok          F              2.93         41.8 NF
##  7 Tok          F              2.85         41.7 NF
##  8 Tok          M              2.25         38.4 StI
##  9 Tok          M              1.97         37.2 StI
## 10 Tok          M              2.43         37.1 NF
## # i 527 more rows
```

```r
kiwi %>% filter(Species_code != 'Tok' | Species_code == 'NIBr')
```

```
## # A tibble: 438 x 5
##    Species_code Gender `Weight(kg)` `Height(cm)` Location
##    <chr>        <chr>         <dbl>        <dbl> <chr>
##  1 GS           M              2.01         42.9 NWN
##  2 NIBr         M              1.81         36.1 E
##  3 NIBr         F              2.89         41.4 W
##  4 NIBr         M              2.05         38.1 E
##  5 NIBr         F              2.74         39.1 W
##  6 GS           F              3.17         47.8 NWN
##  7 NIBr         M              2.29         37.5 W
##  8 GS           M              2.46         45.9 CW
##  9 GS           F              3.72         46.8 NWN
## 10 NIBr         F              2.99         42.5 W
## # i 428 more rows
```

Logical opperators & | > >= < <= != ==

Select

```r
head(kiwi, 5)
```

```
## # A tibble: 5 x 5
##   Species_code Gender `Weight(kg)` `Height(cm)` Location
##   <chr>        <chr>         <dbl>        <dbl> <chr>
## 1 Tok          M              2.05         36.5 StI
## 2 Tok          F              2.40         40.3 SF
## 3 GS           M              2.01         42.9 NWN
## 4 NIBr         M              1.81         36.1 E
## 5 NIBr         F              2.89         41.4 W
```

```r
kiwi %>%  select(1:4)
```

```
## # A tibble: 700 x 4
##    Species_code Gender `Weight(kg)` `Height(cm)`
##    <chr>        <chr>         <dbl>        <dbl>
##  1 Tok          M              2.05         36.5
##  2 Tok          F              2.40         40.3
##  3 GS           M              2.01         42.9
##  4 NIBr         M              1.81         36.1
##  5 NIBr         F              2.89         41.4
##  6 NIBr         M              2.05         38.1
##  7 Tok          F              2.93         41.8
##  8 Tok          F              2.85         41.7
##  9 Tok          M              2.25         38.4
## 10 Tok          M              1.97         37.2
## # i 690 more rows
```

```r
kiwi %>% select(Species_code, Gender)
```

```
## # A tibble: 700 x 2
##    Species_code Gender
##    <chr>        <chr>
##  1 Tok          M
##  2 Tok          F
##  3 GS           M
##  4 NIBr         M
##  5 NIBr         F
##  6 NIBr         M
##  7 Tok          F
##  8 Tok          F
##  9 Tok          M
## 10 Tok          M
## # i 690 more rows
```

```r
kiwi %>%  select(-Location)
```

```
## # A tibble: 700 x 4
##    Species_code Gender `Weight(kg)` `Height(cm)`
##    <chr>        <chr>         <dbl>        <dbl>
##  1 Tok          M              2.05         36.5
##  2 Tok          F              2.40         40.3
##  3 GS           M              2.01         42.9
##  4 NIBr         M              1.81         36.1
##  5 NIBr         F              2.89         41.4
##  6 NIBr         M              2.05         38.1
##  7 Tok          F              2.93         41.8
##  8 Tok          F              2.85         41.7
##  9 Tok          M              2.25         38.4
## 10 Tok          M              1.97         37.2
## # i 690 more rows
```

Mutate

BMI = Weight(kg)/Height(m)^2

```r
Kiwi_BMI <- kiwi %>%
  mutate(`Height(m)` = `Height(cm)` / 100) %>%
  mutate(BMI = `Weight(kg)` / `Height(m)`^2)

Kiwi_BMI
```

```
## # A tibble: 700 x 7
##    Species_code Gender `Weight(kg)` `Height(cm)` Location `Height(m)`   BMI
##    <chr>        <chr>         <dbl>        <dbl> <chr>          <dbl> <dbl>
##  1 Tok          M              2.05         36.5 StI            0.365  15.4
##  2 Tok          F              2.40         40.3 SF             0.403  14.8
##  3 GS           M              2.01         42.9 NWN            0.429  10.9
##  4 NIBr         M              1.81         36.1 E              0.361  13.9
```

```
##   5 NIBr         F              2.89          41.4 W                0.414   16.9
##   6 NIBr         M              2.05          38.1 E                0.381   14.1
##   7 Tok          F              2.93          41.8 NF               0.418   16.8
##   8 Tok          F              2.85          41.7 NF               0.417   16.4
##   9 Tok          M              2.25          38.4 StI              0.384   15.2
##  10 Tok          M              1.97          37.2 StI              0.372   14.2
## # i 690 more rows
```

summarise(), arrange(), group_by()

```
kiwi %>%
  group_by(Species_code) %>%
  tally()
```

```
## # A tibble: 3 x 2
##   Species_code     n
##   <chr>        <int>
## 1 GS             163
## 2 NIBr           275
## 3 Tok            262
```

```
kiwi %>%
  group_by(Species_code, Gender) %>%
  summarise(`mean weight` = mean(`Weight(kg)`), `mean height` =
            mean(`Height(cm)`)) %>%
  arrange(desc(`mean weight`)) %>%
  knitr::kable()
```

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

Species_code	Gender	mean weight	mean height
GS	F	3.344901	48.07531
NIBr	F	2.791593	40.00214
Tok	F	2.790839	40.04825
GS	M	2.350415	45.30854
NIBr	M	2.242889	36.96370
Tok	M	2.203302	37.01345

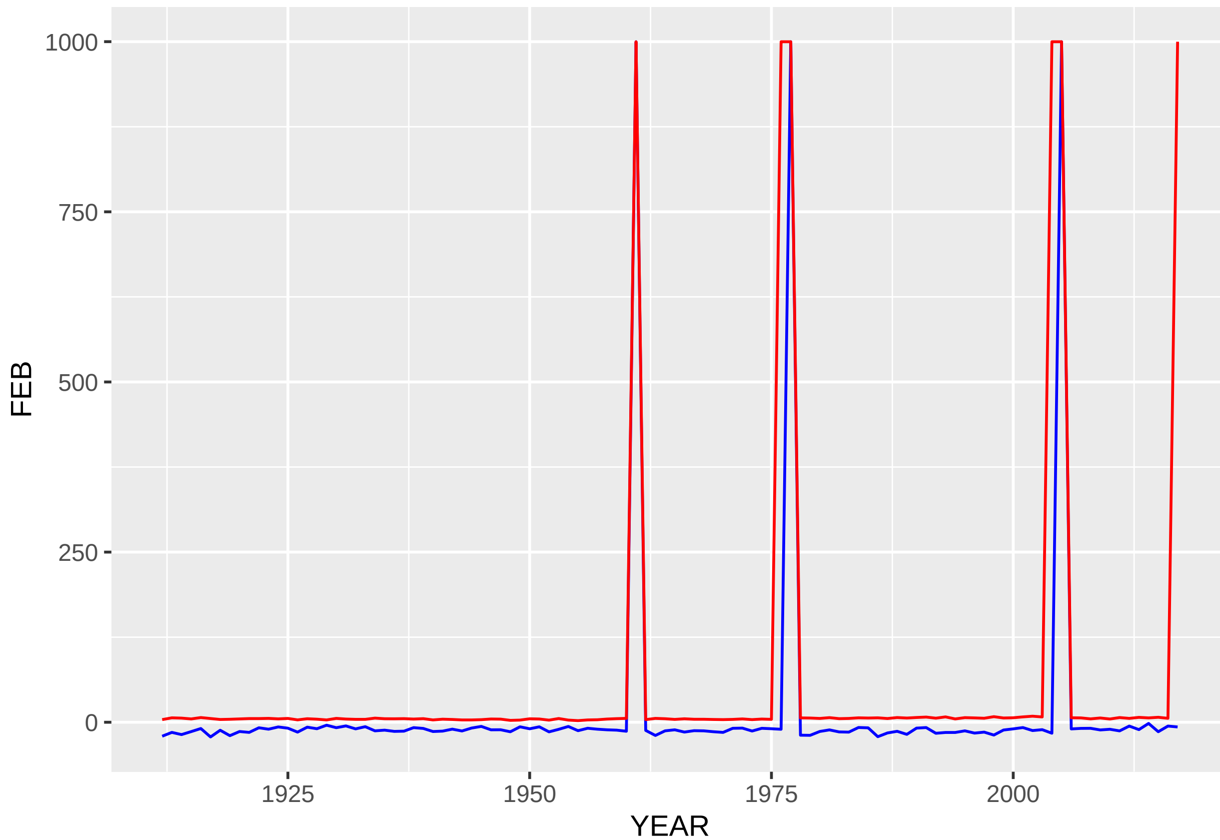
Missing values

```
svalbard <- readr::read_csv('svalbard-climate-1912-2017.csv')
```

```
## Rows: 106 Columns: 18
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## dbl (18): YEAR, JAN, FEB, MAR, APR, MAY, JUN, JUL, AUG, SEP, OCT, NOV, DEC, ...
```

```
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
svalbard %>%
  ggplot() +
  geom_line(aes(x = YEAR, y = FEB), colour = 'blue') +
  geom_line(aes(x = YEAR, y = AUG), colour = 'red')
```

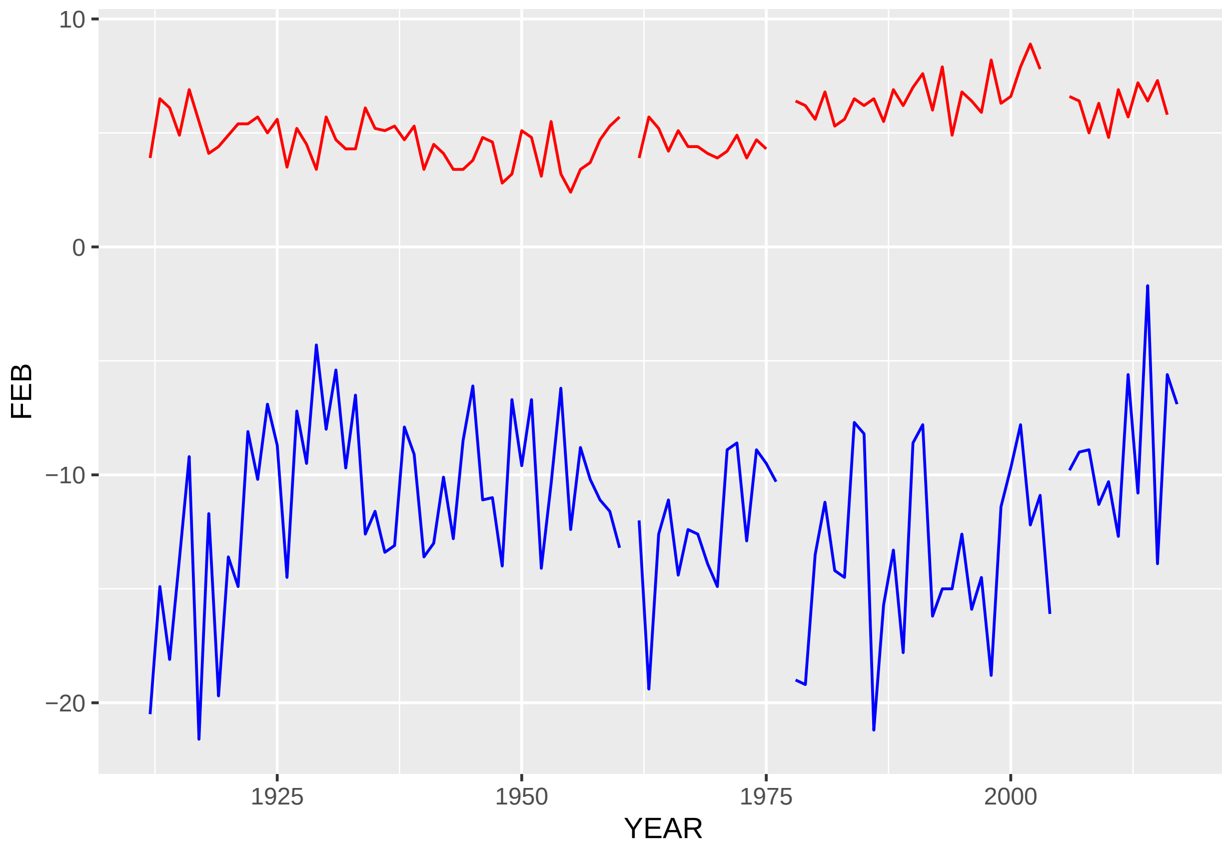



```
svalbard <- readr::read_csv('svalbard-climate-1912-2017.csv', na = c('NA','999.9'))
```

```
## Rows: 106 Columns: 18
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## dbl (18): YEAR, JAN, FEB, MAR, APR, MAY, JUN, JUL, AUG, SEP, OCT, NOV, DEC, ...
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
svalbard %>%
  ggplot() +
  geom_line(aes(x = YEAR, y = FEB), colour = 'blue') +
  geom_line(aes(x = YEAR, y = AUG), colour = 'red')
```

```
## Warning: Removed 1 row containing missing values or values outside the scale range
## (`geom_line()`).
```




```r
mean(svalbard$FEB, na.rm = TRUE)
```

```
## [1] -11.6767
```

```r
sd(svalbard$JAN, na.rm = TRUE)
```

```
## [1] 4.742159
```

```r
median(svalbard$APR, na.rm = TRUE)
```

```
## [1] -9.35
```

```r
svalbard %>%
  filter(!is.na(JAN), YEAR >= 2000)
```

```
## # A tibble: 18 x 18
##     YEAR   JAN   FEB   MAR   APR   MAY   JUN   JUL   AUG   SEP   OCT   NOV   DEC
##    <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
##  1  2000  -7.1  -9.7 -13.5 -11.4  -3     3.5   7.5   6.6   3.5   2.7  -2.3  -8.3
##  2  2001  -6.9  -7.8 -15.2 -11    -1.5   4.8   7.8   7.9   4.6  -3.8  -9.7  -7.6
```

```
##   3  2002 -11.3 -12.2 -16.9  -4.7  -1.9   5.1   8.5   8.9   1.5  -1.6  -4.2  -6.1
##   4  2003 -17.2 -10.9 -15.9  -8.2  -1.1   4.2   8.4   7.8   2    -2.9  -5.3 -17.7
##   5  2004 -15.6 -16.1  NA    NA    NA    NA    NA    NA    NA    NA    NA    NA
##   6  2005 -18    NA    NA    NA    NA    NA    NA    NA    NA    NA    NA    -3.8
##   7  2006  -2.7  -9.8 -13.1   0     0.9   4.8   7.5   6.6   1.7  -5.5  -4.1  -6.3
##   8  2007  -9    -9    -6.7  -9.4  -1.4   5.1   7.8   6.4   2.1  -2.5  -5.3  -8.6
##   9  2008  -7.5  -8.9 -14.7 -10.5  -1.9   3.4   6.2   5     3.3  -5.7  -8.6  -8
## 10  2009 -12.7 -11.3 -11.5 -15.9  -1     3     7.7   6.3   1.7  -3.6  -1.9  -5.5
## 11  2010  -7.2 -10.3 -15.8  -7.7   0     3.5   6.6   4.8   2    -2.9 -11   -11.6
## 12  2011 -15.2 -12.7 -12.7  -5.8  -1.6   4.8   6.9   6.9   4.4  -2.6  -6    -6.6
## 13  2012  -3.4  -5.6  -5.3  -9.2  -2.4   3.9   6.6   5.7   3    -2.7  -6.4  -7.9
## 14  2013  -8.3 -10.8 -14.3  -9.3  -1.5   4.4   7.4   7.2   3.7  -4.7  -8    -8.1
## 15  2014  -4.1  -1.7  -8.6  -9.7  -2     3.8   7.2   6.4   1.6  -1.9  -6.4  -9.6
## 16  2015  -7.3 -13.9  -6.8  -5.3  -2.2   4.6   8.1   7.3   2.8  -1.8  -3.4  -6
## 17  2016  -3.8  -5.6  -6.9  -6.2   1.4   5     9     5.8   4.1   3.2  -0.7  -6
## 18  2017 -10.3  -6.9 -11.6  -8.3  -3.9   4.6   6.9  NA    NA    NA    NA    NA
## # i 5 more variables: 'D-J-F' <dbl>, 'M-A-M' <dbl>, 'J-J-A' <dbl>,
## #   'S-O-N' <dbl>, metANN <dbl>
```

Regression

```r
data <- readr::read_csv('weight-height.csv')
```

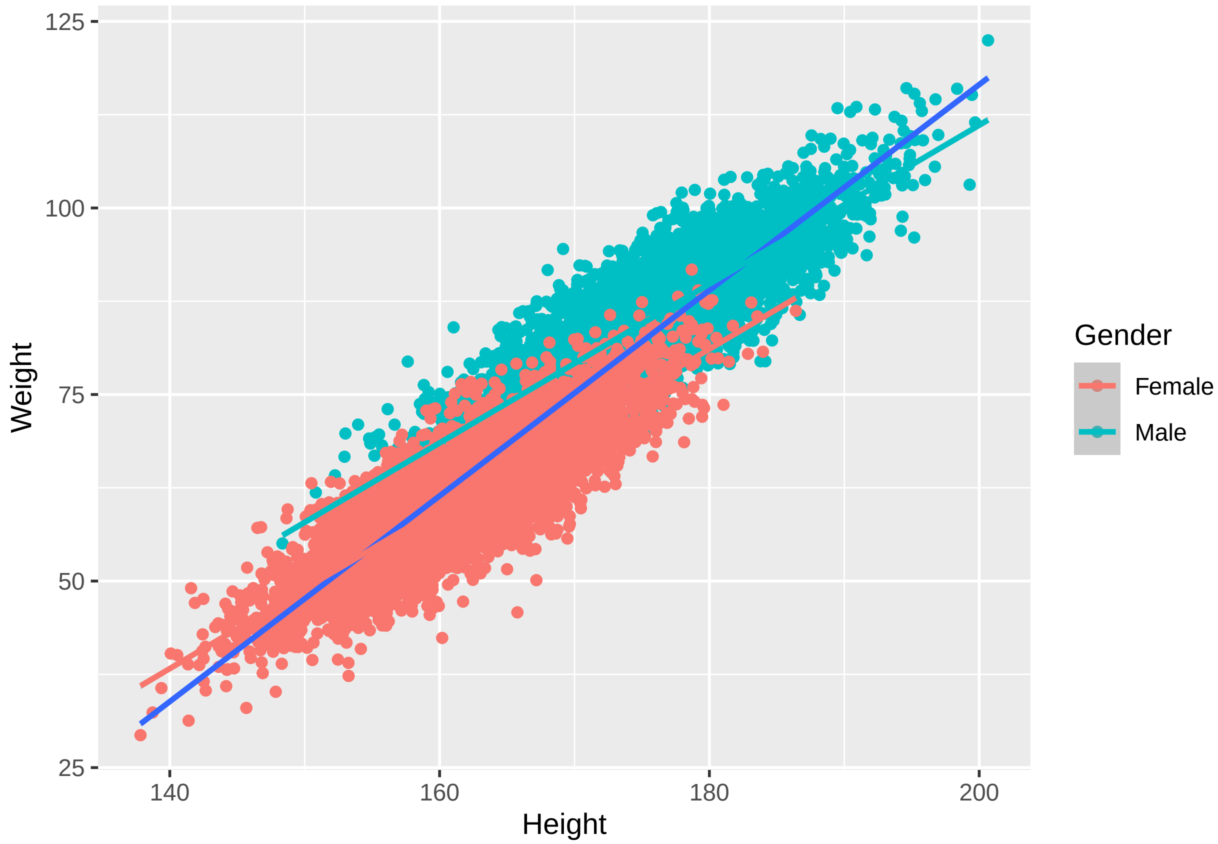
```
## Rows: 10000 Columns: 3
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (1): Gender
## dbl (2): Height, Weight
##
## i Use 'spec()' to retrieve the full column specification for this data.
## i Specify the column types or set 'show_col_types = FALSE' to quiet this message.
```

Linear Regression for Weights and Heights

```r
data <- data %>%
  mutate(Height = Height * 2.54,
         Weight = Weight * 0.453592)
```

```r
ggplot(data) +
  geom_point(aes(x = Height, y = Weight, color = Gender))+
  geom_smooth(aes(x = Height, y = Weight), method = lm)+
     geom_smooth(aes(x = Height, y = Weight, color = Gender), method = lm)
```

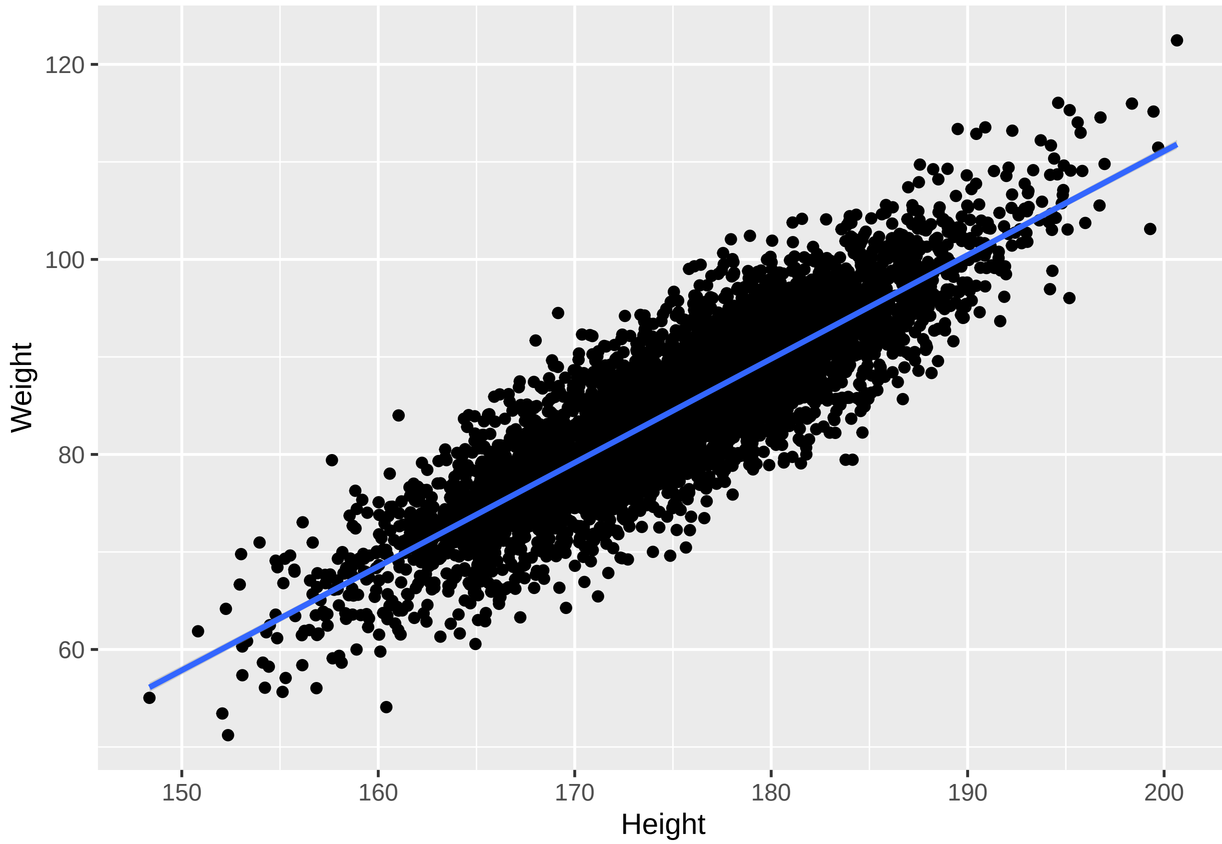
```
## 'geom_smooth()' using formula = 'y ~ x'
## 'geom_smooth()' using formula = 'y ~ x'
```



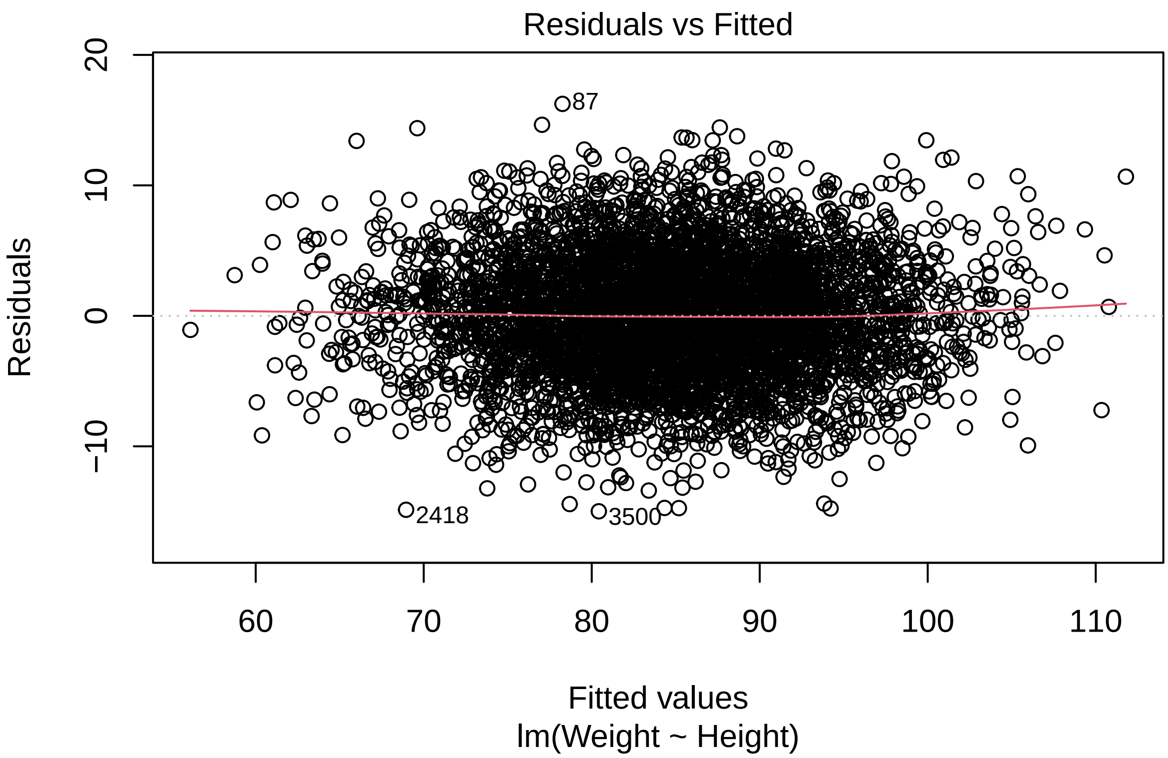

```
male <- data %>% filter(Gender == 'Male')
```

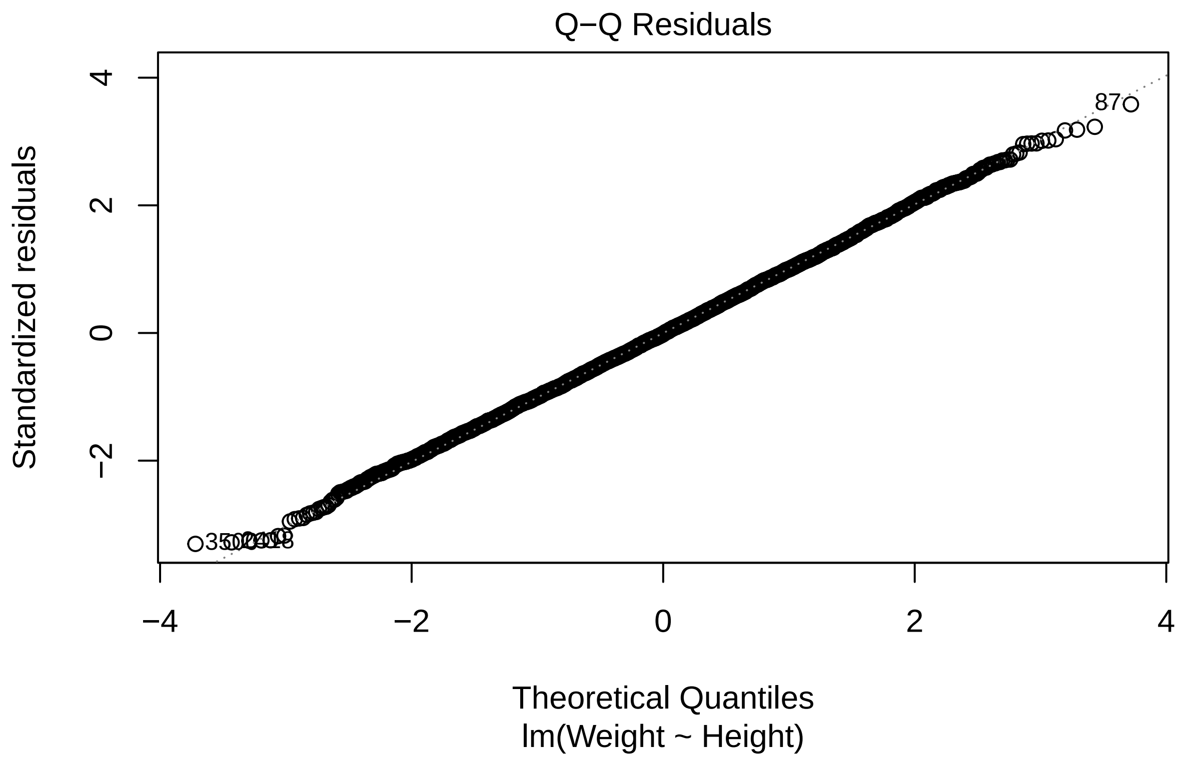
```
ggplot(male) +
  geom_point(aes(x = Height, y = Weight))+
  geom_smooth(aes(x = Height, y = Weight), method = lm)
```

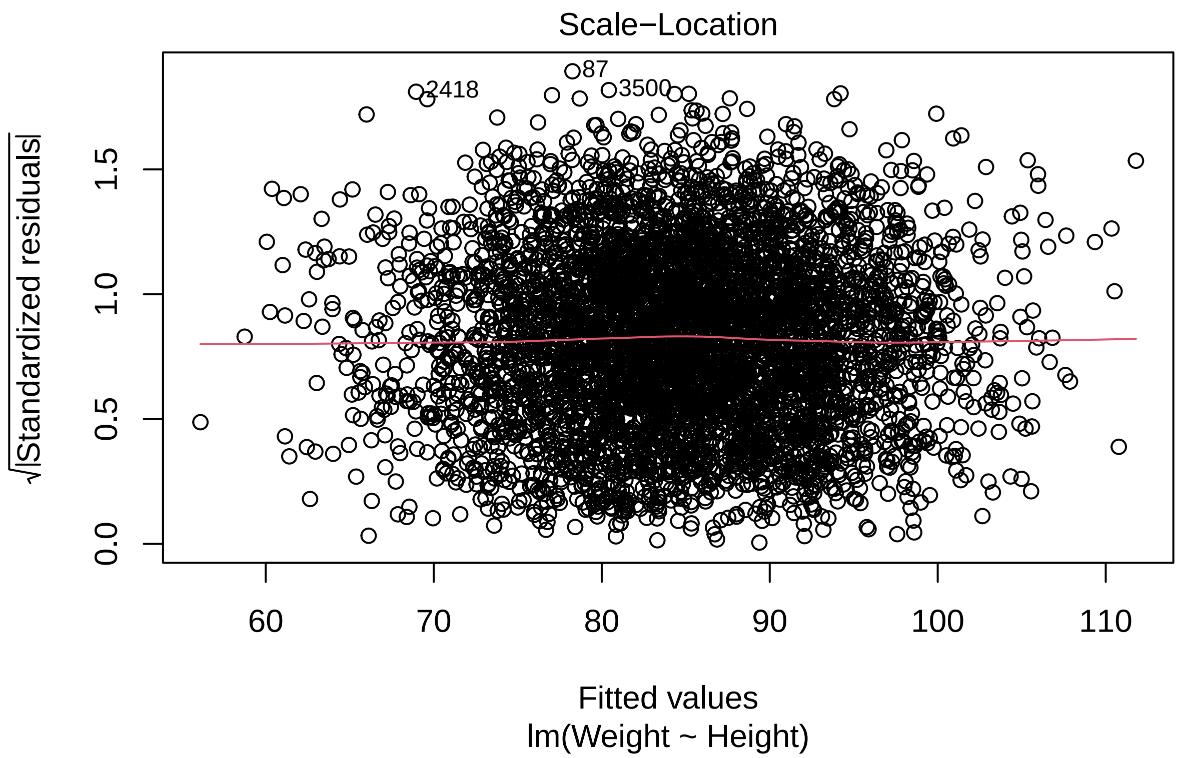
```
## `geom_smooth()` using formula = 'y ~ x'
```



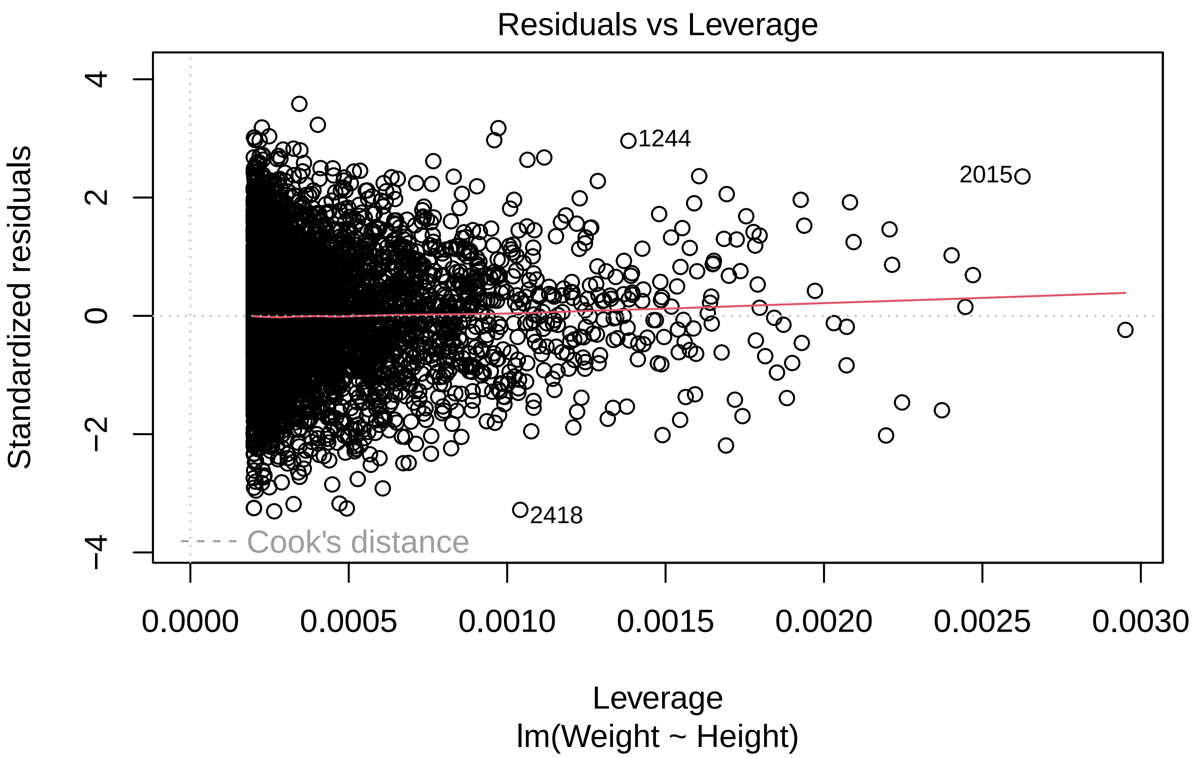

```r
fit_males <- lm(Weight ~ Height, data = male)
plot(fit_males)
```


Residuals vs Fitted
87
2418
3500
Residuals
Fitted values
lm(Weight ~ Height)
20
10
0
−10
60
70
80
90
100
110


Q−Q Residuals
Standardized residuals
Theoretical Quantiles
lm(Weight ~ Height)
87
−4
−2
0
2
4
−2
0
2
4



Scale−Location
87
2418
3500
√|Standardized residuals|
0.0
0.5
1.0
1.5
60
70
80
90
100
110
Fitted values
lm(Weight ~ Height)



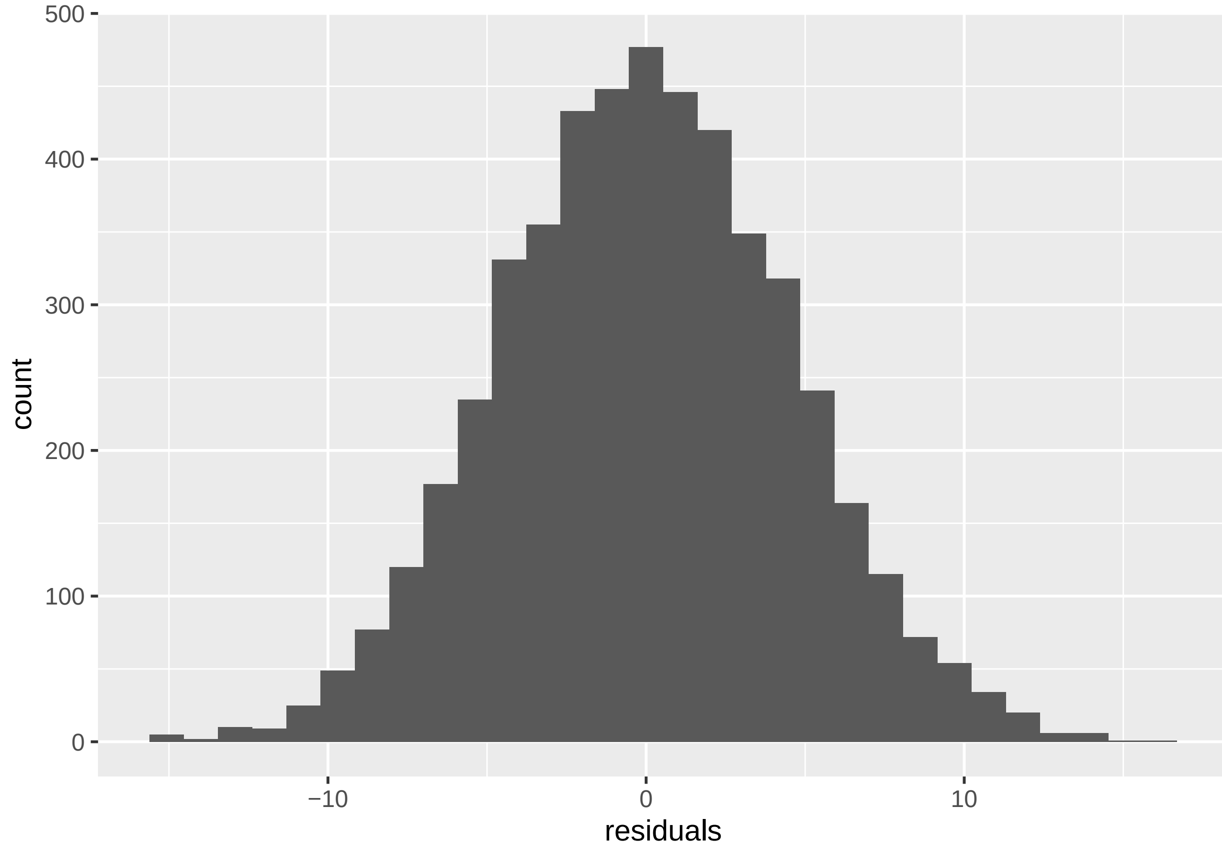

```r
summary(fit_males)
```

```
##
## Call:
## lm(formula = Weight ~ Height, data = male)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -14.9791  -3.0915  -0.0677   3.0740  16.2445
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -1.018e+02  1.547e+00  -65.82   <2e-16 ***
## Height       1.065e+00  8.817e-03  120.75   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.534 on 4998 degrees of freedom
## Multiple R-squared:  0.7447, Adjusted R-squared:  0.7447
## F-statistic: 1.458e+04 on 1 and 4998 DF,  p-value: < 2.2e-16
```

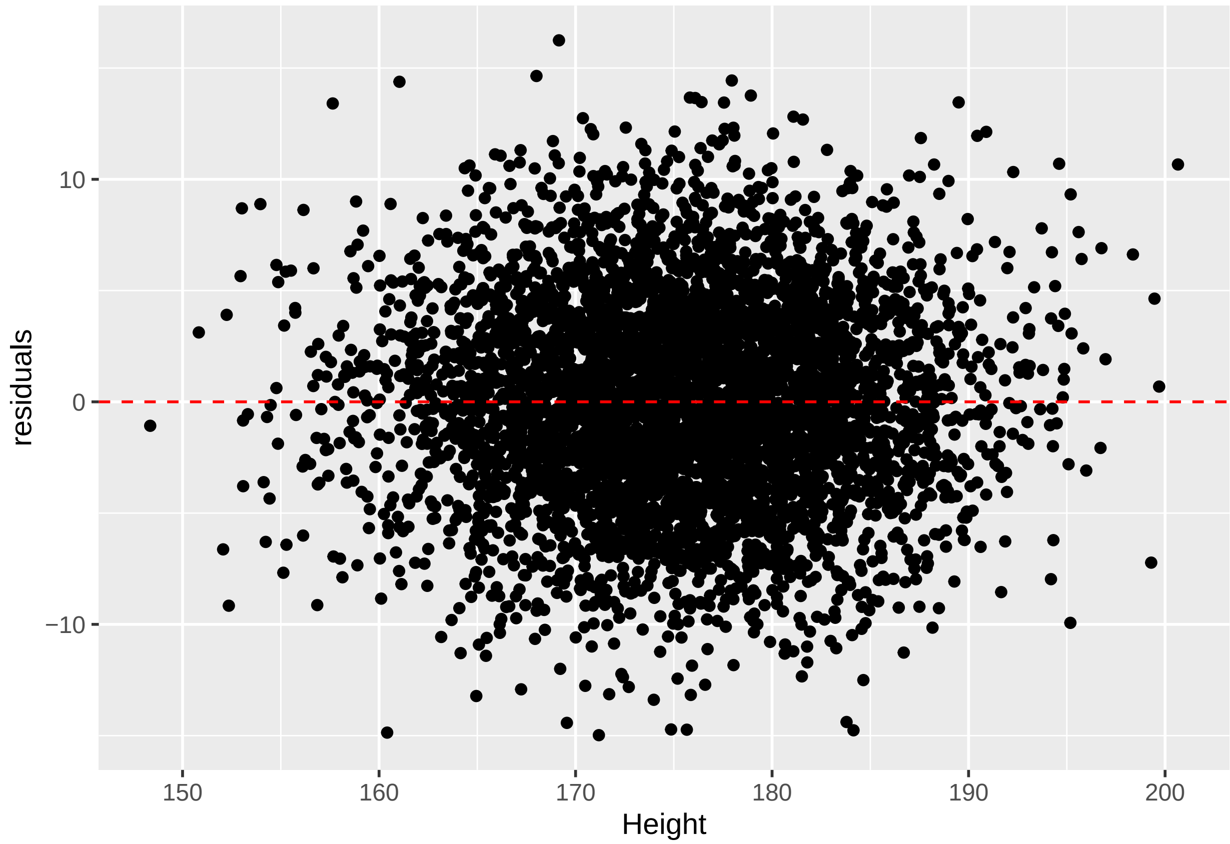
```r
male$residuals <- residuals(fit_males)
```

```r
ggplot(male) +
  geom_histogram(aes(x = residuals))
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```




```r
ggplot(male) +
  geom_point(aes(x=Height, y=residuals))+
  geom_hline(yintercept = 0, color = 'red', linetype = 'dashed')
```




```r
cor(male$Height, male$Weight)
```

```
## [1] 0.8629788
```

A male is 190 cm tall, now we can predict their weight

```r
summary(fit_males)$coefficients[1] + summary(fit_males)$coefficients[2] * 190
```

```
## [1] 100.4528
```

Multiple Regression

```r
library(openintro)
```

```
## Warning: pakke 'openintro' blev bygget under R version 4.4.2
```

```
## Indlæser krævet pakke: airports
```

```
## Warning: pakke 'airports' blev bygget under R version 4.4.2
```

```
## Indlæser krævet pakke: cherryblossom
```

```
## Warning: pakke 'cherryblossom' blev bygget under R version 4.4.2

## Indlæser krævet pakke: usdata

## Warning: pakke 'usdata' blev bygget under R version 4.4.2
```

```r
mariokart
```

```
## # A tibble: 143 x 12
##          id duration n_bids cond  start_pr ship_pr total_pr ship_sp seller_rate
##       <dbl>    <int>  <int> <fct>    <dbl>   <dbl>    <dbl> <fct>         <int>
##  1  1.50e11        3     20 new       0.99    4        51.6 standa~        1580
##  2  2.60e11        7     13 used      0.99    3.99     37.0 firstC~         365
##  3  3.20e11        3     16 new       0.99    3.5      45.5 firstC~         998
##  4  2.80e11        3     18 new       0.99    0        44   standa~           7
##  5  1.70e11        1     20 new       0.01    0        71   media           820
##  6  3.60e11        3     19 new       0.99    4        45   standa~      270144
##  7  1.20e11        1     13 used      0.01    0        37.0 standa~        7284
##  8  3.00e11        1     15 new       1       2.99     54.0 upsGro~        4858
##  9  2.00e11        3     29 used      0.99    4        47   priori~          27
## 10  3.30e11        7      8 used     20.0     4        50   firstC~         201
## # i 133 more rows
## # i 3 more variables: stock_photo <fct>, wheels <int>, title <fct>
```

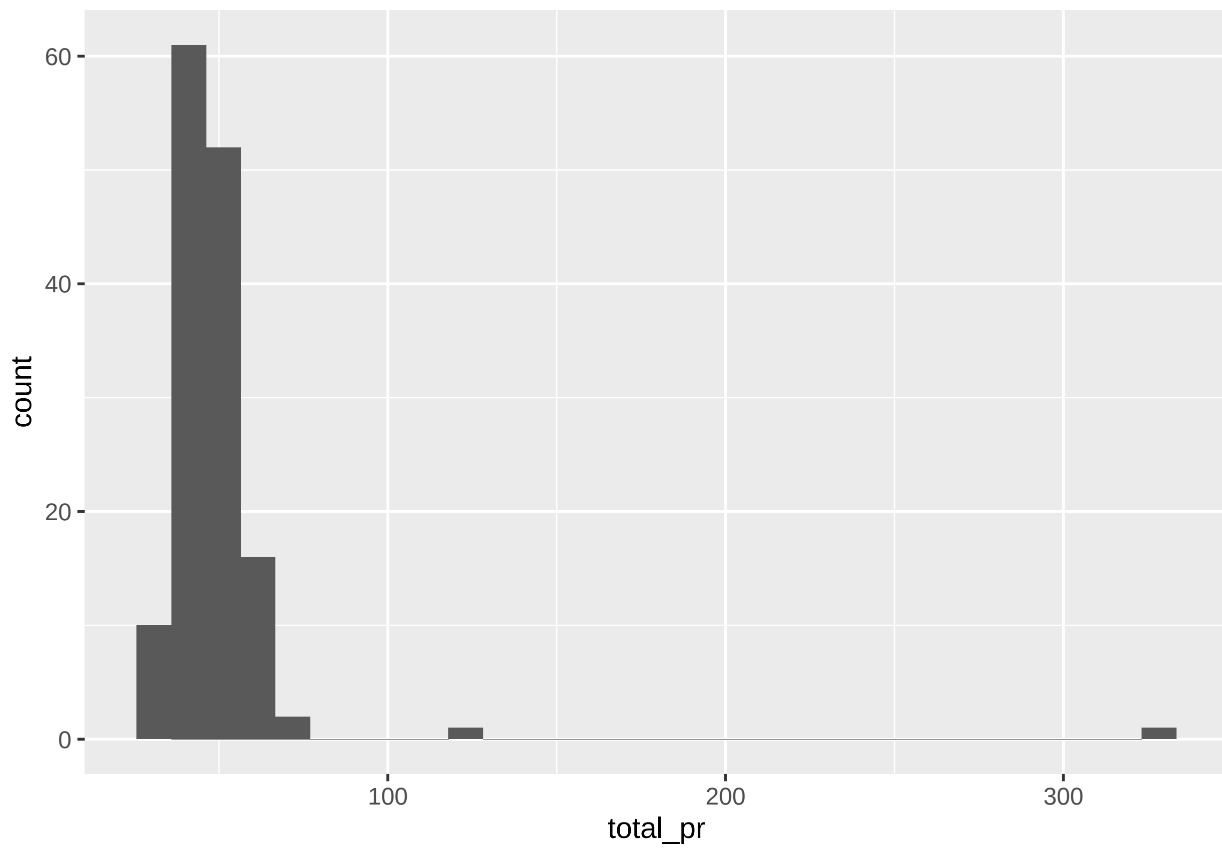
```r
colnames(mariokart)
```

```
##  [1] "id"          "duration"    "n_bids"      "cond"        "start_pr"
##  [6] "ship_pr"     "total_pr"    "ship_sp"     "seller_rate" "stock_photo"
## [11] "wheels"      "title"
```

"id" - Not usefull "duration" - possible
"n_bids" - Possible
"cond" - usefull
"start_pr" - Possible
"ship_pr" - usefull
"total_pr" - TO PREDICT
"ship_sp" - Not usefull
"seller_rate" - usefull "stock_photo" - Possible "wheels" - Usefull
"title" - Not usefull

```r
ggplot(mariokart) +
  geom_histogram(aes(x = total_pr))
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



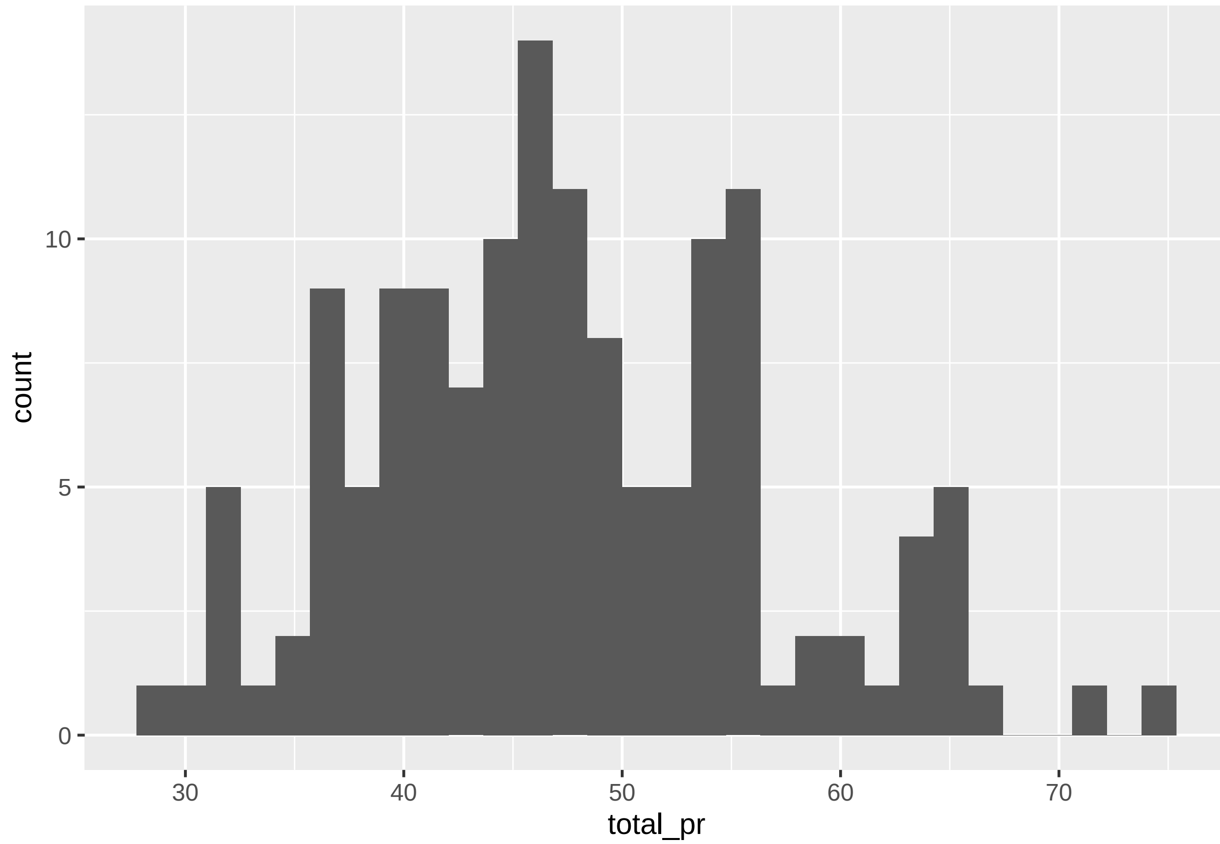

```
mariokart %>% arrange(desc(total_pr))
```

```
## # A tibble: 143 x 12
##             id duration n_bids cond  start_pr ship_pr total_pr ship_sp seller_rate
##          <dbl>    <int>  <int> <fct>    <dbl>   <dbl>    <dbl> <fct>         <int>
##  1    1.10e11        7     22 used      1       25.5     327.  parcel          115
##  2    1.30e11        3     27 used      6.95    4        118.  parcel           41
##  3    3.50e11        1      3 new      70.0     0         75   standa~      118345
##  4    1.70e11        1     20 new       0.01    0         71   media           820
##  5    4.00e11        1      1 new      55.0    11.4       66.4 upsGro~      118345
##  6    1.60e11        7      2 used     55       9.02      65.0 parcel           25
##  7    3.50e11        1      1 new      65.0     0         65.0 standa~      118345
##  8    4.00e11        1      1 new      65.0     0         65.0 standa~      118345
##  9    3.90e11        1      1 new      65.0     0         65.0 standa~      118345
## 10    3.60e11        7     12 used      0.99    4         64.5 standa~         991
## # i 133 more rows
## # i 3 more variables: stock_photo <fct>, wheels <int>, title <fct>
```

```
mk <- mariokart %>% filter(total_pr < 100)
```

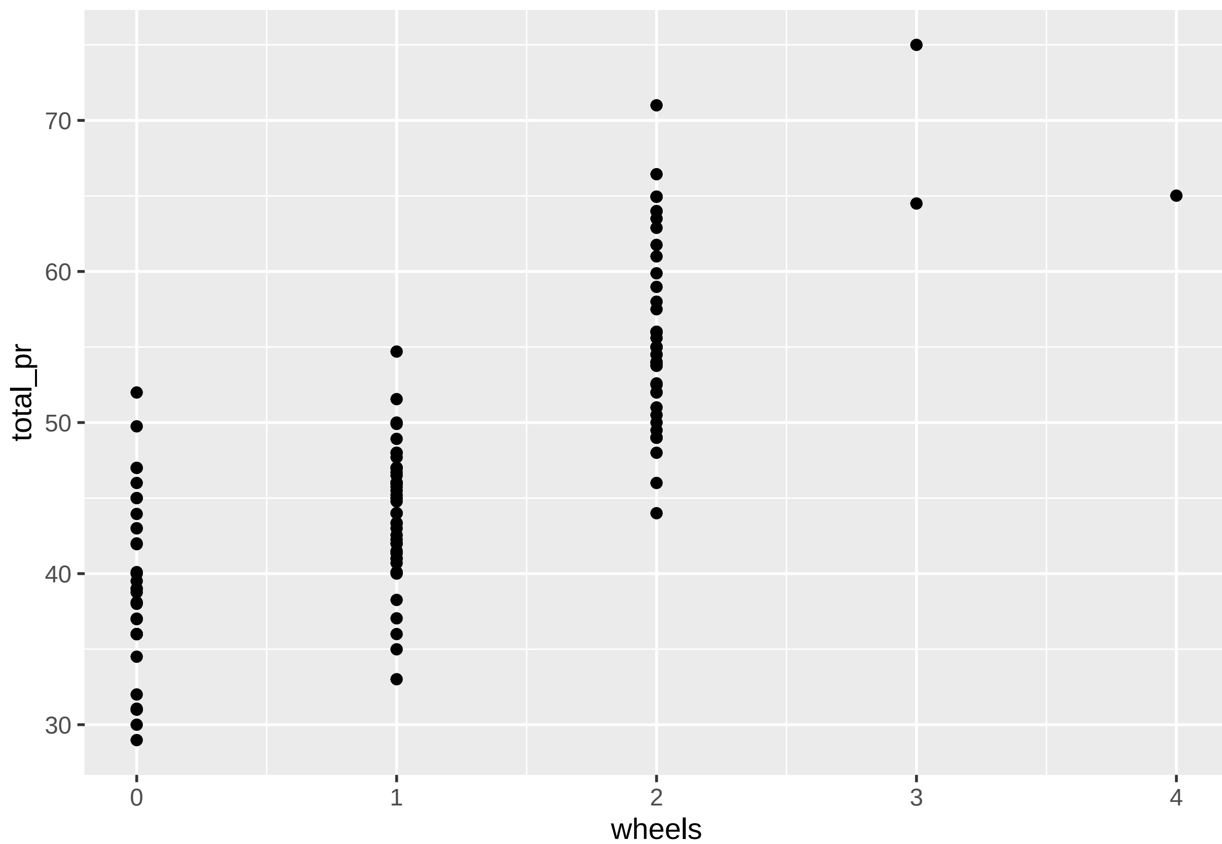
```
ggplot(mk) +
  geom_histogram(aes(x = total_pr))
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



"duration" - possible
"n__bids" - Possible
"cond" - usefull
"start__pr" - Possible
"ship__pr" - usefull
"seller__rate" - usefull "stock__photo" - Possible "wheels" - Usefull

```
ggplot(mk) +
  geom_point(aes(x = wheels, y = total_pr))
```



R^2 Backward Elimination

```r
fit <- lm(total_pr ~ duration +
            n_bids +
            cond +
            start_pr +
            #ship_pr +
            seller_rate +
            #stock_photo +
            wheels,
          data = mk)
summary(fit)
```

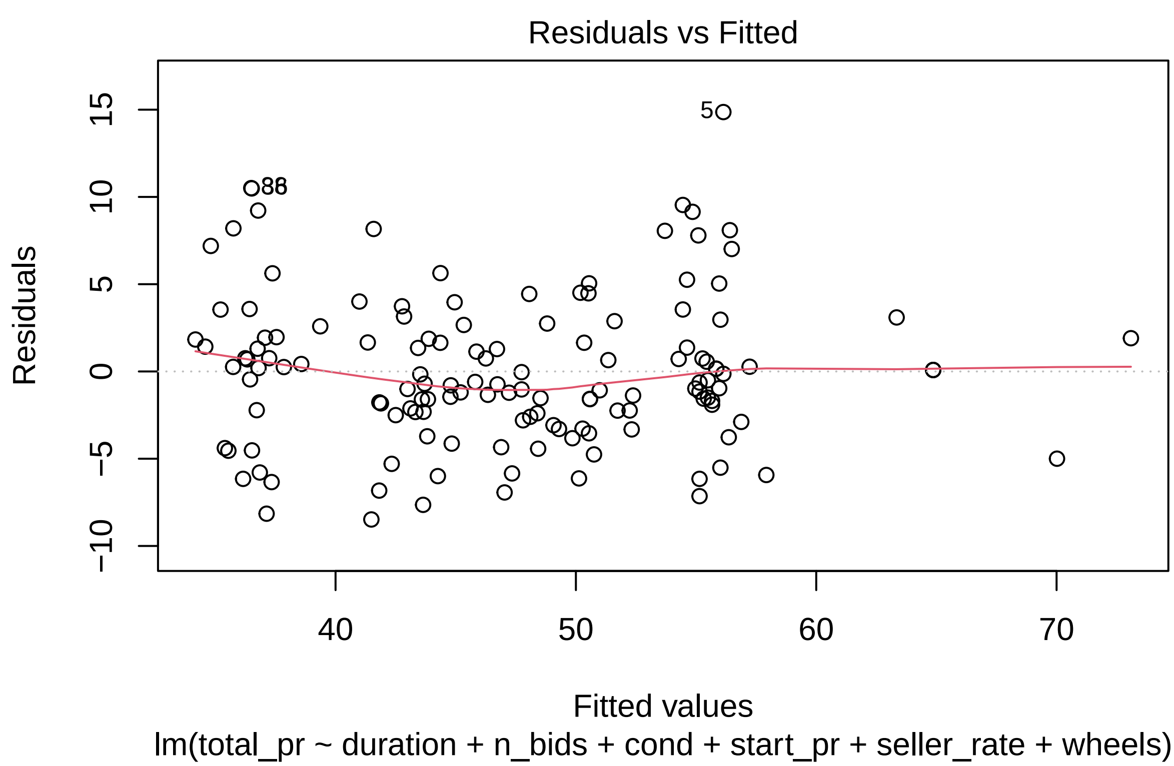
```
##
## Call:
## lm(formula = total_pr ~ duration + n_bids + cond + start_pr +
##     seller_rate + wheels, data = mk)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -8.4814 -2.5041 -0.5024  1.9335 14.8630
##
## Coefficients:
```

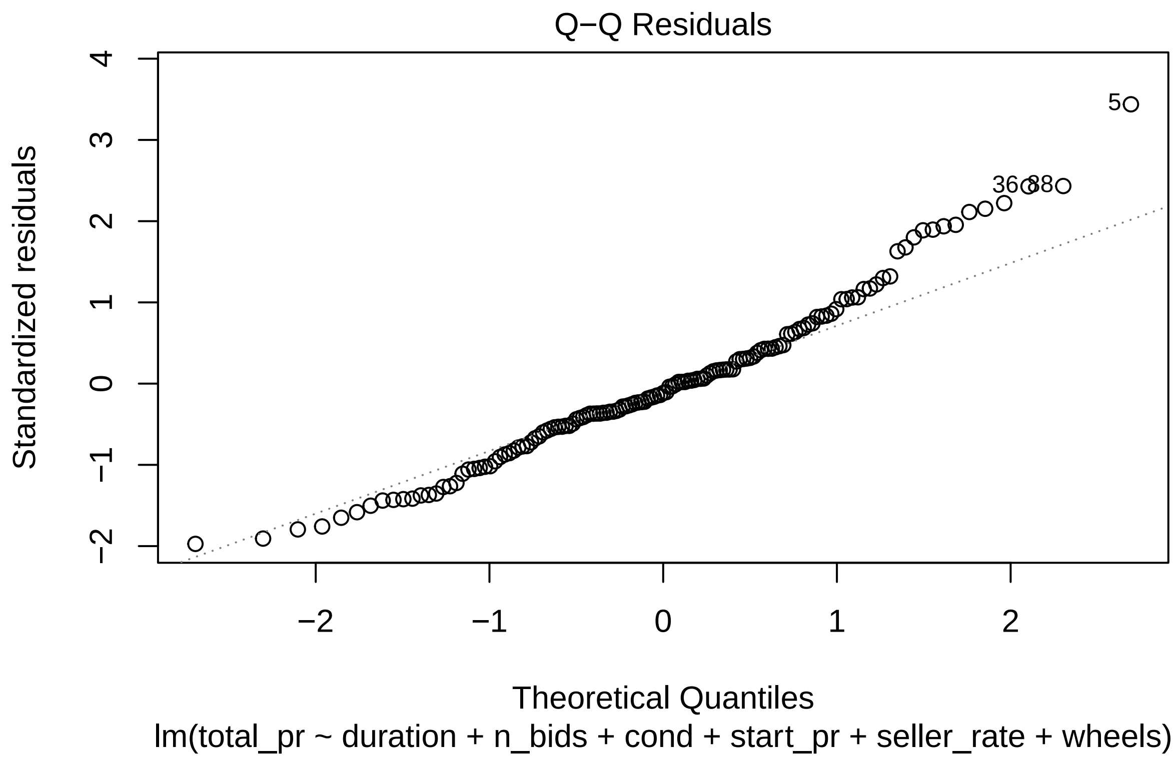
```
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)  3.860e+01  1.807e+00  21.360  < 2e-16 ***
## duration    -1.891e-01  1.733e-01  -1.091   0.2772
## n_bids       1.737e-01  8.575e-02   2.025   0.0448 *
## condused    -4.476e+00  9.310e-01  -4.808 4.04e-06 ***
## start_pr     1.527e-01  3.519e-02   4.340 2.78e-05 ***
## seller_rate  1.796e-05  8.066e-06   2.227   0.0276 *
## wheels       7.117e+00  5.312e-01  13.397  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.387 on 134 degrees of freedom
## Multiple R-squared:  0.7782, Adjusted R-squared:  0.7683
## F-statistic: 78.36 on 6 and 134 DF,  p-value: < 2.2e-16
```

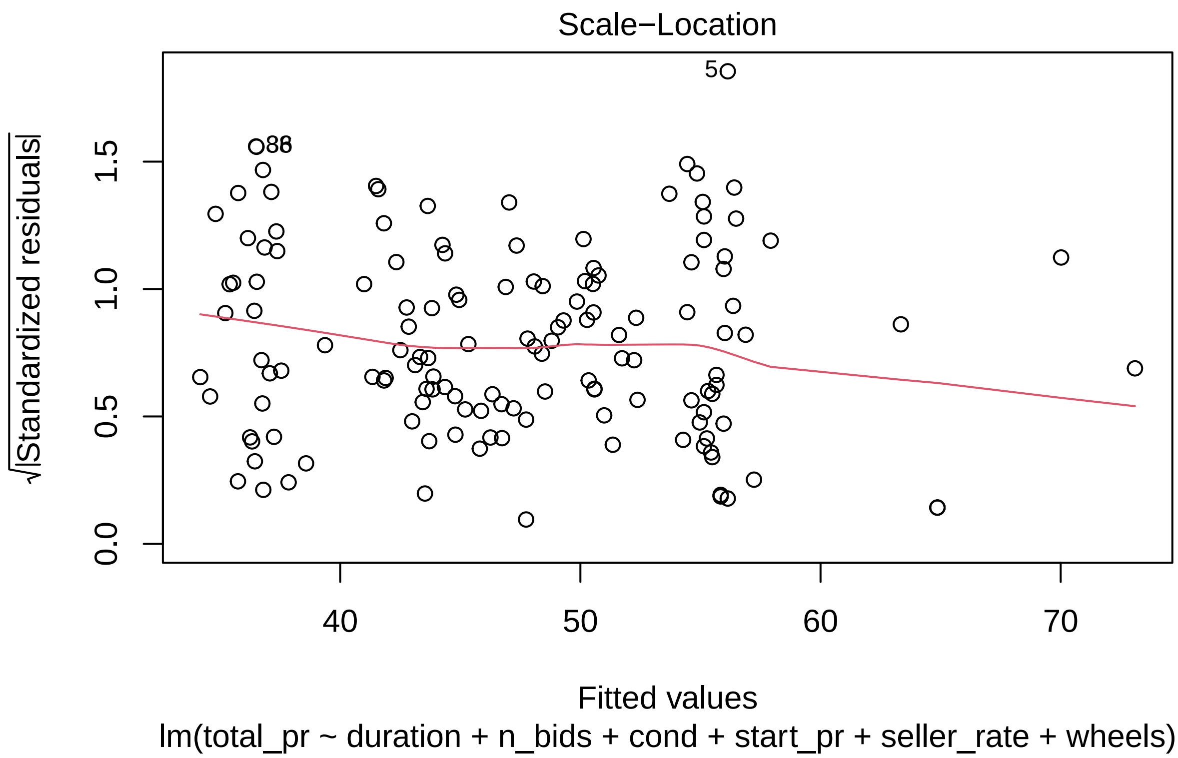
Iteration 1

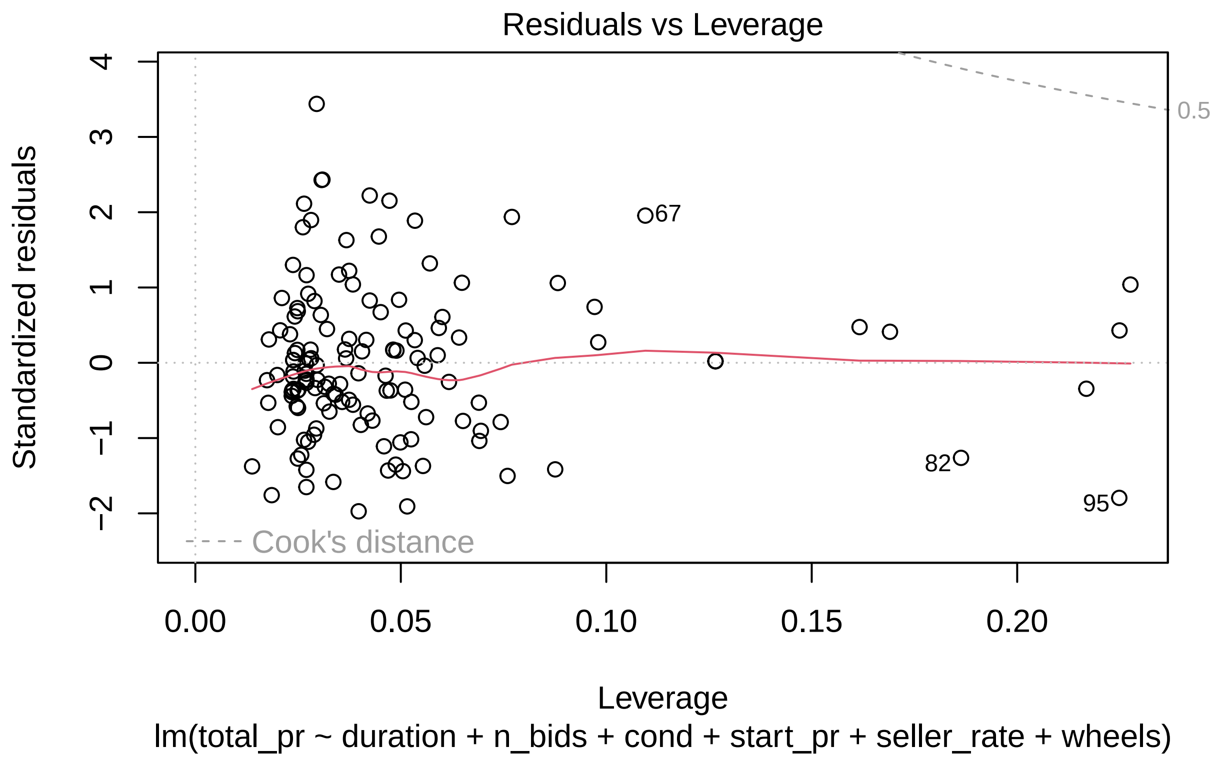
All - 0.7671 ### Iteration 2 No stock photo - 0.7678 ### Iteration 3 No ship_pr - 0.7683

```
plot(fit)
```




Q−Q Residuals
Standardized residuals
Theoretical Quantiles
lm(total_pr ~ duration + n_bids + cond + start_pr + seller_rate + wheels)
5
36
38
−2
−1
0
1
2
3
4


Scale−Location
5
88
√|Standardized residuals|
0.0
0.5
1.0
1.5
40
50
60
70
Fitted values
lm(total_pr ~ duration + n_bids + cond + start_pr + seller_rate + wheels)



total_pr = 3.860e+01 + duration * -1.891e-01
+ n_bids * 1.737e-01
+ condused * -4.476e+00
+ start_pr * 1.527e-01
+ seller_rate * 1.796e-05
+ wheels * 7.117e+00

Exercise

airq402.txt contains information about airfares and passengers for the U.S. Domestic Routes for 4th quarter of 2002. Ryanair wants to break into the U.S. market with a new route in between Casper, Nebraska and San Francisco, California. Currently there are no commercial flights from Casper, and due to that the city is not included in the database. The distance in between two cities is 1200 miles, and is expected to have approximately 100 passengers per week.

```
flights <- readr::read_delim('airq402.txt', delim = ',')
```

```
## Rows: 1000 Columns: 11
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (4): City1, City2, Market Leading Airline, Low Price Airline
## dbl (7): Average Fare, Distance, Average Weekly Passengers, Market Share MLA...
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```r
colnames(flights)
```

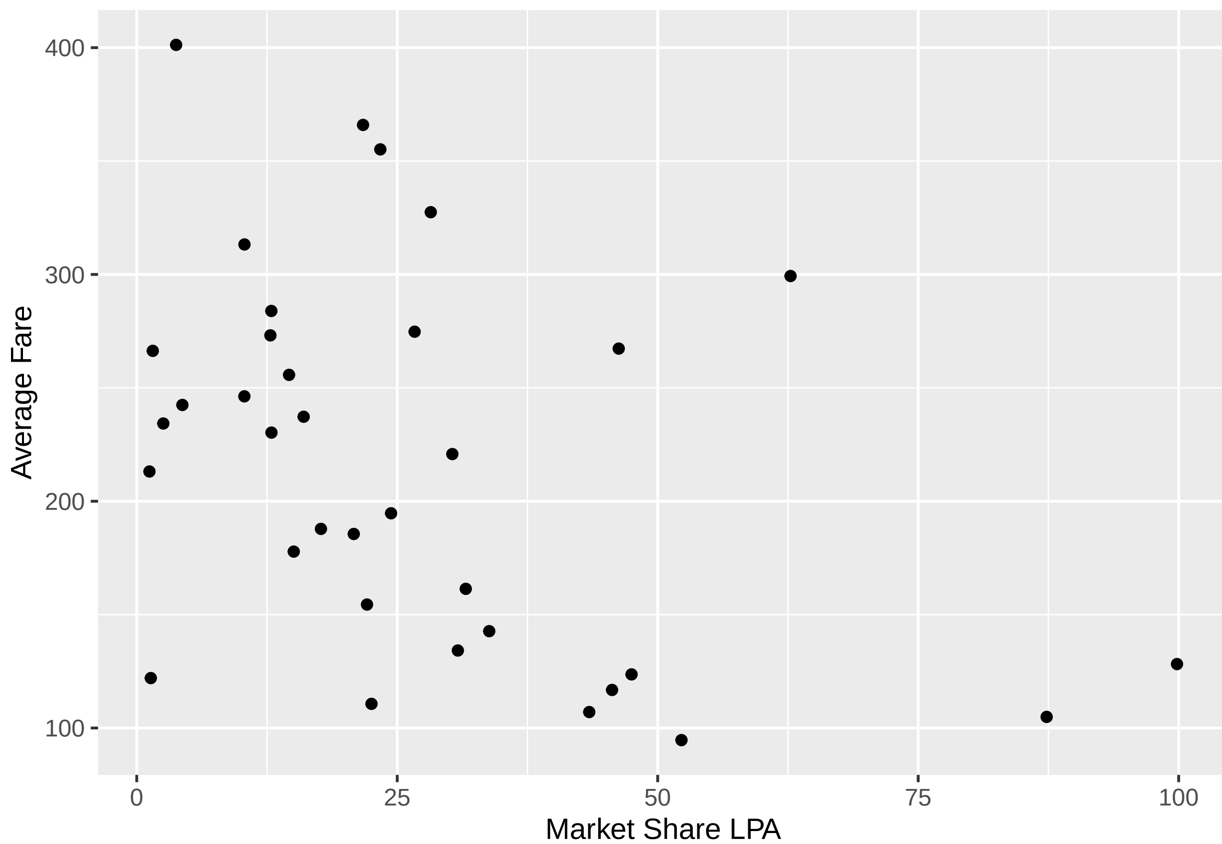
```
##  [1] "City1"                      "City2"
##  [3] "Average Fare"               "Distance"
##  [5] "Average Weekly Passengers" "Market Leading Airline"
##  [7] "Market Share MLA"           "Average Fare MLA"
##  [9] "Low Price Airline"          "Market Share LPA"
## [11] "Average Fare LPA"
```

```r
flights_SFO <- flights %>%
  filter(City1 == 'SFO' | City2 == 'SFO')
flights_SFO
```

```
## # A tibble: 35 x 11
##    City1 City2 `Average Fare` Distance `Average Weekly Passengers`
##    <chr> <chr>          <dbl>    <dbl>                       <dbl>
##  1 ATL   SFO             299.     2139                        790.
##  2 BWI   SFO             266.     2457                        414.
##  3 BOS   SFO             355.     2704                       1270
##  4 BUR   SFO             128.      326                        551.
##  5 CLT   SFO             246.     2296                        322.
##  6 ORD   SFO             188.     1854                       2479.
##  7 CLE   SFO             275.     2161                        249.
##  8 CMH   SFO             178.     2121                        218.
##  9 DFW   SFO             284.     1476                        806.
## 10 DEN   SFO             195.      967                       1318.
## # i 25 more rows
## # i 6 more variables: `Market Leading Airline` <chr>, `Market Share MLA` <dbl>,
## #   `Average Fare MLA` <dbl>, `Low Price Airline` <chr>,
## #   `Market Share LPA` <dbl>, `Average Fare LPA` <dbl>
```

"Average Fare" - Response variable "Distance" - YES
"Average Weekly Passengers" - YES "Market Share MLA"
"Market Share LPA"

```r
ggplot(flights_SFO) +
  geom_point(aes(x = `Market Share LPA`, y = `Average Fare`))
```




```r
fit_flights <- lm(`Average Fare` ~ Distance +
                    `Market Share MLA`+
                    `Average Weekly Passengers` #+
                    #`Market Share LPA`
                    , data = flights_SFO)
summary(fit_flights)
```

```
##
## Call:
## lm(formula = `Average Fare` ~ Distance + `Market Share MLA` +
##     `Average Weekly Passengers`, data = flights_SFO)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -85.077 -28.185  -3.264  24.759  96.167
##
## Coefficients:
##                              Estimate Std. Error t value Pr(>|t|)
## (Intercept)                 -28.36317   38.56109  -0.736   0.4675
## Distance                      0.09822    0.01015   9.677 6.98e-11 ***
## `Market Share MLA`            1.21379    0.45106   2.691   0.0114 *
## `Average Weekly Passengers`   0.01929    0.01015   1.900   0.0668 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
```

```
## Residual standard error: 42.1 on 31 degrees of freedom
## Multiple R-squared:  0.7642, Adjusted R-squared:  0.7414
## F-statistic:  33.5 on 3 and 31 DF,  p-value: 7.528e-10
```

Iteration 1

All - 0.733
Iteration 2 Remove Market Share LPA - 0.7414

Prediction

The distance in between two cities is 1200 miles, and is expected to have approximately 100 passengers per week.

```r
fit_flights$coefficients[1] +
  fit_flights$coefficients[2] * 1200 +
  fit_flights$coefficients[3] * 100
```

```
## (Intercept)
##     210.876
```

Data Conversion

Gather and spread commands

```r
wide_data <- tribble(
  ~country,  ~price2000, ~price2005, ~price2010, ~price2015,
  'Germany', 1001, 2001,1500,1700,
  'Denmark', 1100, 2000,1600,1800,
  'France', 999, 2000, 1550, 1820,
  'Austria', 1010, 2000, 1500, 1700,
  'UK', 1050, 1950, 1600, 1200,
  'Norway', 1150, 2050, 1599, 1600,
  'Poland', 900, 1500, 1000, 1500,
)
```

```r
wide_data %>% gather(year, price, -country)
```

```
## # A tibble: 28 x 3
##    country year      price
##    <chr>   <chr>     <dbl>
##  1 Germany price2000  1001
##  2 Denmark price2000  1100
##  3 France  price2000   999
##  4 Austria price2000  1010
##  5 UK      price2000  1050
##  6 Norway  price2000  1150
##  7 Poland  price2000   900
```

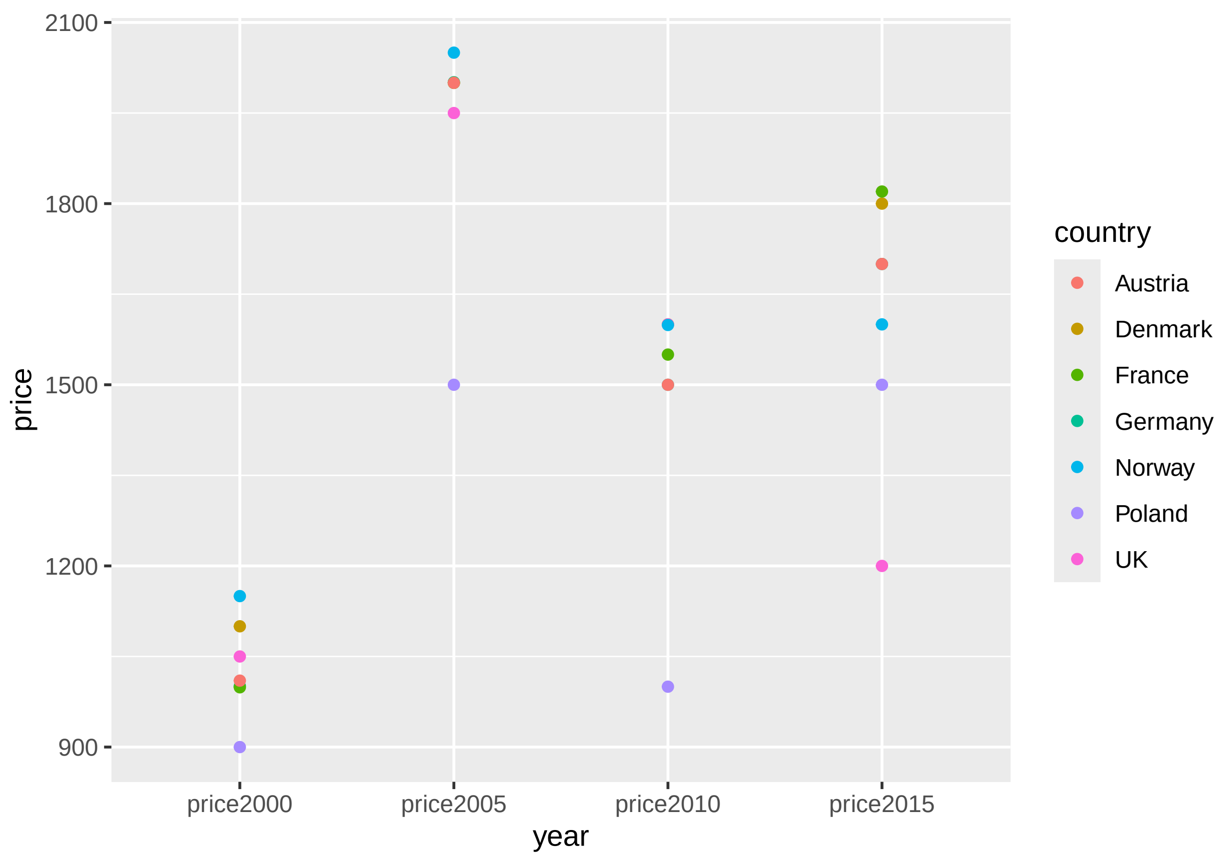
```
##   8 Germany price2005   2001
##   9 Denmark price2005   2000
## 10 France  price2005   2000
## # i 18 more rows
```

```
wide_data %>% gather(year, price, -country) %>% spread(year, price)
```

```
## # A tibble: 7 x 5
##   country price2000 price2005 price2010 price2015
##   <chr>       <dbl>     <dbl>     <dbl>     <dbl>
## 1 Austria      1010      2000      1500      1700
## 2 Denmark      1100      2000      1600      1800
## 3 France        999      2000      1550      1820
## 4 Germany      1001      2001      1500      1700
## 5 Norway       1150      2050      1599      1600
## 6 Poland        900      1500      1000      1500
## 7 UK           1050      1950      1600      1200
```

```
long_data <- wide_data %>% gather(year, price, price2000:price2015)
```

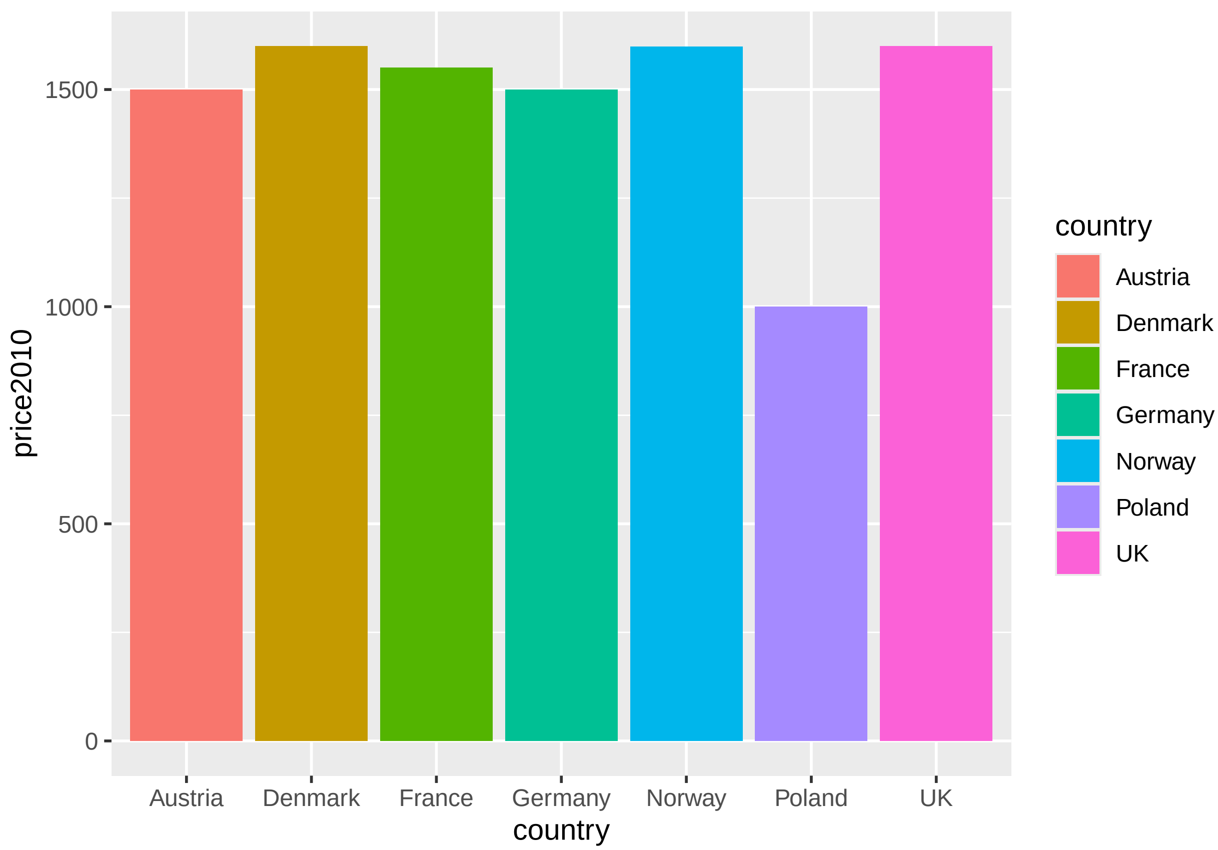
```
ggplot(long_data)+
  geom_point(aes(x = year, y = price, color = country))
```



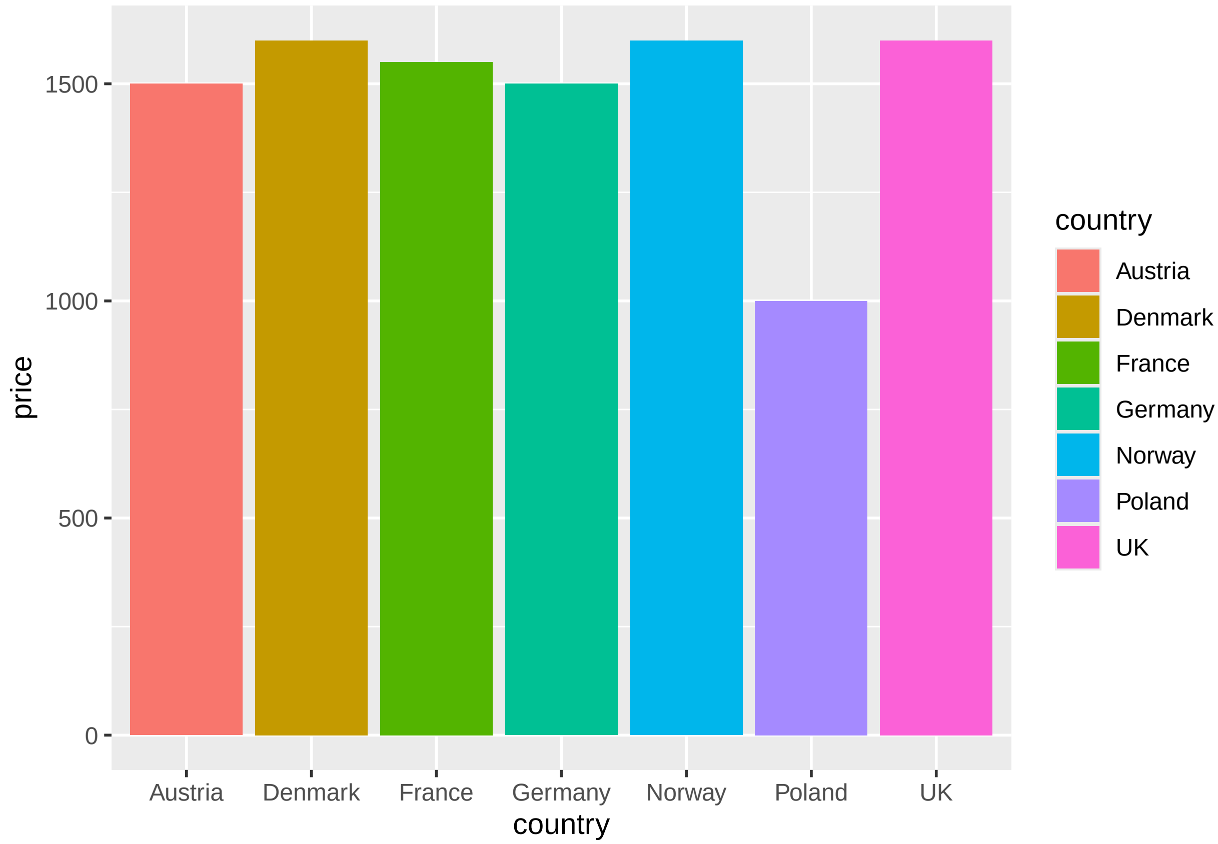
```r
long_data %>% spread(year, price)
```

```
## # A tibble: 7 x 5
##   country price2000 price2005 price2010 price2015
##   <chr>       <dbl>     <dbl>     <dbl>     <dbl>
## 1 Austria      1010      2000      1500      1700
## 2 Denmark      1100      2000      1600      1800
## 3 France        999      2000      1550      1820
## 4 Germany      1001      2001      1500      1700
## 5 Norway       1150      2050      1599      1600
## 6 Poland        900      1500      1000      1500
## 7 UK           1050      1950      1600      1200
```

```r
ggplot(wide_data)+
  geom_bar(aes(x = country, y = price2010, fill = country), stat = 'identity')
```




```r
long_data %>%
  filter(year == 'price2010') %>%
  ggplot()+
  geom_bar(aes(x = country, y = price, fill = country), stat = 'identity')
```



Joins

```
library(gapminder)
```

```
gapminder
```

```
## # A tibble: 1,704 x 6
##    country     continent  year lifeExp      pop gdpPercap
##    <fct>       <fct>     <int>   <dbl>    <int>     <dbl>
##  1 Afghanistan Asia       1952    28.8  8425333      779.
##  2 Afghanistan Asia       1957    30.3  9240934      821.
##  3 Afghanistan Asia       1962    32.0 10267083      853.
##  4 Afghanistan Asia       1967    34.0 11537966      836.
##  5 Afghanistan Asia       1972    36.1 13079460      740.
##  6 Afghanistan Asia       1977    38.4 14880372      786.
##  7 Afghanistan Asia       1982    39.9 12881816      978.
##  8 Afghanistan Asia       1987    40.8 13867957      852.
##  9 Afghanistan Asia       1992    41.7 16317921      649.
## 10 Afghanistan Asia       1997    41.8 22227415      635.
## # i 1,694 more rows
```

```
country_population <- gapminder %>% filter(year == 1987) %>%  select(country, pop)
```

```
country_continent <- gapminder %>% select(country, continent) %>%  unique()
```

```r
country_population %>% left_join(country_continent, by = 'country')
```

```
## # A tibble: 142 x 3
##    country           pop continent
##    <fct>           <int> <fct>
##  1 Afghanistan  13867957 Asia
##  2 Albania       3075321 Europe
##  3 Algeria      23254956 Africa
##  4 Angola        7874230 Africa
##  5 Argentina    31620918 Americas
##  6 Australia    16257249 Oceania
##  7 Austria       7578903 Europe
##  8 Bahrain        454612 Asia
##  9 Bangladesh  103764241 Asia
## 10 Belgium       9870200 Europe
## # i 132 more rows
```

```r
#country_population %>% left_join(country_continent, by = c('CTR' = 'country'))
```

String operations

str_view

```r
x <- c('one', 'lamp', 'table', 'tea', 'mug', 'couch', 'candle', 'spring')
```

```r
str_view(x, 'e')
```

```
## [1] | on<e>
## [3] | tabl<e>
## [4] | t<e>a
## [7] | candl<e>
```

```r
str_view(x, 'c...')
```

```
## [6] | <couc>h
## [7] | <cand>le
```

```r
str_view(x, 'l..p')
```

```
## [2] | <lamp>
```

```r
str_view(x, '.n')
```

```
## [1] | <on>e
## [7] | c<an>dle
## [8] | spr<in>g
```

```r
str_view(x, 't.*')
```

```
## [3] | <table>
## [4] | <tea>
```