Statistics and Data Analysis Assignment

Domen Mohorčič, Larsen Cundrič

09/04/2021

Roller Coaster Dataset Analysis

State hypothesis here...

Basic analysis and plots

```
##
## -- Column specification --------------------------------------------------------
## cols(
##   Name = col_character(),
##   Park = col_character(),
##   City = col_character(),
##   State = col_character(),
##   Country = col_character(),
##   Type = col_character(),
##   Construction = col_character(),
##   Height = col_double(),
##   Speed = col_double(),
##   Length = col_double(),
##   Inversions = col_character(),
##   Numinversions = col_double(),
##   Duration = col_double(),
##   GForce = col_double(),
##   Opened = col_double(),
##   Region = col_character()
## )
```

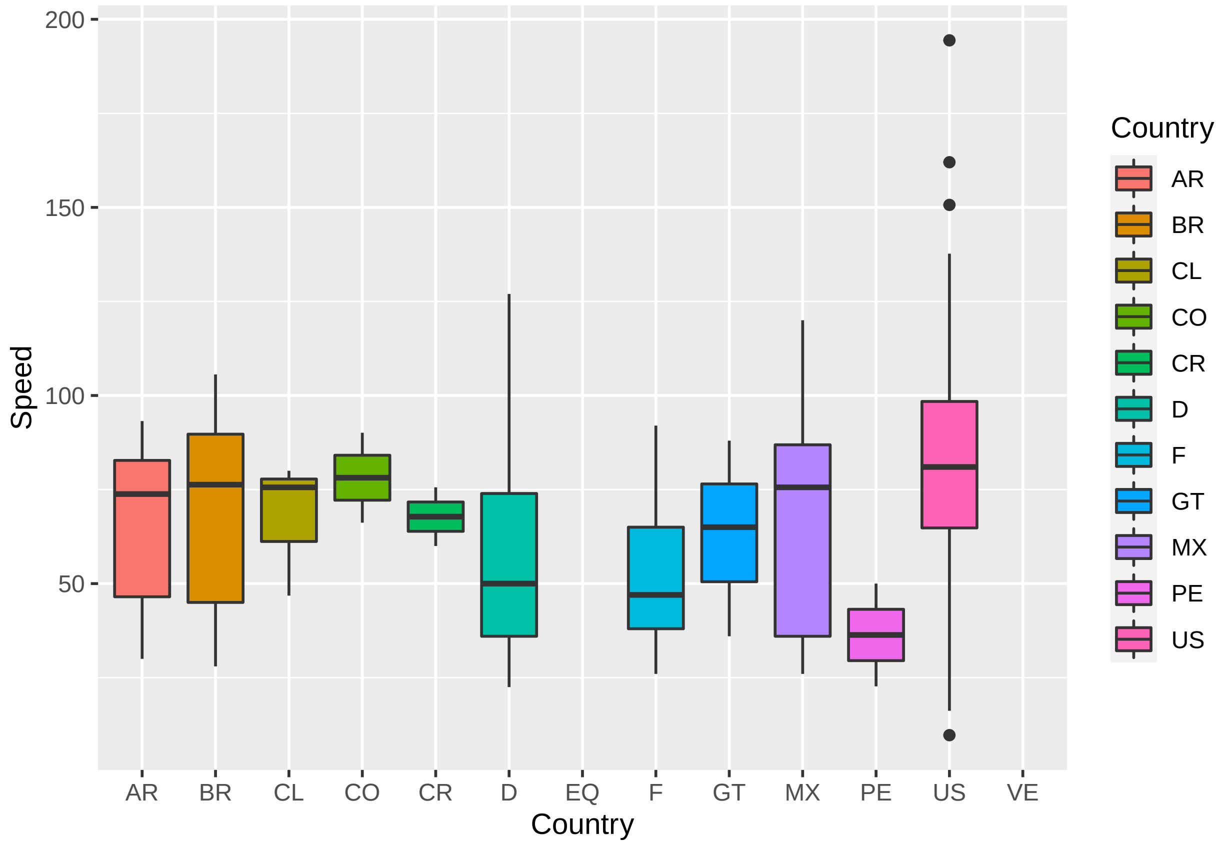
```
# GForce to many missing values..
#
summary(roller_coasters_raw)
```

```
##      Name               Park               City               State          
##  Length:408         Length:408         Length:408         Length:408        
##  Class :character   Class :character   Class :character   Class :character  
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character  
##
##
##
##
```

```
##    Country               Type            Construction            Height
##  Length:408         Length:408         Length:408         Min.   :  2.438
##  Class :character   Class :character   Class :character   1st Qu.:  8.651
##  Mode  :character   Mode  :character   Mode  :character   Median : 18.288
##                                                           Mean   : 23.125
##                                                           3rd Qu.: 33.167
##                                                           Max.   :128.016
##                                                           NA's   :82
##      Speed            Length          Inversions        Numinversions
##  Min.   :  9.72   Min.   :  12.19   Length:408         Min.   : 0.0000
##  1st Qu.: 45.00   1st Qu.: 291.00   Class :character   1st Qu.: 0.0000
##  Median : 68.85   Median : 415.75   Mode  :character   Median : 0.0000
##  Mean   : 69.36   Mean   : 597.04                      Mean   : 0.7843
##  3rd Qu.: 88.95   3rd Qu.: 833.12                      3rd Qu.: 0.0000
##  Max.   :194.40   Max.   :2243.02                      Max.   :10.0000
##  NA's   :138      NA's   :90
##     Duration          GForce          Opened        Region
##  Min.   :  0.3   Min.   :2.100   Min.   :1924   Length:408
##  1st Qu.: 75.0   1st Qu.:3.175   1st Qu.:1991   Class :character
##  Median :108.0   Median :4.500   Median :1999   Mode  :character
##  Mean   :112.5   Mean   :4.115   Mean   :1995
##  3rd Qu.:140.8   3rd Qu.:5.000   3rd Qu.:2004
##  Max.   :300.0   Max.   :6.200   Max.   :2014
##  NA's   :216     NA's   :348     NA's   :28
```

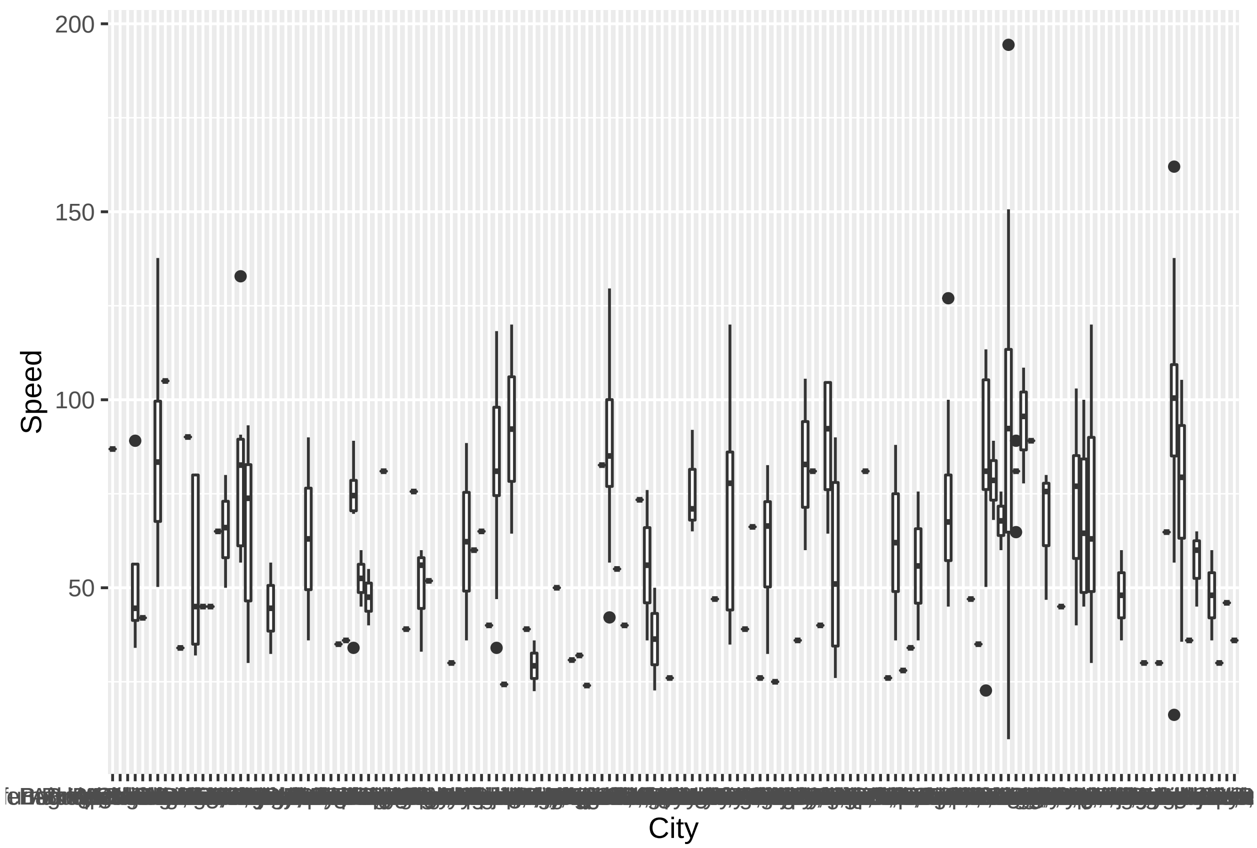
```r
ggplot(data = roller_coasters_raw) +
  geom_boxplot(mapping = aes(x = Country, y = Speed, fill = Country))
```

```
## Warning: Removed 138 rows containing non-finite values (stat_boxplot).
```



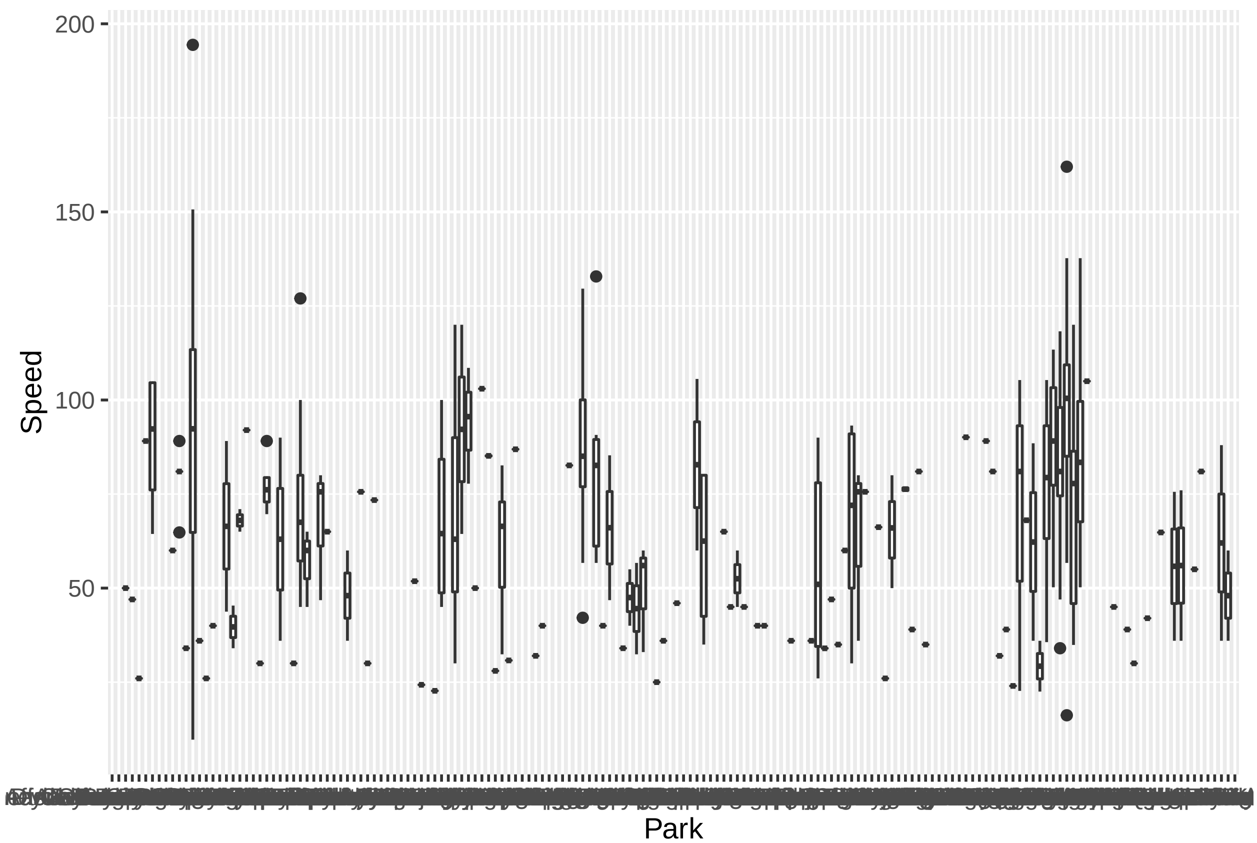

```
ggplot(data = roller_coasters_raw) +
  geom_boxplot(mapping = aes(x = City, y = Speed))
```

```
## Warning: Removed 138 rows containing non-finite values (stat_boxplot).
```



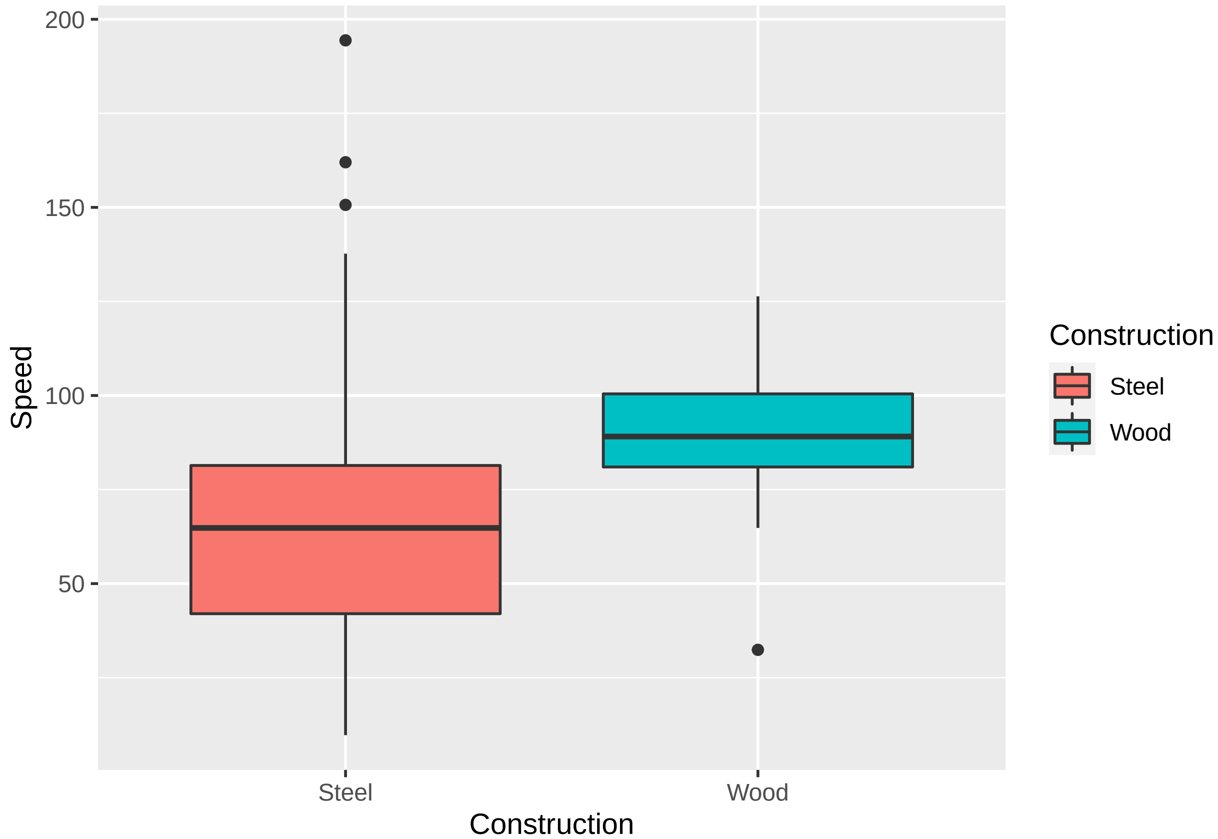

```r
ggplot(data = roller_coasters_raw) +
  geom_boxplot(mapping = aes(x = Park, y = Speed))
```

```
## Warning: Removed 138 rows containing non-finite values (stat_boxplot).
```



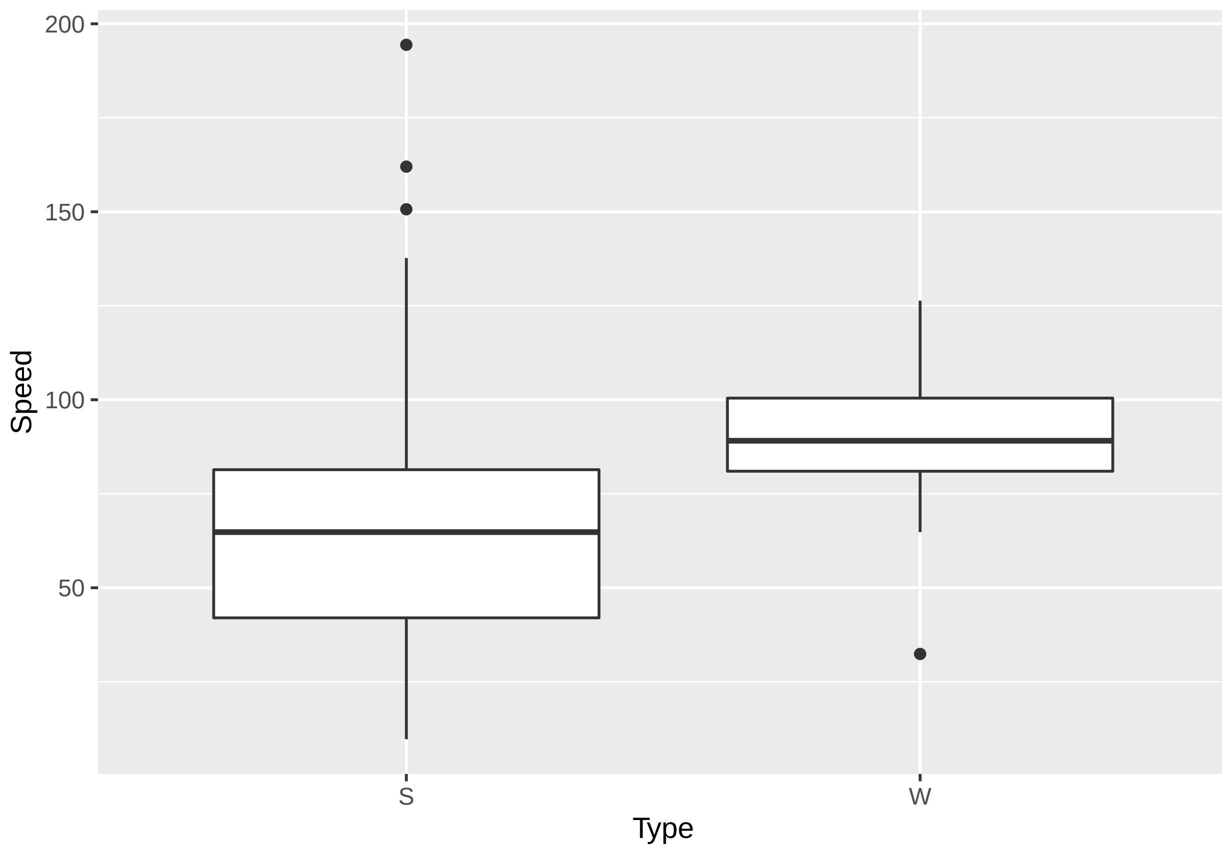

```r
ggplot(data = roller_coasters_raw) +
  geom_boxplot(mapping = aes(x = Construction, y = Speed, fill = Construction))
```

```
## Warning: Removed 138 rows containing non-finite values (stat_boxplot).
```



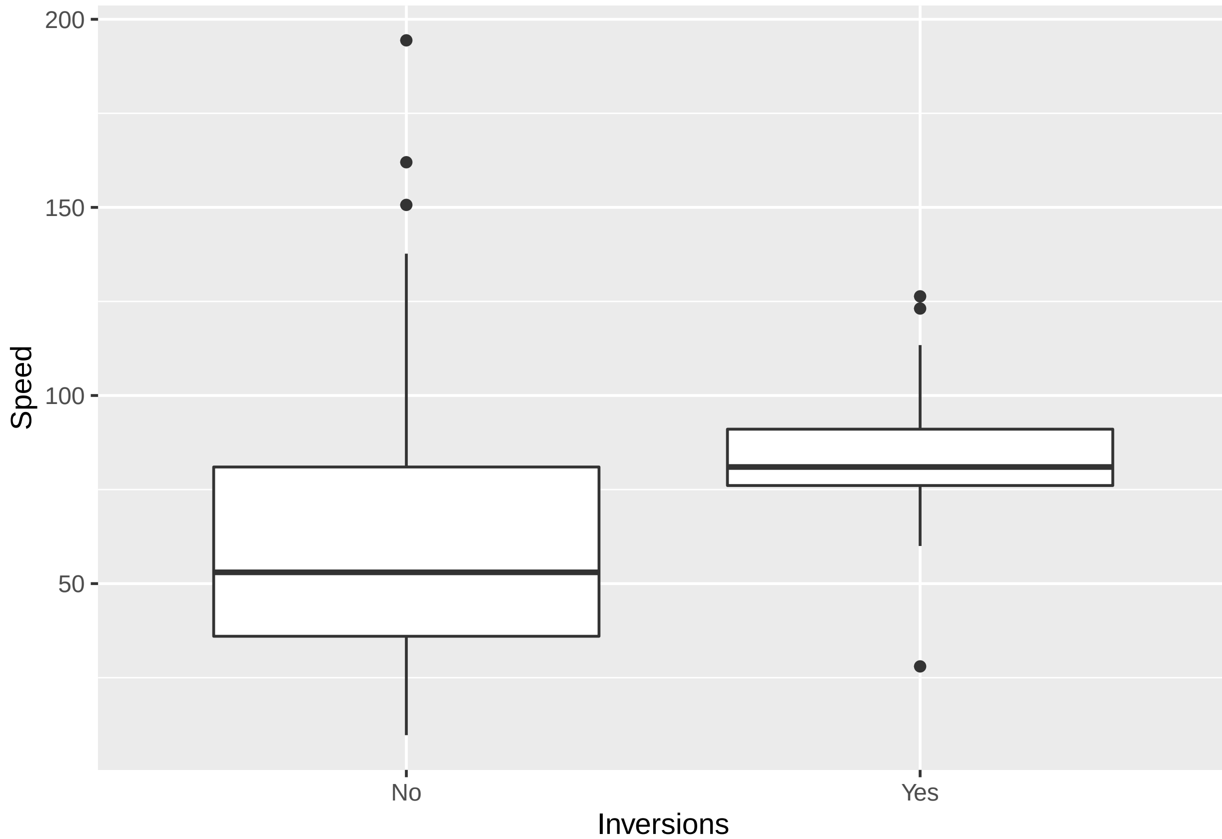

```r
# Type is same as Construction?
ggplot(data = roller_coasters_raw) +
  geom_boxplot(mapping = aes(x = Type, y = Speed))
```

```
## Warning: Removed 138 rows containing non-finite values (stat_boxplot).
```



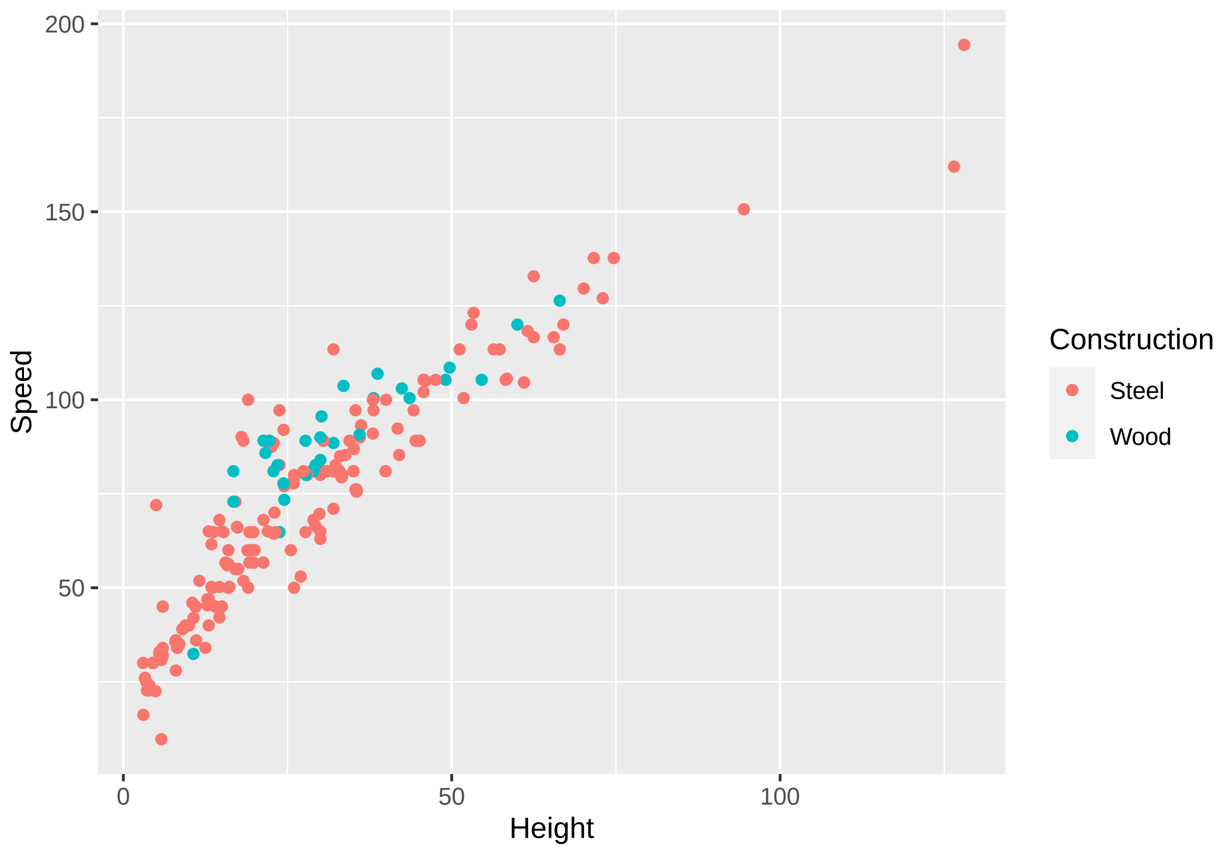

```
ggplot(data = roller_coasters_raw) +
  geom_boxplot(mapping = aes(x = Inversions, y = Speed))
```

```
## Warning: Removed 138 rows containing non-finite values (stat_boxplot).
```



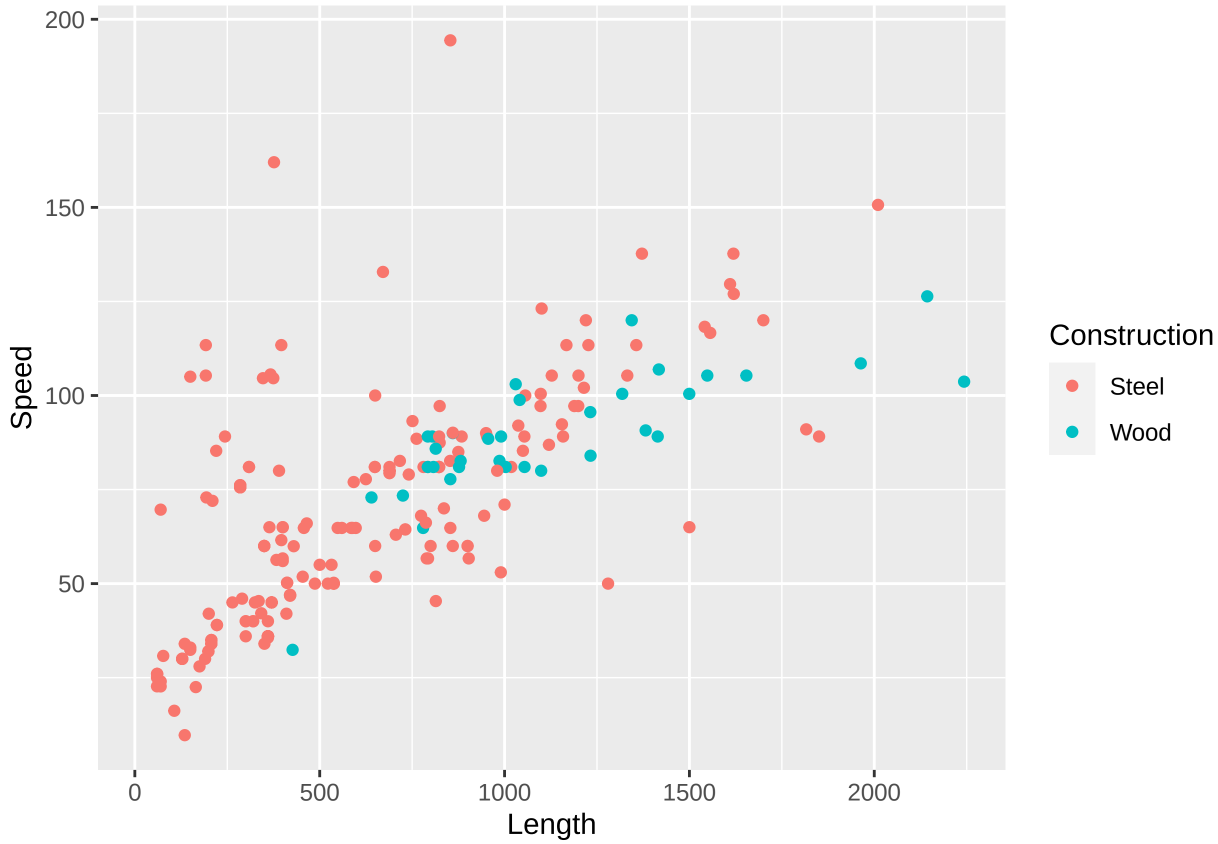

```
roller_coasters_raw %>%
  ggplot() +
    geom_point(aes(x = Height, y = Speed, color = Construction))
```

```
## Warning: Removed 150 rows containing missing values (geom_point).
```



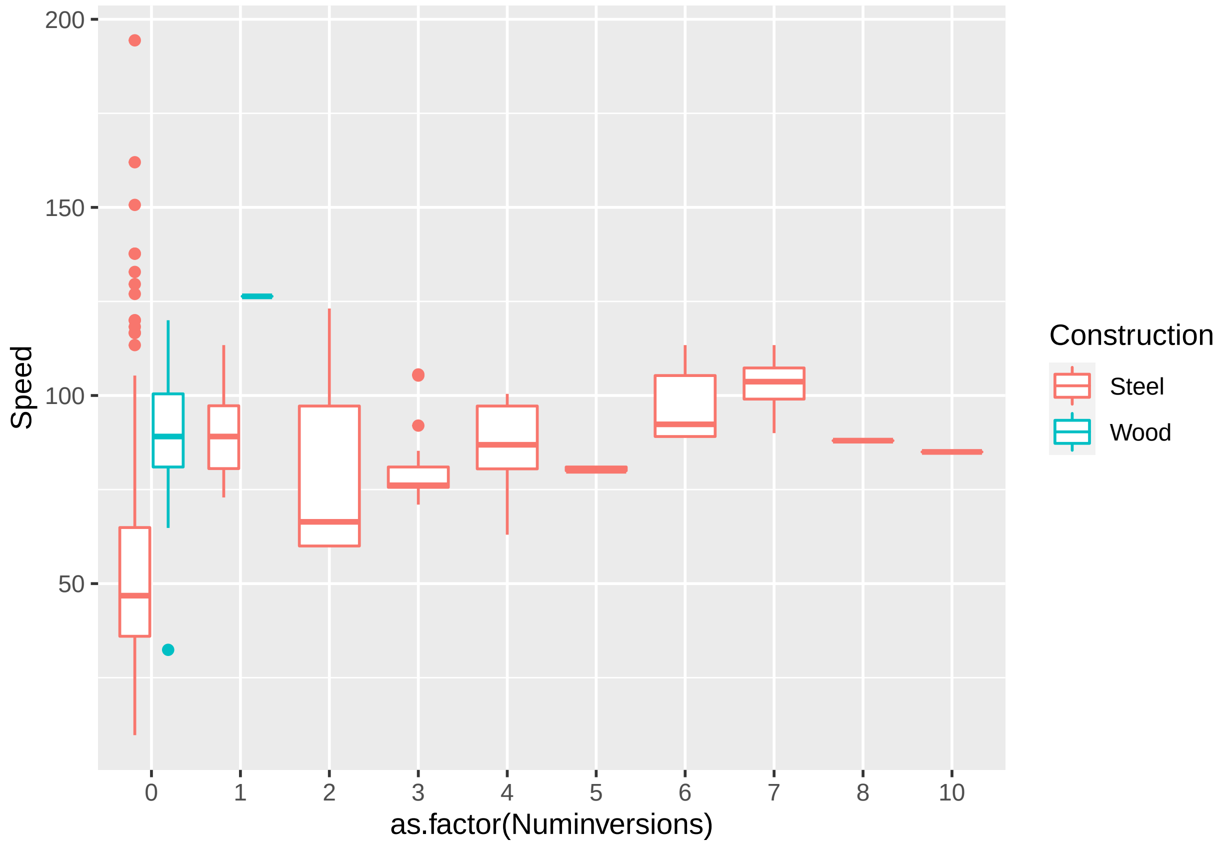

```
roller_coasters_raw %>%
  ggplot() +
    geom_point(aes(x = Length, y = Speed, color = Construction))
```

```
## Warning: Removed 148 rows containing missing values (geom_point).
```



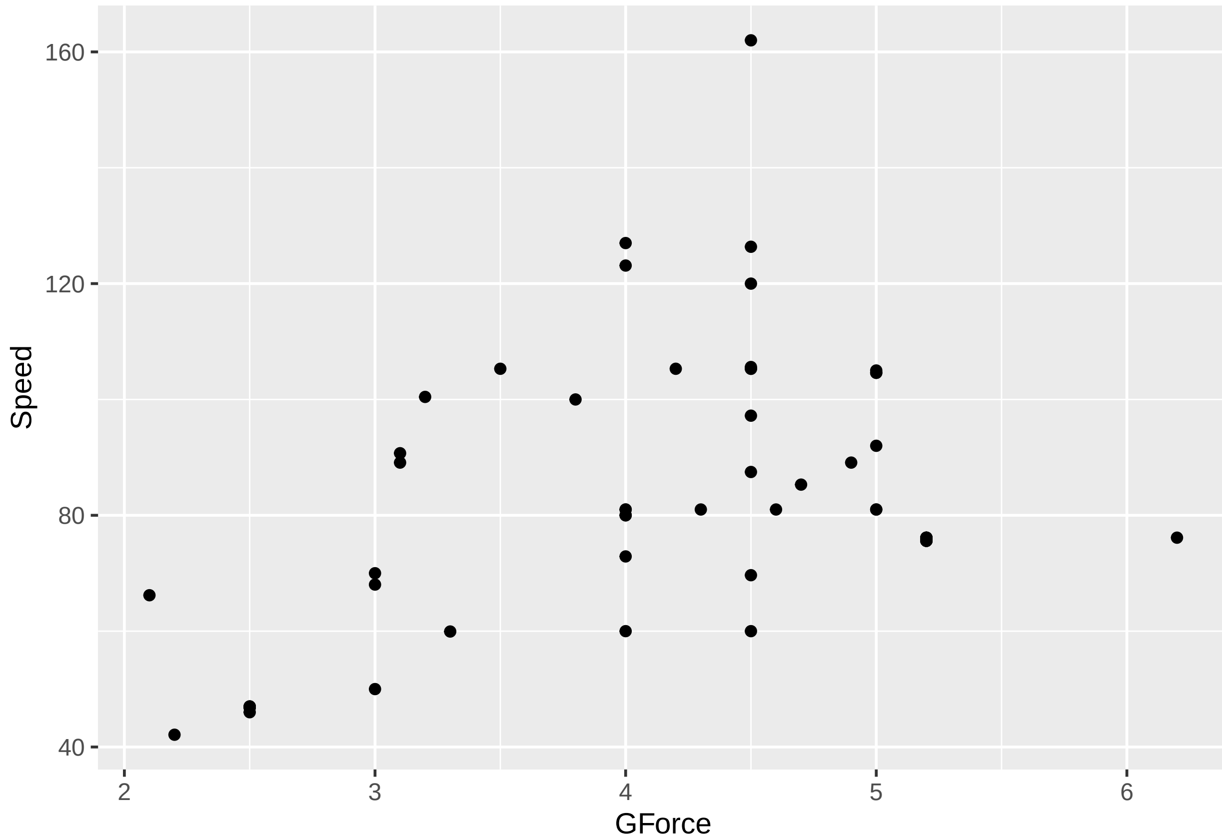

```
roller_coasters_raw %>%
  ggplot() +
    geom_boxplot(aes(x = as.factor(Numinversions), y = Speed, color = Construction))
```

```
## Warning: Removed 138 rows containing non-finite values (stat_boxplot).
```




```
# dost lame ... lahko spustimo
roller_coasters_raw %>%
  filter(!is.na(GForce)) %>%
  ggplot() +
    geom_point(aes(x = GForce, y = Speed))
```

```
## Warning: Removed 2 rows containing missing values (geom_point).
```



Inference and Hypothesis testing

Hypothesis Testing

0) Check CLT conditions Central limit theorem:

- Samples are independent,

- Sample size is bigger or equal to 30,
- Population distribution is not strongly skewed.

1) Set-up the hypothesis
2) Assume threshold values

- $\alpha - significancelevel$ - typically 0.05

3) Calculate the Results:

- point est.
- number of cases
- sd - standard deviation
- se - standard error
- df - degrees of freedom $df = n - 1$

- t-statistics
- p-value

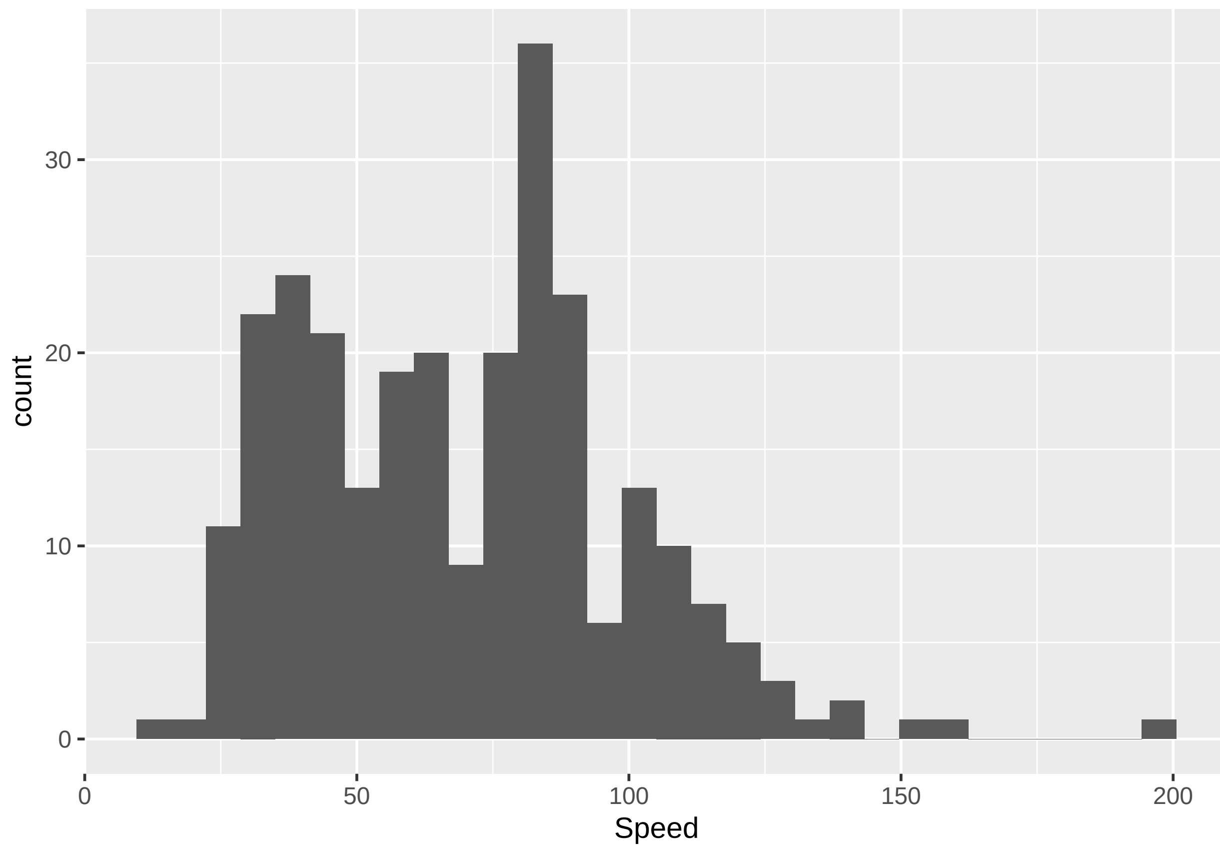
4) Draw conclusions - Accept or reject hypothesis

If we meet those criteria, we can infer about the population based on the analysis we do on the sample

```
ggplot(roller_coasters_raw) +
  geom_histogram(aes(x = Speed))
```

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

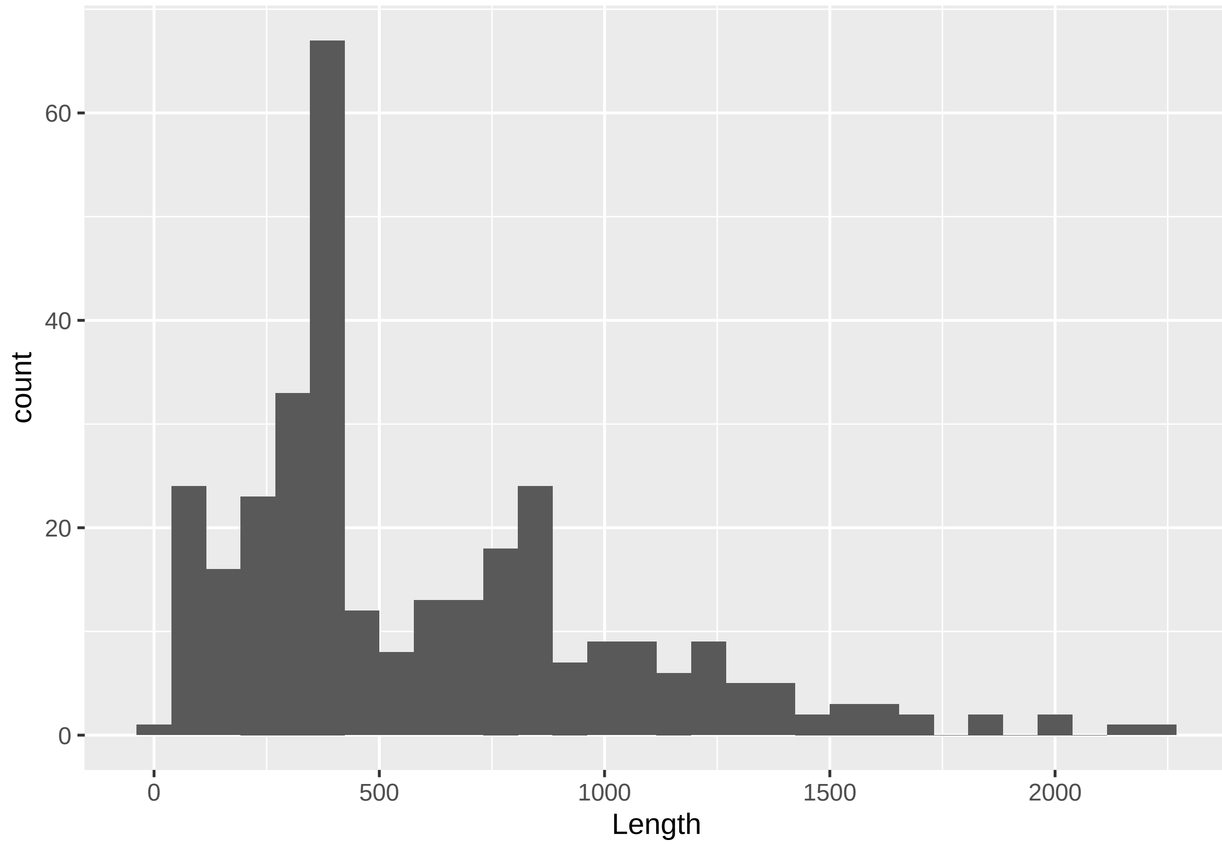
```
## Warning: Removed 138 rows containing non-finite values (stat_bin).
```




```
ggplot(roller_coasters_raw) +
  geom_histogram(aes(x = Length))
```

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

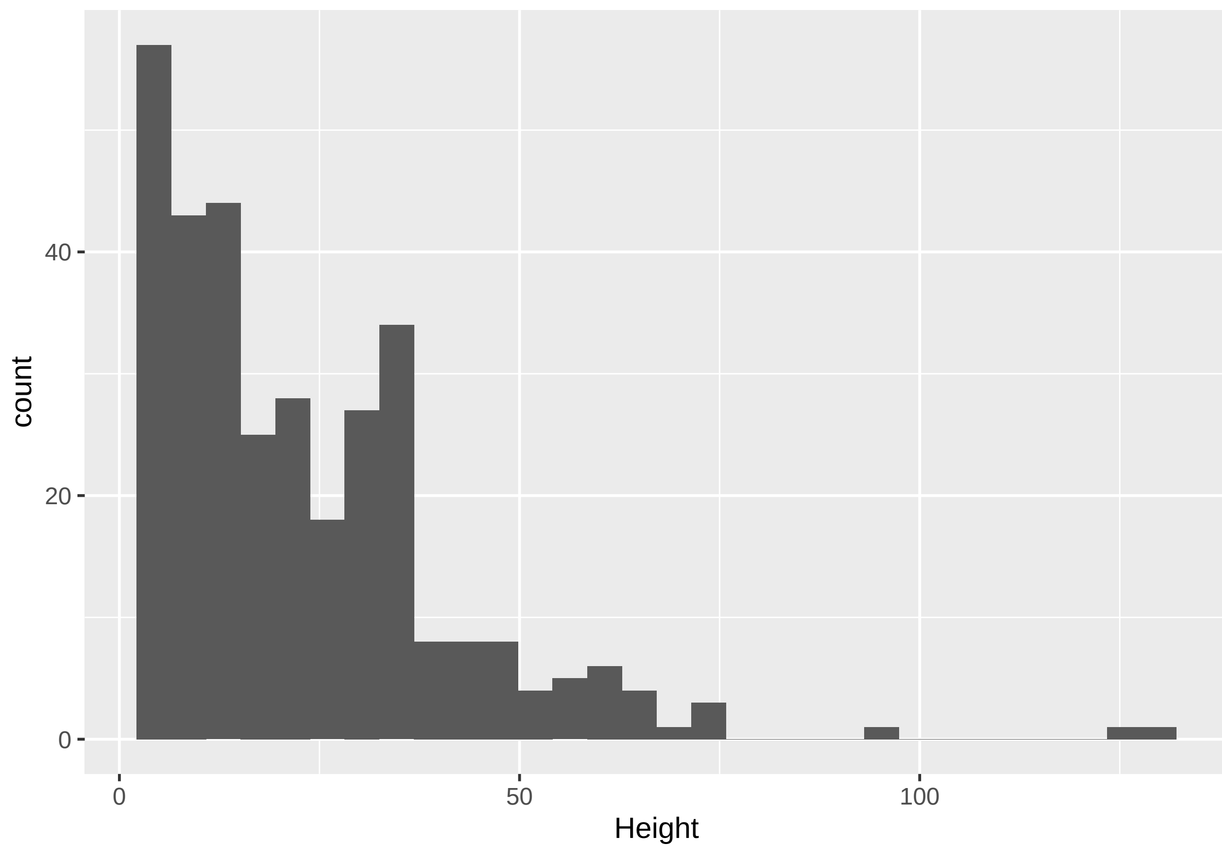
```
## Warning: Removed 90 rows containing non-finite values (stat_bin).
```




```
ggplot(roller_coasters_raw) +
  geom_histogram(aes(x = Height))
```

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

```
## Warning: Removed 82 rows containing non-finite values (stat_bin).
```




```
(symmetry.test(roller_coasters_raw$Speed))
```

```
##
##  m-out-of-n bootstrap symmetry test by Miao, Gel, and Gastwirth (2006)
##
## data:  roller_coasters_raw$Speed
## Test statistic = 0.37053, p-value = 0.786
## alternative hypothesis: the distribution is asymmetric.
## sample estimates:
## bootstrap optimal m
##                  36
```

```
(shapiro.test(roller_coasters_raw$Speed))
```

```
##
##  Shapiro-Wilk normality test
##
## data:  roller_coasters_raw$Speed
## W = 0.96757, p-value = 8.687e-06
```

```
(symmetry.test(roller_coasters_raw$Height))
```

```
##
##  m-out-of-n bootstrap symmetry test by Miao, Gel, and Gastwirth (2006)
```

```
##
## data:  roller_coasters_raw$Height
## Test statistic = 6.772, p-value < 2.2e-16
## alternative hypothesis: the distribution is asymmetric.
## sample estimates:
## bootstrap optimal m
##                  44
```

```
(shapiro.test(roller_coasters_raw$Height))
```

```
##
##  Shapiro-Wilk normality test
##
## data:  roller_coasters_raw$Height
## W = 0.84671, p-value < 2.2e-16
```

```
(symmetry.test(roller_coasters_raw$Length))
```

```
##
##  m-out-of-n bootstrap symmetry test by Miao, Gel, and Gastwirth (2006)
##
## data:  roller_coasters_raw$Length
## Test statistic = 10.298, p-value < 2.2e-16
## alternative hypothesis: the distribution is asymmetric.
## sample estimates:
## bootstrap optimal m
##                 139
```

```
(shapiro.test(roller_coasters_raw$Length))
```

```
##
##  Shapiro-Wilk normality test
##
## data:  roller_coasters_raw$Length
## W = 0.90217, p-value = 1.789e-13
```

We are allowed to infere on Speed, since only Speed meets the Limit Theorem requirements…

```
roller_coasters_speeds <- roller_coasters_raw %>%
  select(Speed) %>%
  filter(!is.na(Speed))
roller_coasters_speeds
```

```
## # A tibble: 270 x 1
##    Speed
##    <dbl>
##  1  194.
##  2  162
##  3  151.
##  4  138.
##  5  127
```

```
##   6  138.
##   7  130.
##   8  120
##   9  126.
##  10  113.
## # ... with 260 more rows
```

Hypothesis 1

H_0 : population mean speed is 70mph $\mu = 70$ H_A : population mean speed is not 70mph $\mu \neq 70$

We want to infere on the Speed of roller coasters...

```r
(point_est_speed <- 70)
```

```
## [1] 70
```

```r
(mean_speed <- mean(roller_coasters_speeds$Speed))
```

```
## [1] 69.36267
```

```r
(sd_speed <- sd(roller_coasters_speeds$Speed)) # standard deviation
```

```
## [1] 29.32774
```

```r
(sem_speed <- sd_speed / nrow(roller_coasters_speeds)) # standard error
```

```
## [1] 0.1086213
```

```r
(df_speed <- nrow(roller_coasters_speeds) - 1)
```

```
## [1] 269
```

```r
(t_speed <- (point_est_speed-mean_speed) / sem_speed)
```

```
## [1] 5.867482
```

p-value

```r
(p_val <- 2*(1- pt(t_speed, df = df_speed)))
```

```
## [1] 1.296661e-08
```

95% confidence intervals

```r
#lower limit
# mean_speed - 1.96 * sem_speed
mean_speed + qt(0.025, df = df_speed) * sem_speed
```

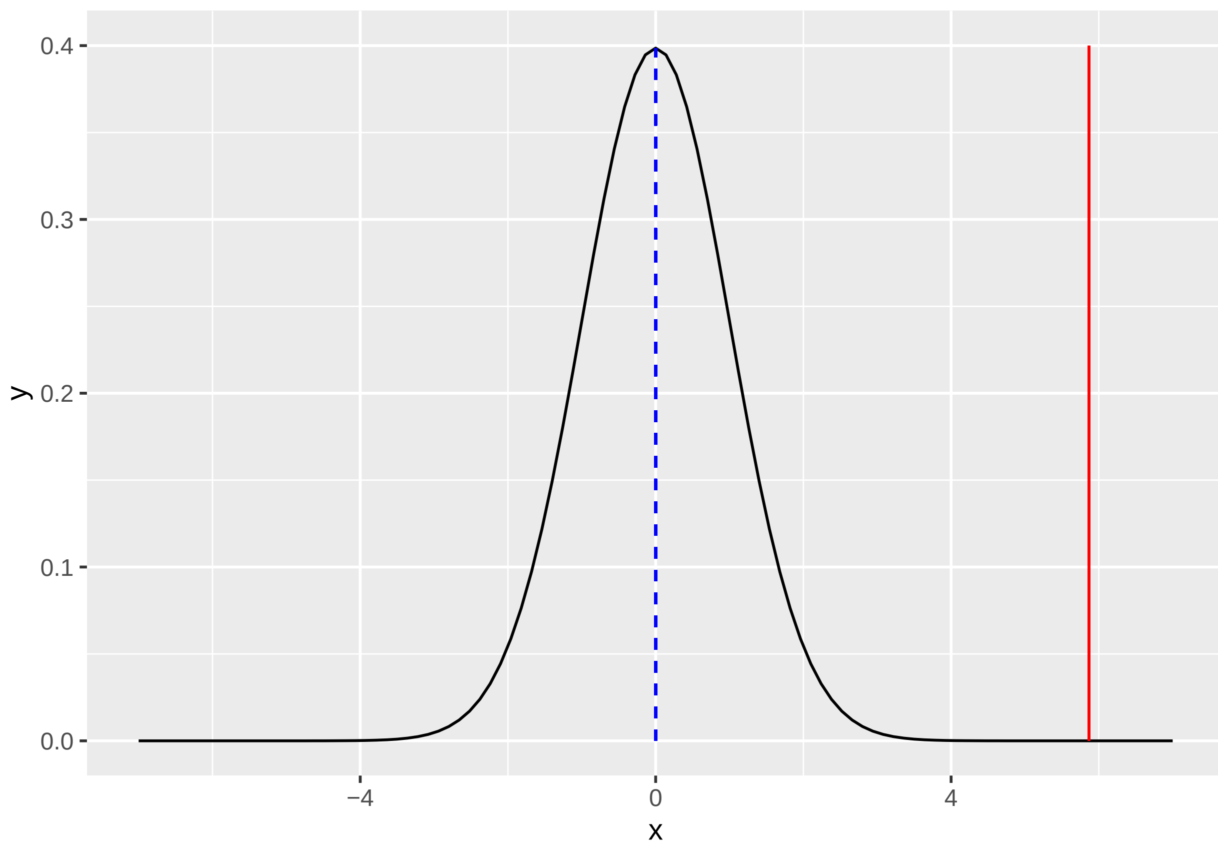
```
## [1] 69.14881
```

```r
#upper limit
# mean_speed + 1.96 * sem_speed
mean_speed + qt(0.975, df = df_speed) * sem_speed
```

```
## [1] 69.57652
```

Let's plot our discovery…

```r
xframe <- seq(-7, 7, length = 100)
ggplot(data.frame(x = xframe), aes(x = x)) +
  stat_function(fun = dt, args = list(df = df_speed)) +
  geom_segment(aes(x = 0, y = 0, xend = 0, yend = dt(0, df = df_speed)),
               color = 'blue',
               linetype = 'dashed') +
  geom_segment(aes(x = t_speed, y = 0, xend = t_speed, yend = 0.4),
               color = 'red')
```



We reject the null hypothesis in favor of the alternative. Mean roller coaster speed is not 70mph!

Difference of means t-test

We want to check if the Wooden roller coasters are on average faster that the Steel ones.

```
roller_coasters_steel <- roller_coasters_raw %>%
  filter(Construction == "Steel" & !is.na(Speed))

roller_coasters_wood <- roller_coasters_raw %>%
  filter(Construction == "Wood" & !is.na(Speed))
```

Check number of instances

```
nrow(roller_coasters_steel)
```

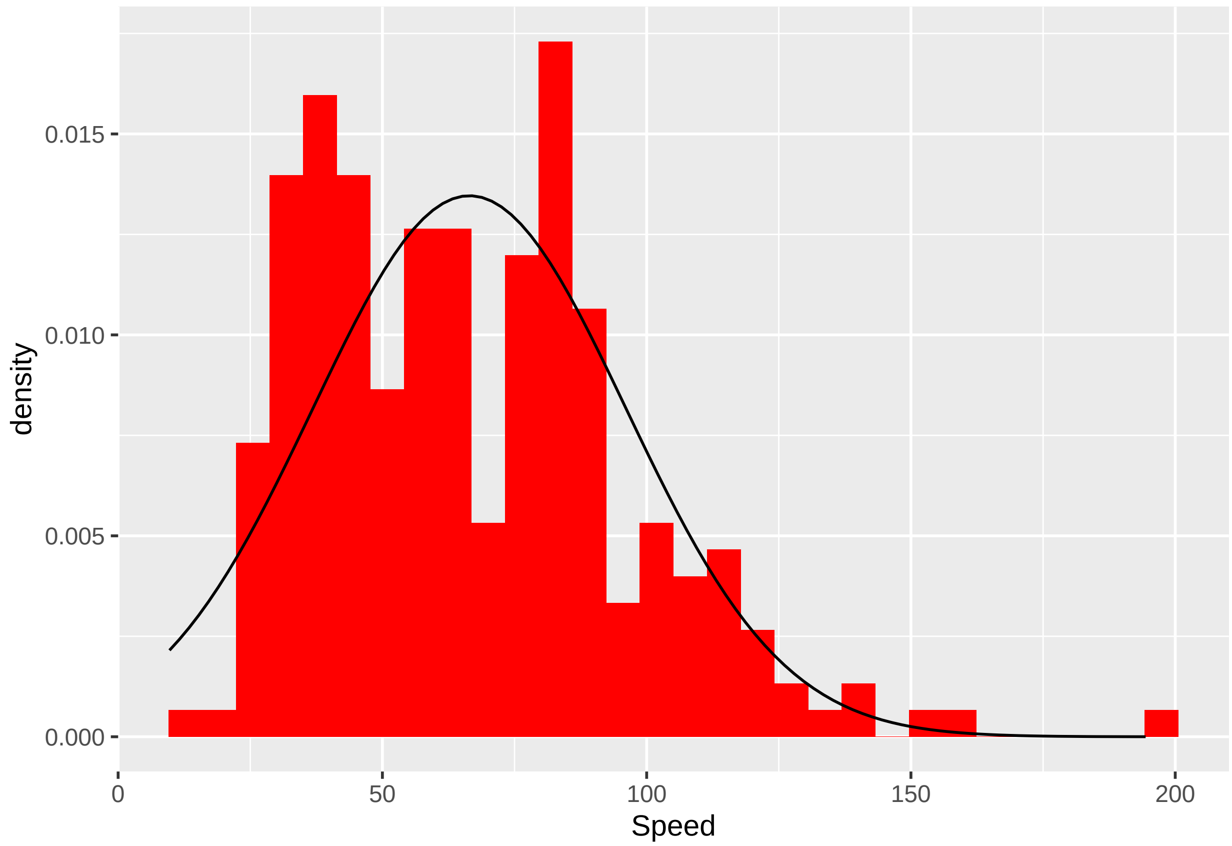
```
## [1] 236
```

```
nrow(roller_coasters_wood)
```

```
## [1] 34
```

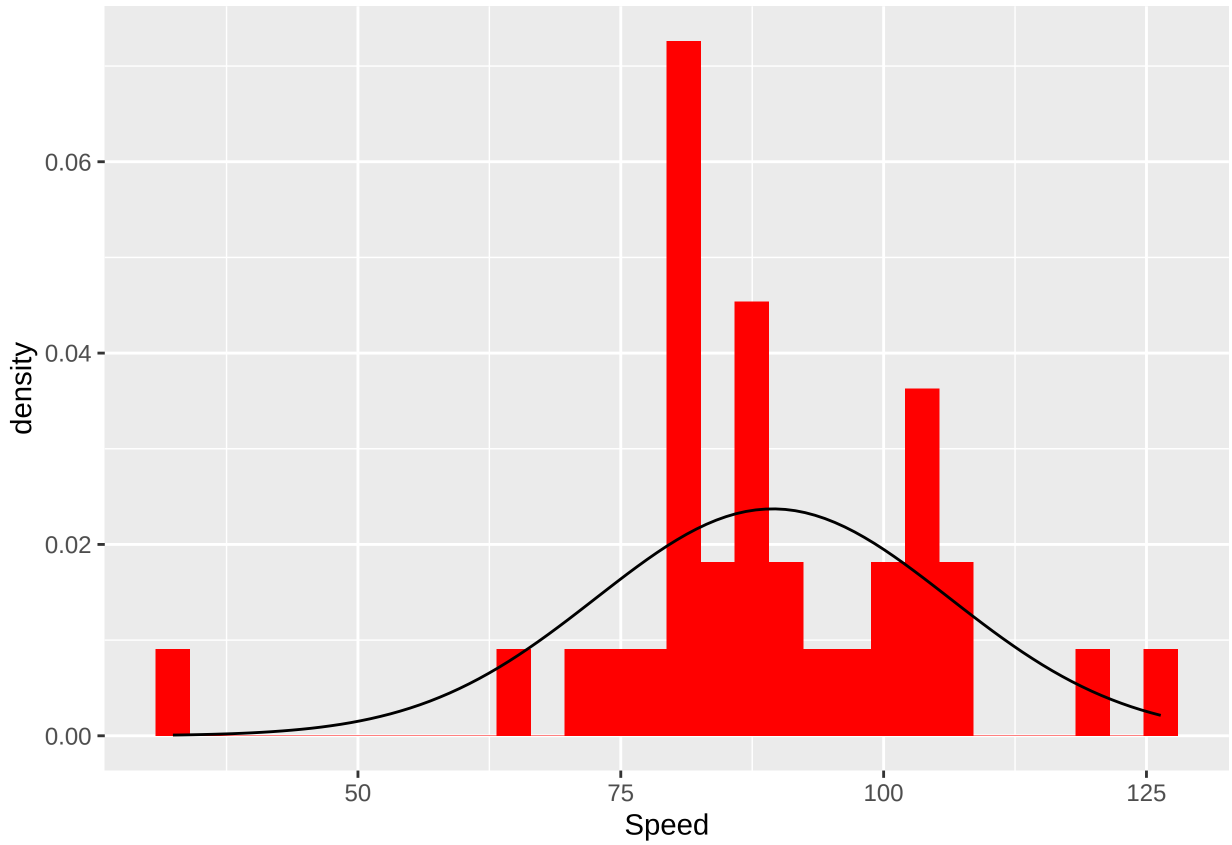
```
ggplot(roller_coasters_steel) +
  geom_histogram(aes(x = Speed, y = ..density..), fill ='red') +
  stat_function(fun = dnorm, args = list(mean = mean(roller_coasters_steel$Speed), sd = sd(roller_coaste
```

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



```r
ggplot(roller_coasters_wood) +
  geom_histogram(aes(x = Speed, y = ..density..), fill ='red') +
  stat_function(fun = dnorm, args = list(mean = mean(roller_coasters_wood$Speed), sd = sd(roller_coaste
```

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



$$H_O : mean_{Wood} - mean_{Steel} = 0$$

$$H_A : mean_{Wood} - mean_{Steel} \neq 0$$

$$\alpha = 0.05$$

```r
(point_est_const <- mean(roller_coasters_wood$Speed) - mean(roller_coasters_steel$Speed))
```

```
## [1] 22.98329
```

```r
# (sample_sd <- sd(kiwi_gs_m$height_cm))
(SE <- sqrt((sd(roller_coasters_wood$Speed)^2/nrow(roller_coasters_wood)) + sd(roller_coasters_steel$Sp
```

```
## [1] 3.470155
```

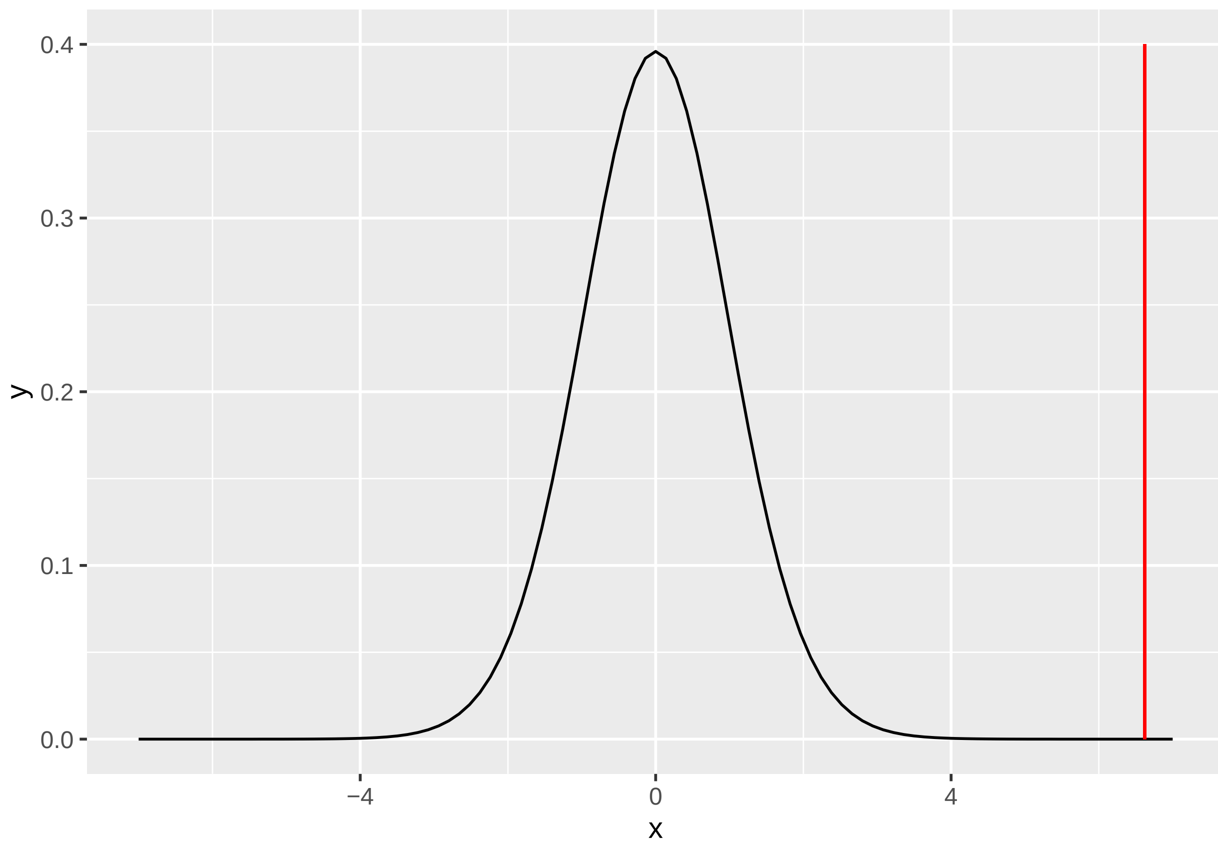
```r
(df <- nrow(roller_coasters_wood) - 1) # less
```

```
## [1] 33
```

```r
(t_stat_const <- (point_est_const - 0) / SE) # t-score!
```

```
## [1] 6.62313
```

```r
ggplot(data.frame(x = seq(-7, 7, length = 100)), aes(x = x)) +
  stat_function(fun = dt, args = list(df = df)) +
  geom_segment(aes(x = t_stat_const, y = 0, xend = t_stat_const, yend = 0.4), color = 'red')
```




```r
(p_val <- 2 * (1 - pt(t_stat_const, df)))
```

```
## [1] 1.560164e-07
```

We reject the NULL hypothesis in favour of the alternative. The difference in means is significant and wooden roller coasters go faster on average.

Correlation Analysis

```r
# precej zanimivi so Height, Length, Numinversions
(cor.test(roller_coasters_raw$Height, roller_coasters_raw$Speed))
```

```
##
##  Pearson's product-moment correlation
##
## data:  roller_coasters_raw$Height and roller_coasters_raw$Speed
## t = 38.222, df = 256, p-value < 2.2e-16
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  0.9019179 0.9388051
## sample estimates:
##       cor
## 0.9224392
```

```r
(cor.test(roller_coasters_raw$Length, roller_coasters_raw$Speed))
```

```
##
##  Pearson's product-moment correlation
##
## data:  roller_coasters_raw$Length and roller_coasters_raw$Speed
## t = 15.582, df = 258, p-value < 2.2e-16
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  0.6278199 0.7540719
## sample estimates:
##       cor
## 0.6962931
```

```r
(cor.test(roller_coasters_raw$Numinversions, roller_coasters_raw$Speed))
```

```
##
##  Pearson's product-moment correlation
##
## data:  roller_coasters_raw$Numinversions and roller_coasters_raw$Speed
## t = 5.5742, df = 268, p-value = 6.061e-08
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  0.2110692 0.4253337
## sample estimates:
##       cor
## 0.3223236
```

```r
(cor.test(roller_coasters_raw$Duration, roller_coasters_raw$Speed))
```

```
##
##  Pearson's product-moment correlation
##
## data:  roller_coasters_raw$Duration and roller_coasters_raw$Speed
## t = 3.9954, df = 162, p-value = 9.781e-05
```

```
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  0.1532868 0.4328823
## sample estimates:
##       cor
## 0.2995011
```

```
(cor.test(roller_coasters_raw$GForce, roller_coasters_raw$Speed))
```

```
##
##  Pearson's product-moment correlation
##
## data:  roller_coasters_raw$GForce and roller_coasters_raw$Speed
## t = 3.3676, df = 56, p-value = 0.001377
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  0.1701111 0.6045861
## sample estimates:
##       cor
## 0.4103754
```

```
(cor.test(roller_coasters_raw$Opened, roller_coasters_raw$Speed)) # not good
```

```
##
##  Pearson's product-moment correlation
##
## data:  roller_coasters_raw$Opened and roller_coasters_raw$Speed
## t = 0.26238, df = 260, p-value = 0.7932
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  -0.1051251  0.1371870
## sample estimates:
##        cor
## 0.01626982
```

```
#pairs(roller_coasters, lower.panel = NULL)
```

Regression

```
roller_coasters <- roller_coasters_raw %>%
  select(Construction, Length, Height, Speed) %>%
  filter(!is.na(Speed) & !is.na(Height) & !is.na(Length)) %>%
  mutate("Steel" = as.numeric(Construction == 'Steel')) %>%
  select(-Construction)
roller_coasters
```

```
## # A tibble: 252 x 4
##    Length Height Speed Steel
##     <dbl>  <dbl> <dbl> <dbl>
```

```
## 1   853.  128.   194.      1
## 2   376.  126.   162       1
## 3  2010.   94.5  151.      1
## 4  1619.   74.7  138.      1
## 5  1620    73    127       1
## 6  1372.   71.6  138.      1
## 7  1610.   70.1  130.      1
## 8  1700    67    120       1
## 9  2143.   66.4  126.      0
## 10   396.   66.4  113.      1
## # ... with 242 more rows
```

```r
## 75% of the sample size
smp_size <- floor(0.75 * nrow(roller_coasters))

## set the seed to make your partition reproducible
set.seed(123)
train_ind <- sample(seq_len(nrow(roller_coasters)), size = smp_size)

(train <- roller_coasters[train_ind, ])
```

```
## # A tibble: 189 x 4
##    Length Height Speed Steel
##     <dbl>  <dbl> <dbl> <dbl>
##  1   412.   16.2  50.2     1
##  2   207     8.5  34.9     1
##  3   538.   13.4  50.2     1
##  4   375.   61   105.      1
##  5   427.   10.7  32.4     0
##  6   774.   14.6  68.0     1
##  7   950    36    90       1
##  8   717.   23.8  82.6     1
##  9   309.   39.9  81       1
## 10   264     6    45       1
## # ... with 179 more rows
```

```r
(test <- roller_coasters[-train_ind, ])
```

```
## # A tibble: 63 x 4
##    Length Height Speed Steel
##     <dbl>  <dbl> <dbl> <dbl>
##  1   376.  126.   162      1
##  2  2010.   94.5  151.     1
##  3   671.   62.5  133.     1
##  4   347.   61    105.     1
##  5   367.   58.2  105.     1
##  6  1167.   57.3  113.     1
##  7  1654.   49.1  105.     0
##  8  1332.   47.5  105.     1
##  9   150    46    105      1
## 10   192.   45.7  105.     1
## # ... with 53 more rows
```

```r
lin_model <- lm(Speed ~ ., data = train)
(summary(lin_model))
```

```
##
## Call:
## lm(formula = Speed ~ ., data = train)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -21.388  -6.020  -0.644   4.466  35.365
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 33.376471   2.513555  13.279  < 2e-16 ***
## Length       0.013174   0.002178   6.047 7.97e-09 ***
## Height       1.248969   0.047545  26.269  < 2e-16 ***
## Steel       -5.752860   2.109654  -2.727  0.00701 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 8.749 on 185 degrees of freedom
## Multiple R-squared:  0.9103, Adjusted R-squared:  0.9088
## F-statistic: 625.7 on 3 and 185 DF,  p-value: < 2.2e-16
```

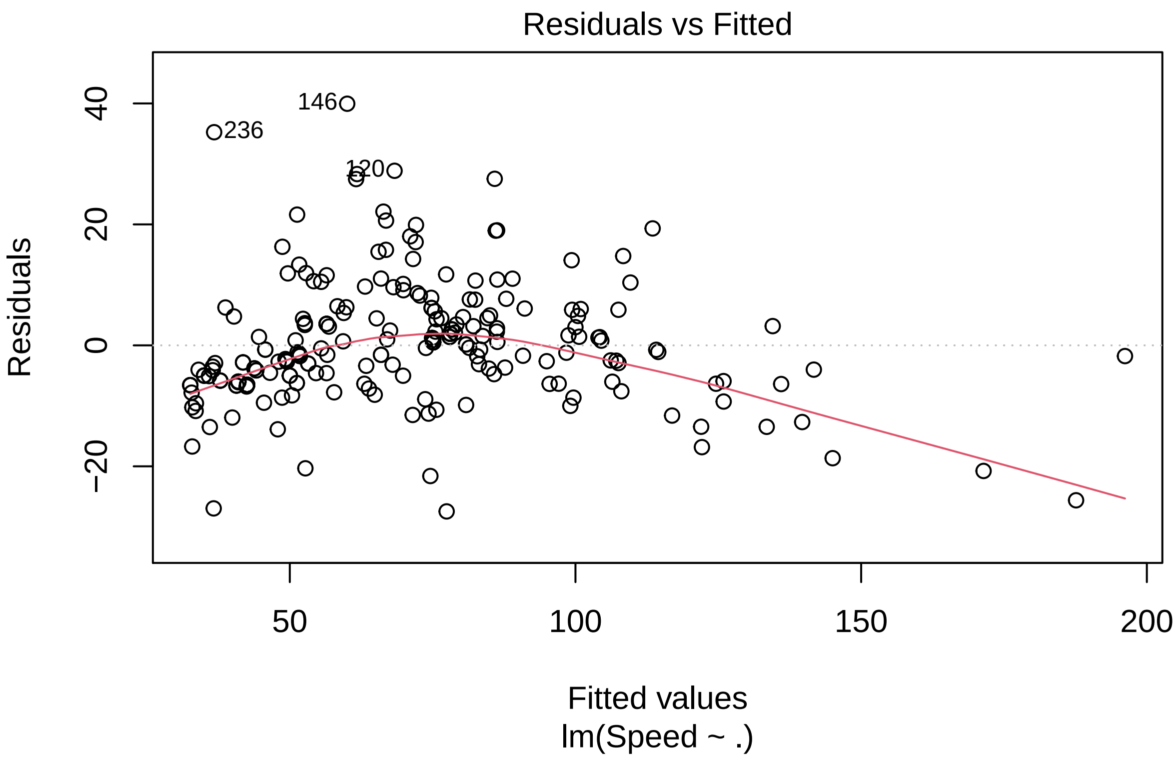
```r
(coef(lin_model))
```

```
## (Intercept)      Length      Height       Steel
## 33.37647051  0.01317372  1.24896948 -5.75286049
```

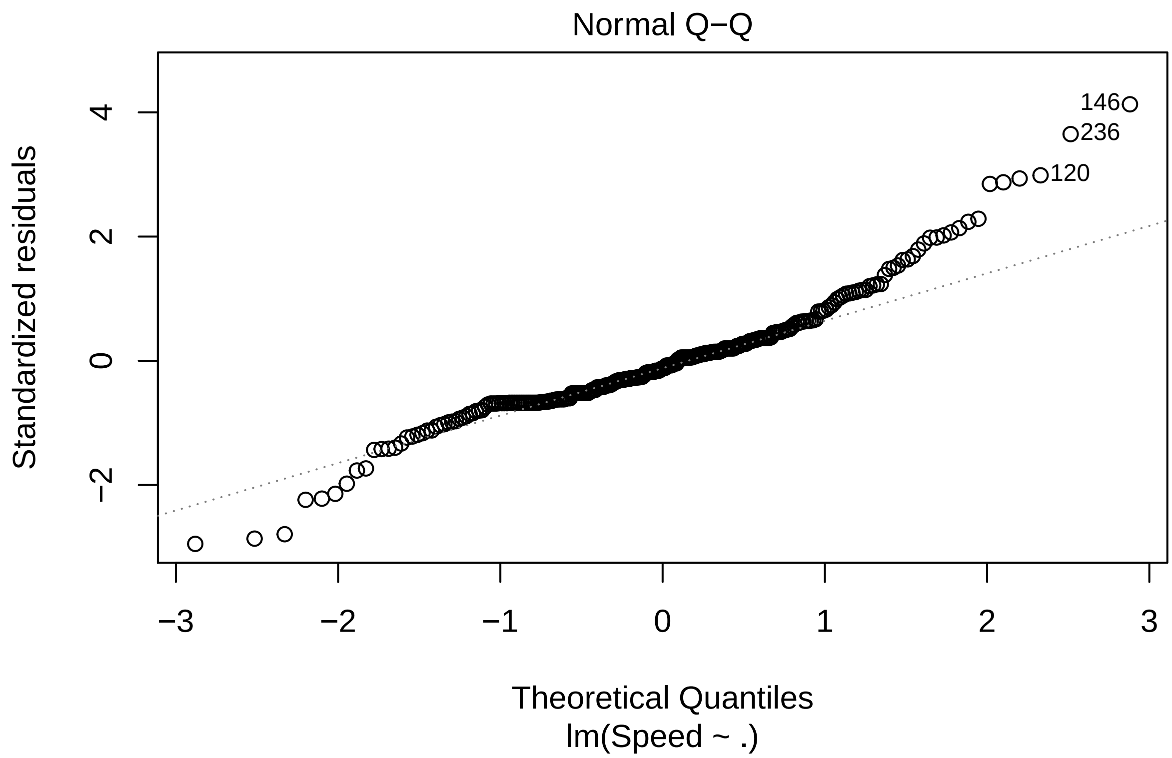
```r
rc_all <- lm(Speed ~ ., data = roller_coasters)
(summary(rc_all))
```

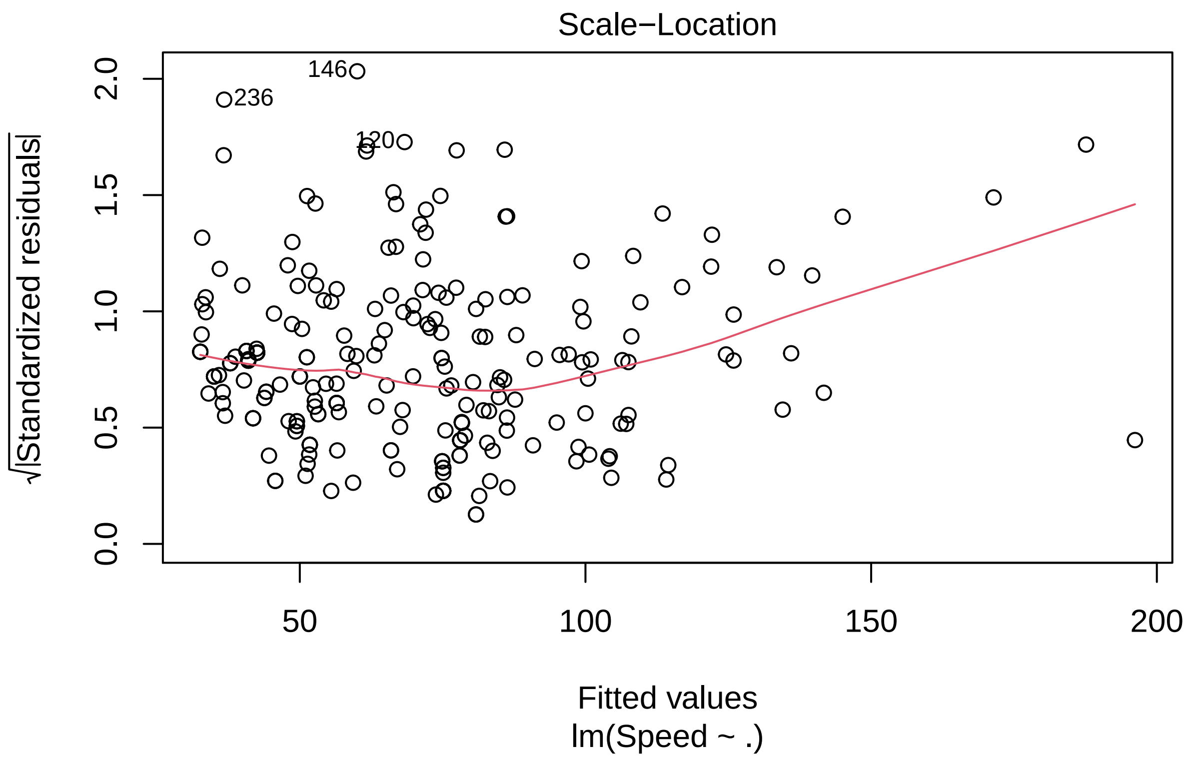
```
##
## Call:
## lm(formula = Speed ~ ., data = roller_coasters)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -27.452  -6.131  -1.180   3.826  39.948
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 33.699899   2.434493  13.843  < 2e-16 ***
## Length       0.014037   0.001915   7.329 3.26e-12 ***
## Height       1.222438   0.039889  30.646  < 2e-16 ***
## Steel       -5.998684   2.046036  -2.932  0.00368 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 9.701 on 248 degrees of freedom
## Multiple R-squared:  0.8946, Adjusted R-squared:  0.8933
## F-statistic: 701.3 on 3 and 248 DF,  p-value: < 2.2e-16
```

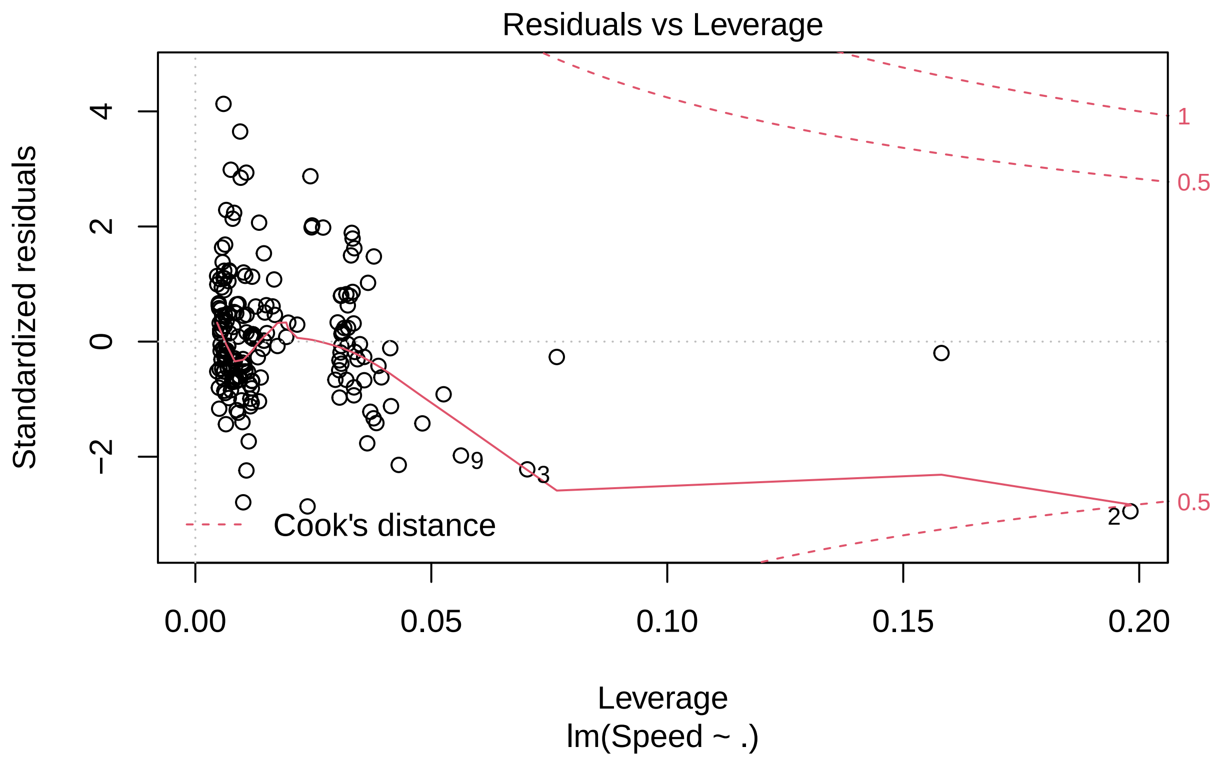
```
plot(rc_all)
```



Residuals vs Fitted
Residuals
Fitted values
lm(Speed ~ .)
40
20
0
−20
50
100
150
200
146
236
120

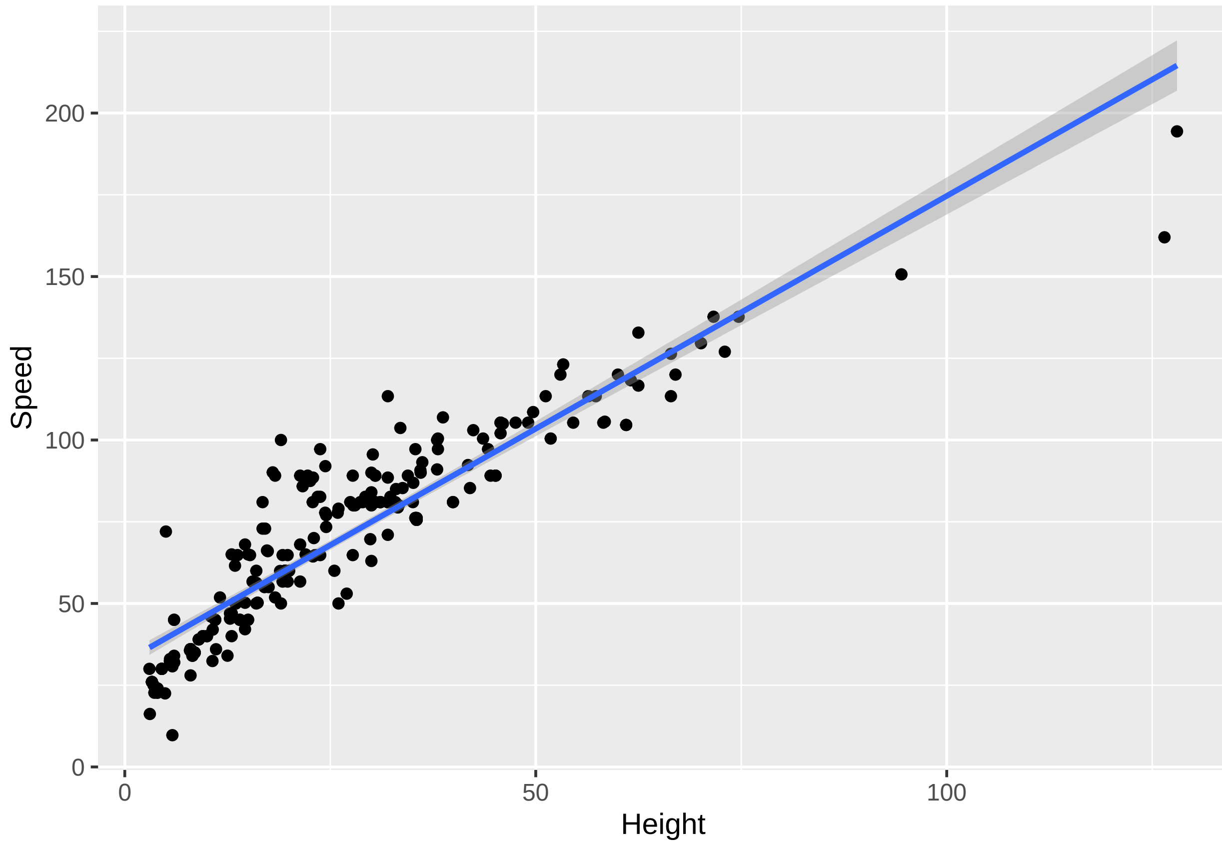

Normal Q−Q
Standardized residuals
Theoretical Quantiles
lm(Speed ~ .)
146
236
120
−2
0
2
4
−3
−2
−1
0
1
2
3


Scale−Location
√|Standardized residuals|
0.0
0.5
1.0
1.5
2.0
146
236
120
50
100
150
200
Fitted values
lm(Speed ~ .)




```
roller_coasters %>% ggplot()+
  geom_point(aes(x = Height, y = Speed))+
  geom_smooth(aes(x = Height, y = Speed), method = lm)
```

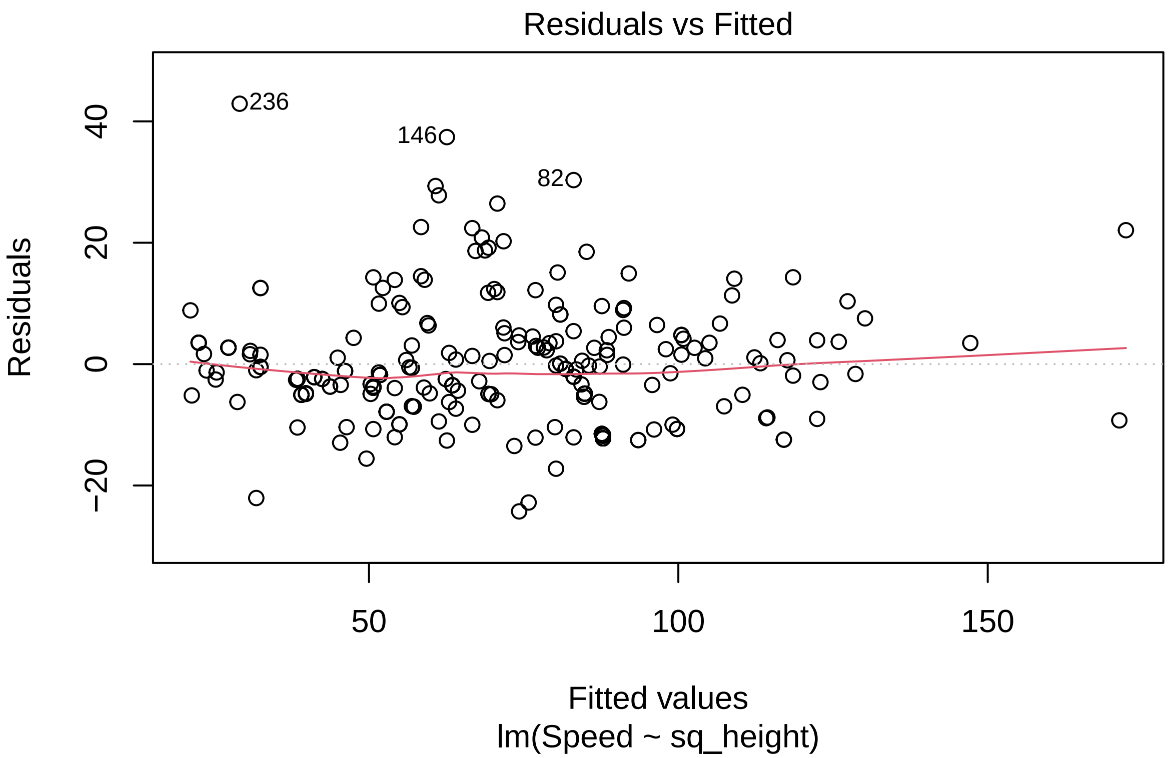
```
## `geom_smooth()` using formula 'y ~ x'
```



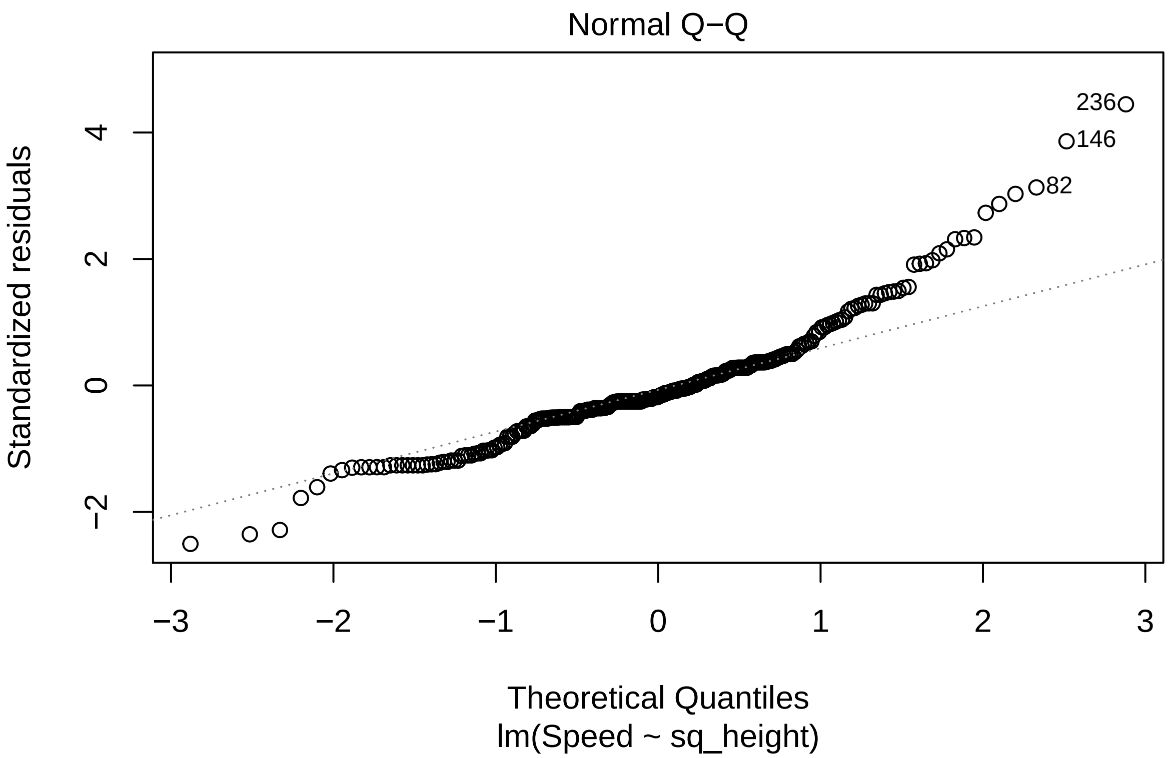

```r
roller_coasters$log_height <- log(roller_coasters$Height)
roller_coasters$sq_height <- sqrt(roller_coasters$Height)

rc_all <- lm(Speed ~ sq_height, data = roller_coasters)
(summary(rc_all))
```

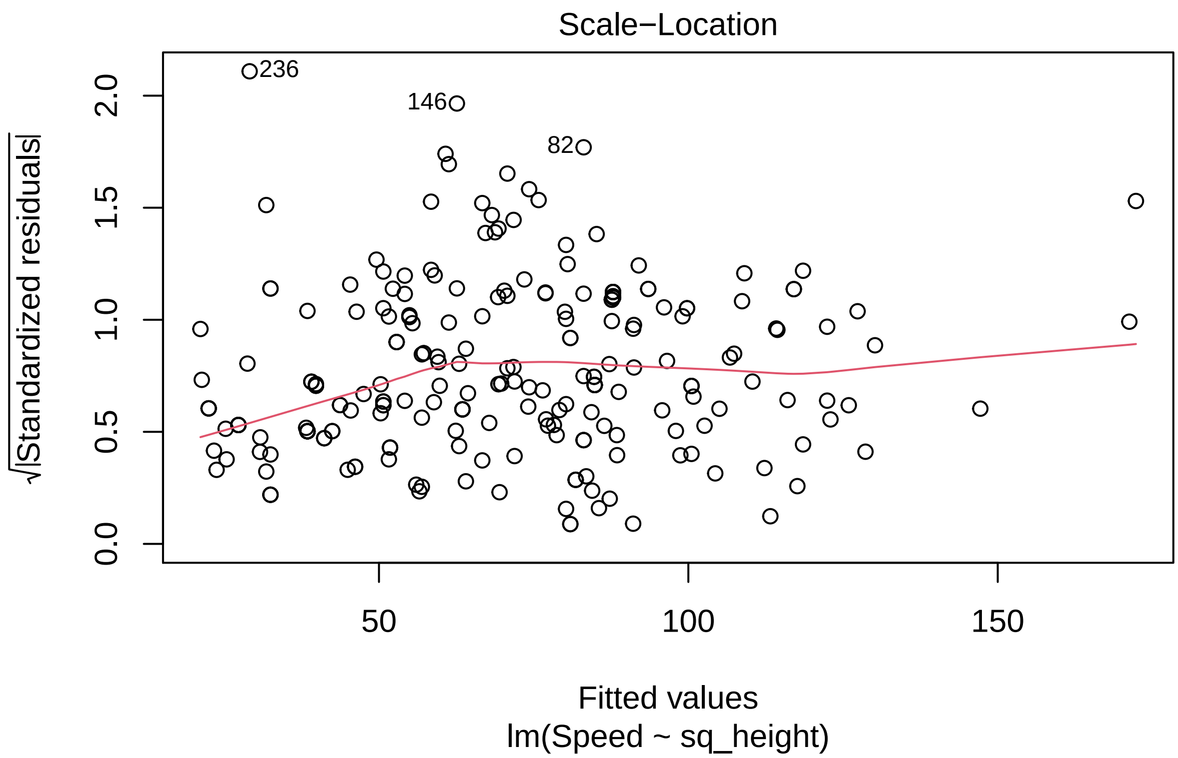
```
##
## Call:
## lm(formula = Speed ~ sq_height, data = roller_coasters)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -24.269  -4.981  -1.706   3.645  42.904
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -6.1846     1.7613  -3.511 0.000529 ***
## sq_height    15.7782     0.3444  45.813  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 9.708 on 250 degrees of freedom
## Multiple R-squared:  0.8936, Adjusted R-squared:  0.8931
## F-statistic:  2099 on 1 and 250 DF,  p-value: < 2.2e-16
```

```
plot(rc_all)
```

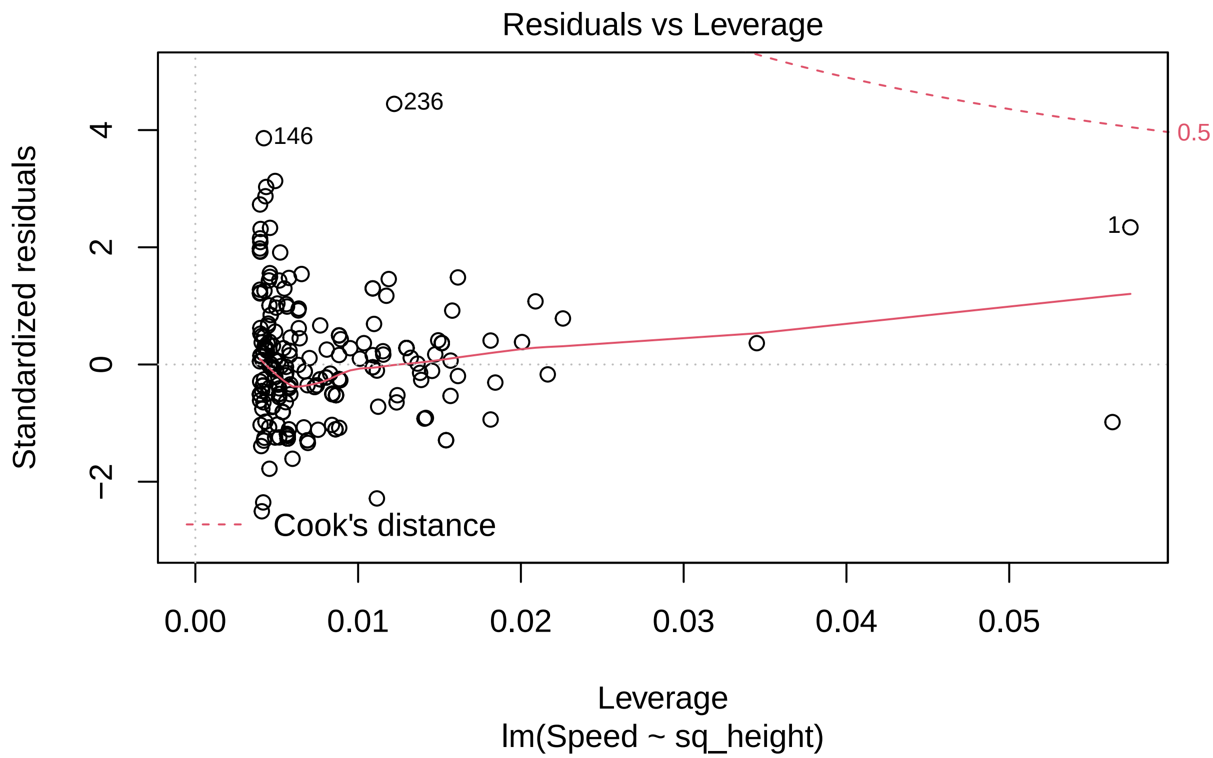


Residuals vs Fitted
236
146
82
40
20
0
−20
Residuals
50
100
150
Fitted values
lm(Speed ~ sq_height)


Normal Q−Q
Standardized residuals
Theoretical Quantiles
lm(Speed ~ sq_height)
236
146
82
4
2
0
−2
−3
−2
−1
0
1
2
3



Scale−Location
236
146
82
0.0
0.5
1.0
1.5
2.0
√|Standardized residuals|
50
100
150
Fitted values
lm(Speed ~ sq_height)



2. Red Billed Seagulls

The dataset seagulls.csv represents the data collected about seagulls in Auckland, New Zeland. Dataset can be found here.

Data was collected on two seperate occasions (summer and winter) and on four different locations: Muriwai (a), Piha (b), Mareatai (c), and Waitawa (d).

They collected seagull's weight, length, and sex, as well as its location and season. Authors of the dataset also point out that none of the locations is a major breeding site.

We also cleaned dataset a bit. Some cases have misspelled "MURIWAI" as "MURWAI". Variables location, coast, season, and sex have been converted from strings to factors, and length was renamed to height, since that is more accurate varable description.

```
seagulls <- read.csv("datasets/seagulls.csv")
seagulls[seagulls$LOCATION == "MURWAI",]$LOCATION <- "MURIWAI"
colnames(seagulls)[2] <- "HEIGHT"
seagulls$LOCATION <- as.factor(seagulls$LOCATION)
seagulls$COAST <- as.factor(seagulls$COAST)
seagulls$SEASON <- as.factor(seagulls$SEASON)
seagulls$SEX <- as.factor(seagulls$SEX)
```

Figure 1: Auckland region

Summary statistics

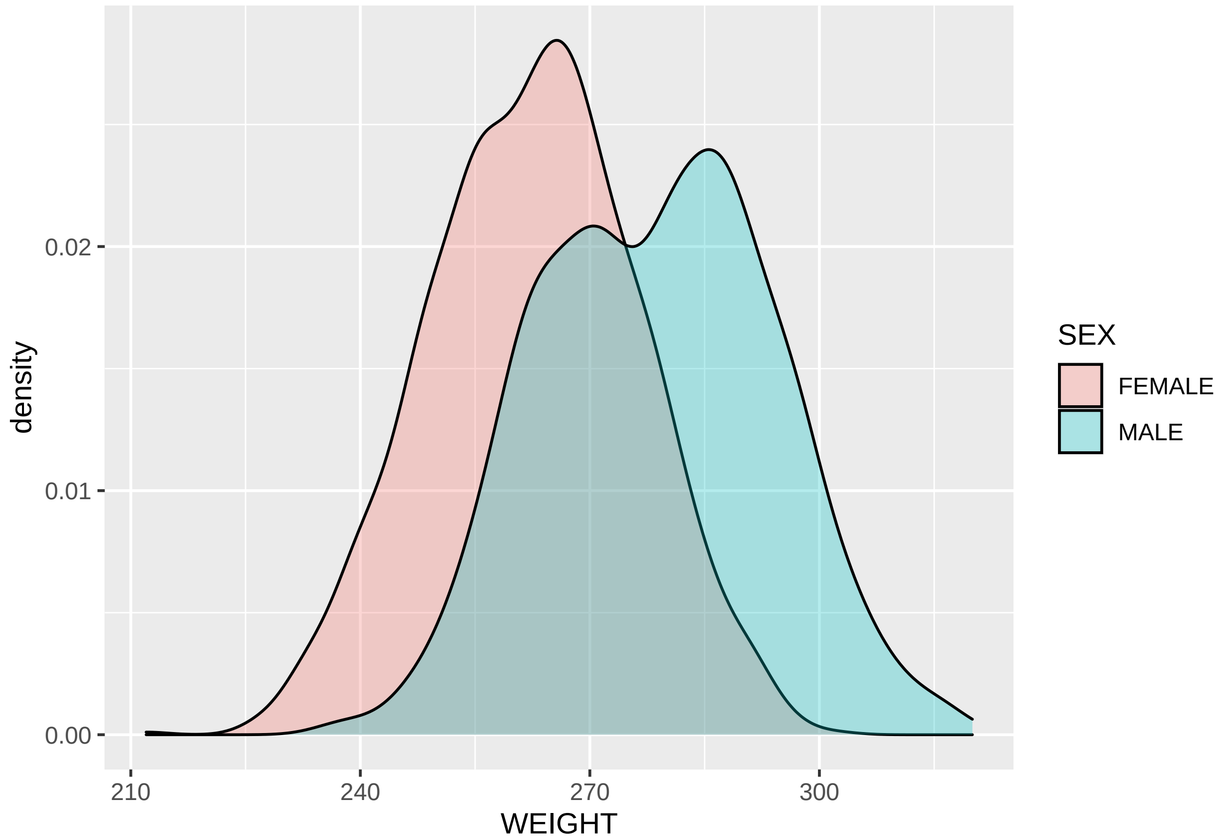
Seagulls dataset has 2487 cases and 6 variables: weight, height, location, coast, season, and sex. Weight and length are numerical, while location, coast, season, and sex are categorical.

```
summary(seagulls)
```

```
##      WEIGHT          HEIGHT          LOCATION      COAST         SEASON
##  Min.   :212.0   Min.   :28.5   MARAETAI:673   EAST:1251   SUMMER:1313
##  1st Qu.:259.0   1st Qu.:35.5   MURIWAI :589   WEST:1236   WINTER:1174
##  Median :269.0   Median :37.1   PIHA    :647
##  Mean   :270.4   Mean   :37.1   WAITAWA :578
##  3rd Qu.:282.0   3rd Qu.:38.8
##  Max.   :320.0   Max.   :44.8
##      SEX
##  FEMALE:1280
##  MALE  :1207
##
##
##
##
```

Weight of seagulls is in grams (g), and its distribution can be seen here:

```
seagulls %>% ggplot()+
  geom_density(aes(x = WEIGHT, fill = SEX), alpha = 0.3)
```



Average weight of males is 278.73g with minimum of 235g and maximum of 320g. Average weight of females is 262.49g with minimum of 212g and maximum of 302g. We can see that weights of males are not normally distributed, while weights of females could be. We can check this with normality test:

```
shapiro.test(seagulls[seagulls$SEX == "MALE",]$WEIGHT)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  seagulls[seagulls$SEX == "MALE", ]$WEIGHT
## W = 0.994, p-value = 8.841e-05
```

```
shapiro.test(seagulls[seagulls$SEX == "FEMALE",]$WEIGHT)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  seagulls[seagulls$SEX == "FEMALE", ]$WEIGHT
## W = 0.99724, p-value = 0.02575
```

We can see that weight is not normally distributed neither for males nor females, but latter are very close to passing the normallity test. We can also check if the distributions are at least symmetric:

```r
symmetry.test(seagulls[seagulls$SEX == "MALE",]$WEIGHT)
```

```
##
##  m-out-of-n bootstrap symmetry test by Miao, Gel, and Gastwirth (2006)
##
## data:  seagulls[seagulls$SEX == "MALE", ]$WEIGHT
## Test statistic = -0.80072, p-value = 0.458
## alternative hypothesis: the distribution is asymmetric.
## sample estimates:
## bootstrap optimal m
##                 330
```

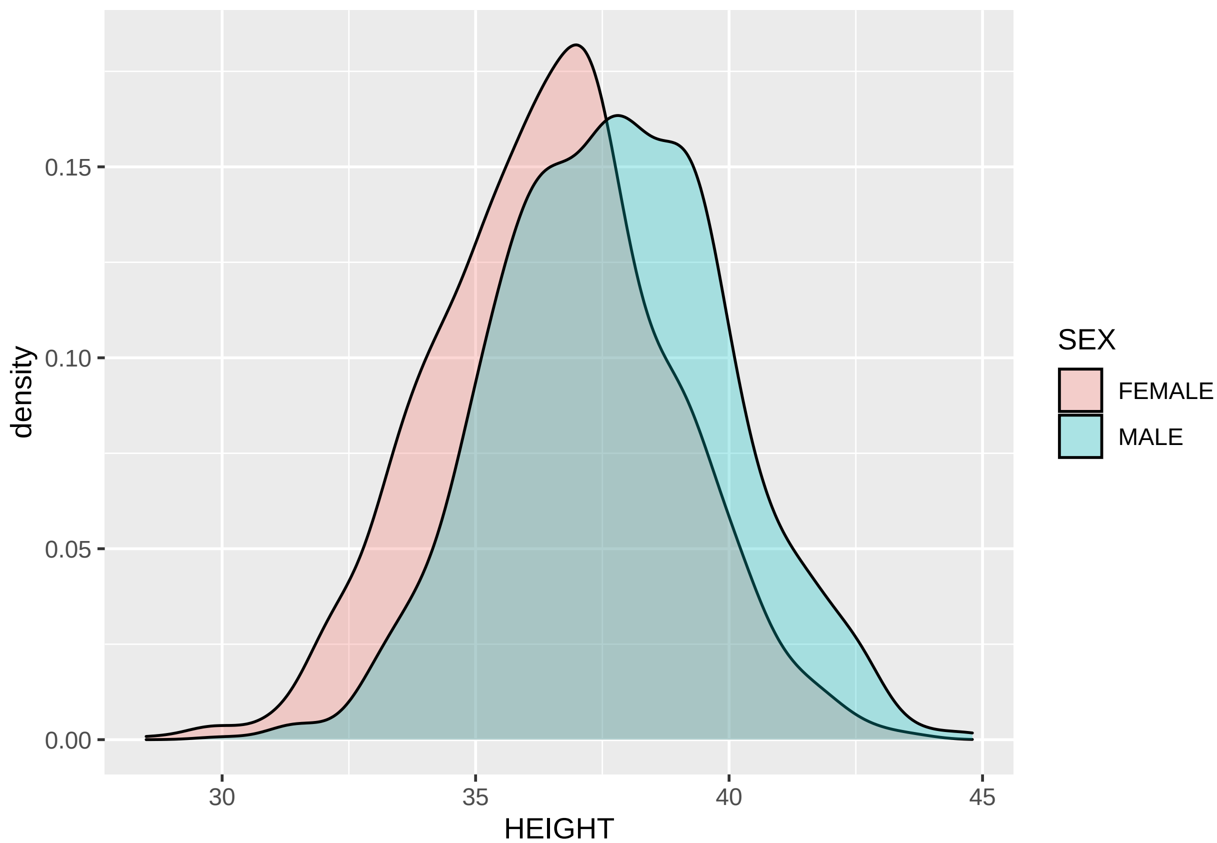
```r
symmetry.test(seagulls[seagulls$SEX == "FEMALE",]$WEIGHT)
```

```
##
##  m-out-of-n bootstrap symmetry test by Miao, Gel, and Gastwirth (2006)
##
## data:  seagulls[seagulls$SEX == "FEMALE", ]$WEIGHT
## Test statistic = -1.773, p-value = 0.14
## alternative hypothesis: the distribution is asymmetric.
## sample estimates:
## bootstrap optimal m
##                 631
```

Both pass symmetry test, meaning they are not strongly skewed and can be used later for inference.

Height of seagulls is in centimeters (cm):

```r
seagulls %>% ggplot()+
  geom_density(aes(x = HEIGHT, fill = SEX), alpha = 0.3)
```



Average height of males is 37.74cm. Smallest male's height is 30cm, while largest is 44.8cm. Female's average height is 36.5cm with minimum of 28.5cm and maximum of 43.7cm. Seagulls height seems more normally distributed than weight, but we can check:

```
shapiro.test(seagulls[seagulls$SEX == "MALE",]$HEIGHT)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  seagulls[seagulls$SEX == "MALE", ]$HEIGHT
## W = 0.9983, p-value = 0.2733
```

```
shapiro.test(seagulls[seagulls$SEX == "FEMALE",]$HEIGHT)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  seagulls[seagulls$SEX == "FEMALE", ]$HEIGHT
## W = 0.9989, p-value = 0.6345
```

We can see that height for both sexes passes as normally distributed.

Location and coast are describing almost the same thing since coast is more broad description of location (Maraetai and Waitawa are under east coast and Muriwai and Piha are under west coast). Locations are almost equally represented in our dataset:

```
summary(seagulls$LOCATION)
```

```
## MARAETAI  MURIWAI     PIHA  WAITAWA
##      673      589      647      578
```

Coast variable is also equaly distributed:

```
summary(seagulls$COAST)
```

```
## EAST WEST
## 1251 1236
```

Season is either winter or summer. There are a little more entries for summer than for winter, but the difference is miniscule:

```
summary(seagulls$SEASON)
```

```
## SUMMER WINTER
##   1313   1174
```

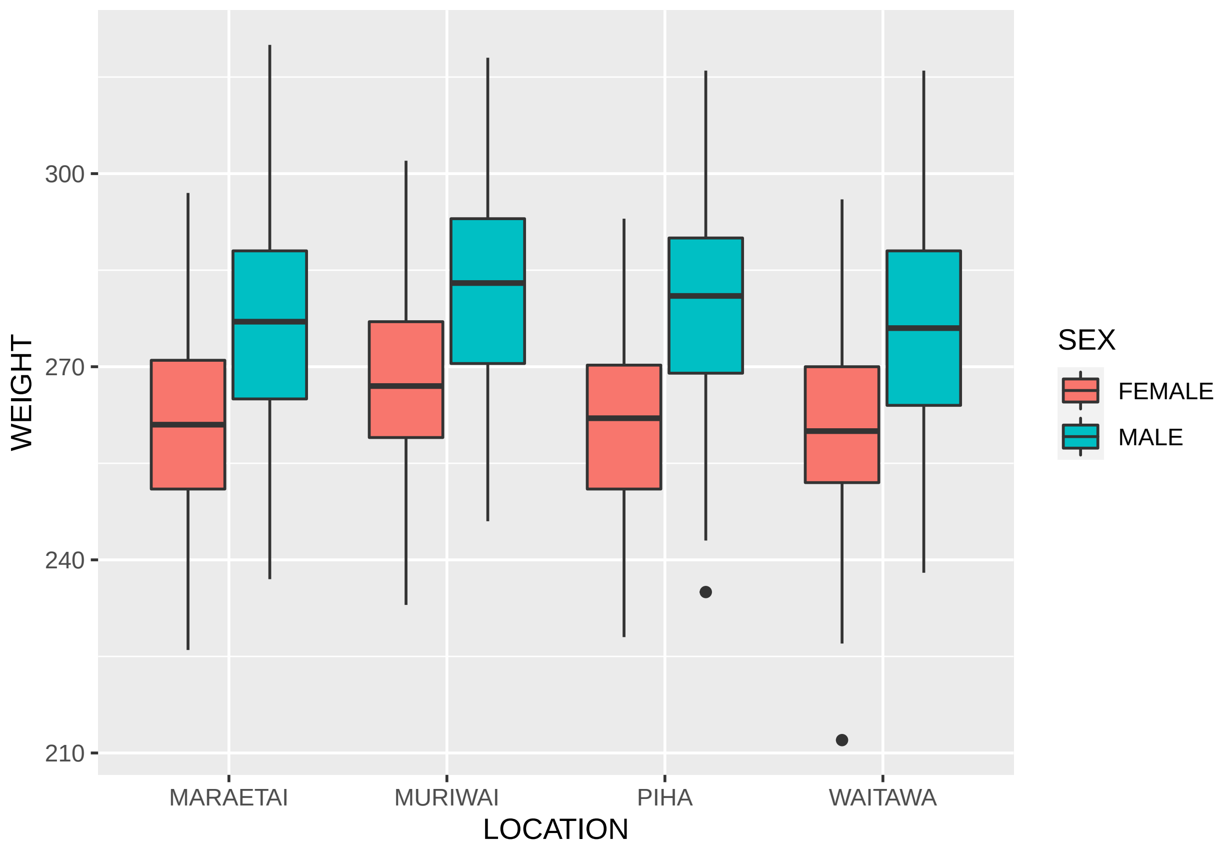
There are more females presented in our dataset but the difference can be ignored:

```
summary(seagulls$SEX)
```

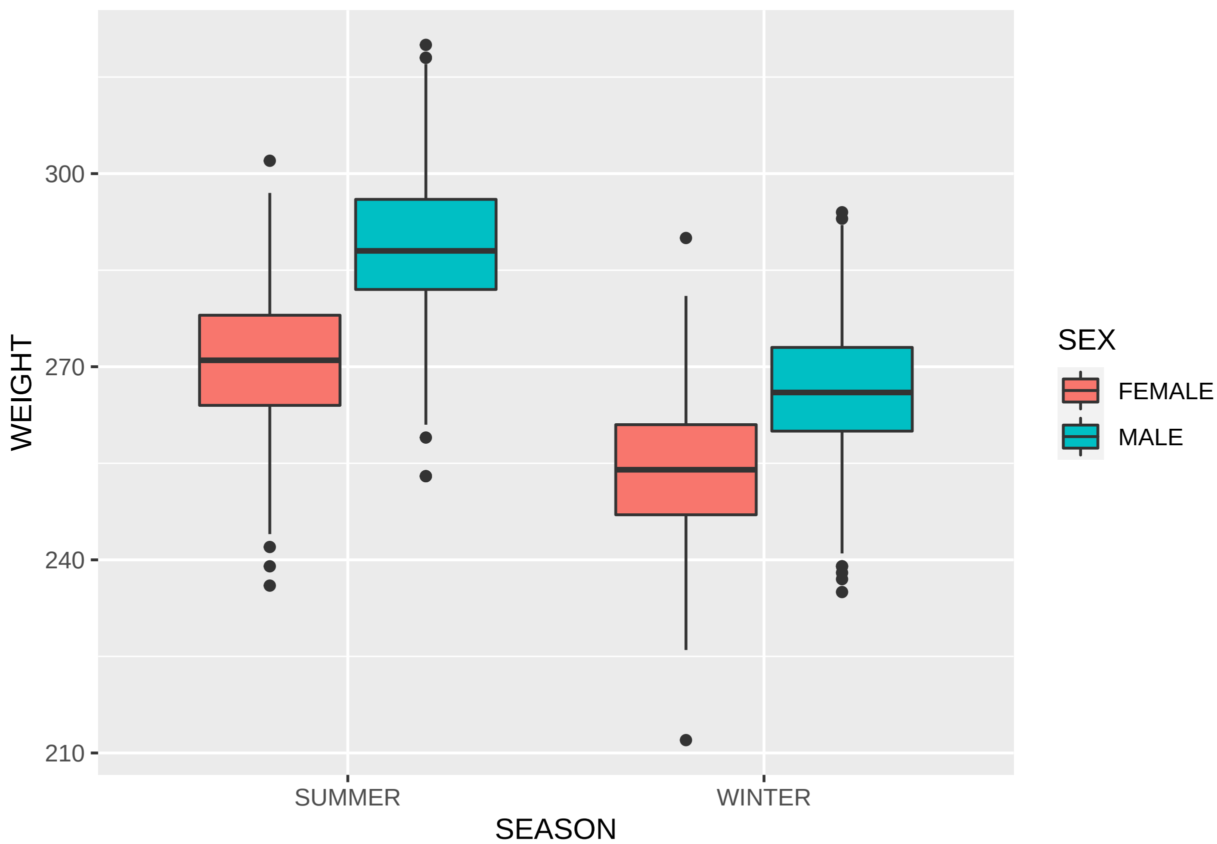
```
## FEMALE   MALE
##   1280   1207
```

We also drew some other plots representing how different variables are connected:

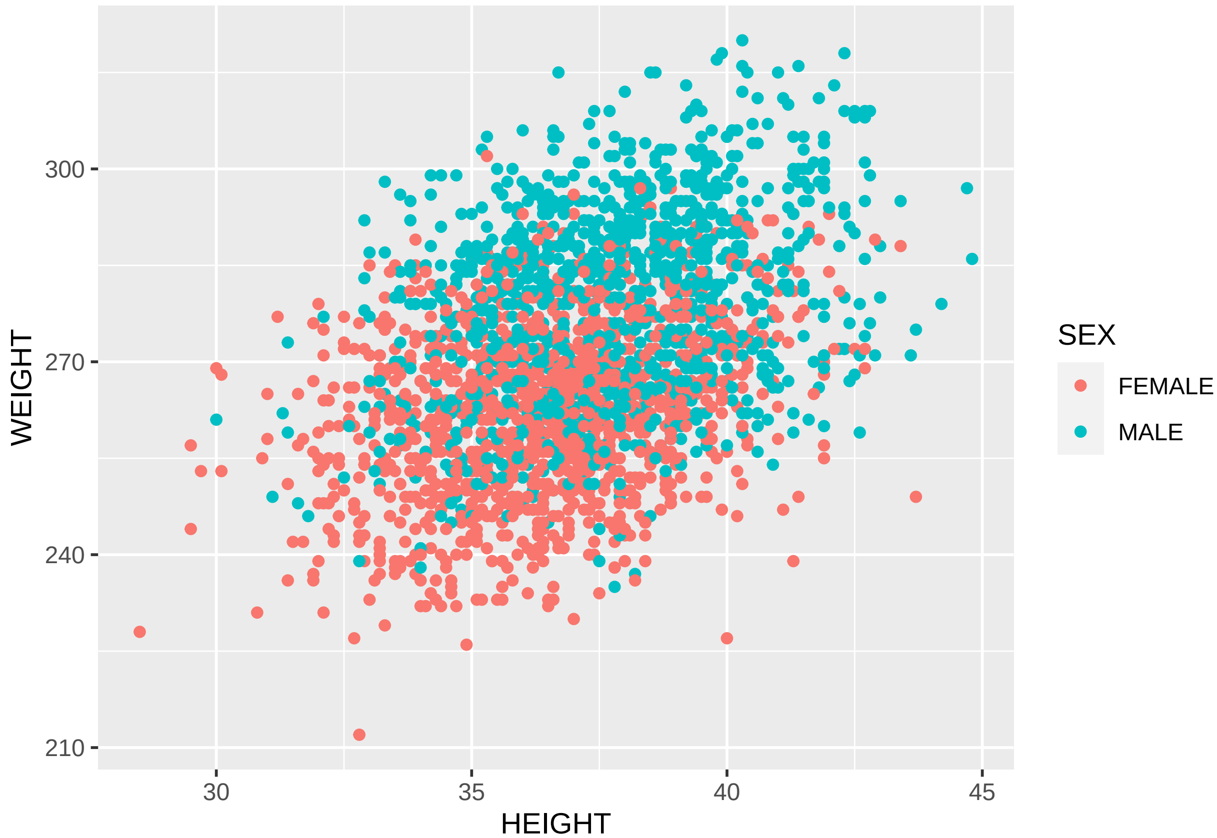
```
seagulls %>% ggplot()+
  geom_boxplot(aes(x = LOCATION, y = WEIGHT, fill = SEX))
```




```
seagulls %>% ggplot()+
  geom_boxplot(aes(x = SEASON, y = WEIGHT, fill = SEX))
```




```
seagulls %>% ggplot()+
  geom_point(aes(x = HEIGHT, y = WEIGHT, color = SEX))
```



Inference

Since we can divide our datasets in many ways, we can also check many differnet hypothesis.

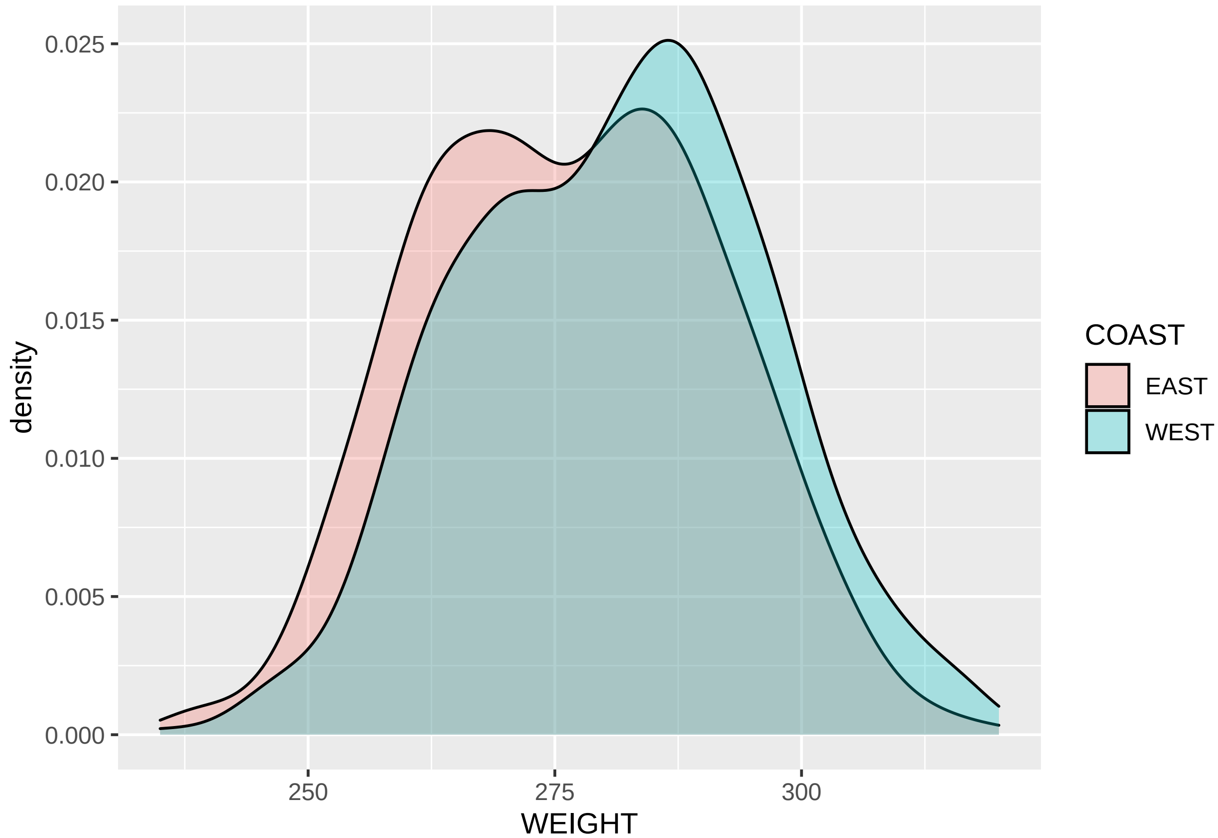
Is weight of males same on east and west coast?

We first divide our dataset into two smaller ones, which represent males from different coasts.

```
sg_east <- seagulls %>% filter(COAST == "EAST", SEX == "MALE")
sg_west <- seagulls %>% filter(COAST == "WEST", SEX == "MALE")
```

Next we need to check CLT conditions. Since samples were collected independently from one another, first condition is true. Next we need to check if both samples have sufficient size. There are 629 males from east and 578 males from west. Both samples are larger than 30, so second condition is also true. Then we need to check if any of samples is skewed. We can draw their distributions and see that they both are somewhat symmetrical.

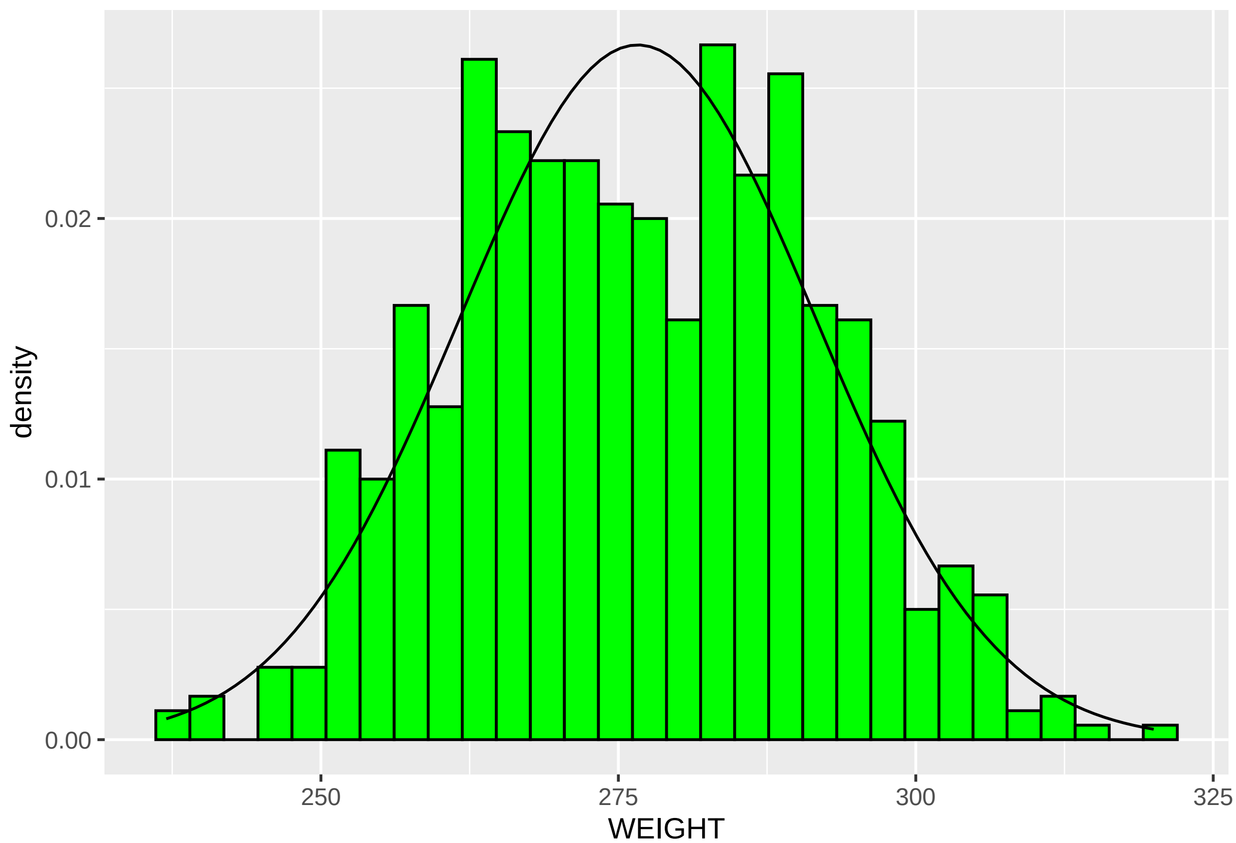
```
seagulls %>% filter(SEX == "MALE") %>% ggplot()+
  geom_density(aes(x = WEIGHT, fill = COAST), alpha = 0.3)
```



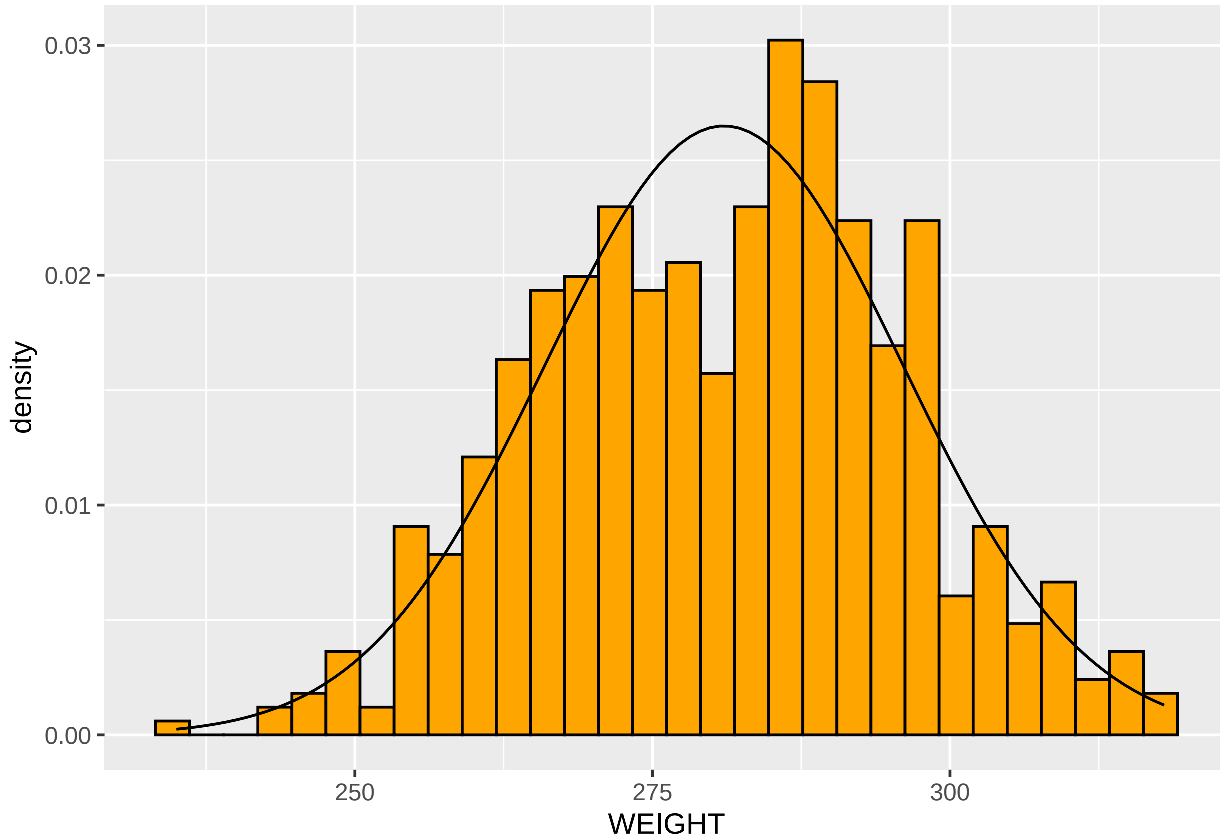
We can also calculate skewness of both distributions. Weight of males from east coast have skewness of 0.0370527 and males from west have skewness of -0.0483849. Both values are small, so we can safely say that neither distribution is strongly skewed.

We also need to check wether cases from groups are independant from each other. Since they were collected on different locations, they are independant. We can check if both groups are normally distributed. For that we can draw histogram of weights and overlay it with normal distribution with same average and standard deviation:

```
east.mean <- mean(sg_east$WEIGHT)
east.sd <- sd(sg_east$WEIGHT)
sg_east %>% ggplot()+
  geom_histogram(aes(x = WEIGHT, y = ..density..), fill = "green", color = "black")+
  stat_function(fun = dnorm, args = list(mean = east.mean, sd = east.sd))
```




```
west.mean <- mean(sg_west$WEIGHT)
west.sd <- sd(sg_west$WEIGHT)
sg_west %>% ggplot()+
  geom_histogram(aes(x = WEIGHT, y = ..density..), fill = "orange", color = "black")+
  stat_function(fun = dnorm, args = list(mean = west.mean, sd = west.sd))
```



Neither distribution seem normally distributed. We can further test that hypothesis with normallity test:

```
shapiro.test(sg_east$WEIGHT)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  sg_east$WEIGHT
## W = 0.99247, p-value = 0.002899
```

```
shapiro.test(sg_west$WEIGHT)
```

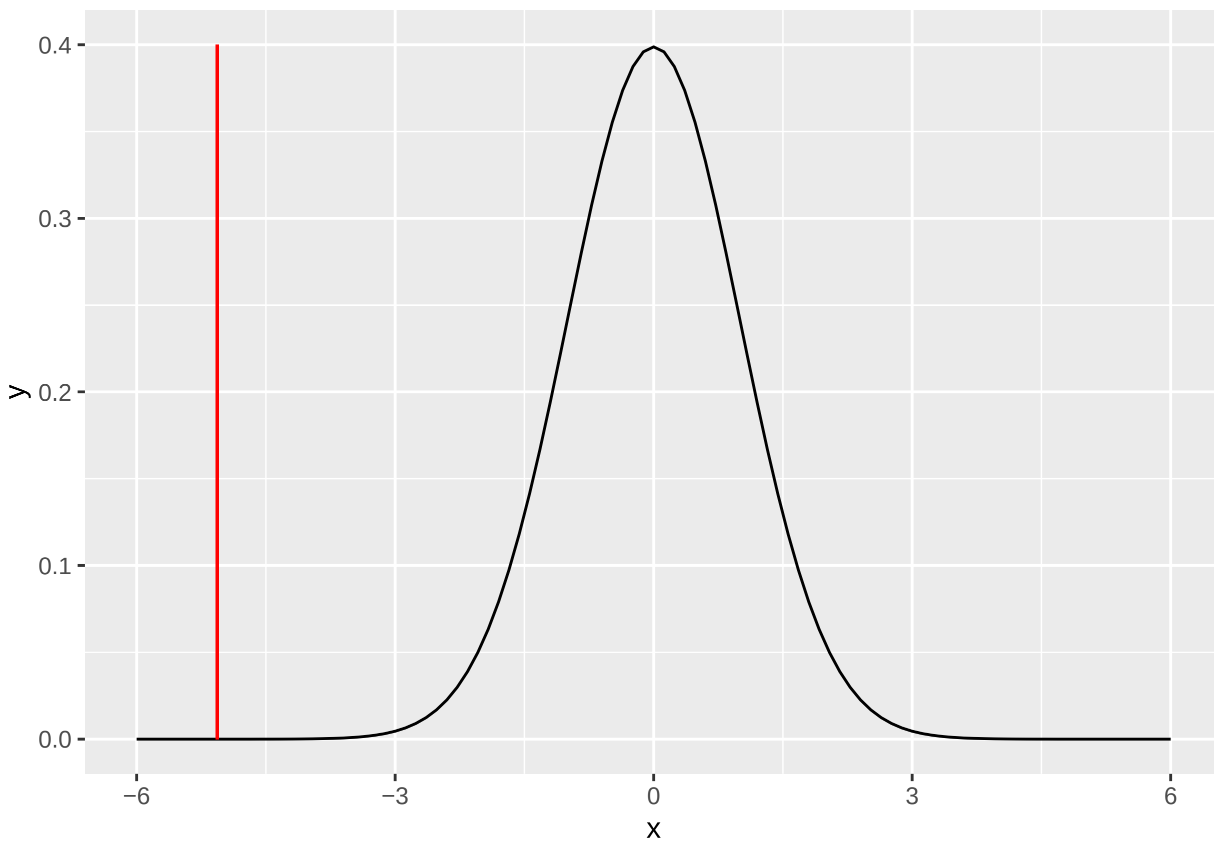
```
##
##  Shapiro-Wilk normality test
##
## data:  sg_west$WEIGHT
## W = 0.99417, p-value = 0.0257
```

Neither group has normal distribution, but they are symmetrical, so we will continue with our hypothesis testing.

Then we set-up the hypothesis: H_0 : Mean weight is the same in east and west coast: $mean_{east} - mean_{west} = 0$ H_A : Mean weight is not the same in east and west coast: $mean_{east} - mean_{west} \neq 0$ We set a threshold value $\alpha = 0.05$.

We calculate our point estimate, standard error, and t-score and plot it:

```
point_estimate <- east.mean - west.mean
SE <- sqrt(east.sd ^ 2 / nrow(sg_east) + west.sd ^ 2 / nrow(sg_west))
df <- min(nrow(sg_east) - 1, nrow(sg_west) - 1)
t_score <- point_estimate / SE

ggplot(data.frame(x = seq(-6, 6, length = 200)), aes(x = x))+
  stat_function(fun = dt, args = list(df = df))+
  geom_segment(aes(x = t_score, y = 0, xend = t_score, yend = 0.4), color="red")
```



We can see that our t-score (red line) falls to the left of student's t-distribution, so our null hypothesis is very likely false. We can further confirm that with our p-value calculation:

```
p_value <- 2 * pt(t_score, df)
```

Since p-value is smaller than α ($5.5286726 \times 10^{-7} < 0.05$), we reject H_0 in favor of H_A. Seagulls on east and west coast do not weight the same. Because our point estimate is negative, we can say that seagulls on west coast weight more than seagulls on east coast.