# Regression

Avalon C.S. Owens, Eric R. Scott

11/30/2018

# Load in ur stuff!

```
library(ggplot2)
library(dplyr)
lion <- read.csv("LionAge.csv")
head(lion)
```

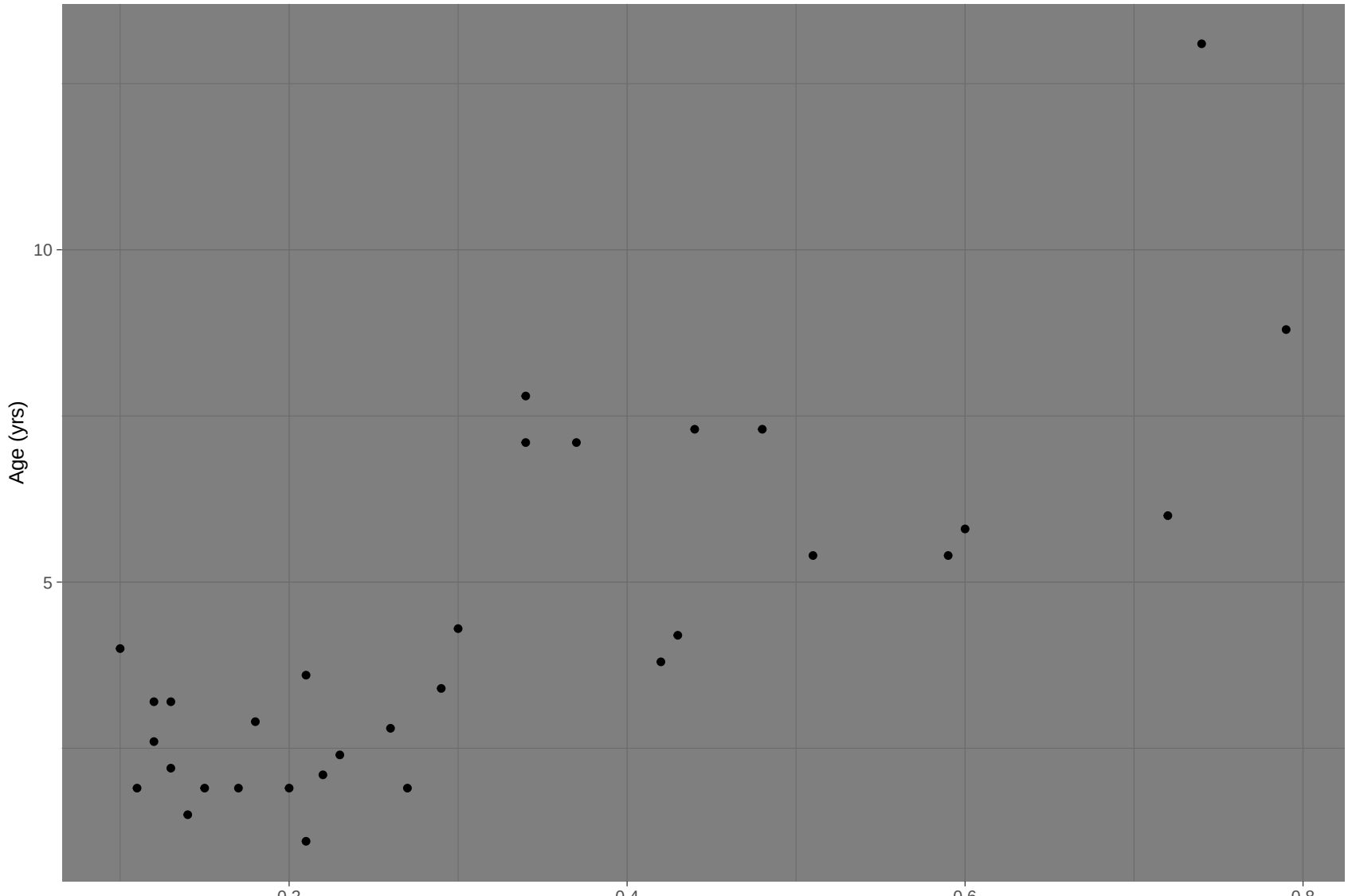
```
##   age proportion.black
## 1 1.1              0.21
## 2 1.5              0.14
## 3 1.9              0.11
## 4 2.2              0.13
## 5 2.6              0.12
## 6 3.2              0.13
```

# Plan for today

- Setting up a regression model with `lm()`
- Checking your regression assumptions
  - Residuals plots
  - Normal probability plots
- Doing a regression test with `summary()`
- Plotting best-fit lines with `geom_smooth()`

# A review of correlation

```
ggplot(lion, aes(x = proportion.black, y = age)) +
  geom_point() + theme_dark() + labs(x = "Proportion black
```



# A review of correlation

```
cor <- cor.test(lion$proportion.black, lion$age, method = '
cor$estimate #Pearson correlation coefficient
```

```
##       cor
## 0.7898272
```

```
cor$estimate^2 #R-squared
```

```
##       cor
## 0.623827
```

# Setting up a regression model with `lm()`

$$Y = \alpha + \beta * X$$

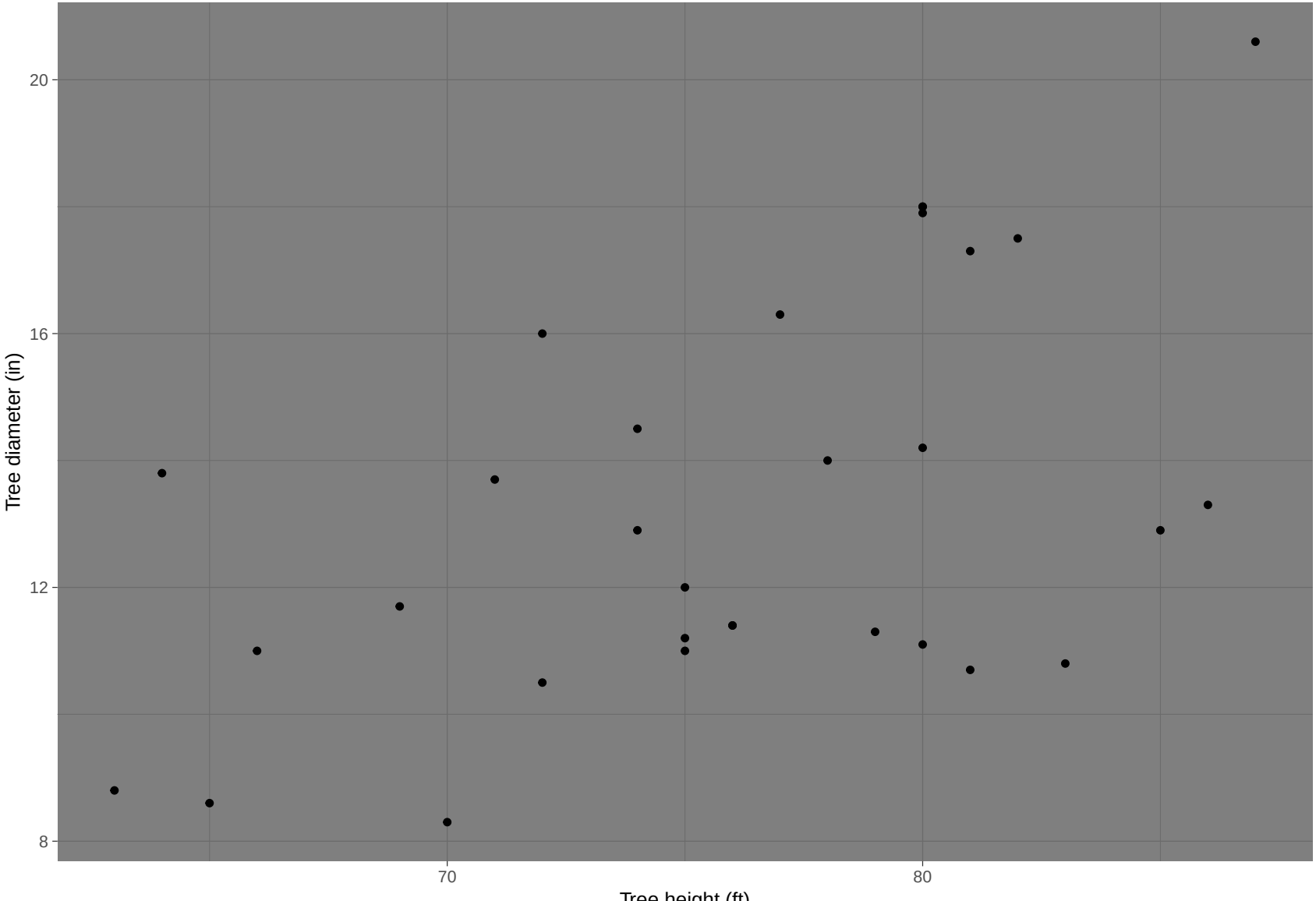
```
lion.model <- lm(age ~ proportion.black, data = lion) #lm s
lion.model
```

```
##
## Call:
## lm(formula = age ~ proportion.black, data = lion)
##
## Coefficients:
##       (Intercept)  proportion.black
##             0.879            10.647
```

# Setting up a regression model with `lm()`

Make your own regression model!

See if you can predict tree girth as a function of tree height



# Setting up a regression model with `lm()`

Make your own regression model!

```
tree.model = lm(Girth ~ Height, data = trees)
tree.model
```

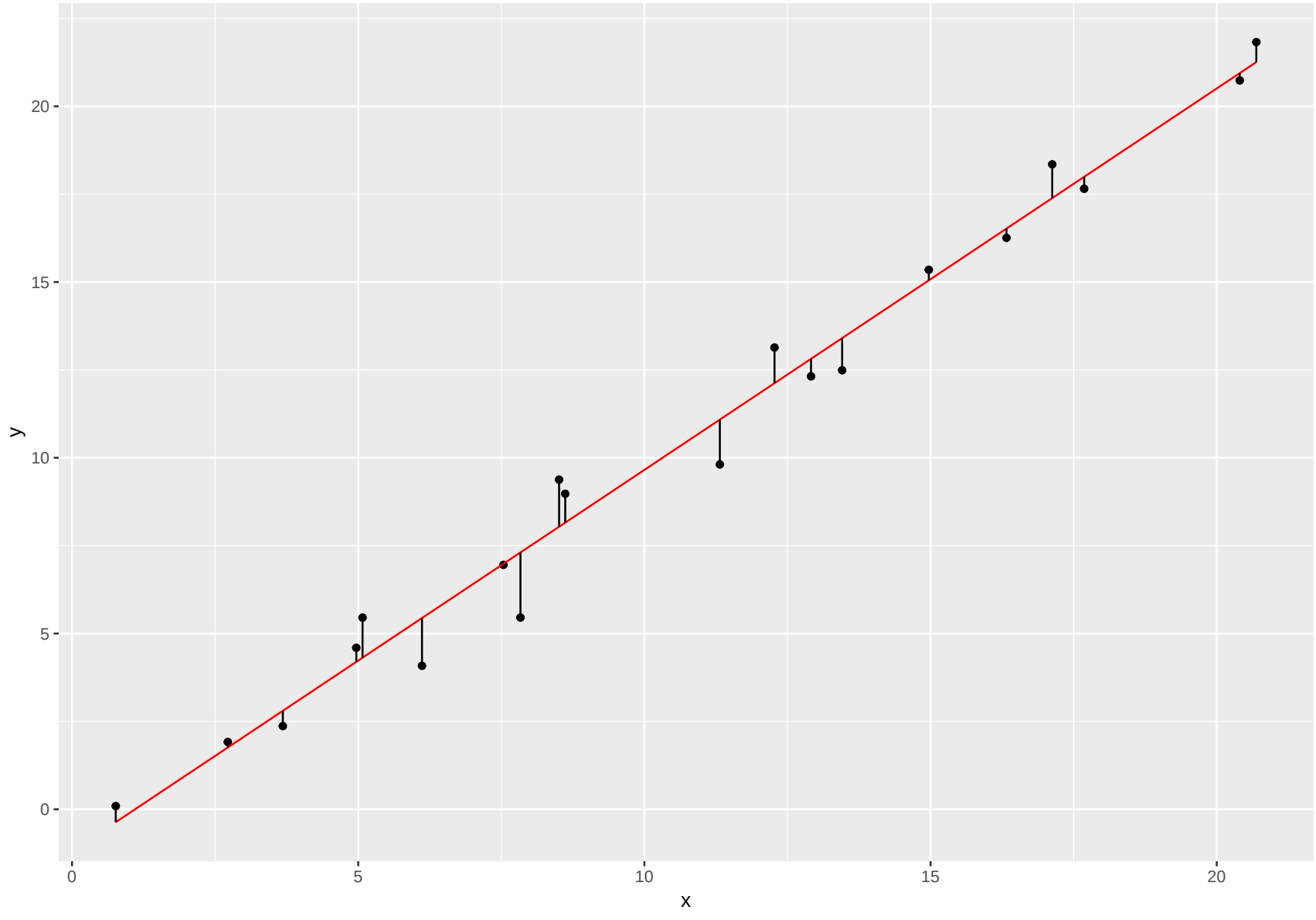
```
##
## Call:
## lm(formula = Girth ~ Height, data = trees)
##
## Coefficients:
## (Intercept)       Height
##     -6.1884       0.2557
```

# Checking your regression assumptions (17.5)

1. LINEARITY: a true linear relationship between $X$ and $Y$ in the underlying population, such that $Y = \alpha + \beta * X$
2. NORMALITY: at every value of $X$, $Y$ is normally distributed
3. VARIANCE: at each value of $X$, the variance of $Y$ is equal

# Residuals plots

# Checking your regression assumptions



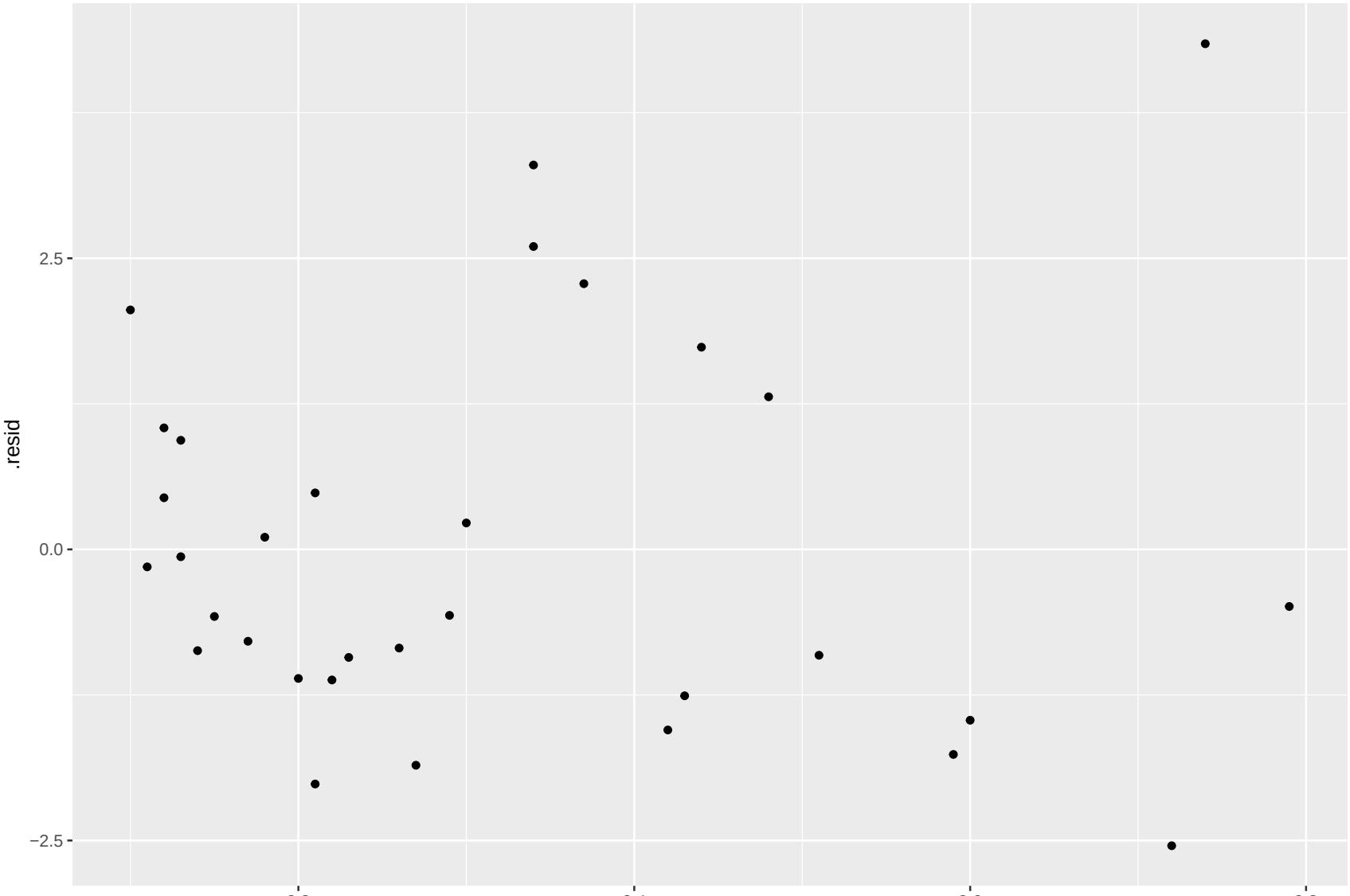
# Checking your regression assumptions

```
fortify(lion.model)
```

```
##      age proportion.black       .hat   .sigma       .cooks
## 1    1.1              0.21 0.04154830 1.653704 3.296860e-0
## 2    1.5              0.14 0.05840900 1.689114 8.944936e-0
## 3    1.9              0.11 0.06808971 1.697046 3.175350e-0
## 4    2.2              0.13 0.06147226 1.697249 4.994137e-0
## 5    2.6              0.12 0.06469916 1.695156 2.610050e-0
## 6    3.2              0.13 0.06147226 1.687765 1.099821e-0
## 7    3.2              0.12 0.06469916 1.685428 1.445531e-0
## 8    2.9              0.18 0.04779243 1.697176 1.033739e-0
## 9    2.4              0.23 0.03820378 1.688175 6.383662e-0
## 10   2.1              0.22 0.03979421 1.683936 9.744761e-0
## 11   1.9              0.20 0.04346603 1.684194 1.047963e-0
## 12   1.9              0.17 0.05020111 1.690621 6.220170e-0
## 13   1.9              0.15 0.05550939 1.693719 3.707729e-0
## 14   1.9              0.27 0.03347848 1.660784 2.211134e-0
## 15   2.8              0.26 0.03441434 1.689723 4.757387e-0
```

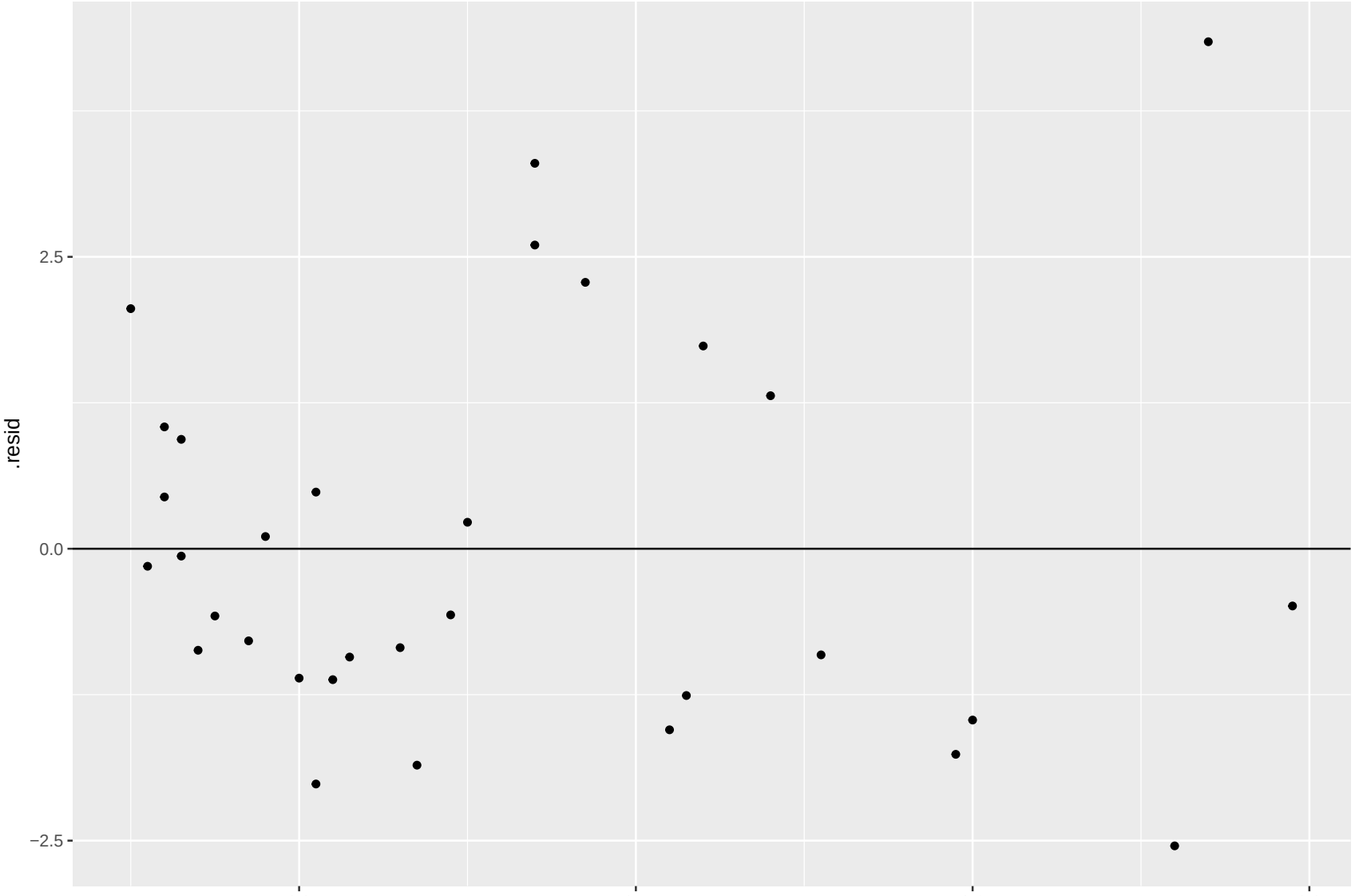
# Checking your regression assumptions

```
ggplot(fortify(lion.model), aes(x = proportion.black, y =
  geom_point()
```



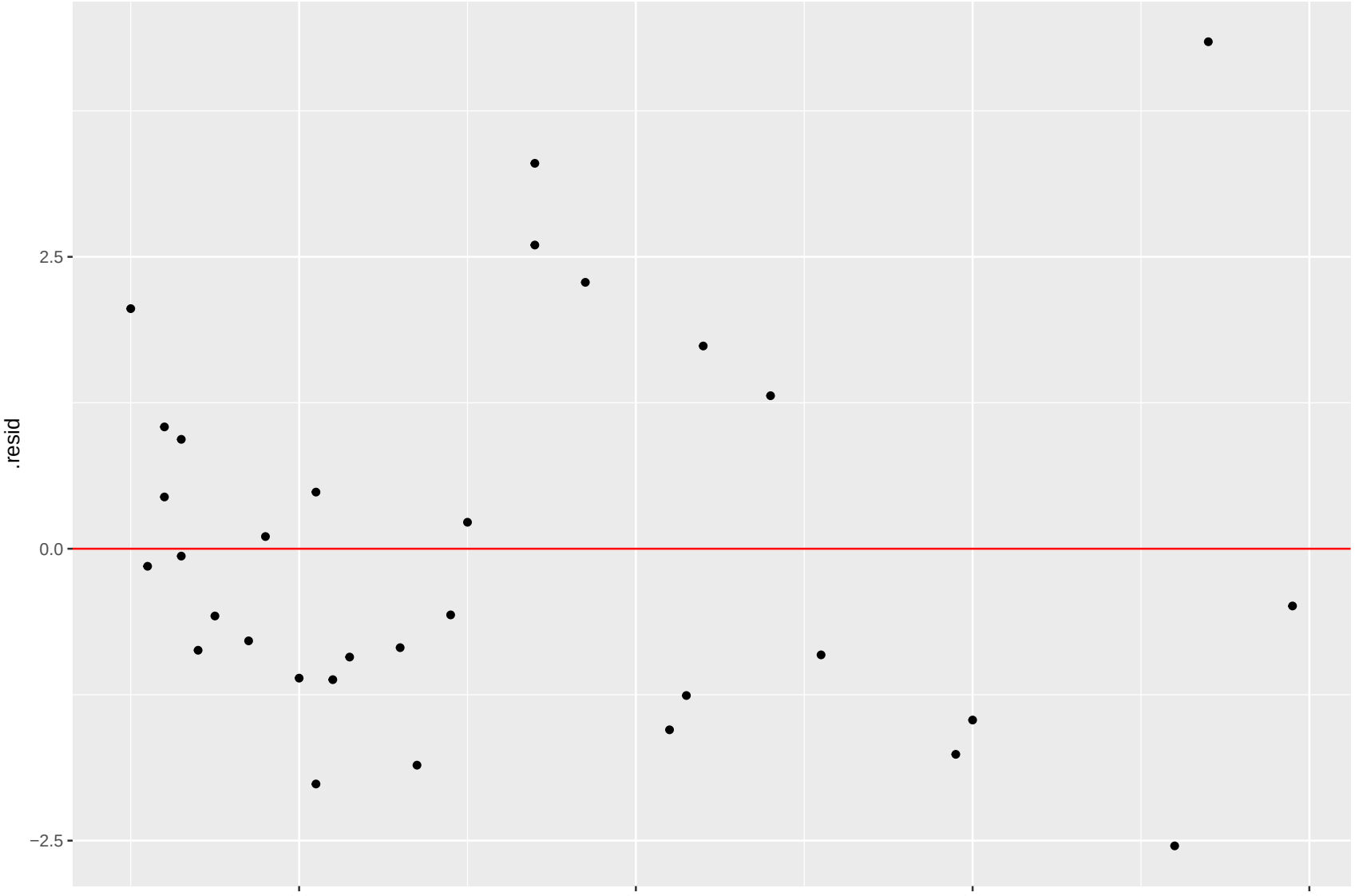
# Checking your regression assumptions

```
ggplot(fortify(lion.model), aes(x = proportion.black, y =
  geom_point() + geom_hline(yintercept = 0)
```



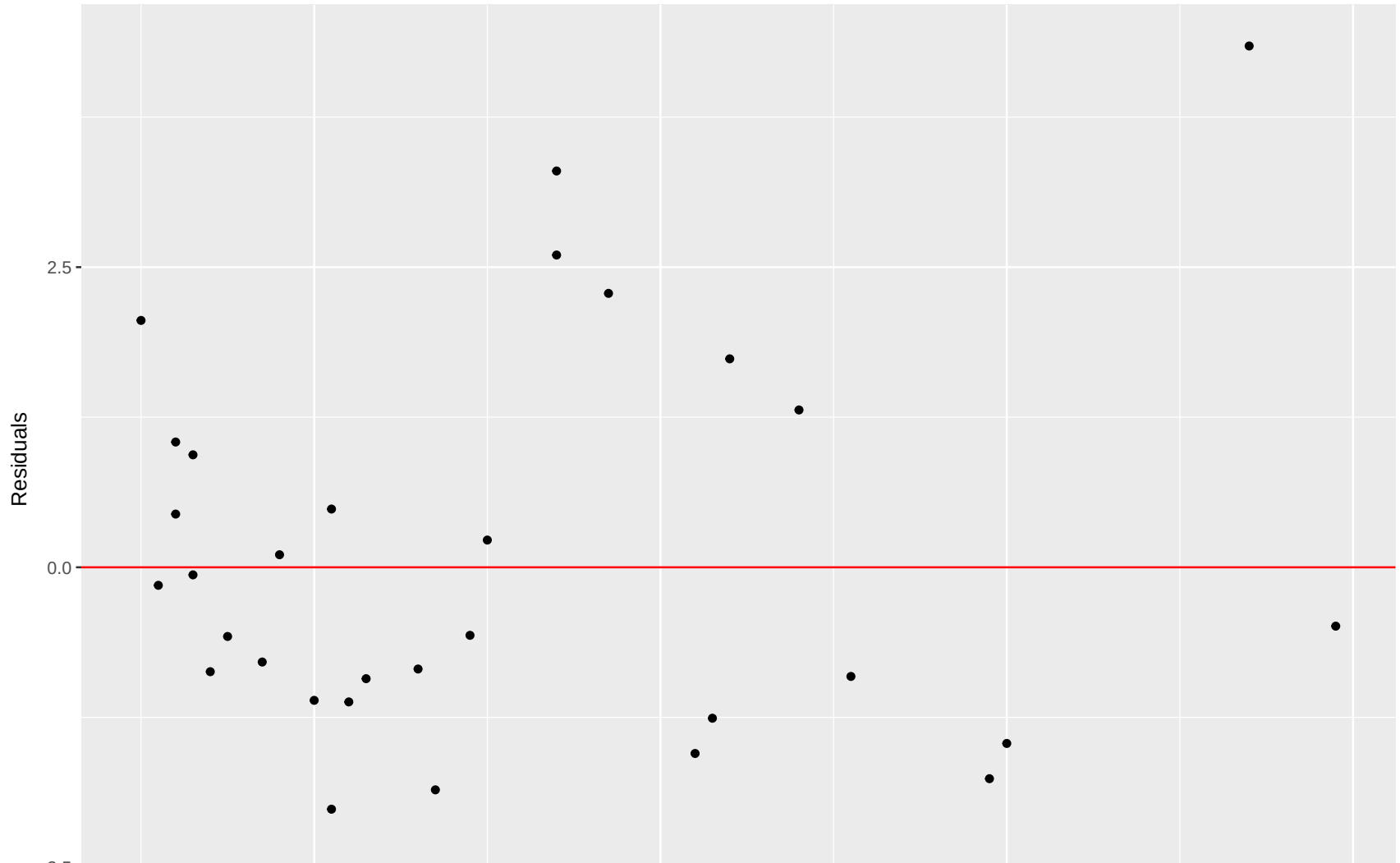
# Checking your regression assumptions

```
ggplot(fortify(lion.model), aes(x = proportion.black, y =
  geom_point() + geom_hline(yintercept = 0, color = "red")
```



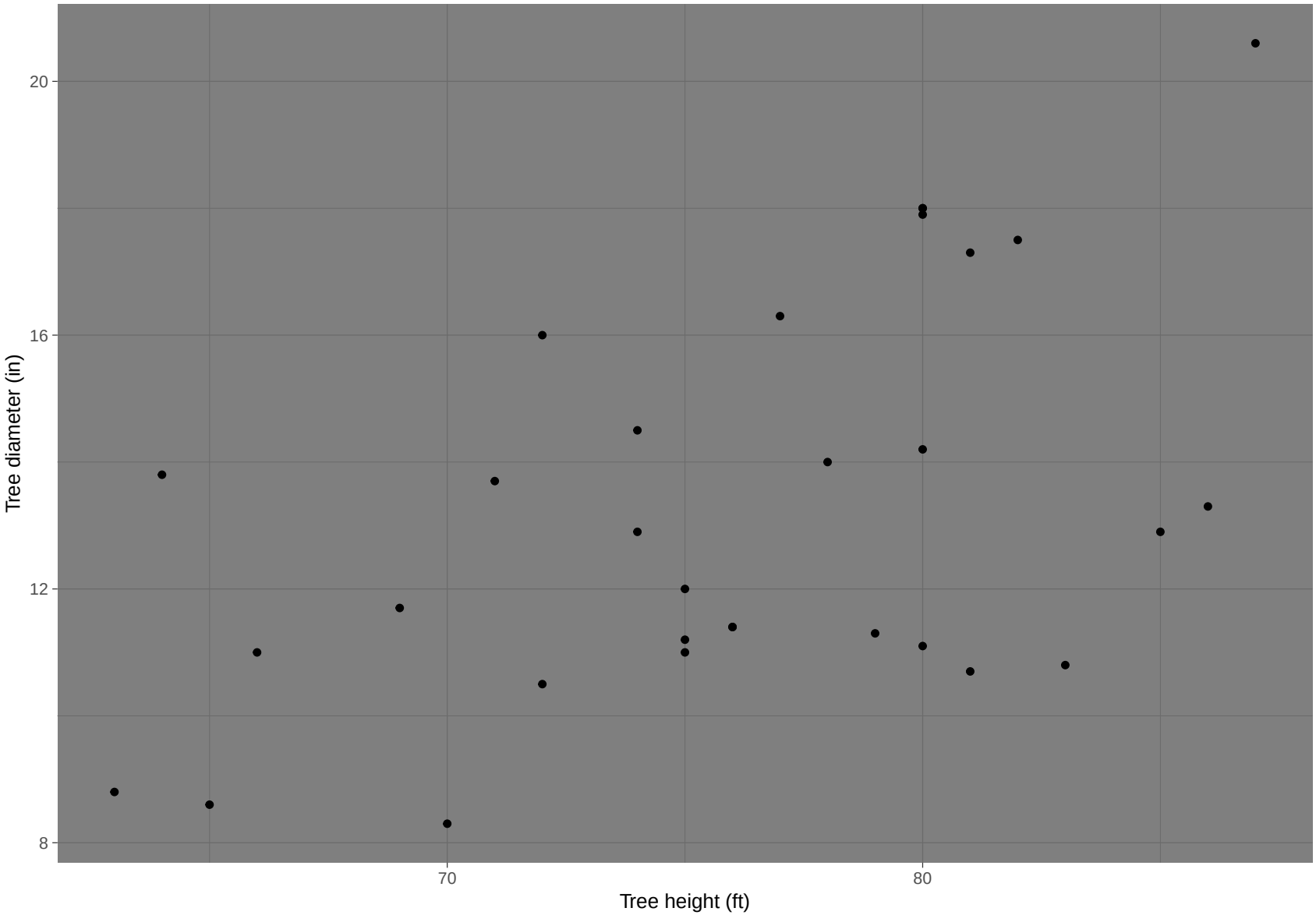
# Checking your regression assumptions

```
ggplot(fortify(lion.model), aes(x = proportion.black, y =
  geom_point() + geom_hline(yintercept = 0, color = "red")
  labs(x = "Proportion black on nose", y = "Residuals")
```



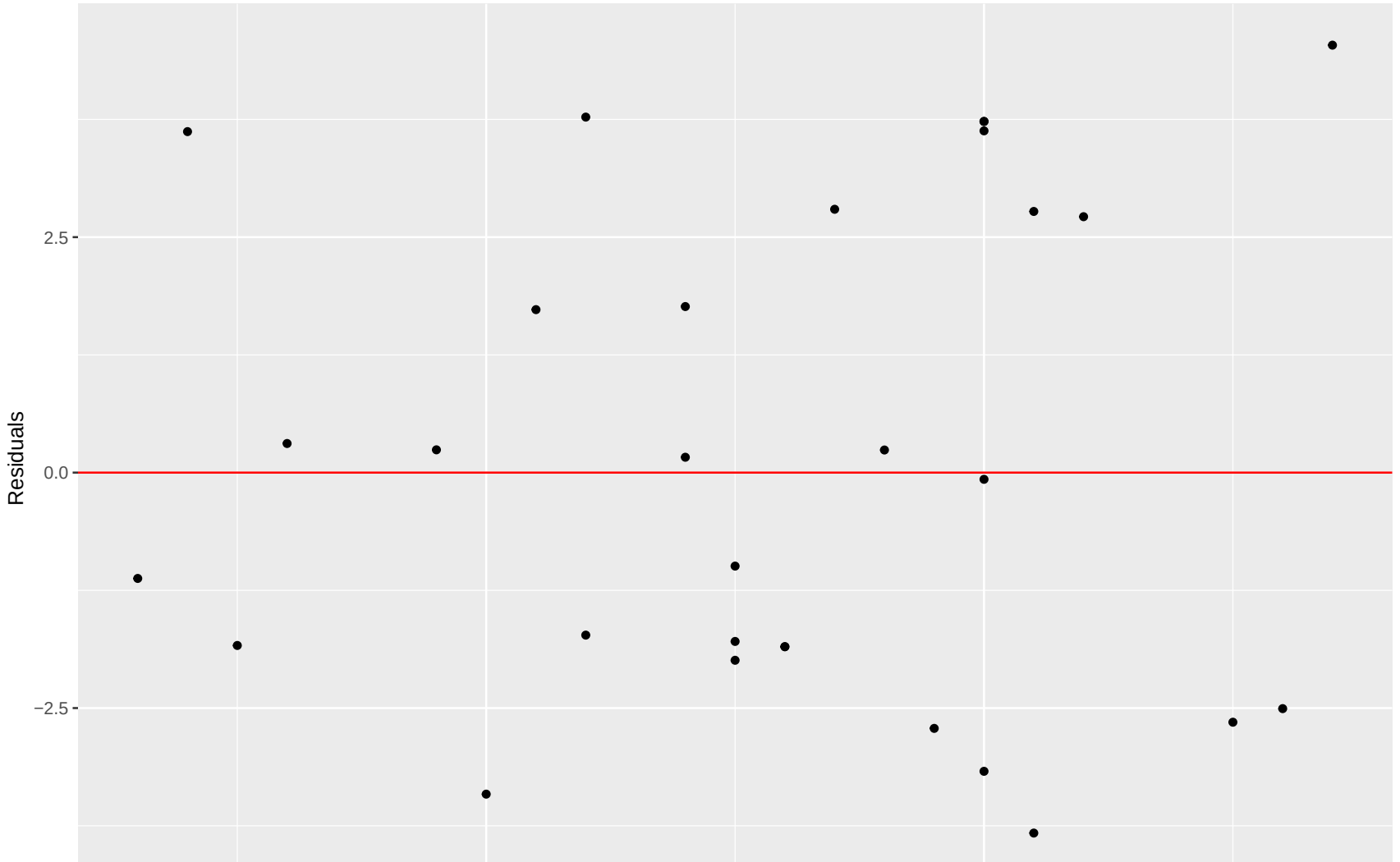
# Checking your regression assumptions

Check the assumptions of your regression model with a plot of your own!



# Checking your regression assumptions

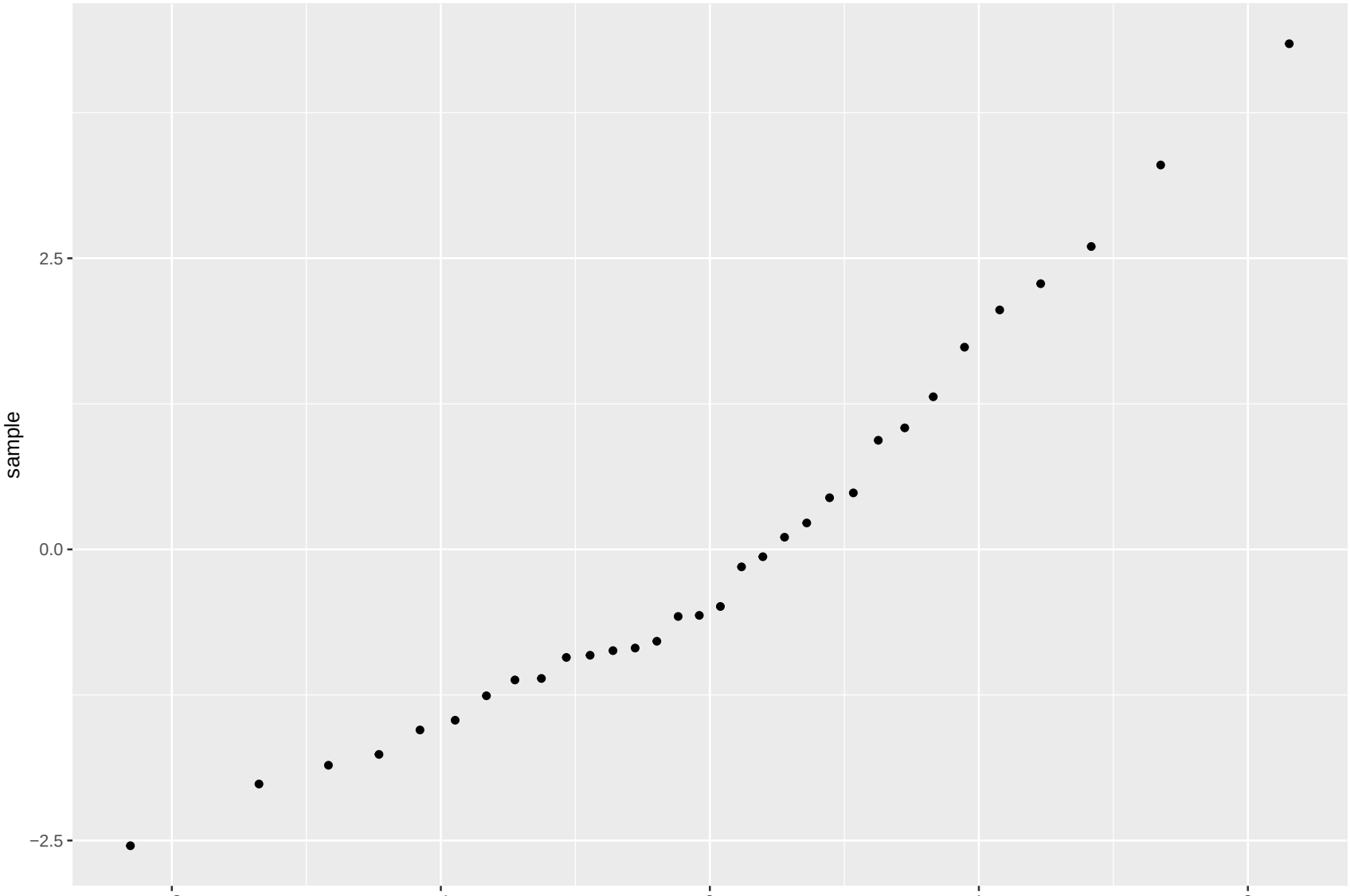
```
ggplot(tree.model, aes(x = Height, y = .resid)) +
  geom_point() + geom_hline(yintercept = 0, color = "red") +
  labs(x = "Height (ft)", y = "Residuals")
```



# Normal probability plots

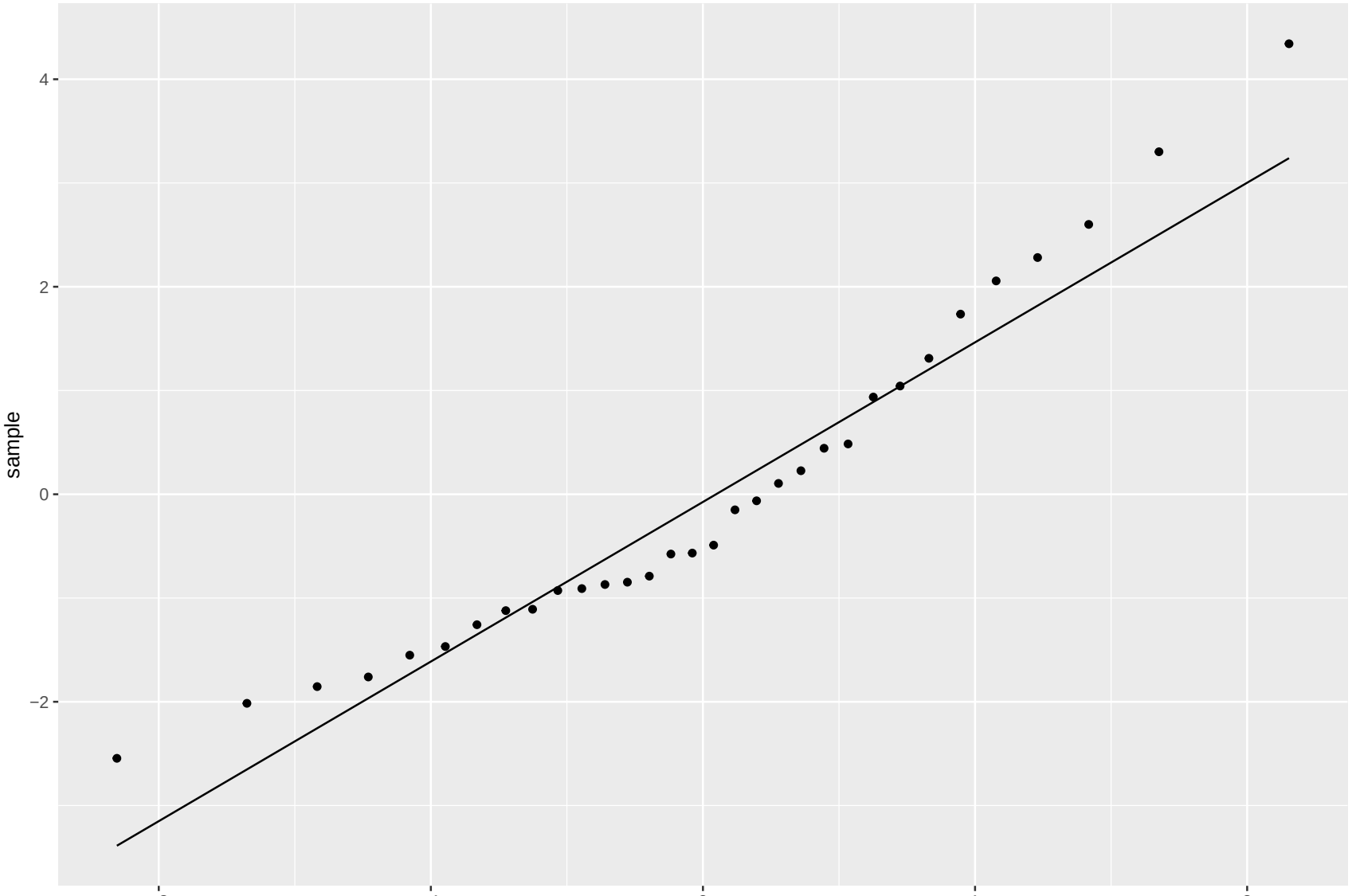
# Checking your regression assumptions

```
ggplot(fortify(lion.model), aes(sample = .resid)) +
  geom_qq()
```



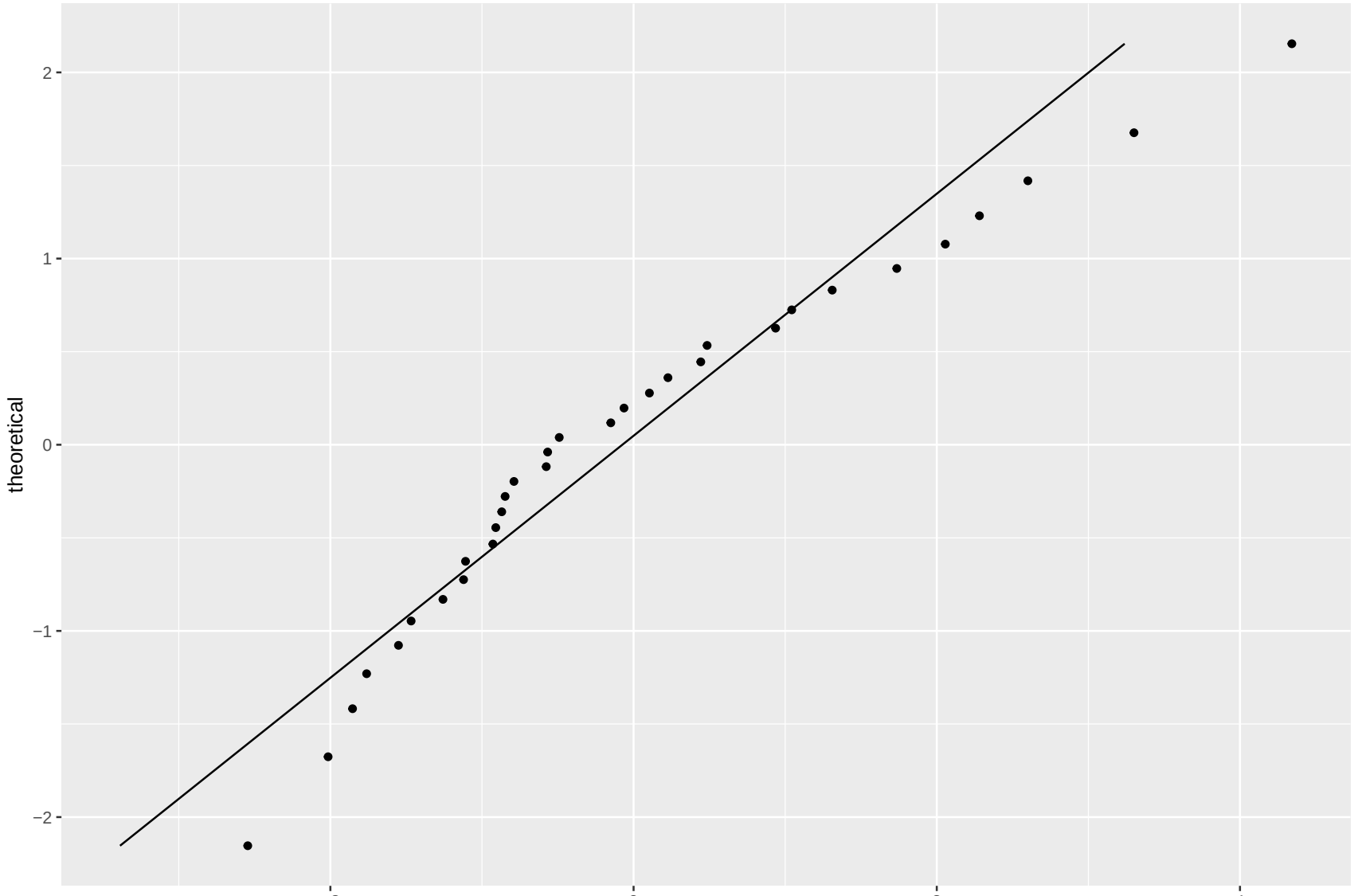
# Checking your regression assumptions

```
ggplot(fortify(lion.model), aes(sample = .resid)) +
  geom_qq() + geom_qq_line()
```



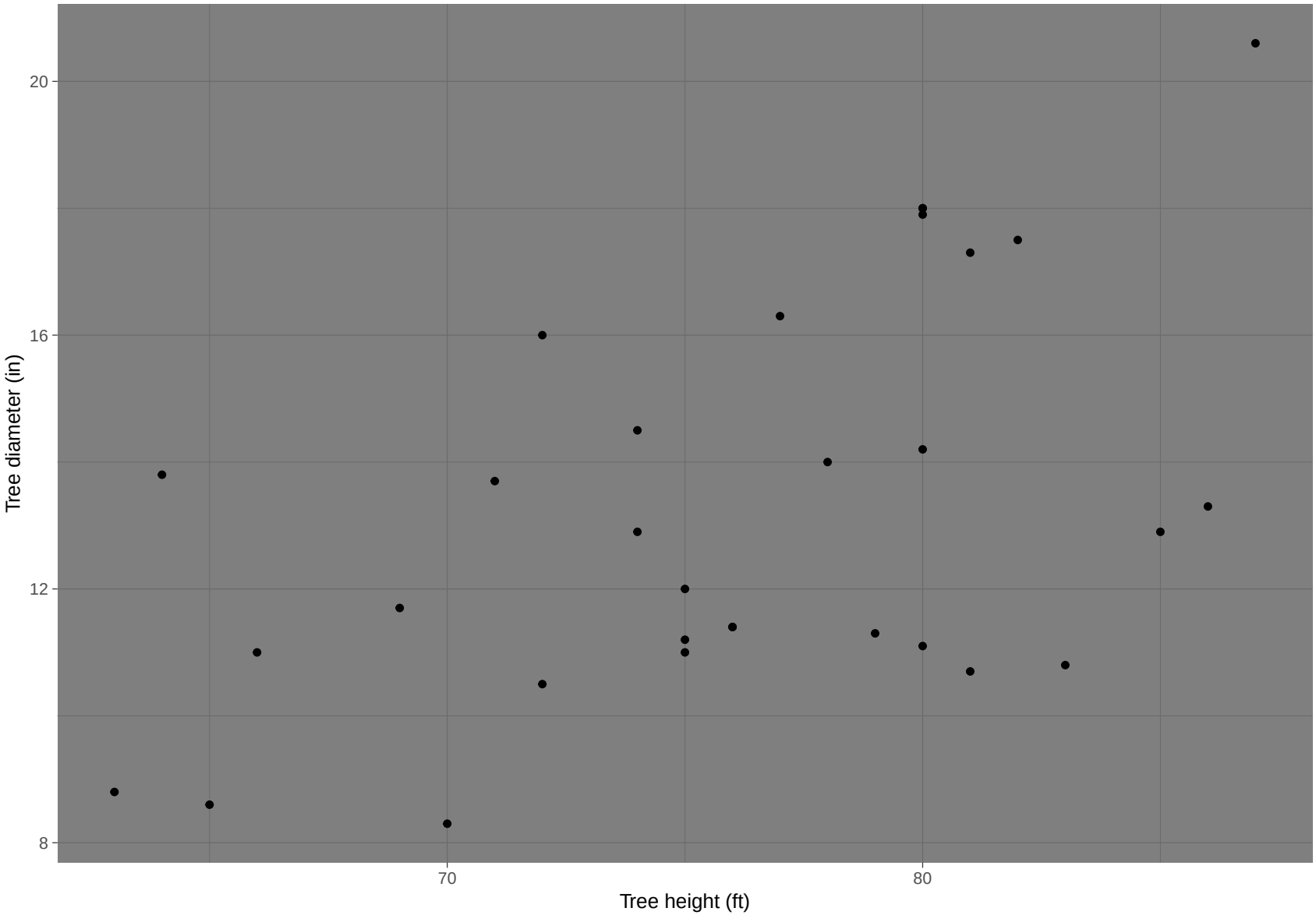
# Checking your regression assumptions

```
ggplot(fortify(lion.model), aes(sample = .resid)) +
  geom_qq() + geom_qq_line() +  coord_flip()
```



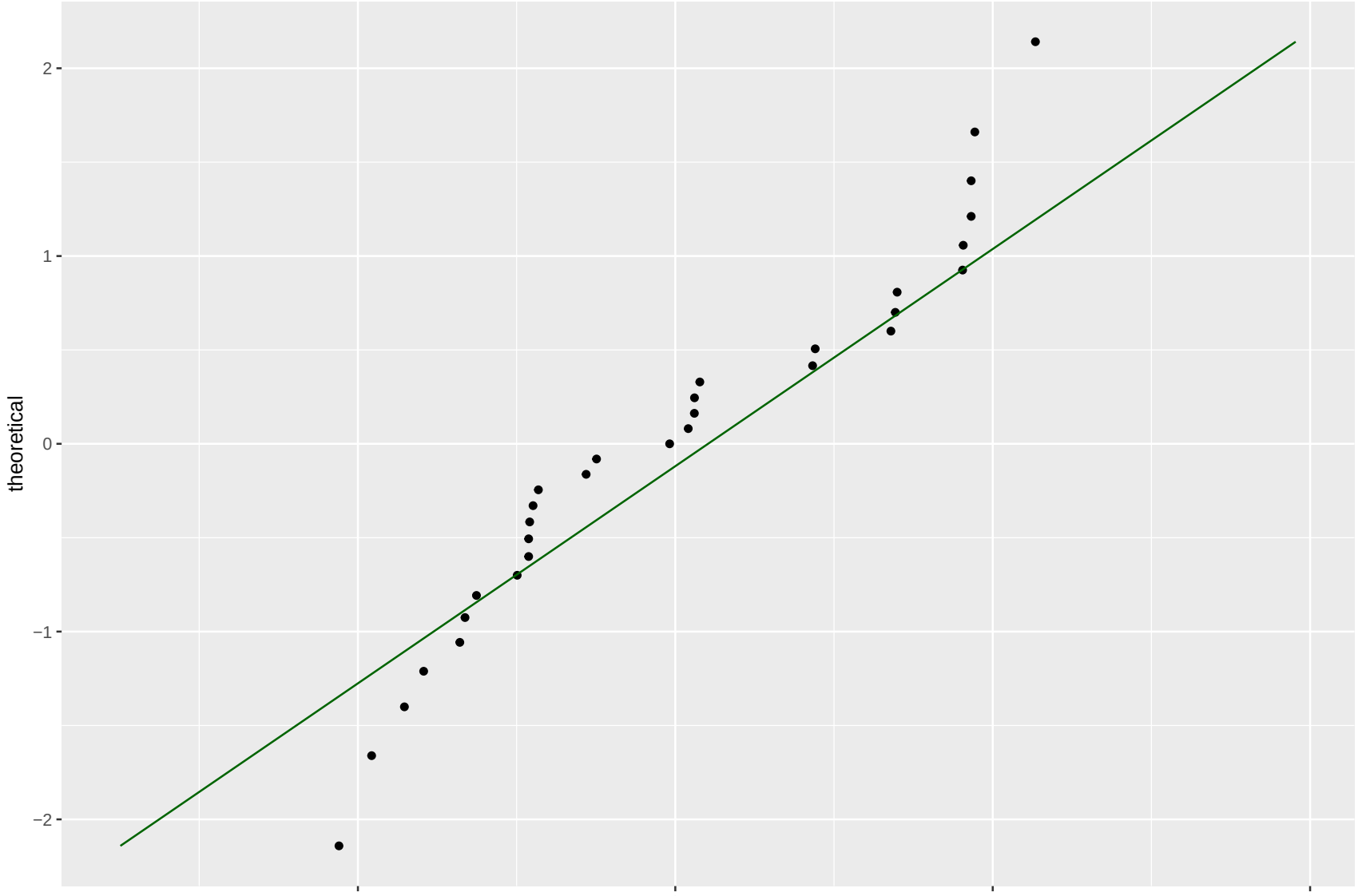
# Checking your regression assumptions

Check the assumptions of your regression model with a plot of your own!



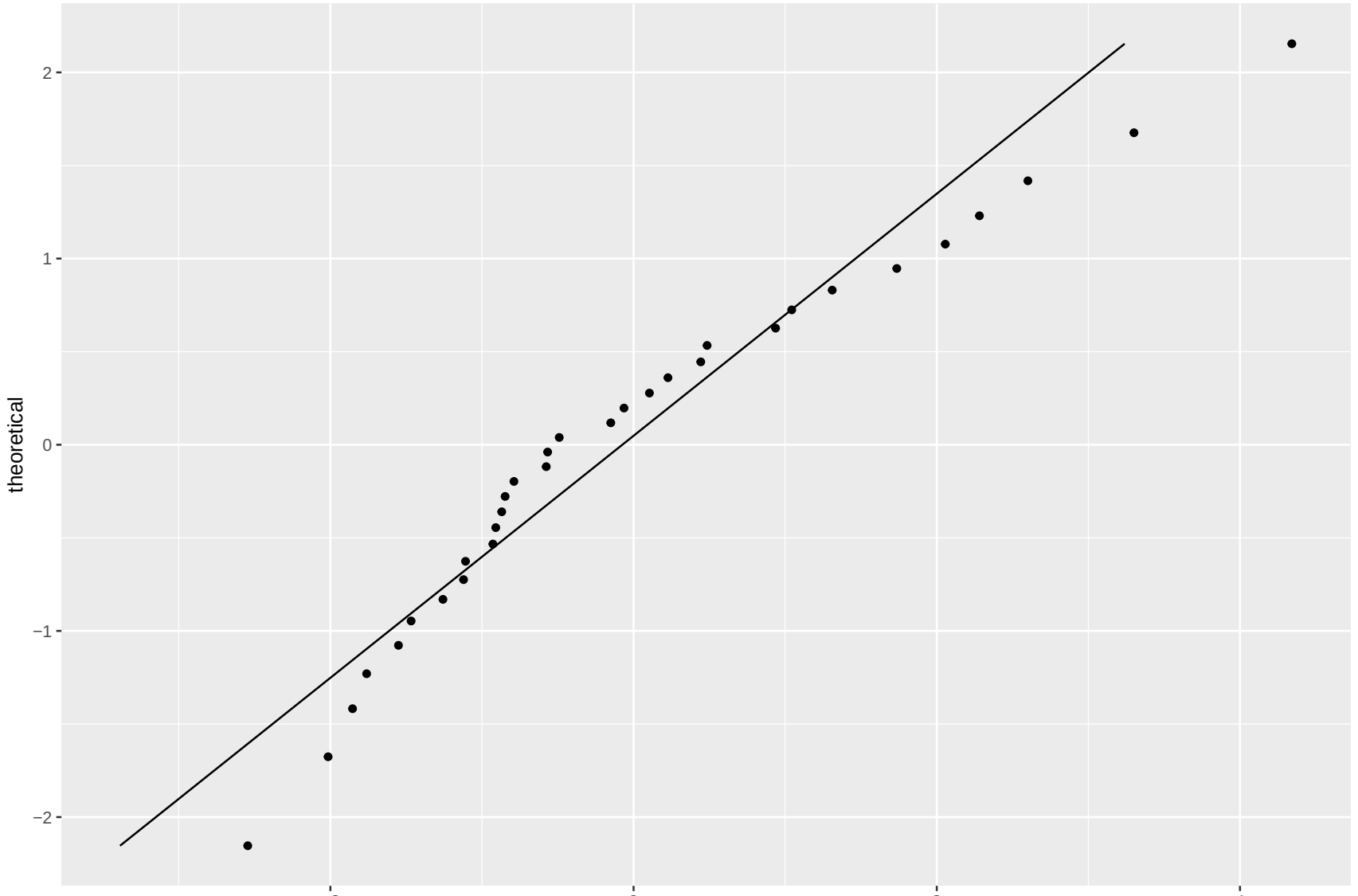
# Checking your regression assumptions

```
ggplot(tree.model, aes(sample = .resid)) +
  geom_qq() + geom_qq_line(color = "darkgreen") + coord_fli
```



# Checking your regression assumptions

```
ggplot(fortify(lion.model), aes(sample = .resid)) +
  geom_qq() + geom_qq_line() + coord_flip()
```



# Transforming your data

```
lion.2 <- lion %>% mutate(age.2 = log(age))
lion.2
```

```
##      age proportion.black      age.2
## 1    1.1              0.21 0.09531018
## 2    1.5              0.14 0.40546511
## 3    1.9              0.11 0.64185389
## 4    2.2              0.13 0.78845736
## 5    2.6              0.12 0.95551145
## 6    3.2              0.13 1.16315081
## 7    3.2              0.12 1.16315081
## 8    2.9              0.18 1.06471074
## 9    2.4              0.23 0.87546874
## 10   2.1              0.22 0.74193734
## 11   1.9              0.20 0.64185389
## 12   1.9              0.17 0.64185389
## 13   1.9              0.15 0.64185389
## 14   1.9              0.27 0.64185389
```

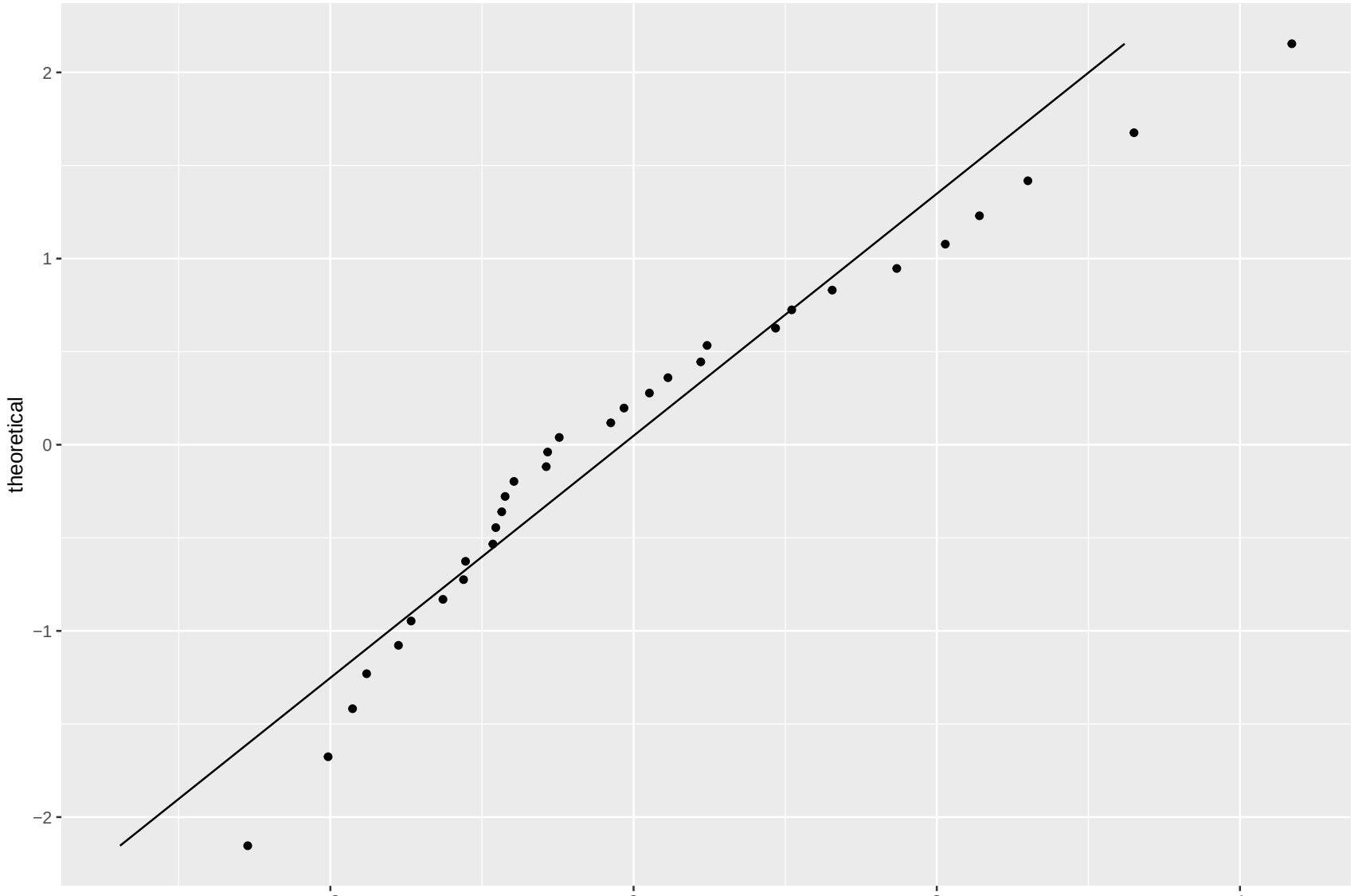
# Transforming your data

```
lion.king.2 <- lm(age.2 ~ proportion.black, data = lion.2)
lion.king.2
```

```
##
## Call:
## lm(formula = age.2 ~ proportion.black, data = lion.2)
##
## Coefficients:
##       (Intercept)  proportion.black
##             0.545             2.308
```

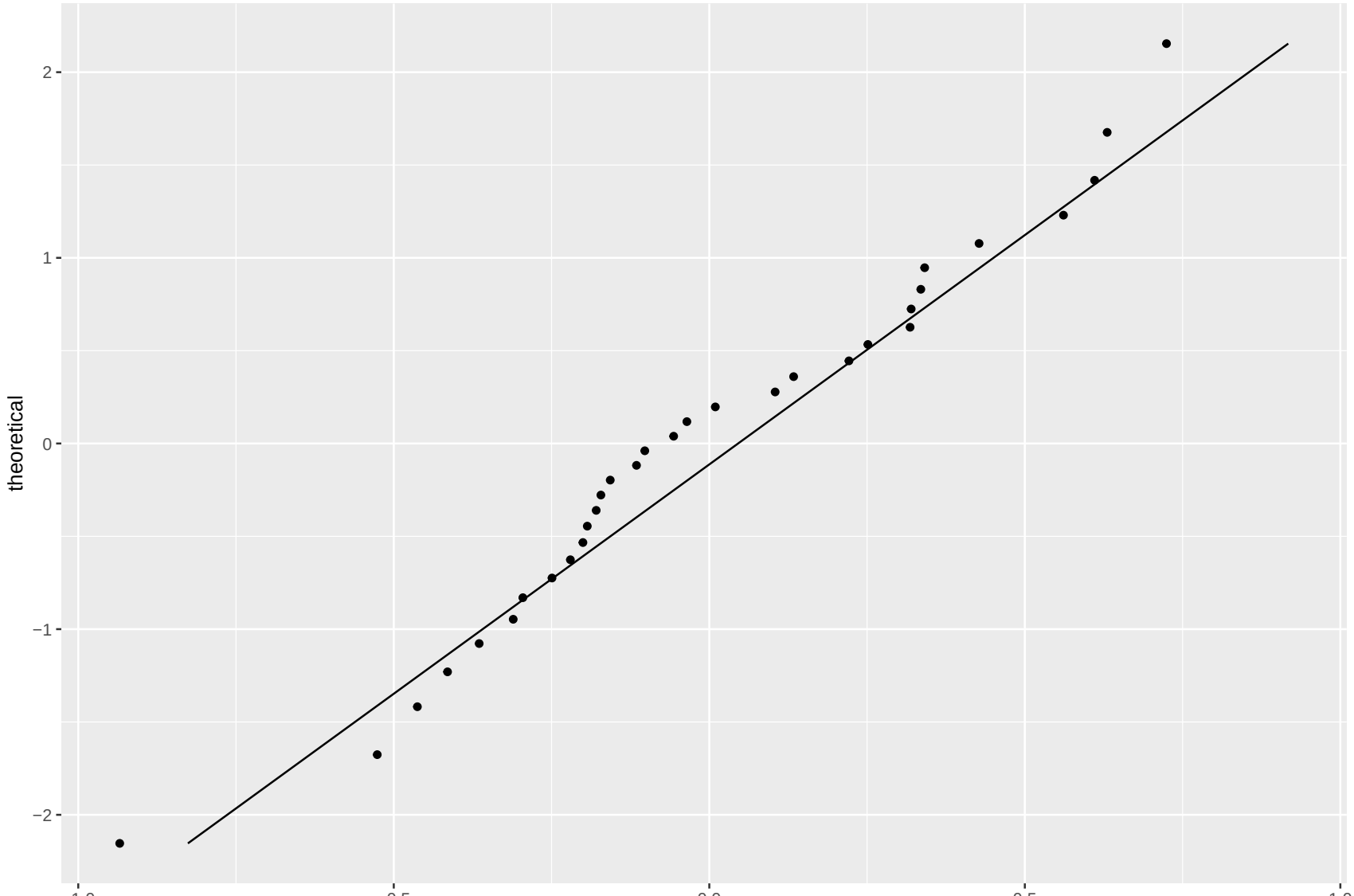
# Re-checking your regression assumptions

```
ggplot(fortify(lion.model), aes(sample = .resid)) +
  geom_qq() + geom_qq_line() + coord_flip()
```



# Re-checking your regression assumptions

```
ggplot(fortify(lion.king.2), aes(sample = .resid)) +
  geom_qq() + geom_qq_line() + coord_flip()
```



# Doing a regression test with summary()

```
summary(lion.model)
```

```
##
## Call:
## lm(formula = age ~ proportion.black, data = lion)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2.5449 -1.1117 -0.5285  0.9635  4.3421
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)        0.8790     0.5688   1.545    0.133
## proportion.black  10.6471     1.5095   7.053 7.68e-08 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1
##
## Residual standard error: 1.669 on 30 degrees of freedom
```

# Doing a regression test with summary()

```
summary(lion.king.2)
```

```
##
## Call:
## lm(formula = age.2 ~ proportion.black, data = lion.2)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.93427 -0.22742 -0.07936  0.31862  0.72453
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)        0.5450     0.1323   4.118 0.000276 **
## proportion.black   2.3078     0.3512   6.572 2.86e-07 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1
##
## Residual standard error: 0.3882 on 30 degrees of freedom
```

FYI. . .

# FYI...

You can also perform a regression test on a linear model with `anova()`

```
anova(lion.king.2)
```

```
## Analysis of Variance Table
##
## Response: age.2
##                   Df Sum Sq Mean Sq F value    Pr(>F)
## proportion.black   1 6.5089  6.5089  43.186 2.856e-07 ***
## Residuals         30 4.5216  0.1507
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1
```

# FYI...

You can calculate 95% confidence intervals for the slope using `confint()`

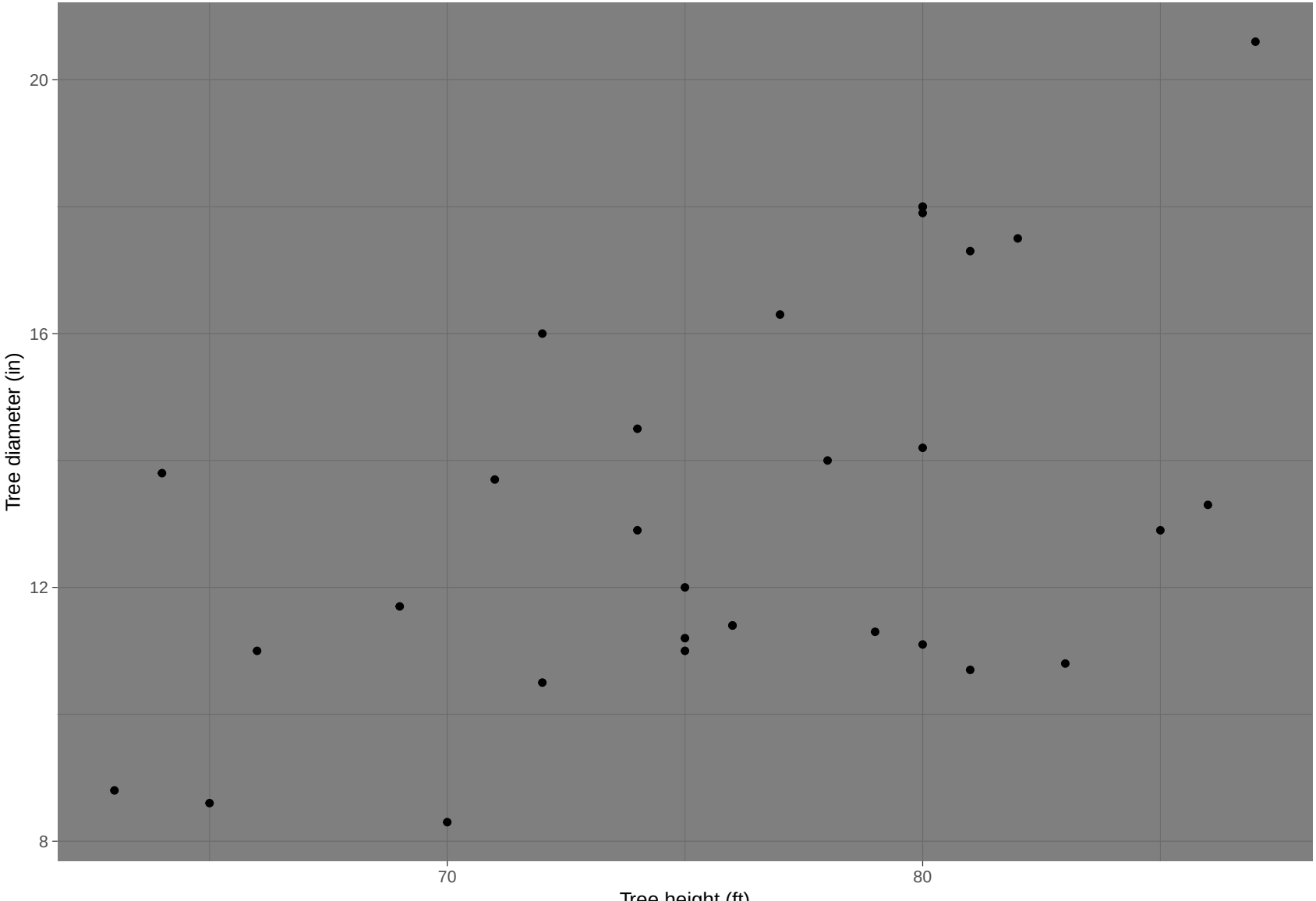
```
confint(lion.king.2)
```

```
##                      2.5 %    97.5 %
## (Intercept)       0.274696 0.8152074
## proportion.black 1.590581 3.0249654
```

# Doing a regression test with `summary()`

Do a regression test of your own on your regression model!

Use the coefficients from the model to write the linear equation



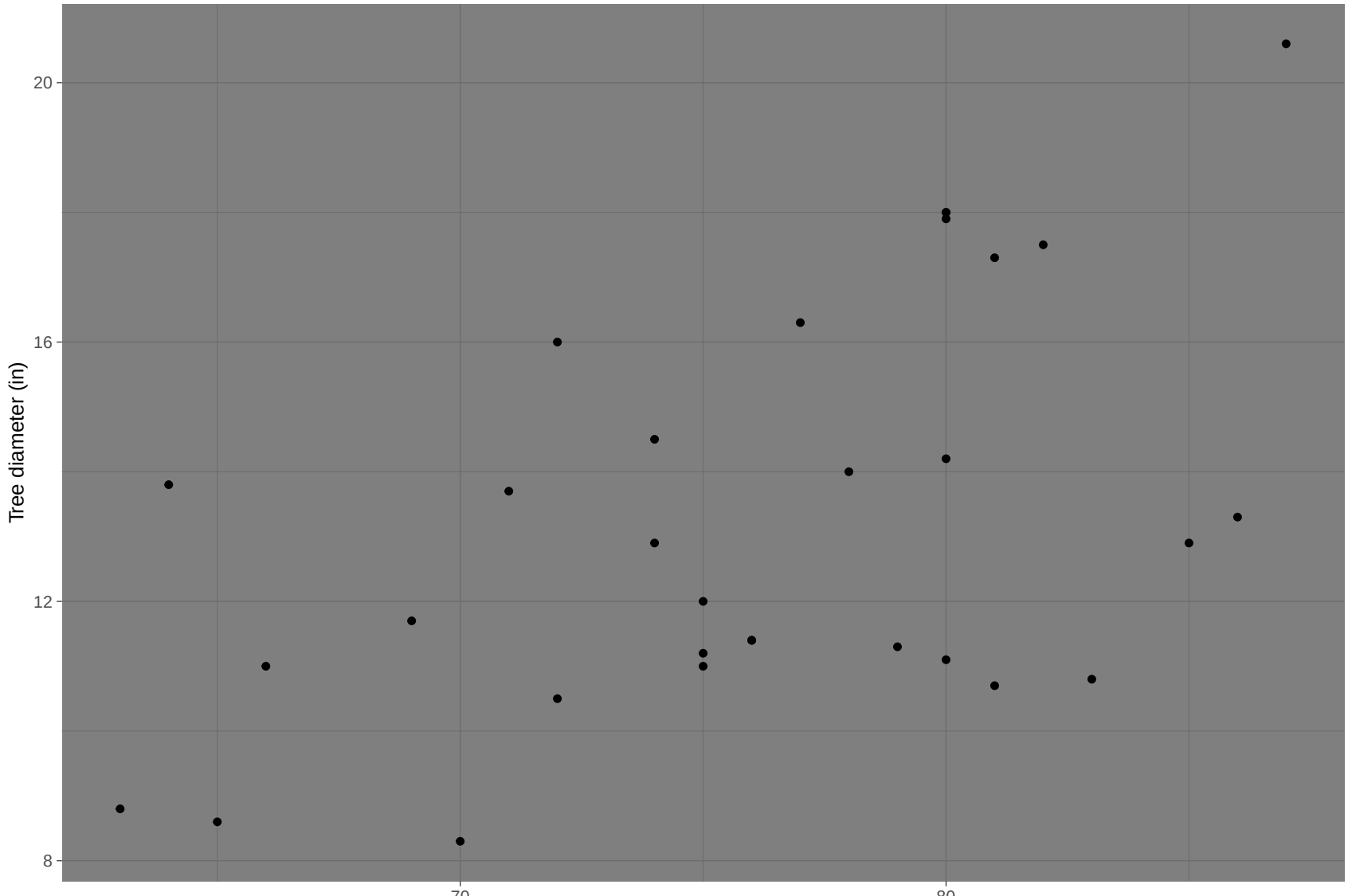
# Doing a regression test with summary()

```
summary(tree.model)
```

```
##
## Call:
## lm(formula = Girth ~ Height, data = trees)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -4.2386 -1.9205 -0.0714  2.7450  4.5384
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -6.18839    5.96020  -1.038  0.30772
## Height       0.25575    0.07816   3.272  0.00276 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1
##
## Residual standard error: 2.728 on 29 degrees of freedom
```

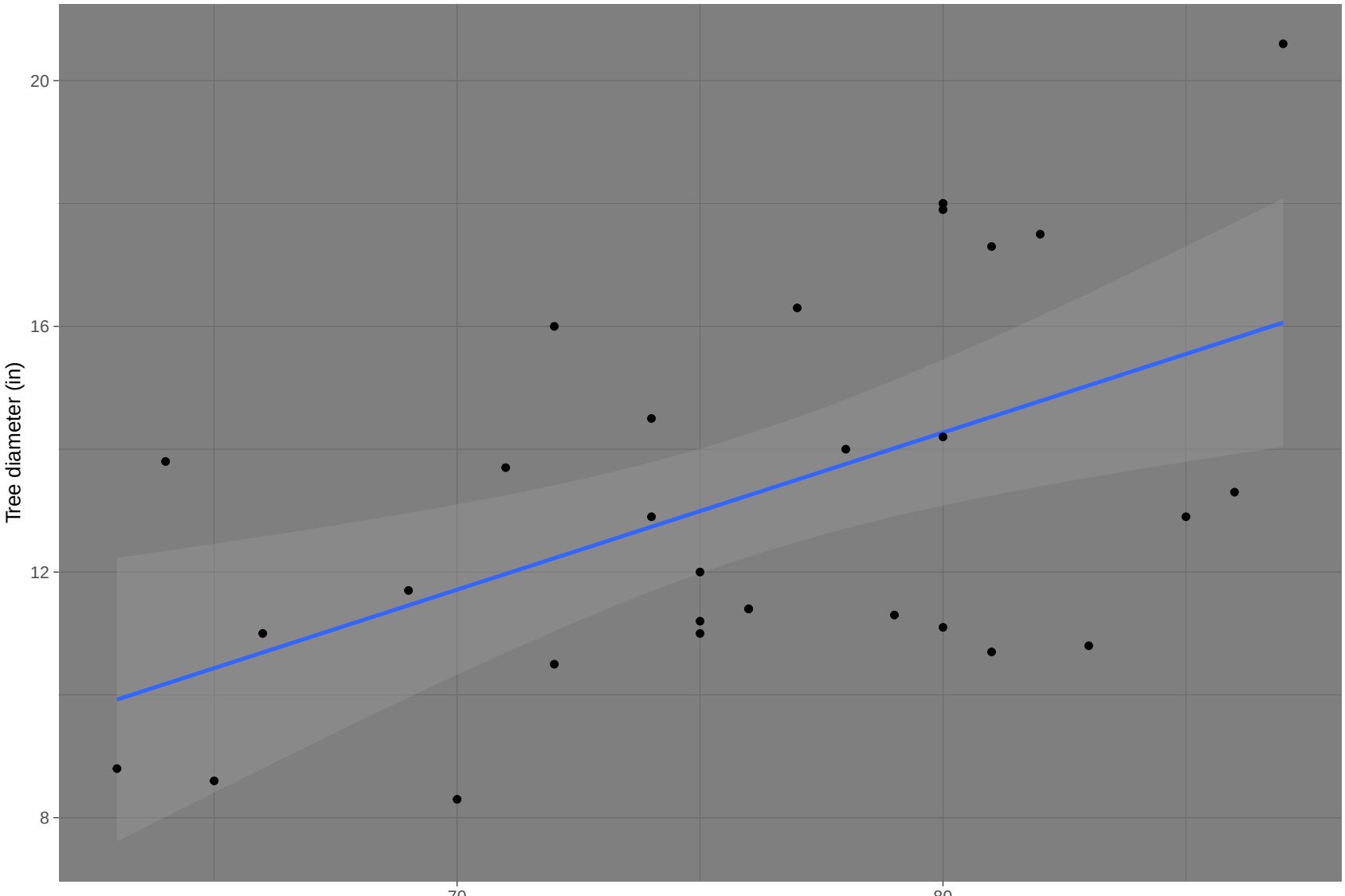
# Plotting best-fit lines with geom_smooth()

```
ggplot(trees, aes(x = Height, y = Girth)) +
  geom_point() + theme_dark() + labs(x = "Tree height (ft)"
```



# Plotting best-fit lines with geom_smooth()

```
ggplot(trees, aes(x = Height, y = Girth)) + geom_smooth(met
  geom_point() + theme_dark() + labs(x = "Tree height (ft)"
```

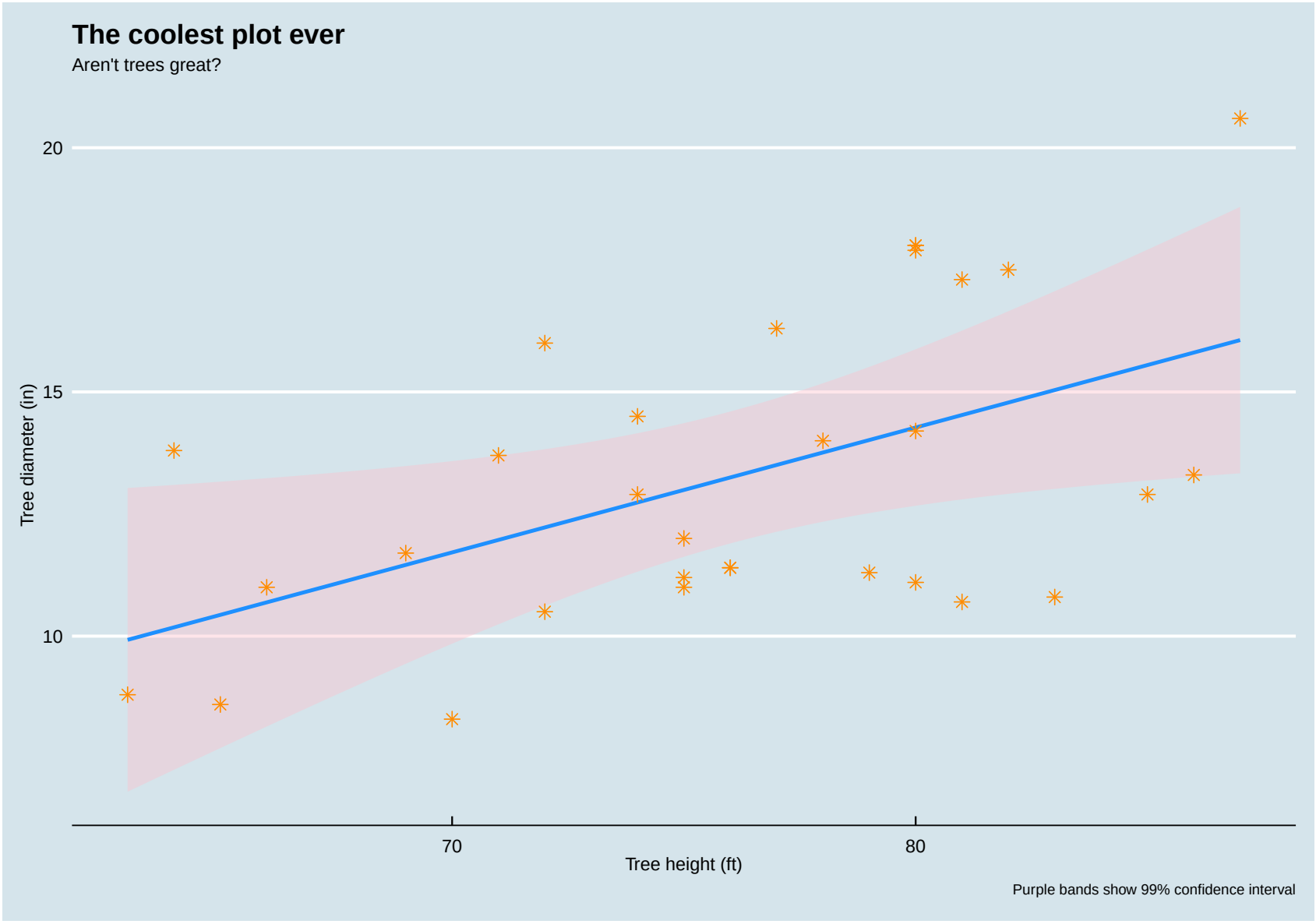


# Plotting best-fit lines with geom_smooth()

```
cool <- ggplot(trees, aes(x = Height, y = Girth)) +
  geom_smooth(method = "lm",
              level = 0.99,
              color = "dodgerblue",
              fill = "pink") +
  geom_point(shape = 8, size = 2.5, color = "darkorange") +
  labs(x = "Tree height (ft)",
       y = "Tree diameter (in)",
       title = "The coolest plot ever",
       subtitle = "Aren't trees great?",
       caption = "Purple bands show 99% confidence interval
```

# Plotting best-fit lines with `geom_smooth()`

```
cool
```



# Changing figure sizes in R chunks