Photo credit: Dave DiCello

17-803 Empirical Methods

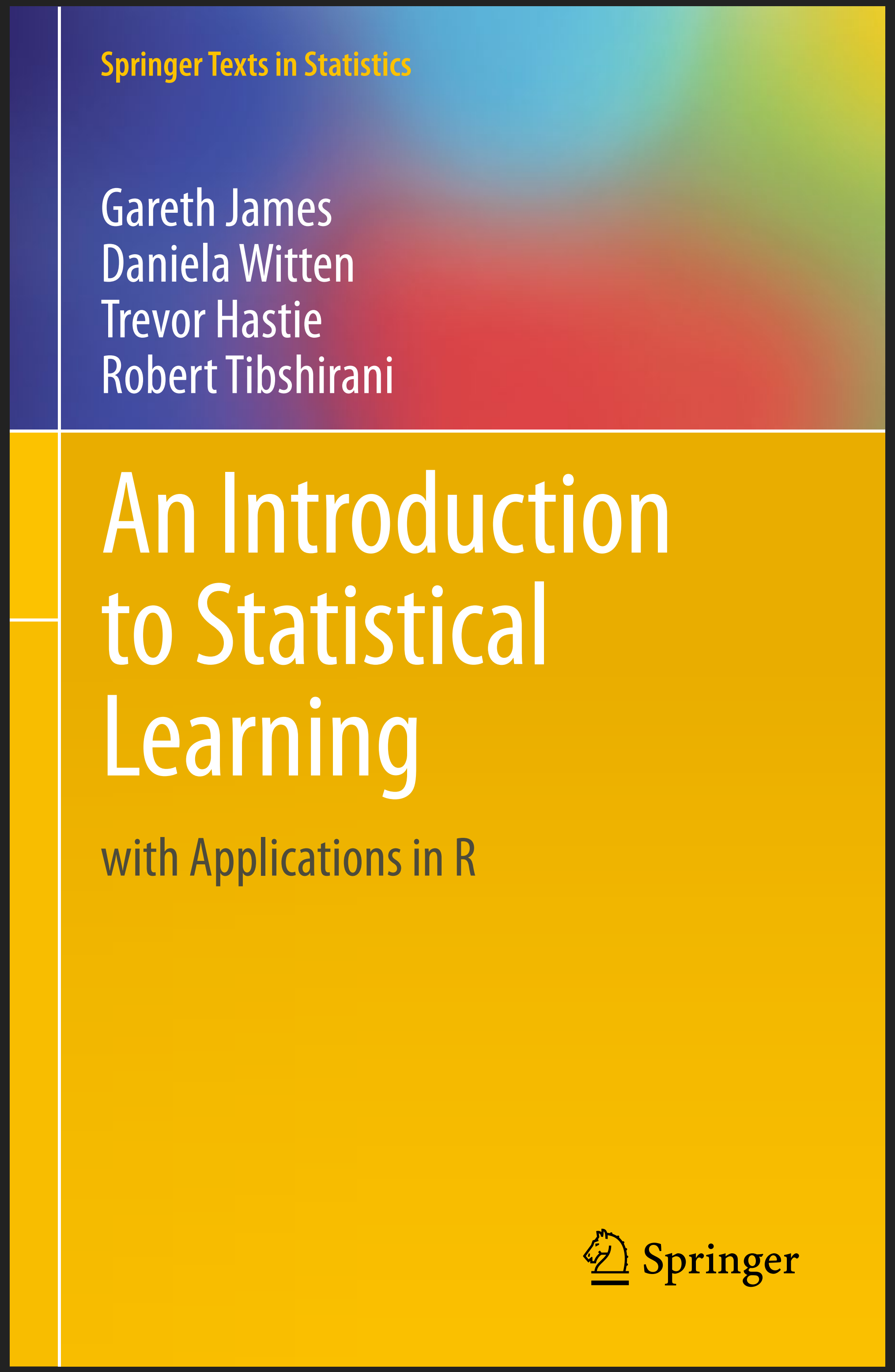
Bogdan Vasilescu, S3D

Regression Modeling (Part 2)

Tuesday, November 1, 2022

Outline for Today

- More linear regression (Rmd + only limited slides)



Springer Texts in Statistics

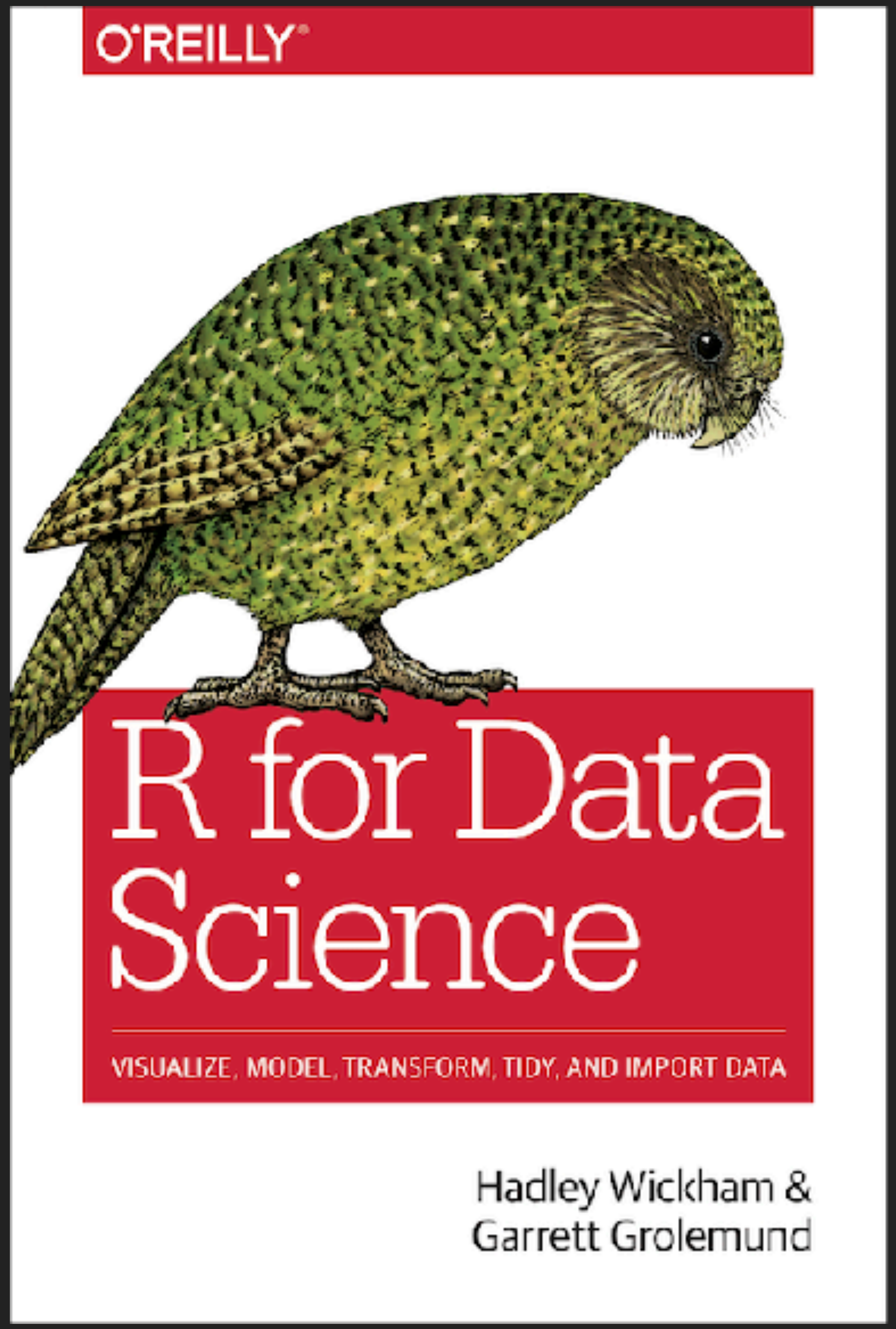
Gareth James
Daniela Witten
Trevor Hastie
Robert Tibshirani

An Introduction to Statistical Learning

with Applications in R

Springer

Ch 3 (Linear regression)



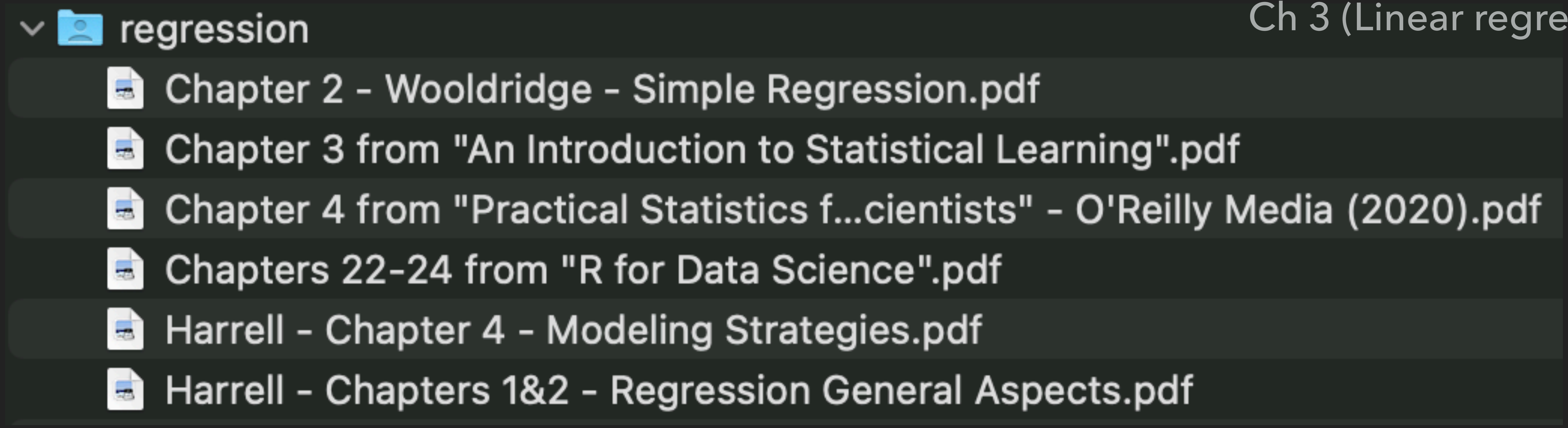
O'REILLY

R for Data Science

VISUALIZE, MODEL, TRANSFORM, TIDY, AND IMPORT DATA

Hadley Wickham &
Garrett Grolemund

Ch 22-24 (Modeling)



regression
- Chapter 2 - Wooldridge - Simple Regression.pdf
- Chapter 3 from "An Introduction to Statistical Learning".pdf
- Chapter 4 from "Practical Statistics f...cientists" - O'Reilly Media (2020).pdf
- Chapters 22-24 from "R for Data Science".pdf
- Harrell - Chapter 4 - Modeling Strategies.pdf
- Harrell - Chapters 1&2 - Regression General Aspects.pdf

More To Read

- Dealing with outliers: https://andrewpbray.github.io/reg/week6B-outliers.html

- Assumptions: https://blog.msbstats.info/posts/2018-08-30-linear-regression-assumptions/

- Diagnostics:
  - Anscombe: https://andrewpbray.github.io/reg/week6A-diagnostics.html
  - https://www.andrew.cmu.edu/user/achoulde/94842/homework/regression_diagnostics.html
  - https://data.library.virginia.edu/diagnostic-plots/
- Q-Q plots: http://seankross.com/2016/02/29/A-Q-Q-Plot-Dissection-Kit.html
- Interactive visualization: https://gallery.shinyapps.io/slr_diag/

- How to code categorical variables in a regression: https://stats.idre.ucla.edu/r/library/r-library-contrast-coding-systems-for-categorical-variables/

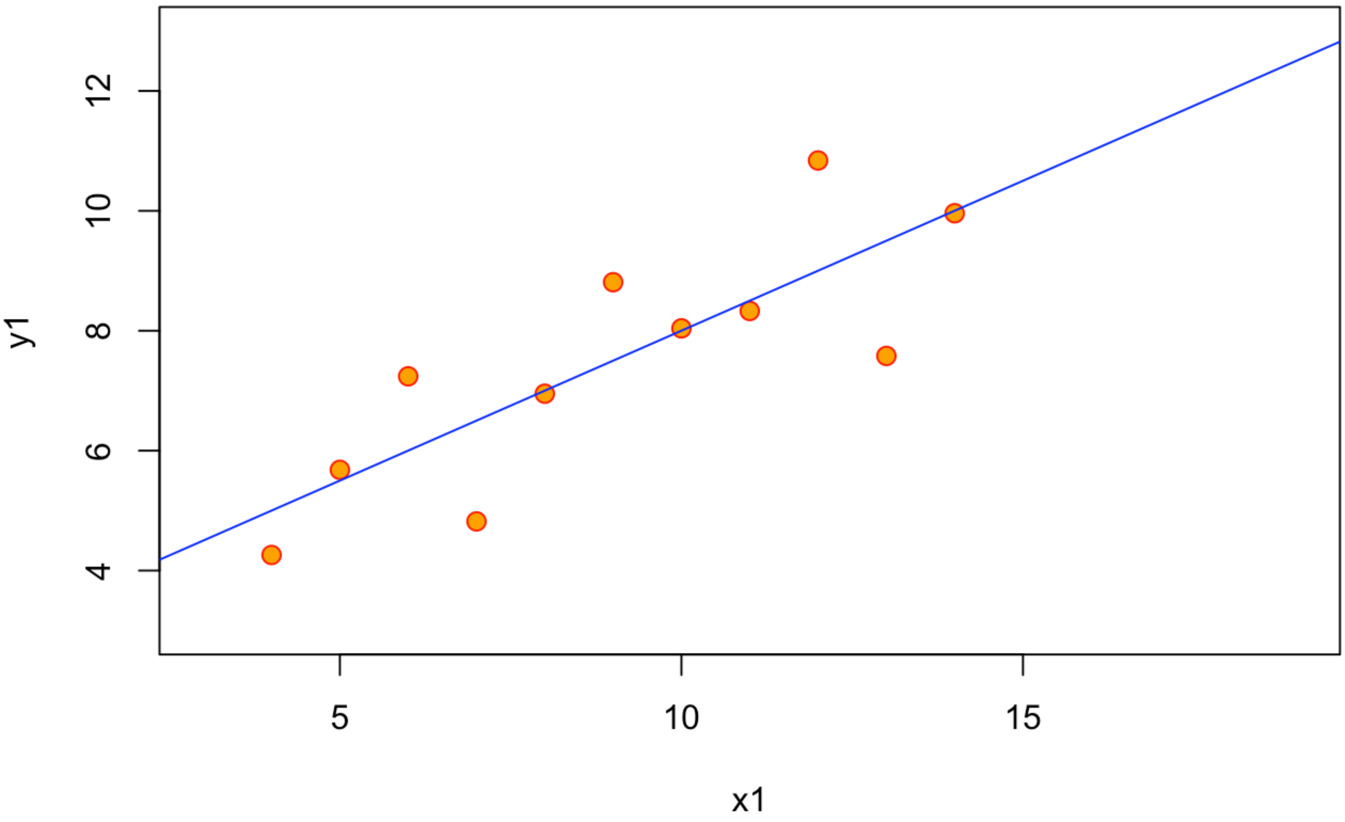
- Understanding model outputs: https://www.andrew.cmu.edu/user/achoulde/94842/
- Alpha vs p-value: https://rationalwiki.org/wiki/Statistical_significance#Alpha_value_versus_p-value

See Also

- CMU 94-842: Programming in R for Analytics:
  - https://www.andrew.cmu.edu/user/achoulde/94842/

Original tutorial by Andrew Bray (Reed College): https://andrewpbray.github.io (https://andrewpbray.github.io)

Simple linear regression



```
##
## Call:
## lm(formula = y1 ~ x1, data = anscombe)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.92127 -0.45577 -0.04136  0.70941  1.83882
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   3.0001     1.1247   2.667  0.02573 *
## x1            0.5001     0.1179   4.241  0.00217 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.237 on 9 degrees of freedom
## Multiple R-squared:  0.6665, Adjusted R-squared:  0.6295
## F-statistic: 17.99 on 1 and 9 DF,  p-value: 0.00217
```

Consider fitted models to three additional data sets

```
m2 = lm(y2 ~ x2, data = anscombe)
summary(m2)
```

```
##
## Call:
## lm(formula = y2 ~ x2, data = anscombe)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1.9009 -0.7609  0.1291  0.9491  1.2691
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)    3.001      1.125   2.667  0.02576 *
## x2             0.500      0.118   4.239  0.00218 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.237 on 9 degrees of freedom
## Multiple R-squared:  0.6662, Adjusted R-squared:  0.6292
## F-statistic: 17.97 on 1 and 9 DF,  p-value: 0.002179
```

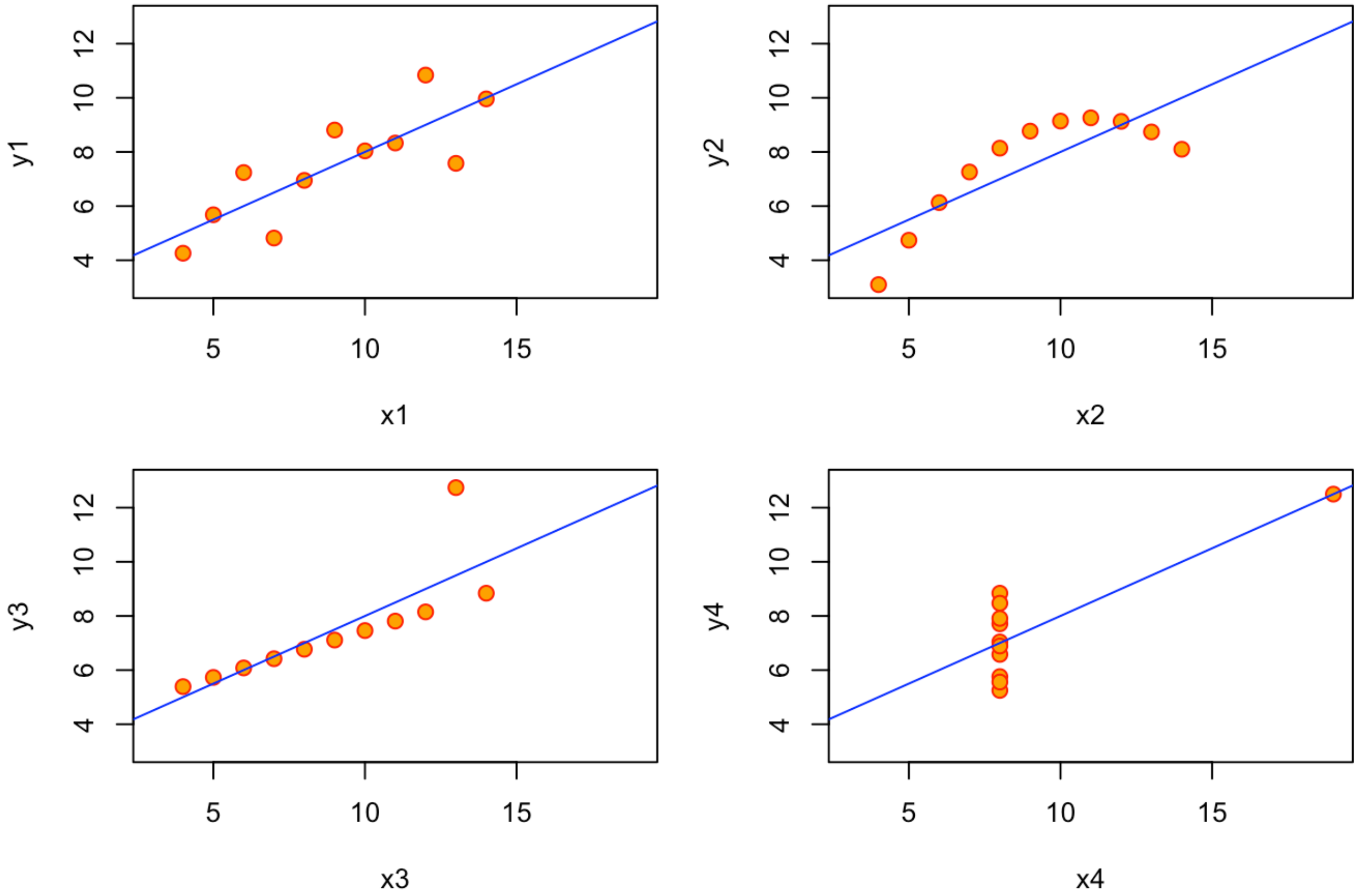
```
m3 = lm(y3 ~ x3, data = anscombe)
summary(m3)
```

```
##
## Call:
## lm(formula = y3 ~ x3, data = anscombe)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1.1586 -0.6146 -0.2303  0.1540  3.2411
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   3.0025     1.1245   2.670  0.02562 *
## x3            0.4997     0.1179   4.239  0.00218 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.236 on 9 degrees of freedom
## Multiple R-squared:  0.6663, Adjusted R-squared:  0.6292
## F-statistic: 17.97 on 1 and 9 DF,  p-value: 0.002176
```

```
m4 = lm(y4 ~ x4, data = anscombe)
summary(m4)
```

```
##
## Call:
## lm(formula = y4 ~ x4, data = anscombe)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -1.751 -0.831  0.000  0.809  1.839
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   3.0017     1.1239   2.671  0.02559 *
## x4            0.4999     0.1178   4.243  0.00216 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.236 on 9 degrees of freedom
## Multiple R-squared:  0.6667, Adjusted R-squared:  0.6297
## F-statistic:    18 on 1 and 9 DF,  p-value: 0.002165
```

Anscombe’s Quartet



For a valid model we need

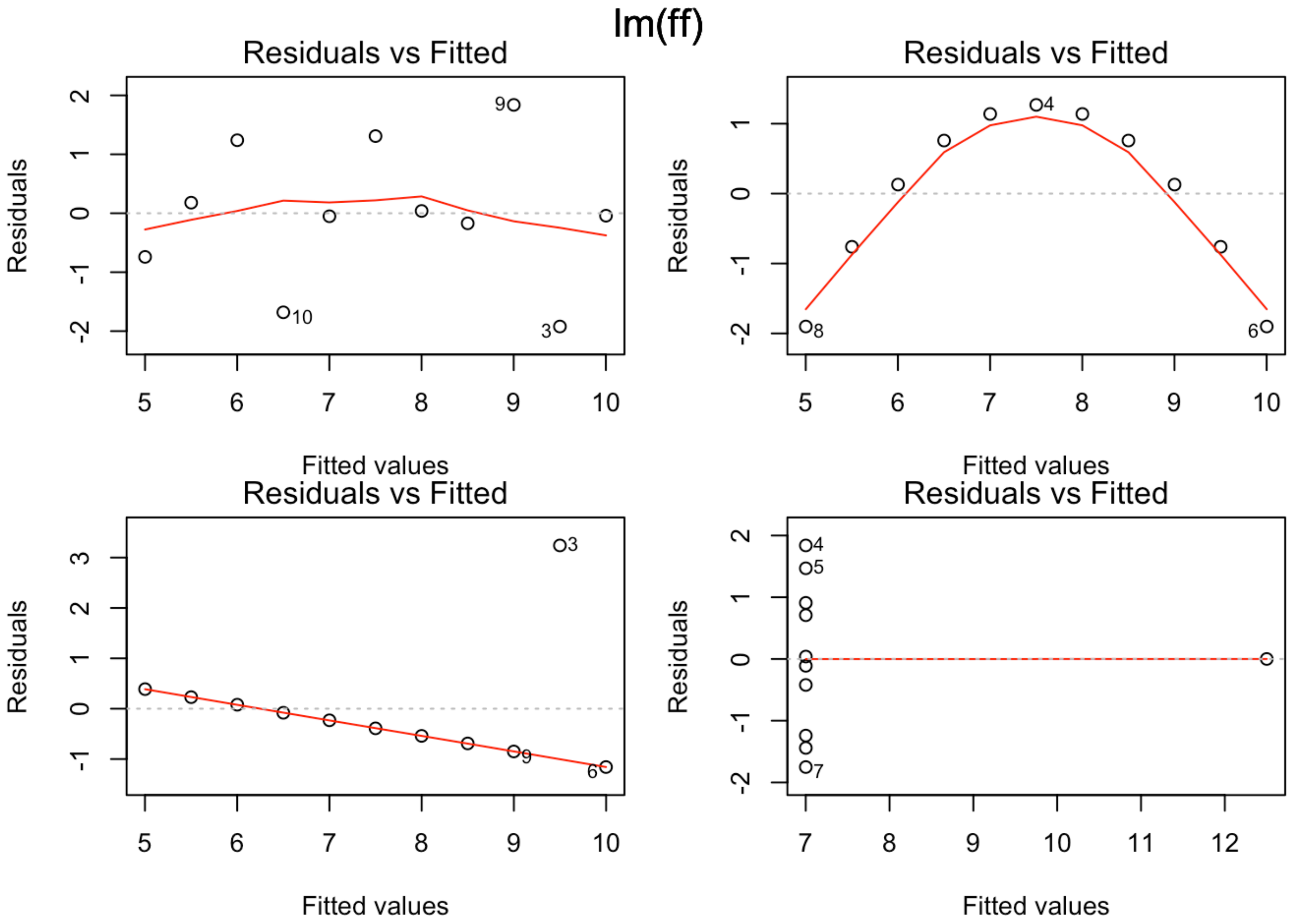
1. The conditional mean of Y|X is a linear function of X.
2. The variance of Y|X is the same for any X.
3. The errors (and thus the Y|X are independent of one another).
4. The errors are normally distributed with mean zero.

5*. No “outliers”.

These can all be assessed with residual plots.

```
plot(anscombe$x1, m1$res) # x versus residuals
plot(m1$fit, m1$res) # x versus fitted
plot(m1, 1) # built-in function
```

Residuals vs fitted



Assessing Normality

We can check the assumption that the errors are normal by looking at the distribution of the residuals. Difficult to do in a residual plot, so we use a QQ plot (for quantile-quantile), aka normal probability plot.

Quantile: The j^{th} quantile, q_j, is the value of a random variable X that fulfills:

$$P(X \leq q_j) = j$$

For the standard normal distribution, $q_{.5} = 0$, $q_{.025} = -1.96$, $q_{.975} = 1.96$.

Constructing a QQ plot

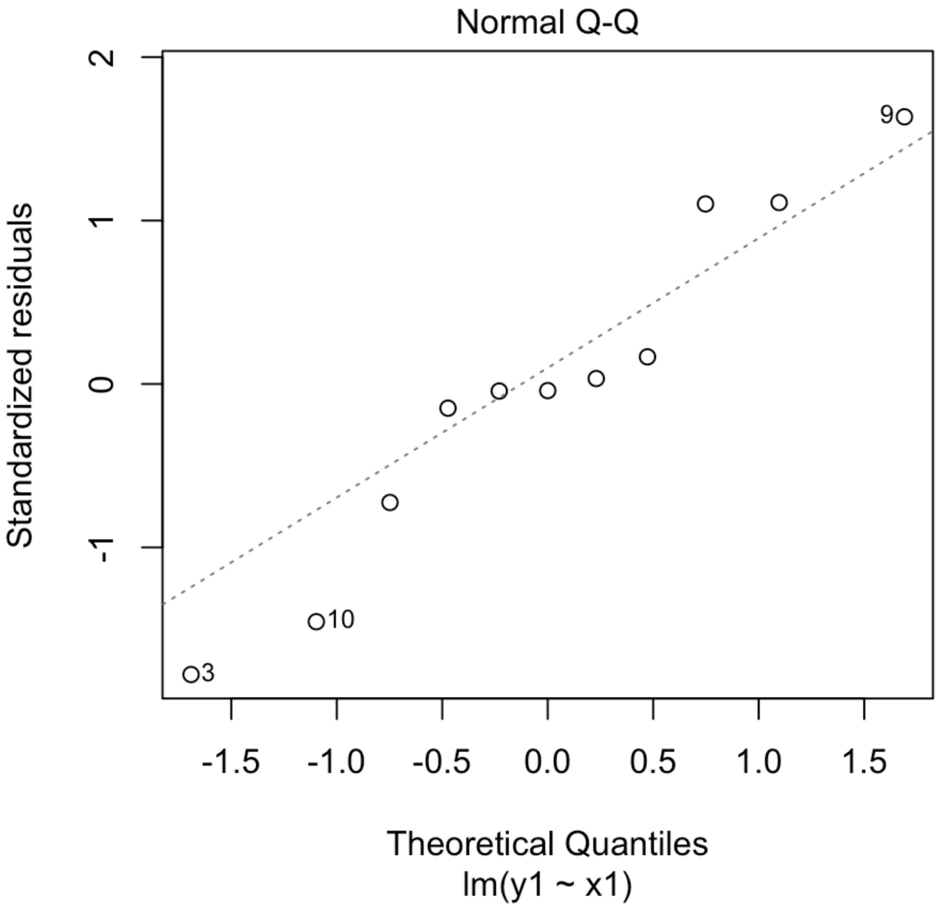
1. Standardize your residuals.

$$\tilde{e}_i = \frac{\hat{e}_i - \bar{\hat{x}}}{s}$$

2. If you have n standardized residuals, you can consider the lowest to be the $1/n$ quantile, the second lowest, the $2/n$ quantile, the median to be the $.5$ quantile, etc.

3. Look up these values for the standard normal distribution and find what the appropriate quantiles would be (this is what `qnorm()` does). These become your theoretical quantiles.
4. Plot the theoretical quantiles against the standardized residuals.

```
plot(m1, 2)
```

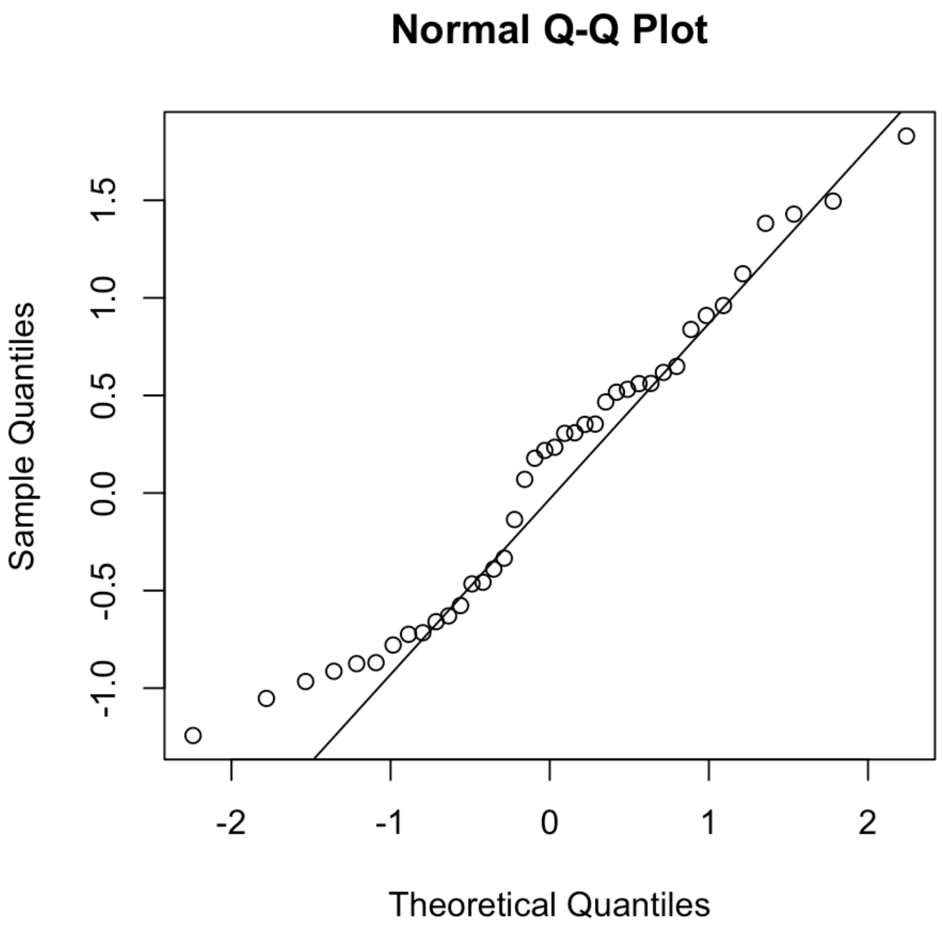


Interpreting a QQ plot

- Perfectly normally distributed residuals would align along the identity line.
- Short tails will veer of the line horizontally.
- Long tails will veer off the line vertically.
- *Expect some variation, even from normal residuals!*

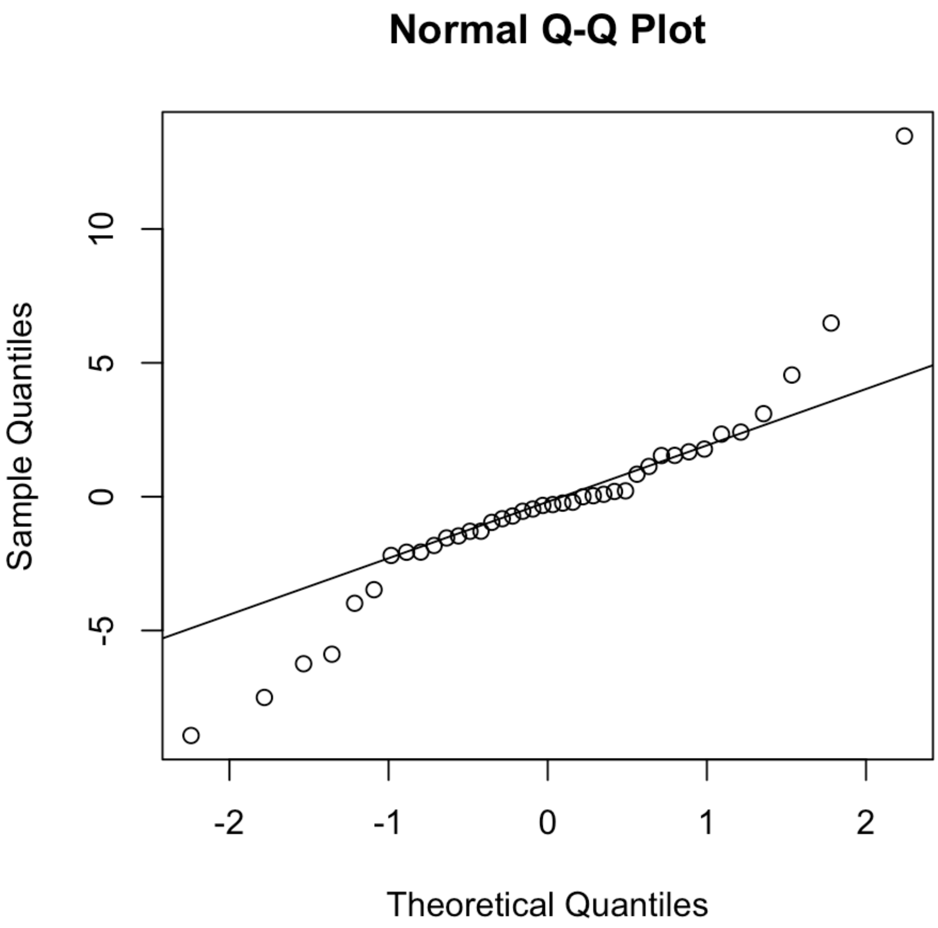
Normal residuals

```
x <- rnorm(40)
qqnorm(x)
qqline(x)
```



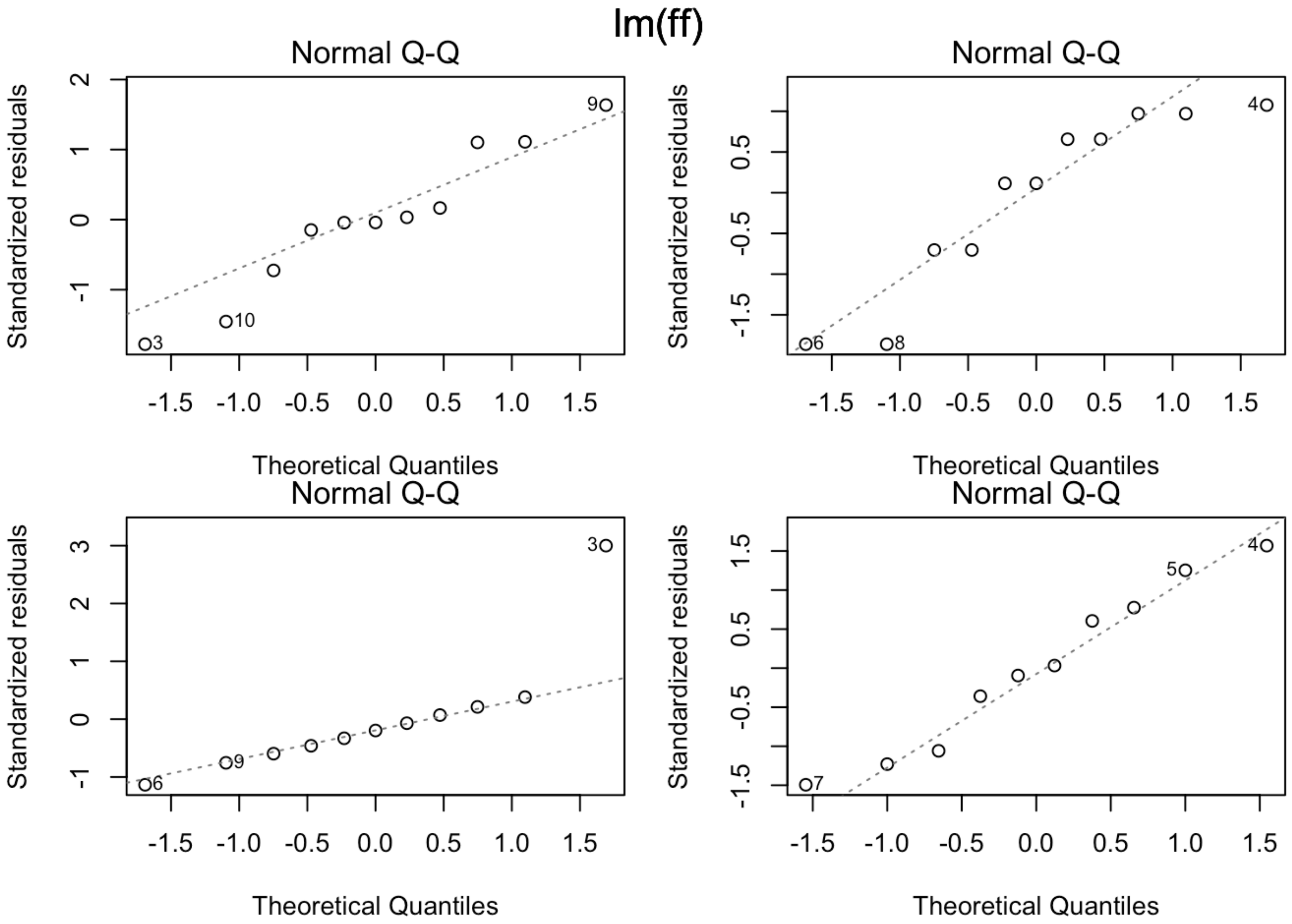
Heavy tailed residuals

```
x <- rt(40, 1)
qqnorm(x)
qqline(x)
```



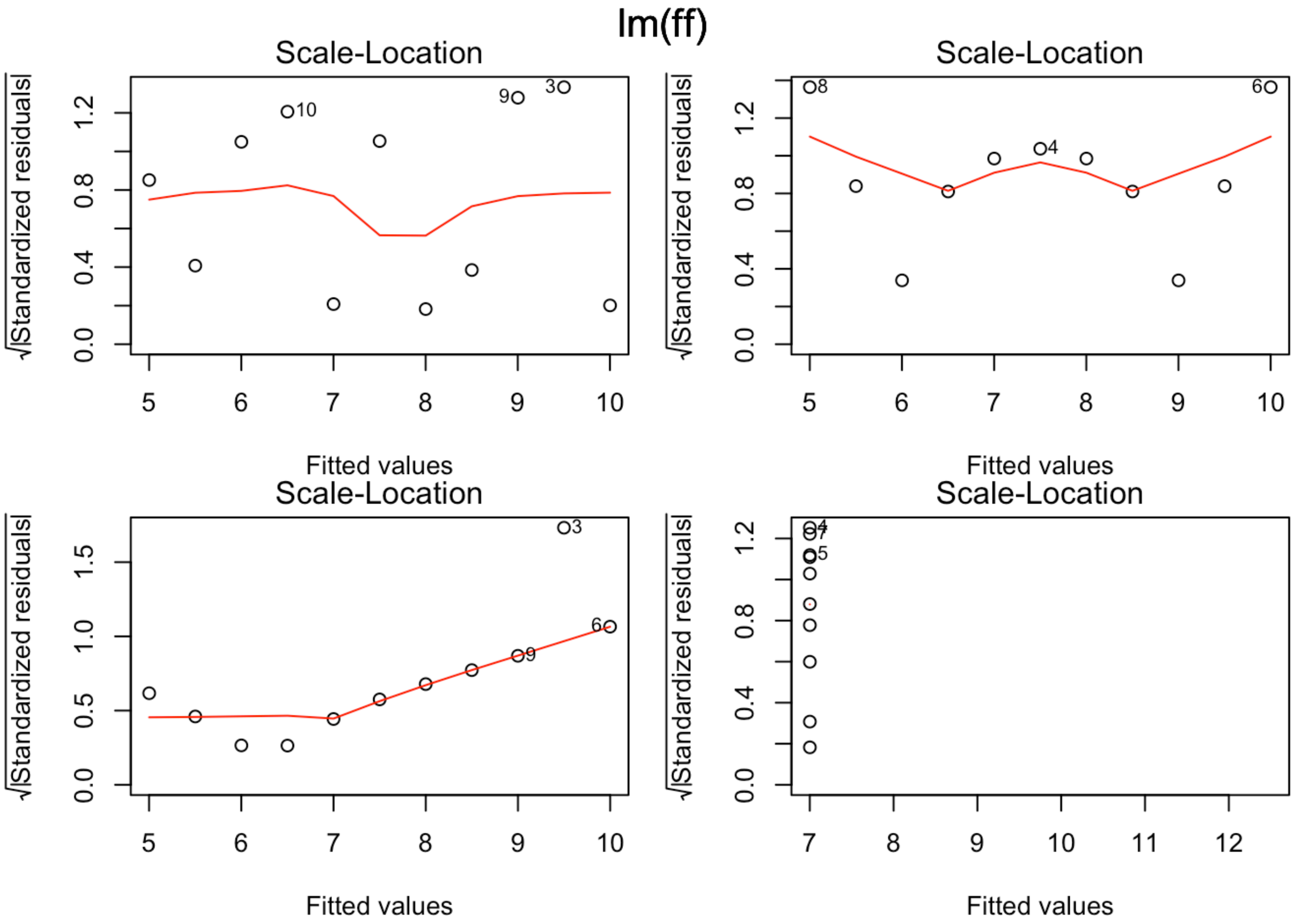
QQ plot

```
## Warning: not plotting observations with leverage one:
##   8
```



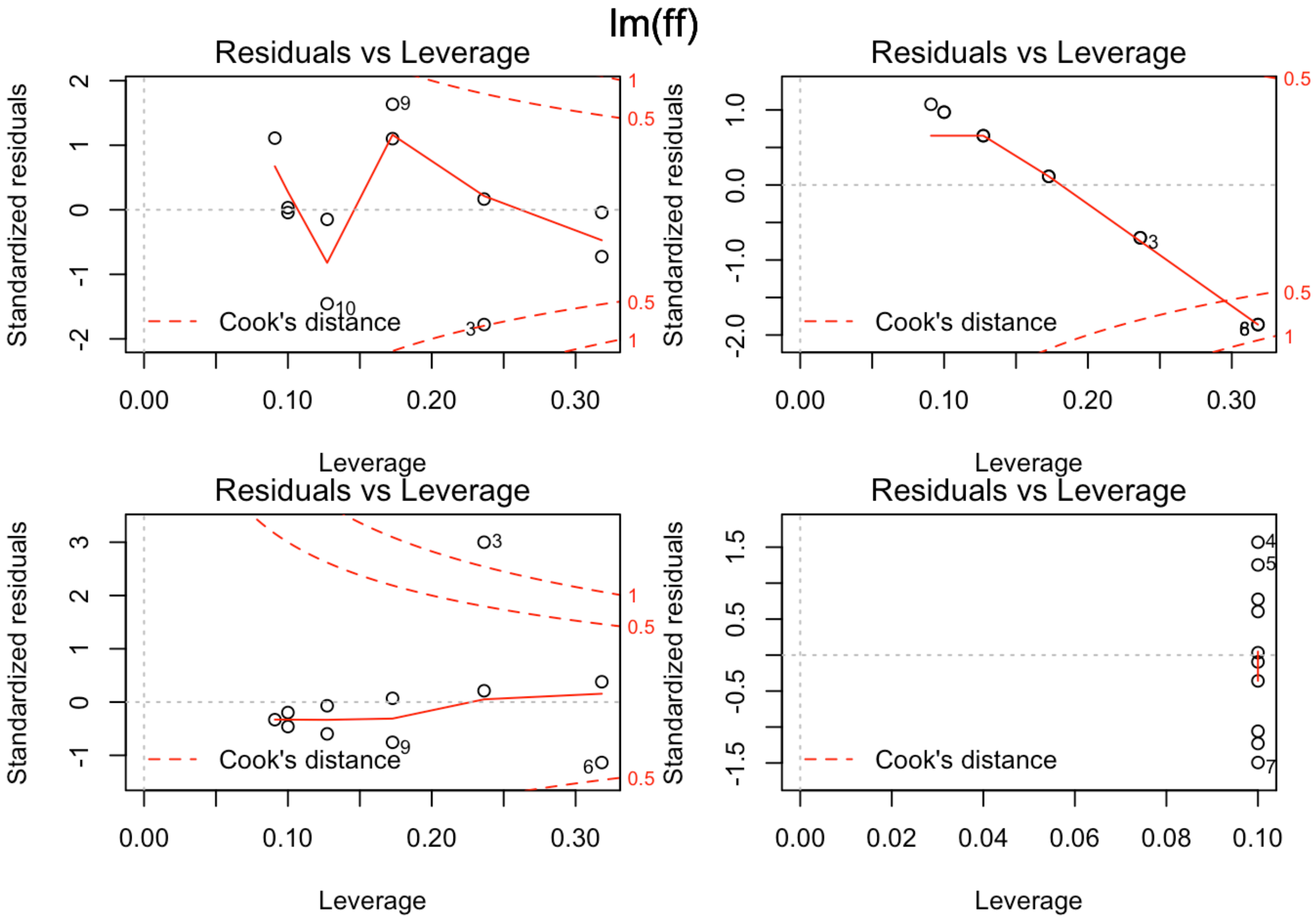
Scale vs location

```
## Warning: not plotting observations with leverage one:
##   8
```



Residuals vs leverage

```
## Warning: not plotting observations with leverage one:
##   8
```


lm(ff)
Residuals vs Leverage
Standardized residuals
Leverage
0.00
0.10
0.20
0.30
2
1
0
-1
-2
9
10
3
Cook's distance
1
0.5
0.5
1
Residuals vs Leverage
Standardized residuals
Leverage
1.0
0.0
-1.0
-2.0
0.00
0.10
0.20
0.30
3
6
Cook's distance
0.5
0.5
1
Residuals vs Leverage
Standardized residuals
Leverage
3
2
1
0
-1
0.00
0.10
0.20
0.30
3
9
6
Cook's distance
1
0.5
0.5
Residuals vs Leverage
Standardized residuals
Leverage
1.5
0.5
-0.5
-1.5
0.00
0.02
0.04
0.06
0.08
0.10
4
5
7
Cook's distance

Multicollinearity

Here’s a closer look at how to diagnose multicollinearity. Remember the example dataset from lecture?

```
y = c(12, 13, 10, 5, 7, 12, 15)
x1 = c(6, 6.5, 5, 2.5, 3.5, 6, 7.5)
x2 = c(6, 6.5, 5, 2.5, 3.5, 6, 7.5)

m = lm(y ~ x1 + x2)
summary(m)
```

```
## Warning in summary.lm(m): essentially perfect fit: summary may be unreliable
```

```
## 
## Call:
## lm(formula = y ~ x1 + x2)
## 
## Residuals:
##          1          2          3          4          5          6          7 
## -1.033e-15  1.830e-15 -2.529e-16  3.068e-16 -1.835e-16 -2.991e-16 -3.684e-16 
## 
## Coefficients: (1 not defined because of singularities)
##               Estimate Std. Error    t value Pr(>|t|)    
## (Intercept) -4.028e-15  1.266e-15 -3.182e+00   0.0245 *  
## x1           2.000e+00  2.290e-16  8.734e+15   <2e-16 ***
## x2                  NA         NA         NA       NA    
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## 
## Residual standard error: 9.83e-16 on 5 degrees of freedom
## Multiple R-squared:      1,  Adjusted R-squared:      1 
## F-statistic: 7.628e+31 on 1 and 5 DF,  p-value: < 2.2e-16
```

```
vif(m)
```

```
## Error in vif.default(m): there are aliased coefficients in the model
```

We can’t even compute the VIF because the two variables are copies of each other. Let’s add some noise to x2.

```
y = c(12, 13, 10, 5, 7, 12, 15)
x1 = c(6, 6.5, 5, 2.5, 3.5, 6, 7.5)
x2 = c(6.1, 6.15, 5.1, 2.51, 3.52, 6.1, 7.6)

m = lm(y ~ x1 + x2)
summary(m)
```

```
## Warning in summary.lm(m): essentially perfect fit: summary may be unreliable
```

```
##
## Call:
## lm(formula = y ~ x1 + x2)
##
## Residuals:
##          1          2          3          4          5          6          7
## -5.413e-16  2.548e-18  1.954e-16  1.792e-16 -2.160e-16  1.924e-16  1.877e-16
##
## Coefficients:
##               Estimate Std. Error    t value Pr(>|t|)
## (Intercept) -2.686e-15  4.498e-16 -5.971e+00  0.00395 **
## x1           2.000e+00  8.624e-16  2.319e+15  < 2e-16 ***
## x2          -5.201e-15  8.658e-16 -6.007e+00  0.00387 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.472e-16 on 4 degrees of freedom
## Multiple R-squared:      1,  Adjusted R-squared:      1
## F-statistic: 3.057e+32 on 2 and 4 DF,  p-value: < 2.2e-16
```

```
vif(m)
```

```
## Warning in summary.lm(object, ...): essentially perfect fit: summary may be
## unreliable
```

```
##       x1       x2
## 113.6997 113.6997
```

Now we get lots of indication that multicollinearity is an issue. Look at the VIF values! Note also how arbitrary the coefficient estimates for x1 and x2 are, given what you know about the data (y = x1 + x2).

Factors in linear regression

Interpreting coefficients of factor variables

In the case of quantitative predictors, we're more or less comfortable with the interpretation of the linear model coefficient as a "slope" or a "unit increase in outcome per unit increase in the covariate". This isn't the right interpretation for factor variables. In particular, the notion of a slope or unit change no longer makes sense when talking about a categorical variable. E.g., what does it even mean to say "unit increase in major" when studying the effect of college major on future earnings?

To understand what the coefficients really mean, let's go back to the birthwt data and try regressing birthweight on mother's race and mother's age.

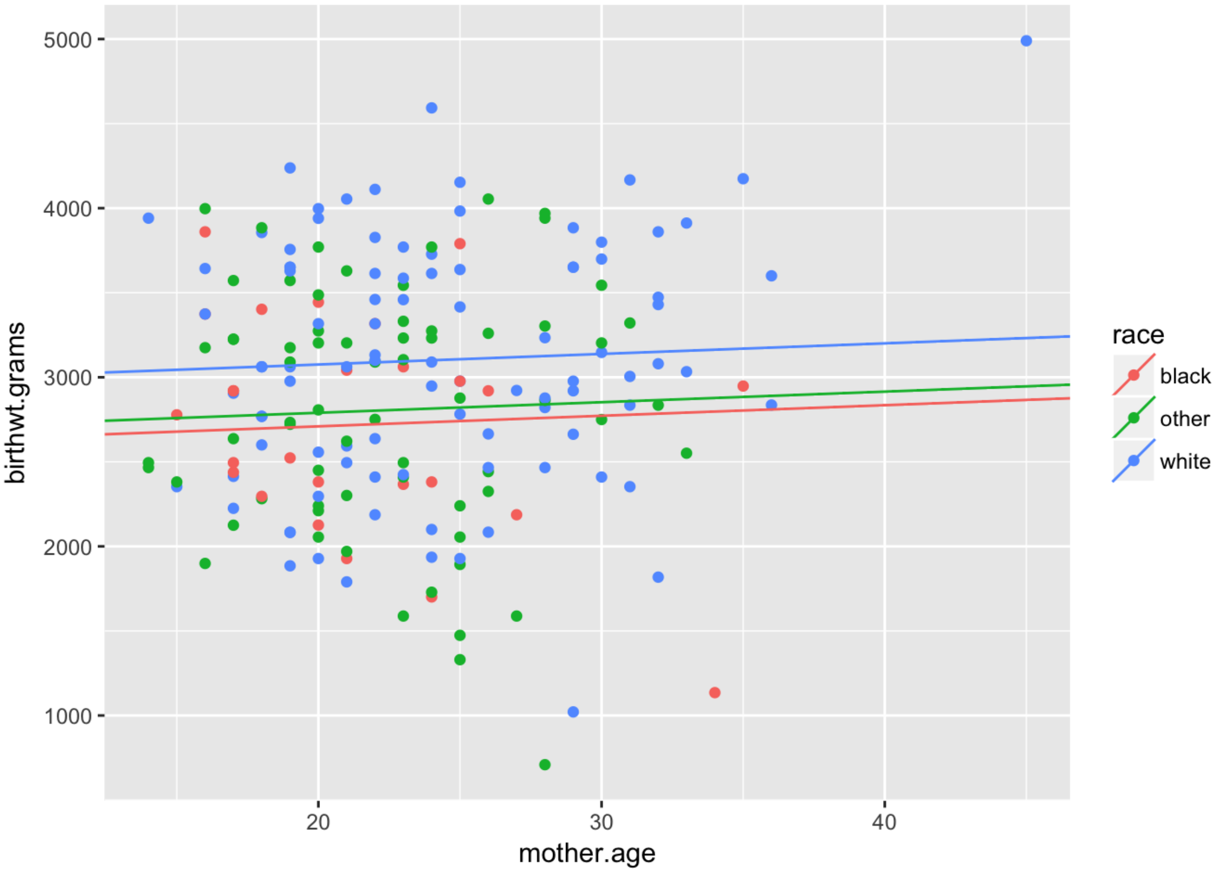
```
# Fit regression model
birthwt.lm <- lm(birthwt.grams ~ race + mother.age, data = birthwt)

# Regression model summary
summary(birthwt.lm)
```

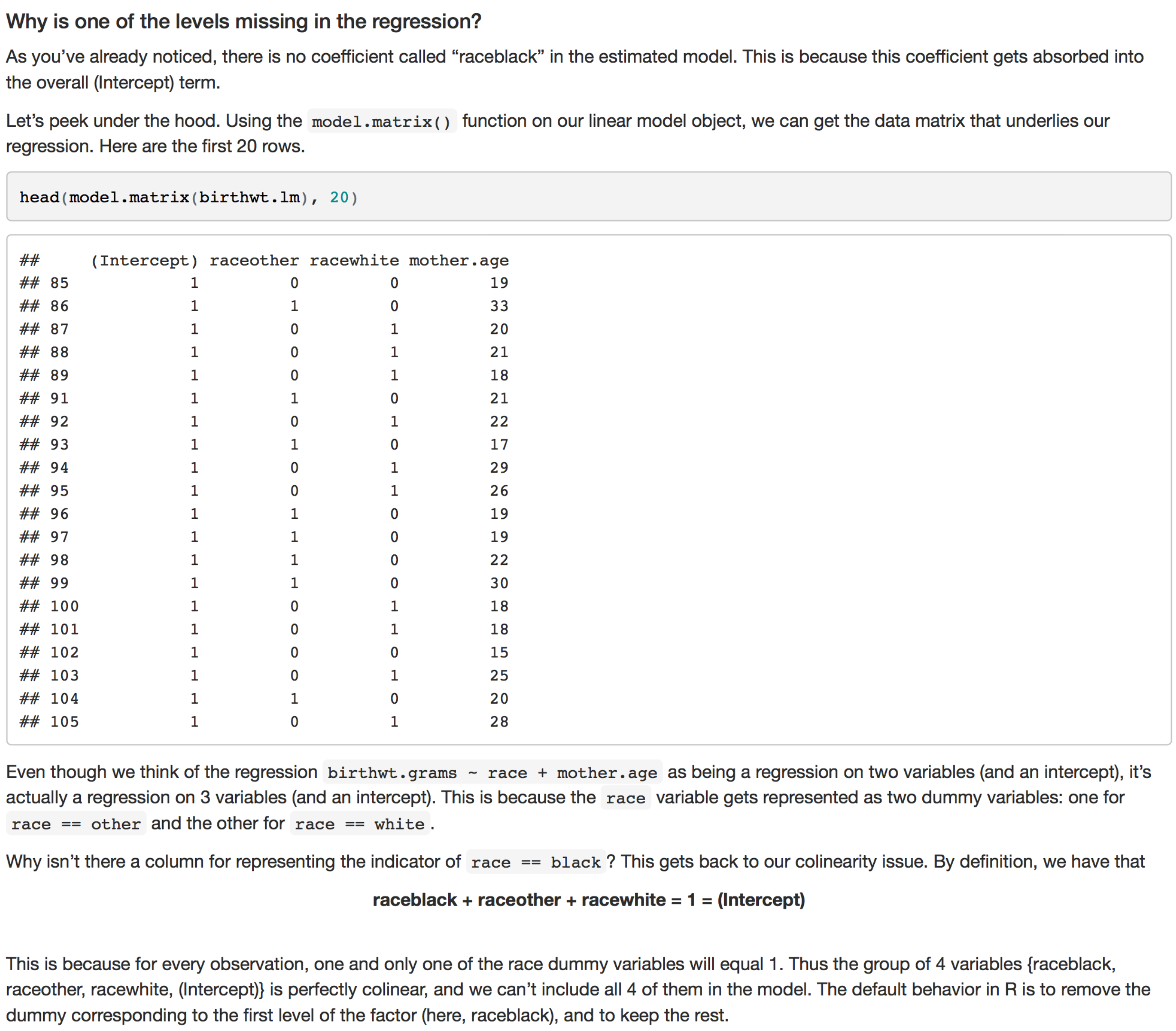
```
## 
## Call:
## lm(formula = birthwt.grams ~ race + mother.age, data = birthwt)
## 
## Residuals:
##      Min       1Q   Median       3Q      Max 
## -2131.57  -488.02    -1.16   521.87  1757.07 
## 
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)    
## (Intercept) 2584.264    258.393  10.001   <2e-16 ***
## raceother     80.249    165.582   0.485    0.628    
## racewhite    365.715    160.636   2.277    0.024 *  
## mother.age     6.288     10.073   0.624    0.533    
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## 
## Residual standard error: 715.7 on 185 degrees of freedom
## Multiple R-squared:  0.05217,    Adjusted R-squared:  0.0368 
## F-statistic: 3.394 on 3 and 185 DF,  p-value: 0.01909
```

Note that there are two coefficients estimated for the race variable (`raceother` and `racewhite`). What's happening here?

When you put a factor variable into a regression, you're allowing a **different intercept at every level of the factor**. In the present example, you're saying that you want to model `birthwt.grams` as

Baby's birthweight = Intercept(based on mother's race) + β * mother's age


5000
4000
3000
2000
1000
birthwt.grams
20
30
40
mother.age
race
black
other
white



Why is one of the levels missing in the regression?

As you've already noticed, there is no coefficient called "raceblack" in the estimated model. This is because this coefficient gets absorbed into the overall (Intercept) term.

Let's peek under the hood. Using the `model.matrix()` function on our linear model object, we can get the data matrix that underlies our regression. Here are the first 20 rows.

```
head(model.matrix(birthwt.lm), 20)
```

```
##     (Intercept) raceother racewhite mother.age
## 85            1         0         0         19
## 86            1         1         0         33
## 87            1         0         1         20
## 88            1         0         1         21
## 89            1         0         1         18
## 91            1         1         0         21
## 92            1         0         1         22
## 93            1         1         0         17
## 94            1         0         1         29
## 95            1         0         1         26
## 96            1         1         0         19
## 97            1         1         0         19
## 98            1         1         0         22
## 99            1         1         0         30
## 100           1         0         1         18
## 101           1         0         1         18
## 102           1         0         0         15
## 103           1         0         1         25
## 104           1         1         0         20
## 105           1         0         1         28
```

Even though we think of the regression `birthwt.grams ~ race + mother.age` as being a regression on two variables (and an intercept), it's actually a regression on 3 variables (and an intercept). This is because the `race` variable gets represented as two dummy variables: one for `race == other` and the other for `race == white`.

Why isn't there a column for representing the indicator of `race == black`? This gets back to our colinearity issue. By definition, we have that

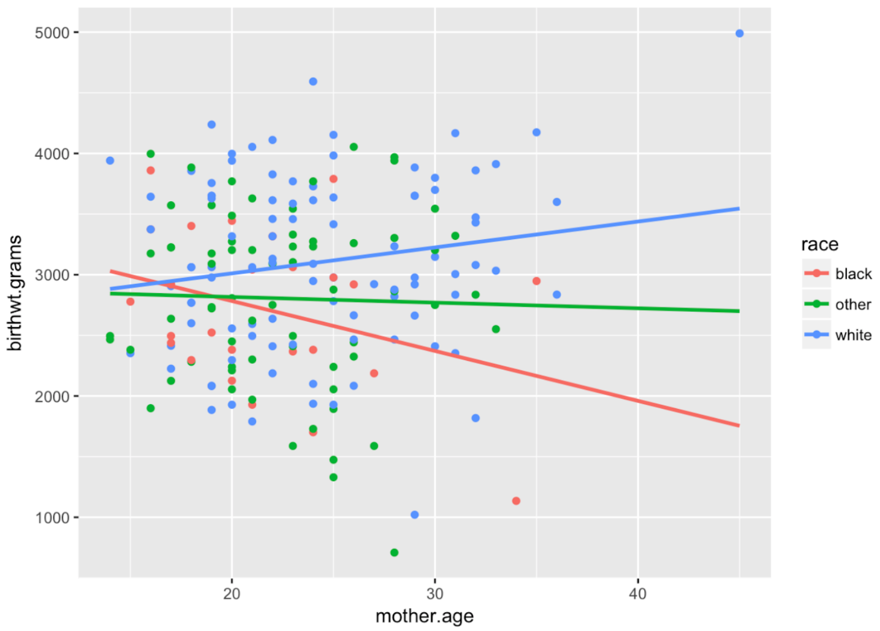
raceblack + raceother + racewhite = 1 = (Intercept)

This is because for every observation, one and only one of the race dummy variables will equal 1. Thus the group of 4 variables {raceblack, raceother, racewhite, (Intercept)} is perfectly colinear, and we can't include all 4 of them in the model. The default behavior in R is to remove the dummy corresponding to the first level of the factor (here, raceblack), and to keep the rest.

How to code categorical variables in a regression:
https://stats.idre.ucla.edu/r/library/r-library-contrast-coding-systems-for-categorical-variables/

Coding schemes covered

Coding Scheme	Comparisons made
Dummy Coding	Compares each level to the reference level, intercept being the cell mean of the reference group
Simple Coding	Compares each level to the reference level, intercept being the grand mean
Deviation Coding	Compares each level to the grand mean
Orthogonal Polynomial Coding	Orthogonal polynomial contrasts
Helmert Coding	Compare levels of a variable with the mean of the subsequent levels of the variable
Reverse Helmert Coding	Compares levels of a variable with the mean of the previous levels of the variable
Forward Difference Coding	Compares adjacent levels of a variable (each level minus the next level)
Backward Difference Coding	Compares adjacent levels of a variable (each level minus the prior level)
User-Defined Coding	User-defined contrast



In this case we have not only race-specific intercepts, but also **race-specific slopes**. The plot above corresponds to the model:

Baby's birthweight = Intercept(based on mother's race) + β(based on mother's race) * mother's age

To specify this interaction model in R, we use the following syntax

```
birthwt.lm.interact <- lm(birthwt.grams ~ race * mother.age, data = birthwt)

summary(birthwt.lm.interact)
```

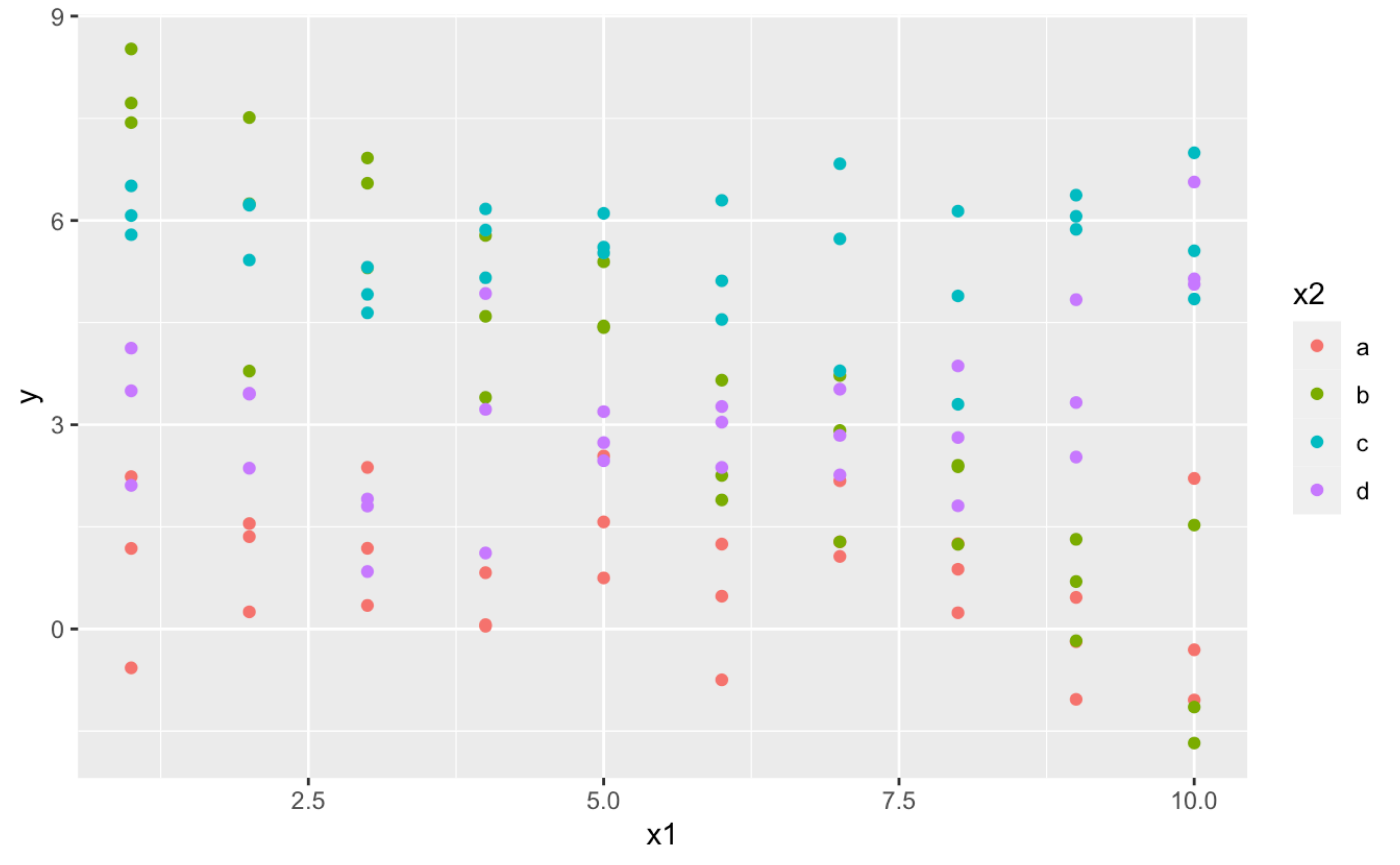
```
##
## Call:
## lm(formula = birthwt.grams ~ race * mother.age, data = birthwt)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2182.35  -474.23    13.48   523.86  1496.51
##
## Coefficients:
##                      Estimate Std. Error t value    Pr(>|t|)
## (Intercept)           3606.33     615.26   5.861 0.000000021 ***
## raceother             -696.74     756.65  -0.921      0.3584
## racewhite            -1022.79     694.21  -1.473      0.1424
## mother.age             -41.17      27.82  -1.480      0.1407
## raceother:mother.age    36.51      33.85   1.078      0.2823
## racewhite:mother.age    62.54      30.67   2.039      0.0429 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 710.7 on 183 degrees of freedom
## Multiple R-squared:  0.07541,    Adjusted R-squared:  0.05015
## F-statistic: 2.985 on 5 and 183 DF,  p-value: 0.01291
```

Plotting interactions among continuous variables in regression models - `jtools::interact_plot`

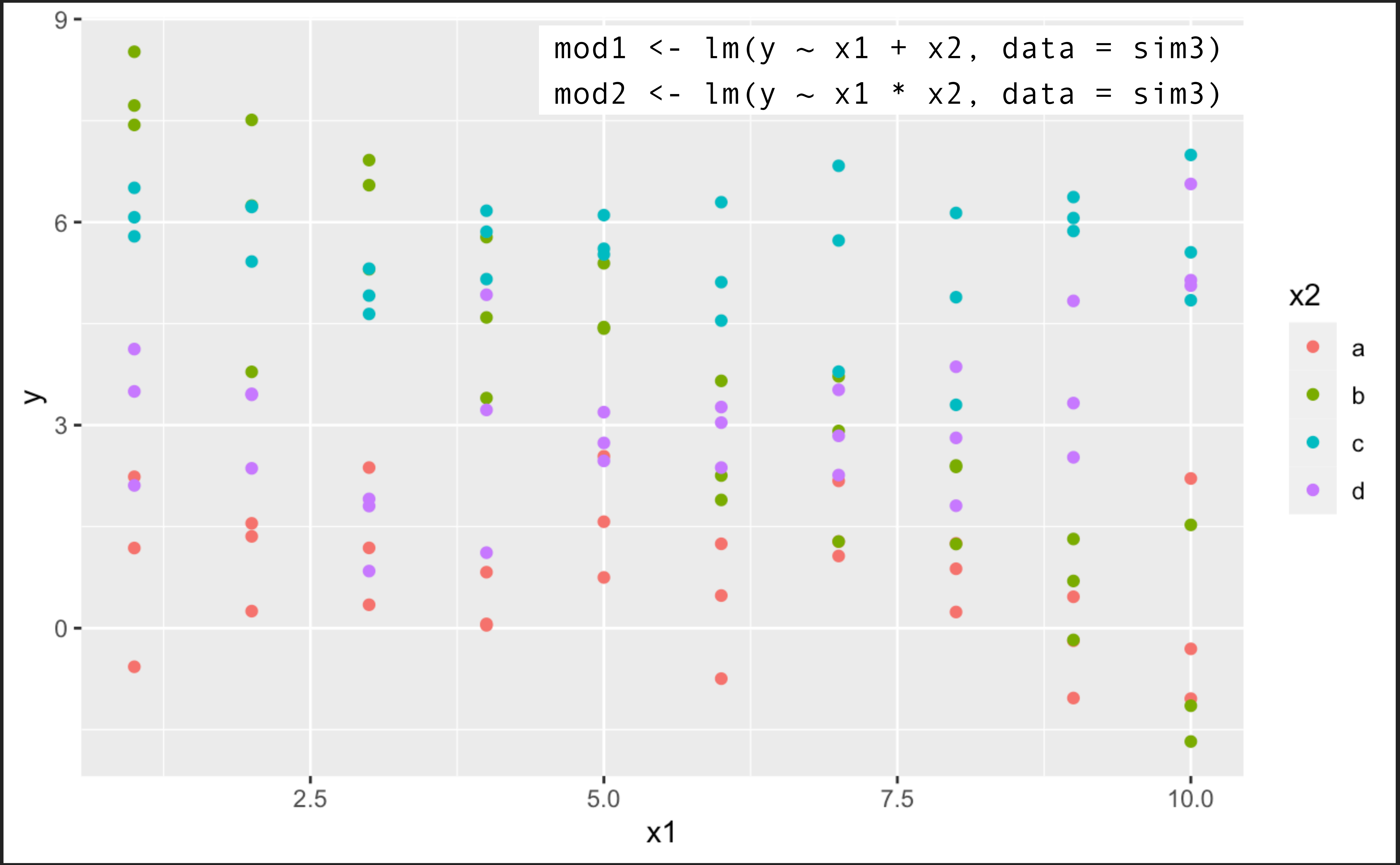
https://cran.r-project.org/web/packages/jtools/vignettes/interactions.html

Another interaction example

What happens when you combine a continuous and a categorical variable?

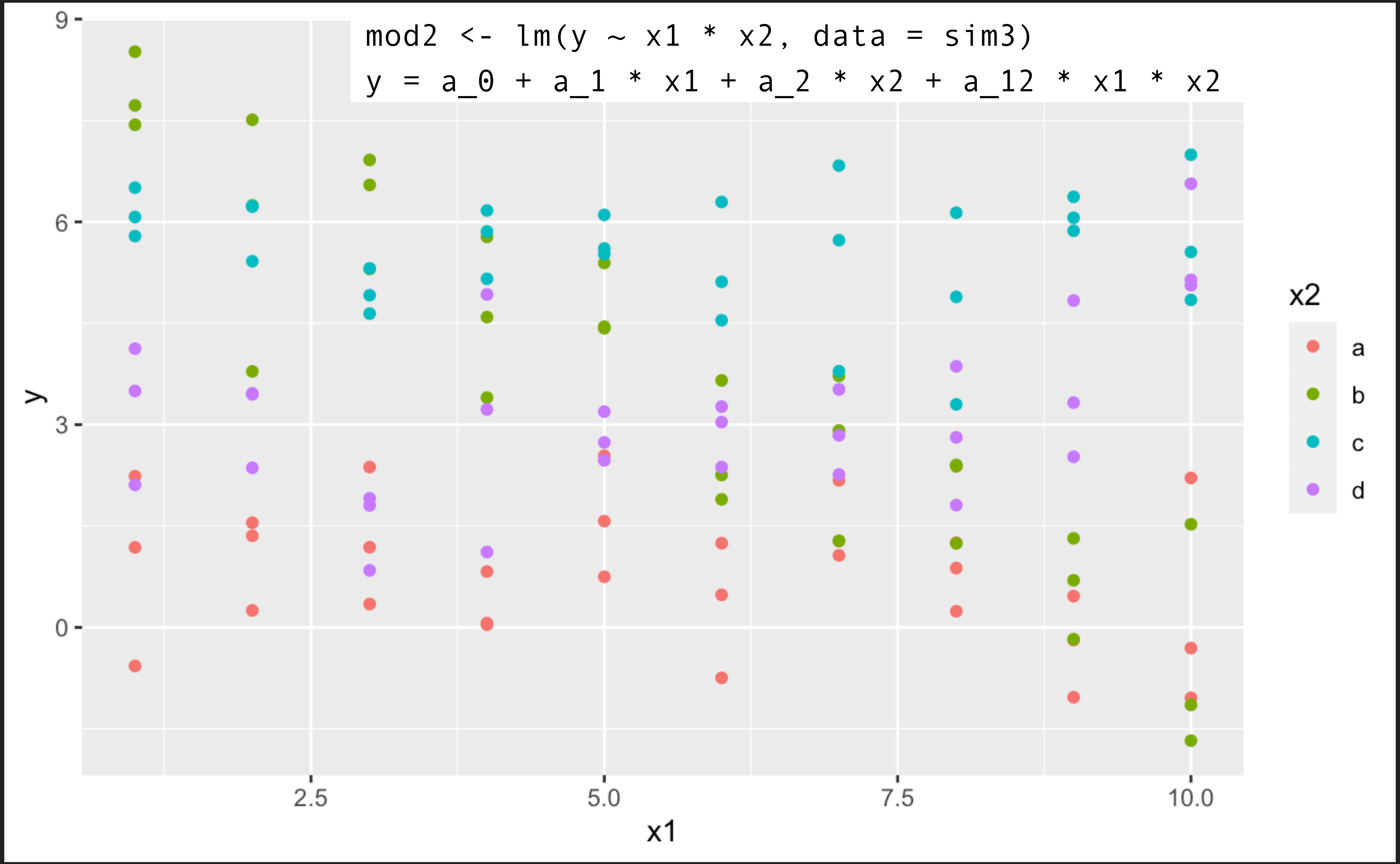


There are two possible models you could fit to this data



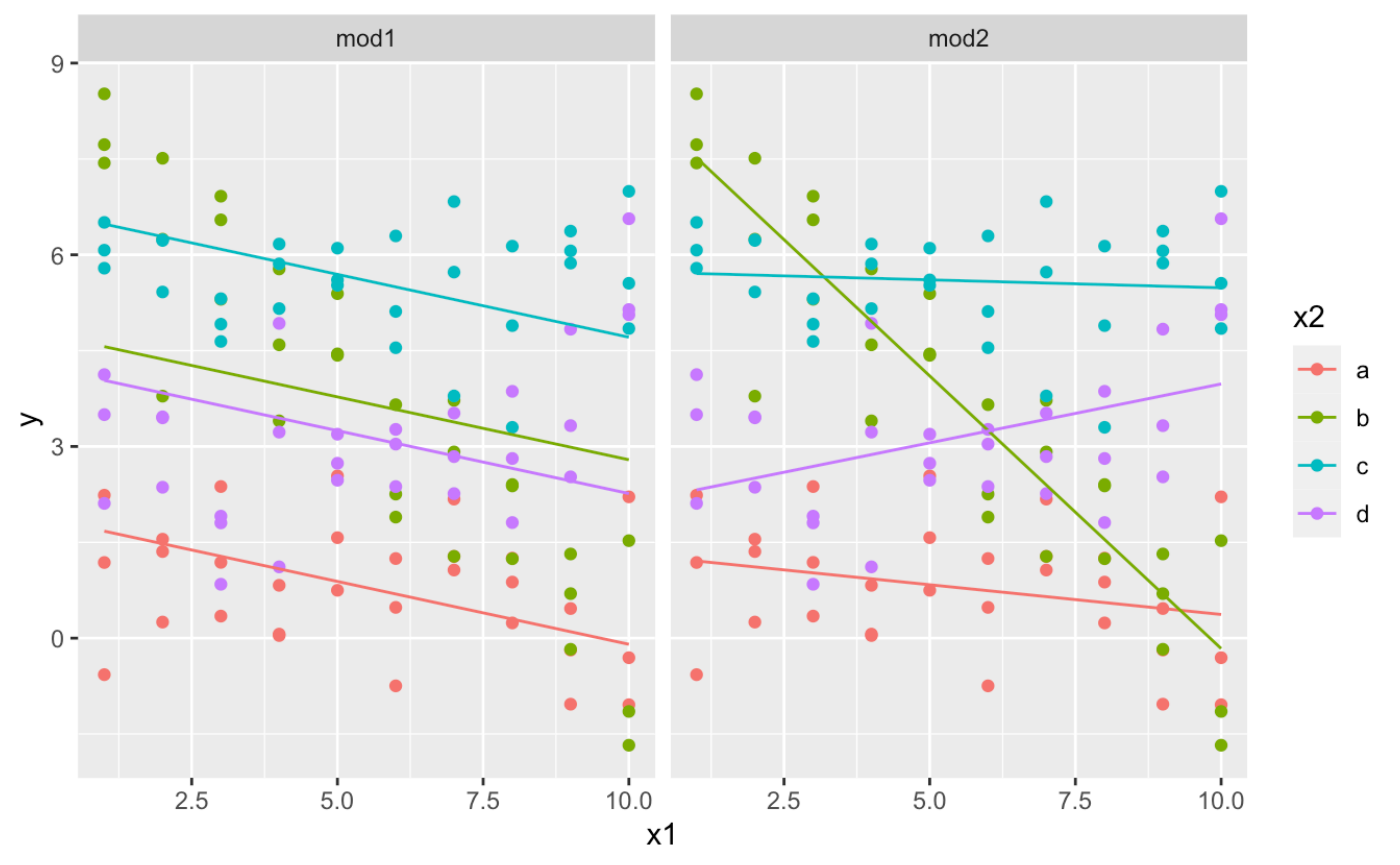
```
mod1 <- lm(y ~ x1 + x2, data = sim3)
mod2 <- lm(y ~ x1 * x2, data = sim3)
```

*: Both the interaction and the individual components are included in the model

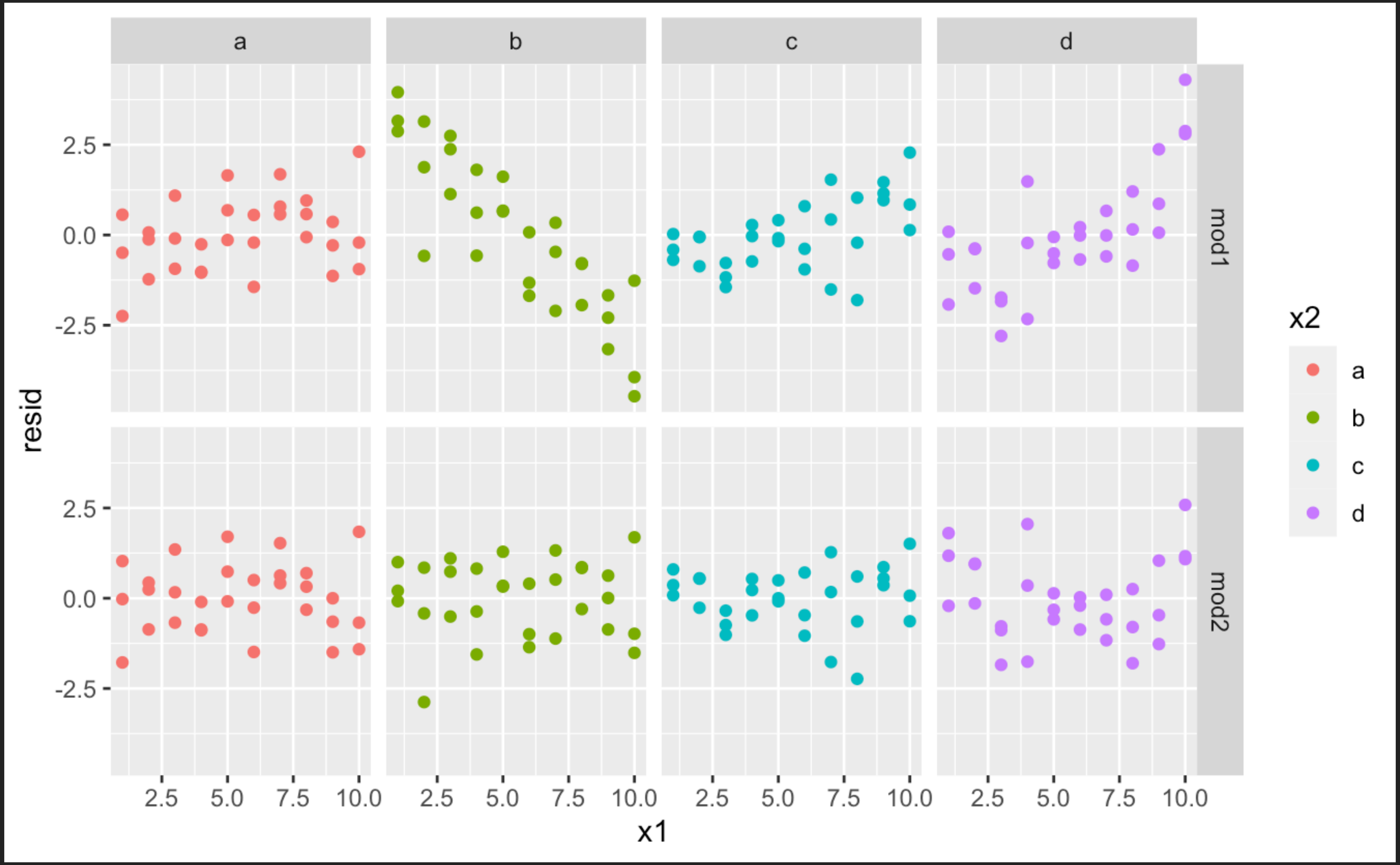


```
mod2 <- lm(y ~ x1 * x2, data = sim3)
y = a_0 + a_1 * x1 + a_2 * x2 + a_12 * x1 * x2
```

The model using * has a different slope and intercept for each line

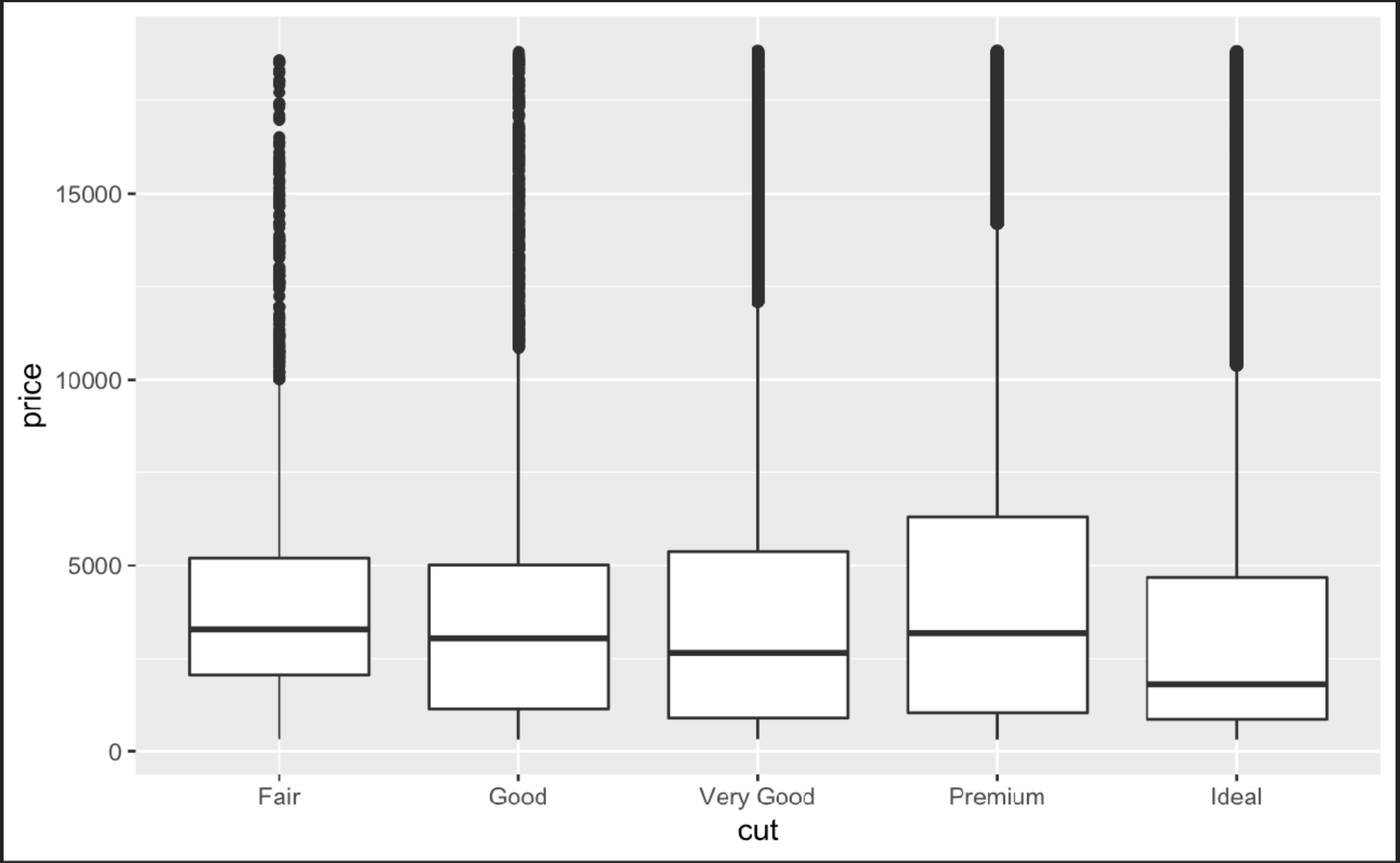


Which model is better for this data?



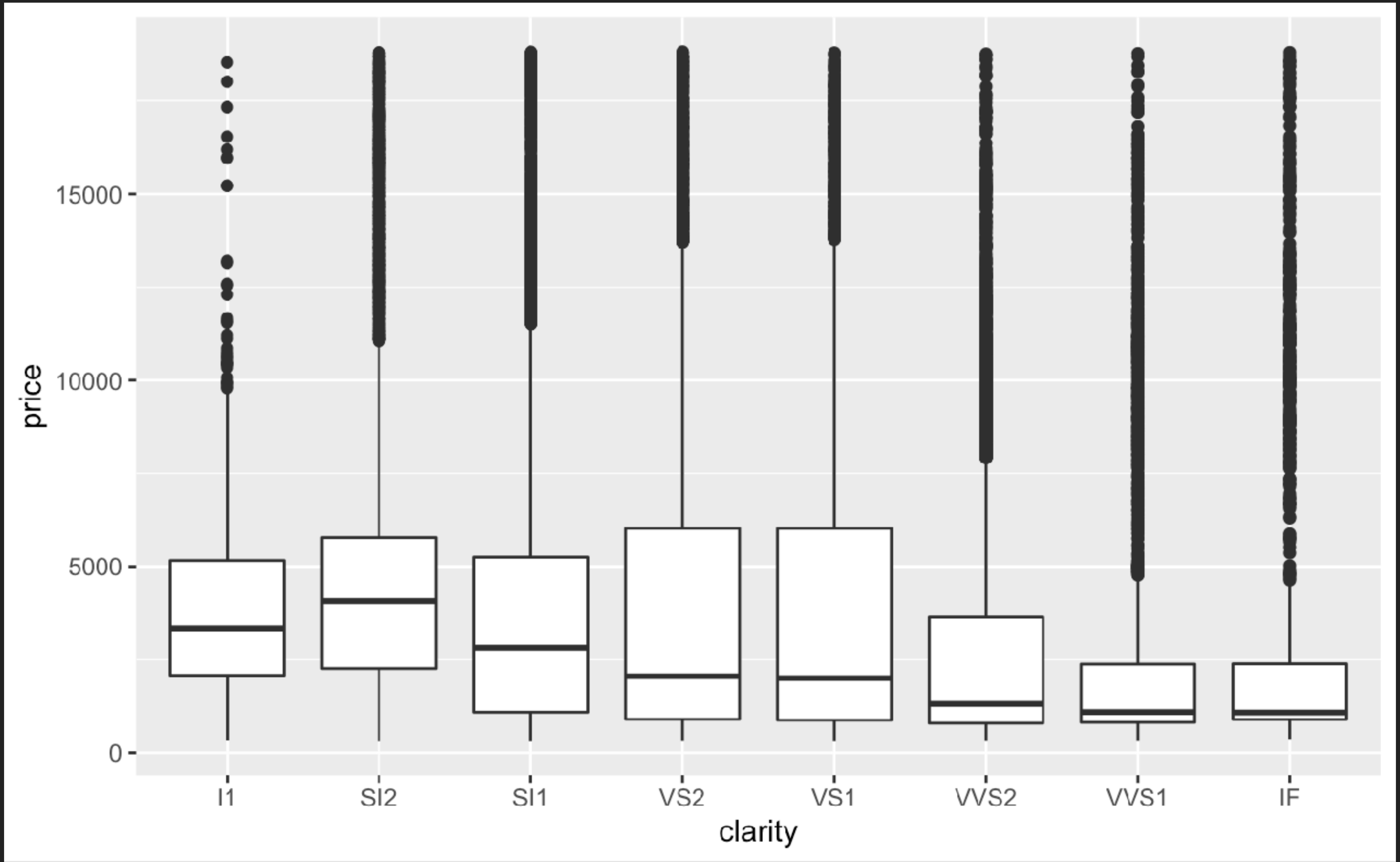
Another example

Why are low quality diamonds more expensive?



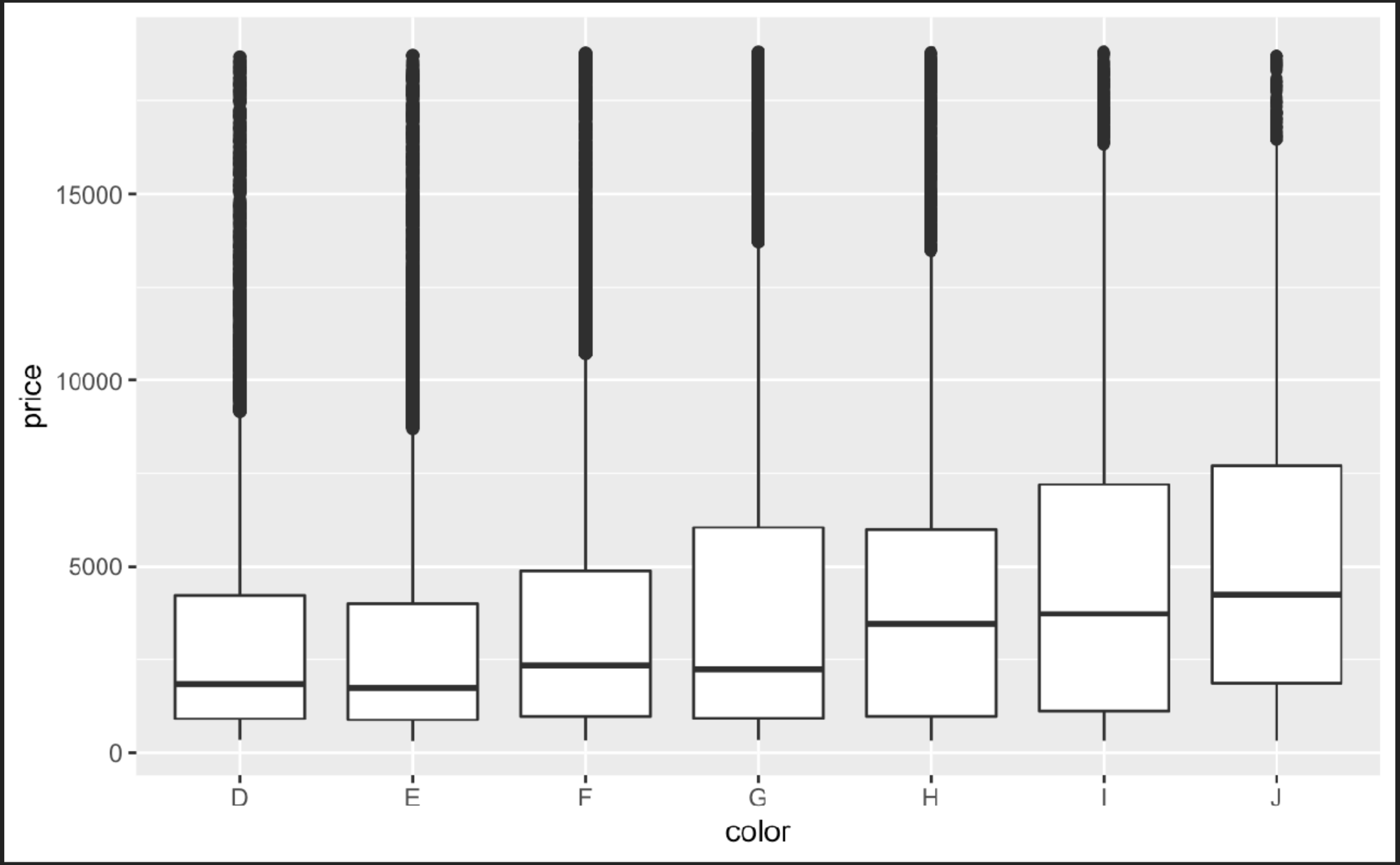
- Fair: worst cut

Why are low quality diamonds more expensive?



- I1: inclusions visible to the naked eye (worst clarity)

Why are low quality diamonds more expensive?



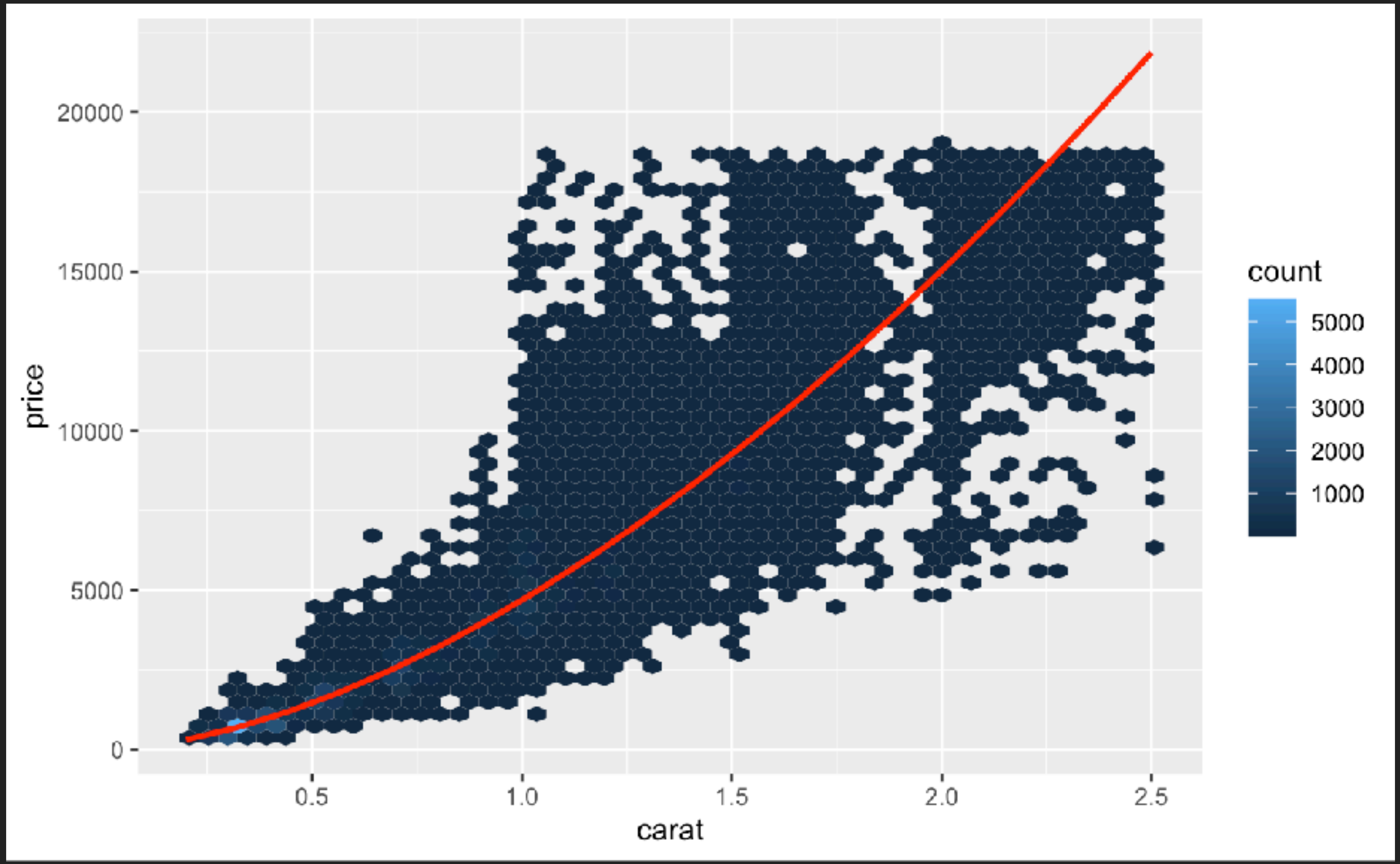
- J: slightly yellow (worst color)

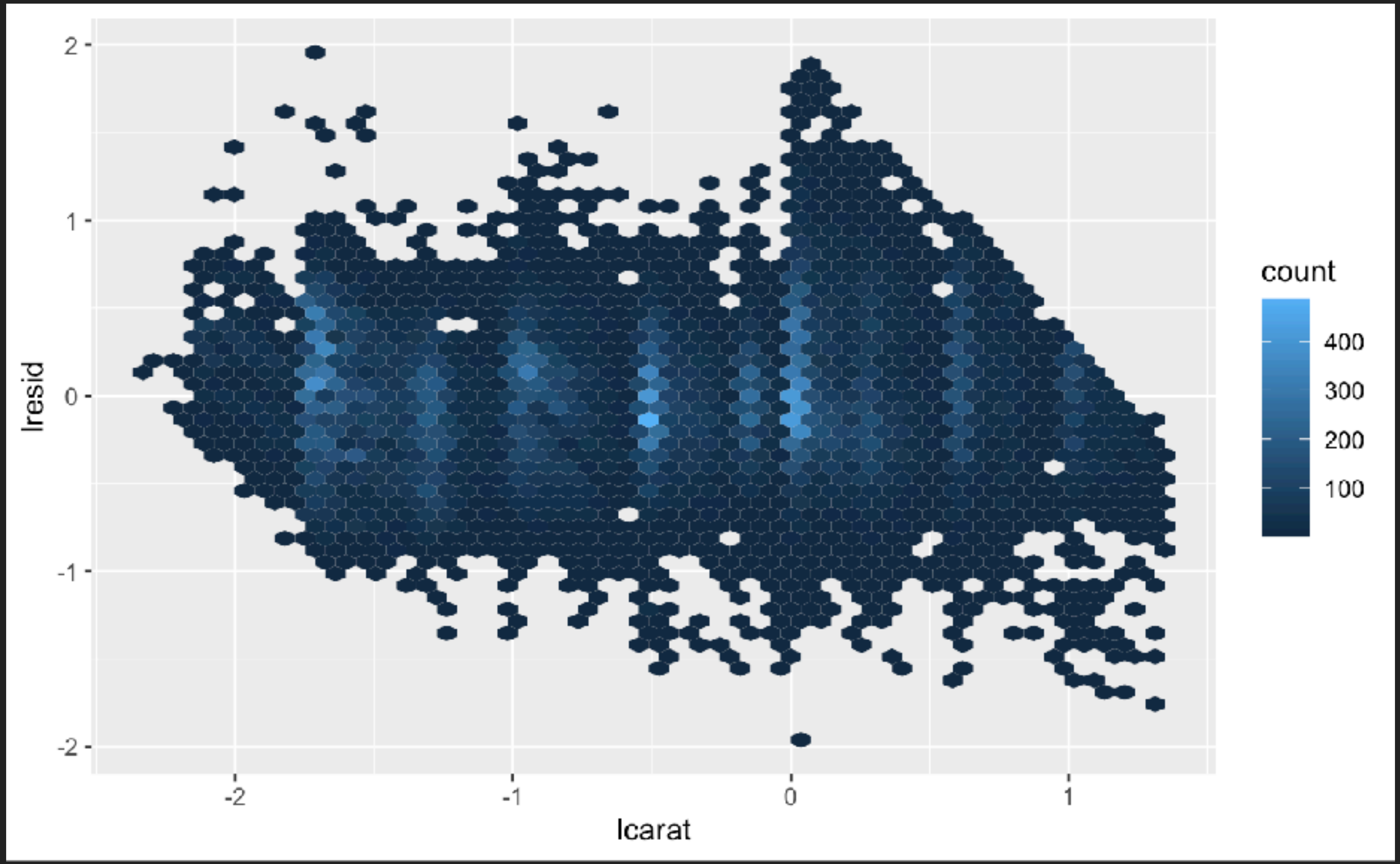
Because lower quality diamonds tend to be larger

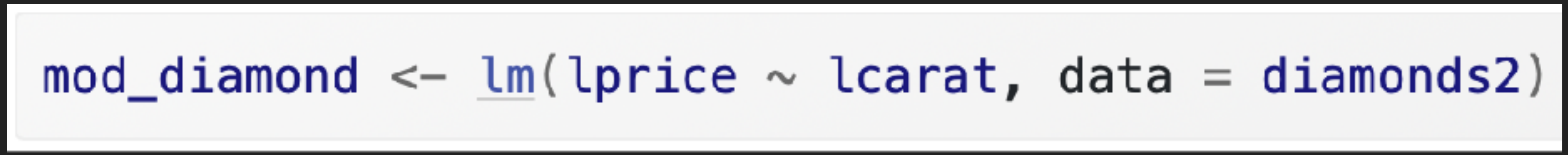
price
15000
10000
5000
0
0
1
2
3
4
5
carat
count
5000
4000
3000
2000
1000
lprice
14
12
10
-2
-1
0
1
lcarat
count
600
400
200

- The weight of the diamond is the single most important factor for determining the price of the diamond.
  - Left: raw
  - Right: log-transformed

Let's remove that strong linear pattern



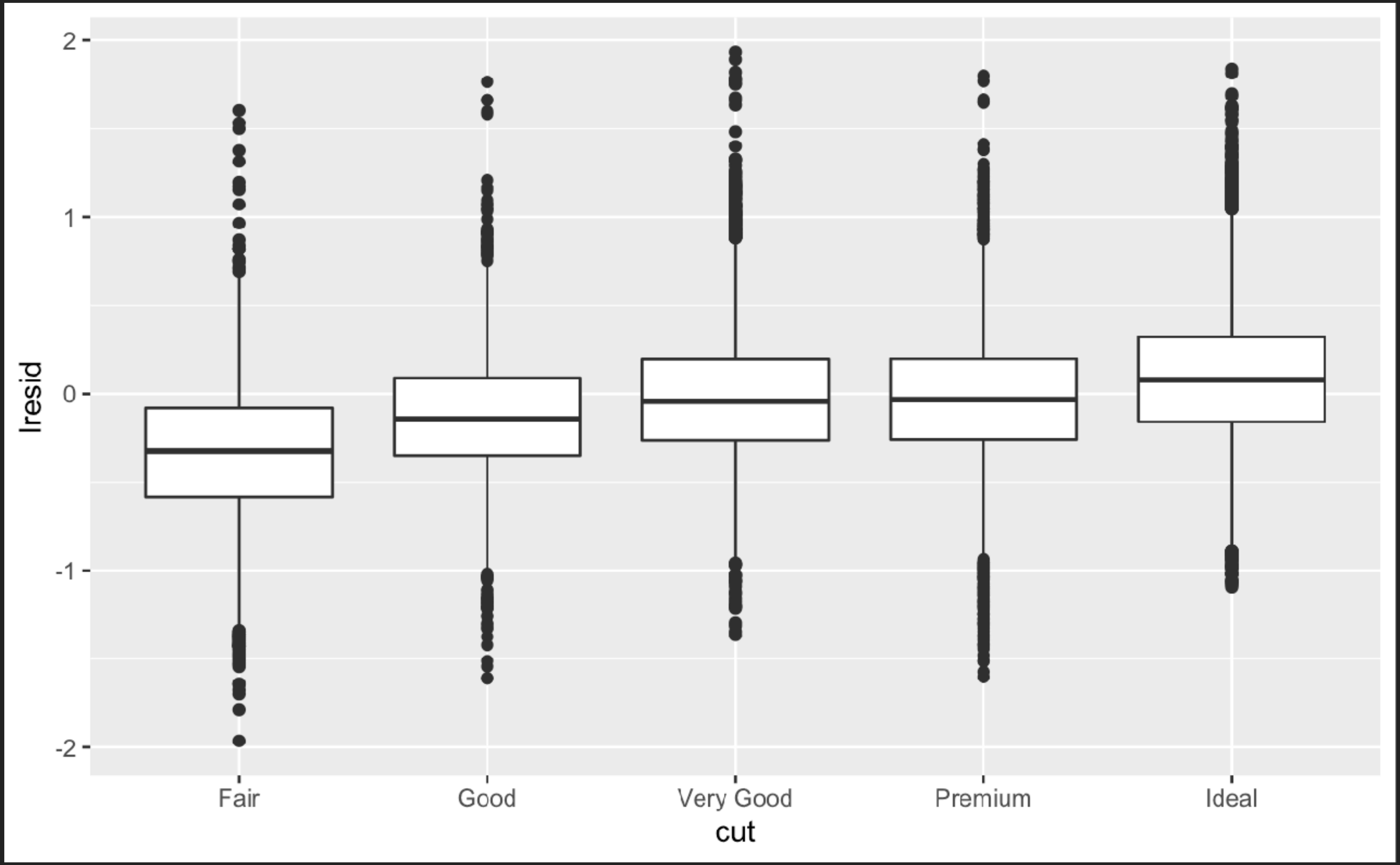




```
mod_diamond <- lm(lprice ~ lcarat, data = diamonds2)
```

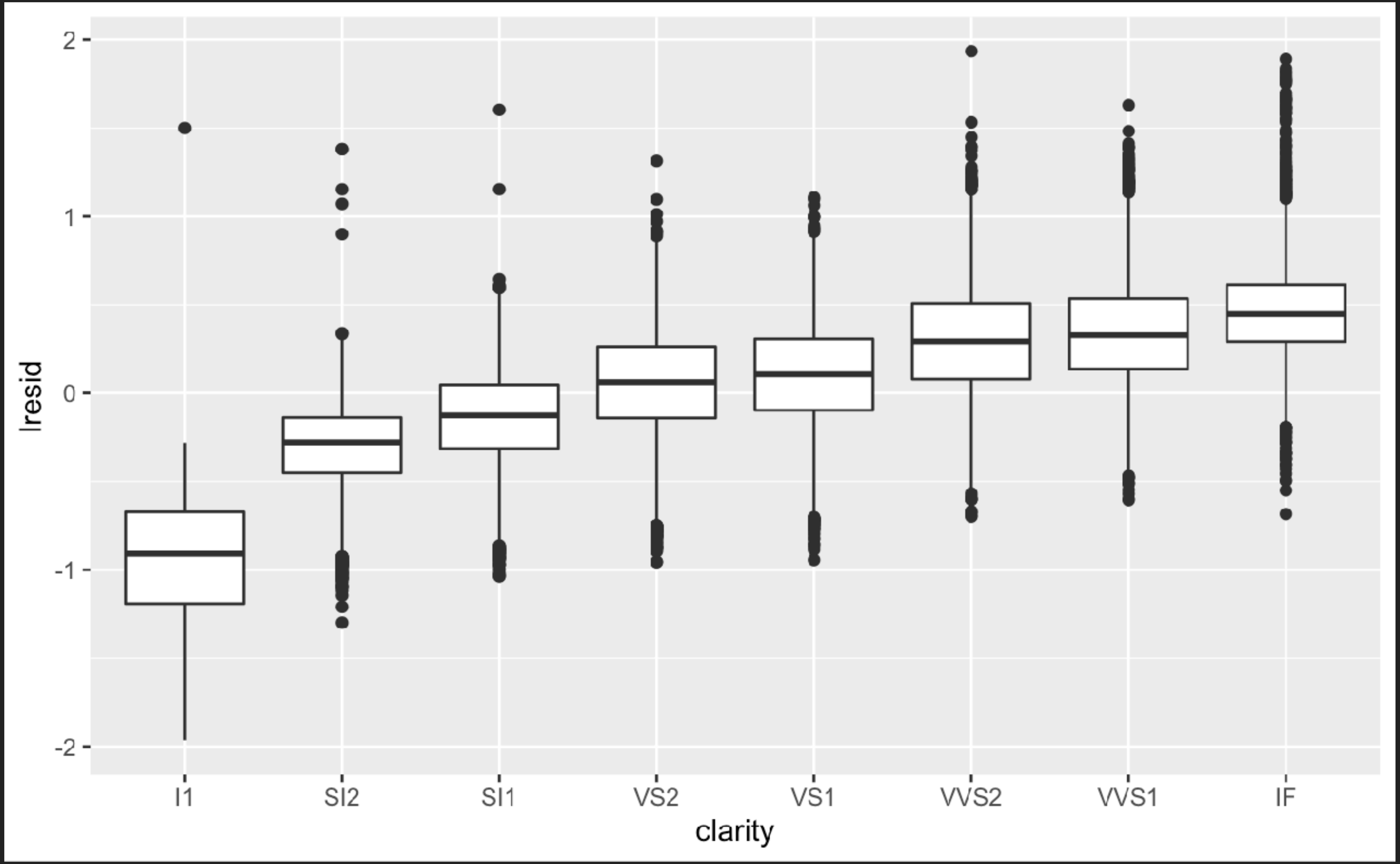
- Residuals confirm that we've successfully removed the strong linear pattern.

Now we see the relationship we expect



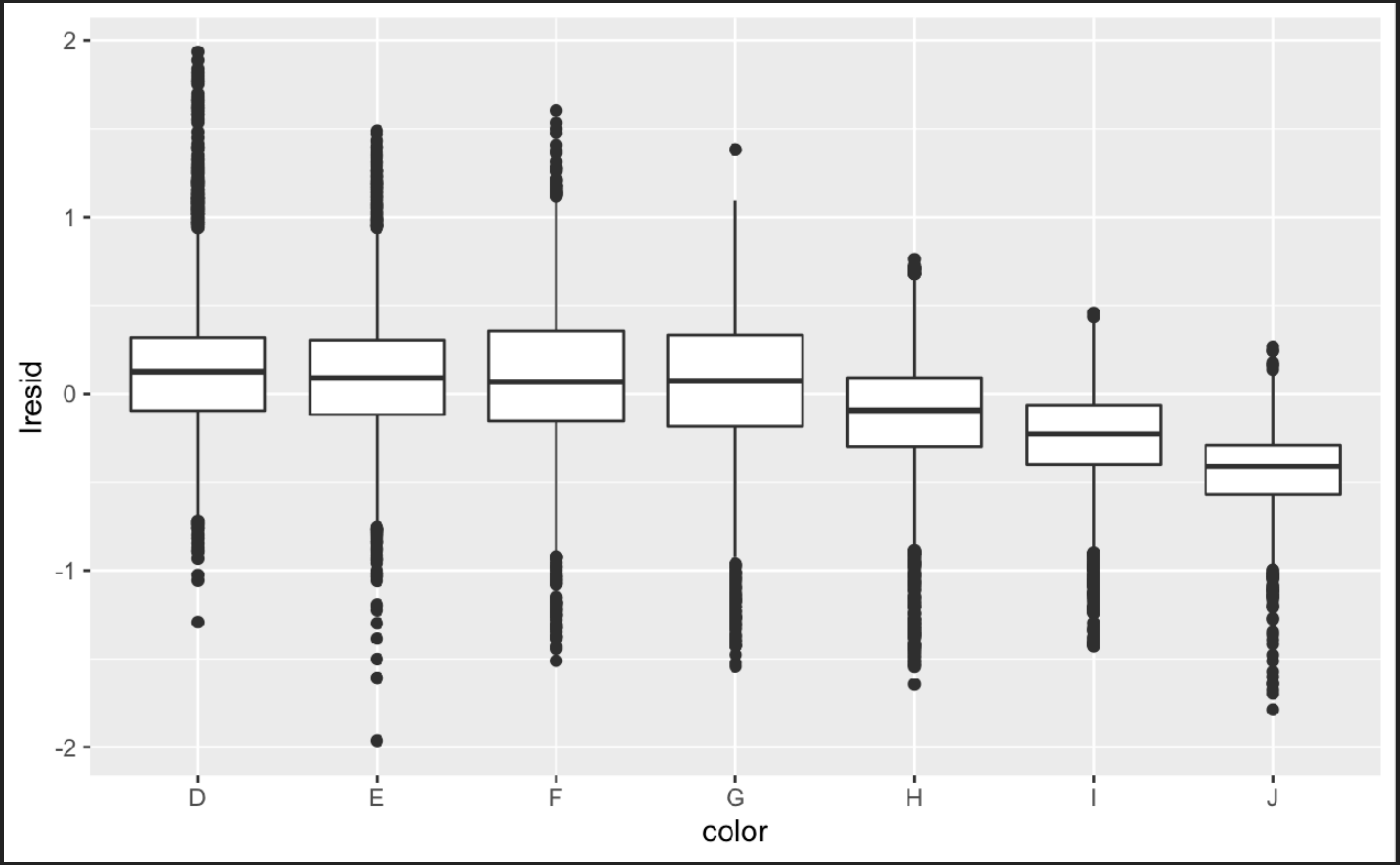
- Re-did our motivating plots using those residuals instead of price.
- Fair: worst cut

Now we see the relationship we expect



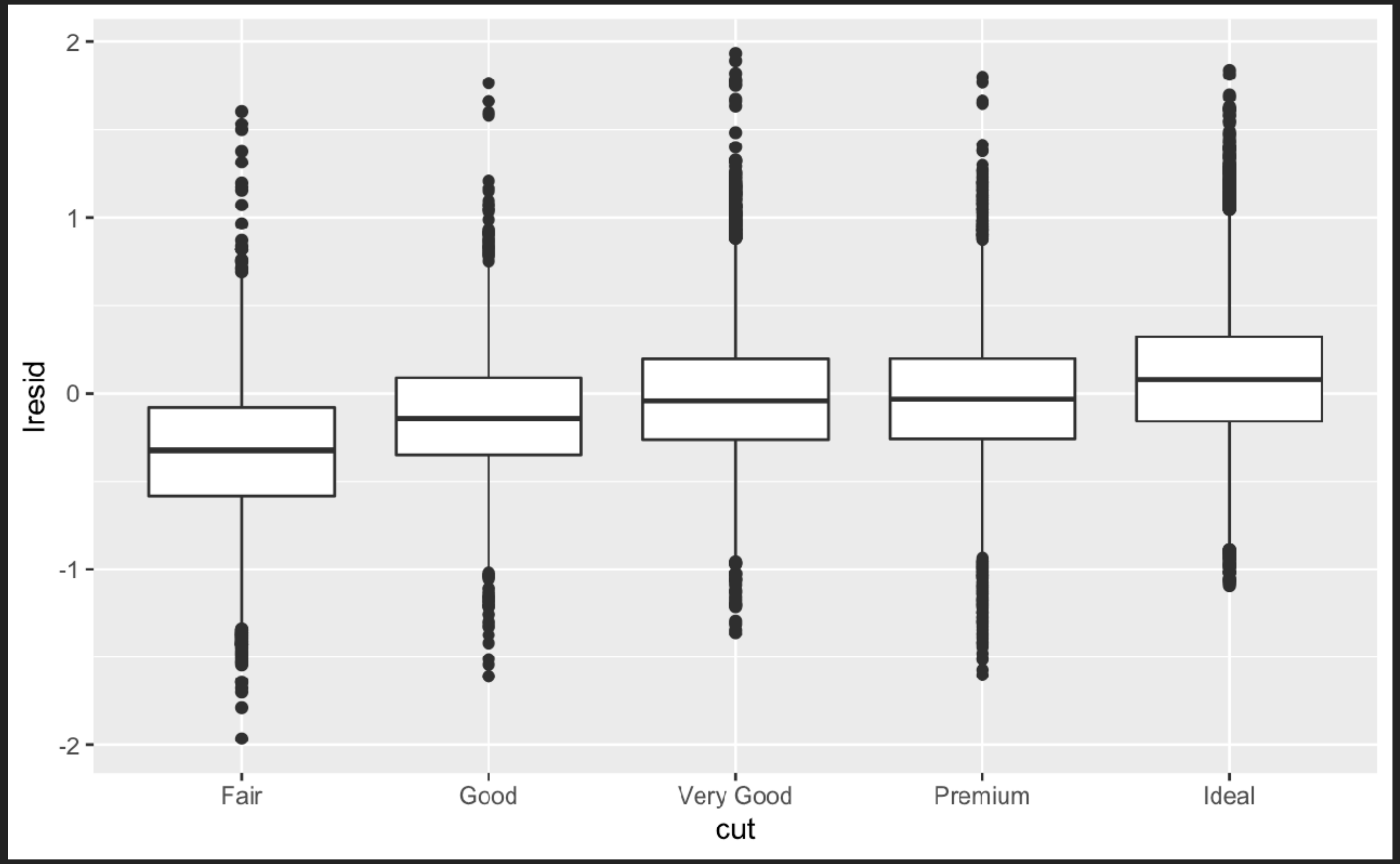
- I1: inclusions visible to the naked eye (worst clarity)

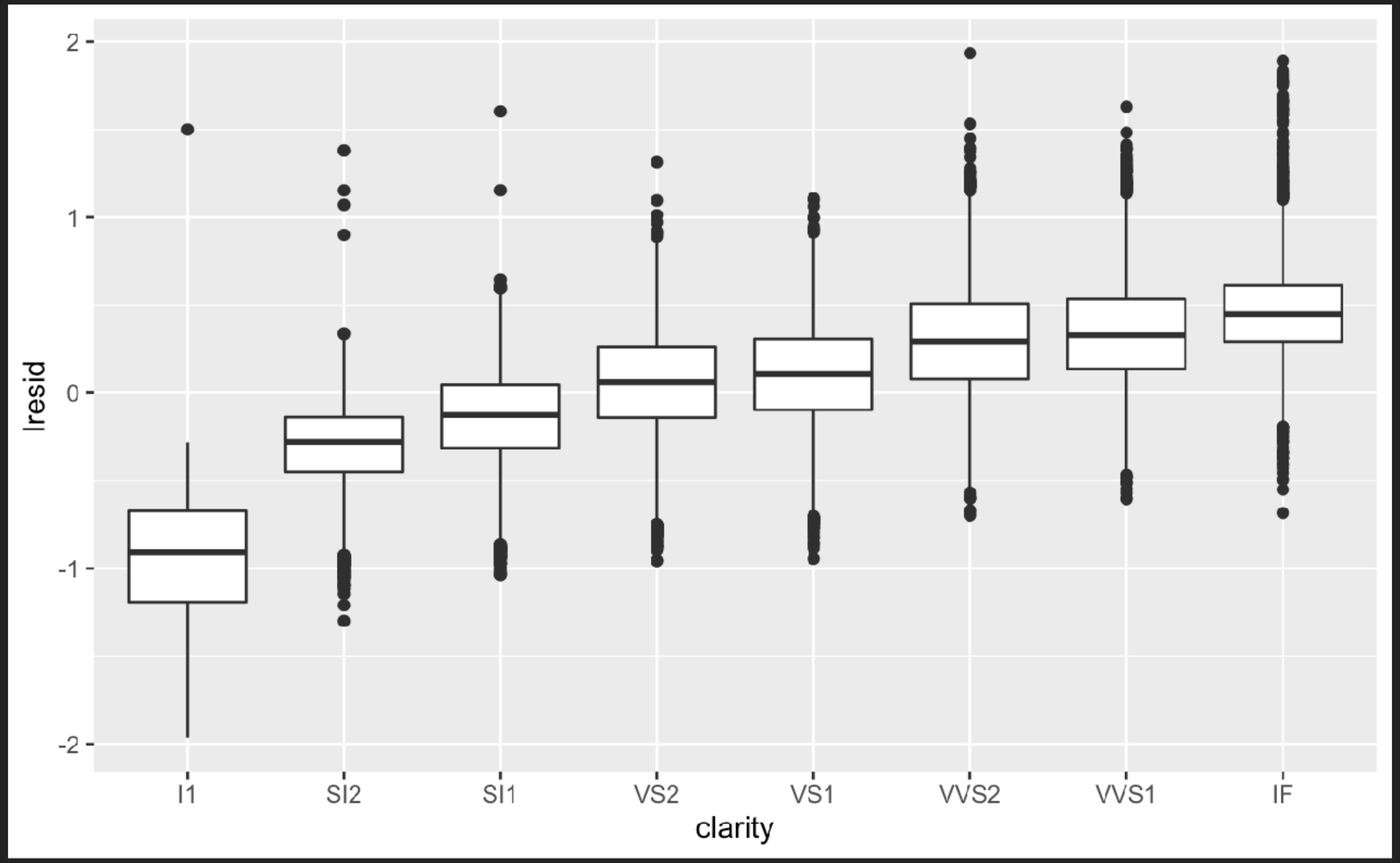
Now we see the relationship we expect

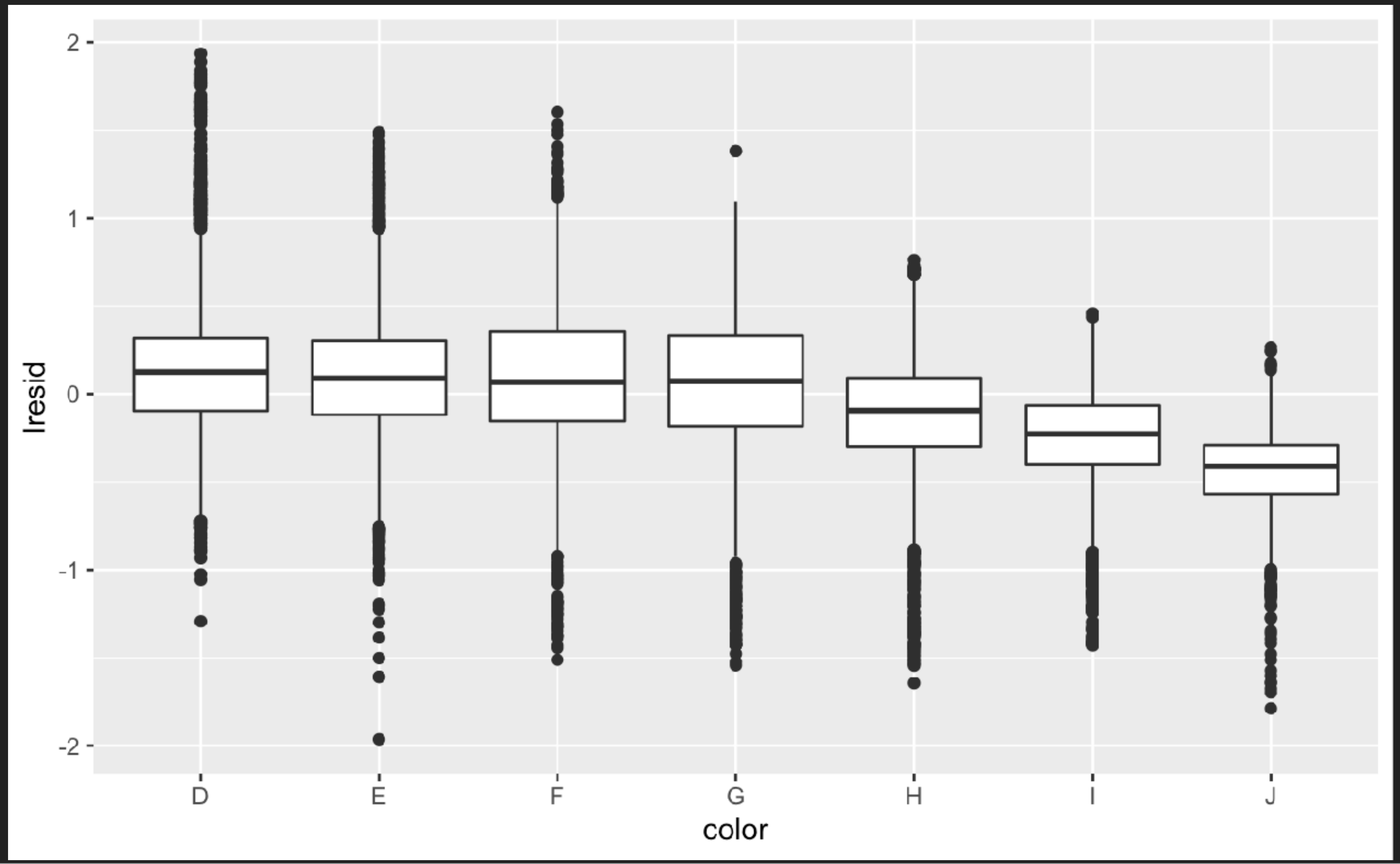


- J: slightly yellow (worst color)

Now we see the relationship we expect

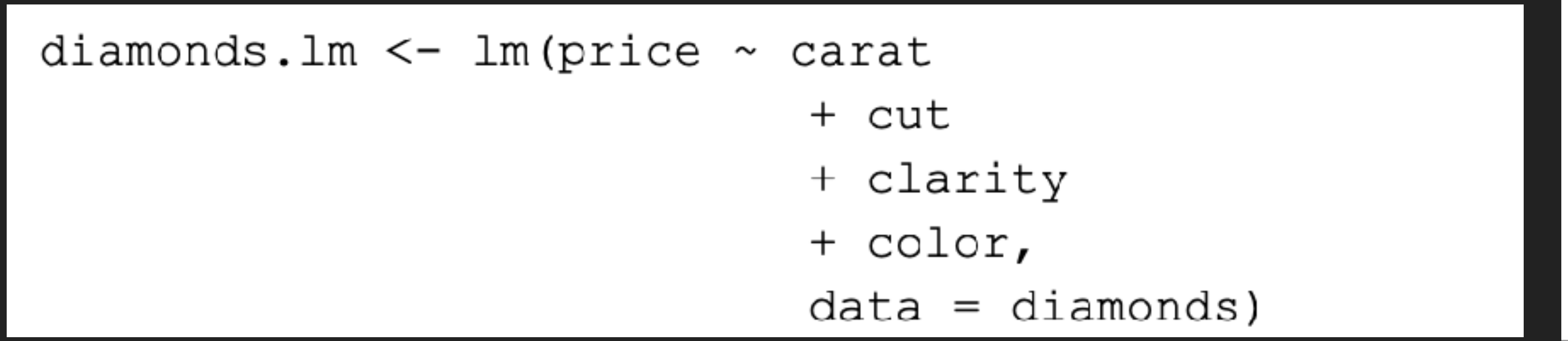




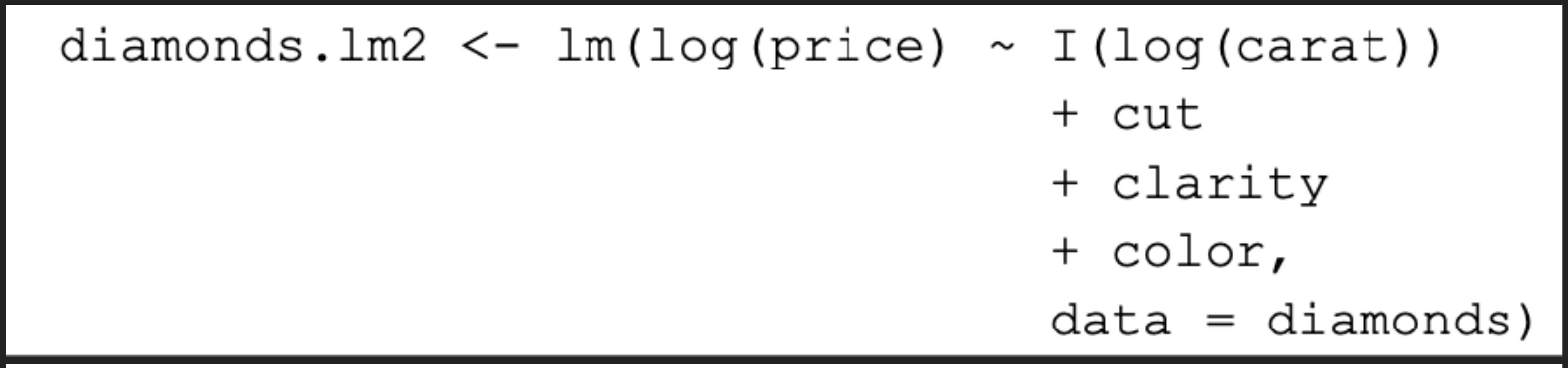


- Re-did our motivating plots using those residuals instead of price.

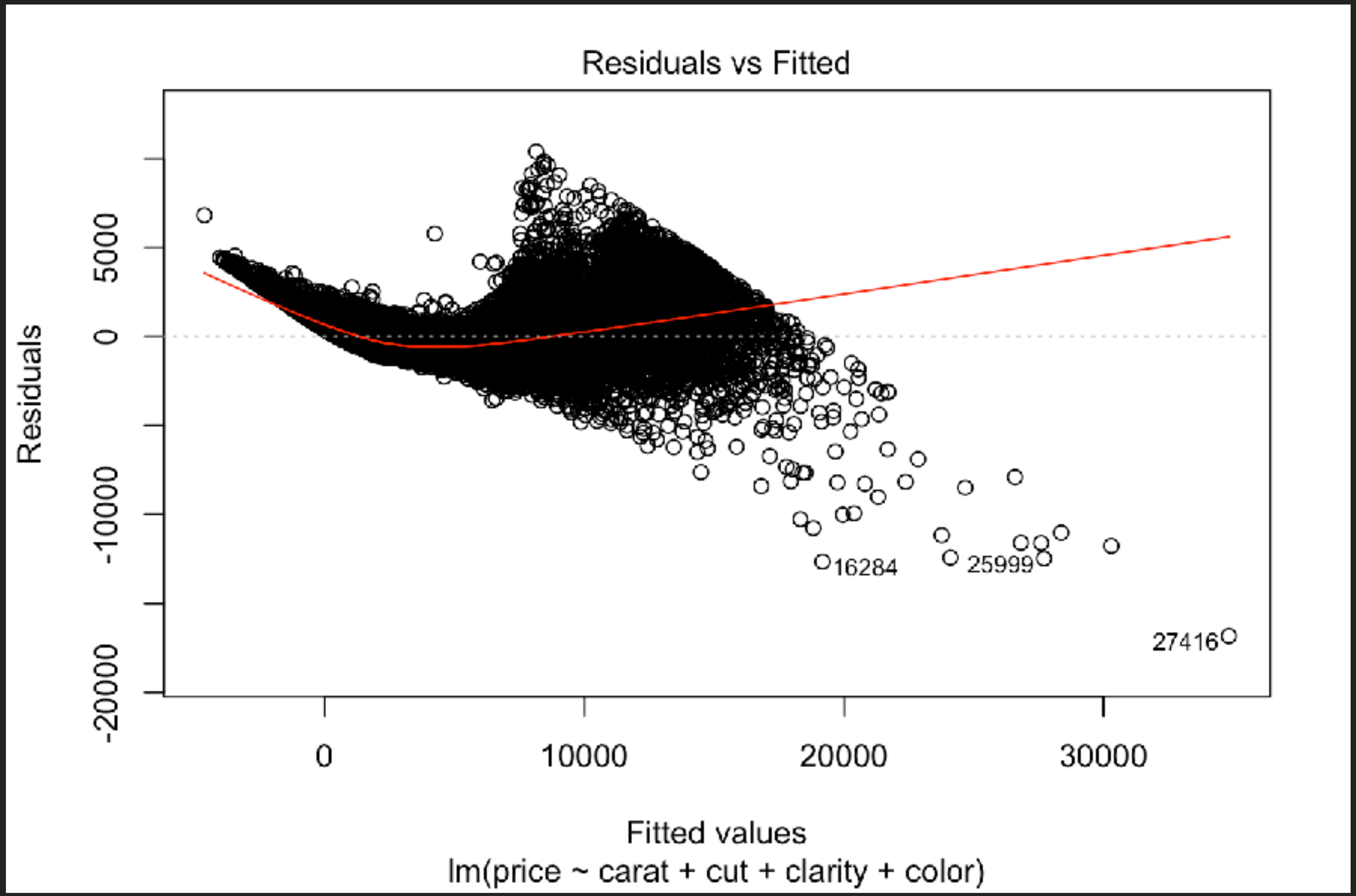
Regression Diagnostics on the Diamonds Example

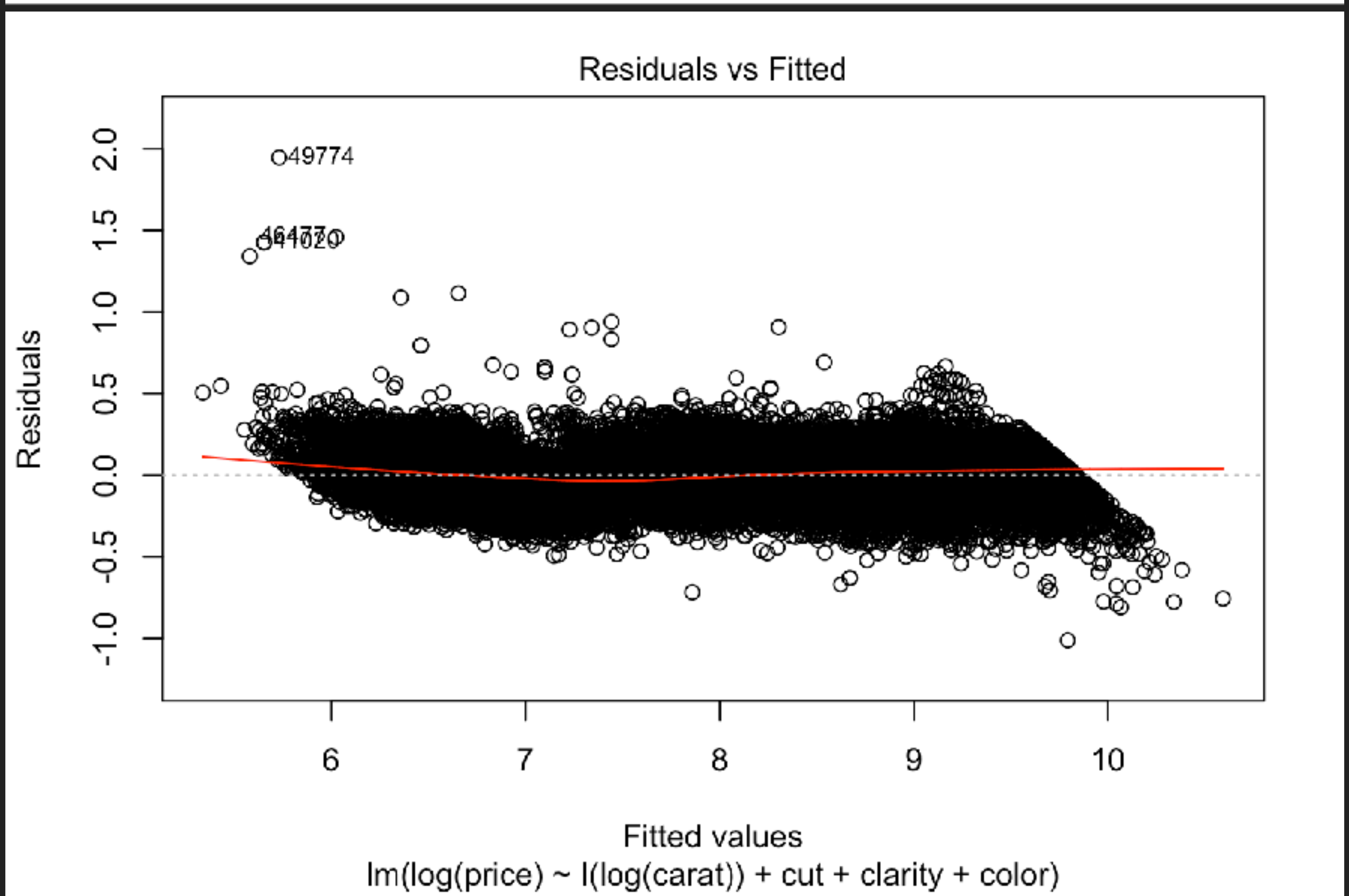


```
diamonds.lm <- lm(price ~ carat
                        + cut
                        + clarity
                        + color,
                        data = diamonds)
```



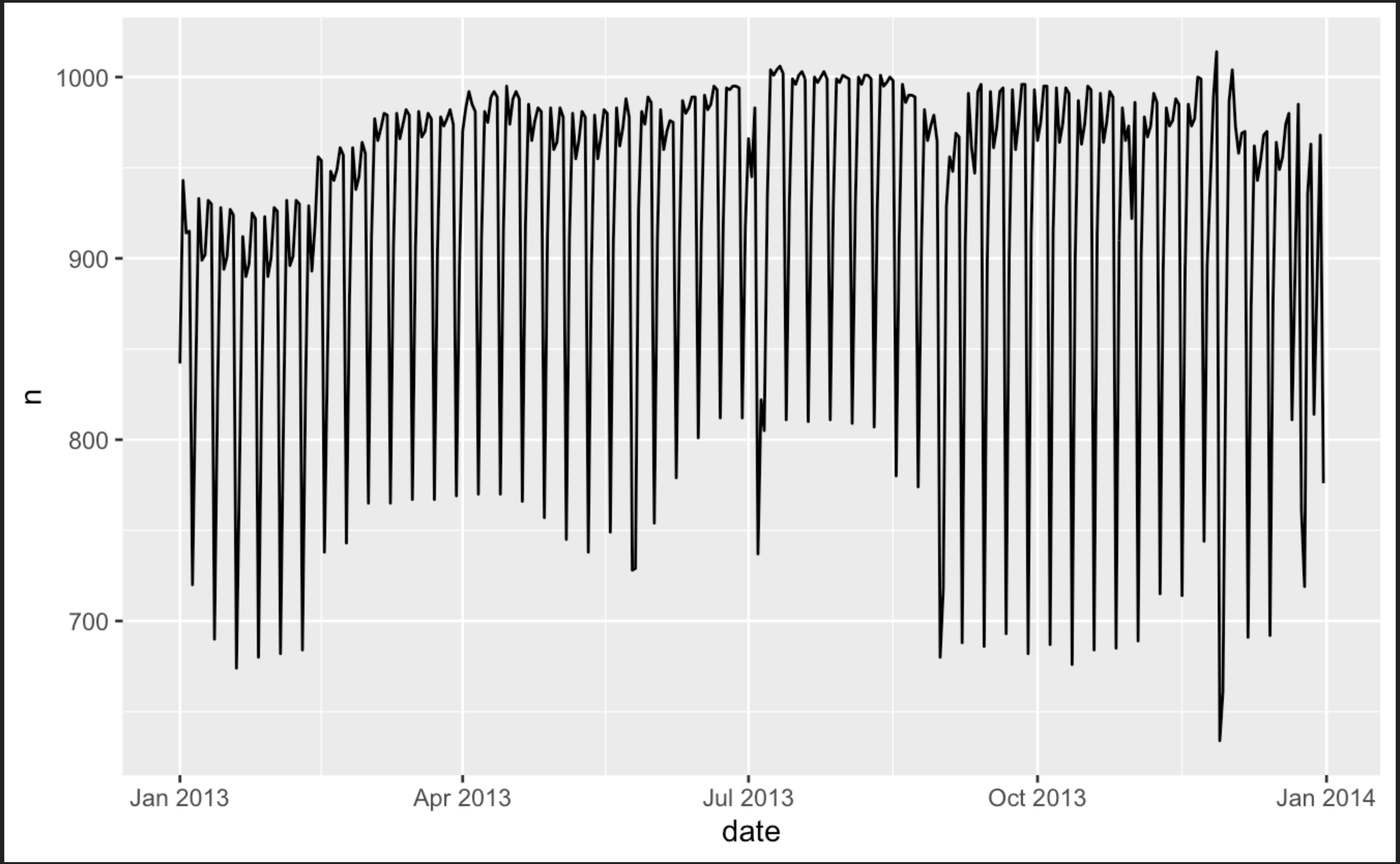
```
diamonds.lm2 <- lm(log(price) ~ I(log(carat))
                               + cut
                               + clarity
                               + color,
                               data = diamonds)
```



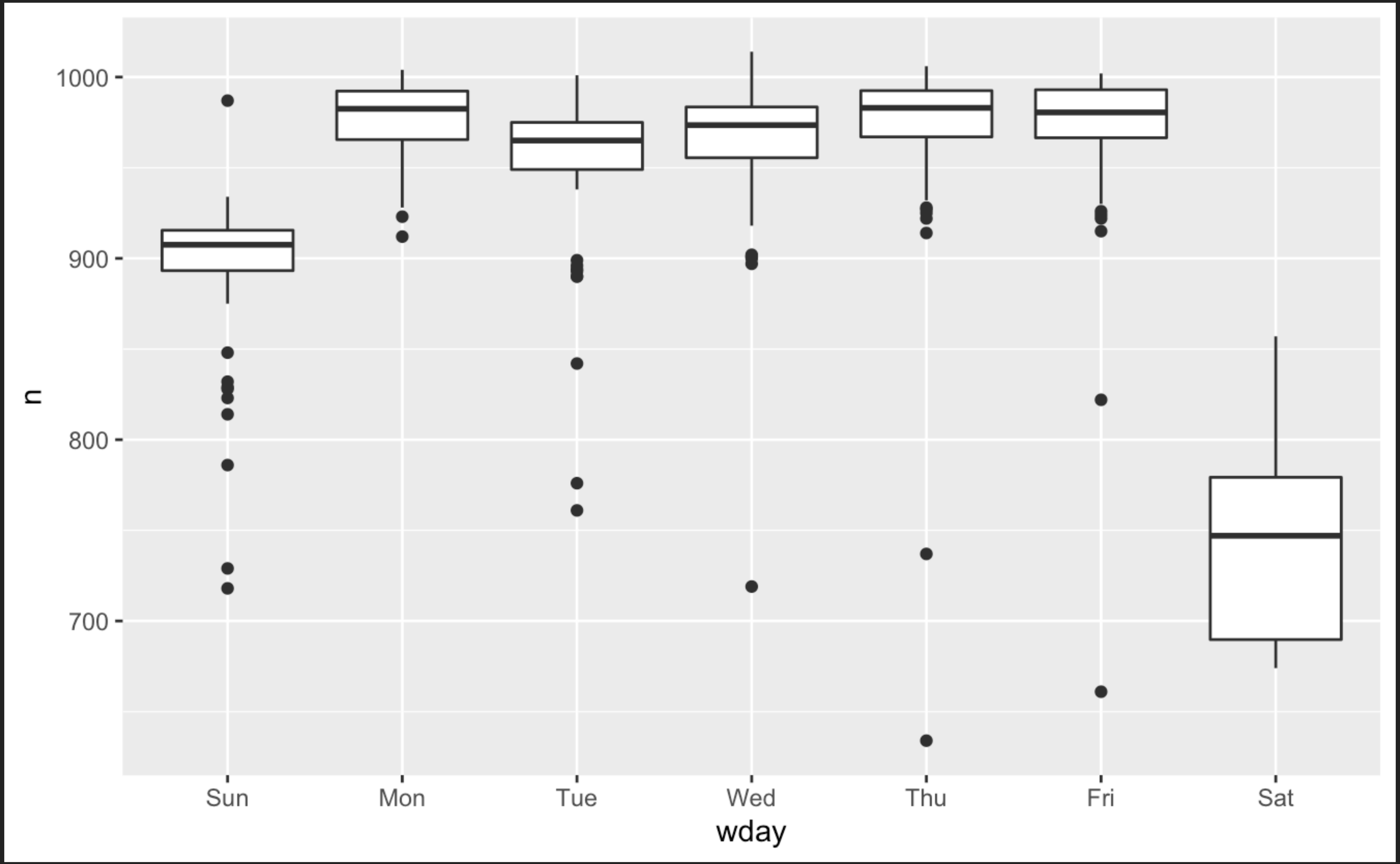


Another example

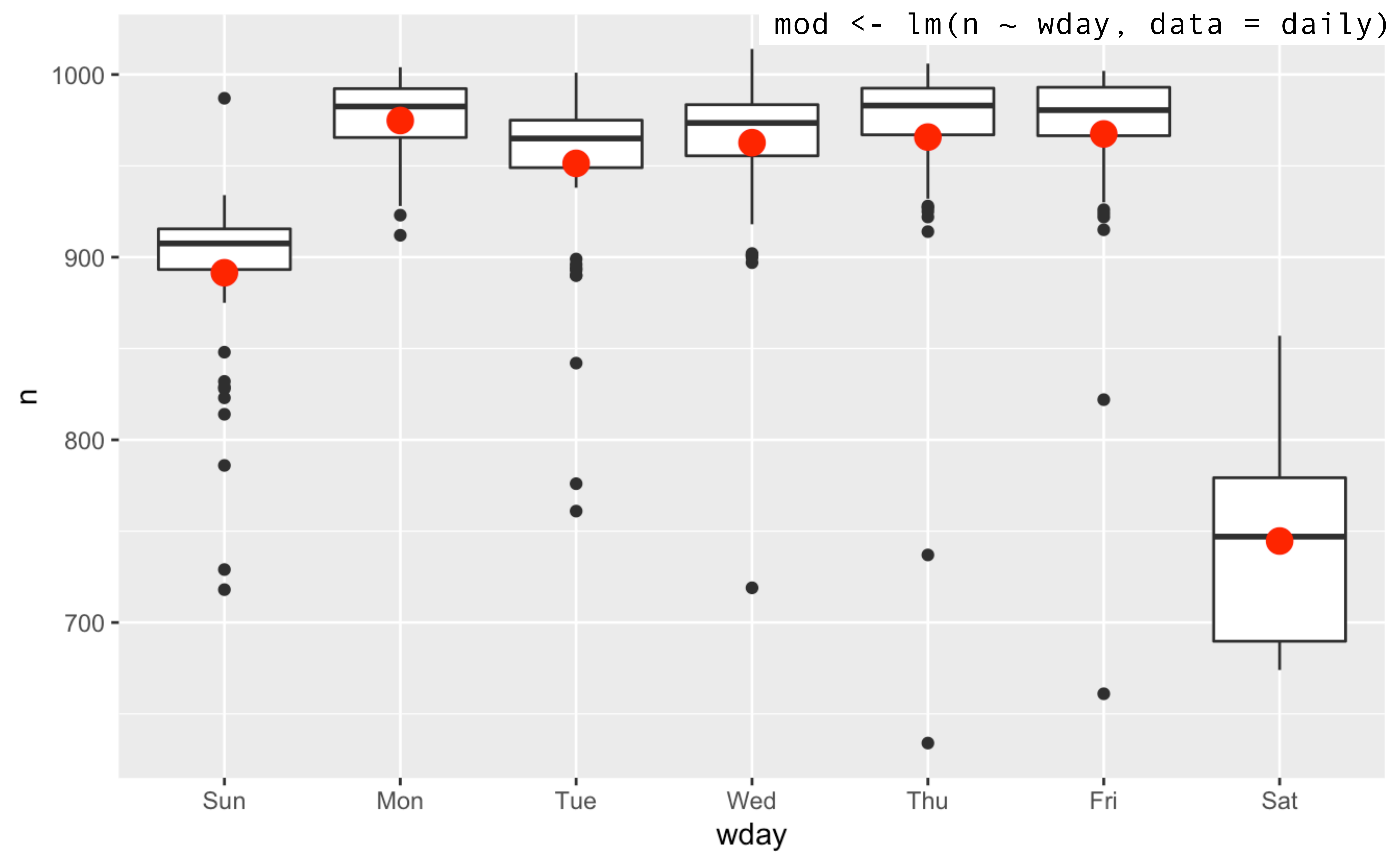
The number of flights that leave NYC per day



A very strong day-of-week effect dominates the subtler patterns

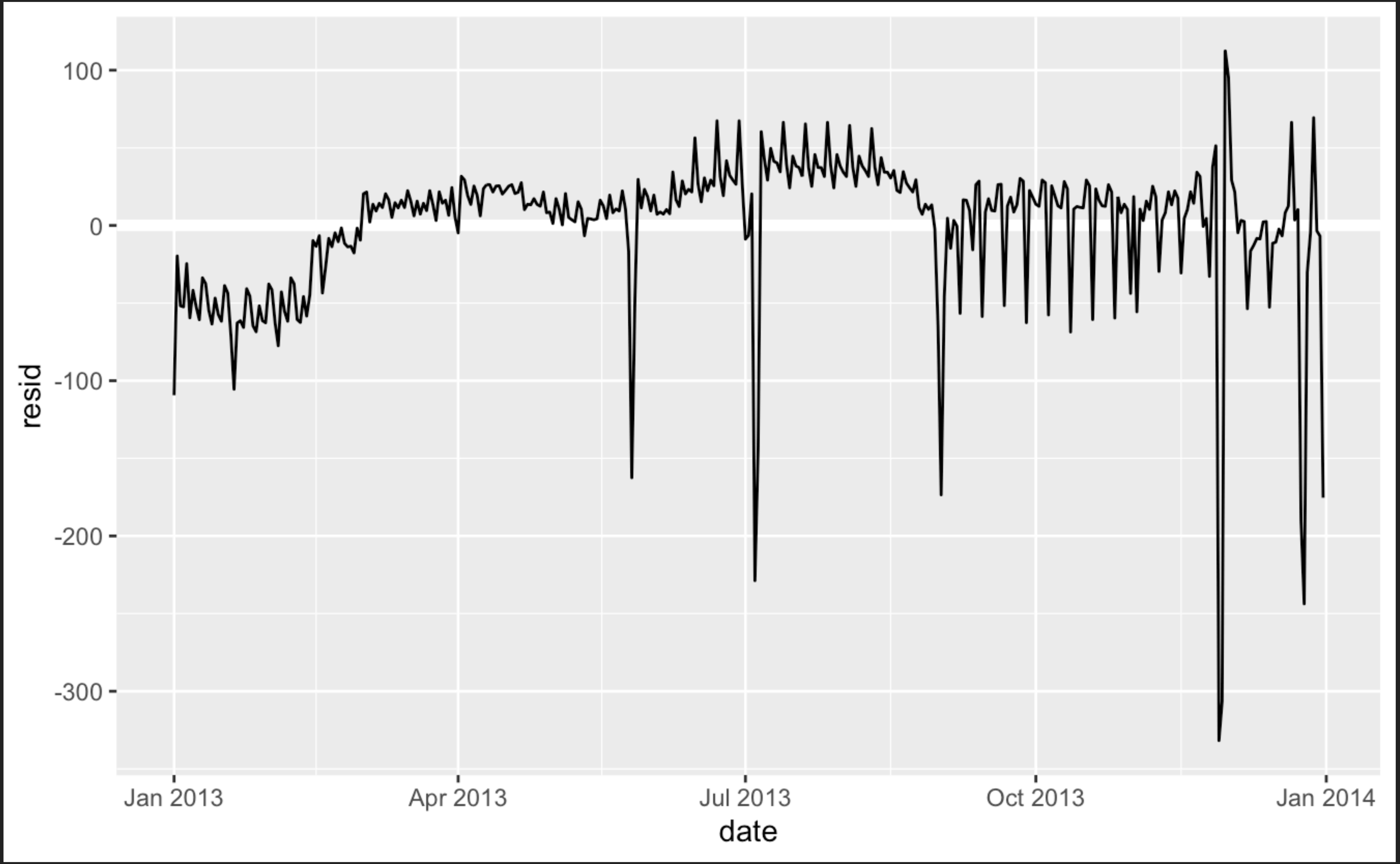


Modeling the week day effect

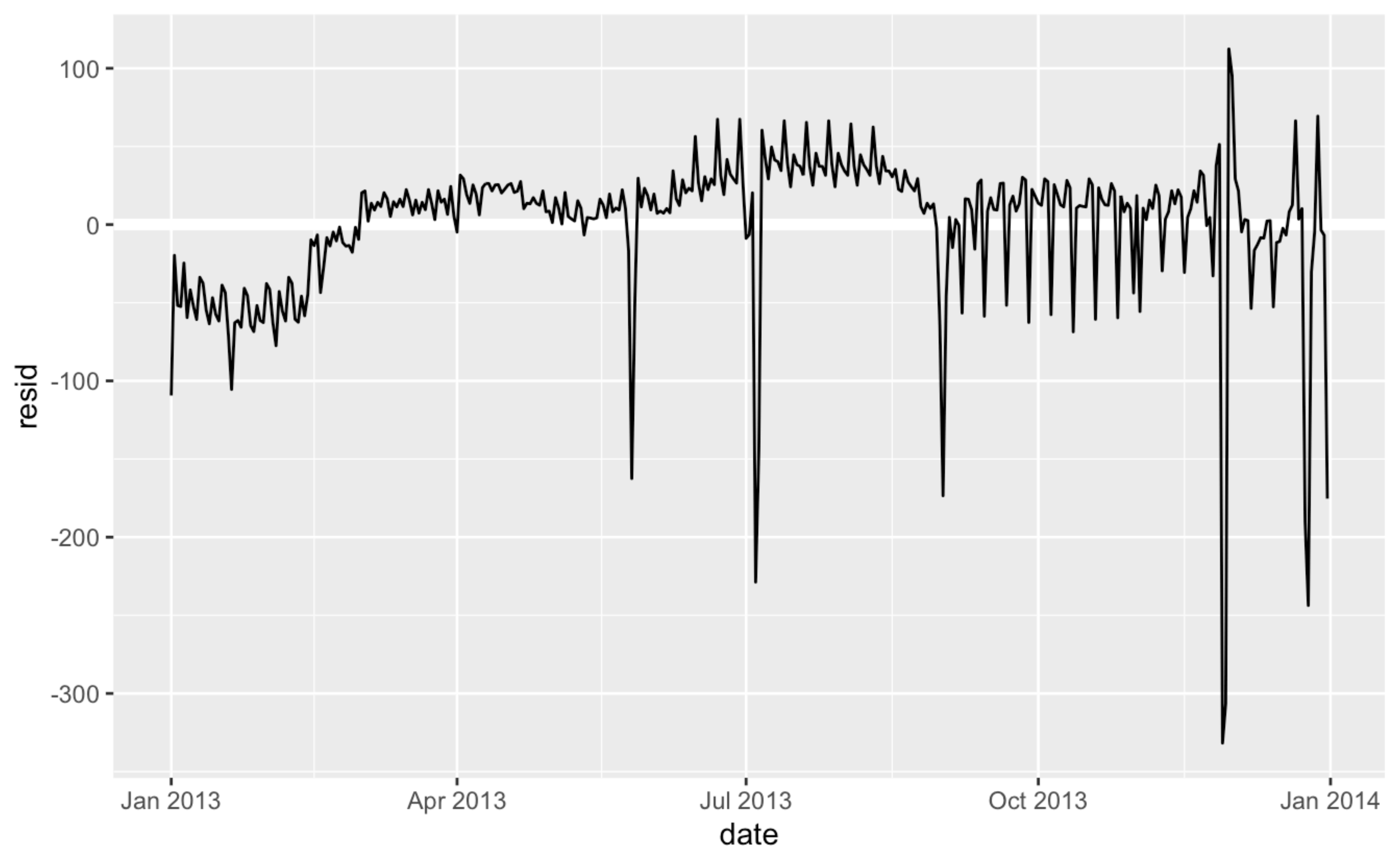


```
mod <- lm(n ~ wday, data = daily)
```

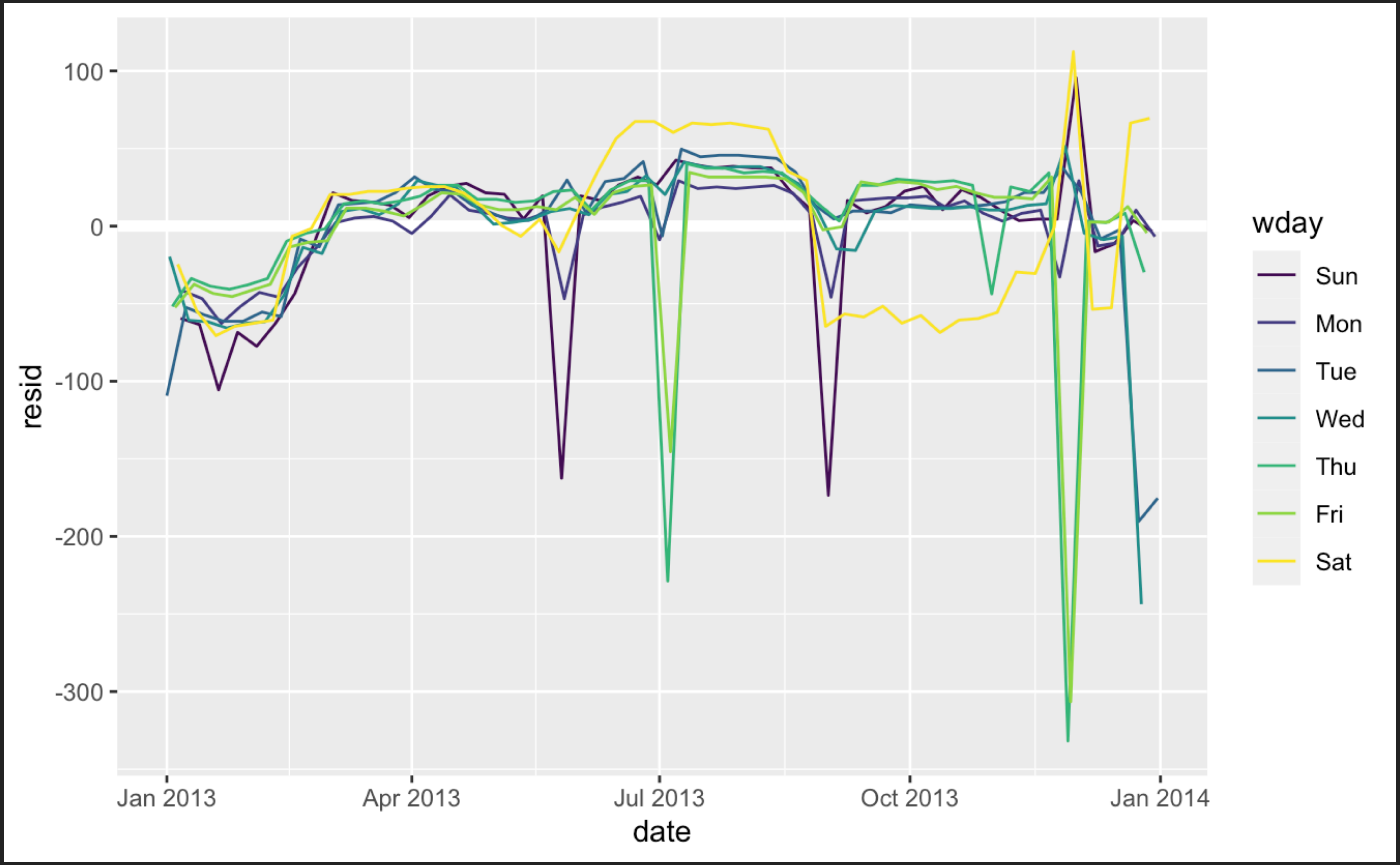
Visualizing the residuals



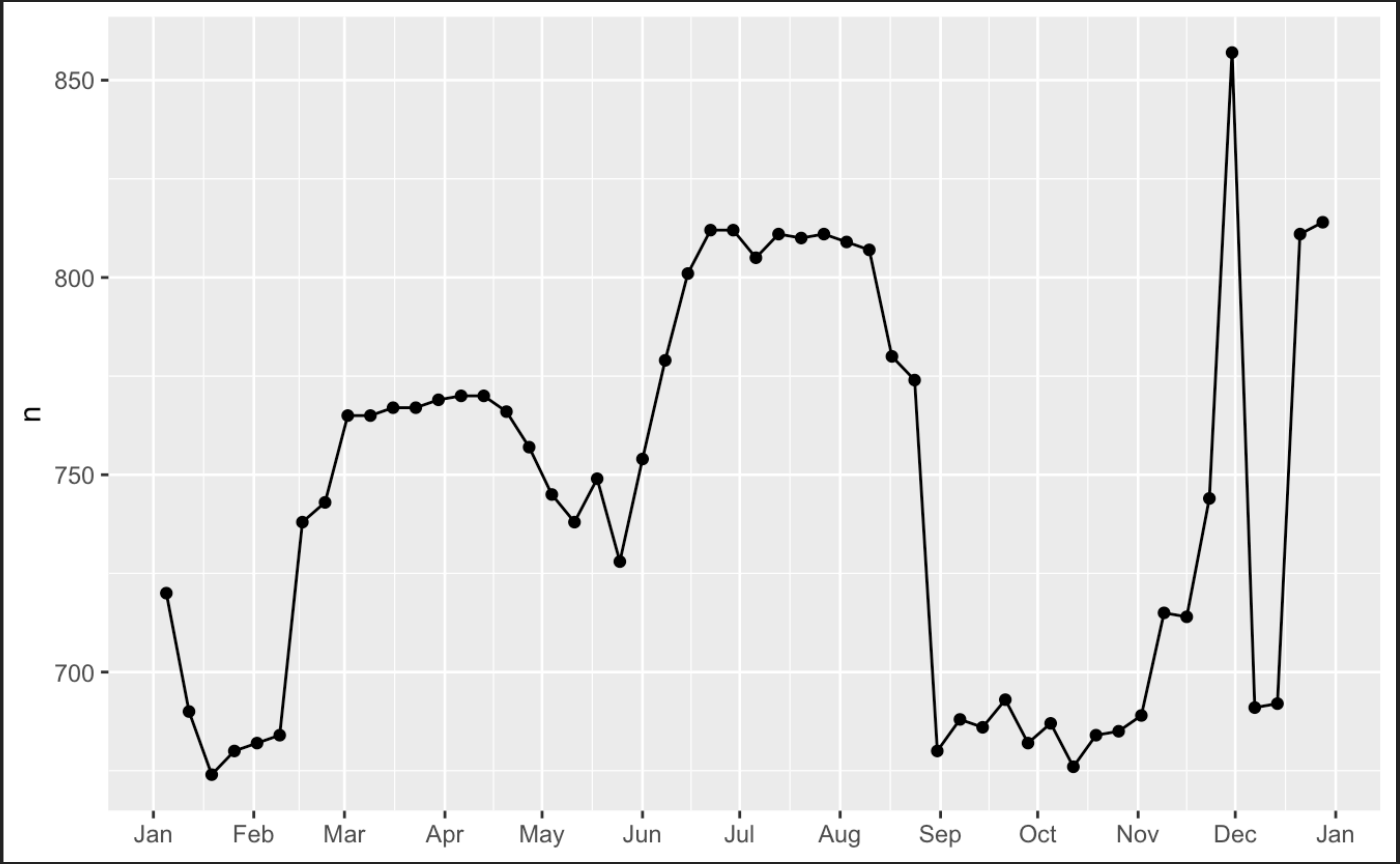
Our model seems to fail starting in June



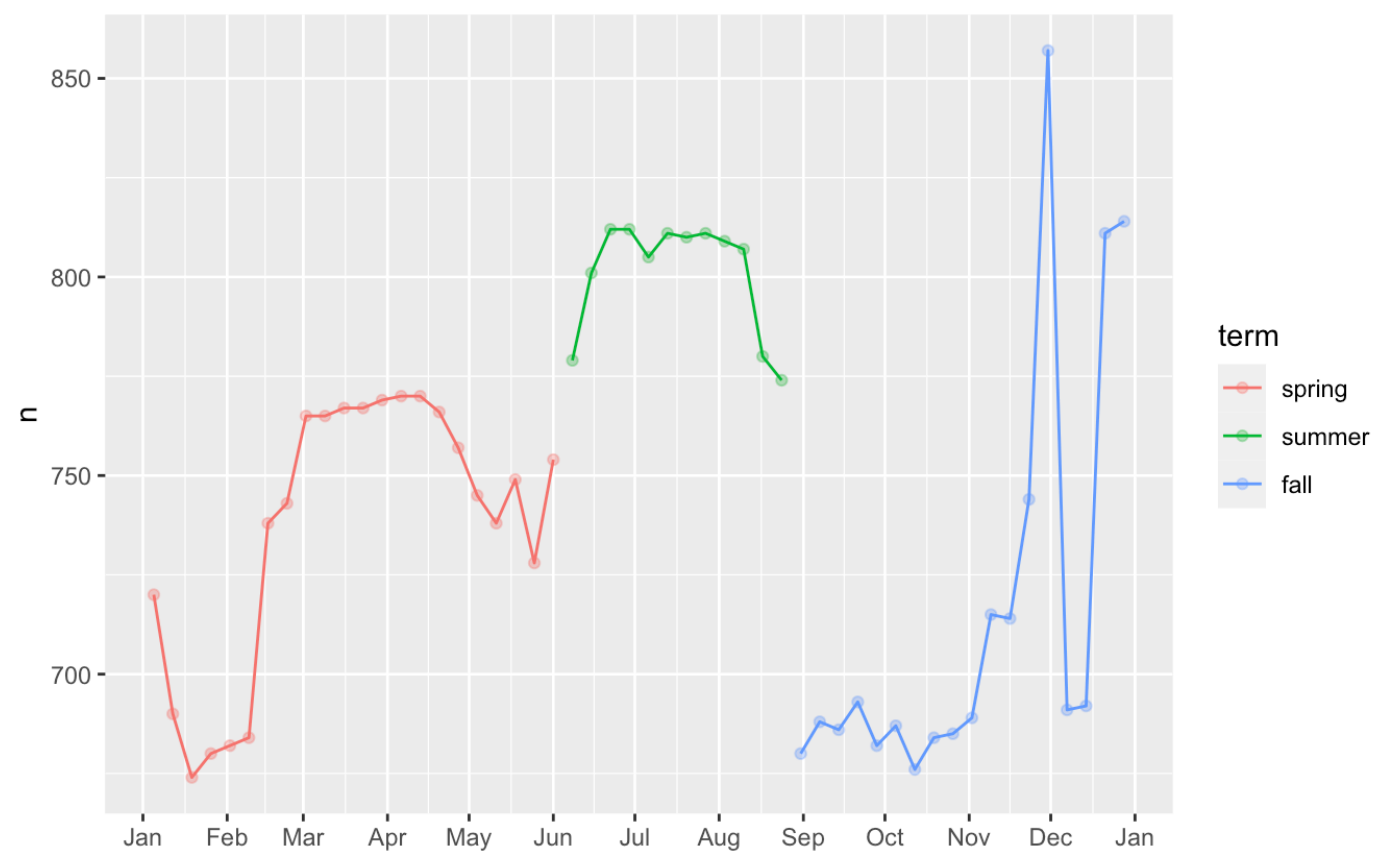
. . . especially when looking at weekend days



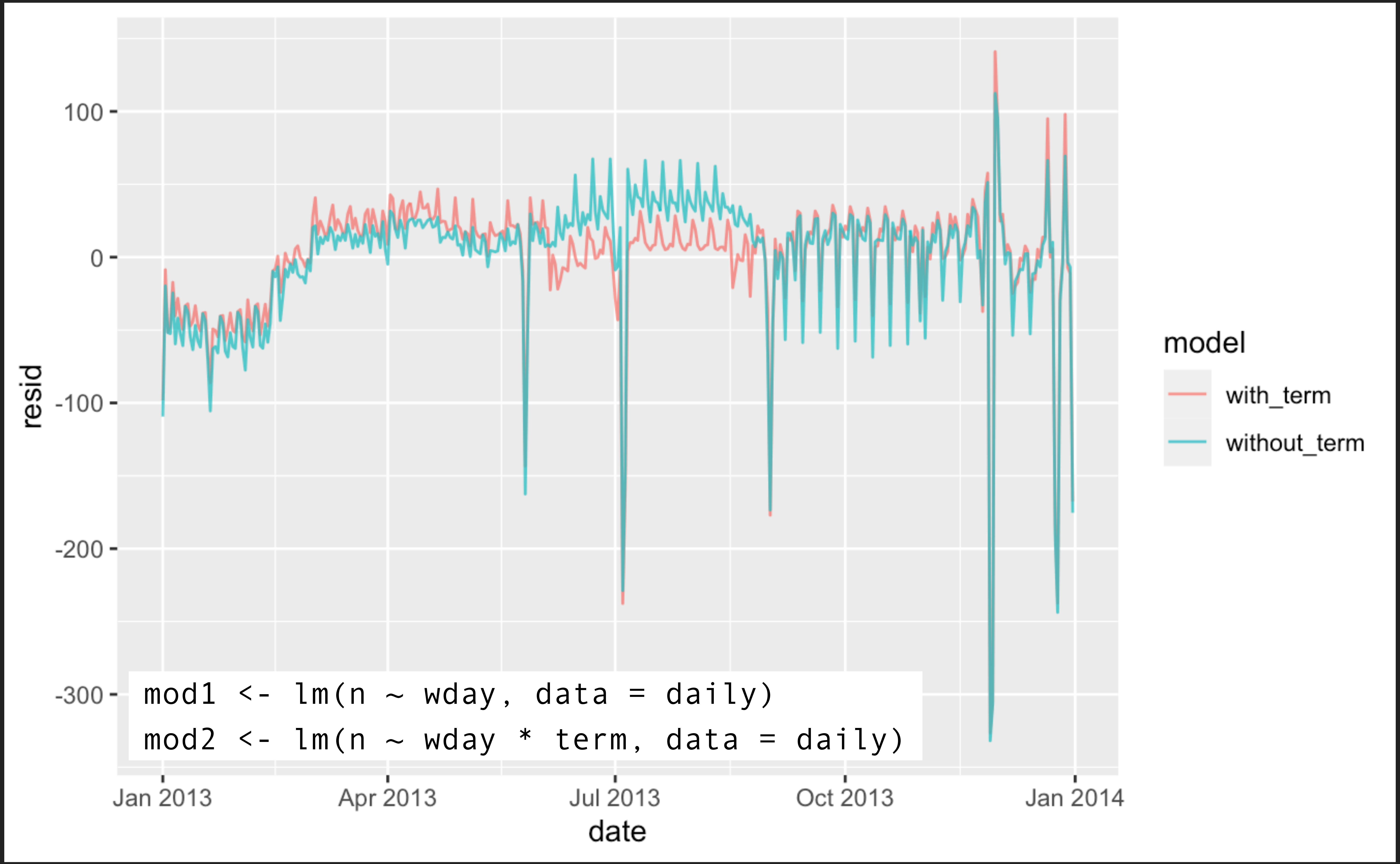
The model fails to accurately predict the number of flights on Saturday



Let's create a "term" variable that roughly captures the three school terms



Fitting a separate day of week effect for each term improves our model



```
mod1 <- lm(n ~ wday, data = daily)
mod2 <- lm(n ~ wday * term, data = daily)
```

Standardized Regression Coefficients

Example: Monthly earnings and years of education

This part of the tutorial is by James M. Murray, Ph.D. University of Wisconsin - La Crosse. https://murraylax.org/rtutorials/multregression_standardized.html (https://murraylax.org/rtutorials/multregression_standardized.html)

In this tutorial, we will focus on an example that explores the relationship between total monthly earnings (MonthlyEarnings) and a number of factors that may influence monthly earnings including including each person's IQ (IQ), a measure of knowledge of their job (Knowledge), years of education (YearsEdu), years experience (YearsExperience), and years at current job (Tenure).

The code below downloads a CSV file that includes data on the above variables from 1980 for 935 individuals, and assigns it to a dataset that we name wages.

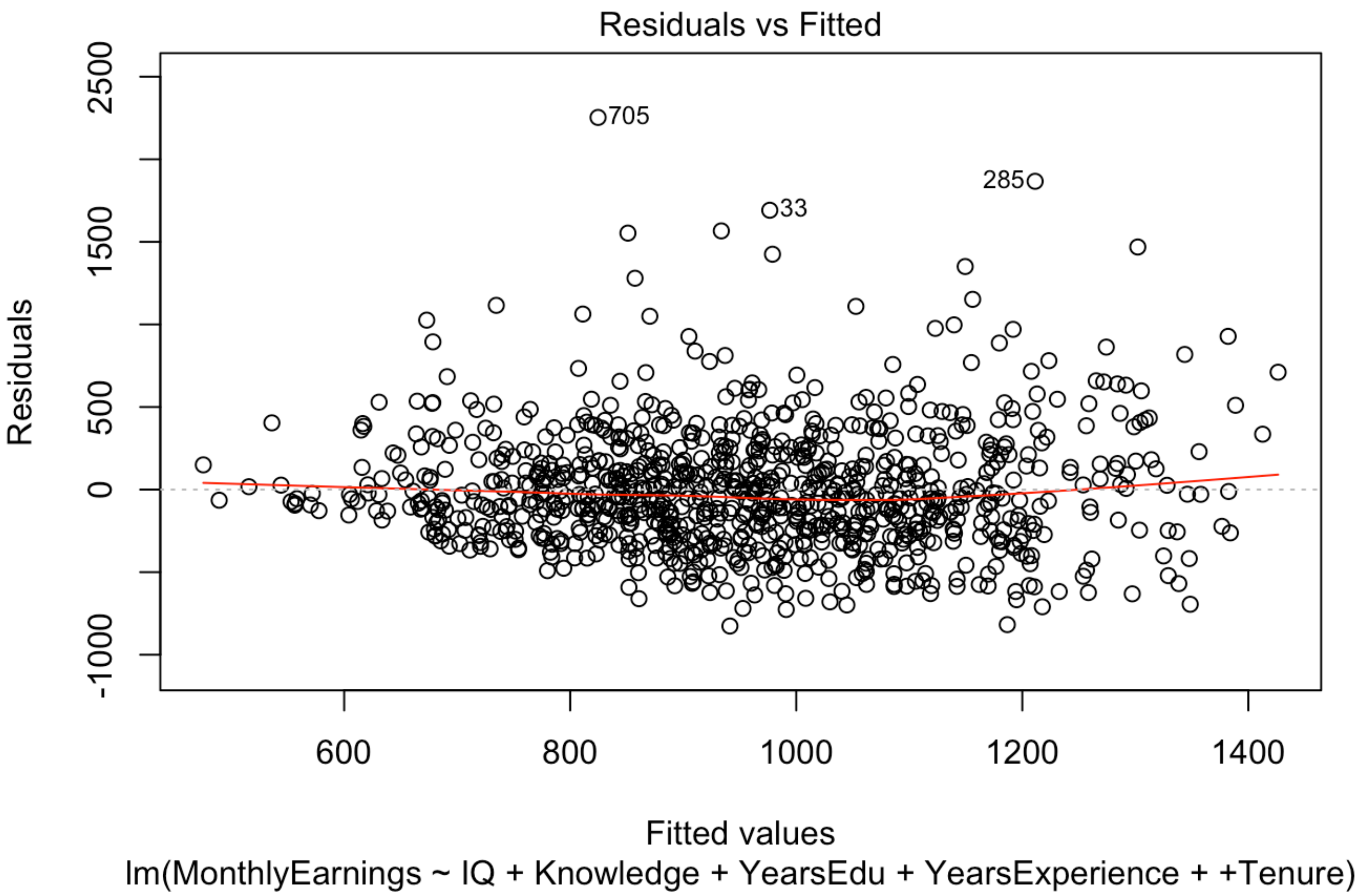
```
download.file( url="http://murraylax.org/datasets/wage2.csv", dest="wage2.csv")
wages <- read.csv("wage2.csv")
```

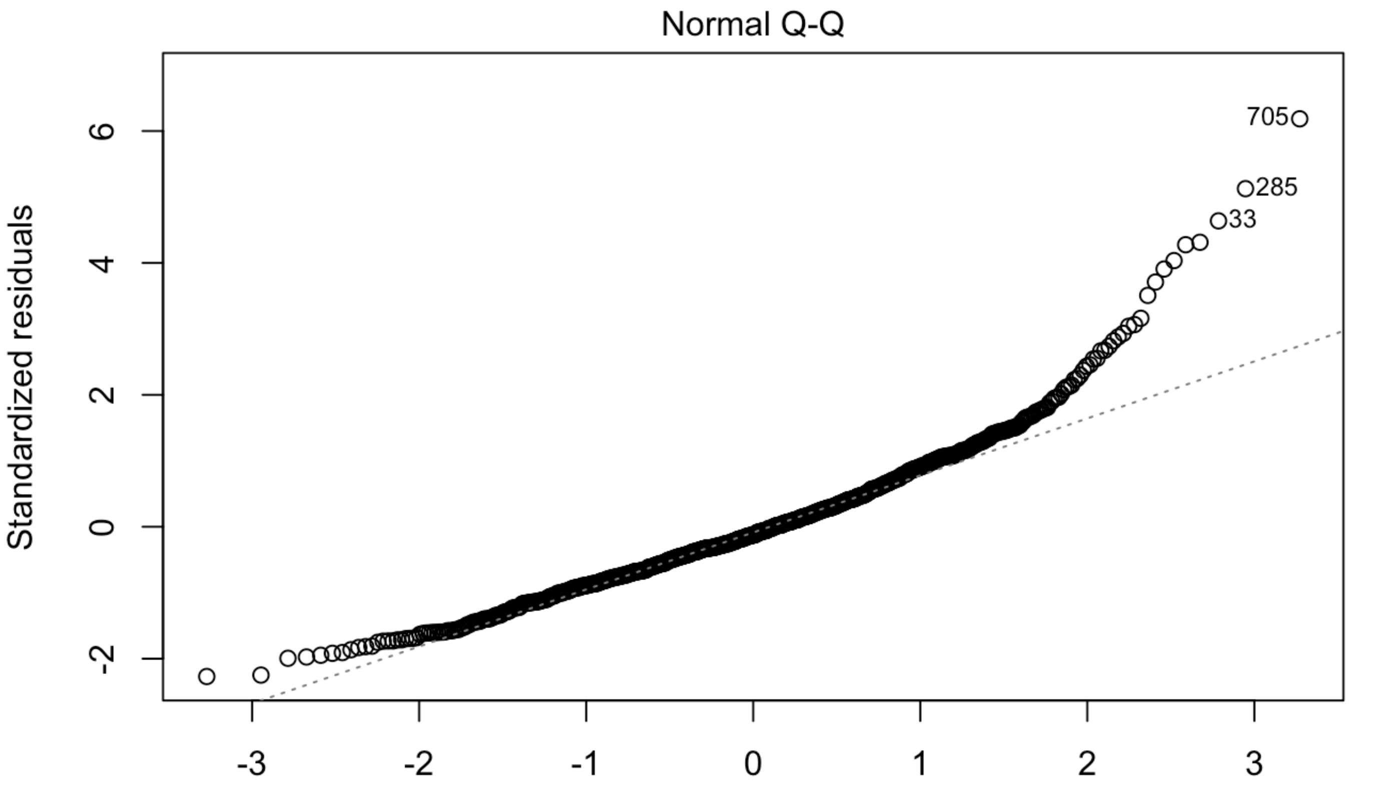
We will estimate the following multiple regression equation using the above five explanatory variables:

```
lmwages <- lm(MonthlyEarnings ~ IQ
                              + Knowledge
                              + YearsEdu
                              + YearsExperience +
                              + Tenure
                    , data = wages)
summary(lmwages)
```

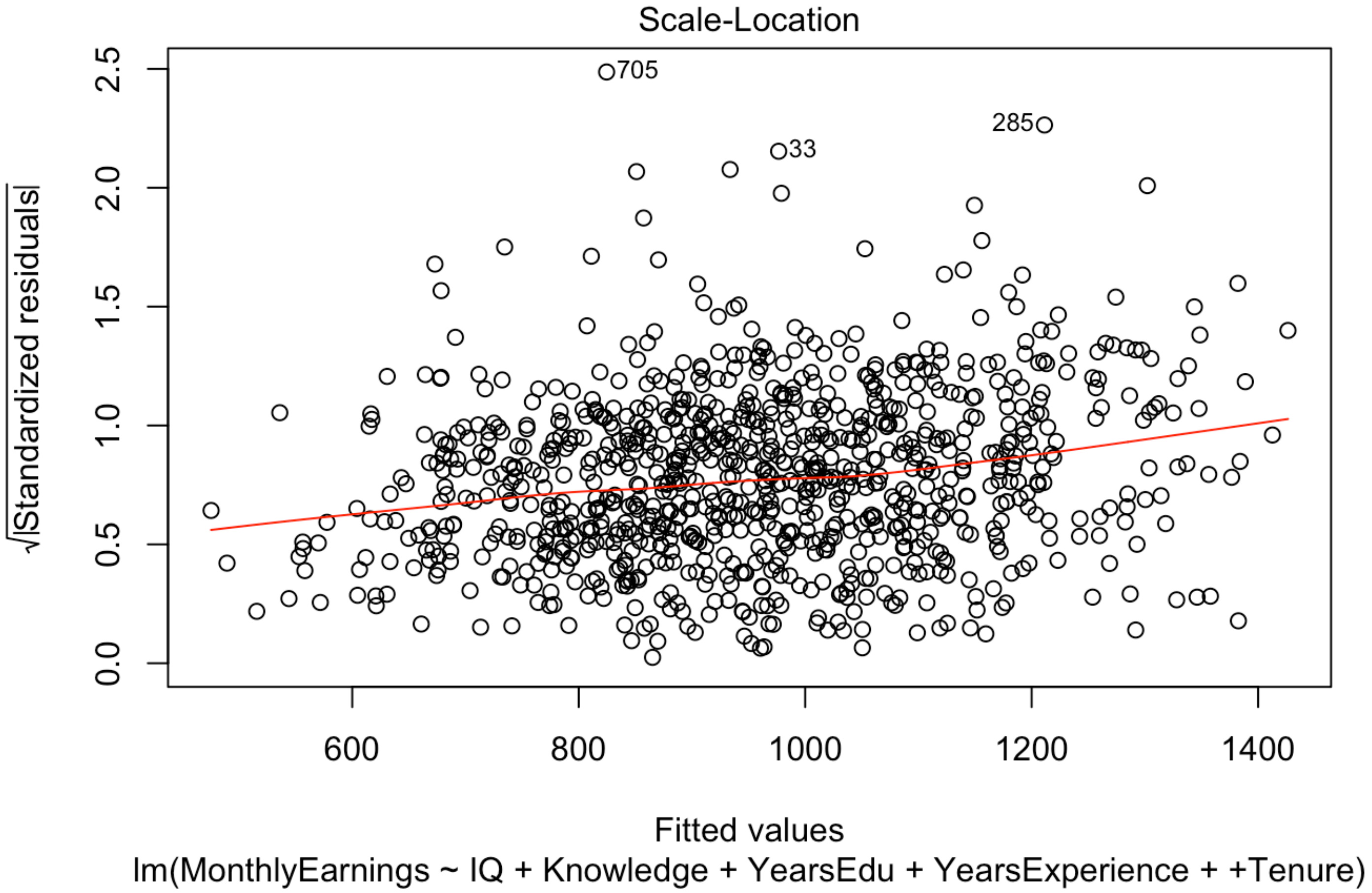
```
##
## Call:
## lm(formula = MonthlyEarnings ~ IQ + Knowledge + YearsEdu + YearsExperience +
##     +Tenure, data = wages)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -826.33 -243.85  -44.83  180.83 2253.35
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)     -531.0392   115.0513  -4.616 4.47e-06 ***
## IQ                 3.6966     0.9651   3.830 0.000137 ***
## Knowledge          8.2703     1.8273   4.526 6.79e-06 ***
## YearsEdu          47.2698     7.2980   6.477 1.51e-10 ***
## YearsExperience   11.8589     3.2494   3.650 0.000277 ***
## Tenure             6.2465     2.4565   2.543 0.011156 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 365.4 on 929 degrees of freedom
## Multiple R-squared:  0.1878, Adjusted R-squared:  0.1834
## F-statistic: 42.97 on 5 and 929 DF,  p-value: < 2.2e-16
```

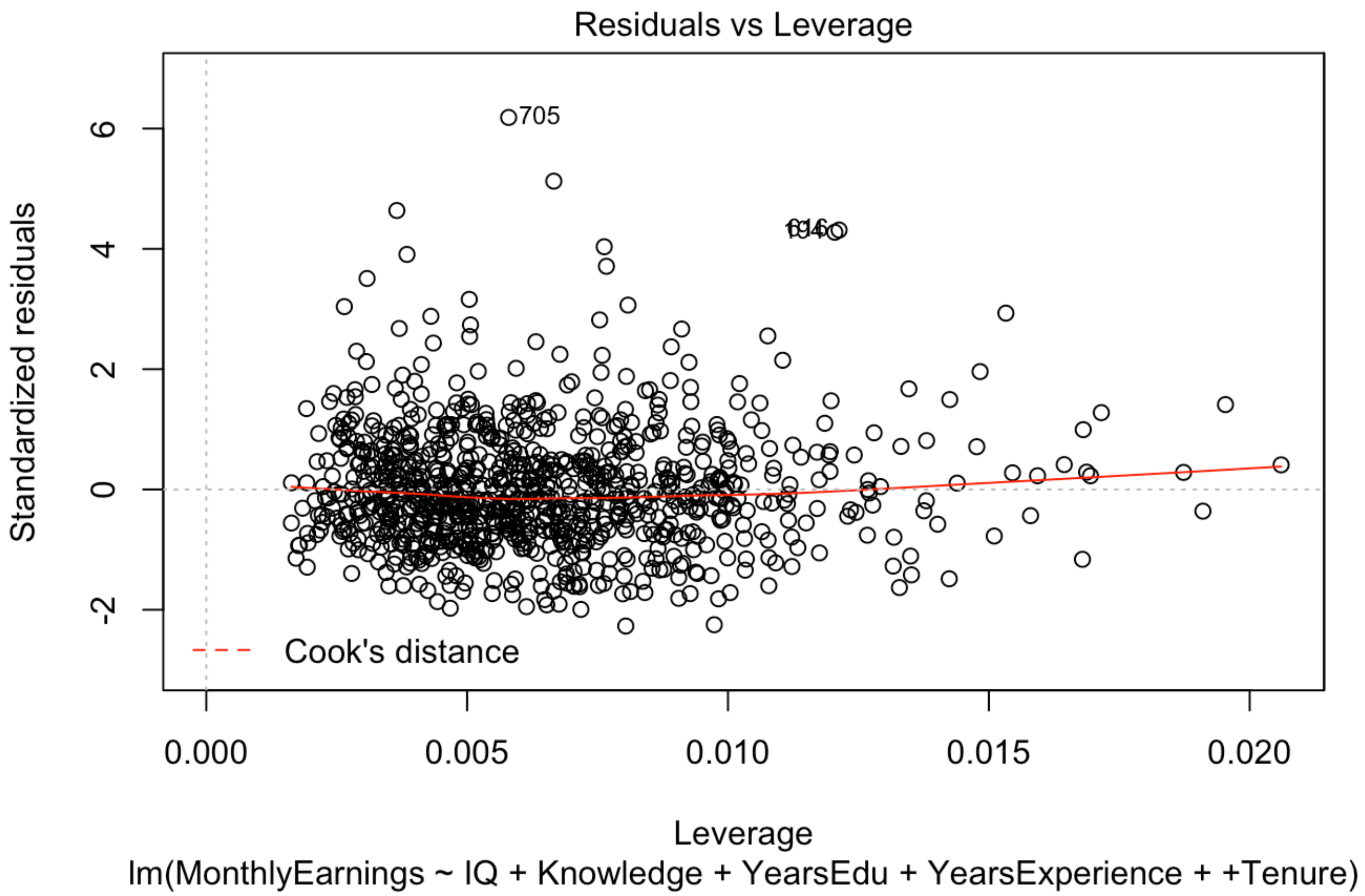
```
plot(lmwages)
```


Residuals vs Fitted
Residuals
2500
1500
500
0
-1000
705
285
33
600
800
1000
1200
1400
Fitted values
lm(MonthlyEarnings ~ IQ + Knowledge + YearsEdu + YearsExperience + +Tenure)

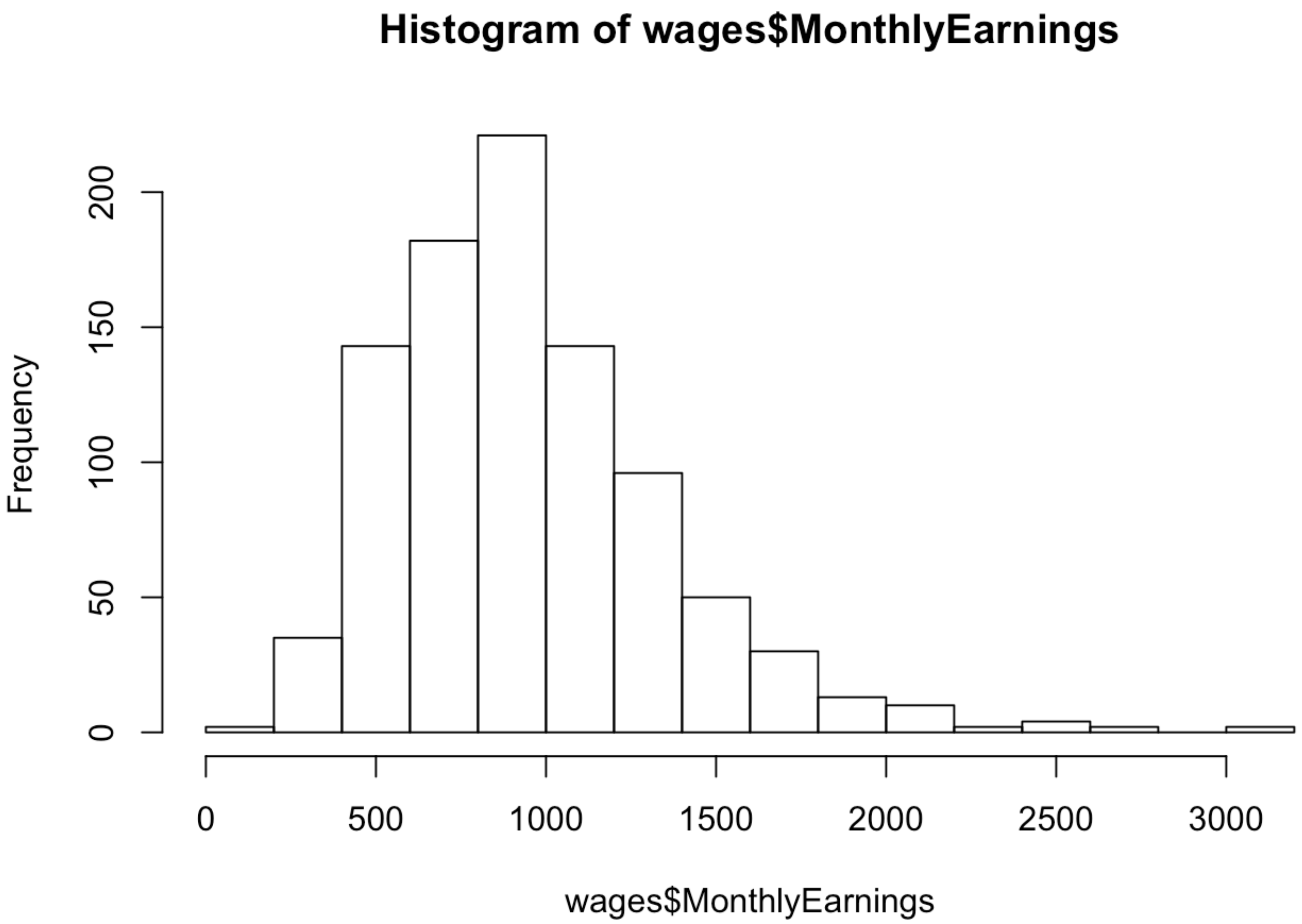

Normal Q-Q
Standardized residuals
6
4
2
0
-2
705
285
33
-3
-2
-1
0
1
2
3

Theoretical Quantiles
lm(MonthlyEarnings ~ IQ + Knowledge + YearsEdu + YearsExperience + +Tenure)





```
hist(wages$MonthlyEarnings)
```



Suppose we wanted to determine which of the following has a bigger impact on monthly earnings: an additional year of experience in your field (i.e. the YearsExperience variable) or an additional year of experience with your current employer (i.e. the Tenure variable). Each of these variables are measured in years and it does make sense to compare these two.

One additional year of education and one additional year of experience are very different things

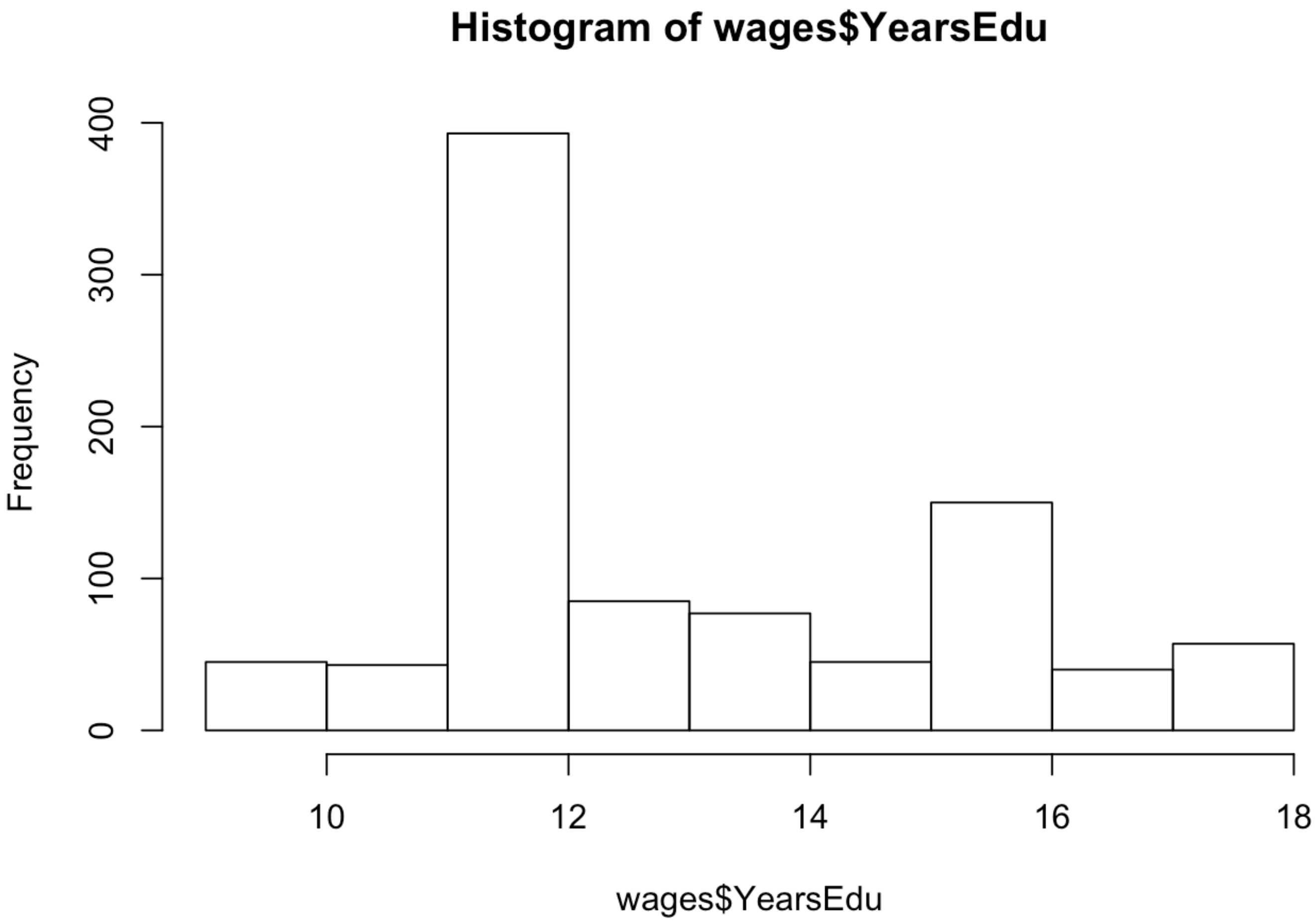
```
table(wages$YearsEdu)
```

```
## 
##   9  10  11  12  13  14  15  16  17  18 
##  10  35  43 393  85  77  45 150  40  57
```

```
summary(wages$YearsEdu)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
##    9.00   12.00   12.00   13.47   16.00   18.00
```

```
hist(wages$YearsEdu)
```



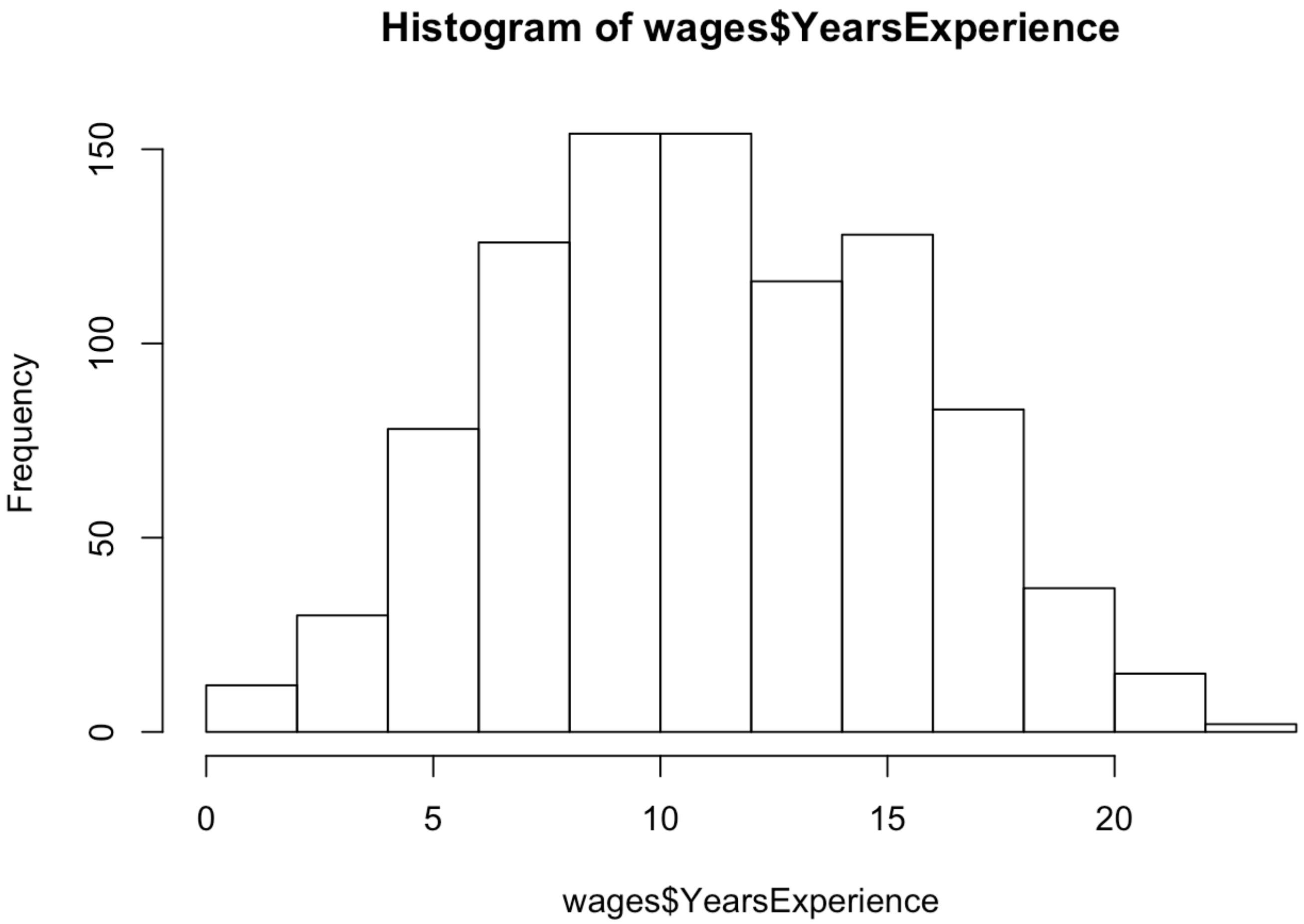
```
table(wages$YearsExperience)
```

```
## 
##  1  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 
## 12  1 29 30 48 54 72 82 72 89 65 62 54 60 68 53 30 23 14 12  3  2
```

```
summary(wages$YearsExperience)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
##    1.00    8.00   11.00   11.56   15.00   23.00
```

```
hist(wages$YearsExperience)
```



Consider the regression below with standardized values for YearsExperience and YearsEdu. Notice the calls to scale() in the regression formula.

```
lmwages <- lm(MonthlyEarnings ~ IQ
                              + Knowledge
                              + scale(YearsEdu)
                              + scale(YearsExperience) +
                              + Tenure
                    , data = wages)
summary(lmwages)
```

```
##
## Call:
## lm(formula = MonthlyEarnings ~ IQ + Knowledge + scale(YearsEdu) +
##     scale(YearsExperience) + +Tenure, data = wages)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -826.33 -243.85  -44.83  180.83 2253.35
##
## Coefficients:
##                         Estimate Std. Error t value Pr(>|t|)
## (Intercept)             242.7433   102.3149   2.373 0.017870 *
## IQ                        3.6966     0.9651   3.830 0.000137 ***
## Knowledge                 8.2703     1.8273   4.526 6.79e-06 ***
## scale(YearsEdu)         103.8353    16.0313   6.477 1.51e-10 ***
## scale(YearsExperience)  51.8778    14.2148   3.650 0.000277 ***
## Tenure                    6.2465     2.4565   2.543 0.011156 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 365.4 on 929 degrees of freedom
## Multiple R-squared:  0.1878, Adjusted R-squared:  0.1834
## F-statistic: 42.97 on 5 and 929 DF,  p-value: < 2.2e-16
```

The output shows that a one standard deviation increase in years of education (which happens to be an additional 2.2 years) leads to a return of $103.84 of additional monthly earnings. A one standard deviation increase in years of experience (which happens to be 4.4 years) leads to a return of $51.88. We can see that increasing education has approximately twice the impact on monthly earnings as increasing experience.

Compare these coefficients to the unscaled regression from Section 1 above. The unscaled regression coefficients were equal to 47.27 and 11.86 for years of education and years of experience, respectively. Failing to standardize the explanatory variables would lead to an incorrect conclusion that education is approximately four times more valuable than experience.

Compare the remaining coefficients. You can see that all other coefficients, standard errors, and all p-values are identical. Linearly scaling a variable in the regression model does not change the results for other variables.

Suppose we wanted to compare how education versus workplace knowledge affects monthly earnings. Education is measured in years, and knowledge is a workplace intelligence test score. These scales are not comparable.

Still, we can standardize each variable. Consider the following regression:

```
lmwages <- lm(MonthlyEarnings ~ IQ
                             + scale(Knowledge)
                             + scale(YearsEdu)
                             + YearsExperience +
                             + Tenure
                    , data = wages)
summary(lmwages)
```

```
##
## Call:
## lm(formula = MonthlyEarnings ~ IQ + scale(Knowledge) + scale(YearsEdu) +
##     YearsExperience + +Tenure, data = wages)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -826.33 -243.85  -44.83  180.83 2253.35
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)      401.2281   106.9480   3.752 0.000187 ***
## IQ                 3.6966     0.9651   3.830 0.000137 ***
## scale(Knowledge)  63.1751    13.9583   4.526 6.79e-06 ***
## scale(YearsEdu)  103.8353    16.0313   6.477 1.51e-10 ***
## YearsExperience   11.8589     3.2494   3.650 0.000277 ***
## Tenure             6.2465     2.4565   2.543 0.011156 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 365.4 on 929 degrees of freedom
## Multiple R-squared:  0.1878, Adjusted R-squared:  0.1834
## F-statistic: 42.97 on 5 and 929 DF,  p-value: < 2.2e-16
```

The regression output reveals that a one standard deviation increase in knowledge of work leads to an increase in monthly earnings equal to $63.18. A one standard deviation increase in education leads to an increase in monthly earnings equal to $103.84. We can conclude that education is relatively more valuable than knowledge of work in terms of increasing monthly earnings.

Let's also check for multicollinearity using the variance inflation factor:

```
vif(lmwages)
```

```
##               IQ scale(Knowledge)  scale(YearsEdu)  YearsExperience
##         1.476318         1.362976         1.797887         1.413546
##           Tenure
##         1.087340
```

```
cor.test(wages$Tenure, wages$YearsExperience)
```

```
##
##  Pearson's product-moment correlation
##
## data:  wages$Tenure and wages$YearsExperience
## t = 7.6737, df = 933, p-value = 4.202e-14
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  0.1823909 0.3030333
## sample estimates:
##       cor
## 0.2436544
```

Credits

- Graphics: Dave DiCello photography (cover)
- Bruce, P., Bruce, A., & Gedeck, P. (2020). Practical Statistics for Data Scientists: 50+ Essential Concepts Using R and Python. O'Reilly Media.
- Goodman, S. (2008). A dirty dozen: Twelve p-value misconceptions. In Seminars in Hematology (Vol. 45, No. 3, pp. 135-140). WB Saunders.
- James, G., Witten, D., Hastie, T., & Tibshirani, R. (2013). An introduction to statistical learning (Vol. 112, p. 18). New York: springer.
- Grolemund, G., & Wickham, H. (2018). R for data science.