Making predictions

INTRODUCTION TO REGRESSION IN R

Richie Cotton
Curriculum Architect at DataCamp

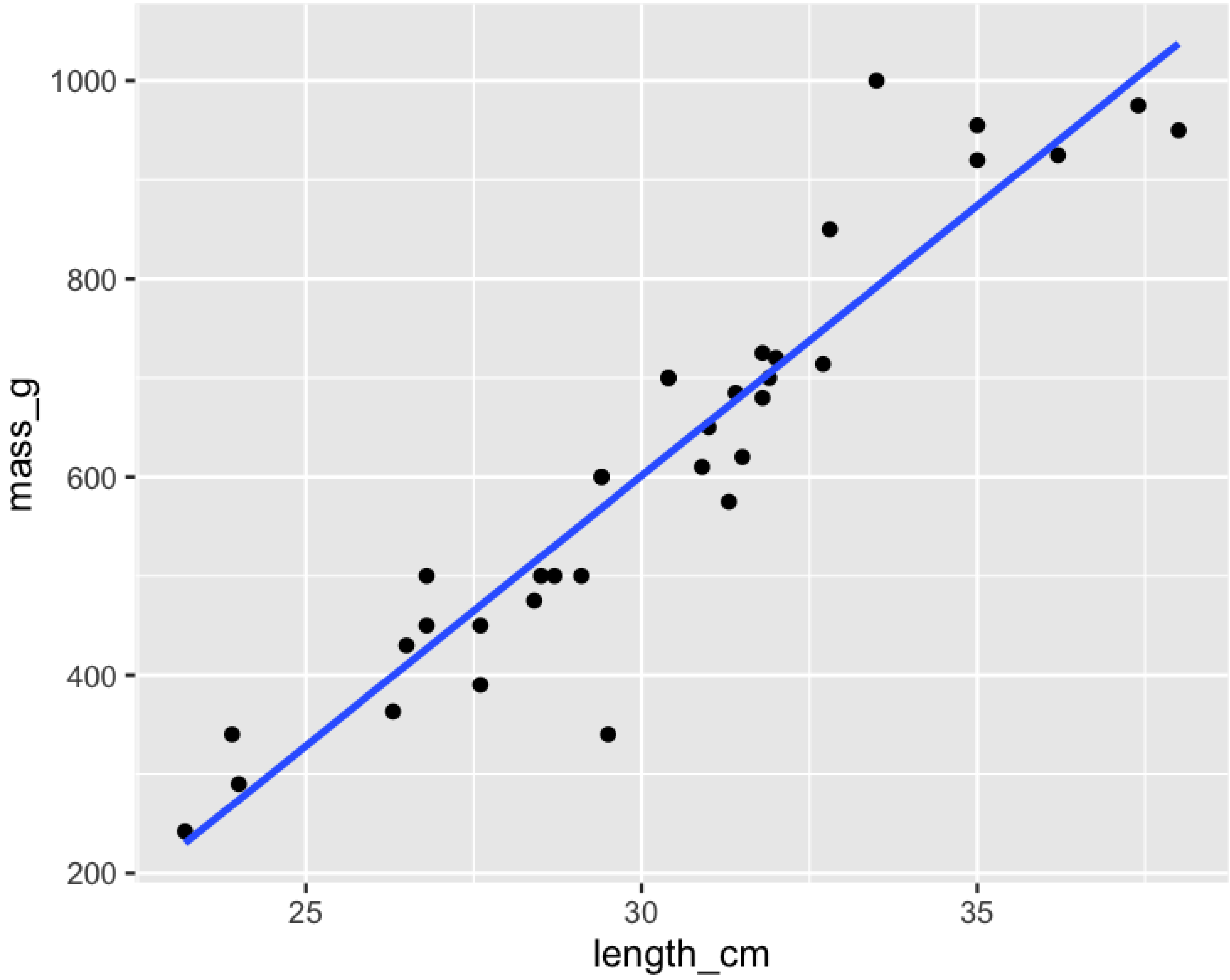
The fish dataset: bream

```r
bream <- fish %>%
  filter(species == "Bream")
```

species	length_cm	mass_g
Bream	23.2	242
Bream	24.0	290
Bream	23.9	340
Bream	26.3	363
Bream	26.5	430
...	...	...

Plotting mass vs. length

```
ggplot(bream, aes(length_cm, mass_g)) +
  geom_point() +
  geom_smooth(method = "lm", se = FALSE)
```



Running the model

```
mdl_mass_vs_length <- lm(mass_g ~ length_cm, data = bream)
```

```
Call:
lm(formula = mass_g ~ length_cm, data = bream)

Coefficients:
(Intercept)    length_cm
   -1035.35        54.55
```

Data on explanatory values to predict

> If I set the explanatory variables to these values,
> what value would the response variable have?

```
library(dplyr)
explanatory_data <- tibble(length_cm = 20:40)
```

Call predict()

```r
library(tibble)
explanatory_data <- tibble(length_cm = 20:40)
```

```r
predict(mdl_mass_vs_length, explanatory_data)
```

```
         1          2          3          4          5          6
  55.65205  110.20203  164.75202  219.30200  273.85198  328.40196
         7          8          9         10         11         12
 382.95194  437.50192  492.05190  546.60188  601.15186  655.70184
        13         14         15         16         17         18
 710.25182  764.80181  819.35179  873.90177  928.45175  983.00173
        19         20         21
1037.55171 1092.10169 1146.65167
```

Predicting inside a data frame

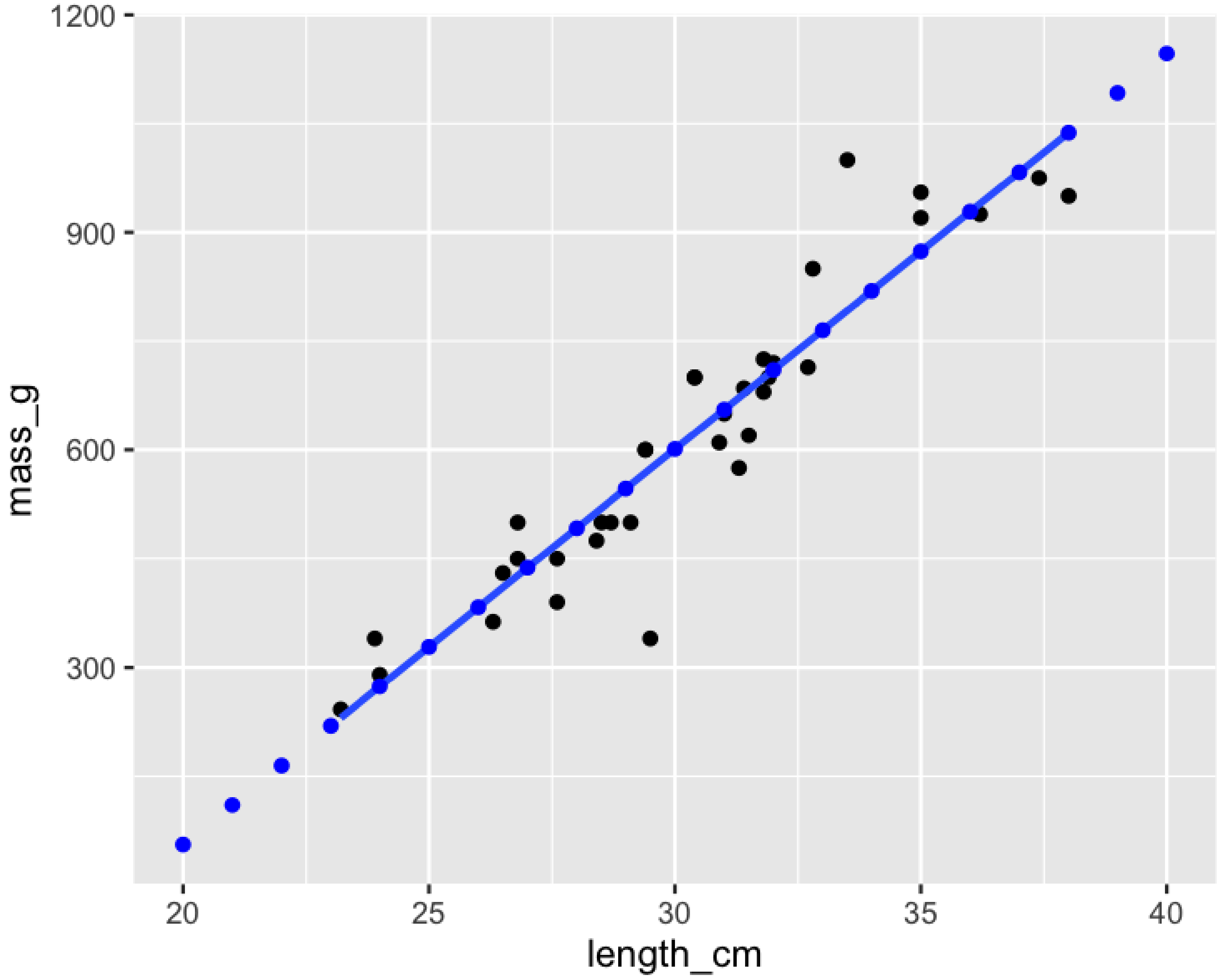
```r
library(dplyr)
explanatory_data <- tibble(length_cm = 20:40)
```

```r
prediction_data <- explanatory_data %>%
  mutate(
    mass_g = predict(
      mdl_mass_vs_length, explanatory_data
    )
  )
```

```
# A tibble: 21 x 2
   length_cm mass_g
       <int>  <dbl>
 1        20   55.7
 2        21  110.
 3        22  165.
 4        23  219.
 5        24  274.
 6        25  328.
 7        26  383.
 8        27  438.
 9        28  492.
10        29  547.
# ... with 11 more rows
```

Showing predictions

```r
ggplot(bream, aes(length_cm, mass_g)) +
  geom_point() +
  geom_smooth(method = "lm", se = FALSE) +
  geom_point(
    data = prediction_data,
    color = "blue"
  )
```

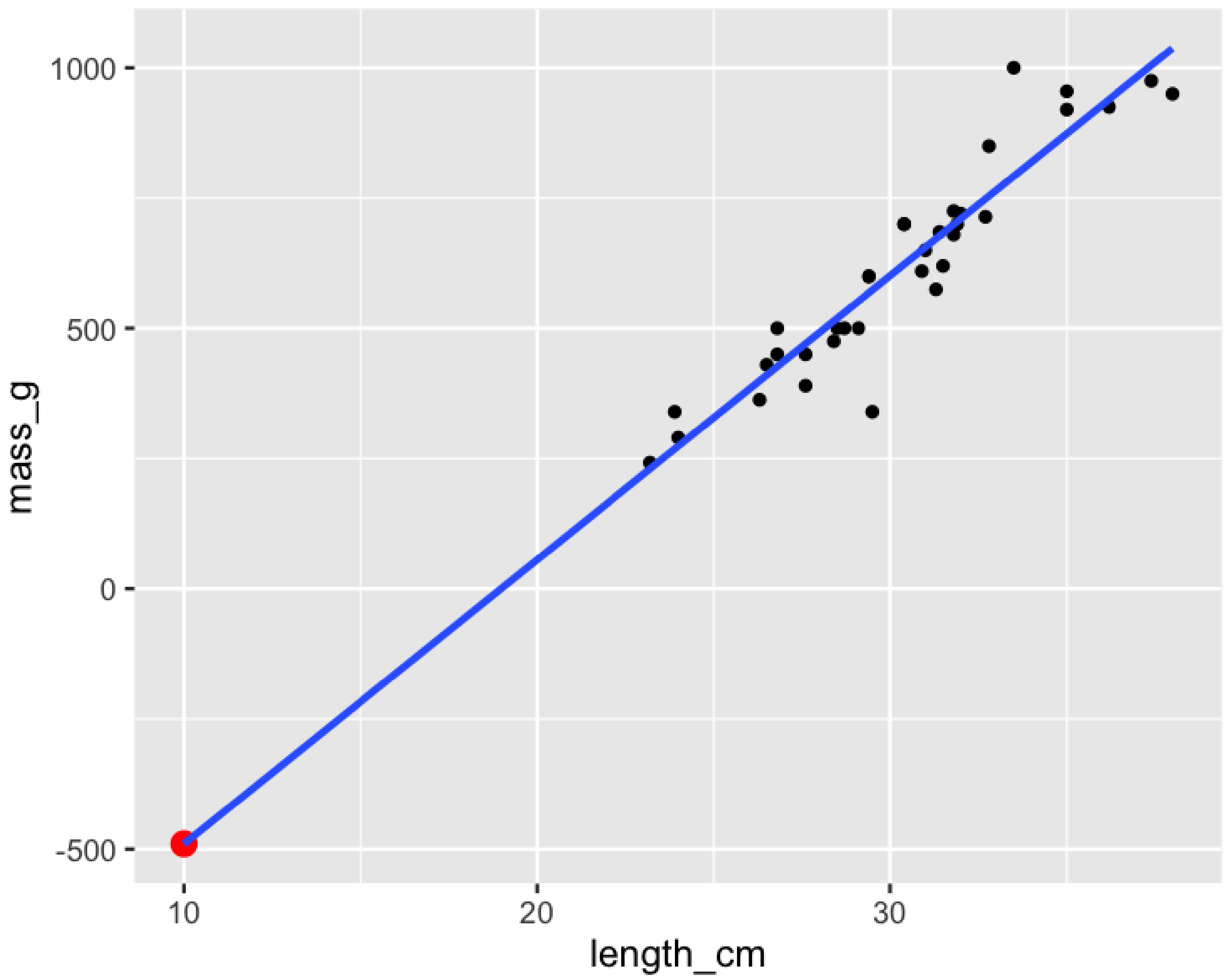


Extrapolating

Extrapolating means making predictions outside the range of observed data.

```
explanatory_little_bream <- tibble(length_cm = 10)
explanatory_little_bream %>%
  mutate(
    mass_g = predict(
      mdl_mass_vs_length, explanatory_little_bream
    )
  )
```

```
# A tibble: 1 x 2
  length_cm mass_g
      <dbl>  <dbl>
1        10  -490.
```



Let's practice!

INTRODUCTION TO REGRESSION IN R

Working with model objects

INTRODUCTION TO REGRESSION IN R

Richie Cotton
Curriculum Architect at DataCamp

coefficients()

```
mdl_mass_vs_length <- lm(mass_g ~ length_cm, data = bream)
```

```
Call:
lm(formula = mass_g ~ length_cm, data = bream)

Coefficients:
(Intercept)    length_cm
   -1035.35        54.55
```

```
coefficients(mdl_mass_vs_length)
```

```
(Intercept)   length_cm
-1035.34757    54.54998
```

fitted()

fitted values: predictions on the original dataset

```
fitted(mdl_mass_vs_length)
```

or equivalently

```
explanatory_data <- bream %>%
  select(length_cm)

predict(mdl_mass_vs_length, explanatory_data)
```

```
       1        2        3        4
230.2120 273.8520 268.3970 399.3169 410.226
       6        7        8        9        1
426.5919 426.5919 470.2319 470.2319 519.326
      11       12       13       14        1
513.8719 530.2369 552.0569 573.8769 568.421
      16       17       18       19        2
568.4219 622.9719 622.9719 650.2468 655.701
      21       22       23       24        2
672.0668 677.5218 682.9768 699.3418 704.796
      26       27       28       29        3
699.3418 710.2518 748.4368 753.8918 792.076
      31       32       33       34        3
873.9018 873.9018 939.3617 1004.8217 1037.551
```

residuals()

Residuals: actual response values minus predicted response values

```
residuals(mdl_mass_vs_length)
```

or equivalently

```
bream$mass_g - fitted(mdl_mass_vs_length)
```

```
      1        2        3        4        5 
 11.788   16.148   71.603  -36.317   19.773 
      6        7        8        9       10 
 23.408   73.408  -80.232  -20.232  -19.327 
     11       12       13       14       15 
-38.872  -30.237  -52.057 -233.877   31.578 
     16       17       18       19       20 
 31.578   77.028   77.028  -40.247   -5.702 
     21       22       23       24       25 
-97.067    7.478  -62.977  -19.342   -4.797 
     26       27       28       29       30 
 25.658    9.748  -34.437   96.108  207.923 
     31       32       33       34       35 
 46.098   81.098  -14.362  -29.822  -87.552 
```

summary()

```
summary(mdl_mass_vs_length)
```

```
Call:
lm(formula = mass_g ~ length_cm, data = bream)

Residuals:
   Min     1Q Median     3Q    Max
-233.9  -35.4   -4.8   31.6  207.9

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) -1035.35     107.97   -9.59  4.6e-11 ***
length_cm      54.55       3.54   15.42  < 2e-16 ***

Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 74.2 on 33 degrees of freedom
Multiple R-squared:  0.878,    Adjusted R-squared:  0.874
F-statistic:  238 on 1 and 33 DF,  p-value: <2e-16
```

summary(): call

```
Call:
lm(formula = mass_g ~ length_cm, data = bream)
```

summary(): residuals

```
Residuals:
   Min      1Q  Median      3Q     Max
-233.9  -35.4   -4.8    31.6   207.9
```

summary(): coefficients

```
Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) -1035.35     107.97   -9.59  4.6e-11 ***
length_cm      54.55       3.54   15.42  < 2e-16 ***

Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

summary(): model metrics

```
Residual standard error: 74.2 on 33 degrees of freedom
Multiple R-squared:  0.878,    Adjusted R-squared:  0.874
F-statistic:  238 on 1 and 33 DF,  p-value: <2e-16
```

tidy()

```r
library(broom)

tidy(mdl_mass_vs_length)
```

```
# A tibble: 2 x 5
  term        estimate std.error statistic  p.value
  <chr>          <dbl>     <dbl>     <dbl>    <dbl>
1 (Intercept)  -1035.     108.       -9.59 4.58e-11
2 length_cm       54.5      3.54     15.4  1.22e-16
```

augment()

```
augment(mdl_mass_vs_length)
```

```
# A tibble: 35 x 9
   mass_g length_cm .fitted .se.fit .resid    .hat .sigma .cooksd .std.resid
    <dbl>     <dbl>   <dbl>   <dbl>  <dbl>   <dbl>  <dbl>   <dbl>      <dbl>
 1    242      23.2    230.    28.1   11.8 0.144     75.3 0.00247      0.172
 2    290      24      274.    25.6   16.1 0.119     75.2 0.00364      0.232
 3    340      23.9    268.    25.9   71.6 0.122     74.1 0.0738       1.03
 4    363      26.3    399.    18.9  -36.3 0.0651    75.0 0.00894     -0.507
 5    430      26.5    410.    18.4   19.8 0.0616    75.2 0.00248      0.275
 6    450      26.8    427.    17.6   23.4 0.0566    75.2 0.00317      0.325
 7    500      26.8    427.    17.6   73.4 0.0566    74.1 0.0311       1.02
 8    390      27.6    470.    15.8  -80.2 0.0452    73.9 0.0291      -1.11
 9    450      27.6    470.    15.8  -20.2 0.0452    75.2 0.00185     -0.279
10    500      28.5    519.    14.1  -19.3 0.0360    75.2 0.00132     -0.265
# ... with 25 more rows
```

glance()

```
glance(mdl_mass_vs_length)
```

```
# A tibble: 1 x 11
  r.squared adj.r.squared sigma statistic  p.value    df logLik   AIC   BIC deviance df.residual
      <dbl>         <dbl> <dbl>     <dbl>    <dbl> <int>  <dbl> <dbl> <dbl>    <dbl>       <int>
1     0.878         0.874  74.2      238. 1.22e-16     2  -199.  405.  409.  181452.          33
```

Let's practice!

Regression to the mean

INTRODUCTION TO REGRESSION IN R

Richie Cotton
Curriculum Architect

The concept

- Response value = fitted value + residual
- "The stuff you explained" + "the stuff you couldn't explain"
- Residuals exist due to problems in the model *and* fundamental randomness
- Extreme cases are often due to randomness
- *Regression to the mean* means extreme cases don't persist over time

Pearson's father son dataset

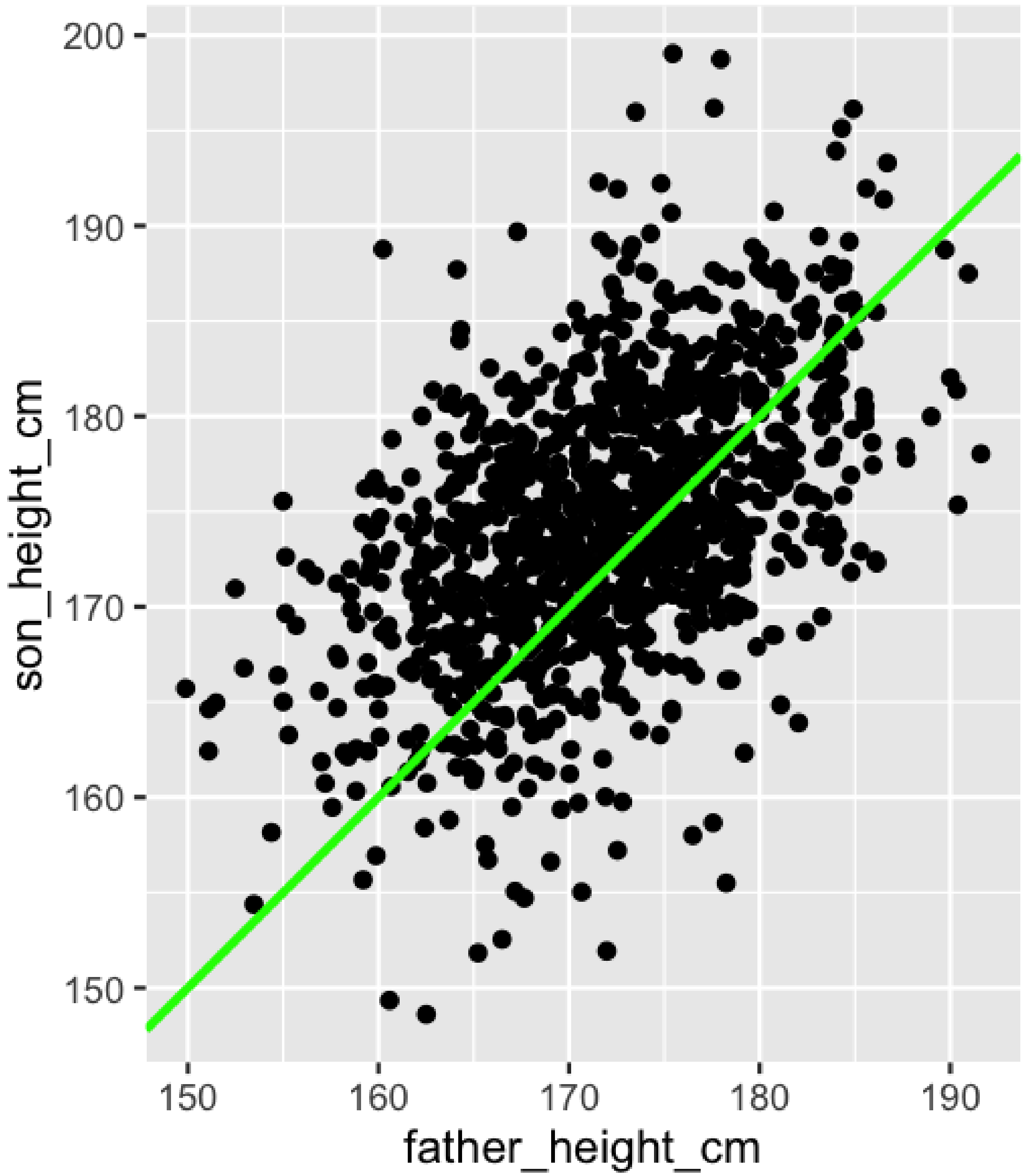
- 1078 father/son pairs
- Do tall fathers have tall sons?

father_height_cm	son_height_cm
165.2	151.8
160.7	160.6
165.0	160.9
167.0	159.5
155.3	163.3
...	...

[1] Adapted from https://www.rdocumentation.org/packages/UsingR/topics/father.son

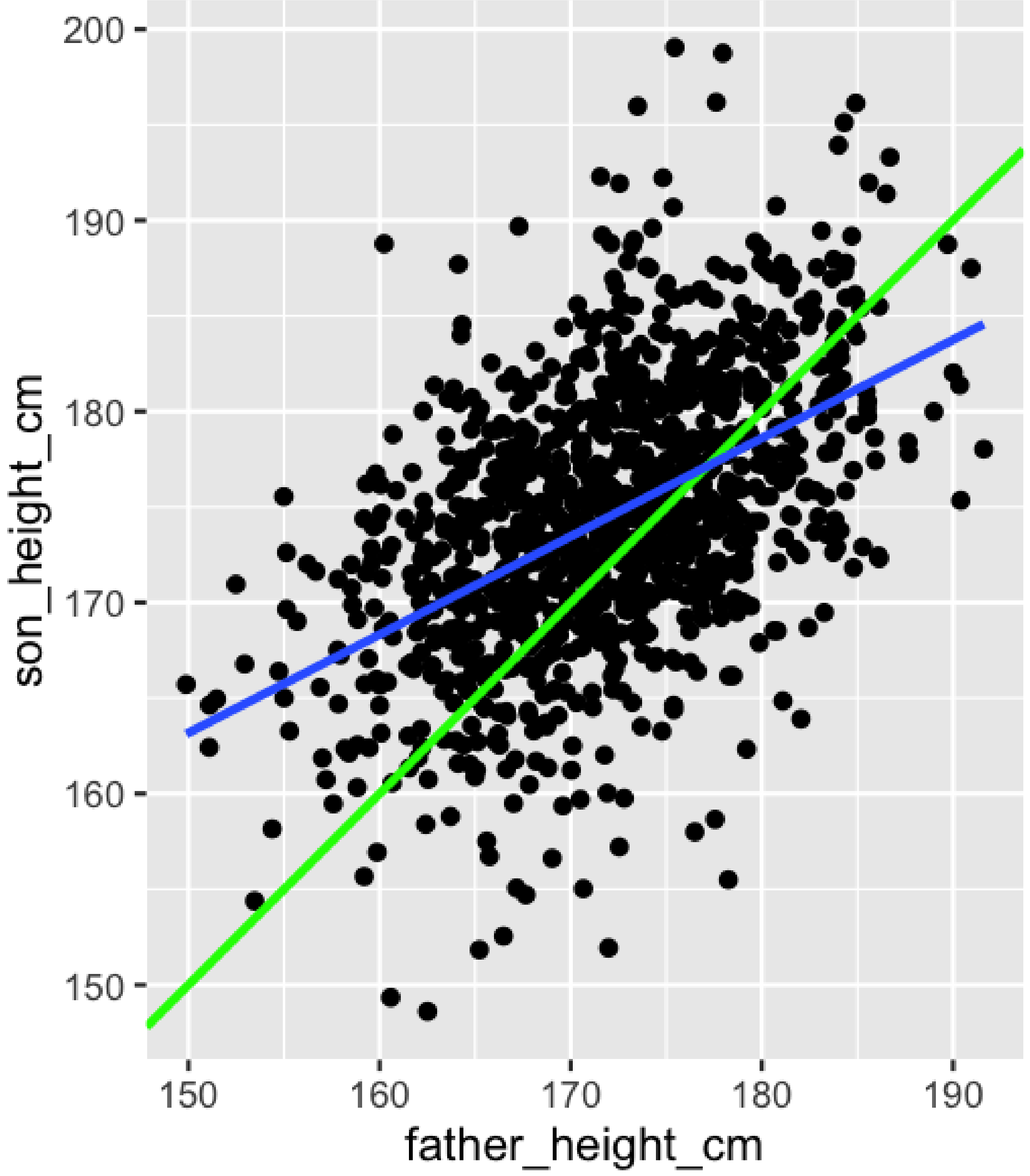
Scatter plot

```r
plt_son_vs_father <- ggplot(
  father_son,
  aes(father_height_cm, son_height_cm)
) +
  geom_point() +
  geom_abline(color = "green", size = 1) +
  coord_fixed()
```



Adding a regression line

```
plt_son_vs_father +
  geom_smooth(method = "lm", se = FALSE)
```



Running a regression

```
mdl_son_vs_father <- lm(
  son_height_cm ~ father_height_cm,
  data = father_son
)
```

```
Call:
lm(formula = son_height_cm ~ father_height_cm, data = father_son)

Coefficients:
     (Intercept)  father_height_cm
          86.072             0.514
```

Making predictions

```
really_tall_father <- tibble(
  father_height_cm = 190
)
predict(mdl_son_vs_father, really_tall_father)
```

```
183.7
```

```
really_short_father <- tibble(
  father_height_cm = 150
)
predict(mdl_son_vs_father, really_short_father)
```

```
163.2
```

Let's practice!

INTRODUCTION TO REGRESSION IN R

Transforming variables

INTRODUCTION TO REGRESSION IN R

Richie Cotton
Curriculum Architect at DataCamp

Perch dataset

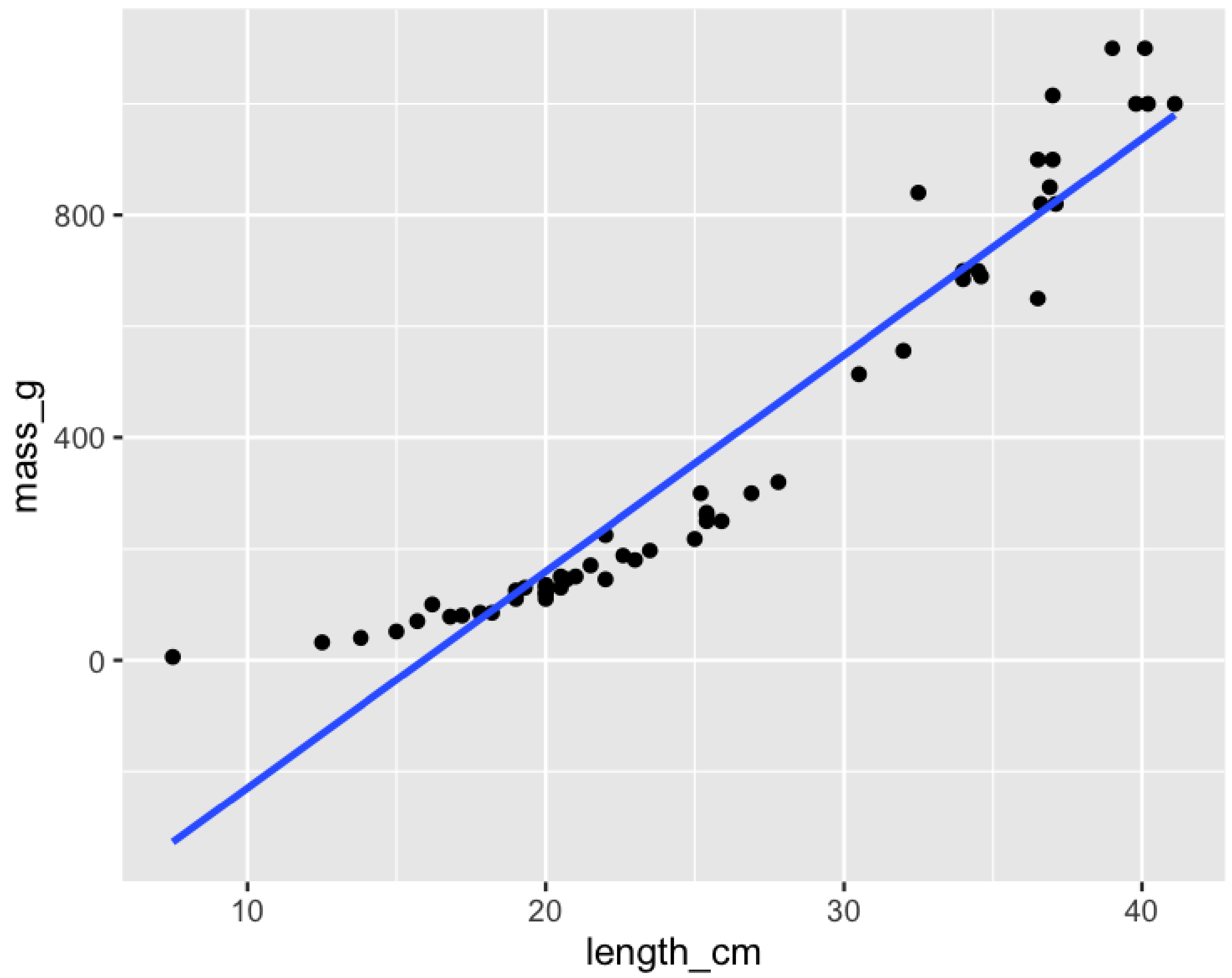
```r
library(dplyr)

perch <- fish %>%
  filter(species == "Perch")
```

species	mass_g	length_cm
Perch	5.9	7.5
Perch	32.0	12.5
Perch	40.0	13.8
Perch	51.5	15.0
Perch	70.0	15.7
…	…	…

It's not a linear relationship

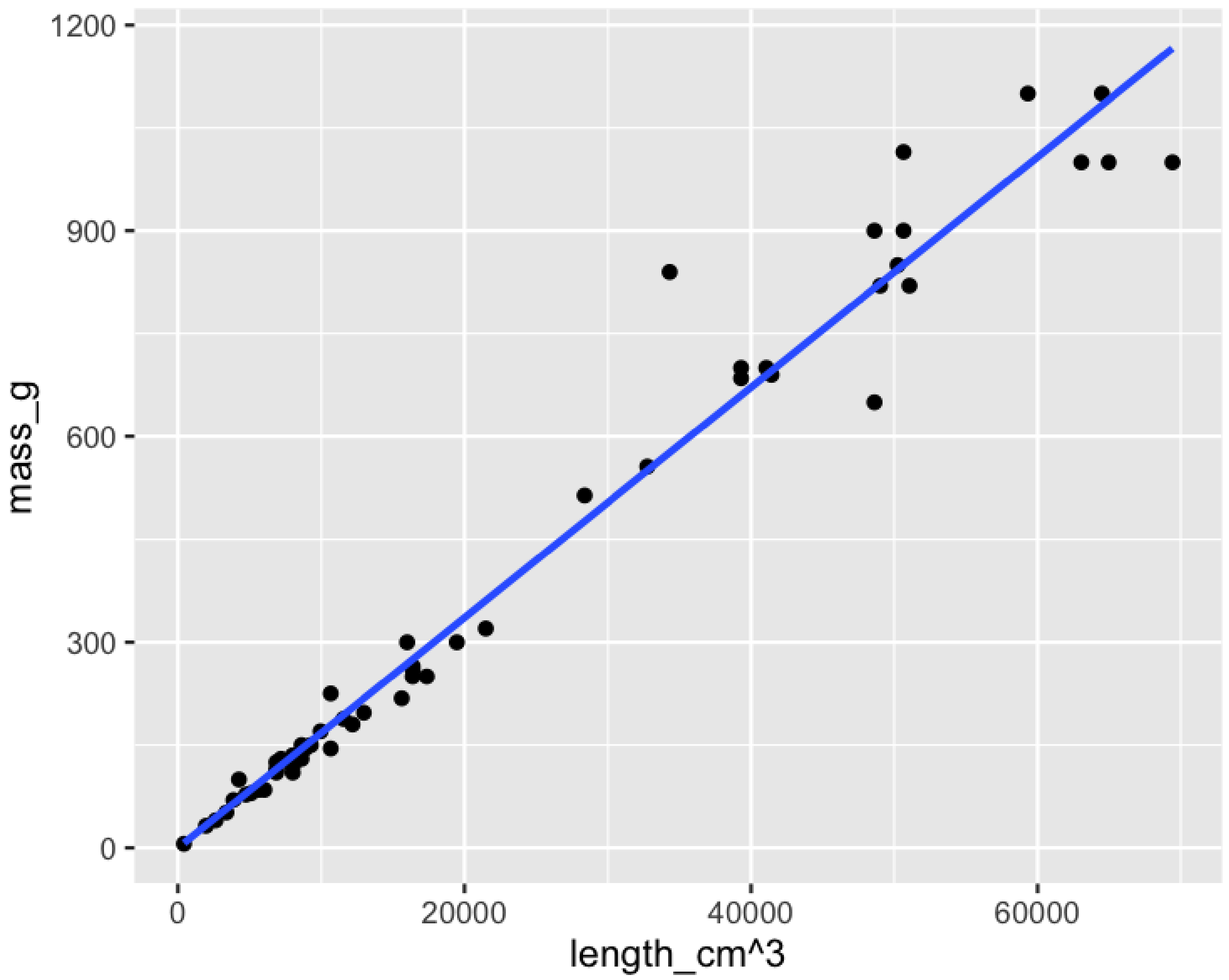
```r
ggplot(perch, aes(length_cm, mass_g)) +
  geom_point() +
  geom_smooth(method = "lm", se = FALSE)
```



Bream vs. perch

Plotting mass vs. length cubed

```
ggplot(perch, aes(length_cm ^ 3, mass_g)) +
  geom_point() +
  geom_smooth(method = "lm", se = FALSE)
```



Modeling mass vs. length cubed

```
mdl_perch <- lm(mass_g ~ I(length_cm ^ 3), data = perch)
```

```
Call:
lm(formula = mass_g ~ I(length_cm^3), data = perch)

Coefficients:
   (Intercept)  I(length_cm^3)
       -0.1175          0.0168
```

Predicting mass vs. length cubed

```r
explanatory_data <- tibble(
  length_cm = seq(10, 40, 5)
)
```

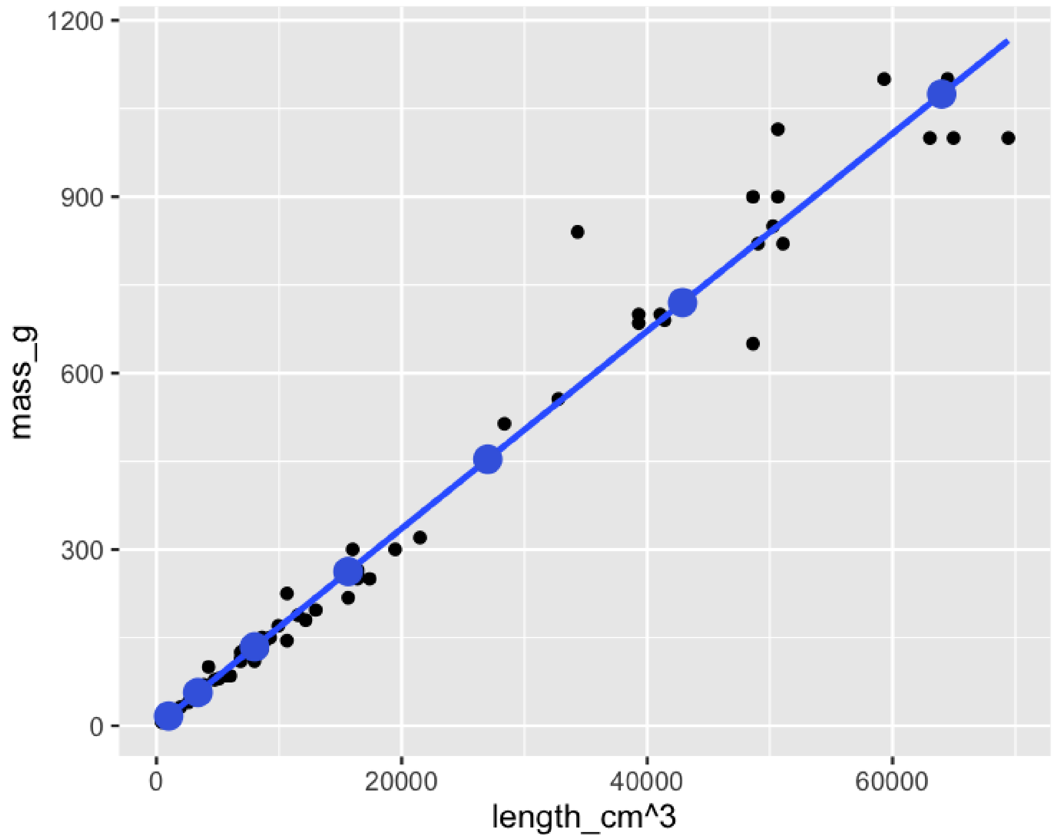
```r
prediction_data <- explanatory_data %>%
  mutate(
    mass_g = predict(mdl_perch, explanatory_data
  )
```

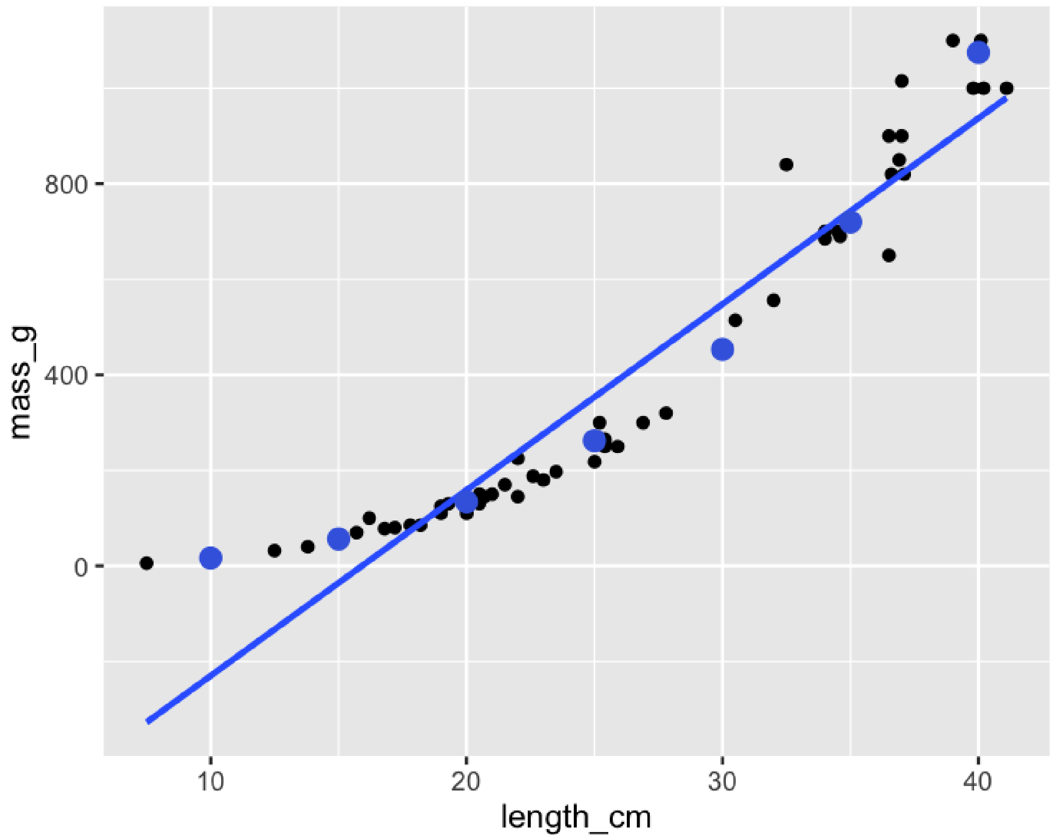
```
# A tibble: 7 x 2
  length_cm mass_g
      <dbl>  <dbl>
1        10   16.7
2        15   56.6
3        20  134.
4        25  262.
5        30  453.
6        35  720.
7        40 1075.
```

Plotting mass vs. length cubed

```
ggplot(perch, aes(length_cm ^ 3, mass_g)) +
  geom_point() +
  geom_smooth(method = "lm", se = FALSE) +
  geom_point(data = prediction_data, color = "blue")
```

```
ggplot(perch, aes(length_cm, mass_g)) +
  geom_point() +
  geom_smooth(method = "lm", se = FALSE) +
  geom_point(data = prediction_data, color = "blue")
```





Facebook advertising dataset

How advertising works

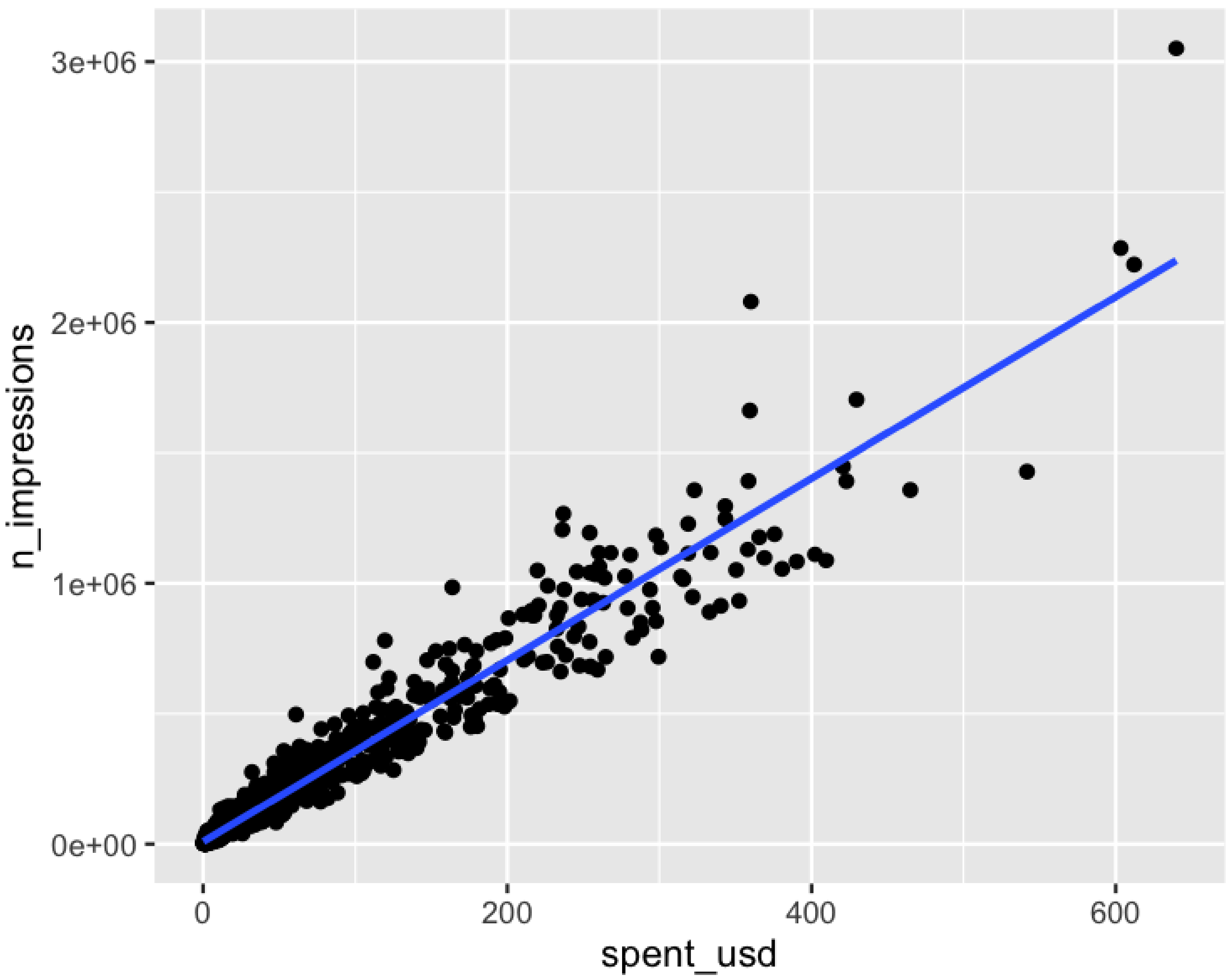
1. Pay Facebook to shows ads.
2. People see the ads ("impressions").
3. Some people who see it, click it.

- 936 rows
- Each row represents 1 advert

spent_usd	n_impressions	n_clicks
1.43	7350	1
1.82	17861	2
1.25	4259	1
1.29	4133	1
4.77	15615	3
...	...	...

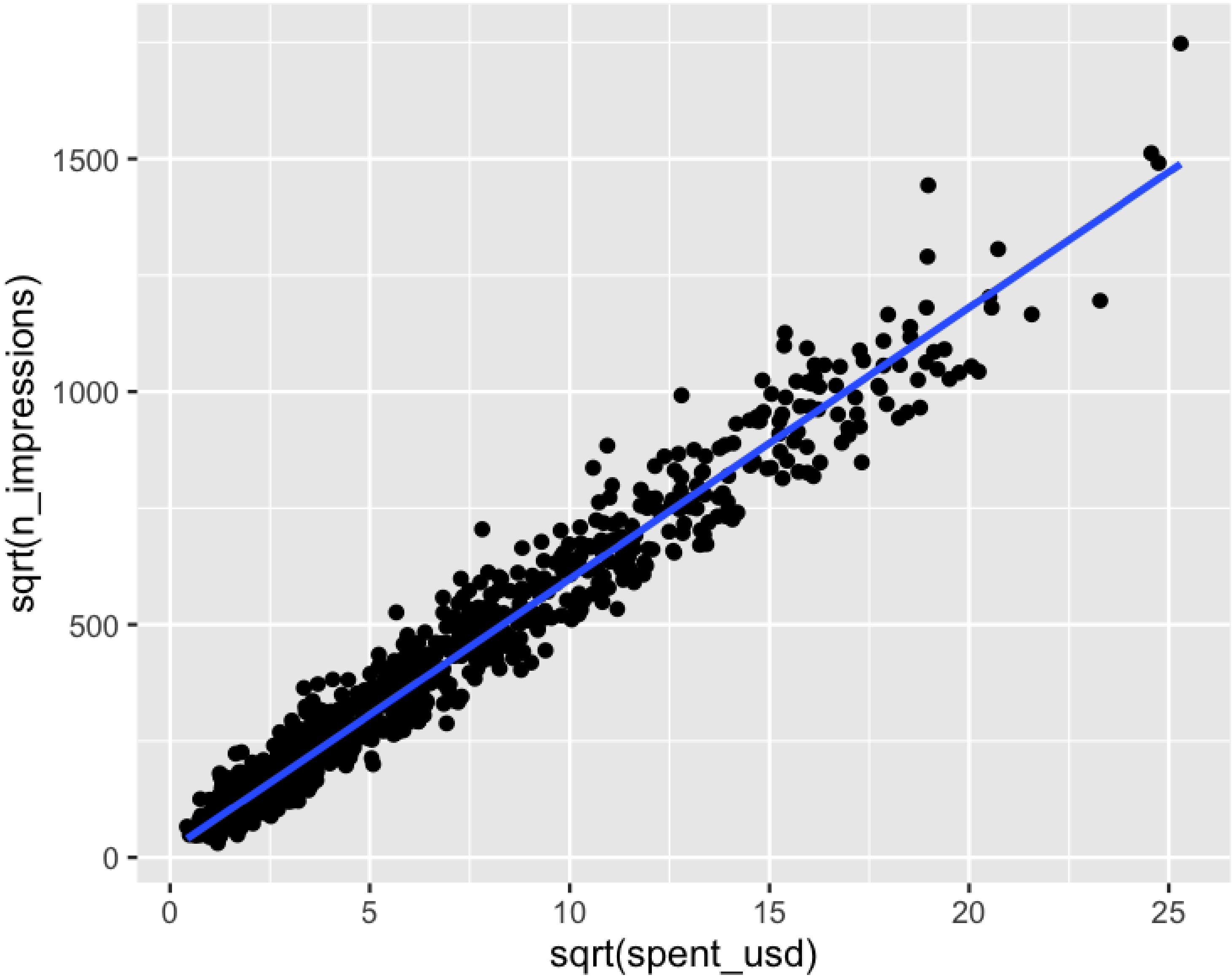
Plot is cramped

```
ggplot(
  ad_conversion,
  aes(spent_usd, n_impressions)
) +
  geom_point() +
  geom_smooth(method = "lm", se = FALSE)
```



Square root vs square root

```
ggplot(
  ad_conversion,
  aes(sqrt(spent_usd), sqrt(n_impressions))
) +
  geom_point() +
  geom_smooth(method = "lm", se = FALSE)
```



Modeling and predicting

```
mdl_ad <- lm(
  sqrt(n_impressions) ~ sqrt(spent_usd),
  data = ad_conversion
)
```

```
explanatory_data <- tibble(
  spent_usd = seq(0, 600, 100)
)
```

```
prediction_data <- explanatory_data %>%
  mutate(
    sqrt_n_impressions = predict(
      mdl_ad, explanatory_data
    ),
    n_impressions = sqrt_n_impressions ^ 2
  )
```

```
# A tibble: 7 x 3
  spent_usd sqrt_n_impressions n_impressions
      <dbl>              <dbl>         <dbl>
1         0               15.3          235.
2       100              598.        357289.
3       200              839.        703890.
4       300             1024.       1048771.
5       400             1180.       1392762.
6       500             1318.       1736184.
7       600             1442.       2079202.
```

Let's practice!

INTRODUCTION TO REGRESSION IN R