Regression - Solutions

Setup

```
library(tidyverse)
library(janitor)
kids <- read_csv('https://ditraglia.com/data/child_test_data.csv')
kids <- clean_names(kids)
kids <- kids |>
  mutate(mom_education = if_else(mom_hs == 1, 'HS', 'NoHS')) |>
  mutate(mom_education = fct_relevel(mom_education, 'NoHS'))
```

Exercise A - (10 min)

1. Interpret the regression coefficients from the following regression. Is there any way to change the model so the intercept has a more natural interpretation?

```
lm(kid_score ~ mom_iq, kids)
```

2. Use `ggplot2` to make a scatter plot with `mom_age` on the horizontal axis and `kid_score` on the vertical axis.
3. Use `geom_smooth()` to add the regression line `kid_score ~ mom_age` to your plot. What happens if you drop `method = 'lm'`?
4. Plot a histogram of the residuals from the regression `kid_score ~ mom_iq`. using `ggplot` with a bin width of 5. Is anything noteworthy?
5. Calculate the residuals “by hand” by subtracting the fitted values from `reg1` from the column `kid_score` in `kids`. Check that this gives the same result as `resid()`.
6. As long as you include an intercept in your regression, the residuals will sum to zero. Verify numerically using the residuals from the preceding part.
7. Regression residuals are uncorrelated with any predictors included in the regression. Verify numerically using the residuals you computed in part 5.

Solution

Part 1

Comparing two kids whose moms differ by one IQ point, we would predict that the kid whose mom has the higher IQ will score 0.61 points higher, on average, on the test. The intercept lacks a meaningful interpretation: it is the predicted test score for a kid whose mom has an IQ of zero, an impossible score.

```
lm(kid_score ~ mom_iq, kids)
```

```
Call:
lm(formula = kid_score ~ mom_iq, data = kids)

Coefficients:
(Intercept)       mom_iq
      25.80         0.61
```

While this doesn't affect our predictions in any way, it is arguably better to re-express the regression model so the intercept *does* have a meaningful interpretation. Consider the population linear regression $Y = \alpha + \beta X + U$. The least squares estimate of α is $\widehat{\alpha} = \bar{y} - \widehat{\beta}\bar{x}$ where $\widehat{\beta}$ is the least-squares estimate of β. In other words: the least squares regression line passes through the point $(\bar{x}, \bar{y})$. We can use this fact to give the regression intercept a meaningful interpretation as follows:

```
kids |>
  mutate(mom_iq = mom_iq - mean(mom_iq)) |>
  lm(kid_score ~ mom_iq, data = _)
```

```
Call:
lm(formula = kid_score ~ mom_iq, data = mutate(kids, mom_iq = mom_iq -
    mean(mom_iq)))

Coefficients:
(Intercept)       mom_iq
      86.80         0.61
```

Another way to obtain the same result is as follows:

```
lm(kid_score ~ I(mom_iq - mean(mom_iq)), kids)
```

```
Call:
lm(formula = kid_score ~ I(mom_iq - mean(mom_iq)), data = kids)

Coefficients:
             (Intercept)  I(mom_iq - mean(mom_iq))
                   86.80                      0.61
```

Replacing `mom_iq` with `mom_iq - mean(mom_iq)` gives us a transformed x variable with $\bar{x} = 0$. Hence, the $\widehat{\beta}\bar{x}$ term in the expression for $\widehat{\alpha}$ vanishes and we end up with $\widehat{\alpha} = \bar{y}$. Just to double-check:

```
mean(kids$kid_score)
```

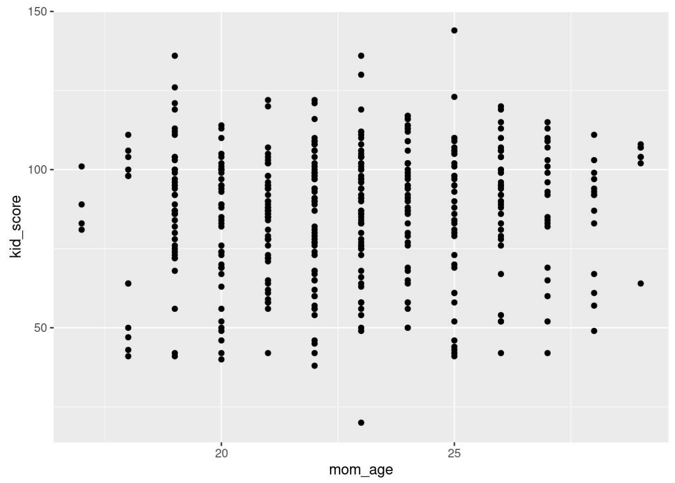
```
[1] 86.79724
```

In this transformed regression, the slope has the same interpretation as before, but now the intercept has a meaningful interpretation as well.

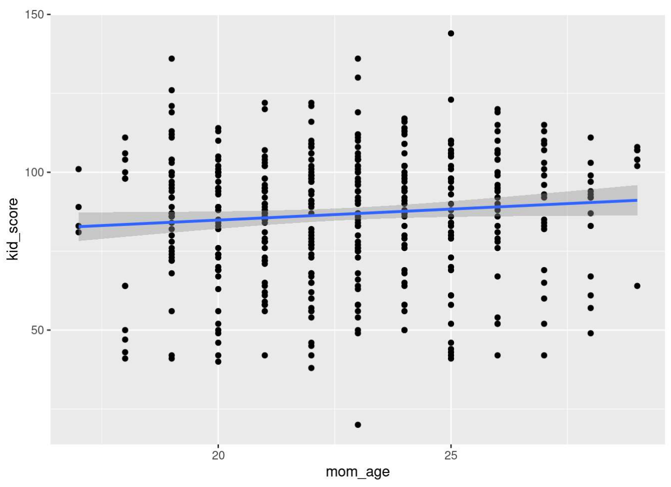
Parts 2 and 3

Consulting the help file for `geom_smooth()` we see that dropping `method = 'lm'` causes `ggplot2` to plot a smoother (a non-parametric regression line) using `stats::loess()` or `mgcv::gam()`, depending on how many observations are in our dataset.

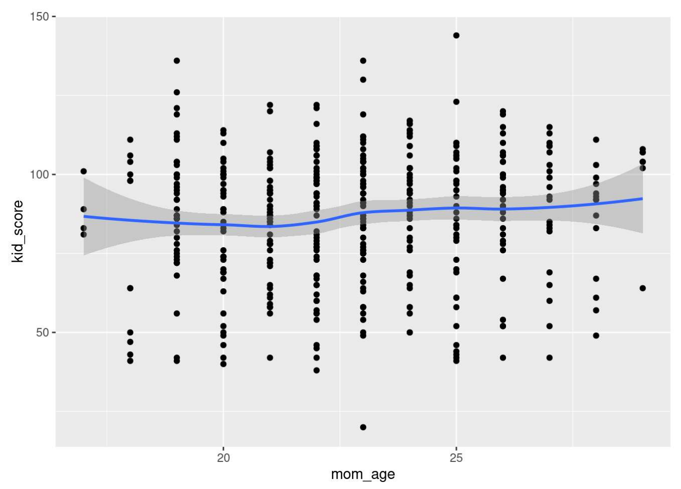
```
myplot <- kids |>
  ggplot(aes(x = mom_age, y = kid_score)) +
  geom_point()

myplot
```

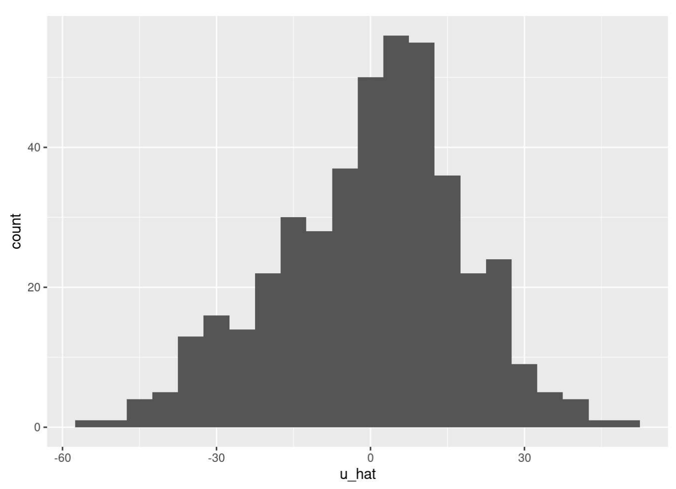


```
myplot +
  geom_smooth(method = 'lm')
```



```
myplot +
  geom_smooth()
```





Part 4

There appears to be some asymmetry in the distribution of the residuals. This is not necessarily a problem for our regression, but it may be interesting to think about why this could arise.

```r
reg <- lm(kid_score ~ mom_iq, kids)
tibble(u_hat = resid(reg)) |>
  ggplot(aes(x = u_hat)) +
  geom_histogram(binwidth = 5)
```

Parts 5-7

```r
alpha_hat <- coef(reg)[1]
beta_hat <- coef(reg)[2]
kids_with_residuals <- kids |>
  mutate(residuals = kid_score - alpha_hat - beta_hat * mom_iq)

all.equal(kids_with_residuals$residuals, resid(reg),
          check.attributes = FALSE)
```

```
[1] TRUE
```

```r
kids_with_residuals |>
  summarize(mean(residuals), cor(residuals, mom_iq))
```

```
# A tibble: 1 × 2
  `mean(residuals)` `cor(residuals, mom_iq)`
              <dbl>                    <dbl>
1          3.12e-14                 6.57e-16
```

Exercise B - (5 min)

1. Regress `mom_iq` on `kid_score` and `mom_hs`. Store your results in an object called `reg_reverse` and summarize with `tidy()` and `glance()`.
2. Extract the regression estimate, standard error, t-statistic, and p-value for the predictor `kid_score` *only*.
3. Plot the fitted values on the x-axis and `mom_iq` on the y-axis. Add a 45-degree line with `geom_abline()`.
4. Regress `mom_iq` on the fitted values from `reg_reverse`. Display the estimated regression coefficients and R-squared. Explain your results.

Solution

Parts 1-2

```r
# Part 1
library(broom)
reg_reverse <- lm(mom_iq ~ kid_score + mom_hs, kids)
reg_reverse |>
  tidy() |>
  knitr::kable(digits = 2)
```

term	estimate	std.error	statistic	p.value
(Intercept)	68.87	2.82	24.38	0
kid_score	0.30	0.03	9.31	0
mom_hs	6.83	1.58	4.31	0

```r
reg_reverse |>
  glance() |>
  knitr::kable(digits = 2)
```

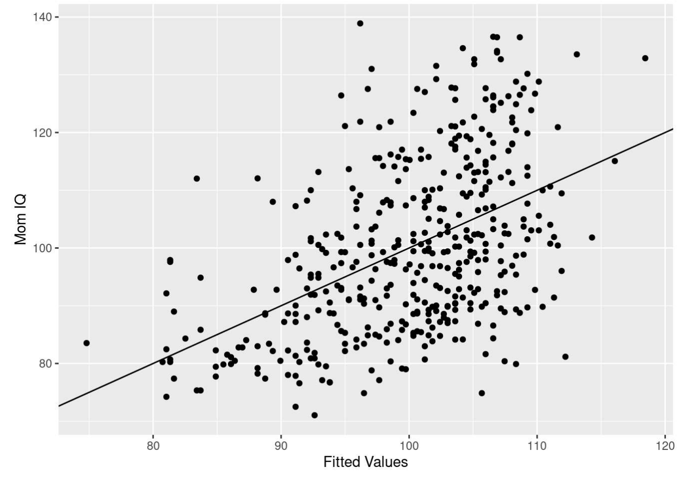
r.squared	adj.r.squared	sigma	statistic	p.value	df	logLik	AIC	BIC	deviance	df.residua
0.23	0.23	13.16	65.82	0	2	-1732.78	3473.56	3489.85	74631.64	431

```r
# Part 2
tidy(reg_reverse) |>
  filter(term == 'kid_score') |>
  select(estimate, std.error, statistic, p.value) |>
  knitr::kable(digits = 2)
```

estimate	std.error	statistic	p.value
0.3	0.03	9.31	0

Parts 3 and 4

```r
kids_augmented <- augment(reg_reverse, kids)
kids_augmented |>
  ggplot(aes(x = .fitted, y = mom_iq)) +
  geom_point() +
  geom_abline(intercept = 0, slope = 1) +
  xlab('Fitted Values') +
  ylab('Mom IQ')
```



When we regress Y_i on $\widehat{Y}_i$, the fitted values from a regression of Y_i on X_i, we get an intercept of zero and a slope of one:

```r
reg_y_vs_fitted <- lm(mom_iq ~ .fitted, kids_augmented)
reg_y_vs_fitted |>
  tidy() |>
  knitr::kable()
```

term	estimate	std.error	statistic	p.value
(Intercept)	0	8.7287661	0.00000	1
.fitted	1	0.0870593	11.48642	0

This makes sense. Suppose we wanted to choose α_0 and α_1 to minimize $\sum_{i=1}^{n}(Y_i - \alpha_0 - \alpha_1 \widehat{Y}_i)^2$ where $\widehat{Y}_i = \widehat{\beta}_0 + X_i'\widehat{\beta}_1$. This is equivalent to minimizing

$$\sum_{i=1}^{n}\left[Y_i - (\alpha_0 + \alpha_1\widehat{\beta}_0) - X_i'(\alpha_1\widehat{\beta}_1)\right]^2.$$

By construction $\widehat{\beta}_0$ and $\widehat{\beta}_1$ minimize $\sum_{i=1}^n (Y_i - \beta_0 - X_i'\beta_1)^2$, so unless $\widehat{\alpha_0} = 0$ and $\widehat{\alpha_1} = 1$ we'd have a contradiction! Similar reasoning explains why the R-squared values for the two regressions are the same:

```
c(glance(reg_reverse)$r.squared, glance(reg_y_vs_fitted)$r.squared)
```

```
[1] 0.2339581 0.2339581
```

The R-squared of a regression equals $1 - \text{SS}_{\text{residual}}/\text{SS}_{\text{total}}$

$$\text{SS}_{\text{total}} = \sum_{i=1}^n (Y_i - \bar{Y})^2, \quad \text{SS}_{\text{residual}} = \sum_{i=1}^n (Y_i - \widehat{Y}_i^2)$$

The total sum of squares is the same for both regressions because they have the same outcome variable. The residual sum of squares is the same because $\widehat{\alpha}_0 = 0$ and $\widehat{\alpha}_1 = 1$ together imply that both regressions have the same fitted values.

Exercise C - (8 min)

1. What does it *mean* to regress Y on an intercept only? Explain briefly.
2. Above we regressed `kid_score` on `mom_age` and `I(mom_age^2)`. Try omitting the `I()` wrapper around age-squared. What happens?
3. Run these regressions and interpret the results:
    a. `kid_score` on `mom_iq` and `mom_education`
    b. `kid_score` on `mom_iq, mom_education` and their interaction.
4. Above we used `ggplot2` to plot the regression line `mom_iq ~ kid_score`. Re-run the associated code but add `color = mom_education` as an extra argument to `aes()` and `se = FALSE` as an extra argument to `geom_smooth()`. What happens?

Solution

Part 1

A regression with *only* an intercept minimizes $\sum_{i=1}^n (Y_i - \beta)^2$ over β. Differentiating with respect to β, the first order condition is $-2\sum_{i=1}^n (Y_i - \widehat{\beta}) = 0$. Because the objective function is convex, this characterizes the global minimum. Re-arranging and solving for $\widehat{\beta}$ gives $\widehat{\beta} = \frac{1}{n}\sum_{i=1}^n Y_i$. In other words $\widehat{\beta} = \bar{Y}$, the sample mean. This makes sense: the sample mean is the obvious prediction of the next Y-observation if you have no other information to work with. Here we've shown that it is also the least squares solution.

Part 2

It won't work as expected: in a formula the symbol `^` has a special meaning related to creating interactions between variables.

Parts 3-4

In each of these two regressions, we fit a *different line* for the predictive relationship between `mom_iq` and `kid_score` depending on whether a kid's mom graduated from high school.

In the first regression, the two lines are constrained to have the same slope. For two kids whose moms have the same educational attainment but whose moms differ in IQ by point point, we would predict that the kid whose mom is higher will score around 0.6 points higher. For two kids whose moms have the same IQ but different educational attainment, we would predict that the kid whose mom graduated from high school will score about 6 points higher.

```
# 3a
lm(kid_score ~ mom_iq + mom_education, kids) |>
  tidy() |>
  knitr::kable(digits = 2, col.names = c('', 'Estimate', 'SE', 't-stat', 'p-value'))
```

	Estimate	SE	t-stat	p-value
(Intercept)	25.73	5.88	4.38	0.00
mom_iq	0.56	0.06	9.31	0.00
mom_educationHS	5.95	2.21	2.69	0.01

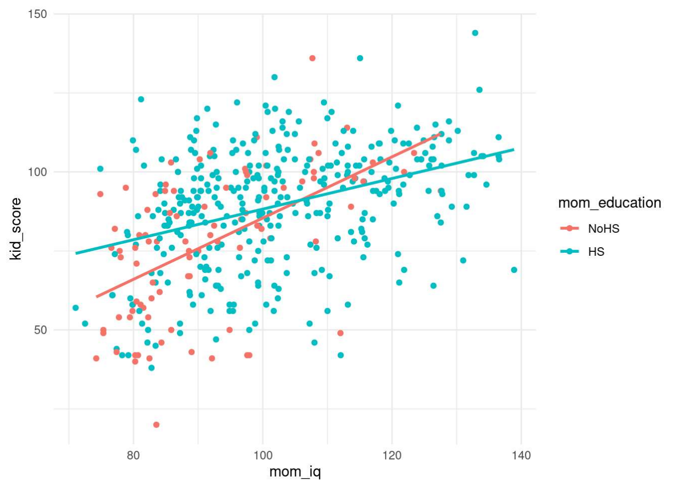
In the second regression, we allow both the intercept *and* slope to vary depending on whether a kid's mom graduated from high school. In other words, we add an *interaction* between `mom_iq` and `mom_education`. It's not straightforward to interpret the intercepts in this model, but we do find convincing evidence of a difference of slopes: for a kid whose mom did not graduate from high school, each additional IQ point increases our prediction of `kid_score` by *half a point more* than for a kid whose mom graduated from high school. This is an example with a very *strong* interaction effect.

```
# 3b
lm(kid_score ~ mom_iq * mom_education, kids) |>
  tidy() |>
  knitr::kable(digits = 2, col.names = c('', 'Estimate', 'SE', 't-stat', 'p-value'))
```

	Estimate	SE	t-stat	p-value
(Intercept)	-11.48	13.76	-0.83	0.4
mom_iq	0.97	0.15	6.53	0.0
mom_educationHS	51.27	15.34	3.34	0.0
mom_iq:mom_educationHS	-0.48	0.16	-2.99	0.0

This predictive difference jumps out clearly in the following plot. Notice how easy it is to plot different regression lines for different groups using `ggplot2`. We simply simply specify the aesthetic mapping `color` when setting up the plot and `geom_smooth()` figures out what we wanted. To make the plot easier to read, I added the option `se = FALSE` to remove the regression error bands and I used `theme_minimal()`.

```
kids |>
  ggplot(aes(x = mom_iq, y = kid_score, color = mom_education)) +
  geom_point() +
  geom_smooth(method = 'lm', se = FALSE) +
  theme_minimal()
```



Exercise D - (8 min)

1. Test the joint null hypothesis that the slope and intercept of the predictive relationship between `kid_score` and `mom_iq` is the same for kids whose mothers graduated from high school and those whose mothers did not. Does the p-value change much if you use the asymptotic version of the test rather than the finite-sample F?
2. Let `n` be the number of rows in `kids`. Generate two random vectors as follows:
    - `x` is a vector of `n` independent standard normal observations.
    - `z` equals `mom_iq` plus independent standard normal noise.

    Carry out a new regression, `reg_augmented`, that adds the predictors `x` and `z` to your earlier regression of `kid_score` on `mom_iq, mom_education` and their interaction. Finally, test the null hypothesis that `x` and `z` are irrelevant for predicting `kid_score` after taking into account `mom_iq` and `mom_education`. Interpret your findings. Do the results of the test make sense?

Solution

We find very strong evidence *against* the null hypothesis that there is no difference in the slope or intercept of the relationship between `kid_score` and `mom_iq` depending on `mom_education`:

```
# Part 1
library(car)

reg_interact <- lm(kid_score ~ mom_iq * mom_education, kids)

reg_interact |>
  tidy() |>
  knitr::kable(digits = 2)
```

term	estimate	std.error	statistic	p.value
(Intercept)	-11.48	13.76	-0.83	0.4
mom_iq	0.97	0.15	6.53	0.0
mom_educationHS	51.27	15.34	3.34	0.0
mom_iq:mom_educationHS	-0.48	0.16	-2.99	0.0

```
restrictions1 <- c('mom_educationHS = 0', 'mom_iq:mom_educationHS = 0')
linearHypothesis(reg_interact, restrictions1)
```

```
Linear hypothesis test

Hypothesis:
mom_educationHS = 0
mom_iq:mom_educationHS = 0

Model 1: restricted model
Model 2: kid_score ~ mom_iq * mom_education

  Res.Df    RSS Df Sum of Sq      F    Pr(>F)
1    432 144137
2    430 138879  2    5258.7 8.1411 0.0003386 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
linearHypothesis(reg_interact, test = 'Chisq', restrictions1)
```

```
Linear hypothesis test

Hypothesis:
mom_educationHS = 0
mom_iq:mom_educationHS = 0

Model 1: restricted model
Model 2: kid_score ~ mom_iq * mom_education
```

```
  Res.Df    RSS Df Sum of Sq  Chisq Pr(>Chisq)
1    432 144137
2    430 138879  2    5258.7 16.282  0.0002913 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

We know that `x` and `z` should be irrelevant for predicting `kid_score`: `x` is just random noise, and `z` equals a noisy measure of a predictor that is already included in the regression. Unsurprisingly, the p-value for the test of the null hypothesis that the coefficients on `x` and `z` are both zero is quite large:

```
# Part 2
set.seed(1693)
reg_augmented <- kids |>
  mutate(x = rnorm(nrow(kids)), z = mom_iq + rnorm(nrow(kids))) |>
  lm(kid_score ~ mom_iq * mom_education + x + z, data = _)

reg_augmented |>
  tidy() |>
  knitr::kable(digits = 2)
```

term	estimate	std.error	statistic	p.value
(Intercept)	-12.42	13.82	-0.90	0.37
mom_iq	0.53	0.90	0.58	0.56
mom_educationHS	52.21	15.41	3.39	0.00
x	-0.63	0.84	-0.74	0.46
z	0.45	0.89	0.51	0.61
mom_iq:mom_educationHS	-0.49	0.16	-3.03	0.00

```
restrictions2 <- c('x = 0', 'z = 0')
linearHypothesis(reg_augmented, restrictions2)
```

```
Linear hypothesis test

Hypothesis:
x = 0
z = 0

Model 1: restricted model
Model 2: kid_score ~ mom_iq * mom_education + x + z

  Res.Df    RSS Df Sum of Sq      F Pr(>F)
1    430 138879
2    428 138617  2    261.82 0.4042 0.6678
```