Lecture 16: Exploring multiple regression

Contents

Learning Objectives: See how coefficients, p-values, and predictions change when you add and remove variables from a multiple linear regression model.

Instructions: Submit written answers to the numbered questions on pencil and paper! This activity is Mini Quiz 6! You don't need to write a lot in your handwritten answers, but I should be able to tell that you were making progress and learning something! You can work in groups!

Getting Started

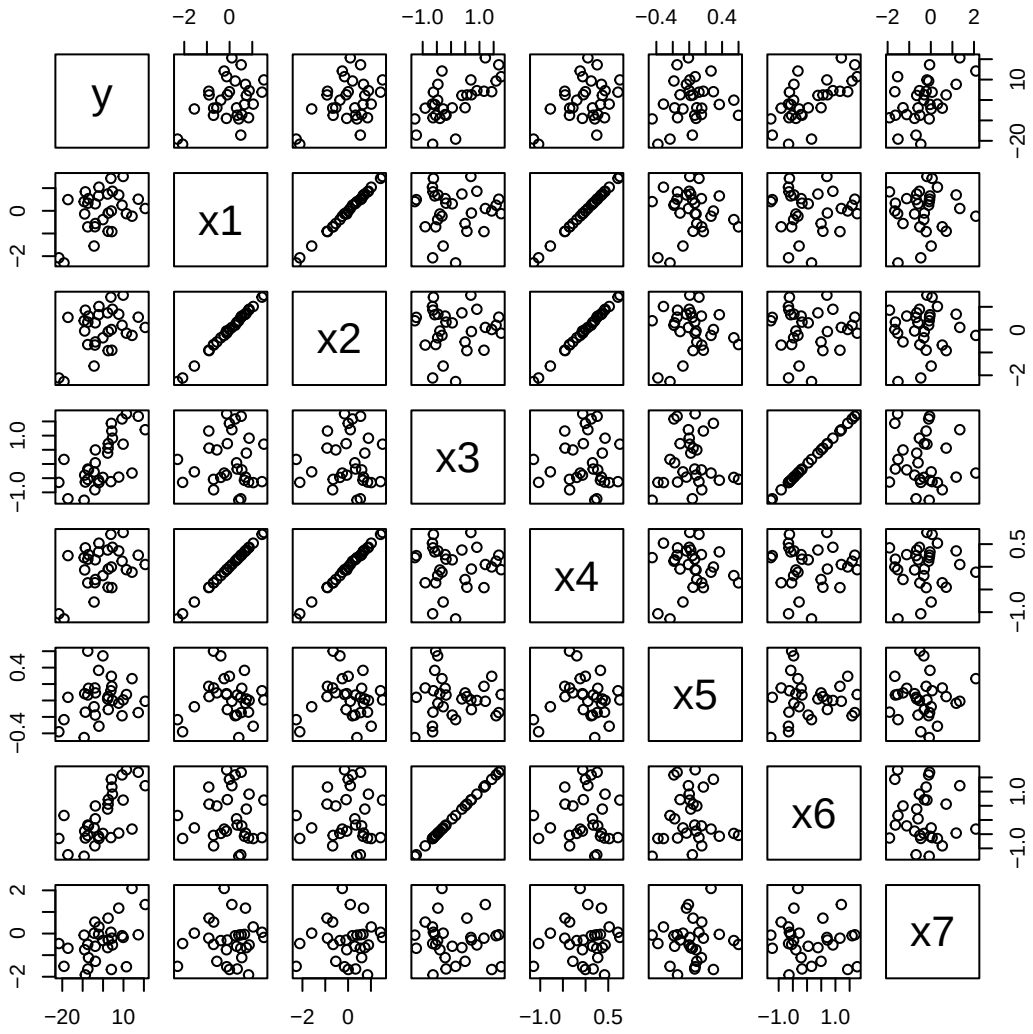
I made you two fake datasets to explore some properties of multiple regression. Download them with the following code:

```
library(tidyverse)
dat.small <- read.csv("https://anna-neufeld.github.io/stat202_tutorials/Week9/dat.small.csv")
dat.big <- read.csv("https://anna-neufeld.github.io/stat202_tutorials/Week9/dat.big.csv")
```

1. Starting with dat.small

This dataset has 30 observations, one response variable `y`, and 7 predictor variables `x1`, `x2`, ..., `x7`. The following code will show you relationships between all of your variables.

```
pairs(dat.small)
```



The following code and output tells you about the pairwise correlations between all variables in your dataset.For example, 0.395 is the correlation between y and $x2$ in this dataset. The values on the diagonal are all 1, since every variable is perfectly correlated with itself!

```
cor(dat.small)
```

	y	x1	x2	x3	x4	x5	x6	x7
y	1.0000000	0.4037688	0.3953713	0.6584705	0.4037688	0.1849520	0.6589385	0.4923300
x1	0.4037688	1.0000000	0.9994114	0.0000000	1.0000000	0.0153318	0.0011602	0.0578495
x2	0.3953713	0.9994114	1.0000000	-0.0041397	0.9994114	0.0308933	-0.0029802	0.0471196
x3	0.6584705	0.0000000	-0.0041397	1.0000000	0.0000000	0.0009792	0.9999993	-0.1126423
x4	0.4037688	1.0000000	0.9994114	0.0000000	1.0000000	0.0153318	0.0011602	0.0578495
x5	0.1849520	0.0153318	0.0308933	0.0009792	0.0153318	1.0000000	0.0009970	0.0266500
x6	0.6589385	0.0011602	-0.0029802	0.9999993	0.0011602	0.0009970	1.0000000	-0.1125751
x7	0.4923300	0.0578495	0.0471196	-0.1126423	0.0578495	0.0266500	-0.1125751	1.0000000

We will now try out a bunch of regressions of y on subsets of the predictor variables x1,. . . ,x7. For each exercise, try out some code in your console, and then record a handwritten answer on your mini quiz.

1. **Highly correlated predictors:** Compare a linear regression of y on x1 to a linear regression of y on x1 and x2. Specifically, compare your two models in terms of: the coefficient for x1, its standard error, its p-value, and R^2. What did you learn from this problem?

2. **Perfectly uncorrelated predictors:** Compare a linear regression of `y` on `x1` to a linear regression of `y` on `x1` and `x3`. What do you notice about: the coefficient on `x1`, its standard error, its p-value, and R^2? What did you learn from this problem?

3. **Perfectly correlated predictors:** What happens if you fit a linear regression of `y` on `x1` and `x4`? What do you learn from this?

4. **Mostly uncorrelated predictors:** What happens if you fit a linear regression of `y` on `x1` and `x5` and `x6` and `x7`. How does this model compare to ones from the previous parts? Play around with this model by adding or removing one variable at a time, and be sure to check out the correlations between different pairs of these variables. What did you learn from this, and how does it relate to the previous parts of the question?

5. **What would be your pick for a "final model" for this dataset?** Be sure that you can justify your answer!

2. Moving on to dat.big

The dataset `dat.big` that you downloaded above has 30 observations and 29 variables. The following command will let you fit a regression of `y` on all other variables in the dataset.

```
mod.full <- lm(y~., data=dat.big)
```

Check the R^2 of this model. While you are at it, compare your predicted values (stored in `mod.full$fitted`) to the actual values of y in your dataset. What is going on in this dataset?

Do some visualizing and exploring of your data. Consider some smaller models that have fewer than 29 variables in them.

6. Based on your exploration, do you think that the best model is the one with the highest R^2? Why or why not?

3. Optional: only move on to this if you have time!

The dataset `dat.big.test`, which can be downloaded below, comes from exactly the same population as `dat.big`.

```
library(tidyverse)
dat.big.test <- read.csv("https://anna-neufeld.github.io/stat202_tutorials/Week9/dat.big.test.csv")
```

In this question, we will see that the giant `mod.full` from Question 2 does a very bad job making predictions for unseen examples, despite the fact that its predictions were perfect on its own dataset. This phenomenon is known as *overfitting*: `mod.full` was so complex that it perfectly traced the training dataset values, but cannot generalize to unseen examples. This is an example of why we cannot directly trust R^2 as a measure of model performance.

Run the following code, which computes the SSE of this "full model" when it is applied to the test set.

```
preds.test <- predict(mod.full, newdata=dat.big.test)
sum((preds.test - dat.big.test$y)^2)
```

Compare this SSE to a much simpler model, which only uses `x1` and `x5` for predictions. What do you notice?

```
mod.small <- lm(y~x1+x5, data=dat.big)
preds.test.small <- predict(mod.small, newdata=dat.big.test)
sum((preds.test.small - dat.big.test$y)^2)
```

It is not, however, the case that the simplest model is always the best. We can see this by considering the intercept-only model, which is worse than the model above.

```
mod.intercept <- lm(y~1, data=dat.big)
preds.intercept <- predict(mod.intercept, newdata=dat.big.test)
sum((preds.intercept- dat.big.test$y)^2)
```

3. Based on the past few explorations, as well as what you explored in Question 2, what do you think is the "best model" for this dataset?

In real life, a common way to do model selection is to select the model that achieves the highest predictive accuracy on a test dataset. We will return to this idea during the last week of class.

Takeaways

I hope you learned that:

- Multiple regression is tricky in the presence of correlated predictors.
  - When the correlation between two predictors is 1, the model can literally not be fit (the design matrix is singular)!
  - When the correlation between two predictors is quite high, the model is *unstable*. This also has to do with near-singularity (tiny eigenvalues in the design matrix).
  - When the correlation between two predictors is actually 0, adding the second variable to the model does not change the coefficient on the first!
  - When the correlation between two predictors is small but non-zero, adding a new variable to the model always changes the coefficient on the first, but it might only change it slightly.
- Measures like R^2, computed using the same dataset that was used to fit the model, will always increase when you increase the number of predictors. so just looking at an increasing R^2 is not a good way to say that a more complex model is "better".
  - A test set is a better way to assess the ability of a model to generate accurate predictions for a *new sample*.

We will return to some of these topics next week when we study model selection!

Acknowledgements

The formatting of this tutorial was adopted from an OpenIntro lab.