Chapter 2

DJM

30 January 2018

What is this chapter about?

> Problems with regression, and in particular, linear regression

A quick overview:

1. The truth is almost never linear.
2. Collinearity **can** cause difficulties for numerics and interpretation.
3. The estimator depends strongly on the marginal distribution of X.
4. Leaving out important variables is bad.
5. Noisy measurements of variables can be bad, but it may not matter.

Asymptotic notation

- The Taylor series expansion of the mean function $\mu(x)$ at some point u

$$\mu(x) = \mu(u) + (x-u)^\top \frac{\partial \mu(x)}{\partial x}|_{x=u} + O(\|x-u\|^2)$$

- The notation $f(x) = O(g(x))$ means that for any x there exists a constant C such that $f(x)/g(x) < C$.
- More intuitively, this notation means that the remainder (all the higher order terms) are about the size of the distance between x and u or smaller.
- So as long as we are looking at points u near by x, a linear approximation to $\mu(x) = \mathbb{E}[Y \mid X = x]$ is reasonably accurate.

What is bias?

- We need to be more specific about what we mean when we say **bias**.
- Bias is neither good nor bad in and of itself.
- A very simple example: let $Z_1, \ldots, Z_n \sim N(\mu, 1)$.
- We don't know μ, so we try to use the data (the Z_i's) to estimate it.
- I propose 3 estimators:
    1. $\widehat{\mu}_1 = 12$,
    2. $\widehat{\mu}_2 = Z_6$,
    3. $\widehat{\mu}_3 = \overline{Z}$.
- The **bias** (by definition) of my estimator is $\mathbb{E}[\widehat{\mu}] - \mu$.
- Calculate the bias and variance of each estimator.

Regression in general

- If I want to predict Y from X, it is almost always the case that

$$\mu(x) = \mathbb{E}[Y \mid X = x] \neq x^\top \beta$$

- There are always those errors $O(\|x - u\|)^2$, so the **bias** is not zero.
- We can include as many predictors as we like, but this doesn't change the fact that the world is **non-linear**.

Covariance between the prediction error and the predictors

- In theory, we have (if we know things about the state of nature)

$$\beta^* = \arg\min_\beta \mathbb{E}\left[\|Y - X\beta\|^2\right] = \mathrm{Cov}[X,\ X]^{-1}\,\mathrm{Cov}[X,\ Y]$$

- Define $v^{-1} = \mathrm{Cov}[X,\ X]^{-1}$.
- Using this optimal value β^*, what is $\mathrm{Cov}[Y - X\beta^*,\ X]$?

$$\begin{aligned}
\mathrm{Cov}[Y - X\beta^*,\ X] &= \mathrm{Cov}[Y,\ X] - \mathrm{Cov}[X\beta^*,\ X] && \text{(Cov is linear)}\\
&= \mathrm{Cov}[Y,\ X] - \mathrm{Cov}\left[X(v^{-1}\mathrm{Cov}[X,\ Y]),\ X\right] && \text{(substitute the def. of } \beta^*\text{)}\\
&= \mathrm{Cov}[Y,\ X] - \mathrm{Cov}[X,\ X]\,v^{-1}\mathrm{Cov}[X,\ Y] && \text{(Cov is linear in the first arg)}\\
&= \mathrm{Cov}[Y,\ X] - \mathrm{Cov}[X,\ Y] = 0.
\end{aligned}$$

Bias and Collinearity

- Adding or dropping variables may impact the bias of a model
- Suppose $\mu(x) = \beta_0 + \beta_1 x_1$. It **is** linear. What is our estimator of β_0?
- If we instead estimate the model $y_i = \beta_0$, our estimator of β_0 will be biased. How biased?
- But now suppose that $x_1 = 12$ always. Then we don't need to include x_1 in the model. Why not?
- Form the matrix $[1\ x_1]$. Are the columns collinear? What does this actually mean?

When two variables are collinear, a few things happen.

1. We cannot **numerically** calculate $(\mathbf{X}^\top\mathbf{X})^{-1}$. It is rank deficient.
2. We cannot **intellectually** separate the contributions of the two variables.
3. We can (and should) drop one of them. This will not change the bias of our estimator, but it will alter our interpretations.
4. Collinearity appears most frequently with many categorical variables.
5. In these cases, software **automatically** drops one of the levels resulting in the baseline case being in the intercept. Alternately, we could drop the intercept!
6. High-dimensional problems (where we have more predictors than data points) also lead to rank deficiencies.
7. There are methods (regularizing) which attempt to handle this issue (both the numerics and the interpretability). We may have time to cover them slightly.

White noise

White noise is a stronger assumption than **Gaussian**.

Consider a random vector ϵ.

1. $\epsilon \sim \mathrm{N}(0, \Sigma)$.
2. $\epsilon_i \sim \mathrm{N}(0, \sigma^2(x_i))$.
3. $\epsilon \sim \mathrm{N}(0, \sigma^2 I)$.

The third is white noise. The ϵ are normal, their variance is constant for all i and independent of x_i, and they are independent.

Asymptotic efficiency

> This and MLE are covered in 420.

There are many properties one can ask of estimators $\widehat{\theta}$ of parameters θ

1. Unbiased: $\mathbb{E}\left[\widehat{\theta}\right] - \theta = 0$
2. Consistent: $\widehat{\theta} \xrightarrow{n\rightarrow\infty} \theta$
3. Efficient: $\mathbb{V}\left[\widehat{\theta}\right]$ is the smallest of all unbiased estimators
4. Asymptotically efficient: Maybe not efficient for every n, but in the limit, the variance is the smallest of all unbiased estimators.
5. Minimax: over all possible estimators in some class, this one has the smallest MSE for the worst problem.
6. ...

Problems with R-squared

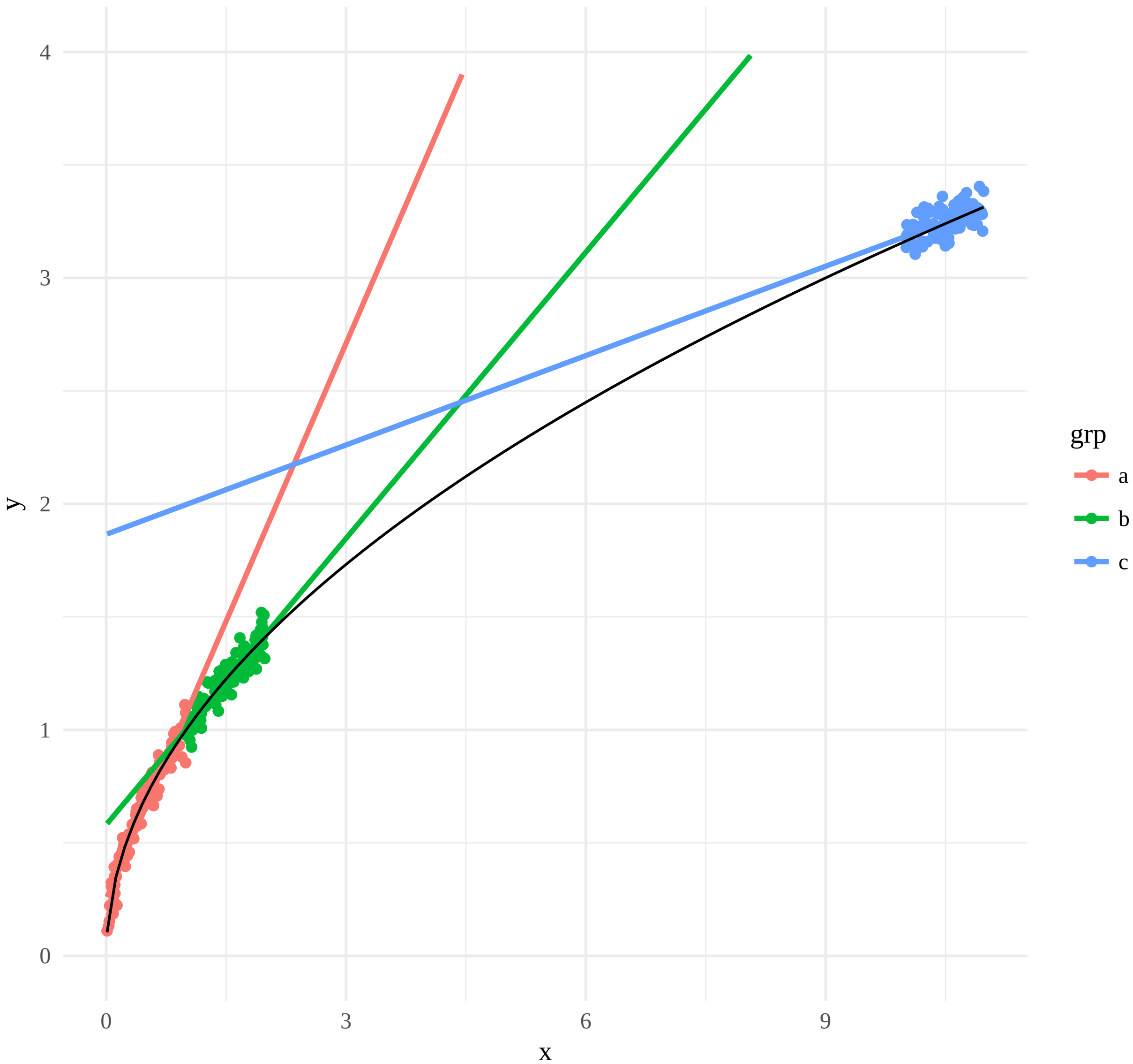
$$R^2 = 1 - \frac{SSE}{\sum_{i=1}^n (Y_i - \overline{Y})^2} = 1 - \frac{MSE}{\frac{1}{n}\sum_{i=1}^n (Y_i - \overline{Y})^2} = 1 - \frac{SSE}{SST}$$

- This gets spit out by software
- X and Y are both normal with (empirical) correlation r, then $R^2 = r^2$
- In this nice case, it measures how tightly grouped the data is about the regression line
- Data that is tightly grouped about the regression line can be predicted accurately by the regression line.
- Unfortunately, the implication does not go both ways.
- High R^2 can be achieved in many ways, same with low R^2
- You should just ignore it completely (and the adjusted version), and encourage your friends to do the same

High R-squared with non-linear relationship

```r
genY <- function(X, sig) Y = sqrt(X)+sig*rnorm(length(X))
sig=0.05; n=100
X1 = runif(n,0,1)
X2 = runif(n,1,2)
X3 = runif(n,10,11)
df = data.frame(x=c(X1,X2,X3), grp = rep(letters[1:3],each=n))
df$y = genY(df$x,sig)
ggplot(df, aes(x,y,color=grp)) + geom_point() +
```

```
  geom_smooth(method = 'lm', fullrange=TRUE,se = FALSE) +
  ylim(0,4) + stat_function(fun=sqrt,color='black')
```




```
df %>% group_by(grp) %>% summarise(rsq = summary(lm(y~x))$r.sq)
```

```
## # A tibble: 3 x 2
##   grp      rsq
##   <fctr> <dbl>
## 1 a      0.930
## 2 b      0.868
## 3 c      0.369
```