Gov 50: 14. More Regression and Model Fit

Matthew Blackwell

Harvard University

Roadmap

1. Model fit

2. Multiple regression

1/ Model fit

Presidential popularity and the midterms

- Does popularity of the president or recent changes in the economy better predict midterm election outcomes?

Name	Description
`year`	midterm election year
`president`	name of president
`party`	Democrat or Republican
`approval`	Gallup approval rating at midterms
`rdi_change`	% change in real disposable income over the year before midterms
`seat_change`	change in the number of House seats for the president's party

```r
library(gov50data)
midterms
```

```
## # A tibble: 20 x 6
##     year president  party approval seat_change rdi_change
##    <dbl> <chr>      <chr>    <dbl>       <dbl>      <dbl>
##  1  1946 Truman     D           33         -55     NA    
##  2  1950 Truman     D           39         -29      8.2  
##  3  1954 Eisenhower R           61          -4      1    
##  4  1958 Eisenhower R           57         -47      1.1  
##  5  1962 Kennedy    D           61          -4      5    
##  6  1966 Johnson    D           44         -47      5.3  
##  7  1970 Nixon      R           58          -8      6.6  
##  8  1974 Ford       R           54         -43      6.4  
##  9  1978 Carter     D           49         -11      7.7  
## 10  1982 Reagan     R           42         -28      4.8  
## 11  1986 Reagan     R           63          -5      5.1  
## 12  1990 H.W. Bush  R           58          -8      5.6  
## 13  1994 Clinton    D           46         -53      3.9  
## 14  1998 Clinton    D           66           5      5.6  
## 15  2002 W. Bush    R           63           6      2.6  
## 16  2006 W. Bush    R           38         -30      5.7  
## 17  2010 Obama      D           45         -63      3.5  
## 18  2014 Obama      D           40         -13      4.6  
## 19  2018 Trump      R           38         -42      4.1  
## 20  2022 Biden      D           42          NA     -0.003
```

Fitting the approval model

```r
fit.app <- lm(seat_change ~ approval, data = midterms)
fit.app
```

```
##
## Call:
## lm(formula = seat_change ~ approval, data = midterms)
##
## Coefficients:
## (Intercept)     approval
##      -96.58         1.42
```

For a one-point increase in presidential approval, the predicted seat change increases by 1.42

Fitting the income model

```
fit.rdi <- lm(seat_change ~ rdi_change, data = midterms)
fit.rdi
```

```
##
## Call:
## lm(formula = seat_change ~ rdi_change, data = midterms)
##
## Coefficients:
## (Intercept)   rdi_change
##      -29.41         1.21
```

For a one-point increase in the change in real disposable income, the predicted seat change increases by 1.21

Comparing models

- How well do the models “fit the data”?
  - How well does the model predict the outcome variable in the data?

Model fit

Model prediction error:

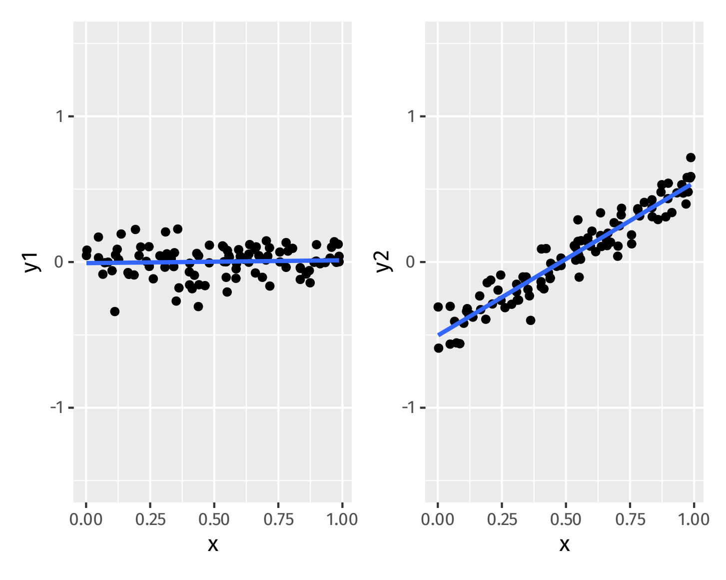
$$\text{prediction error} = \sum_{i=1}^{n} (\text{actual}_i - \text{predicted}_i)^2$$

Prediction error for regression: **Sum of squared residuals**

$$\text{SSR} = \sum_{i=1}^{n} \left(Y_i - \widehat{Y}_i \right)^2$$

Lower SSR is better, right?

These two regression lines have approximately the same SSR:



Benchmarking model fit

Benchmarking our predictions using the **proportional reduction in error**:

$$\frac{\text{reduction in prediction error using model}}{\text{baseline prediction error}}$$

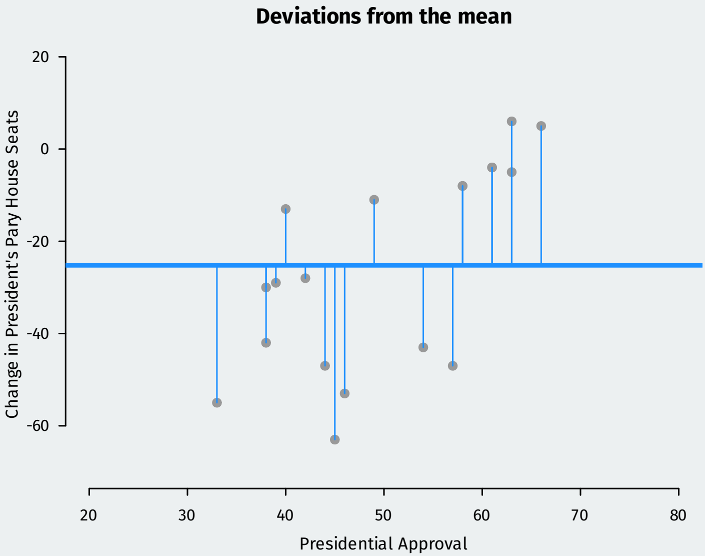
Baseline prediction error without a regression is using the mean of Y to predict. This is called the **Total sum of squares**:

$$\text{TSS} = \sum_{i=1}^{n} (Y_i - \overline{Y})^2$$

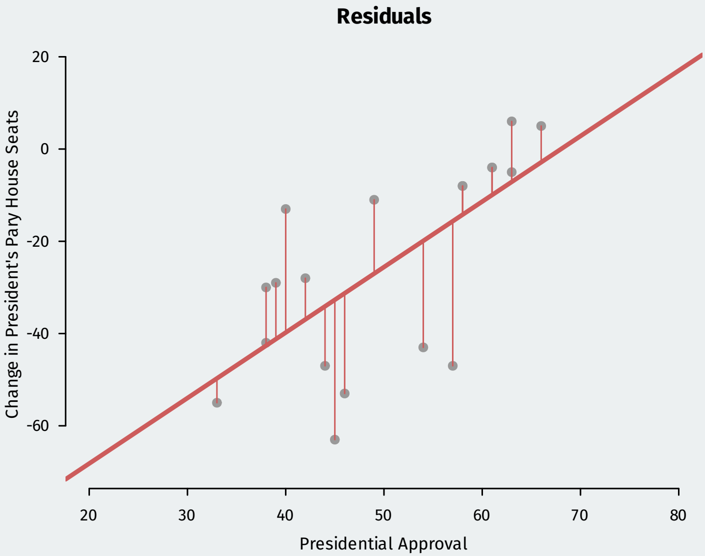
Leads to the **coefficient of determination**, R^2, one summary of LS model fit:

$$R^2 = \frac{TSS - SSR}{TSS} = \frac{\text{how much smaller LS prediction errors are vs mean}}{\text{prediction error using the mean}}$$

Total SS vs SSR



Total SS vs SSR



Model fit in R

- To access R^2 from the `lm()` output, use the `summary()` function:

```
fit.app.sum <- summary(fit.app)
fit.app.sum$r.squared
```

```
## [1] 0.45
```

- Compare to the fit using change in income:

```
fit.rdi.sum <- summary(fit.rdi)
fit.rdi.sum$r.squared
```

```
## [1] 0.012
```

- Which does a better job predicting midterm election outcomes?

Accessing model fit via `broom` package

We can also access summary statistics like model fit using the `glance()` function from `broom`:

```r
library(broom)
glance(fit.app)
```

```
## # A tibble: 1 x 12
##   r.squared adj.r.squared sigma statistic p.value    df
##       <dbl>         <dbl> <dbl>     <dbl>   <dbl> <dbl>
## 1     0.450         0.418  16.9      13.9 0.00167     1
## # i 6 more variables: logLik <dbl>, AIC <dbl>, BIC <dbl>,
## #   deviance <dbl>, df.residual <int>, nobs <int>
```

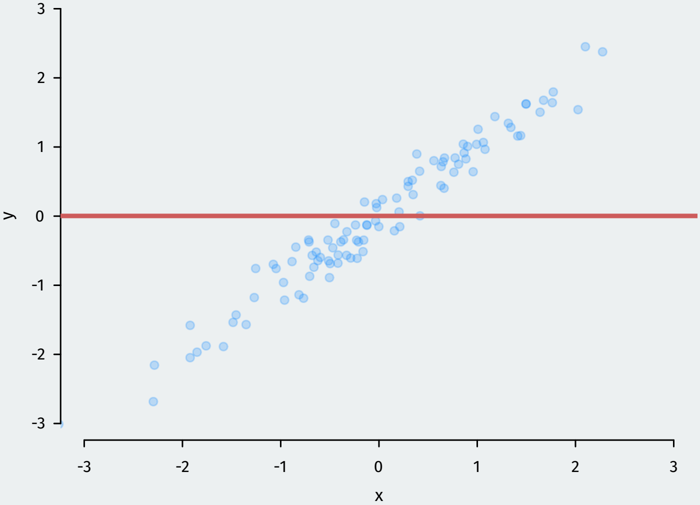
Fake data, better fit

- Little hard to see what's happening in that example.
- Let's look at fake variables x and y:

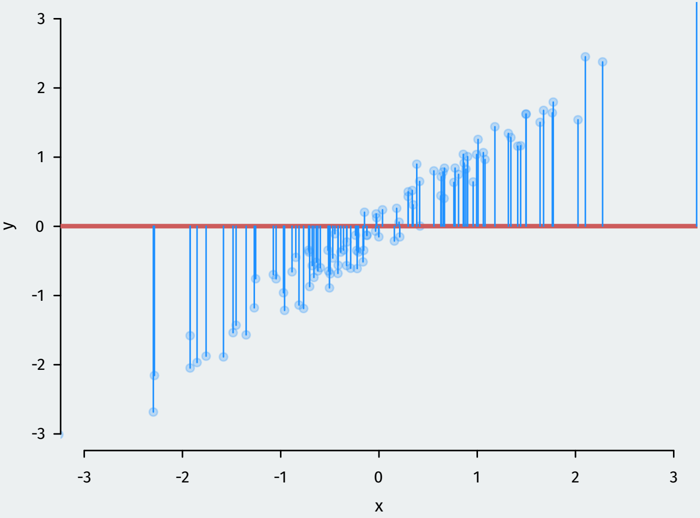
```
fit.x <- lm(y ~ x)
```

- Very good model fit: $R^2 \approx 0.95$

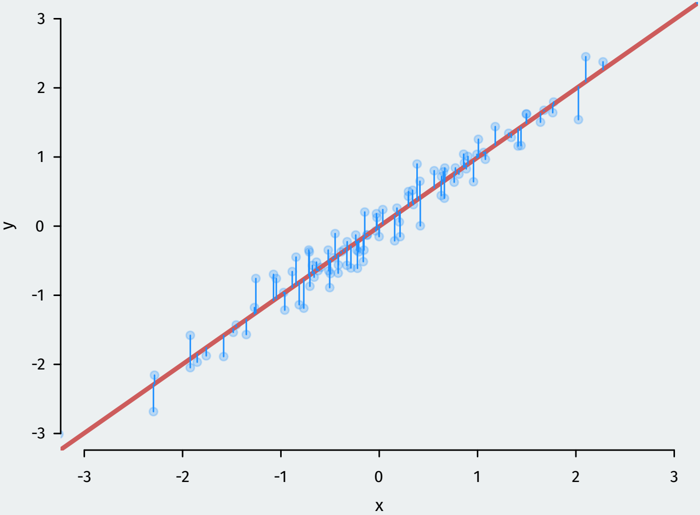
Fake data, better fit



Fake data, better fit

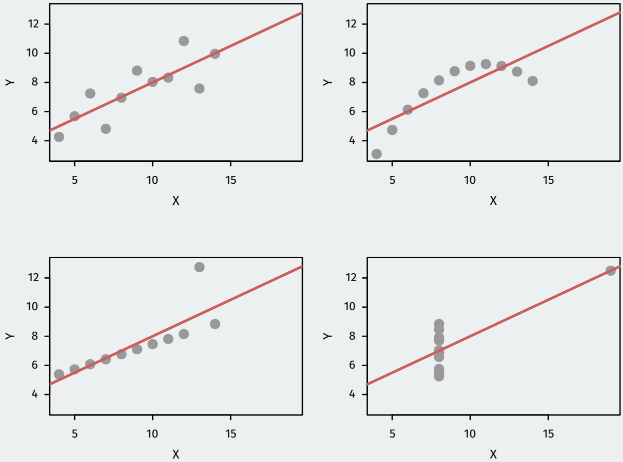


Fake data, better fit



Is R-squared useful?

- Can be very misleading. Each of these samples have the same R^2 even though they are vastly different:



Overfitting

- **In-sample fit**: how well your model predicts the data used to estimate it.
    - R^2 is a measure of in-sample fit.
- **Out-of-sample fit**: how well your model predicts new data.
- **Overfitting**: OLS optimizes in-sample fit; may do poorly out of sample.
    - Example: predicting winner of Democratic presidential primary with gender of the candidate.
    - Until 2016, gender was a **perfect** predictor of who wins the primary.
    - Prediction for 2016 based on this: Bernie Sanders as Dem. nominee.
    - Bad out-of-sample prediction due to overfitting!

2/ Multiple regression

Multiple predictors

What if we want to predict Y as a function of many variables?

$$\texttt{seat\_change}_i = \alpha + \beta_1 \texttt{approval}_i + \beta_2 \texttt{rdi\_change}_i + \epsilon_i$$

Why?

- Better predictions (at least in-sample).
- Better interpretation as **ceteris paribus** relationships:
  - β_1 is the relationship between `approval` and `seat_change` holding `rdi_change` constant.
  - **Statistical control** in a cross-sectional study.

Multiple regression in R

```r
mult.fit <- lm(seat_change ~ approval + rdi_change,
               data = midterms)
mult.fit
```

```
##
## Call:
## lm(formula = seat_change ~ approval + rdi_change, data = midterms)
##
## Coefficients:
## (Intercept)     approval   rdi_change
##     -117.23         1.53         3.22
```

- $\hat{\alpha} =$ -117.2: average seat change president has 0% approval and no change in income levels.
- $\hat{\beta}_1 = 1.53$: average increase in seat change for additional percentage point of approval, **holding RDI change fixed**
- $\hat{\beta}_2 = 3.217$: average increase in seat change for each additional percentage point increase of RDI, **holding approval fixed**

Least squares with multiple regression

- How do we estimate the coefficients?
- The same exact way as before: minimize prediction error!
- Residuals (aka prediction error) with multiple predictors:

$$Y_i - \widehat{Y}_i = \texttt{seat\_change}_i - \hat{\alpha} - \hat{\beta}_1 \texttt{approval}_i - \hat{\beta}_2 \texttt{rdi\_change}_i$$

- Find the coefficients that minimizes the **sum of the squared residuals**:

$$\text{SSR} = \sum_{i=1}^{n} \hat{\epsilon}_i^2 = (Y_i - \hat{\alpha} - \hat{\beta}_1 X_{i1} - \hat{\beta}_2 X_{i2})^2$$

Model fit with multiple predictors

- R^2 mechanically increases when you add a variables to the regression.
    - But this could be overfitting!!
- Solution: penalize regression models with more variables.
    - Occam's razor: **simpler models are preferred**
- Adjusted R^2: lowers regular R^2 for each additional covariate.
    - If the added covariates doesn't help predict, adjusted R^2 goes down!

Comparing model fits

```r
glance(fit.app) |>
  select(r.squared, adj.r.squared, sigma)
```

```
## # A tibble: 1 x 3
##   r.squared adj.r.squared sigma
##       <dbl>         <dbl> <dbl>
## 1     0.450         0.418  16.9
```

```r
glance(mult.fit) |>
  select(r.squared, adj.r.squared, sigma)
```

```
## # A tibble: 1 x 3
##   r.squared adj.r.squared sigma
##       <dbl>         <dbl> <dbl>
## 1     0.468         0.397  16.7
```