POLSCI 9590: Methods I

OLS Regression I

Dave Armstrong

Videos

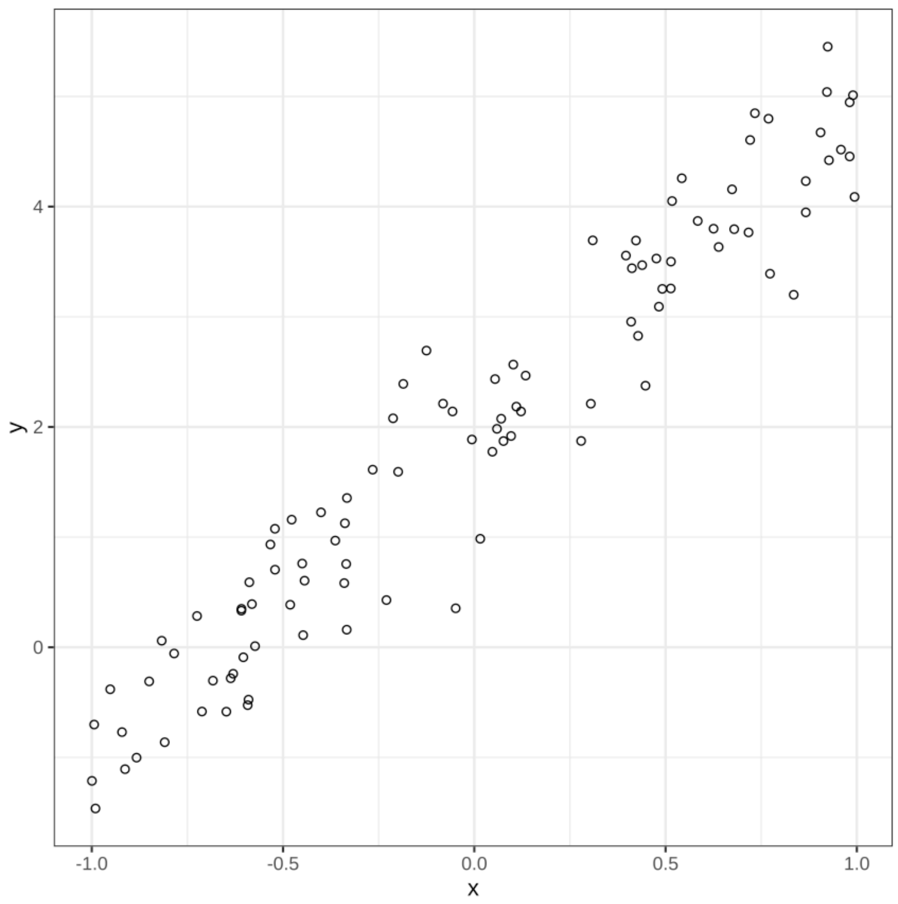
In the videos for today, we learned about:

OLS Regression

- Understanding Least Squares
- Interpreting Model Output

Data

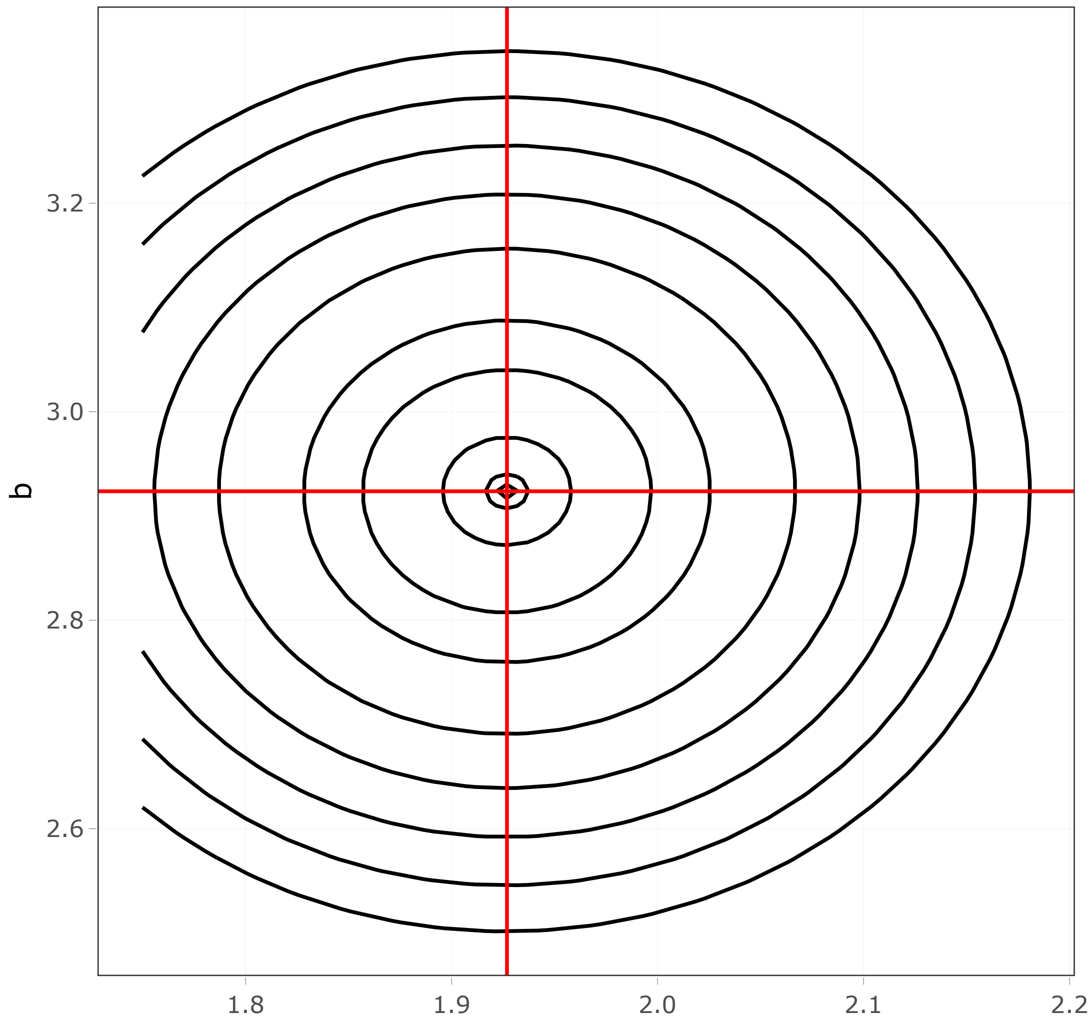
```
dat = rio::import("example1.dta")
library(tidyverse)
library(rio)
ggplot(dat,
       mapping=aes(x=x, y=y)) +
  geom_point(pch=1) +
  theme_bw() +
  theme(aspect.ratio=1)
```



Residual Sums of Squares

```
x <- dat$x
y <- dat$y
a <- seq(1.75, 2.25, length=100)
b <- seq(2.5, 3.5, length=100)
eg <- expand.grid(a=a, b=b)
yhats <- sapply(1:nrow(eg), function(i)eg[i,1] + eg[i,2]*x)
rss <- apply(yhats, 2, function(x)sum((x-y)^2))
eg$rss <- rss
fitmat <- matrix(eg$rss,
                 nrow=length(a),
                 ncol=length(b))
library(plotly)
plot_ly(x = ~a, y=~b, z=~fitmat) %>%
  add_surface(colorbar=list(title="RSS"))
```

Contour Plot of RSS



Model

R Python Stata

```r
mod <- lm(y ~ x)
summary(mod)
```

```
##
## Call:
## lm(formula = y ~ x)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.43079 -0.29872 -0.01345  0.35675  1.13206
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  1.92693    0.05003   38.51   <2e-16 ***
## x            2.92367    0.08321   35.13   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.5003 on 98 degrees of freedom
## Multiple R-squared:  0.9265,    Adjusted R-squared:  0.9257
## F-statistic:  1234 on 1 and 98 DF,  p-value: < 2.2e-16
```

Interpretation

- When `x` is equal to zero, we *expect* `y` to be 1.927.
- For every one-unit increase in `x`, we expect `y` to increase by 2.924.

Other Numbers in the Output: Coefficient Table.

- `Std. Error` is the standard error of the coefficient - how variable is that quantity if repeated sampling.
- `t value` is $\frac{\text{Estimate}}{\text{Std. Error}}$. This t-statistic is on $n - df_{mod}$ degrees of freedom where n is the sample size and df_{mod} is the number of parameters estimated by the model.
- `Pr(>|t|)` is the p-value testing $H_0 : \beta = 0$ against the alternative $H_A : \beta \neq 0$. For a one-sided alternative (assuming you got the direction right), divide this value by 2.

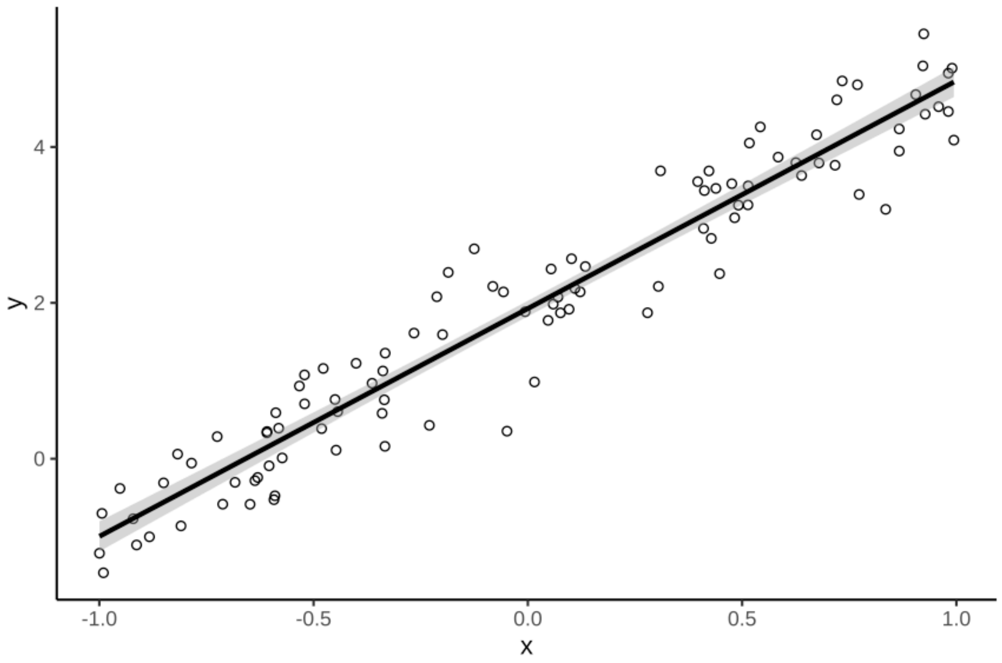
Other Numbers in the Output: Model Fit

- `Residual Standard Error` is $\sqrt{\frac{\sum_{i=1}^{n} e_i^2}{n - df_{mod}}}$ - this is the standard deviation of the residuals. Compare to the standard deviation of `y`.
- `Multiple R-squared` is the proportion of variance explained in `y` by all of the independent variables. It is also the squared correlation between observed `y` values and the predicted `y` values from the model.
- `Adjusted R-squared` more on this next week
- `F-statistic` tests the joint (often called "omnibus") hypothesis: $H_0 : \alpha = \beta = 0$ against the alternative that at least one of the terms doesn't equal zero.

Making a Plot

R | Python | Stata

```
ggplot(dat, aes(x=x, y=y)) +
  geom_point(shape=1) +
  geom_smooth(method="lm",
              col="black") +
  theme_classic() +
  labs(x="x", y="y")
```



Example: Demonstrations and Corruption

R Python Stata

```r
demo <- rio::import("demo.dta")
demo <- rio::factorize(demo)
dmod1 <- lm(demodays ~ corrupt, data=demo)
summary(dmod1)
```

```
##
## Call:
## lm(formula = demodays ~ corrupt, data = demo)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2.1432 -0.9761  0.0017  0.7912  2.5923
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  1.53404    0.29601   5.182 1.83e-06 ***
## corrupt      0.32384    0.05286   6.126 3.98e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.136 on 74 degrees of freedom
## Multiple R-squared:  0.3365,	Adjusted R-squared:  0.3275
## F-statistic: 37.53 on 1 and 74 DF,  p-value: 3.978e-08
```

- For every one-unit increase in corruption, we expect demonstrations to increase by 0.32.
  - This is a statistically significant finding because $p < 0.05$.
- When corruption is zero, we would expect demonstrations to be approximately 1.5.

Is the Result *Substantively* Significant?

One way we could figure this out is by seeing how big of a change in y a standard deviation change in x makes:

```
s <- sd(demo$corrupt, na.rm=TRUE)
s*0.32
```

```
## [1] 0.7943325
```

Now, let's see how big of a change that is for the DV:

```
.79/sd(demo$demodays, na.rm=TRUE)
```

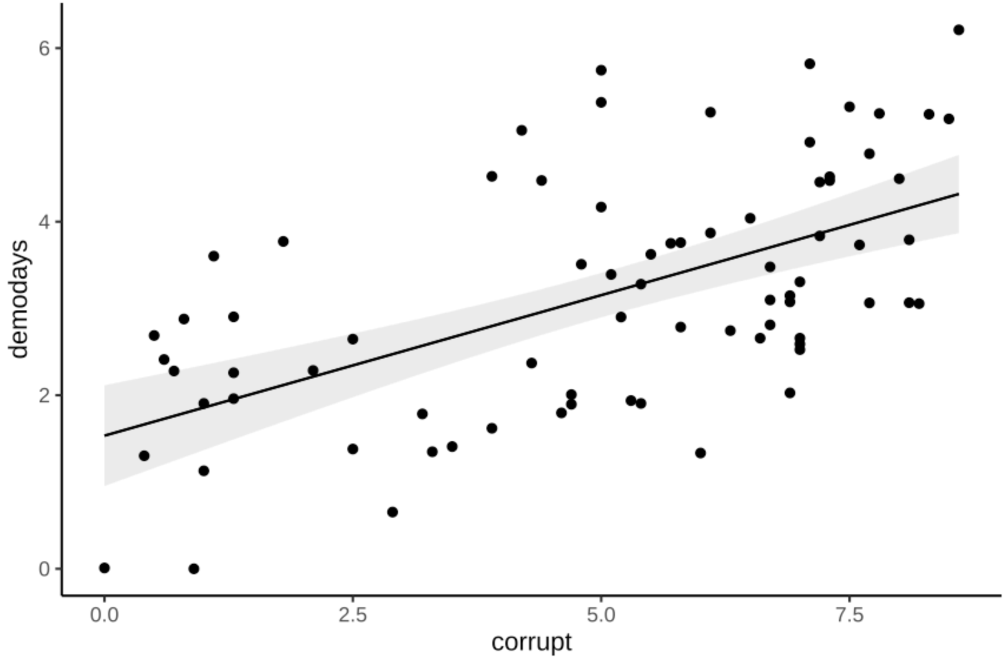
```
## [1] 0.5700975
```

It's about 57% of a standard deviation in terms of the expected change in y. That is pretty good. The bigger this number, the more substantively interesting the result is.

Plot

R Python Stata

```
library(marginaleffects)
plot_predictions(dmod1,
                 condition="corrupt",
                 points=1) +
  theme_classic()
```



Categorical IVs

With categorical IVs, we need to transform them before we can use them in our regression. We need to make a *dummy variable* (one that has only values 0 and 1) for each level of the IV.

id	X
1	a
2	c
3	a
4	b
5	a
6	c
7	c
8	b

id	X = a	X = b	X = c
1	1	0	0
2	0	0	1
3	1	0	0
4	0	1	0
5	1	0	0
6	0	0	1
7	0	0	1
8	0	1	0

Categorical IVs (2)

We can't actually include *all* of the dummies for each category because they are *perfectly collinear*.

- If there are m categories, we can actually represent all of the relevant information in $m-1$ dummy variables.

Let's say that x_a is the dummy variable for $x = a$ and so forth. We know if the categoris are exhaustive and mutually exclusive (which they will be for a single variable):

$$\begin{aligned} x_a + x_b + x_c = 1 \\ x_a = 1 - x_b - x_c \end{aligned}$$

In words,

- if I know the observation is in category b, I know for certain that it is not in category a
- if I know that it is not it categories b or c, then it must be in category a.

Categorical IVs (3)

If we have m categories, we need $m - 1$ dummy variables to represent those categories in our regression model.

id	X
1	a
2	c
3	a
4	b
5	a
6	c
7	c
8	b

id	X = a	X = b	X = c
1	1	0	0
2	0	0	1
3	1	0	0
4	0	1	0
5	1	0	0
6	0	0	1
7	0	0	1
8	0	1	0

id	X = b	X = c
1	0	0
2	0	1
3	0	0
4	1	0
5	0	0
6	0	1
7	0	1
8	1	0

Reference Category

R | Python | Stata

```r
df <- import("cat_example.dta")
df <- factorize(df)
df %>% group_by(x) %>%
  summarise(m = mean(y),
            s = sd(y),
            n = n())
```

```
## # A tibble: 3 × 4
##   x         m     s     n
##   <fct> <dbl> <dbl> <int>
## 1 a      13.5  3.05    10
## 2 b      25.0  2.41    10
## 3 c      25.9  3.21    10
```

```r
xmod <- lm(y ~ x, data=df)
summary(xmod)
```

```
##
## Call:
## lm(formula = y ~ x, data = df)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.7753 -2.4069 -0.4901  2.1859  5.5868
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  13.5394     0.9201  14.715 2.04e-14 ***
## xb           11.4286     1.3012   8.783 2.13e-09 ***
## xc           12.4098     1.3012   9.537 3.88e-10 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.91 on 27 degrees of freedom
## Multiple R-squared:  0.8064,	Adjusted R-squared:  0.792
## F-statistic: 56.22 on 2 and 27 DF,  p-value: 2.365e-10
```

Interpretation

- `(Intercept)`: When $x_b = 0$ and $x_c = 0$ (which we know is when $x_a = 1$), we would expect y to be 13.54. This is significantly different from zero.

- `xb`: The difference between the mean of y when $x = a$ and the mean of y when $x = b$ is 11.43. So, we would expect y to be $13.54 + 11.43 = 24.97$ when $x = b$. This difference is statistically significant.

- `xc`: The difference between the mean of y when $x = a$ and the mean of y when $x = c$ is 12.41. So, we would expect y to be $13.54 + 12.41 = 25.95$ when $x = c$. This difference is statistically significant.

Interpretation: Equation

The model above, suggests the following equation:

$$\hat{y} = 13.54 + 11.43 \times x_b + 12.41 \times x_c$$

So, we could get predictions for each category:

- $x = a \to \hat{y} = 13.54 + 11.43 \times 0 + 12.41 \times 0 = 13.54$
- $x = b \to \hat{y} = 13.54 + 11.43 \times 1 + 12.41 \times 0 = 24.97$
- $x = c \to \hat{y} = 13.54 + 11.43 \times 0 + 12.41 \times 1 = 25.95$

Looking at Comparisons

R   Python   Stata

```r
avg_predictions(xmod, variables="x")
```

```
##
##  x Estimate Std. Error    z Pr(>|z|)     S 2.5 % 97.5 %
##  a     13.5       0.92 14.7   <0.001 160.4  11.7   15.3
##  b     25.0       0.92 27.1   <0.001 536.3  23.2   26.8
##  c     25.9       0.92 28.2   <0.001 578.9  24.1   27.8
##
## Columns: x, estimate, std.error, statistic, p.value, s.value, conf.low, conf.high
## Type:  response
```

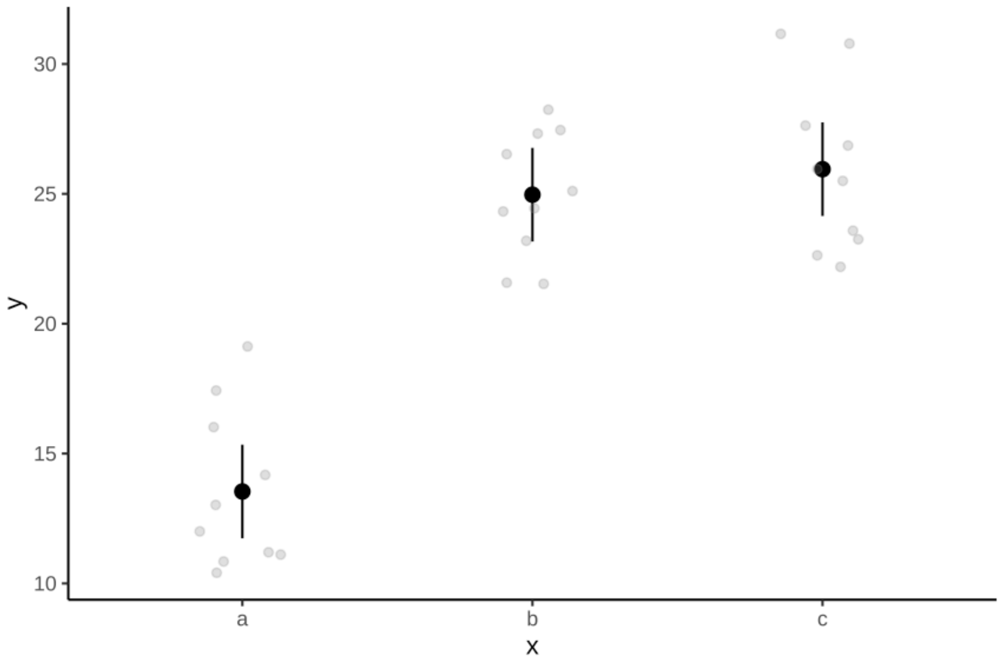
```r
avg_predictions(xmod, variables="x", hypothesis = "pairwise")
```

```
##
##   Term Estimate Std. Error      z Pr(>|z|)    S  2.5 % 97.5 %
##  a - b  -11.429        1.3 -8.783   <0.001 59.1 -13.98  -8.88
##  a - c  -12.410        1.3 -9.537   <0.001 69.2 -14.96  -9.86
##  b - c   -0.981        1.3 -0.754    0.451  1.1  -3.53   1.57
##
## Columns: term, estimate, std.error, statistic, p.value, s.value, conf.low, conf.high
## Type:  response
```

Plotting Predictions

R | Python | Stata

```
plot_predictions(xmod, condition="x") +
  geom_point(data=df, aes(x=x, y=y), position=position_jitter(wic
  theme_classic()
```



Exercises: Demonstrations

Using the demonstrations data, do the following:

Q1

1. Regress `demodays` on one of the other quantitative variables (that's not `corrupt`).
2. Interpret the regression.
3. Plot the regression line.

Q1

1. Regress `demodays` on one of the qualitative variables in the data set.
2. Interpret the regression.
3. Plot the regression line.