Session 2: Regressions

Jennifer Lin

2022-04-05

This week, our aim is to understand how to compute more complex statistics in R, specifically with regressions. Our goals are threefold:

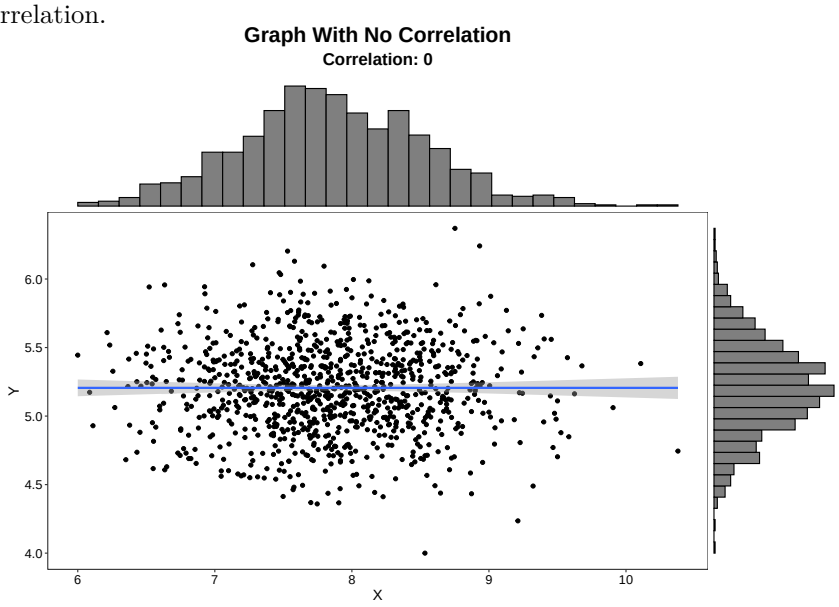
1. Reinforce understanding of the basics of linear regression
2. Learn to conduct regressions in R
3. Learn ways to effectively display regression results

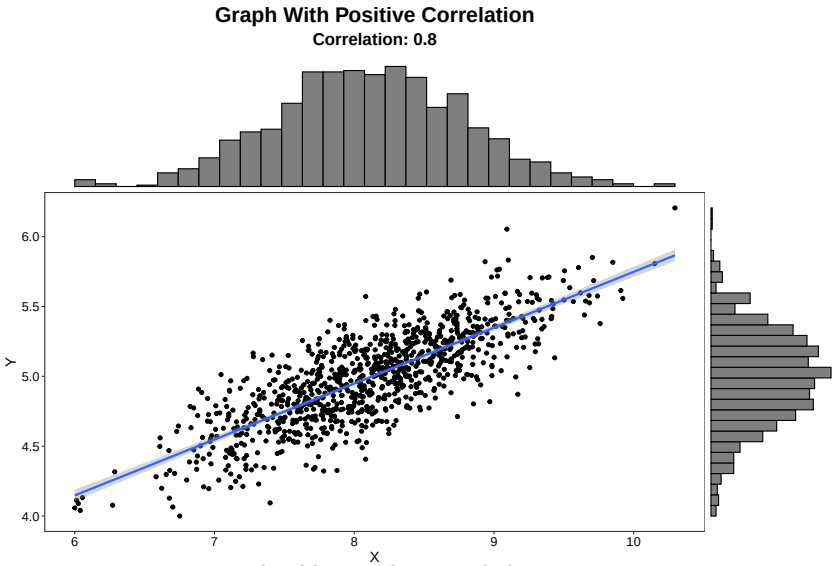
In some statistics classes, you may have learned that

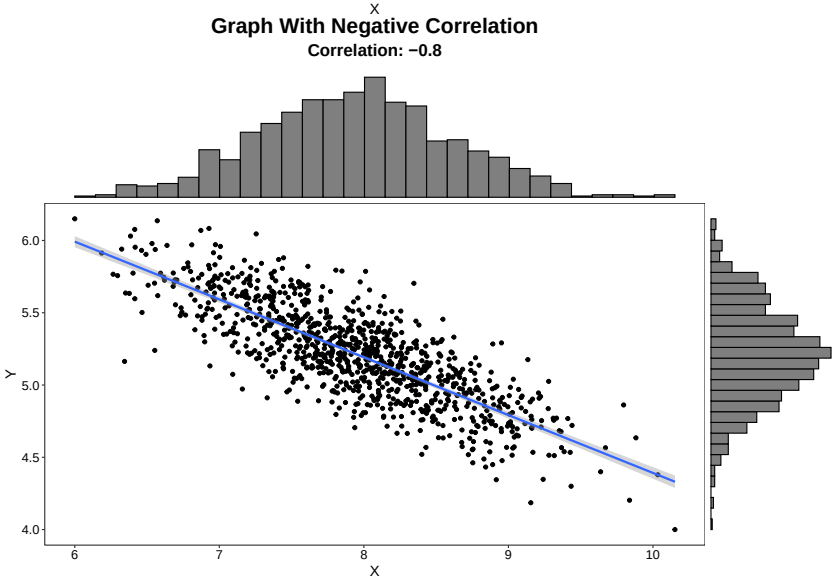
Ordinary **L**east **S**quares (OLS) is the **B**est **L**inear **U**nbiased **E**stimator

For today, I am not as interested in going into what this means specifically. Rather, I am interested in the mechanics that gets us to create an OLS in R. Throughout the session today, bear in mind that regression is a line that sums up the relationship between X and Y.

You might have seen graphs that shows a line running through a series of points. These are regression lines. Below, I have some examples of points that have no correlation, positive correlation and negative correlation.







Behind these plots, and regression, in general, is the slope-intercept model that you may have learned in high school Algebra. If you would recall,

$$Y = mX + b$$

Here, `m` is the slope and `b` is the y-intercept.
In a regression, we have a similar model, where

$$y = \alpha + \beta X + \epsilon$$

or as it is sometimes presented

$$y = a + bX$$

Here, α is the y-intercept and β is the slope.

In some models, we are often only interested in the effects of one X on Y. However, in the real world, things are noisy, and we often need to control for some factors in our regressions to reflect this fact. We can control for any number of other variables that we think might have an effect on the Y variable of interest. Controlling for something is "subtracting out" the effect of X_2 on Y. As a result, the regression model can get more complicated

$$y = \alpha + \beta_1 X_1 + \beta_2 X_2 + \ldots + \beta_k X_k + \epsilon$$

In general, we run regressions to accomplish one of two goals:

1. For **summarizing** data: Calculates one line that describes the relationship between two variables the best and gives us a statistic that summarizes this relationship. This is the *coefficient*.
2. For making **inferences**: The line provides an average for the relationship for X and Y and we can use it to predict Y values from other X values that might not already be in the dataset.

So when we run a regression, how do we interpret it?

Unit	Meaning
Y	Outcome for any X (input)
β	Slope (rise/run)
α	Y-intercept, where X is 0
ϵ	Error term

When interpreting a regression output, our focus is the]*beta* values such that *A 1-unit increase in X is associated with a coefficient-sized (β) change in Y.*[1]

[1] β is also known as the **coefficient**.

Regressions with Survey Data

To put the math into practice, let's look at survey data from the American National Elections Studies. This study is fielded every four years to assess American attitudes towards political matters during Presidential Election years

- The data include items on demographics (race, gender, age) along with an assortment of feeling thermometer variables.
- Feeling Thermometer variables start with `FT_` and all range from 0-100.

```
ANES <- read.csv("ANES_2020.csv")
```

In this coming sections, we will look at examples of the following:

1. Continuous IV and Continuous DV
2. Continuous IV and Continuous DV with continuous controls
3. Continuous DV with categorical IV
4. Continuous DV with categorical IV and categorical controls

Continuous IV and Continuous DV

In this first regression, we look at two continuous variables, feelings towards Dr. Anthony Fauci and feelings towards the CDC, both of which are coded on a numeric 0-100 scale. Here are the variables and the model that results from these considerations.

Unit	Variable	Meaning
X_1	`FT_Fauci`	Feelings towards Anthony Fauci
Y	`FT_CDC`	Feelings towards the CDC

$$Y_{CDC} = \alpha + \beta_{Fauci} X_{Fauci} + \epsilon$$

We can use `lm` to run the regression model. In general, the format for `lm()` goes as follows

```
lm(
  outcome ~ predictor 1 + predictor 2 + ... + predictor n,
  data = dataframe)
```

Here is it in practice.

```
model1 <- lm(
  FT_CDC ~ FT_Fauci,
  data = ANES)
```

To show the output, I can simply type the name of the object which I stored the regression model as. However, as you can see, it is quite minimal.

```
model1
```

```
##
## Call:
## lm(formula = FT_CDC ~ FT_Fauci, data = ANES)
##
## Coefficients:
## (Intercept)     FT_Fauci
##     38.5365       0.4642
```

For a more comprehensive output, we can use `summary()` instead

```
summary(model1)
```

```
##
## Call:
## lm(formula = FT_CDC ~ FT_Fauci, data = ANES)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -84.953 -11.745   0.047  13.972  61.464
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 38.536458   0.556954   69.19   <2e-16 ***
## FT_Fauci     0.464165   0.007464   62.19   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 19.05 on 7151 degrees of freedom
##   (1127 observations deleted due to missingness)
## Multiple R-squared:  0.351,  Adjusted R-squared:  0.3509
## F-statistic:  3868 on 1 and 7151 DF,  p-value: < 2.2e-16
```

By way of interpretation, on average, a one unit increase in feelings towards Dr. Anthony Fauci leads to a 0.46 unit increase in feelings towards the CDC.

Continuous IV and Continuous DV with continuous controls

In this next model, we will build off of the previous model but will add two new variables, which are both continuous and numeric. They are age and the 7-category party identification.

Unit	Variable	Meaning
X_1	`FT_Fauci`	Feelings towards Anthony Fauci
X_2	`age`	Age
X_3	`pid7`	Party ID (7-Category)
Y	`FT_CDC`	Feelings towards the CDC

$$Y_{CDC} = \alpha + \beta_{Fauci} X_{Fauci} + \beta_{age} X_{age} + \beta_{pid7} X_{pid7} + \epsilon$$

When we run the model, we are simply adding things to the previous model.

```
model2 <- lm(
  FT_CDC ~ FT_Fauci + age + pid7,
```

```
  data = ANES)

summary(model2)

##
## Call:
## lm(formula = FT_CDC ~ FT_Fauci + age + pid7, data = ANES)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -85.978 -11.419   0.388  13.914  61.848
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 42.160563   1.101122  38.289  < 2e-16 ***
## FT_Fauci     0.440590   0.009236  47.703  < 2e-16 ***
## age          0.006668   0.013817   0.483    0.629
## pid7        -0.594624   0.121897  -4.878  1.1e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 19.03 on 6870 degrees of freedom
##   (1406 observations deleted due to missingness)
## Multiple R-squared:  0.3539, Adjusted R-squared:  0.3536
## F-statistic:  1254 on 3 and 6870 DF,  p-value: < 2.2e-16
```

Interpretations and Display

From each of the summary tables above, there are some core components of a regression output that should be communicated.

Unit	Meaning
Coefficients	β values for the regression
(Intercept)	Y-intercept, where X is 0
Estimate	Slope
Std. Error	Standard Error
t value	T Statistic – standard score
p-value	Probability of getting result by chance
Signif. Codes	Asterisks that symbolize "rarity"
R^2	Variation in Y explained by X
F	Model performance measure

Of course, not all of these components would be included in a graph, but a neat table is a good place to start. Note that you should

never include raw output in a paper or presentation. There are neat packages that can help you clean up your output. `modelsummary` is one of them.

```
modelsummary(
  list(model1, model2),
  stars = TRUE)
```

	Model 1	Model 2
(Intercept)	38.536***	42.161***
	(0.557)	(1.101)
FT_Fauci	0.464***	0.441***
	(0.007)	(0.009)
age		0.007
		(0.014)
pid7		−0.595***
		(0.122)
Num.Obs.	7153	6874
R2	0.351	0.354
R2 Adj.	0.351	0.354
AIC	62 463.2	60 018.2
BIC	62 483.8	60 052.4
Log.Lik.	−31 228.612	−30 004.090
F	3867.526	1254.197
RMSE	19.05	19.03

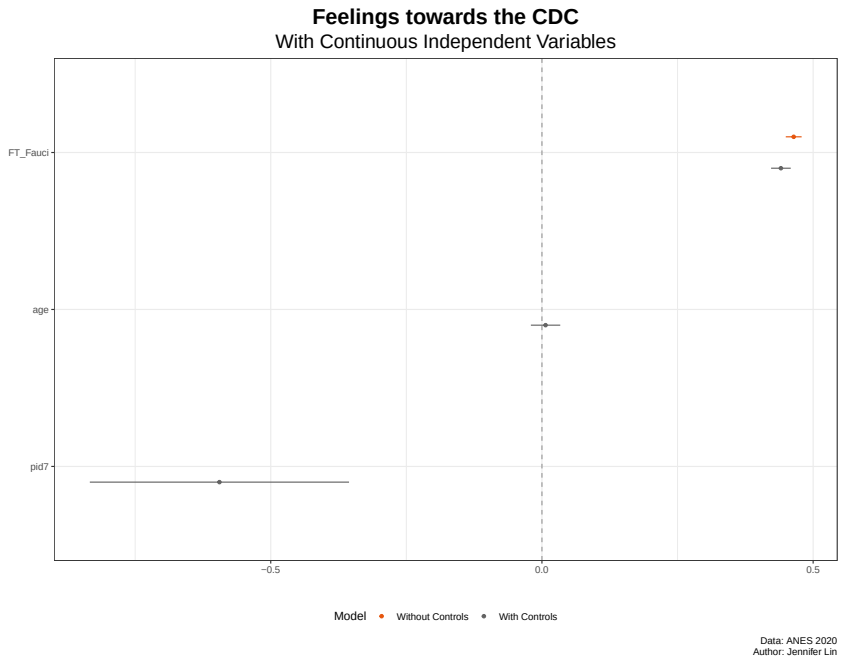
+ p < 0.1, * p < 0.05, ** p < 0.01, *** p < 0.001

We can also plot the coefficients on a graph. The `dotwhizker` package gives us nifty tools to do this, along with the ultimate graphics package `ggplot`[2].

[2] More on `ggplot` later in this class.

```
dwplot(list(model1, model2),
       vline = geom_vline(
         xintercept = 0, colour = "grey60", linetype = 2))+
  scale_color_manual(
    values = c("#636363", "#e6550d"),
    name = "Model",
    labels = c("With Controls", "Without Controls")
  )+
  labs(
    title = "Feelings towards the CDC",
    subtitle = "With Continuous Independent Variables",
    caption = "Data: ANES 2020
```

```
    Author: Jennifer Lin"
  )+
  theme_bw()+
  theme(
    title      = element_text(colour="black"),
    plot.title = element_text(size = 20, hjust = 0.5, face = 'bold'),
    plot.subtitle = element_text(size = 18, hjust = 0.5),
    legend.position = 'bottom'
  )
```



Continuous DV with categorical IV

Now, we move to categorical predictors. For the next two examples, I will be using feelings towards feminists and a selection of predictors. For this first model, let's consider party ID.

Unit	Variable	Meaning
X_1	`PARTY`	Party ID (3-Category)
Y	`FT_Feminists`	Feelings towards Feminists

$$Y_{Feminists} = \alpha + \beta_{PARTY} X_{PARTY} + \epsilon$$

```
model3 <- lm(
  FT_Feminists ~ PARTY,
  data = ANES)
```

```
summary(model3)
```

```
##
## Call:
## lm(formula = FT_Feminists ~ PARTY, data = ANES)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -73.040 -13.729   3.972  13.972  56.271
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)        73.0397     0.3922  186.24   <2e-16 ***
## PARTYIndependent -17.0114     0.8796  -19.34   <2e-16 ***
## PARTYRepublican  -29.3109     0.5730  -51.15   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 22.97 on 7301 degrees of freedom
##   (976 observations deleted due to missingness)
## Multiple R-squared:  0.265,  Adjusted R-squared:  0.2648
## F-statistic:  1316 on 2 and 7301 DF,  p-value: < 2.2e-16
```

Continuous DV with categorical IV and categroical controls

Like the previous section with continuous variables, we can also add controls with categorical variables.

Unit	Variable	Meaning
X_1	`PARTY`	Party ID (3-Category)
X_2	`FEMALE`	Gender: Female
X_3	`MINORITY`	Race: Minority
X_4	`VOTED_2020`	Voted in 2020
Y	`FT_Feminists`	Feelings towards the Feminists

Here, I am adding a series of logical variables, known as *dummy variables* which are coded `TRUE` if yes and `FALSE` if no. Coding variables like this is the equivalent of coding them 0 if no and 1 if yes.

Here is the resulting model:

$$Y_{Feminists} = \alpha + \beta_{PARTY}X_{PARTY} + \beta_{FEMALE}X_{FEMALE} + \beta_{MINORITY}X_{MINORITY} + \beta_{voted}X_{voted} + \epsilon$$

```
model4 <- lm(
```

```
  FT_Feminists ~ PARTY + FEMALE + MINORITY + Voted_2020,
  data = ANES)

summary(model4)
```

```
##
## Call:
## lm(formula = FT_Feminists ~ PARTY + FEMALE + MINORITY + Voted_2020,
##     data = ANES)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -76.437 -14.271   2.909  13.659  62.922
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)        69.2445     0.8187  84.574  < 2e-16 ***
## PARTYIndependent -16.0458     0.8885 -18.059  < 2e-16 ***
## PARTYRepublican  -29.3458     0.5861 -50.065  < 2e-16 ***
## FEMALETRUE         5.0966     0.5378   9.476  < 2e-16 ***
## MINORITYTRUE      -2.8207     0.6223  -4.533 5.92e-06 ***
## Voted_2020TRUE     2.0960     0.7009   2.991  0.00279 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 22.78 on 7298 degrees of freedom
##   (976 observations deleted due to missingness)
## Multiple R-squared:  0.2773, Adjusted R-squared:  0.2768
## F-statistic:   560 on 5 and 7298 DF,  p-value: < 2.2e-16
```

We can display this model and the previous one in a graph and table like we did the models with continuous predictors.

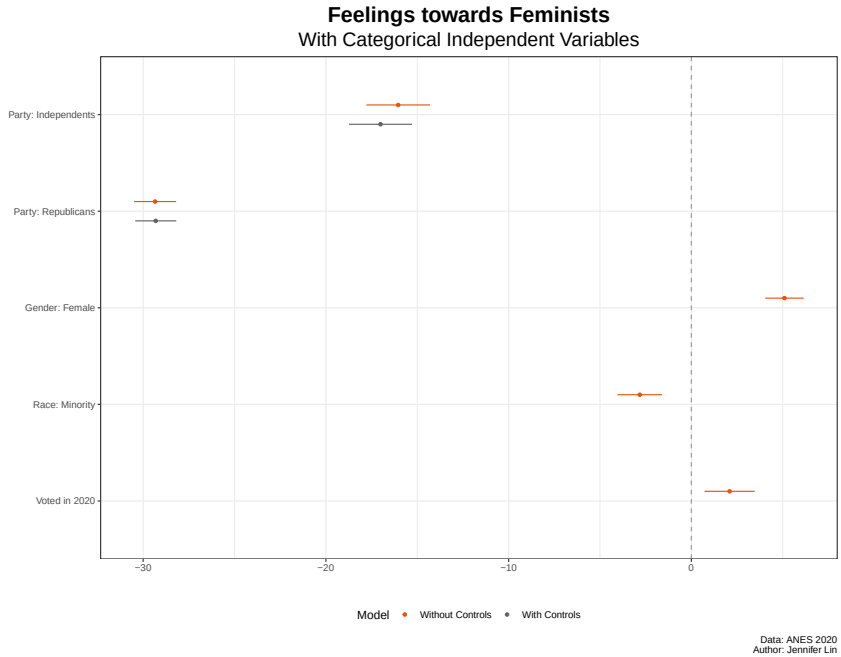
```
modelsummary(
  list(model3, model4),
  stars = TRUE
  )


dwplot(list(model3, model4),
       vline = geom_vline(
         xintercept = 0, colour = "grey60", linetype = 2)) %>%
  relabel_predictors(
    PARTYIndependent = "Party: Independents",
    PARTYRepublican = "Party: Republicans",
    FEMALETRUE = "Gender: Female",
```

	Model 1	Model 2
(Intercept)	73.040***	69.245***
	(0.392)	(0.819)
PARTYIndependent	−17.011***	−16.046***
	(0.880)	(0.889)
PARTYRepublican	−29.311***	−29.346***
	(0.573)	(0.586)
FEMALETRUE		5.097***
		(0.538)
MINORITYTRUE		−2.821***
		(0.622)
Voted_2020TRUE		2.096**
		(0.701)
Num.Obs.	7304	7304
R2	0.265	0.277
R2 Adj.	0.265	0.277
AIC	66 516.1	66 398.9
BIC	66 543.7	66 447.1
Log.Lik.	−33 254.074	−33 192.435
F	1316.045	560.003
RMSE	22.97	22.78

+ $p < 0.1$, * $p < 0.05$, ** $p < 0.01$, *** $p < 0.001$

```
  MINORITYTRUE = "Race: Minority",
  Voted_2020TRUE = "Voted in 2020"
)+
scale_color_manual(
  values = c("#636363", "#e6550d"),
  name = "Model",
  labels = c("With Controls", "Without Controls")
)+
labs(
  title = "Feelings towards Feminists",
  subtitle = "With Categorical Independent Variables",
  caption = "Data: ANES 2020
  Author: Jennifer Lin"
)+
theme_bw()+
theme(
  title      = element_text(colour="black"),
  plot.title = element_text(size = 20, hjust = 0.5, face = 'bold'),
  plot.subtitle = element_text(size = 18, hjust = 0.5),
  legend.position = 'bottom'
)
```



Exercises

1. Pick a continuous variable to serve as your dependent variable and pick a handful of variables that might serve as reasonable indepen-

dent variables. Write their names down.
2. Write down a reasonable regression model you might run.
3. Conduct the regression.
4. Interpret the results. What does the coefficient for your main independent variable tell us about the relationship between that variable and the dependent variable?