Example Paper

Ryan Safner

11/29/2018

Abstract

In this paper, I use fake data I created to demonstrate how to organize your files and manage your workflow effectively.

Introduction

I am managing all of my files (paper, references, data, code, and figures) by creating an `R Project`. This sets my working directory `wd()` to a folder on my computer where the `R Project` is stored. Everything that I put within this folder (including folders, for my Data, my Figures, etc.) is accessible from the same starting directory. I can send the entire project to you, and you can use all of the content easily, since all files are referenced *relative* to the project's directory folder.

Literature Review

> "Here is an example quote from this article." Smith and Jones (2018, 2)

Doe, Gibberish, and Fakerson (2016, 12) says "this short quote", disagreeing with Smith and Jones (2018).

See also the references appear at the end of the document now (we just need to add a `# References` or `# Bibliography` at the end of the file to create a section for them). It lists all the references you cited in alphabetical order.

Data Creation

I first created my random data with the script `01-generate-data.R` in my **Scripts** folder: i.e. `/Scripts/01-generate-data.R`. I reproduced the code here in an `R` chunk in this `R Markdown` document:

```
# You normally would not need this script, but for illustrative purposes, I am going to generate
# my own "fake" data to use for this project, rather than loading some pre-existing dataset

# generate random data
set.seed(1) # make "random" results reproducible

# make dataframe df of random data

shape_types<-c("square", "circle","rectangle", "triangle")
shape<-sample(shape_types,500,replace=TRUE)

df<-data.frame(id=seq(1,500,1), #
               x=rnorm(500,5,1), # 500 random draws from a normal distr with mean 5 and sd 1
               y=rnorm(500,1,10), # 500 random draws from a uniform distr between 1 and 10
               shape=factor(shape))

# a second dataframe with useful information about the same 500 individuals
```

```
states<-c("AK","AL","AR","AZ","CA", "CO", "CT", "DC", "DE", "FL", "GA", "HI",
          "IA", "ID", "IL", "IN", "KS", "KY", "LA", "MA", "MD", "ME", "MI",
          "MN", "MO", "MS", "MT", "NC", "ND", "NE", "NH", "NJ", "NM", "NV",
          "NY", "OH", "OK", "OR", "PA", "RI", "SC", "SD", "TN", "TX", "UT",
          "VA", "VT", "WA", "WI", "WV", "WY")
state<-sample(states,500,replace=TRUE)

df2<-data.frame(id=seq(1,500,1), #
                state=factor(state))

# save data to csv
write.csv(df, file="Data/rawdata1.csv")
write.csv(df2, file="Data/rawdata2.csv")
```

This produces two files, `rawdata1.csv` and `rawdata2.csv`, which I saved in my **Data** folder (i.e. `/Data`).

Data Wrangling

I then loaded the `fake` data I made with my second script in **Scripts**, `Scripts/02-data.cleaning.R`, reproduced below.

```
# Load data
mydf1<-read.csv("Data/rawdata1.csv")
mydf2<-read.csv("Data/rawdata2.csv")

# join data
library("dplyr")
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
clean_data<-inner_join(mydf1, mydf2, key=id)
```

```
## Joining, by = c("X", "id")
```

```
# save as clean data
write.csv(clean_data, file="Data/clean_data.csv")
```

This takes my separate data frames and joins them together into a single `data.frame`, which I saved for later use as `clean_data.csv`.

Summary Statistics

```
library("stargazer")
```

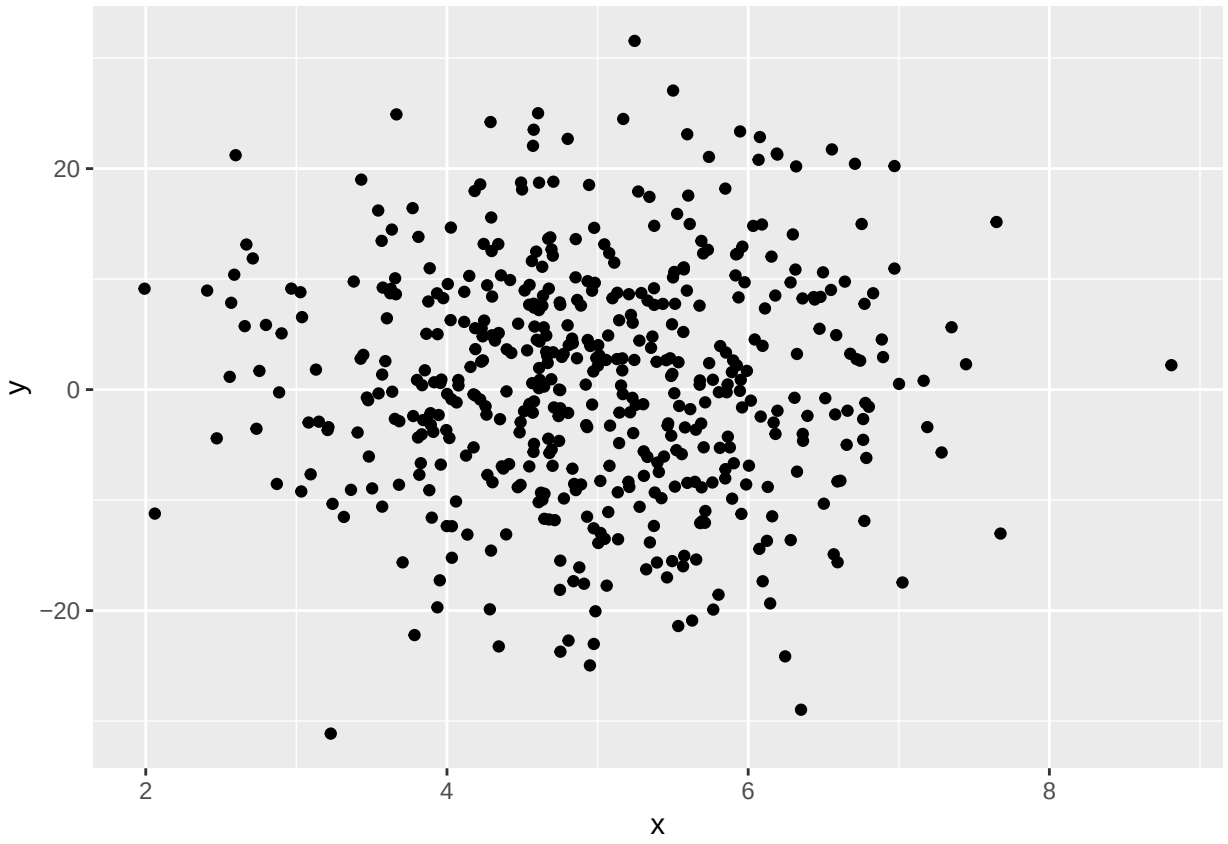
```
##
## Please cite as:
##  Hlavac, Marek (2018). stargazer: Well-Formatted Regression and Summary Statistics Tables.
##  R package version 5.2.2. https://CRAN.R-project.org/package=stargazer
```

```
stats<-subset(clean_data,select=c("x","y"))

stargazer(data=stats, type="latex",  digits=2, summary.stat = c("n", "mean", "sd", "min", "p25", "media
```

Statistic	N	Mean	St. Dev.	Min	Pctl(25)	Median	Pctl(75)	Max
x	500	4.97	1.06	1.99	4.27	4.93	5.69	8.81
y	500	0.76	10.44	−31.13	−6.61	0.85	8.25	31.56

Plots

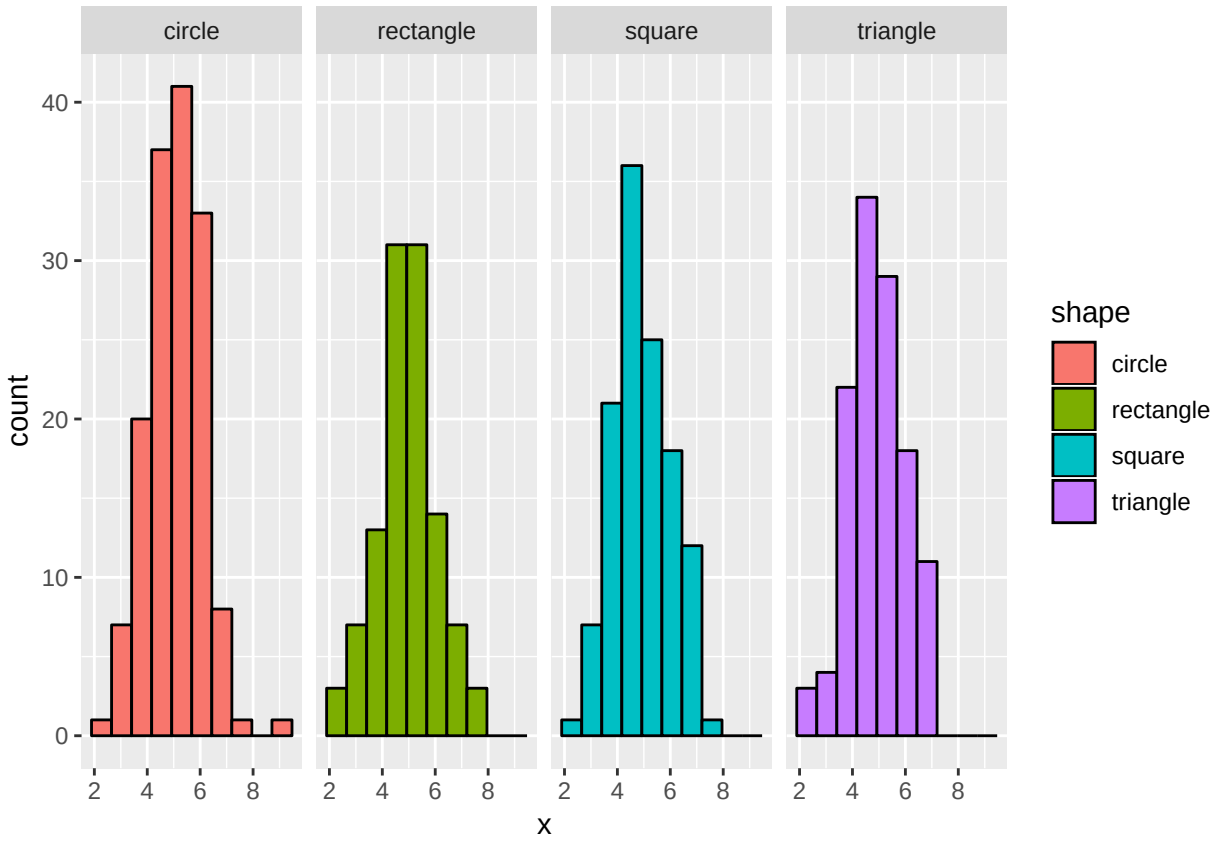
```
# load packages
library("ggplot2")

# load data

clean_data<-read.csv("Data/clean_data.csv")

# scatterplot

s1<-ggplot(data = clean_data, aes(x=x, y=y))+
  geom_point()
s1
```



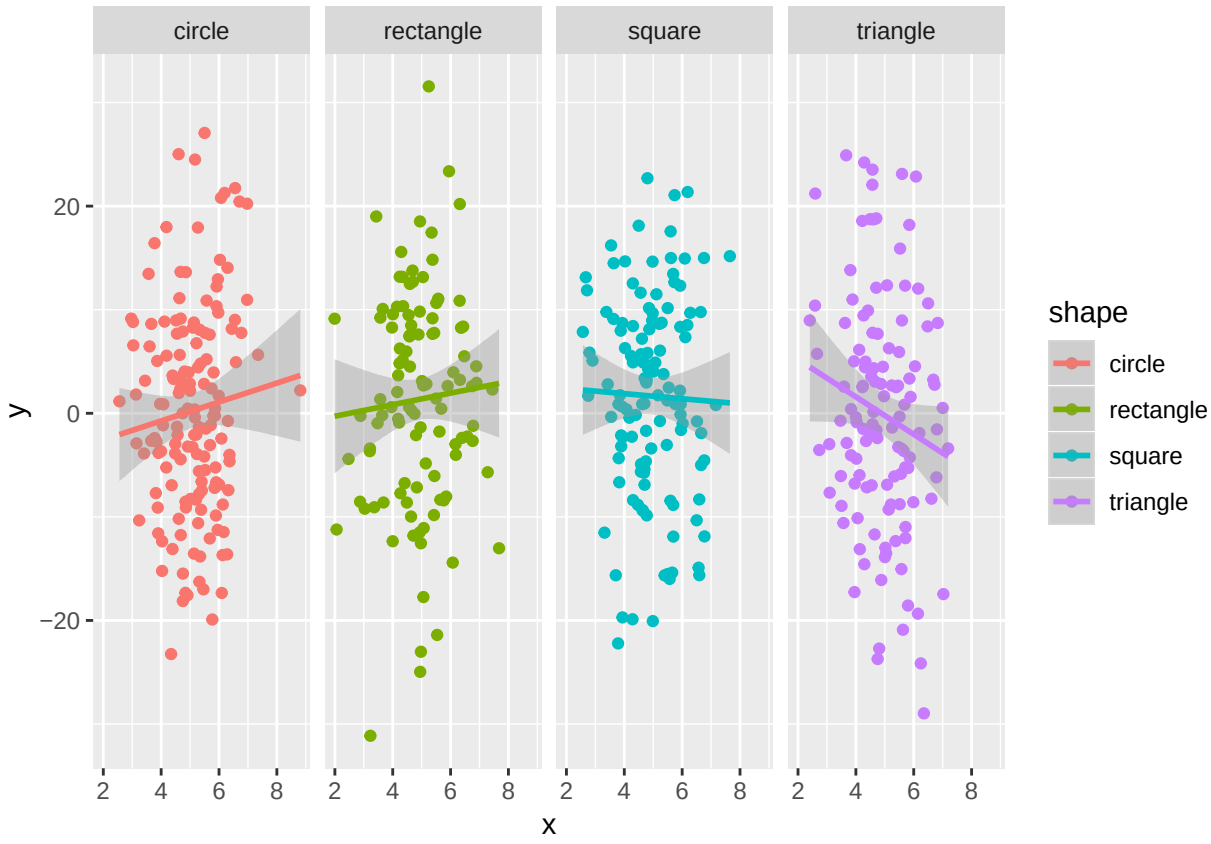

```
ggsave("Figures/fig1", device="eps")
```

```
## Saving 6.5 x 4.5 in image
```

```
h1<-ggplot(data = clean_data, aes(x=x, fill=shape))+
  geom_histogram(color="black", bins=10)+
  facet_grid(cols=vars(shape))
h1
```




```
s2<-ggplot(data = clean_data, aes(x=x, y=y, color=shape))+
  geom_point()+
  geom_smooth(method="lm")+
  facet_grid(cols=vars(shape))
s2
```

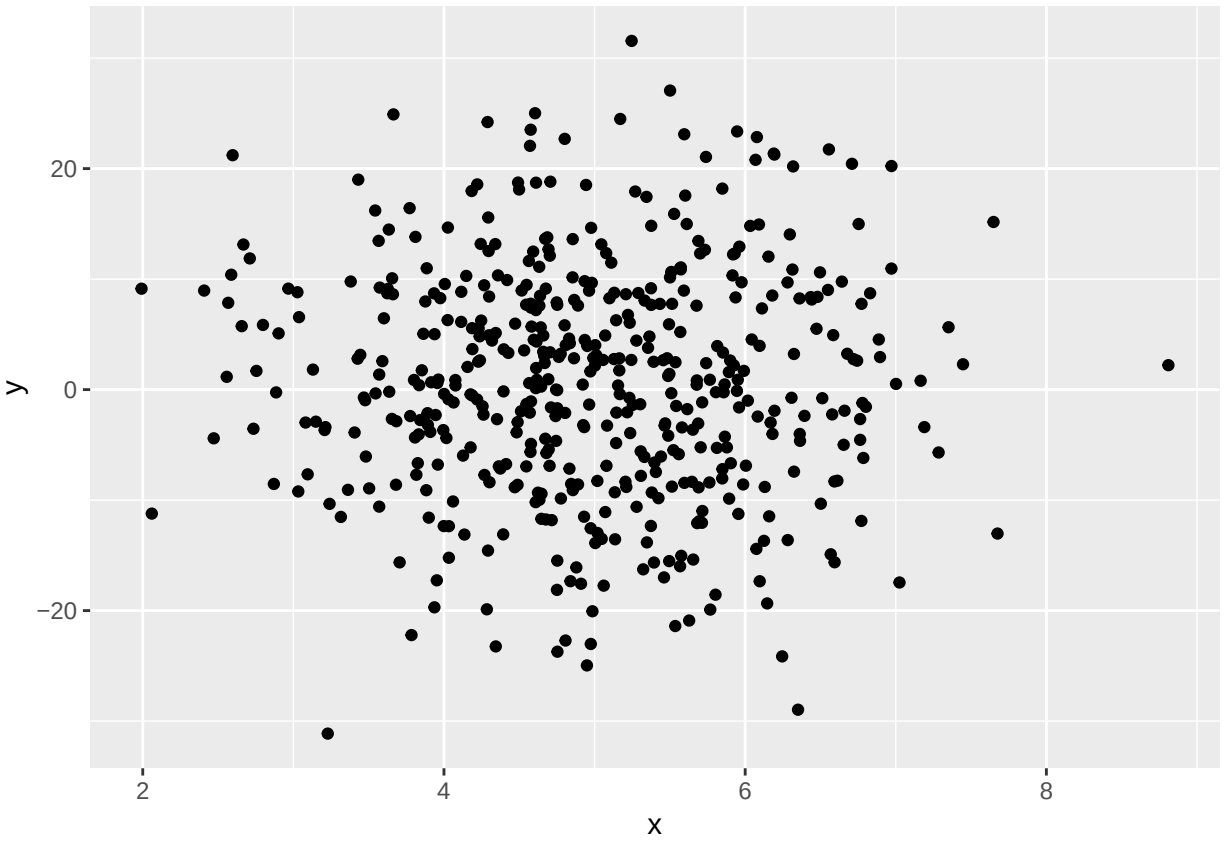



```
ggsave("Figures/fig2", device="eps")
```

```
## Saving 6.5 x 4.5 in image
```

```
## Warning in grid.Call.graphics(C_polygon, x$x, x$y, index): semi-
## transparency is not supported on this device: reported only once per page
```

```
s3<-ggplot(data = clean_data, aes(x=x, y=y))+
  geom_point()
s3
```




```
ggsave("Figures/fig3", device="eps")
```

```
## Saving 6.5 x 4.5 in image
```

```
#
```

```
reg1<-lm(y~x, data=clean_data)
summary(reg1)
```

Call: lm(formula = y ~ x, data = clean_data)

Residuals: Min 1Q Median 3Q Max -32.114 -7.299 0.022 7.541 30.838

Coefficients: Estimate Std. Error t value Pr(>|t|) (Intercept) 1.4016 2.2479 0.624 0.533 x -0.1300 0.4428 -0.294 0.769

Residual standard error: 10.45 on 498 degrees of freedom Multiple R-squared: 0.000173, Adjusted R-squared: -0.001835 F-statistic: 0.08616 on 1 and 498 DF, p-value: 0.7692

```
reg2<-lm(y~x+shape, data=clean_data)
summary(reg2)
```

Call: lm(formula = y ~ x + shape, data = clean_data)

Residuals: Min 1Q Median 3Q Max -32.662 -6.955 -0.164 7.218 30.250

Coefficients: Estimate Std. Error t value Pr(>|t|) (Intercept) 0.8090 2.4150 0.335 0.738 x -0.1099 0.4442 -0.248 0.805 shaperectangle 1.0756 1.3199 0.815 0.415 shapesquare 1.4012 1.2815 1.093 0.275 shapetriangle -0.3318 1.2818 -0.259 0.796

Residual standard error: 10.46 on 495 degrees of freedom Multiple R-squared: 0.004864, Adjusted R-squared: -0.003177 F-statistic: 0.6049 on 4 and 495 DF, p-value: 0.6593

```
reg3<-lm(y~x+shape+state, data=clean_data)
summary(reg3)
```

Call: lm(formula = y ~ x + shape + state, data = clean_data)

Residuals: Min 1Q Median 3Q Max -27.6314 -6.9783 -0.1014 6.6692 30.5991

Coefficients: Estimate Std. Error t value Pr(>|t|)
(Intercept) -2.31210 4.52145 -0.511 0.6094
x 0.01894 0.47164 0.040 0.9680
shaperectangle 0.83310 1.39970 0.595 0.5520
shapesquare 0.72491 1.37942 0.526 0.5995
shapetriangle -0.60904 1.36798 -0.445 0.6564
stateAL -3.55173 5.48245 -0.648 0.5174
stateAR 7.06940 5.15435 1.372 0.1709
stateAZ -3.37505 5.15498 -0.655 0.5130
stateCA 4.99132 5.01861 0.995 0.3205
stateCO 3.80758 4.75489 0.801 0.4237
stateCT 2.60907 4.76916 0.547 0.5846
stateDC 5.25705 4.92840 1.067 0.2867
stateDE 2.33952 6.03577 0.388 0.6985
stateFL 3.83136 4.71187 0.813 0.4166
stateGA -1.59294 4.66476 -0.341 0.7329
stateHI 5.58282 5.02335 1.111 0.2670
stateIA 2.33798 4.69743 0.498 0.6189
stateID -1.07841 5.32279 -0.203 0.8395
stateIL 0.89104 4.63915 0.192 0.8478
stateIN 9.67361 5.77692 1.675 0.0947 . stateKS -0.97596 5.71849 -0.171 0.8646
stateKY 2.52638 4.60085 0.549 0.5832
stateLA 1.51014 6.04194 0.250 0.8027
stateMA 5.11136 5.03054 1.016 0.3102
stateMD 4.06125 4.77956 0.850 0.3959
stateME -3.80143 6.07647 -0.626 0.5319
stateMI 4.46636 4.92777 0.906 0.3652
stateMN 7.76509 4.91201 1.581 0.1146
stateMO 3.21416 5.02126 0.640 0.5224
stateMS 0.93808 5.72914 0.164 0.8700
stateMT 2.03317 4.72454 0.430 0.6672
stateNC 2.71308 4.82683 0.562 0.5743
stateND 0.89830 5.15534 0.174 0.8618
stateNE 2.49000 4.84943 0.513 0.6079
stateNH 4.68952 5.17762 0.906 0.3656
stateNJ 4.85446 4.77994 1.016 0.3104
stateNM 2.80148 5.05794 0.554 0.5799
stateNV -5.33912 7.20308 -0.741 0.4589
stateNY 5.16232 4.84103 1.066 0.2868
stateOH 5.60256 6.05106 0.926 0.3550
stateOK 4.70911 5.30066 0.888 0.3748
stateOR -3.66738 5.49654 -0.667 0.5050
statePA -0.97506 5.16530 -0.189 0.8504
stateRI 8.52005 5.03071 1.694 0.0910 . stateSC 6.58968 4.91383 1.341 0.1806
stateSD 4.75076 6.09679 0.779 0.4363
stateTN 5.22506 5.29623 0.987 0.3244
stateTX 3.08806 4.91829 0.628 0.5304
stateUT 5.53742 4.77009 1.161 0.2463

stateVA 3.83155 5.15992 0.743 0.4581
stateVT -2.08268 5.18584 -0.402 0.6882
stateWA 0.25279 5.14065 0.049 0.9608
stateWI 0.49125 4.84918 0.101 0.9194
stateWV 1.10369 5.29683 0.208 0.8350
stateWY 1.60161 4.84835 0.330 0.7413
— Signif. codes: 0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 10.57 on 445 degrees of freedom Multiple R-squared: 0.08604, Adjusted R-squared: -0.02487 F-statistic: 0.7758 on 54 and 445 DF, p-value: 0.8749

```
library("stargazer")

stargazer(reg1, reg2, reg3, omit=c(state), column.labels = c("","","With State-effects"), type="latex",
```

	Dependent variable:		
	y		
			With State-effects
	(1)	(2)	(3)
x	−0.130	−0.110	0.019
	(0.443)	(0.444)	(0.472)
shaperectangle		1.076	0.833
		(1.320)	(1.400)
shapesquare		1.401	0.725
		(1.282)	(1.379)
shapetriangle		−0.332	−0.609
		(1.282)	(1.368)
Constant	1.402	0.809	−2.312
	(2.248)	(2.415)	(4.521)
Observations	500	500	500
R^2	0.0002	0.005	0.086
Adjusted R^2	−0.002	−0.003	−0.025
Residual Std. Error	10.449 (df = 498)	10.456 (df = 495)	10.568 (df = 445)
F Statistic	0.086 (df = 1; 498)	0.605 (df = 4; 495)	0.776 (df = 54; 445)
Note:			*p<0.1; **p<0.05; ***p<0.01

Doe, John, Erica Gibberish, and Frank Fakerson. 2016. *How Did This Get Published?* New York, NY: Sketchy Publishing, Inc.

Smith, Robert, and Anne Jones. 2018. “A Test Article.” *Fake Articles Quarterly* 12 (2): 1–21.