Exploratory and Confirmatory Data Analysis

dapr2

dapr2 Teaching Team

Department of Psychology
The University of Edinburgh

AY 2022-2023

Learning Objectives

- Some exploratory situations
  - I have a hypothesis but I'm not quite sure how to test it with the variables I have
  - I think some variables could be relevant to a DV but I'm not sure which ones
- Working through an example

The Issues:

- We're often interested in the relationship between variables but don't have clear predictions about how they're related
  - For example, I might be interested in why some tweets go viral and others don't
- The number of possible predictors related to this question is huge and it's not obvious which ones will be most important
  - Includes a photo? Humor? Many, many possible predictors

The Issues:

- Sometimes I might have a hypothesis, but it could be tested in multiple ways and I'm not sure how best to test it.
  - For example, I think a tweet including a photo will be retweeted more. What kind of photo though? Any photo? Happy photos?

Exploratory Analyses

- The context I've describe above is a case of exploratory analysis.
- Exploratory analyses can take many forms, but they share in common the fact that you, the researcher, don't have extremely specific predictions about the relationship between your independent variable and your dependent variable
- The exploratory phase of data analysis is a great way to learn a lot about your data, but you also need to be on-guard that you don't think you've detected signal when you've actually detected noise.
    - More on this in a bit

Exploratory analyses done wrong

- "You cannot find your starting hypothesis in your final results. It makes the stats go all wonky." - Ben Goldacre
- If you treat an exploratory result as if you had that hypothesis from the start, then it can cause problems. You will trick yourself.

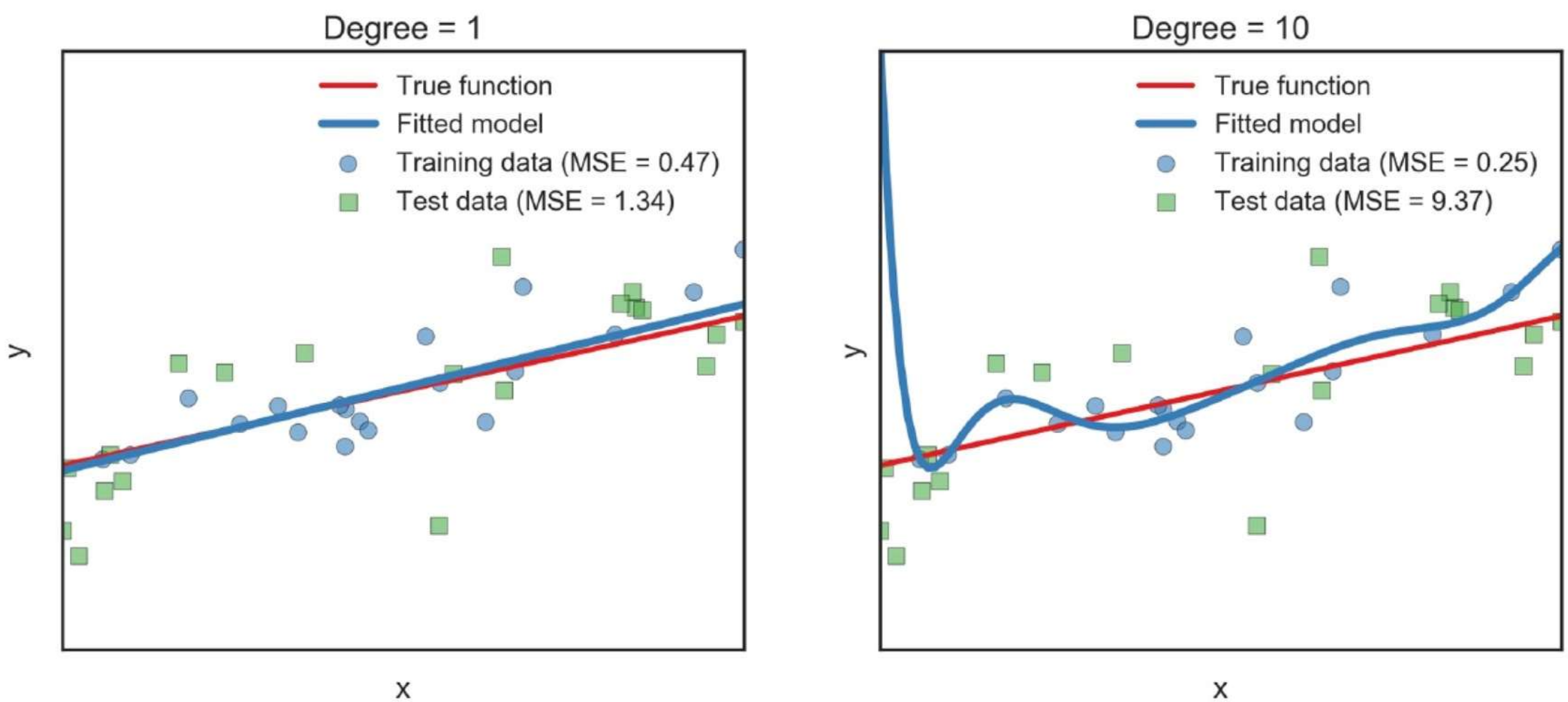
Exploratory analyses done wrong

- Exploratory analyses done poorly: Measure many variables (gender, personality characteristics, age, etc.) and only report those that yield a statistically significant result (stargazing)
    - Include in your paper only those experiments that produced the desired outcome,
    - Treat experiments or initial analyses that didn't turn out favorably as "pilots"
- Why is this problematic?
- In a word, this will lead to model overfitting.

Overfitting

- Overfitting is the tendency for statistical models to mistakenly fit sample-specific noise as if it were signal
  - In a sample of N = 50 with 20 uncorrelated predictors, each correlated 0.1 with the DV, the observed (and overfitted) R2 value will, on average, be 0.45
  - Gives the impression that one could predict values of the DV rather successfully.
  - True R2 in this situation is only 0.07. Even worse, the average out-of-sample test value of R2 is only 0.02!

R Squared is very optimistic



Overfitting continued

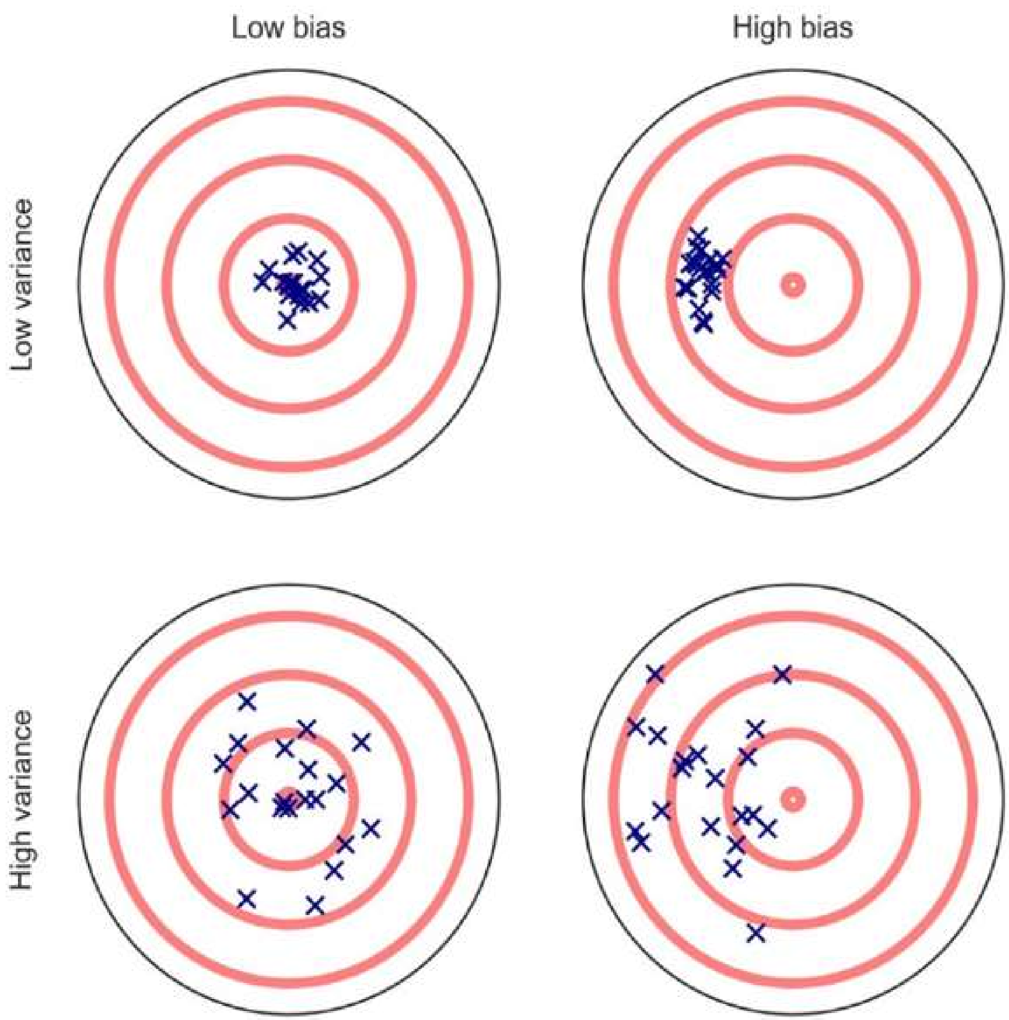
- Don't trust estimates of model performance if those estimates are obtained by "testing" the model on the same data on which it was originally trained
- We need a method for doing exploratory analyses without tricking ourselves.

An aside: The link between p-hacking and overfitting

- p-hacking is a special case of overfitting. Specifically, it is procedural overfitting (Yarkoni and Westfall, 2018). It takes place prior to (or in parallel with) model estimation
  - For example, during data cleaning, model selection, or choosing which analyses to report

We often want to explore though. How do we do that in a principled way?

- First, need to distinguish bias and variance
  - Bias: the tendency for a model to consistently produce answers that are wrong in a particular direction (e.g., estimates that are consistently too high).
  - Variance: the extent to which a model's fitted parameters will tend to deviate from their central tendency across different datasets.


Low bias
High bias
Low variance
High variance

Bias-Variance Tradeoff

- Liberal, flexible data analysis is a low-bias but high-variance approach
  - Almost any pattern in data can potentially be detected, at the cost of a high rate of spurious identifications
  - This is exploratory data analysis
- An approach that favors strict adherence to a fixed set of procedures as a high bias, low-variance approach
  - Only a limited range of patterns can be identified, but the risk of pattern hallucination is low
  - This is confirmatory data analysis

What to do? Consider lots of possibilities but focus on minimizing prediction error (no stargazing!)

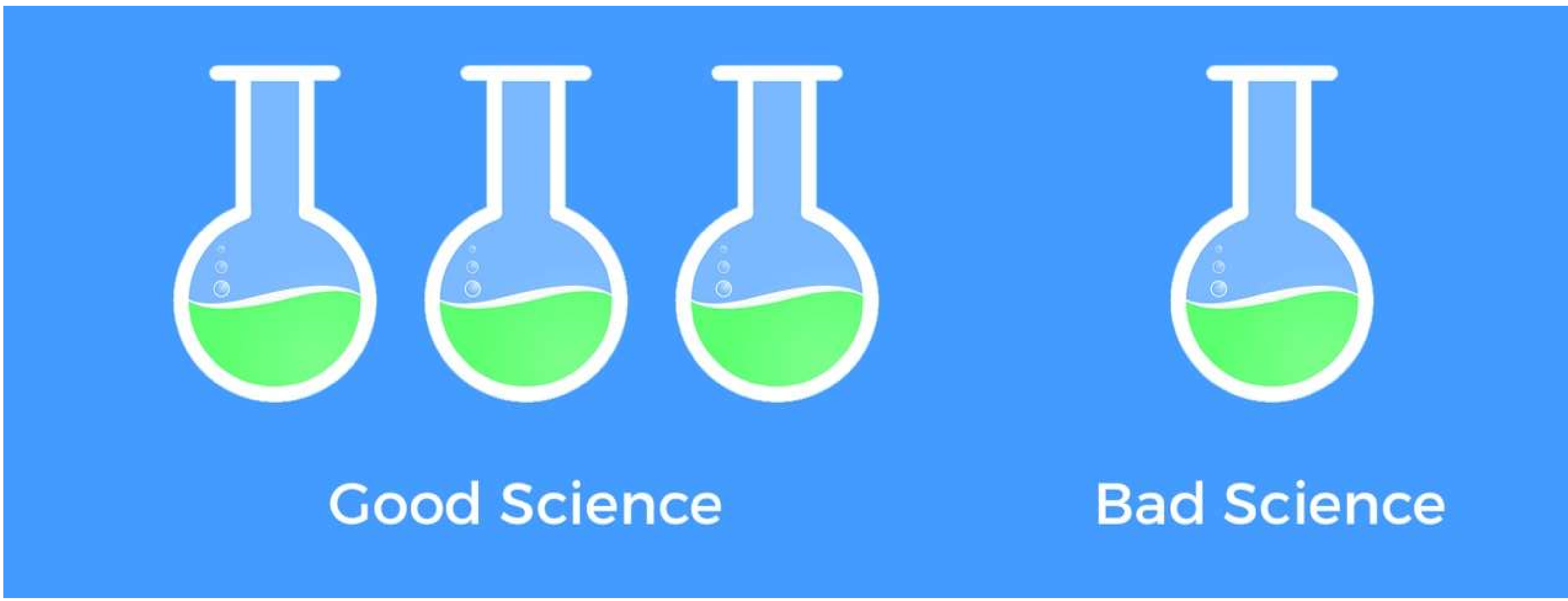
- What's required to do exploratory data analysis that gives you *information* on which you can do confirmatory research?
  - Datasets large enough to support training models
  - Accurately estimate prediction error to assess performance and improve model
  - Exert control over the bias-variance tradeoff when appropriate

Cross-validation

- All of these are directly related to cross-validation and replication
  - To assess our models, we need to quantify out-of-sample prediction error
  - Cross-validation: vaious techniques involved in training and testing a model on different samples of data

Cross-validation

- Canonical cross-validation
    - The classical replication study, where a model is trained on one dataset and then tested on a completely independent dataset. Most typical of experimental research. Less common in correlational research.

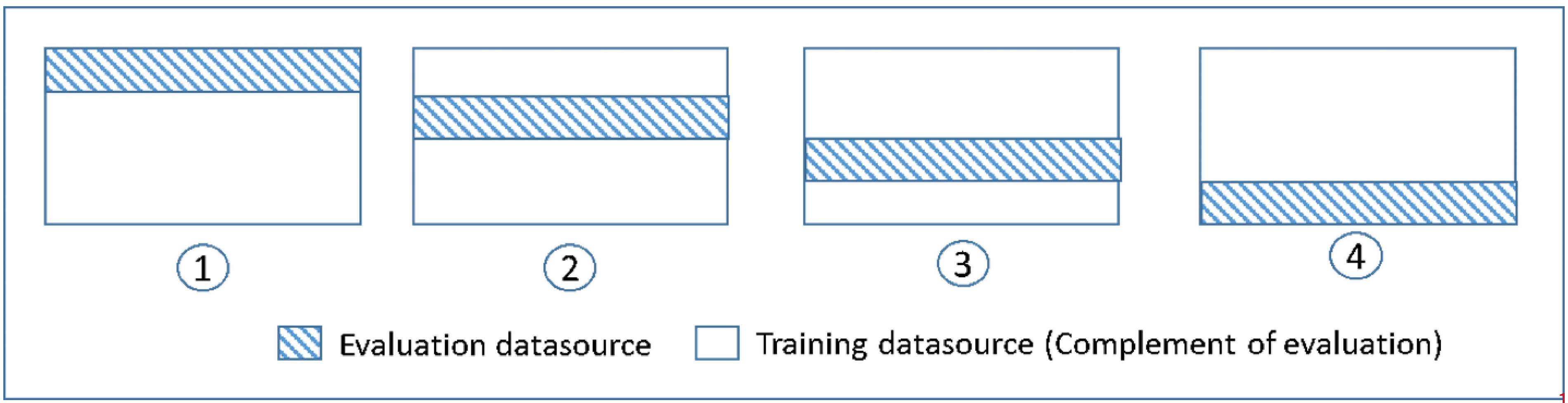


Cross-validation

- Sometimes you can't collect more data though
  - One giant study you want to analyze was run once
  - There is a limited population
  - Limited funds to collect more data

Recycle your dataset.

- Don't assign each observation exclusively to either the training or the test datatsets - do both!
  - Known as K-Folding where *K* is the number of folds
  - In one "fold" (essentially a subset of your data), one half of the data is used for training and the other half is for testing
  - In a second fold, the datasets are reversed, and the training set and test sets exchange roles.
  - Typical number of folds is 10




FUTURE

Read data into R

```r
Exploredf <- read_csv("df1.csv")
```

```
## New names:
## Rows: 5000 Columns: 7
## -- Column specification
## -------------------------------------------------------- Delimiter: "," chr
## (3): screen_name, media_type, text dbl (4): ...1, ...2, favorite_count,
## retweet_count
## i Use `spec()` to retrieve the full column specification for this data. i
## Specify the column types or set `show_col_types = FALSE` to quiet this message.
## * `` -> `...1`
## * `...1` -> `...2`
```

```r
head(Exploredf)
```

```
## # A tibble: 6 x 7
##    ...1  ...2 screen_name favorite_count retweet_count media_type text
##   <dbl> <dbl> <chr>                <dbl>         <dbl> <chr>      <chr>
## 1     1 78462 DogsTrust               41            11 Photo      "With our #Ca~
## 2     2 78731 DogsTrust               28            11 Photo      "Running acro~
## 3     3 48668 Greenpeace             466           166 Nophoto    "When billion~
```

```
Exploredf1 <- Exploredf%>%
  mutate(Postnumber = 1:n())%>%
  select(-c(...1))

sentiment <- Exploredf1 %>%
  unnest_tokens(output = "word", input = "text")
```

```
sentiment_dictionary1 <- get_sentiments("bing")
head(sentiment_dictionary1)
```

```
## # A tibble: 6 x 2
##   word        sentiment
##   <chr>       <chr>
## 1 2-faces     negative
## 2 abnormal    negative
## 3 abolish     negative
## 4 abominable  negative
## 5 abominably  negative
## 6 abominate   negative
```

```
sentiment_dictionary2 <- get_sentiments("afinn")
head(sentiment_dictionary2)
```

```
## # A tibble: 6 x 2
##   word       value
##   <chr>      <dbl>
## 1 abandon       -2
## 2 abandoned     -2
## 3 abandons      -2
## 4 abducted      -2
## 5 abduction     -2
## 6 abductions    -2
```

```
sentiment_dictionary3 <- get_sentiments("nrc")
head(sentiment_dictionary3)
```

```
## # A tibble: 6 x 2
##   word      sentiment
##   <chr>     <chr>
## 1 abacus    trust
## 2 abandon   fear
## 3 abandon   negative
## 4 abandon   sadness
## 5 abandoned anger
## 6 abandoned fear
```

```r
sentiment1df <- merge(sentiment, sentiment_dictionary1, by = "word")
head(sentiment1df)
```

```
##          word    ...2  screen_name favorite_count retweet_count media_type
## 1 abominably  72496         peta             78            34    Nophoto
## 2    absence  85636          WWF            220            70    Nophoto
## 3  abundance 122417 AWF_Official             92            24      Photo
## 4  abundance  73701         peta              0             0    Nophoto
## 5  abundance  93624    Defenders             53            23    Nophoto
## 6  abundance   1507       oceana            126            23      Photo
##   Postnumber sentiment
## 1       3665  negative
## 2       2455  negative
## 3       2377  positive
## 4       2664  positive
## 5       1056  positive
## 6        407  positive
```

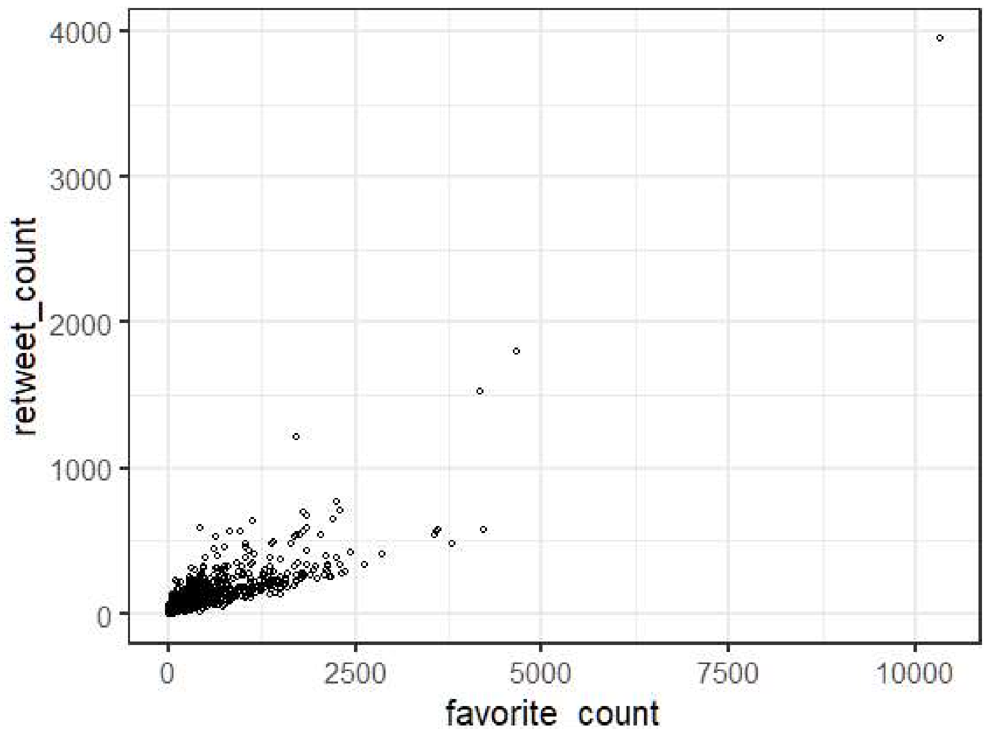
```r
library(summarytools)
```

```
##
## Attaching package: 'summarytools'

## The following object is masked from 'package:tibble':
##
##     view
```

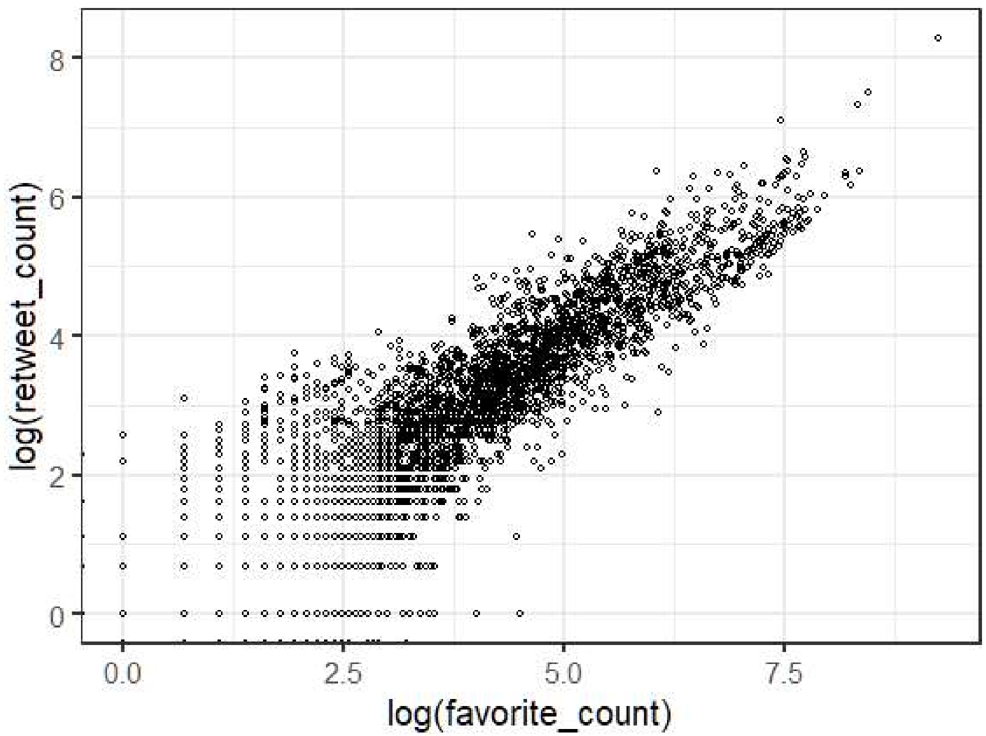
```r
view(dfSummary(sentiment1df))
```

```
## Switching method to 'browser'

## Output file written: C:\Users\zachs\AppData\Local\Temp\RtmpcBRvG5\file17f0762944b1.html
```

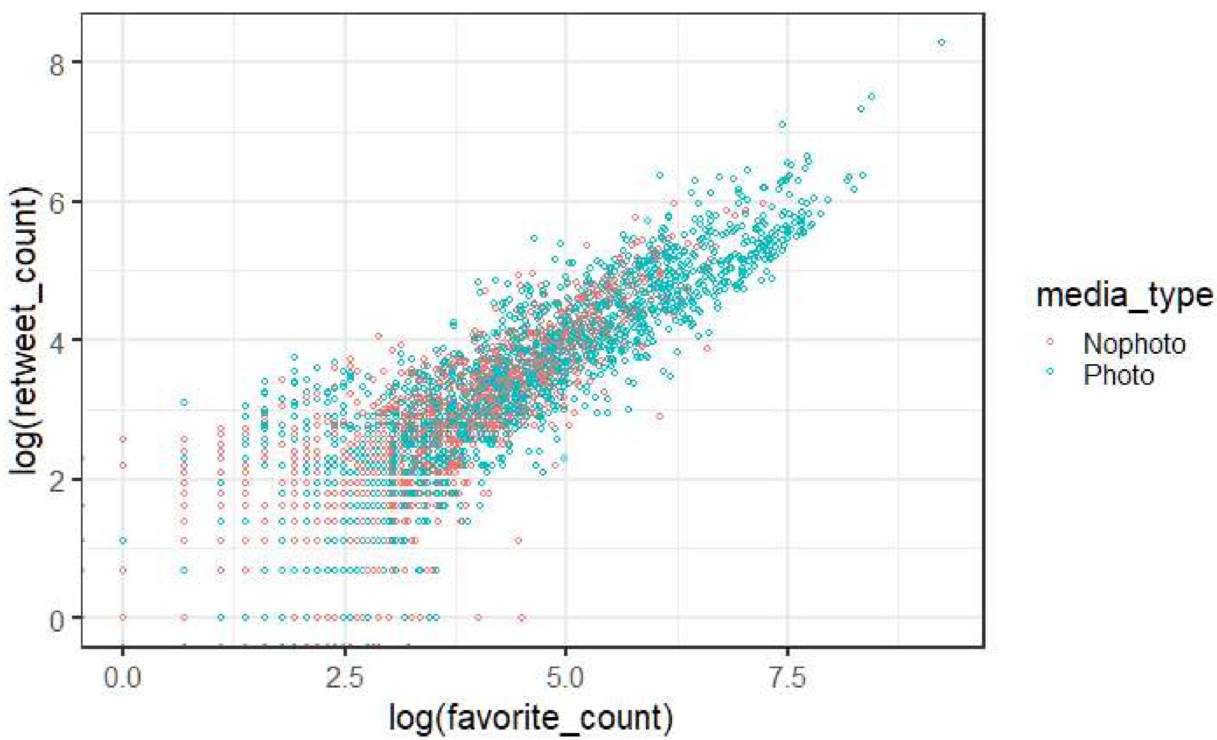
```
ggplot(Exploredf1)+
  geom_point(aes(y=retweet_count, x = favorite_count), shape=1)+
  theme_bw(20)
```



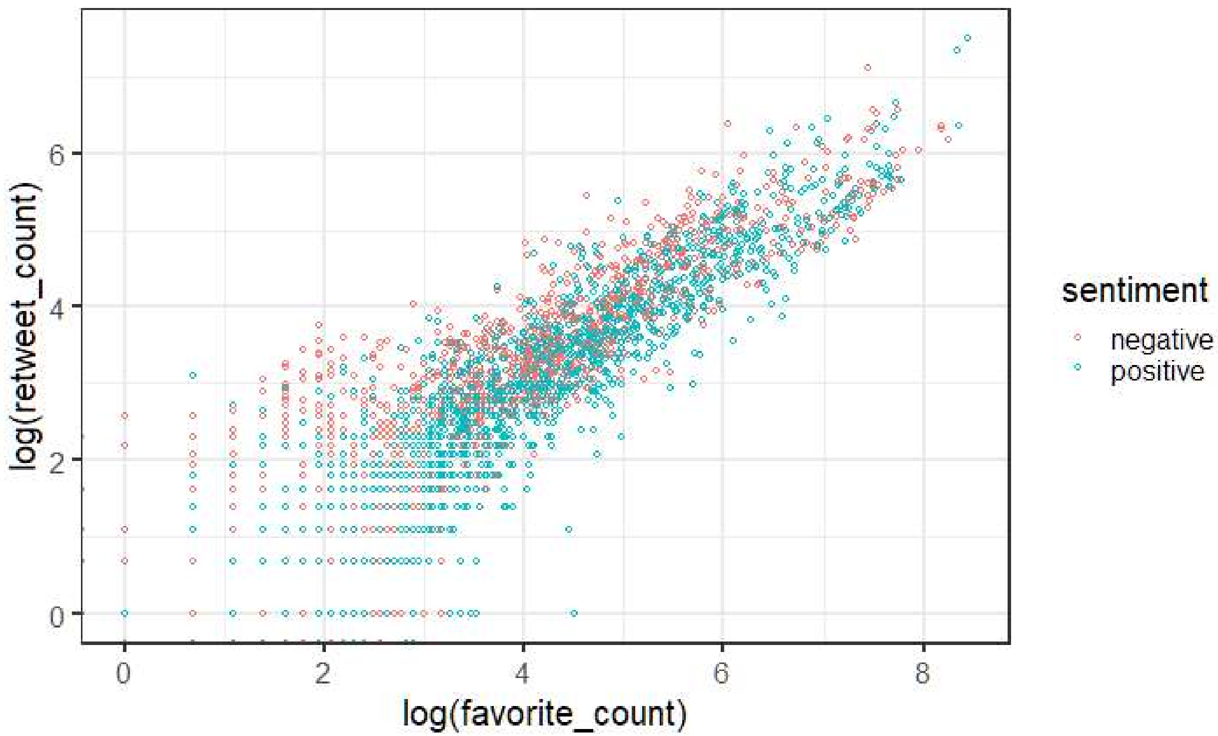
```
ggplot(Exploredf1)+
  geom_point(aes(y=log(retweet_count), x = log(favorite_count)), shape=1)+
  theme_bw(20)
```



```
ggplot(Exploredf1)+
  geom_point(aes(y=log(retweet_count), x = log(favorite_count), colour=media_type), shape=1)+
  theme_bw(20)
```



```
ggplot(sentiment1df)+
  geom_point(aes(y=log(retweet_count), x = log(favorite_count), colour=sentiment), shape=1)+
  theme_bw(20)
```



```
m1 <- lm(retweet_count ~ media_type, data = sentiment1df)
summary(m1)
```

```
##
## Call:
## lm(formula = retweet_count ~ media_type, data = sentiment1df)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
##  -68.95  -42.95  -18.95    4.24 1731.05
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)       19.762      1.443   13.70   <2e-16 ***
## media_typePhoto   49.191      1.988   24.75   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 89.9 on 8202 degrees of freedom
## Multiple R-squared:  0.06947,    Adjusted R-squared:  0.06936
## F-statistic: 612.4 on 1 and 8202 DF,  p-value: < 2.2e-16
```

```r
m2 <- lm(retweet_count ~ media_type + sentiment, data=sentiment1df)
summary(m2)
```

```
##
## Call:
## lm(formula = retweet_count ~ media_type + sentiment, data = sentiment1df)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
##  -78.32  -43.76  -13.77    4.67 1736.24
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)         28.333      1.873   15.12  < 2e-16 ***
## media_typePhoto     49.984      1.985   25.18  < 2e-16 ***
## sentimentpositive  -14.560      2.039   -7.14 1.01e-12 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 89.62 on 8201 degrees of freedom
## Multiple R-squared:  0.07522,    Adjusted R-squared:  0.075
## F-statistic: 333.5 on 2 and 8201 DF,  p-value: < 2.2e-16
```

```r
library(purrr)
library(modelr)
```

```
## Warning: package 'modelr' was built under R version 4.1.3
```

```r
cv  <- crossv_kfold(sentiment1df, k = 10)

cv
```

```
## # A tibble: 10 x 3
##    train                     test                     .id
##    <named list>              <named list>             <chr>
##  1 <resample [7,383 x 8]> <resample [821 x 8]> 01
##  2 <resample [7,383 x 8]> <resample [821 x 8]> 02
##  3 <resample [7,383 x 8]> <resample [821 x 8]> 03
##  4 <resample [7,383 x 8]> <resample [821 x 8]> 04
##  5 <resample [7,384 x 8]> <resample [820 x 8]> 05
##  6 <resample [7,384 x 8]> <resample [820 x 8]> 06
##  7 <resample [7,384 x 8]> <resample [820 x 8]> 07
##  8 <resample [7,384 x 8]> <resample [820 x 8]> 08
##  9 <resample [7,384 x 8]> <resample [820 x 8]> 09
## 10 <resample [7,384 x 8]> <resample [820 x 8]> 10
```

```
models0  <- map(cv$train, ~lm(retweet_count ~ 1, data = .))
models1  <- map(cv$train, ~lm(retweet_count ~ media_type, data = .))
models2  <- map(cv$train, ~lm(retweet_count ~ media_type + sentiment, data = .))
```

How does map function work?

```
my_list <- list(
  c(1,2,6),
  c(4,7,1),
  c(9,1,5)
)
```

```
my_list
```

```
## [[1]]
## [1] 1 2 6
##
## [[2]]
## [1] 4 7 1
##
## [[3]]
## [1] 9 1 5
```

```
my_list[[1]]
```

```
## [1] 1 2 6
```

```
my_list[[1]] %>% mean()
```

```
## [1] 3
```

```
my_list[[2]] %>% mean()
```

```
## [1] 4
```

```
my_list[[3]] %>% mean()
```


- -

```
my_list
```

```
## [[1]]
## [1] 1 2 6
##
## [[2]]
## [1] 4 7 1
##
## [[3]]
## [1] 9 1 5
```

```
my_list %>% map(mean)
```

```
## [[1]]
## [1] 3
##
## [[2]]
## [1] 4
##
## [[3]]
## [1] 5
```

```
map(my_list, mean)
```

```
## [[1]]
## [1] 3
```

38 / 61

```r
# gives a vector of dbl

my_list %>% map_dbl(mean)
```

```
## [1] 3 4 5
```

```r
# anonymous function
# ~ creation function with no specific name in tidyverse
# . every element in the list


my_list %>% map(~ . * 2)
```

```
## [[1]]
## [1]  2  4 12
##
## [[2]]
## [1]  8 14  2
##
## [[3]]
## [1] 18  2 10
```

```
get_pred  <- function(model, test_data){
  data  <- as.data.frame(test_data)
  pred  <- add_predictions(data, model)
  return(pred)
}

pred0  <- map2_df(models0, cv$test, get_pred, .id = "Run")
pred1  <- map2_df(models1, cv$test, get_pred, .id = "Run")
pred2  <- map2_df(models2, cv$test, get_pred, .id = "Run")
```

Mean Squared Error to assess model fit

$$\mathrm{MSE} = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2$$

MSE = mean squared error

n = number of data points

Y_i = observed values

$\hat{Y}_i$ = predicted values

- To find the MSE, take the observed value, subtract the predicted value, and square that difference. Repeat that for all observations. Then, sum all of those squared values and divide by the number of observations.

Interpretting Mean Squared Error

- An MSE of zero, meaning that the estimator $\hat{\theta}$ predicts observations of the parameter θ with perfect accuracy.
- Two or more statistical models may be compared using their MSEs as a measure of how well they explain a given set of observations.
- Mean squared error has the disadvantage of heavily weighting outliers. This property, undesirable in many applications, has led researchers to use alternatives such as the mean absolute error.

```
MSE0  <- pred0 %>% group_by(Run) %>%
  summarise(MSE = mean((retweet_count - pred)^2))

MSE0
```

```
## # A tibble: 10 x 2
##    Run      MSE
##    <chr>  <dbl>
##  1 1      5371.
##  2 10    15987.
##  3 2      5232.
##  4 3      9980.
##  5 4      7747.
##  6 5     10138.
##  7 6      8114.
##  8 7      6248.
##  9 8      8788.
## 10 9      9250.
```

```
MSE1  <- pred1 %>% group_by(Run) %>%
  summarise(MSE = mean( (retweet_count - pred)^2))

MSE1
```

```
## # A tibble: 10 x 2
##    Run      MSE
##    <chr>  <dbl>
##  1 1      4930.
##  2 10    15116.
##  3 2      4795.
##  4 3      9447.
##  5 4      7011.
##  6 5      9380.
##  7 6      7691.
##  8 7      5779.
##  9 8      8238.
## 10 9      8462.
```

```
MSE2  <- pred2 %>% group_by(Run) %>%
  summarise(MSE = mean( (retweet_count - pred)^2))

MSE2
```

```
## # A tibble: 10 x 2
##    Run      MSE
##    <chr>  <dbl>
##  1 1      4898.
##  2 10    14964.
##  3 2      4718.
##  4 3      9438.
##  5 4      7005.
##  6 5      9372.
##  7 6      7670.
##  8 7      5742.
##  9 8      8204.
## 10 9      8363.
```

```
mean(MSE0$MSE)
```

```
## [1] 8685.354
```

```
mean(MSE1$MSE)
```

```
## [1] 8084.726
```

```
mean(MSE2$MSE)
```

```
## [1] 8037.398
```

```r
Confirmdf <- read_csv("df2.csv")
```

```
## New names:
## Rows: 20000 Columns: 7
## -- Column specification
## -------------------------------------------------------- Delimiter: "," chr
## (3): screen_name, media_type, text dbl (4): ...1, ...2, favorite_count,
## retweet_count
## i Use `spec()` to retrieve the full column specification for this data. i
## Specify the column types or set `show_col_types = FALSE` to quiet this message.
## * `` -> `...1`
## * `...1` -> `...2`
```

```r
head(Confirmdf)
```

```
## # A tibble: 6 x 7
##    ...1   ...2 screen_name    favorite_count retweet_count media_type text
##   <dbl>  <dbl> <chr>                   <dbl>         <dbl> <chr>      <chr>
## 1     1    539 oceana                    438           121 Photo      GOOD NEWS~
## 2     2  22589 sascampaigns                0             0 Nophoto    @Isabella~
## 3     3  62705 ClimateReality             49            14 Nophoto    We<U+0092~
## 4     4 114033 pawtitions                  9            32 Photo      Demand ha~
## 5     5  38309 therightblue                1             0 Nophoto    Wildfire ~
## 6     6  62185 ClimateReality            654           226 Nophoto    Last mont~
```

Similar computations as on Exploredf

```
Confirmdf <- Confirmdf%>%
  mutate(Postnumber = 1:n())%>%
  select(-c(...1))

confirmsentiment <- Confirmdf %>%
  unnest_tokens(output = "word", input = "text")
```

```
sentiment_dictionary1 <- get_sentiments("bing")
head(sentiment_dictionary1)
```

```
## # A tibble: 6 x 2
##   word       sentiment
##   <chr>      <chr>
## 1 2-faces    negative
## 2 abnormal   negative
## 3 abolish    negative
## 4 abominable negative
## 5 abominably negative
## 6 abominate  negative
```

```
sentiment_dictionary2 <- get_sentiments("afinn")
head(sentiment_dictionary2)
```

```
## # A tibble: 6 x 2
##   word       value
##   <chr>      <dbl>
## 1 abandon       -2
## 2 abandoned     -2
## 3 abandons      -2
## 4 abducted      -2
## 5 abduction     -2
## 6 abductions    -2
```

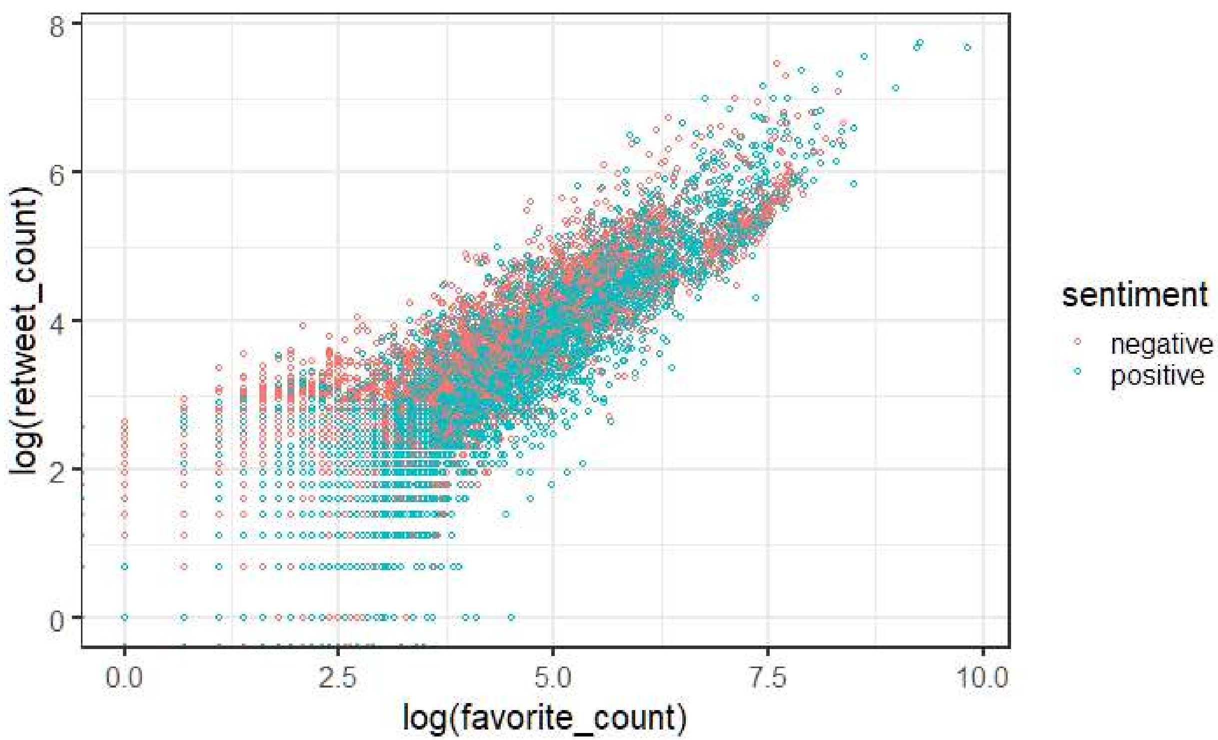
```
sentiment_dictionary3 <- get_sentiments("nrc")
head(sentiment_dictionary3)
```

```
## # A tibble: 6 x 2
##   word      sentiment
##   <chr>     <chr>
## 1 abacus    trust
## 2 abandon   fear
## 3 abandon   negative
## 4 abandon   sadness
## 5 abandoned anger
## 6 abandoned fear
```

```r
confirmsentiment1df <- merge(confirmsentiment, sentiment_dictionary1, by = "word")
head(confirmsentiment1df)
```

```
##       word   ...2     screen_name favorite_count retweet_count media_type
## 1 abnormal  70074       HSIGlobal            420           168    Nophoto
## 2 abnormal  80025    MoveTheWorld            315           331      Photo
## 3 abnormal  32344    savingoceans              6             2      Photo
## 4  abolish 112289 Network4Animals             23            21      Photo
## 5  abolish 101041   FarmSanctuary             48            10      Photo
## 6   abound  48231      Greenpeace             46            19    Nophoto
##   Postnumber sentiment
## 1      12913  negative
## 2      11822  negative
## 3       6203  negative
## 4       9532  negative
## 5       3559  negative
## 6      16354  positive
```

```
ggplot(confirmsentiment1df)+
  geom_point(aes(y=log(retweet_count), x = log(favorite_count), colour=sentiment), shape=1)+
  theme_bw(20)
```



Issues though...

```
predictedvalues <- predict(m1, data=sentiment1df, interval="prediction")
```

```
## Warning in predict.lm(m1, data = sentiment1df, interval = "prediction"): predictions on current data re
```

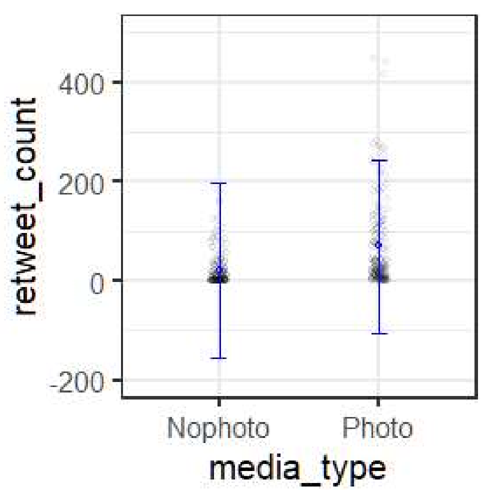
```
View(predictedvalues)
```

```r
v1 <- c(19.76, -156, 196)
v2 <- c(68.95279, -107, 245)
v3 <- c("Nophoto", "Photo")

smalldf <- rbind(v1,v2)

smalldf <- as.data.frame(smalldf)%>%
  rename("meanretweet" = "V1",
         "lower" = "V2",
         "upper" = "V3")

smalldf <- cbind(smalldf, v3)

smalldf <- smalldf %>%
  rename("media_type" = "v3")

sampleConfirmdf <- confirmsentiment1df %>%
  sample_n(400)
```

What's wrong with this picture?

```
ggplot()+
geom_jitter(data=sampleConfirmdf, aes(x = media_type, y = retweet_count), width = .05, height=.01
geom_point(data=smalldf, aes(x= media_type, y= meanretweet), shape = 1, colour = "blue")+
geom_errorbar(data=smalldf, aes(x= media_type, ymin = lower,ymax=upper), width=.1, colour = "blue'
scale_y_continuous(limits = c(-200,500))+
theme_bw(20)
```

```
## Warning: Removed 5 rows containing missing values (`geom_point()`).
```



Impossible values in our error bars

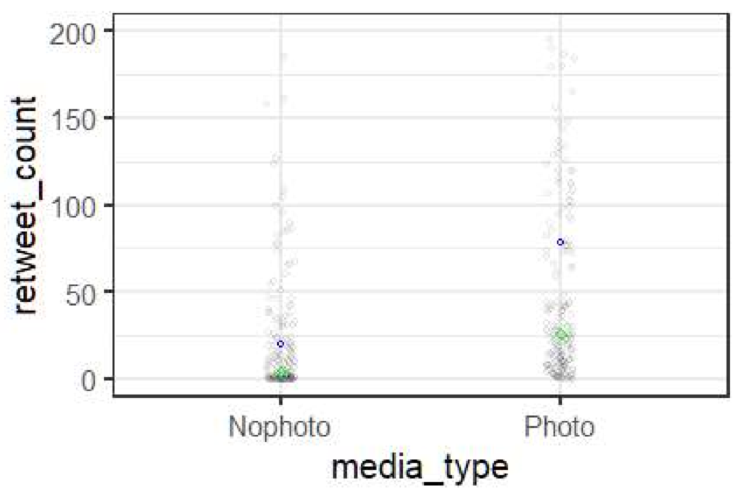
- Need a GLM that takes into account the data are counts!

```r
m1poss <- glm(retweet_count ~ media_type, data = Confirmdf, family = "poisson")
summary(m1poss)
```

```
## 
## Call:
## glm(formula = retweet_count ~ media_type, family = "poisson", 
##     data = Confirmdf)
## 
## Deviance Residuals: 
##    Min      1Q  Median      3Q     Max  
## -10.81   -5.71   -4.60    0.42  428.98  
## 
## Coefficients:
##                 Estimate Std. Error z value Pr(>|z|)    
## (Intercept)     2.789954   0.002365  1179.9   <2e-16 ***
## media_typePhoto 1.277500   0.002737   466.8   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## 
## (Dispersion parameter for poisson family taken to be 1)
```

```
sampleConfirmdf %>%
  group_by(media_type)%>%
  summarise(meanrt = mean(retweet_count),
            medianrt = median(retweet_count))%>%
  ggplot()+
  geom_jitter(data=sampleConfirmdf, aes(x = media_type, y = retweet_count), width = .05, height=.01
  geom_point(aes(x= media_type, y= meanrt), shape = 1, colour = "blue")+
  geom_point(aes(x= media_type, y= medianrt), shape = 5, colour = "green")+
  scale_y_continuous(limits = c(0,200))+
  theme_bw(20)
```

```
## Warning: Removed 56 rows containing missing values (`geom_point()`).
```



```
m2poss <- glm(retweet_count ~ media_type + sentiment , data = sentiment1df, family = "poisson")
summary(m2poss)
```

```
##
## Call:
## glm(formula = retweet_count ~ media_type + sentiment, family = "poisson",
##     data = sentiment1df)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -12.911   -6.852   -4.662    0.719   93.321
##
## Coefficients:
##                   Estimate Std. Error z value Pr(>|z|)
## (Intercept)        3.155850   0.003983   792.2   <2e-16 ***
## media_typePhoto    1.267198   0.004053   312.7   <2e-16 ***
## sentimentpositive -0.312762   0.003292   -95.0   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for poisson family taken to be 1)
##
##     Null deviance: 770433  on 8203  degrees of freedom
## Residual deviance: 644619  on 8201  degrees of freedom
## AIC: 677966
##
## Number of Fisher Scoring iterations: 6
```