Politics Are Afoot!

w203: Statistics for Data Science

The Setup

There is *a lot* of money that is spent in politics in Presidential election years. Like, a lot, a lot. Estimates and analysis from the US Federal Election Comission, puts the total amount at about $14,400,000,000 ($14.4 billion USD). For context, Twitter's 2020 annual revenue was about $3,500,000,000 ($3.5 billion USD).

The work

Install the package, `fec16`.

```
## install.packages('fec16')
```

This package is a compendium of spending and results from the 2016 election cycle. In this dataset are 9 different datasets that cover:

- `candidates`: candidate attributes, like their name, a unique id of the candidate, the election year under consideration, the office they're running for, etc.
- `results_house`: race attributes, like the name of the candidates running in the election, a unique id of the candidate, the number of `general_votes` garnered by each candidate, and other information.
- `campaigns`: financial information for each house & senate campaign. This includes a unique candidate id, the total receipts (how much came in the doors), and total disbursements (the total spent by the campaign), the total contributed by party central committees, and other information.

Your task

Your task is to describe the relationship between spending on a candidate's behalf and the votes they receive.

If it is helpful to structure your response, you might want to place yourself into a scenario where you are advising a person or business about whether they should make a political donation. While the benefits that accrue as a result of a successful investment are unclear, you can be quite sure that investing with **no** return (i.e. more spending does not increase the chances of winning) is a bad idea.

Your work

- We want to keep this work *relatively* constrained, which is why we're providing you with data through the `fec16` package. It is possible to gather all the information from current FEC reports, but it would require you to make a series of API calls that would pull us away from the core modeling tasks that we want you to focus on instead.
- Throughout this assignment, limit yourself to functions that are within the `tidyverse` family of packages: `dplyr`, `ggplot`, `patchwork`, and `magrittr` for wrangling and exploration and `base`, `stats`, `sandwich` and `lmtest` for modeling and testing. You do not *have* to use these packages; but try to limit yourself to using only these.
- Our choice to encourage you to use only these packages is to try to cut down on the amount of searching that you do: to help you avoid looking for the *"one package that does the thing I need it to do."* Certainly,

such a package exists, but it will very likely be more productive for you to write things yourself than to try and find it for this homework.

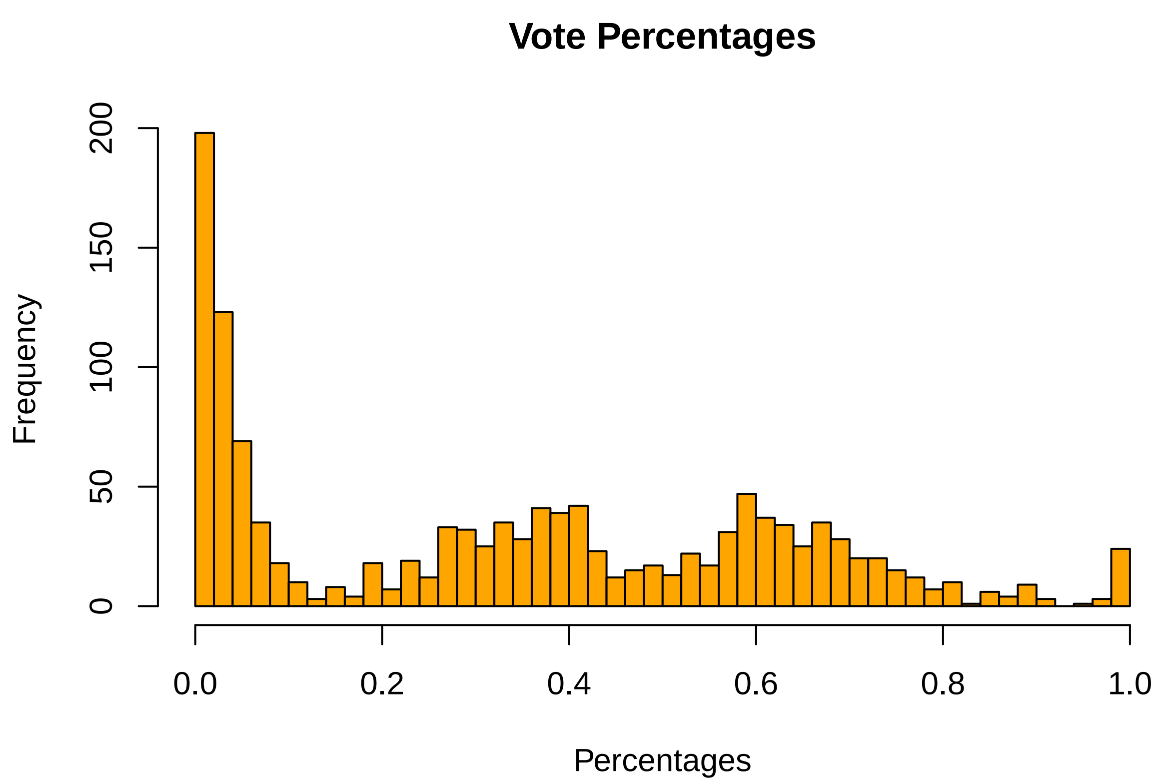
```
candidates    <- fec16::candidates
results_house <- fec16::results_house
campaigns     <- fec16::campaigns
```

1. What does the distribution of votes and of spending look like?

1. (3 points) In separate histograms, show both the distribution of votes (measured in `results_house$general_percent` for now) and spending (measured in `ttl_disb`). Use a log transform if appropriate for each visualization. How would you describe what you see in these two plots?

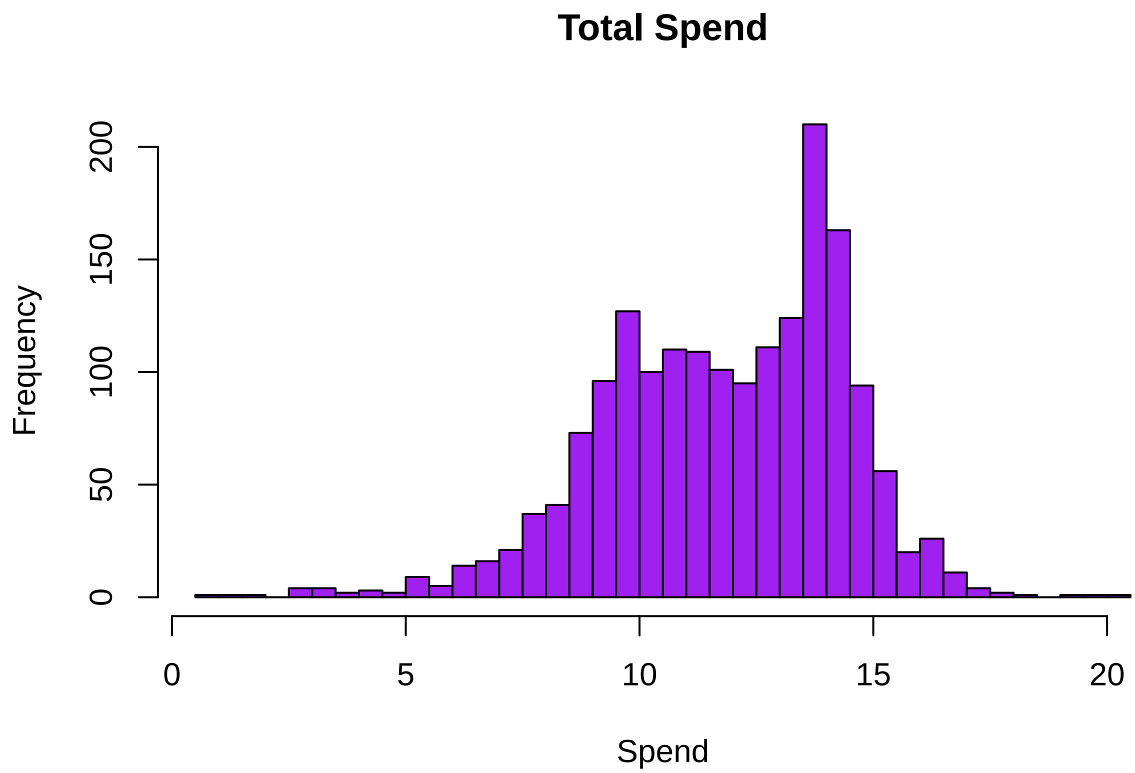
```
# explore the data

# plot the data
hist(results_house$general_percent, col = 'orange', breaks = 40, xlab = 'Percentages', main='Vote Percen
```




```
hist(log(campaigns$ttl_disb), col = 'purple', breaks = 40, xlab = 'Spend', main='Total Spend')

## Warning in log(campaigns$ttl_disb): NaNs produced
```



The original histogram of the results_house$general_percentisheavilyskewedwithatail.Theloghistogramofthecampaigns
is a standard normal distribution.

2. Exploring the relationship between spending and votes.

2. (3 points) Create a new dataframe by joining `results_house` and `campaigns` using the `inner_join` function from `dplyr`. (We use the format `package::function` – so `dplyr::inner_join`.) Does this data frame contain all the data that was present in the two frames that you're joining together, or has some data been dropped? As you're manipulating data, keep a keen eye for what is, and what is not making it through your data → analysis → reporting pipeline.

```
df_inner_join <- inner_join(results_house, campaigns)
```

```
## Joining, by = "cand_id"
```

```
# The dataframe results_house has 12 columns. The dataframe campaigns has 25 columns.
```

The inner join of the two dataframes does # contain all the columns with a total of 37 columns. The data does not match, as the original tables have 4008 rows but the join has 1342.

3. (3 points) Produce a scatter plot of `general_votes` on the y-axis and `ttl_disb` on the x-axis. What do you observe about the shape of the joint distribution?

```
ggplot(df_inner_join, aes(x = log10(ttl_disb), y =log10(general_votes) ) ) +
  geom_point() +
  ggtitle('Spend vs Votes') +
  labs(x = 'Spend', y = 'Votes' ) +
  geom_point(col='pink') +
```

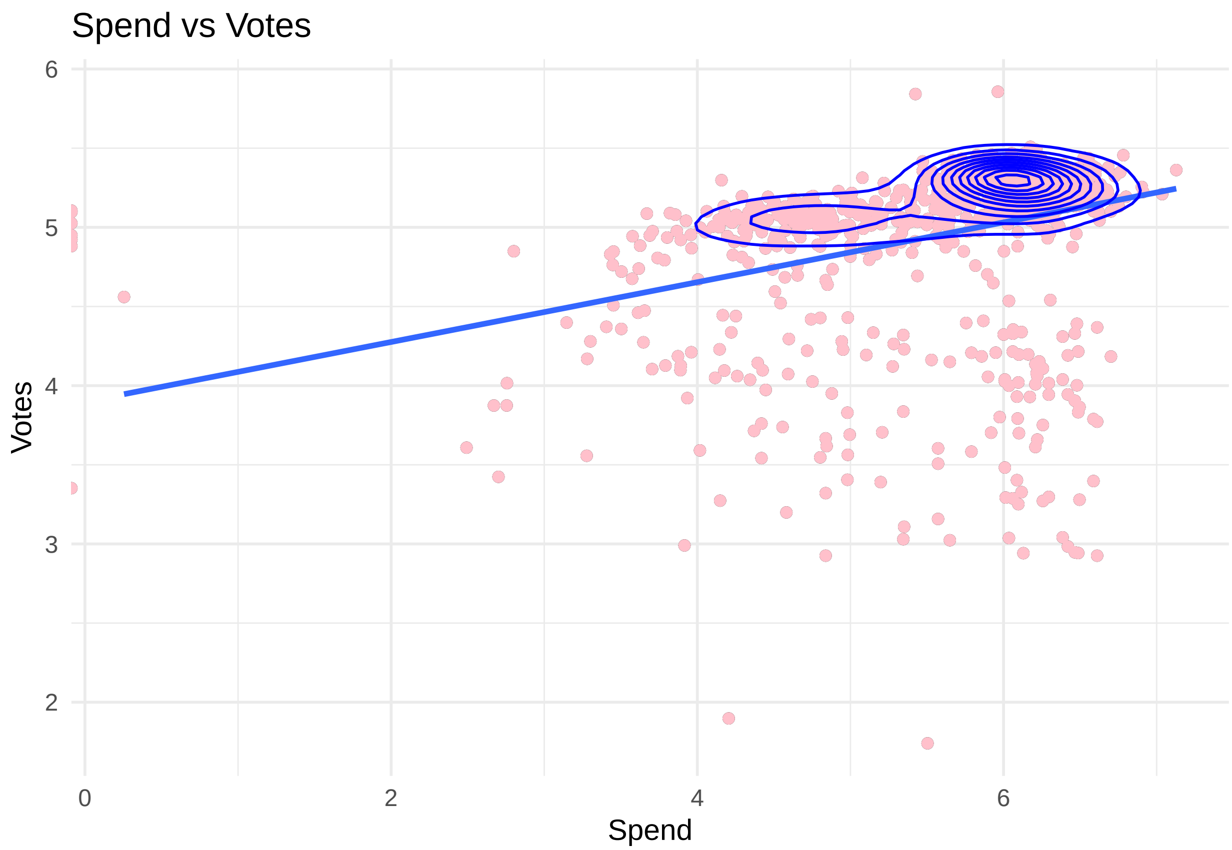
```
  geom_smooth(method = lm, se = FALSE) +
  geom_density2d(col = 'blue')
```

```
## `geom_smooth()` using formula 'y ~ x'

## Warning: Removed 469 rows containing non-finite values (stat_smooth).

## Warning: Removed 469 rows containing non-finite values (stat_density2d).

## Warning: Removed 462 rows containing missing values (geom_point).
## Removed 462 rows containing missing values (geom_point).
```



The log normalize the distribution. The data is heavly crowded to the right side. You can expect that an increase in the spend will increases the votes recieved. You can clearly see the density area in the right side.

4. (3 points) Create a new variable to indicate whether each individual is a "Democrat", "Republican" or "Other Party".

- Here's an example of how you might use `mutate` and `case_when` together to create a variable.

```
df_inner_join <- df_inner_join %>%
  mutate(
    party_new = case_when(
      party == "REP" ~ "Republican",
      party == "DEM" ~ "Democrat",
      TRUE           ~ "Other Party"
    ))
```

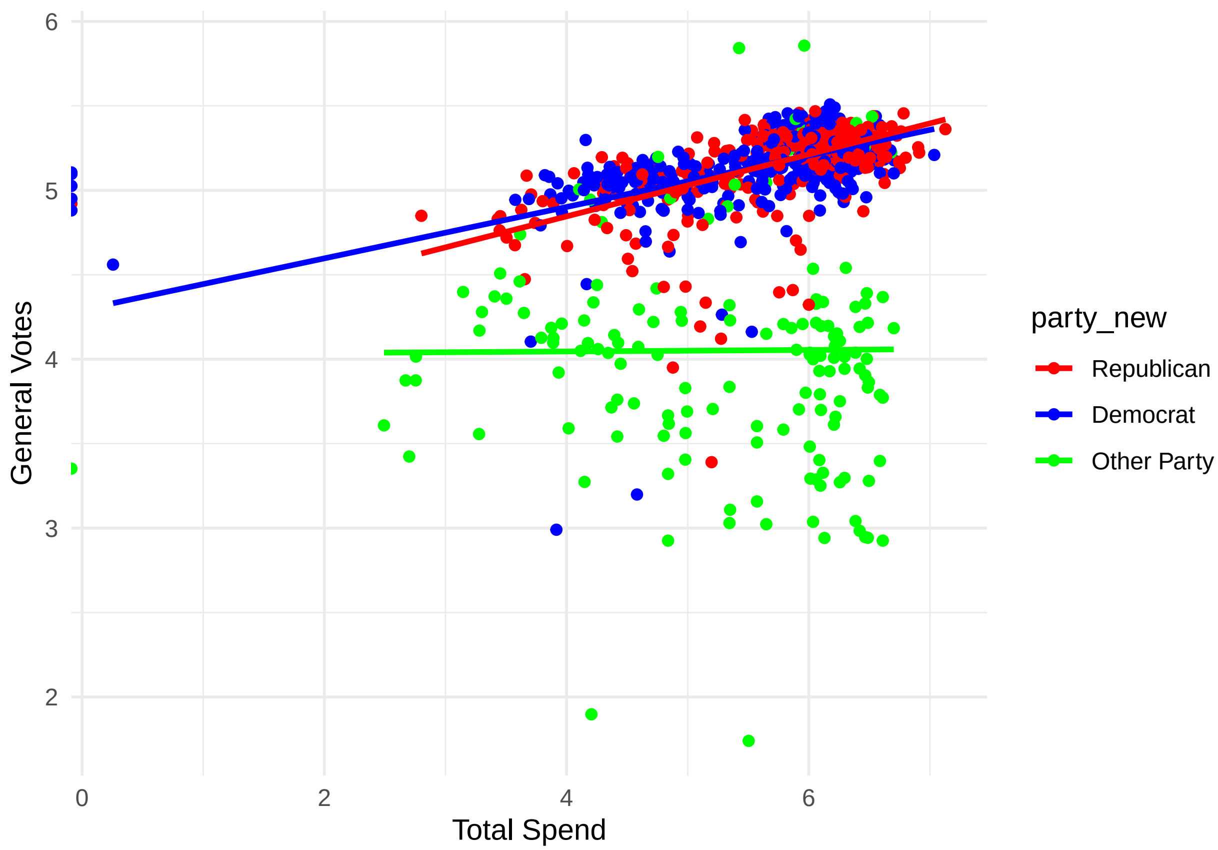
Once you've produced the new variable, plot your scatter plot again, but this time adding an argument into the `aes()` function that colors the points by party membership. What do you observe about the distribution of all three variables?

```
ggplot(df_inner_join, aes(x = log10(ttl_disb), y = log10(general_votes), color = party_new) ) +
  xlab("Total Spend") +
  ylab("General Votes") +
  geom_point() +
  geom_smooth(method = lm, se = FALSE) +
  scale_color_manual(
    values = c("Republican" = "red",
               "Democrat" = "blue",
               "Other Party" = "green")
  )
```

```
## `geom_smooth()` using formula 'y ~ x'

## Warning: Removed 469 rows containing non-finite values (stat_smooth).

## Warning: Removed 462 rows containing missing values (geom_point).
```



The data is very crowded and sked to the left. What is now visable are the distinct political party clusters of the density distribution.

Produce a Descriptive Model

For this section, rather than us providing you with `'fill in: '` prompts, you can write in whatever way is most effective for you. Please, limit this section to no more than three printed pages. (Your client – aka the

TAs – have a finite attention span!)

5. (5 Points) Given your observations, produce a linear model that you think does a good job at describing the relationship between candidate spending and votes they receive. You should decide what transformation to apply to spending (if any), what transformation to apply to votes (if any) and also how to include the party affiliation.
6. (3 points) Evaluate the Large-Sample Linear Model Assumptions
7. (3 points) Interpret the model coefficients you estimate.

- Tasks to keep in mind as you're writing about your model:
  - At the time that you're writing and interpreting your regression coefficients you'll be *deep* in the analysis. Nobody will know more about the data than you do, at that point. *So, although it will feel tedious, be descriptive and thorough in describing your observations.*
  - It can be hard to strike the balance between: on the one hand, writing enough of the technical underpinnings to know that your model meets the assumptions that it must; and, on the other hand, writing little enough about the model assumptions that the implications of the model can still be clear. We're starting this practice now, so that by the end of Lab 2 you will have had several chances to strike this balance.

```
# wrangle data drop any NA and 0 from data
df_inner_join %>% drop_na(general_votes)
```

```
## # A tibble: 880 x 38
##    state district_id cand_id   incumbent party primary_votes primary_percent
##    <chr> <chr>       <chr>     <lgl>     <chr>         <dbl>           <dbl>
##  1 AL    01          H4AL01123 TRUE      REP           71310           0.601
##  2 AL    02          H0AL02087 TRUE      REP           78689           0.664
##  3 AL    02          H6AL02167 FALSE     DEM              NA          NA
##  4 AL    03          H2AL03032 TRUE      REP           77432           0.760
##  5 AL    03          H4AL03061 FALSE     DEM              NA          NA
##  6 AL    04          H6AL04098 TRUE      REP           86660           0.812
##  7 AL    05          H0AL05163 TRUE      REP              NA          NA
##  8 AL    05          H6AL05202 FALSE     DEM              NA          NA
##  9 AL    06          H4AL06098 TRUE      REP              NA          NA
## 10 AL    06          H6AL06127 FALSE     DEM              NA          NA
## # ... with 870 more rows, and 31 more variables: runoff_votes <dbl>,
## #   runoff_percent <dbl>, general_votes <dbl>, general_percent <dbl>,
## #   won <lgl>, footnotes <chr>, cand_name <chr>, cand_ici <chr>, pty_cd <dbl>,
## #   cand_pty_affiliation <chr>, ttl_receipts <dbl>, trans_from_auth <dbl>,
## #   ttl_disb <dbl>, trans_to_auth <dbl>, coh_bop <dbl>, coh_cop <dbl>,
## #   cand_contrib <dbl>, cand_loans <dbl>, other_loans <dbl>,
## #   cand_loan_repay <dbl>, other_loan_repay <dbl>, debts_owed_by <dbl>, ...
```

```
df_inner_join %>% drop_na(ttl_disb)
```

```
## # A tibble: 1,342 x 38
##    state district_id cand_id   incumbent party primary_votes primary_percent
##    <chr> <chr>       <chr>     <lgl>     <chr>         <dbl>           <dbl>
##  1 AL    01          H4AL01123 TRUE      REP           71310          0.601
##  2 AL    01          H6AL01060 FALSE     REP           47319          0.399
##  3 AL    02          H0AL02087 TRUE      REP           78689          0.664
##  4 AL    02          H6AL02142 FALSE     REP           33015          0.278
##  5 AL    02          H6AL02159 FALSE     REP            6856          0.0578
##  6 AL    02          H6AL02167 FALSE     DEM              NA         NA
##  7 AL    03          H2AL03032 TRUE      REP           77432          0.760
```

```
##  8 AL    03          H6AL03157 FALSE     REP           24474          0.240
##  9 AL    03          H4AL03061 FALSE     DEM              NA         NA
## 10 AL    04          H6AL04098 TRUE      REP           86660          0.812
## # ... with 1,332 more rows, and 31 more variables: runoff_votes <dbl>,
## #   runoff_percent <dbl>, general_votes <dbl>, general_percent <dbl>,
## #   won <lgl>, footnotes <chr>, cand_name <chr>, cand_ici <chr>, pty_cd <dbl>,
## #   cand_pty_affiliation <chr>, ttl_receipts <dbl>, trans_from_auth <dbl>,
## #   ttl_disb <dbl>, trans_to_auth <dbl>, coh_bop <dbl>, coh_cop <dbl>,
## #   cand_contrib <dbl>, cand_loans <dbl>, other_loans <dbl>,
## #   cand_loan_repay <dbl>, other_loan_repay <dbl>, debts_owed_by <dbl>, ...
```

```r
df_inner_join <- filter(df_inner_join, general_votes > 0, ttl_disb > 0)

# linear regression model on full data
model_1 <- lm(general_votes ~ ttl_disb, data = df_inner_join)
model_2 <- lm(log(general_votes) ~ log(ttl_disb), data = df_inner_join)
model_3 <- lm(log(general_votes) ~ ttl_disb, data = df_inner_join)
model_4 <- lm(log(general_votes) ~ log(ttl_disb) +  incumbent + factor(party_new), data = df_inner_join

summary(model_1)
```

```
##
## Call:
## lm(formula = general_votes ~ ttl_disb, data = df_inner_join)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -179002  -44061    3859   55984  583751
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 1.218e+05  3.521e+03  34.589  < 2e-16 ***
## ttl_disb    1.420e-02  2.112e-03   6.725 3.17e-11 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 78560 on 871 degrees of freedom
## Multiple R-squared:  0.04936,    Adjusted R-squared:  0.04827
## F-statistic: 45.23 on 1 and 871 DF,  p-value: 3.17e-11
```

```r
summary(model_2)
```

```
##
## Call:
## lm(formula = log(general_votes) ~ log(ttl_disb), data = df_inner_join)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -7.3618  0.0359  0.4364  0.6678  2.1171
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)    8.97524    0.28137  31.898   <2e-16 ***
## log(ttl_disb)  0.18890    0.02156   8.763   <2e-16 ***
## ---
```

```
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.231 on 871 degrees of freedom
## Multiple R-squared:  0.08102,    Adjusted R-squared:  0.07996
## F-statistic: 76.79 on 1 and 871 DF,  p-value: < 2.2e-16
```

```
summary(model_3)
```

```
##
## Call:
## lm(formula = log(general_votes) ~ ttl_disb, data = df_inner_join)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -7.3051  0.0086  0.4467  0.7455  2.1463
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 1.127e+01  5.710e-02 197.388  < 2e-16 ***
## ttl_disb    1.308e-07  3.425e-08   3.819 0.000143 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.274 on 871 degrees of freedom
## Multiple R-squared:  0.01647,    Adjusted R-squared:  0.01534
## F-statistic: 14.59 on 1 and 871 DF,  p-value: 0.0001433
```

```
summary(model_4)
```

```
##
## Call:
## lm(formula = log(general_votes) ~ log(ttl_disb) + incumbent +
##     factor(party_new), data = df_inner_join)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -5.2197 -0.1624  0.0770  0.2533  4.2375
##
## Coefficients:
##                                Estimate Std. Error t value Pr(>|t|)
## (Intercept)                    10.79183    0.20855  51.748  < 2e-16 ***
## log(ttl_disb)                   0.06979    0.01708   4.087 4.77e-05 ***
## incumbentTRUE                   0.33331    0.06596   5.053 5.31e-07 ***
## factor(party_new)Other Party -2.44922    0.07674 -31.915  < 2e-16 ***
## factor(party_new)Republican  -0.01835    0.06000  -0.306     0.76
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.7968 on 868 degrees of freedom
## Multiple R-squared:  0.6165, Adjusted R-squared:  0.6147
## F-statistic: 348.8 on 4 and 868 DF,  p-value: < 2.2e-16
```

Findings Analysis