fec16 Exploratory Data Analysis

Richard Robbins

March 23, 2022

Contents

1 Introduction

The w203 Unit 10 homework assignment revolves around some of the Federal Election Commission 2016 datasets available in the `fec16` R package. As I worked through that assignment I noticed several anomalies. In the wake of the assignment I decided to go back and take a much closer look at the data. This document summarizes what I found. This is a work in progress. I have identified what I believe to be the most meaningful issues in the data.

2 Setup

This review looks at two of the `fec16` datasets, `campaigns` and `results_house`. The working datasets for this exercise, `df.reference` and `df.refined` are derived from `campaigns` and `results_house`. The `df.reference` dataset is formed by an inner join of `campaigns` and `results_house`, after which candidates for whom votes were not recorded are removed. The derivation of `df.refined` is described in detail below. The working datasets are limited to columns of interest.

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

```
df.reference <- inner_join(campaigns, results_house, by="cand_id") %>%
  drop_na(general_votes) %>%
  select (cand_id, cand_name, pty_cd, cand_pty_affiliation, party, incumbent,
          ttl_disb, general_votes, general_percent, won, state, district_id)
```

3 Troubling Signs

3.1 Distribution of General Election Voting Percentages

Figure 1 is a histogram of the general election voting percentages for candidates receiving votes as reflected in the `results_house` dataset. I was surprised to see the asymmetric mass on the left hand side of the histogram. I thought that perhaps seeing a larger group of candidates receiving low percentages might have reflected a third or fourth candidate in a contest between two other more dominant candidates. But I struggled to come up with a satisfactory explanation nonetheless.

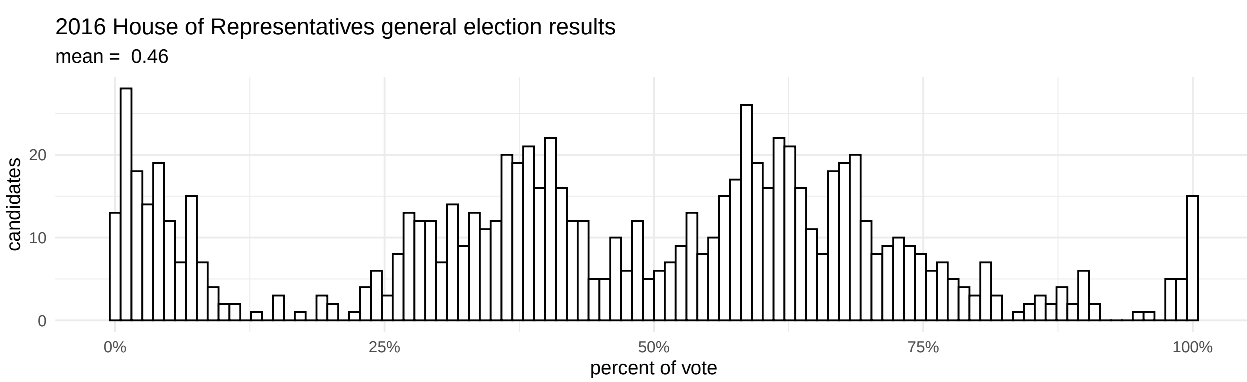

Figure 1: Voting Percentages

3.2 Party Affiliation

There are three ways to identify a candidate's party affiliation from the `campaigns` and `results_house` datasets: `results_house$party`, `campaigns$cand_pty_affiliation` and `campaigns$pty_cd`. Here are the unique values from each.

```
(unique (results_house$party))
```

```
##  [1] "REP"         "DEM"         "LIB"         "NAF"         "W"
##  [6] "IND"         "W(GRE)/GRE"  "W(LIB)"      "W(GRE)"      "W(D)"
## [11] "NOP"         "GRE"         "W(R)/R"      "PAF"         "W(NOP)"
## [16] "WF"          "IP"          "R/W"         "DCG"         "LBF"
## [21] "NPA"         "N"           "CON"         "W(R)"        "NNE"
## [26] "OTH"         "W(IND)"      "U"           "UST"         "W(D)/D"
## [31] "NLP"         "WC"          "DFL"         "IDP"         "LMN"
## [36] "REF"         "VPA"         "NPY"         "IAP"         "WDB"
## [41] "AO"          "MGW"         "RNN"         "PIP"         "FPR"
## [46] "EG"          "WUA"         "NSA"         "WOP"         "NBP"
## [51] "FI"          "LMP"         "TED"         "WTP"         "CRV"
## [56] "WEP"         "R/TRP"       "BLM"         "HBP"         "SID"
## [61] "TGP"         "PCC"         "UPJ"         "DNL"         "D/IP"
## [66] "W(IP)"       "R/IP"        "IP/R"        "PRO"         "D/PRO/WF/IP"
## [71] "R/CON"       "PG"          "W(D)/W"      "NPP"         "PPD"
## [76] "PRI"         "PPT"         "AM"          "UN"          "D/R"
## [81] "LBU"         "INP"         "WRN"         "TC"
```

```
(unique (campaigns$cand_pty_affiliation))
```

```
##  [1] "REP" "DEM" "IND" "GRE" "NNE" "UNK" "NPA" "LIB" "W"   "NON" "CON" "OTH"
## [13] "DFL" "IDP" "N/A" "UN"  "NPP" "PPT" "AMP" "CST" "GWP" "N"   "PBP" "PFD"
## [25] "PPY" "SEP"
```

```
(unique (campaigns$pty_cd))
```

```
## [1] 2 1 3
```

Party affiliation can be mapped as follows:

$$\text{for party or cand\_pty\_affiliation: } \begin{cases} \text{DEM} & \rightarrow \textit{Democrat} \\ \text{REP} & \rightarrow \textit{Republican} \\ \textit{else} & \rightarrow \textit{Other} \end{cases}$$

$$\text{for pty\_cd: } \begin{cases} 1 & \rightarrow \textit{Democrat} \\ 2 & \rightarrow \textit{Republican} \\ 3 & \rightarrow \textit{Other} \end{cases}$$

Which yields:

```
##                      Republicans Democrats Other
## party                        357       366   157
## cand_pty_affiliation         406       423    51
## pty_cd                       406       423    51
```

Using `party` yields 106 more "other party" candidates than the alternatives. The `cand_pty_affiliation` and `pty_cd` fields, as mapped, are equivalent for our purposes.

Let's review the winners (using `results_house$won`):

```
##                      Republicans Democrats Other
## party                        237       189    64
## cand_pty_affiliation         259       225     6
## pty_cd                       259       225     6
```

The numbers are troubling. The dataset indicates that 490 people were elected to the House of Representatives in the 2016 election. However, the number of voting representatives in the House of Representatives is fixed by law at no more than 435. Moreover, after the 2016 general election, all of the voting members of the House of Representatives were either Republicans or Democrats. There are also six non-voting members: a delegate representing the District of Columbia, a resident commissioner representing Puerto Rico, as well as one delegate for each of the other four permanently inhabited U.S. territories: American Samoa, Guam, the Northern Mariana Islands and the U.S. Virgin Islands. The were, in fact, 441 people elected to the House of Representatives in the election.

3.3 Multiple Observations For Some Candidates

While working with the data, I found instances of candidate IDs appearing more than once in `results_house` and, to a lesser degree, candidate names appearing more than once in `campaigns`.

To explore this, a function to count the number of times any candidate id (`cand_id`) appears more than once in a dataset.

```
redundant.ids <- function (dataset){
  if ("cand_id" %in% colnames (dataset))
    {dataset %>% group_by (cand_id) %>% count() %>% filter (n>1)}
  else {NA}
}
```

The table below shows the degree to which the various datasets include more than one observation for a single candidate id.

```
##         dataset redundant.ids
## 1     campaigns             0
## 2 results_house            97
## 3  df.reference            52
```

Lets take a look at the data for a few candidates appearing more than once the working dataset.

```
## # A tibble: 10 x 7
##    cand_id   cand_name       pty_cd party ttl_disb general_votes general_percent
##    <chr>     <chr>            <dbl> <chr>    <dbl>         <dbl>           <dbl>
##  1 HOCT03072 DELARUO, ROSA L      1 DEM    1150879.        192274          0.621
##  2 HOCT03072 DELARUO, ROSA L      1 WF     1150879.         21298          0.0688
##  3 HONY29054 REED, THOMAS W~      2 REP    3072934.        136964          0.490
##  4 HONY29054 REED, THOMAS W~      2 CRV    3072934.         16420          0.0587
##  5 HONY29054 REED, THOMAS W~      2 REF    3072934.           876          0.00313
##  6 HONY29054 REED, THOMAS W~      2 IDP    3072934.          6790          0.0243
```

```
##   7 H2CT02112 COURTNEY, JOSE~       1 DEM   1154847.        186210         0.564
##   8 H2CT02112 COURTNEY, JOSE~       1 WF    1154847.         22608         0.0685
##   9 H2CT05131 ESTY, ELIZABETH       1 DEM   1447329.        163499         0.529
## 10 H2CT05131 ESTY, ELIZABETH       1 WF    1447329.         15753         0.0510
```

There is a pattern. The rows for the candidates differ in that it appears that a candidate's total vote and percent of vote received are splintered and allocated to several parties.

The campaign spend field (`ttl_disb`) field is consistent for a candidate. That is an artifact of how we joined `campaigns` and `results_house`. The `campaigns` dataset is the source of the campaign spending information and that dataset does not contain redundant campaign ID observations. When we perform the join, the spending information is replicated in each redundant observation in `results_house`. As a result, our working dataset not only has too many other party candidates, campaign spend is inflated.

Upon review of generally available election returns, for the candidates listed above, I confirmed that the total votes and correct percent of votes for each can be determined by adding the amounts shown in their respective redundant observations.

4 Mitigation

4.1 The Approach

I have come to learn that some states allow candidates to appear on multiple party lines, and that some reports separate vote totals for each party. Therefore, for analysis that involves candidate totals, it is necessary to aggregate across all party lines within a district. For analysis that focuses on two-party vote totals, it is necessary to account for major party candidates who receive votes under multiple party labels. This is a topic discussed in this codebook maintained by the MIT Election Data Science Lab.

The refined date frame can be derived from the original as follows:

```
df.refined <- df.reference %>%
  group_by(cand_id, cand_name, pty_cd, incumbent, won, ttl_disb, state) %>%
  summarise (general_votes = sum(general_votes),
             general_percent = sum(general_percent),
             .groups = "keep")
```

4.2 Assessing Mitigation Effectiveness

Let's review what the dataset looks like after the mitigation step described above.

```
print(NROW(df.reference))
```

```
## [1] 880
```

```
print(NROW(df.refined))
```

```
## [1] 792
```

```
print(redundant.ids(df.refined))
```

```
## # A tibble: 1 x 2
## # Groups:   cand_id [1]
##   cand_id       n
##   <chr>     <int>
## 1 H2NY03089     2
```

1. The number of observations has been reduced from 880 to 792.
2. The number of redundant candidate IDs has been reduced to 1.

The breakdown by party (using `pty_cd`) is as follows:

```
##        Republicans Democrats Other
## pty_cd         366       376    50
```

And if we isolate on the winners:

```
##        Republicans Democrats Other
## pty_cd         236       194     6
```

4.3 Examining "Other Party" Winners

The six candidates identified as winning but not affiliated with a major party are:

```
## # A tibble: 6 x 9
## # Groups:   cand_id, cand_name, pty_cd, incumbent, won, ttl_disb, state [6]
##   cand_id  cand_name         pty_cd incumbent won   ttl_disb state general_votes
##   <chr>    <chr>              <dbl> <lgl>     <lgl>    <dbl> <chr>         <dbl>
## 1 H0MN040~ MCCOLLUM, BETTY        3 TRUE      TRUE   966856. MN           203299
## 2 H2IL131~ DAVIS, RODNEY L        3 TRUE      TRUE  2364757. IL           187583
## 3 H2MN081~ NOLAN, RICHARD M.      3 TRUE      TRUE  2892902. MN           179098
## 4 H4MO081~ SMITH, JASON T         3 TRUE      TRUE  1312777. MO           229792
## 5 H6PR000~ GONZALEZ COLON, ~      3 FALSE     TRUE   917851. PR           718591
## 6 H8MP000~ SABLAN, GREGORIO~      3 TRUE      TRUE    56475. MP            10605
## # ... with 1 more variable: general_percent <dbl>
```

- Betty McCollum and Richard Nolan are Democrats. The dataset is incorrect.
- Rodney David and Jason Smith are Republicans. The dataset is incorrect.
- Jennifer Gonzalez is neither a Republican nor a Democrat. She represents Puerto Rico as a delegate.
- Greogio Sablan is now a Republican. In 2016 he was neither a Republican nor a Democrat. He represents the Northern Mariana Islands as a delegate.

So, while there are 436 people identified as winners in the 2016 House of Representatives election, that includes 2 delegates who are not considered voting members of the House. Accordingly, every voting member has been accounted for as a Republican or Democrat. However, the total number of candidates elected was 441 consisting of 435 members with voting privileges plus six additional non-voting members, as described above.

4.4 The Lone Duplicate Entry Remaining

That leaves one person with duplicate entries in the dataset. That distinction goes to Peter T. King. His record was not merged because when we formed the grouping, to be conservative, we added several indicators from the dataset to prevent merging records that warranted further examination. In this case, the dataset for Mr. King has one row indicating he is not an incumbent who lost and another indicating he is an incumbent who won. In fact, Mr. King was an incumbent and he won re-election in 2016 with 181,506 votes.

```
## # A tibble: 2 x 9
## # Groups:   cand_id, cand_name, pty_cd, incumbent, won, ttl_disb, state [2]
##   cand_id   cand_name        pty_cd incumbent won   ttl_disb state general_votes
##   <chr>     <chr>             <dbl> <lgl>     <lgl>    <dbl> <chr>         <dbl>
## 1 H2NY03089 KING, PETER T. ~      2 FALSE     FALSE 1310730. NY            23935
## 2 H2NY03089 KING, PETER T. ~      2 TRUE      TRUE  1310730. NY           157571
## # ... with 1 more variable: general_percent <dbl>
```

4.5 Revisiting that Histogram

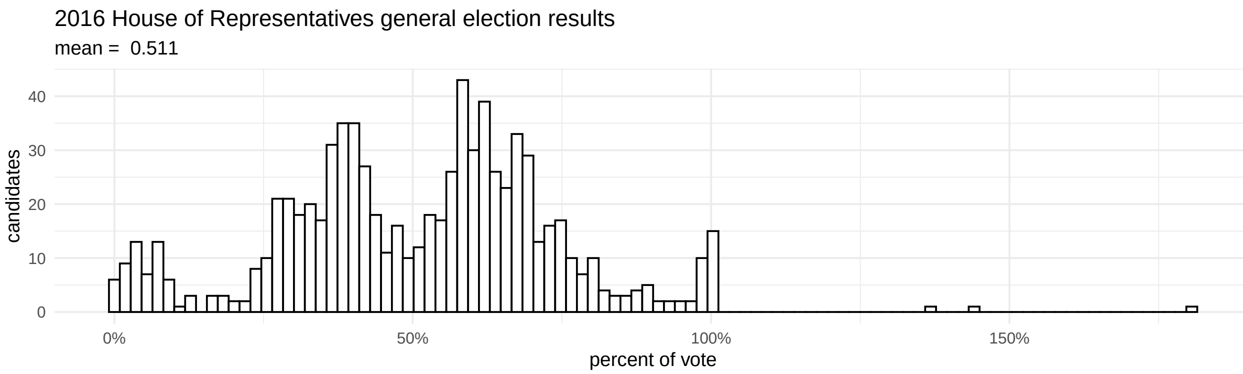

Figure 2: Revised Voting Percentages

Figure 2 reproduces the histogram from Figure 1 using the refined dataset. The asymmetric mass on the left hand side of the histogram is no longer present. The histogram appears to be much more symmetric.

However, we now appear to have a few candidates receiving more than 100% of the vote in their races.

5 Vote Totals, Write In Votes ansd Uncontested Elections

Figure 2 suggests we should take a closer look at the voting data contained in the `results_house` dataset. We should not see anyone with a vote percentage that exceeds 100%. In addition, it's reasonable to expect that in most cases, the sum of the votes received by candidates on the ballot will be less than total votes cast in order to account for write in candidates. Finally, we should think about uncontested elecetions. We explore each of these concepts below.

5.1 Vote Totals

We can derive the implied total number of votes cast from the number of votes received by a candidate and the percentage of votes received by that candidate, *i.e.,* $votes\_cast = \frac{general\_votes}{general\_percent}$. We would expect that number to be the same when calculated across every observation for a particular race.

First, let's see votes cast for some observations.

```
results_house %>%
  drop_na(general_votes) %>%
  mutate (votes_cast = general_votes/general_percent) %>%
  select (state, district_id, cand_id, general_votes, general_percent, votes_cast)
```

```
## # A tibble: 1,291 x 6
##    state district_id cand_id   general_votes general_percent votes_cast
##    <chr> <chr>       <chr>             <dbl>           <dbl>      <dbl>
##  1 AL    01          H4AL01123        208083           0.964     215893
##  2 AL    02          H0AL02087        134886           0.488     276584
##  3 AL    02          H6AL02167        112089           0.405     276584
##  4 AL    03          H2AL03032        192164           0.669     287104
##  5 AL    03          H4AL03061         94549           0.329     287104
##  6 AL    04          H6AL04098        235925           0.985     239444
##  7 AL    05          H0AL05163        205647           0.667     308326
##  8 AL    05          H6AL05202        102234           0.332     308326
##  9 AL    06          H4AL06098        245313           0.745     329306
## 10 AL    06          H6AL06127         83709           0.254     329306
## # ... with 1,281 more rows
```

That looks good. The rows for a stete and district pair imply a consistent number of votes cast for the candidates running in that particular contest.

Next we calculate the number of unique votes cast totals for each contest.

```
results_house %>%
  drop_na(general_votes) %>%
  mutate (votes_cast = general_votes/general_percent) %>%
  group_by(state, district_id) %>%
  summarise(unique_votes_cast = length(unique(votes_cast)), .groups = "keep")
```

```
## # A tibble: 443 x 3
## # Groups:   state, district_id [443]
##    state district_id unique_votes_cast
##    <chr> <chr>                   <int>
##  1 AK    00                          1
##  2 AL    01                          1
##  3 AL    02                          1
##  4 AL    03                          1
##  5 AL    04                          1
##  6 AL    05                          1
##  7 AL    06                          1
##  8 AL    07                          1
##  9 AR    01                          1
## 10 AR    02                          1
## # ... with 433 more rows
```

That sample shows a consistent number for the contests shown. Let's look for exceptions.

```r
results_house %>%
  drop_na(general_votes) %>%
  mutate (votes_cast = general_votes/general_percent) %>%
  group_by(state, district_id) %>%
  summarise(unique_votes_cast = length(unique(votes_cast)), .groups = "keep") %>%
  filter(unique_votes_cast > 1)
```

```
## # A tibble: 71 x 3
## # Groups:   state, district_id [71]
##    state district_id unique_votes_cast
##    <chr> <chr>                   <int>
##  1 AS    00                          2
##  2 CA    19                          2
##  3 CA    20                          2
##  4 CA    22                          2
##  5 CA    27                          2
##  6 CA    31                          2
##  7 CA    46                          2
##  8 CA    47                          2
##  9 CA    52                          2
## 10 CO    01                          2
## # ... with 61 more rows
```

That test shows more than a few contests where we appear to be getting inconsistent total votes cast numbers. Perhaps this is due to rounding errors. Let's round the votes cast number to the nearest integer.

```r
results_house %>%
  drop_na(general_votes) %>%
  mutate (votes_cast = round(general_votes/general_percent, 0)) %>%
  group_by(state, district_id) %>%
  summarise(unique_votes_cast = length(unique(votes_cast)), .groups = "keep") %>%
  filter(unique_votes_cast > 1)
```

```
## # A tibble: 2 x 3
## # Groups:   state, district_id [2]
##   state district_id unique_votes_cast
##   <chr> <chr>                   <int>
## 1 KS    01                          2
## 2 PA    07                          2
```

We are down to just two suspicious contests. Let's take a closer look.

```r
results_house %>%
  drop_na(general_votes) %>%
  mutate (votes_cast = round(general_votes/general_percent, 0)) %>%
  group_by(state, district_id) %>%
  summarise(unique_votes_cast = length(unique(votes_cast)),
            unique_vote_list = paste(unique(votes_cast), collapse=", "),
            .groups = "keep") %>%
  filter(unique_votes_cast > 1)
```

```
## # A tibble: 2 x 4
## # Groups:   state, district_id [2]
##   state district_id unique_votes_cast unique_vote_list
##   <chr> <chr>                   <int> <chr>
## 1 KS    01                          2 257971, NA
## 2 PA    07                          2 NA, 379649
```

It seems that we were unable to calculate the total votes cast in some instances because the denominator, `general_percent` was missing. Let's see.

```
results_house %>%
  drop_na(general_votes) %>%
  filter(is.na (general_percent)) %>%
  select (state, district_id, cand_id, general_votes, general_percent)
```

```
## # A tibble: 2 x 5
##   state district_id cand_id   general_votes general_percent
##   <chr> <chr>       <chr>             <dbl>           <dbl>
## 1 KS    01          H6KS01146           874              NA
## 2 PA    07          H0PA07082        225678              NA
```

So we have two rows missing the percentage of votes received by the candidate.

5.2 How Can A Candidate Exceed 100% Of The Vote

Now that we have gone to the trouble of confirming that the `general_votes` and `general_percent` fields yield a consistent number for total votes cast in each district, how can we explain why some candidates vote percentages as shown in Figure 2 exceed 100%.

Here are the candidates from our deduped dataset with more than 100% of the vote.

```
df.refined %>% filter(general_percent > 1)
```

```
## # A tibble: 3 x 9
## # Groups:   cand_id, cand_name, pty_cd, incumbent, won, ttl_disb, state [3]
##   cand_id   cand_name        pty_cd incumbent won   ttl_disb state general_votes
##   <chr>     <chr>             <dbl> <lgl>     <lgl>    <dbl> <chr>         <dbl>
## 1 H2HI02110 HANABUSA, COLLE~      1 FALSE     TRUE   489871. HI           274500
## 2 H6KY01110 COMER, JAMES          2 FALSE     TRUE  1070732. KY           426769
## 3 H6PA02171 EVANS, DWIGHT         1 FALSE     TRUE  1498445. PA           602953
## # ... with 1 more variable: general_percent <dbl>
```

Let's go back and look at the original results data for each.

```
results_house %>% filter(cand_id %in% c("H2HI02110", "H6KY01110", "H6PA02171"))
```

```
## # A tibble: 6 x 13
##   state district_id       cand_id  incumbent party primary_votes primary_percent
##   <chr> <chr>             <chr>    <lgl>     <chr>         <dbl>           <dbl>
## 1 HI    01 - FULL TERM    H2HI021~ FALSE     DEM           74022           0.804
## 2 HI    01 - UNEXPIRED T~ H2HI021~ FALSE     DEM              NA          NA
```

```
## 3 KY    01 - FULL TERM    H6KY011~ FALSE     REP           24342            0.606
## 4 KY    01 - UNEXPIRED T~ H6KY011~ FALSE     REP              NA           NA
## 5 PA    02 - FULL TERM    H6PA021~ FALSE     DEM           75515            0.422
## 6 PA    02 - UNEXPIRED T~ H6PA021~ FALSE     DEM              NA           NA
## # ... with 6 more variables: runoff_votes <dbl>, runoff_percent <dbl>,
## #   general_votes <dbl>, general_percent <dbl>, won <lgl>, footnotes <chr>
```

Aha! There's something unusual about these district identification labels. After a little bit of additional research, I realized that in some states, when there is a vacancy created in advance of the normal expiration of a term of office, the election for the balance of the unexpired term is held separate and apart from the election for the next succeeding full term. So, the `results_house` data is showing the "normal" full term election results with a "FULL TERM" district suffix and the unexpired term election results with the "UNEXPIRED TERM" district suffix.

We will remove results for the unexpired term elections and normalize the "FULL TERM" labels.

```
df.refined <- df.reference %>%
  filter (!grepl ("UNEXPIRED", district_id)) %>%
  mutate (district_id = str_pad(str_extract(district_id, "[:digit:]{1,2}"),
                                2,
                                pad="0")) %>%
  group_by(cand_id, cand_name, pty_cd, incumbent, won, ttl_disb, state, district_id) %>%
  summarise (general_votes = sum(general_votes),
             general_percent = sum(general_percent),
             .groups = "keep")
```

Now let's look at the voting percentage distribution histogram,

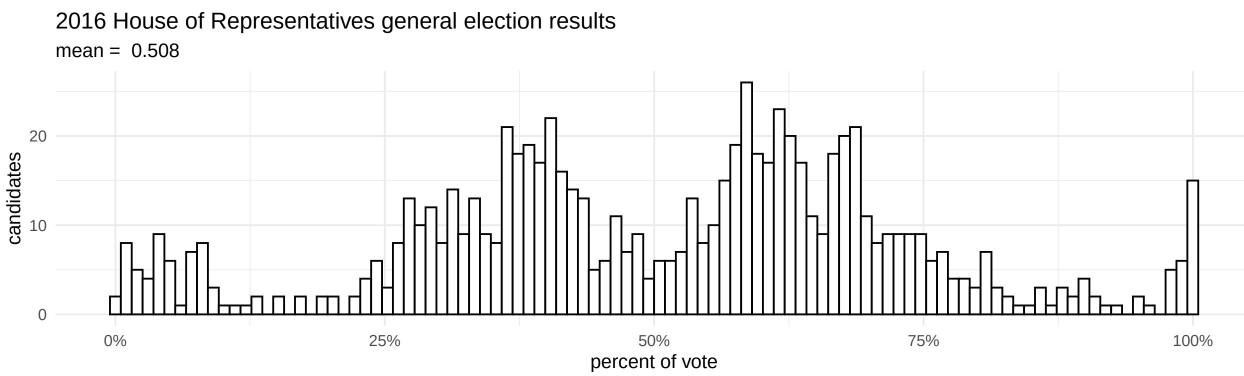

Figure 3: Revised Voting Percentages

The voting percentage histrogram looks much better. The pronounced density mass on the left attributable to slivers of splintered votes has been removed and the distortions due to merging the elections for full and unexpired terms has also been removed.

5.3 Write In Votes

Now that we have confirmed that the data for the candidates participating in a contest implies a consistent total number of votes cast in the election, let's compare the total number of votes accounted for in each contest with the number of votes cast that we have derived. When the numbers match that means that there are no write in ballots included in the observations for that contest. When the total votes recorded is less than the total vote cast number we expect for that election the deficit represents either missing candidates or write in ballots. The total number of votes recorded should never exceed the votes cast number we recorded.

```
vote_reconciliation <- df.refined %>%
  mutate (implied_votes_cast = round(general_votes/general_percent, 0)) %>%
  group_by(state, district_id) %>%
  summarise(implied_votes_cast = max(implied_votes_cast, na.rm=TRUE),
            votes_recorded = sum(general_votes),
            vote_deficit = implied_votes_cast - votes_recorded,
            .groups = "keep")

summary(vote_reconciliation$vote_deficit)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##       0       0    4578   24038   27746  193457
```

```
sum(vote_reconciliation$vote_deficit == 0)
```

```
## [1] 124
```

```
sum(between(vote_reconciliation$vote_deficit, 1, 1000))
```

```
## [1] 80
```

```
sum(between(vote_reconciliation$vote_deficit, 1001, 10000))
```

```
## [1] 49
```

```
sum(vote_reconciliation$vote_deficit > 10000)
```

```
## [1] 185
```

```
vote_reconciliation %>% filter (vote_deficit > 10000)
```

```
## # A tibble: 185 x 5
## # Groups:   state, district_id [185]
##    state district_id implied_votes_cast votes_recorded vote_deficit
##    <chr> <chr>                    <dbl>          <dbl>        <dbl>
##  1 AK    00                      308198         266107        42091
##  2 AL    02                      276584         246975        29609
##  3 AR    01                      241047         183866        57181
##  4 AR    02                      302464         287819        14645
##  5 AR    03                      280907         217192        63715
##  6 AR    04                      244159         182885        61274
##  7 AZ    01                      280710         263964        16746
##  8 AZ    02                      315679         179806       135873
##  9 AZ    08                      298971         204942        94029
## 10 CA    22                      234966         158755        76211
## # ... with 175 more rows
```

More work to do . . . ..

Let's repeat our vote total integrity test.

```
df.refined %>%
  mutate (votes_cast = round(general_votes/general_percent, 0)) %>%
  group_by(state, district_id) %>%
  summarise(unique_votes_cast = length(unique(votes_cast)), .groups = "keep") %>%
  filter(unique_votes_cast > 1)
```

```
## # A tibble: 2 x 3
## # Groups:   state, district_id [2]
##   state district_id unique_votes_cast
##   <chr> <chr>                   <int>
## 1 KS    01                          2
## 2 PA    07                          2
```

No changes there.

6 Using the rebuilt dataframe

```
df.final <- results_house %>%

  # The dataset includes full term races (the norm) and several races
  # for unexpired terms.  We eliminate the races for unexpired terms, as
  # they are special elections and separate from the full term elections.

  filter (!grepl ("UNEXPIRED", district_id)) %>%

  # Fix non-conforming district_id labels by extracting the district number
  # and preserving it as a two character string padded with leading zeros.
  # These labels reference full term races in the presence of unexpired
  # term races, which we just eliminated.

  mutate (district_id = str_pad(str_extract(district_id, "[:digit:]{1,2}"),
                                2,
                                pad="0")) %>%

  # There are a few races where no votes were recorded because a candidate
  # ran unopposed, so rather than drop rows without votes recorded, we
  # keep any row for a winner including those who did not receive votes as well
  # as any row where a candidate received votes in the general election.

  filter (won | !is.na(general_votes)) %>%

  # There are six non-voting members: a delegate representing the
  # District of Columbia, a resident commissioner representing Puerto Rico,
  # as well as one delegate for each of the other four permanently inhabited
  # U.S. territories: American Samoa, Guam, the Northern Mariana Islands and
  # the U.S. Virgin Islands.  We create a new indicator for those races.
```

```r
  mutate (non_voting = state %in% c("AS", "DC", "GU", "MP", "PR", "VI"),
          .after = "district_id") %>%

  # Peter King, candidate H2NY03089 was the incumbent in the 2nd district of NY.
  # He also won the general election for that seat.  A few rows in the dataset
  # indicate that he was not the incumbet and that he lost.  We correct that.
  # The correction is needed for the next transformation to work correctly.

  mutate (incumbent = ifelse (cand_id == "H2NY03089", TRUE, incumbent)) %>%
  mutate (won = ifelse (cand_id == "H2NY03089", TRUE, won)) %>%

  # Some states allow candidates to appear on multiple party lines, separate
  # vote totals are indicated for each party.  Therefore, for analysis that
  # involves candidate totals, it is necessary to aggregate across all party
  # lines within a district.  For analysis that focuses on two-party vote
  # totals, it is necessary to account for major party candidates who receive
  # votes under multiple party labels.

  group_by (state, district_id, non_voting, cand_id, incumbent, won) %>%
  summarise(general_votes=sum(general_votes),
            general_percent=sum(general_percent),
            votes_cast=general_votes/general_percent,
            party.labels = paste(unique(party), collapse=", "),
            .groups = "keep")

# Now that the results dataset has been cleaned up, let's count the number of
# candidates.  This is useful in case we want to analyze uncontested elections.

candidate_counts <- df.final %>%
  group_by(state, district_id) %>%
  summarise(candidate_count = length(unique(cand_id)),
            .groups = "keep")

df.final <- inner_join(df.final, candidate_counts,
                       by = c("state", "district_id"))

df.final <- inner_join(campaigns, df.final, by="cand_id") %>%
  select (state, district_id, candidate_count,non_voting,
          cand_id, cand_name, pty_cd, incumbent, won,
          general_votes, general_percent, votes_cast, ttl_disb)
```

6.1 Revisiting that Histogram Again

7 What About Homework 10?

A review of the model that I built for Unit 10 is beyond what I aim to achieve in this summary. However, I did redo much of my work using the refined dataset that reflects the changes described here.

Had I used this dataset I would have submitted a different model. Using the original dataset, I concluded that there was a statistically significant different between Republicans and Democrats when it came to the impact of campaign spending on votes received. In my model the constant associated with being a Democrat was higher than that of being a Republican, however, the effectiveness of campaign spend for Republicans

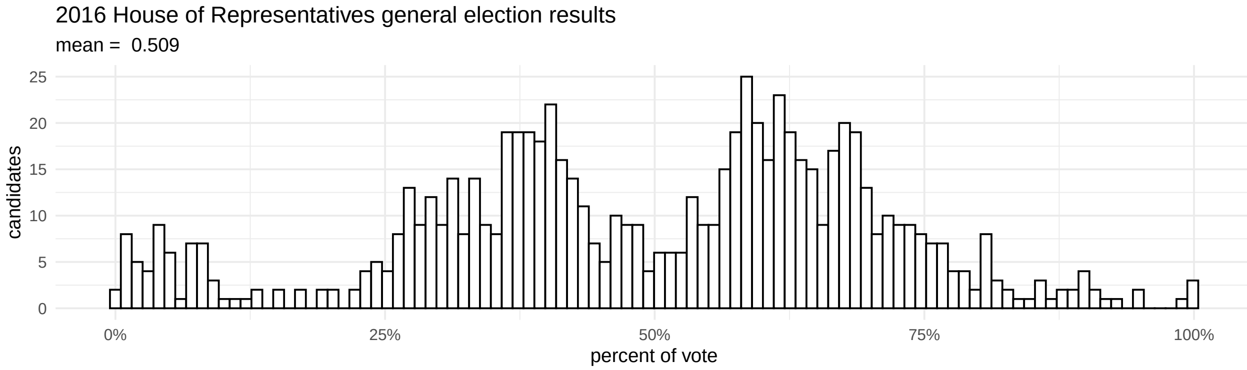

Figure 4: Revised Voting Percentages Again

outpaced that of Democrats. When I repeat my work using this data, that difference disappears. With this data, I found no difference between Republicans and Democrats. In both cases there was a significant difference between being affiliated with one of the major parties or not. With the cleaner data, the total residuals dropped and adjusted R^2 went from .258 to .426.