DATA 698: Masters Research Project

Gabriella Martinez & Gabriel Campos

Last edited December 13, 2024

Contents

Packages

```r
#load libraries
library(car)
library(caret)
library(corrplot)
library(ggplot2)
library(janitor)
library(Hmisc)
library(randomForest)
library(reshape2)
library(rvest)
library(tidyverse)
library(tidycensus)
library(httr)
library(xml2)
library(kableExtra)
```

```r
# Define the path to the Key folder
api_key_file_path <- file.path(".", "Key", "api_key.txt")

# Read the API key from the file
api_key <- readLines(api_key_file_path, warn = FALSE)

# Print the API key (for debugging purposes; avoid doing this in production)
cat("API Key:", api_key, "\n")
```

```
## API Key: 60a577bbf5f66f4985ca219cc061a2a6a7d7b52f
```

Data Load

Election Data

Data was source from **Harvard Dataverse**, an open-source data repository platform developed by Harvard University. It is designed to facilitate the sharing, preservation, and citation of research data across various disciplines. Harvard Dataverse is part of the larger Dataverse Project, which provides an open-source platform for institutions to host their own Dataverse installations. The data was extracted to *countypres_2000-2020.csv* and loaded onto our projects github.

```r
# Data sourced
#https://dataverse.harvard.edu/dataset.xhtml?persistentId=doi:10.7910/DVN/VOQCHQ
# Retrieved from github and stored onto elections dataframe

elect_df <- read_csv(paste0(git_url,"countypres_2000-2020.csv"))
```

```
## Rows: 72617 Columns: 12
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (8): state, state_po, county_name, county_fips, office, candidate, party...
## dbl (4): year, candidatevotes, totalvotes, version
##
## 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.
```

```r
#glimpse(elections)
```

Data Cleaning (Elections)

```r
#identify empty and NA values. 57 NA values in the county_fips column
colSums(elect_df == "" | is.na(elect_df))
```

```
##           year          state       state_po    county_name    county_fips
##              0              0              0              0             57
##         office      candidate          party candidatevotes     totalvotes
##              0              0              0              0              0
##        version           mode
##              0              0
```

```r
elect_df %>%
  filter(is.na(county_fips))
```

```
## # A tibble: 57 x 12
##     year state        state_po county_name    county_fips office candidate party
##    <dbl> <chr>        <chr>    <chr>          <chr>       <chr>  <chr>     <chr>
##  1  2000 CONNECTICUT  CT       STATEWIDE WRI~ <NA>        US PR~ AL GORE   DEMO~
##  2  2000 MAINE        ME       MAINE UOCAVA   <NA>        US PR~ AL GORE   DEMO~
##  3  2000 RHODE ISLAND RI       FEDERAL PRECI~ <NA>        US PR~ AL GORE   DEMO~
##  4  2000 CONNECTICUT  CT       STATEWIDE WRI~ <NA>        US PR~ GEORGE W~ REPU~
##  5  2000 MAINE        ME       MAINE UOCAVA   <NA>        US PR~ GEORGE W~ REPU~
##  6  2000 RHODE ISLAND RI       FEDERAL PRECI~ <NA>        US PR~ GEORGE W~ REPU~
##  7  2000 CONNECTICUT  CT       STATEWIDE WRI~ <NA>        US PR~ RALPH NA~ GREEN
##  8  2000 MAINE        ME       MAINE UOCAVA   <NA>        US PR~ RALPH NA~ GREEN
##  9  2000 RHODE ISLAND RI       FEDERAL PRECI~ <NA>        US PR~ RALPH NA~ GREEN
## 10  2000 CONNECTICUT  CT       STATEWIDE WRI~ <NA>        US PR~ OTHER     OTHER
## # i 47 more rows
## # i 4 more variables: candidatevotes <dbl>, totalvotes <dbl>, version <dbl>,
## #   mode <chr>
```

```r
elect_df %>%
  filter(is.na(county_fips)) %>%
  select(state_po, county_name, county_fips) %>%
  distinct()
```

```
## # A tibble: 4 x 3
##   state_po county_name          county_fips
##   <chr>    <chr>                <chr>
## 1 CT       STATEWIDE WRITEIN    <NA>
## 2 ME       MAINE UOCAVA         <NA>
## 3 RI       FEDERAL PRECINCT     <NA>
## 4 DC       DISTRICT OF COLUMBIA <NA>
```

```r
#clean elections data
elect_data_df <- elect_df %>%
  #new name = old name
  rename(state_abbr = state_po, pol_identity = party, FIPS = county_fips) %>%
  mutate(FIPS = ifelse(state_abbr == "DC", "11001", FIPS))

#there are 52 NAs remaining
elect_nas_df <- elect_data_df %>%
```

```r
  filter(is.na(FIPS))

elect_nas_df %>%
  count(state_abbr, county_name)
```

```
## # A tibble: 3 x 3
##   state_abbr county_name           n
##   <chr>      <chr>             <int>
## 1 CT         STATEWIDE WRITEIN    16
## 2 ME         MAINE UOCAVA         16
## 3 RI         FEDERAL PRECINCT     20
```

The remaining **NA** values in the **FIPS** column are votes assigned at a state-wide level, not to any count. The *"MAINE UOCAVA"* county record for the state of Maine represents the count of votes from Uniformed Service & Overseas **(UOCAVA)** Voters. The "STATEWIDE WRITEIN" for Connecticut represents the count of votes for self-selected candidates not on the presidential ballot. It is unclear what the *"FEDERAL PRECINCT"* for the state of Rhode Island exactly represents. Either way, our analysis will be conducted at the county level, so these records cannot be used.

Next we will assess the effect that removing these votes will have on our overall analysis.

```r
#nas
nrow(elect_nas_df)
```

```
## [1] 52
```

```r
# Determine the total number of records in the table.
nrow(elect_nas_df)
```

```
## [1] 52
```

```r
round(nrow(elect_nas_df)/nrow(elect_data_df)*100,3)
```

```
## [1] 0.072
```

```r
# Determine the total number of votes cast across all counties in all elections.
elect_vt_cnt_df <- elect_data_df %>%
  summarise(count= sum(candidatevotes))

elect_vt_cnt_df
```

```
## # A tibble: 1 x 1
##       count
##       <dbl>
## 1 782944050
```

```r
# Determine how many votes are associated with state-level counts
elect_null_fips_cnt_df <- elect_nas_df %>%
  summarise(count=sum(candidatevotes))

elect_null_fips_cnt_df
```

```
## # A tibble: 1 x 1
##   count
##   <dbl>
## 1 13009
```

```r
round((elect_null_fips_cnt_df$count/elect_vt_cnt_df$count)*100,3)
```

```
## [1] 0.002
```

There were 52 records with state-level counts and null FIPS values in the data, representing 13009 votes. This amounts to 0.072% of the total records and 0.002% of the total votes.

The records with state-level counts and null FIPS values represent a small percentage of the total, and they are unlikely to change the overall analysis. Given our assessment, the records will be removed.

```
#transform data- drop NAs, keep dem and gop only, group records for each candidate by county and year
elect_cand_vt_df <- elect_data_df %>%
  filter(!is.na(FIPS), pol_identity %in% c('DEMOCRAT', 'REPUBLICAN')) %>%
  group_by(FIPS,county_name,
           state, candidate,
           year, pol_identity,
           totalvotes) %>%
  summarise(candidate_votes = sum(candidatevotes)) %>%
  ungroup() %>%
  arrange(FIPS, year)
```

```
## `summarise()` has grouped output by 'FIPS', 'county_name', 'state',
## 'candidate', 'year', 'pol_identity'. You can override using the `.groups`
## argument.
```

```
#spread the candidate votes values
elect_pivot_df <- elect_cand_vt_df %>%
   pivot_wider(id_cols = c(year, FIPS, county_name, state, totalvotes),
               names_from = pol_identity,
               values_from = candidate_votes) %>%
  rename(votes_dem = DEMOCRAT,  votes_gop = REPUBLICAN
        #votes_other = OTHER,votes_grn = GREEN, votes_lib = LIBERTARIAN
        )
```

Census Bureau data

About Census Bureau American Community Survey (ACS) data https://www.census.gov/programs-surveys/acs/guidance/estimates.html

Citizen Voting Age Population

Citizen Voting Age Population, Census Bureau population estimates generated using the American Community Survey

```
#CVAP- Citizen Voting Age Population, Census Bureau population estimates
#generated using the American Community Survey

#https://www.census.gov/programs-surveys/decennial-census/about/voting-rights
#/cvap.2010.html#list-tab-1518558936 (2008)
cens_cvap2008 <-
  read_csv(paste0(git_url,
                  "CountyCVAP_2006-2010.csv"
                  #,"?token=GHSAT0AAAAAACXYKDAYQCHUVJY2V6BVWU7SZXPAZJQ"
                  )) %>%
  rename_with(tolower) %>%
  mutate(year=2008)

#https://www.census.gov/programs-surveys/decennial-census/about/voting-rights
#/cvap.2014.html#list-tab-1518558936 (2012)
```

```r
cens_cvap2012 <-
  read_csv(paste0(git_url,
           "CountyCVAP_2010-2014.csv"
           #,"?token=GHSAT0AAAAAACXYKDAYHOL27SGWSEL2AS6IZXPAYSQ"
           )) %>%
  rename_with(tolower) %>%
  mutate(year=2012)

#https://www.census.gov/programs-surveys/decennial-census/about/voting-rights
#/cvap/2014-2018-CVAP.html (2016)
cens_cvap2016 <-
  read_csv(paste0(git_url,
                  "CountyCVAP_2014-2018.csv"
                  #,"?token=GHSAT0AAAAAACXYKDAZJU7ABMJMRNP5WOSIZXPATUQ"
                  )) %>%
  mutate(year=2016)

#https://www.census.gov/programs-surveys/decennial-census/about/voting-rights
#/cvap/2017-2021-CVAP.html (2020)
cens_cvap2020 <-
  read_csv(paste0(git_url,
                  "CountyCVAP_2017-2021.csv"
                  #,"?token=GHSAT0AAAAAACXYKDAYJWVR6SZPSH4NRMSSZXPASSQ"
                  )) %>%
  mutate(year=2020)

cens_cvap_df <- rbind(cens_cvap2008,
                      cens_cvap2012,
                      cens_cvap2016,
                      cens_cvap2020) %>%
  filter(lntitle == 'Total', !str_detect(geoname, "Puerto Rico")) %>%
  mutate(FIPS = str_sub(geoid, -5)) %>%
  select(c('year', 'FIPS', 'geoname', 'cvap_est'))

#identify empty and NA values
colSums(cens_cvap_df == "" | is.na(cens_cvap_df))
```

```r
vot_info_df <- left_join(elect_pivot_df, cens_cvap_df, by = c("FIPS", "year"))

vot_info_df
```

Merge with Election data

```
## # A tibble: 18,928 x 9
##     year FIPS  county_name state totalvotes votes_dem votes_gop geoname cvap_est
##    <dbl> <chr> <chr>       <chr>      <dbl>     <dbl>     <dbl> <chr>      <dbl>
##  1  2000 01001 AUTAUGA     ALAB~      17208      4942     11993 <NA>          NA
##  2  2004 01001 AUTAUGA     ALAB~      20081      4758     15196 <NA>          NA
##  3  2008 01001 AUTAUGA     ALAB~      23641      6093     17403 Autaug~    38010
##  4  2012 01001 AUTAUGA     ALAB~      23932      6363     17379 Autaug~    40545
##  5  2016 01001 AUTAUGA     ALAB~      24973      5936     18172 Autaug~    41305
##  6  2020 01001 AUTAUGA     ALAB~      27770      7503     19838 Autaug~    43905
##  7  2000 01003 BALDWIN     ALAB~      56480     13997     40872 <NA>          NA
```

```
## 8  2004 01003 BALDWIN     ALAB~      69320      15599      52971 <NA>          NA
## 9  2008 01003 BALDWIN     ALAB~      81413      19386      61271 Baldwi~   130865
## 10  2012 01003 BALDWIN     ALAB~      85338      18424      66016 Baldwi~   144120
## # i 18,918 more rows
```

```
ls_states <- sort(str_to_title(unique(vot_info_df$state)))
```

```
#identify empty and NA values
colSums(vot_info_df == "" | is.na(vot_info_df))
```

```
##        year        FIPS county_name       state  totalvotes   votes_dem
##           0           0           0           0           0           0
##   votes_gop     geoname    cvap_est
##           0        6467        6467
```

```
vot_info_NAs_df <- vot_info_df %>%
  filter(is.na(geoname), is.na(cvap_est))

vot_info_NAs_df
```

```
## # A tibble: 6,467 x 9
##     year FIPS  county_name state totalvotes votes_dem votes_gop geoname cvap_est
##    <dbl> <chr> <chr>       <chr>      <dbl>     <dbl>     <dbl> <chr>      <dbl>
##  1  2000 01001 AUTAUGA     ALAB~      17208      4942     11993 <NA>          NA
##  2  2004 01001 AUTAUGA     ALAB~      20081      4758     15196 <NA>          NA
##  3  2000 01003 BALDWIN     ALAB~      56480     13997     40872 <NA>          NA
##  4  2004 01003 BALDWIN     ALAB~      69320     15599     52971 <NA>          NA
##  5  2000 01005 BARBOUR     ALAB~      10395      5188      5096 <NA>          NA
##  6  2004 01005 BARBOUR     ALAB~      10777      4832      5899 <NA>          NA
##  7  2000 01007 BIBB        ALAB~       7101      2710      4273 <NA>          NA
##  8  2004 01007 BIBB        ALAB~       7600      2089      5472 <NA>          NA
##  9  2000 01009 BLOUNT      ALAB~      17973      4977     12667 <NA>          NA
## 10  2004 01009 BLOUNT      ALAB~      21504      3938     17386 <NA>          NA
## # i 6,457 more rows
```

```
unique(vot_info_NAs_df$year)
```

```
## [1] 2000 2004 2008 2012 2016 2020
```

```
vot_info_df <- vot_info_df %>%
  filter(year >= 2008)

vot_info_NAs_2df <- vot_info_df %>%
  filter(is.na(geoname), is.na(cvap_est))

vot_info_NAs_2df
```

```
## # A tibble: 158 x 9
##     year FIPS  county_name state totalvotes votes_dem votes_gop geoname cvap_est
##    <dbl> <chr> <chr>       <chr>      <dbl>     <dbl>     <dbl> <chr>      <dbl>
##  1  2008 02001 DISTRICT 1  ALAS~       6970      2597      4149 <NA>          NA
##  2  2012 02001 DISTRICT 1  ALAS~       7722      1518      5899 <NA>          NA
##  3  2016 02001 DISTRICT 1  ALAS~       6638      2573      3180 <NA>          NA
##  4  2020 02001 DISTRICT 1  ALAS~       7314      3477      3511 <NA>          NA
##  5  2008 02002 DISTRICT 2  ALAS~       7735      3468      4029 <NA>          NA
##  6  2012 02002 DISTRICT 2  ALAS~       9058      3096      5509 <NA>          NA
##  7  2016 02002 DISTRICT 2  ALAS~       5492      1585      3188 <NA>          NA
```

```
## 8  2020 02002 DISTRICT 2  ALAS~       6136      2104      3674 <NA>          NA
## 9  2008 02003 DISTRICT 3  ALAS~       8767      5657      2829 <NA>          NA
## 10 2012 02003 DISTRICT 3  ALAS~       6069      2034      3769 <NA>          NA
## # i 148 more rows
```

```
vot_info_df <- vot_info_df %>%
  filter(state != "ALASKA")

vot_info_NAs_3df <- vot_info_df %>%
  filter(is.na(geoname), is.na(cvap_est))

vot_info_NAs_3df
```

```
## # A tibble: 6 x 9
##    year FIPS  county_name state  totalvotes votes_dem votes_gop geoname cvap_est
##   <dbl> <chr> <chr>       <chr>       <dbl>     <dbl>     <dbl> <chr>      <dbl>
## 1  2008 36000 KANSAS CITY MISSO~     153219    120102     31854 <NA>          NA
## 2  2012 36000 KANSAS CITY MISSO~     136802    105670     29509 <NA>          NA
## 3  2016 36000 KANSAS CITY MISSO~     128601     97735     24654 <NA>          NA
## 4  2020 36000 KANSAS CITY MISSO~     136645    107660     26393 <NA>          NA
## 5  2012 51515 BEDFORD     VIRGI~       2805      1225      1527 <NA>          NA
## 6  2016 51515 BEDFORD     VIRGI~          0         0         0 <NA>          NA
```

```
vot_info_clean_df <- vot_info_df %>%
  filter(FIPS %in% c('29095', '36000', '51019', '51515')) %>%
  arrange(year, FIPS)

vot_info_clean_df
```

```
## # A tibble: 15 x 9
##     year FIPS  county_name state totalvotes votes_dem votes_gop geoname cvap_est
##    <dbl> <chr> <chr>       <chr>      <dbl>     <dbl>     <dbl> <chr>      <dbl>
##  1  2008 29095 JACKSON     MISS~     186047     90722     92833 Jackso~   481045
##  2  2008 36000 KANSAS CITY MISS~     153219    120102     31854 <NA>          NA
##  3  2008 51019 BEDFORD     VIRG~      35830     11017     24420 Bedfor~    51755
##  4  2008 51515 BEDFORD     VIRG~       2734      1208      1497 Bedfor~     4595
##  5  2012 29095 JACKSON     MISS~     174764     78283     93199 Jackso~   493440
##  6  2012 36000 KANSAS CITY MISS~     136802    105670     29509 <NA>          NA
##  7  2012 51019 BEDFORD     VIRG~      37425     10209     26679 Bedfor~    58850
##  8  2012 51515 BEDFORD     VIRG~       2805      1225      1527 <NA>          NA
##  9  2016 29095 JACKSON     MISS~     173275     71237     91557 Jackso~   506340
## 10  2016 36000 KANSAS CITY MISS~     128601     97735     24654 <NA>          NA
## 11  2016 51019 BEDFORD     VIRG~      42525      9768     30659 Bedfor~    61205
## 12  2016 51515 BEDFORD     VIRG~          0         0         0 <NA>          NA
## 13  2020 29095 JACKSON     MISS~     196418     92182    100142 Jackso~   523040
## 14  2020 36000 KANSAS CITY MISS~     136645    107660     26393 <NA>          NA
## 15  2020 51019 BEDFORD     VIRG~      48669     12176     35600 Bedfor~    62435
```

```
vot_info_clean_df %>%
  count(FIPS, state, county_name, geoname) %>%
  filter(geoname %in% c("Jackson County, Missouri", "Bedford County, Virginia")) %>%
  select(-n)
```

```
## # A tibble: 2 x 4
##   FIPS  state    county_name geoname
##   <chr> <chr>    <chr>       <chr>
```

```
## 1 29095 MISSOURI JACKSON     Jackson County, Missouri
## 2 51019 VIRGINIA BEDFORD     Bedford County, Virginia
```

```r
# Define the counties to filter and group data by year and state
vot_co_grps_df <- vot_info_df %>%
  filter(FIPS %in% c('29095', '36000', '51019', '51515')) %>%
  group_by(year, state) %>%
  summarise(    # Concatenate FIPS codes and county names
    FIPS = paste(unique(FIPS), collapse = ", "),
    county_name = paste(unique(county_name), collapse = ", "),
            across(where(is.numeric), sum, na.rm = TRUE)) %>%
  mutate(geoname = case_when(state == "MISSOURI" ~ "Jackson County, Missouri",
                             state == "VIRGINIA" ~ "Bedford County, Virginia"))
```

```
## Warning: There was 1 warning in `summarise()`.
## i In argument: `across(where(is.numeric), sum, na.rm = TRUE)`.
## i In group 1: `year = 2008` and `state = "MISSOURI"`.
## Caused by warning:
## ! The `...` argument of `across()` is deprecated as of dplyr 1.1.0.
## Supply arguments directly to `.fns` through an anonymous function instead.
##
##   # Previously
##   across(a:b, mean, na.rm = TRUE)
##
##   # Now
##   across(a:b, \(x) mean(x, na.rm = TRUE))

## `summarise()` has grouped output by 'year'. You can override using the
## `.groups` argument.
```

```r
vot_co_grps_df
```

```
## # A tibble: 8 x 9
## # Groups:   year [4]
##    year state  FIPS  county_name totalvotes votes_dem votes_gop cvap_est geoname
##   <dbl> <chr>  <chr> <chr>            <dbl>     <dbl>     <dbl>    <dbl> <chr>
## 1  2008 MISSO~ 2909~ JACKSON, K~     339266    210824    124687   481045 Jackso~
## 2  2008 VIRGI~ 5101~ BEDFORD          38564     12225     25917    56350 Bedfor~
## 3  2012 MISSO~ 2909~ JACKSON, K~     311566    183953    122708   493440 Jackso~
## 4  2012 VIRGI~ 5101~ BEDFORD          40230     11434     28206    58850 Bedfor~
## 5  2016 MISSO~ 2909~ JACKSON, K~     301876    168972    116211   506340 Jackso~
## 6  2016 VIRGI~ 5101~ BEDFORD          42525      9768     30659    61205 Bedfor~
## 7  2020 MISSO~ 2909~ JACKSON, K~     333063    199842    126535   523040 Jackso~
## 8  2020 VIRGI~ 51019 BEDFORD          48669     12176     35600    62435 Bedfor~
```

```r
#remove the previous observations
vot_info_df <- vot_info_df %>%
  filter(!FIPS %in% c('29095', '36000', '51019', '51515'))

#replace with the calculated observations
vot_info_df <- rbind(vot_info_df, vot_co_grps_df)


ls_FIPS <- unique(vot_info_df$FIPS)
```

```r
length(ls_FIPS)
```

Clean up

```
## [1] 3114
```

```r
co_names <- vot_info_df %>%
  group_by(state, county_name) %>%
  mutate(county_name = str_to_title(county_name),
         state = str_to_title(state)) %>%
  summarise(n=n())
```

```
## `summarise()` has grouped output by 'state'. You can override using the
## `.groups` argument.
```

```r
length(co_names)
```

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

```r

vot_info_df %>%
  group_by(year) %>%
  summarise(total_dem = scales::comma(sum(votes_dem)),
            total_gop = scales::comma(sum(votes_gop))) %>%
  mutate(result = if_else(total_gop > total_dem,
                          "Republican Party","Democratic Party")) %>%
  kableExtra::kable() %>%
  kableExtra::kable_minimal()
```

Popular Vote

year	total_dem	total_gop	result
2008	69,324,684	59,734,854	Democratic Party
2012	65,628,040	60,500,800	Democratic Party
2016	65,724,133	62,814,943	Democratic Party
2020	81,109,594	74,028,963	Democratic Party

```r
rm(list = ls(pattern = "^elect_|^cens_"))
```

```r

vot_info_df <- vot_info_df %>%
  group_by(state, year) %>%
  summarise(totalvotes = sum(totalvotes),
            votes_dem = sum(votes_dem),
            votes_gop = sum(votes_gop),
            cvap_est = sum(cvap_est)) %>%
  ungroup() %>%
  arrange(state, year)
```

Aggregate by State

```
## `summarise()` has grouped output by 'state'. You can override using the
## `.groups` argument.
```

```r
#49 states + DC, Alaska has been removed
length(unique(vot_info_df$state))
```

```
## [1] 50
```

```r
# Assuming your data frame is `state_data`
vot_info_df %>%
  kableExtra::kable() %>%
  kableExtra::kable_minimal()
```

state	year	totalvotes	votes_dem	votes_gop	cvap_est
ALABAMA	2008	2099819	813479	1266546	3481380
ALABAMA	2012	2070353	795696	1255925	3600120
ALABAMA	2016	2123367	729547	1318250	3671115
ALABAMA	2020	2323282	849624	1441170	3782980
ARIZONA	2008	2293475	1034707	1230111	4110885
ARIZONA	2012	2299254	1025232	1233654	4444230
ARIZONA	2016	2604277	1161167	1252401	4812760
ARIZONA	2020	3385294	1672143	1661686	5000090
ARKANSAS	2008	1086617	422310	638017	2090155
ARKANSAS	2012	1069468	394409	647744	2152350
ARKANSAS	2016	1129896	380494	684872	2195865
ARKANSAS	2020	1219069	423932	760647	2211560
CALIFORNIA	2008	13561900	8274473	5011781	22329310
CALIFORNIA	2012	13038547	7854285	4839958	23881285
CALIFORNIA	2016	14181595	8753788	4483810	25232630
CALIFORNIA	2020	17500881	11110250	6006429	25916215
COLORADO	2008	2401361	1288576	1073589	3403825
COLORADO	2012	2569217	1322998	1185050	3679115
COLORADO	2016	2780220	1338870	1202484	3979310
COLORADO	2020	3256980	1804352	1364607	4194465
CONNECTICUT	2008	1647085	1000291	628041	2493100
CONNECTICUT	2012	1557885	905083	634892	2564230
CONNECTICUT	2016	1644920	897572	673215	2600980
CONNECTICUT	2020	1823857	1080831	714717	2638020
DELAWARE	2008	412412	255459	152374	638160
DELAWARE	2012	413937	242584	165484	674335
DELAWARE	2016	442997	235603	185127	704105
DELAWARE	2020	504010	296268	200603	733785
DISTRICT OF COLUMBIA	2008	265853	245800	17367	435875
DISTRICT OF COLUMBIA	2012	293764	267070	21381	475400
DISTRICT OF COLUMBIA	2016	311268	282830	12723	511190
DISTRICT OF COLUMBIA	2020	344356	317323	18586	512335
FLORIDA	2008	8391639	4282366	4046212	12812550
FLORIDA	2012	8474179	4237756	4163447	13673530
FLORIDA	2016	9420039	4504975	4617886	14724115
FLORIDA	2020	11067456	5297045	5668731	15394170
GEORGIA	2008	3925278	1844137	2048744	6476095
GEORGIA	2012	3900050	1773827	2078688	6882855
GEORGIA	2016	4114711	1877963	2089104	7254710
GEORGIA	2020	4998482	2474507	2461837	7568140

HAWAII	2008	452742	325201	120429	941525
HAWAII	2012	434221	306266	120937	989180
HAWAII	2016	428937	266891	128847	1016485
HAWAII	2020	574457	366127	196855	1045190
IDAHO	2008	655032	236440	403012	1056005
IDAHO	2012	652274	212787	420911	1116700
IDAHO	2016	690433	189765	409055	1192740
IDAHO	2020	867361	287021	554119	1298405
ILLINOIS	2008	5523051	3419673	2031527	8717360
ILLINOIS	2012	5241891	3019512	2135102	8939910
ILLINOIS	2016	5558707	3090729	2146015	9055150
ILLINOIS	2020	6033744	3471915	2446891	9133875
INDIANA	2008	2751054	1374039	1345648	4649360
INDIANA	2012	2624534	1152887	1420543	4773195
INDIANA	2016	2734958	1033126	1557286	4876215
INDIANA	2020	3033121	1242416	1729519	4964975
IOWA	2008	1536820	828940	682379	2222845
IOWA	2012	1582180	822544	730617	2273775
IOWA	2016	1566031	653669	800983	2312630
IOWA	2020	1690871	759061	902009	2348205
KANSAS	2008	1235872	514765	699655	1989370
KANSAS	2012	1156254	439908	689809	2043800
KANSAS	2016	1184403	427005	671018	2077570
KANSAS	2020	1372303	570323	771406	2110075
KENTUCKY	2008	1826508	751985	1048462	3189860
KENTUCKY	2012	1797212	679370	1087190	3281575
KENTUCKY	2016	1924149	628854	1202971	3338185
KENTUCKY	2020	2134996	772285	1326418	3378365
LOUISIANA	2008	1959085	781574	1148015	3241175
LOUISIANA	2012	1994065	809141	1152262	3385550
LOUISIANA	2016	2029032	780154	1178638	3452750
LOUISIANA	2020	2148062	856034	1255776	3455660
MAINE	2008	731163	421923	295273	1029250
MAINE	2012	710126	399235	291418	1044330
MAINE	2016	743941	354718	334945	1059545
MAINE	2020	822534	430473	359899	1082850
MARYLAND	2008	2631596	1629467	959862	3964245
MARYLAND	2012	2707327	1677844	971869	4142465
MARYLAND	2016	2781446	1677928	943169	4262390
MARYLAND	2020	3037031	1985023	976414	4388175
MASSACHUSETTS	2008	3081336	1904103	1108885	4602190
MASSACHUSETTS	2012	3167767	1921290	1188314	4799870
MASSACHUSETTS	2016	3274555	1995196	1090893	4964685
MASSACHUSETTS	2020	3658005	2382202	1167202	5105065
MICHIGAN	2008	5001766	2872579	2048639	7266075
MICHIGAN	2012	4730961	2564569	2115256	7347660
MICHIGAN	2016	4799284	2268839	2279543	7472660
MICHIGAN	2020	5539302	2804040	2649852	7592235
MINNESOTA	2008	2910369	1573354	1275409	3783745
MINNESOTA	2012	2936561	1546167	1320225	3920505

MINNESOTA	2016	2944813	1367716	1322951	4037275
MINNESOTA	2020	3277171	1717077	1484065	4161265
MISSISSIPPI	2008	1285259	554662	724497	2146430
MISSISSIPPI	2012	1285584	562949	710746	2201510
MISSISSIPPI	2016	1209357	485131	700714	2228665
MISSISSIPPI	2020	1313759	539398	756764	2225530
MISSOURI	2008	2925205	1441911	1445814	4384200
MISSOURI	2012	2757312	1223796	1482440	4503005
MISSOURI	2016	2807381	1071068	1594511	4585990
MISSOURI	2020	3025962	1253014	1718736	4635925
MONTANA	2008	496072	232156	243860	742830
MONTANA	2012	483932	201839	267928	774020
MONTANA	2016	497147	177709	279240	804260
MONTANA	2020	603640	244786	343602	835520
NEBRASKA	2008	801281	333319	452979	1284805
NEBRASKA	2012	794379	302081	475064	1324485
NEBRASKA	2016	844227	284494	495961	1358805
NEBRASKA	2020	951712	374583	556846	1391790
NEVADA	2008	967848	533736	412827	1701525
NEVADA	2012	1014918	531373	463567	1830225
NEVADA	2016	1125385	539260	512058	1973640
NEVADA	2020	1404911	703314	669608	2099150
NEW HAMPSHIRE	2008	710970	384826	316534	987480
NEW HAMPSHIRE	2012	710931	369561	329918	1013645
NEW HAMPSHIRE	2016	744296	348526	345790	1048205
NEW HAMPSHIRE	2020	803833	424937	365660	1079640
NEW JERSEY	2008	3838498	2215422	1613207	5838030
NEW JERSEY	2012	3640292	2125101	1477568	6002830
NEW JERSEY	2016	3874046	2148278	1601933	6117610
NEW JERSEY	2020	4549353	2608335	1883274	6384675
NEW MEXICO	2008	830158	472422	346832	1383790
NEW MEXICO	2012	783758	415335	335788	1448040
NEW MEXICO	2016	798319	385234	319667	1485495
NEW MEXICO	2020	923965	501614	401894	1522115
NEW YORK	2008	7591233	4769700	2742298	13004820
NEW YORK	2012	7061925	4324228	2223397	13425020
NEW YORK	2016	7707363	4547562	2814589	13686695
NEW YORK	2020	8661735	5230985	3244798	14182055
NORTH CAROLINA	2008	4310789	2142651	2128474	6607015
NORTH CAROLINA	2012	4505372	2178391	2270395	7015220
NORTH CAROLINA	2016	4741564	2189316	2362631	7413170
NORTH CAROLINA	2020	5524802	2684292	2758773	7615615
NORTH DAKOTA	2008	316621	141278	168601	503755
NORTH DAKOTA	2012	322932	124966	188320	535565
NORTH DAKOTA	2016	344360	93758	216794	562650
NORTH DAKOTA	2020	361819	114902	235595	571035
OHIO	2008	5698048	2933388	2674491	8547620
OHIO	2012	5580822	2827621	2661407	8678500
OHIO	2016	5496487	2394164	2841005	8797920
OHIO	2020	5922202	2679165	3154834	8909350

OKLAHOMA	2008	1462661	502496	960165	2647100
OKLAHOMA	2012	1334872	443547	891325	2749200
OKLAHOMA	2016	1452992	420375	949136	2819185
OKLAHOMA	2020	1560699	503890	1020280	2852300
OREGON	2008	1827864	1037291	738475	2692180
OREGON	2012	1789270	970488	754175	2830545
OREGON	2016	2001336	1002106	782403	3002260
OREGON	2020	2374321	1340383	958448	3135110
PENNSYLVANIA	2008	5977981	3266523	2649934	9475240
PENNSYLVANIA	2012	5742040	2990274	2680434	9676880
PENNSYLVANIA	2016	6115402	2926441	2970733	9748290
PENNSYLVANIA	2020	6915283	3458229	3377674	9893015
RHODE ISLAND	2008	471766	296571	165391	761675
RHODE ISLAND	2012	445719	279409	157151	773770
RHODE ISLAND	2016	463416	251888	180490	789060
RHODE ISLAND	2020	516383	306210	199837	819450
SOUTH CAROLINA	2008	1920969	862449	1034896	3312710
SOUTH CAROLINA	2012	1964118	865941	1071645	3515420
SOUTH CAROLINA	2016	2103027	855373	1155389	3731345
SOUTH CAROLINA	2020	2513329	1091541	1385103	3836595
SOUTH DAKOTA	2008	377708	170924	203054	590660
SOUTH DAKOTA	2012	363815	145039	210610	616000
SOUTH DAKOTA	2016	370093	117458	227721	635415
SOUTH DAKOTA	2020	422609	150471	261043	645585
TENNESSEE	2008	2600124	1087437	1479178	4582675
TENNESSEE	2012	2458577	960709	1462330	4785590
TENNESSEE	2016	2508027	870695	1522925	4964900
TENNESSEE	2020	3053851	1143711	1852475	5138905
TEXAS	2008	8077795	3528633	4479328	15277005
TEXAS	2012	7993851	3308124	4569843	16529510
TEXAS	2016	8969226	3877868	4685047	17859500
TEXAS	2020	11315056	5259126	5890347	18729795
UTAH	2008	952370	327670	596030	1696055
UTAH	2012	1017440	251813	740600	1831250
UTAH	2016	1131430	310676	515231	1982910
UTAH	2020	1495354	560282	865139	2143405
VERMONT	2008	325046	219262	98974	481700
VERMONT	2012	299290	199239	92698	491550
VERMONT	2016	315077	178573	95369	494675
VERMONT	2020	370826	242826	112708	512080
VIRGINIA	2008	3723260	1959532	1725005	5578940
VIRGINIA	2012	3854489	1971820	1822522	5877505
VIRGINIA	2016	3984631	1981473	1769443	6096235
VIRGINIA	2020	4462600	2413568	1962430	6256040
WASHINGTON	2008	3036878	1750848	1229216	4593025
WASHINGTON	2012	3125516	1755396	1290670	4866940
WASHINGTON	2016	3209214	1742718	1221747	5173965
WASHINGTON	2020	4087631	2369612	1584651	5413420
WEST VIRGINIA	2008	713451	303857	397466	1440470
WEST VIRGINIA	2012	670440	238269	417655	1456980

WEST VIRGINIA	2016	713051	188794	489371	1442025
WEST VIRGINIA	2020	794652	235984	545382	1422125
WISCONSIN	2008	2983417	1677211	1262393	4161005
WISCONSIN	2012	3071434	1620985	1410966	4269765
WISCONSIN	2016	2975753	1381823	1404440	4347400
WISCONSIN	2020	3297352	1630673	1610065	4437215
WYOMING	2008	256035	82868	164958	405095
WYOMING	2012	249061	69286	170962	427305
WYOMING	2016	255849	55973	174419	432285
WYOMING	2020	278503	73491	193559	431010

```
vot_info_fin <- vot_info_df %>%
  mutate(#voters who did not choose the Democratic or Republican party
         votes_other = totalvotes - votes_dem - votes_gop,
         #voter share attributes
         voter_share_major_party = (votes_dem + votes_gop) / totalvotes,
         voter_share_dem = votes_dem/totalvotes,
         voter_share_gop = votes_gop/totalvotes,
         voter_share_other = votes_other/totalvotes,
         #raw differences
         rawdiff_dem_vs_gop = votes_dem - votes_gop,
         rawdiff_gop_vs_dem = votes_gop - votes_dem,
         rawdiff_dem_vs_other = votes_dem - votes_other,
         rawdiff_gop_vs_other = votes_gop - votes_other,
         rawdiff_other_vs_dem = votes_other - votes_dem,
         rawdiff_other_vs_gop = votes_other - votes_gop,
         #percentage difference
         pctdiff_dem_vs_gop =
           (votes_dem - votes_gop) / totalvotes,
         pctdiff_gop_vs_dem =
           (votes_gop - votes_dem) / totalvotes,
         pctdiff_dem_vs_other =
           (votes_dem - votes_other) / totalvotes,
         pctdiff_gop_vs_other =
           (votes_gop - votes_other) / totalvotes,
         pctdiff_other_vs_dem =
           (votes_other - votes_dem) / totalvotes,
         pctdiff_other_vs_gop =
           (votes_other - votes_gop) / totalvotes,
         #voter turnout
         voter_turnout = totalvotes/cvap_est,
         voter_turnout_majparty =
           (votes_dem+votes_gop)/cvap_est,
         voter_turnout_dem = votes_dem/cvap_est,
         voter_turnout_gop = votes_gop/cvap_est,
         voter_turnout_other =votes_other/cvap_est,
         # get winning political party
         winning_party =
           case_when(votes_dem > votes_gop &
                       votes_dem > votes_other ~ "Democratic Party",
                     votes_gop > votes_dem &
                       votes_gop > votes_other ~ "Republican Party",
```

```
                      votes_other > votes_dem &
                        votes_other > votes_gop ~ "Other Party"),
         pct_margin_of_victory =
           case_when(winning_party == "Democratic Party"
                      ~ round(
                        ((votes_dem - votes_gop) / totalvotes)
                        *100,3), #votes_dem > votes_gop
                      winning_party == "Republican Party"
                      ~ round(
                        ((votes_gop - votes_dem) / totalvotes)
                        *100,3), #votes_gop > votes_dem
                      ),
         # create binary outcome version of the variable for model use
         winning_party_binary =
           case_when(votes_dem > votes_gop &
                        votes_dem > votes_other ~ 0,
                      votes_gop > votes_dem &
                        votes_gop > votes_other ~ 1,
                      votes_other > votes_dem &
                        votes_other > votes_gop ~ 2),
         )
```

Calculate additional columns

```
vot_info_fin %>%
  group_by(year, winning_party) %>%
  summarise(count= n()) %>%
  pivot_wider(id_cols = year,
              names_from = winning_party,
              values_from = count) %>%
  mutate(result = case_when(`Republican Party` > `Democratic Party` ~
                              "Republican Party",
                            `Democratic Party` > `Republican Party` ~
                              "Democratic Party",
                            `Democratic Party` == `Republican Party` ~
                              "Tie"
                            )
         ) %>%
  kableExtra::kable() %>%
  kableExtra::kable_minimal()
```

By State Result

```
## `summarise()` has grouped output by 'year'. You can override using the
## `.groups` argument.
```

year	Democratic Party	Republican Party	result
2008	29	21	Democratic Party
2012	27	23	Democratic Party
2016	21	29	Republican Party
2020	26	24	Democratic Party

```r
summary(vot_info_fin$voter_turnout)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.4220  0.5763  0.6215  0.6229  0.6675  0.7875
```

```r
vot_info_fin <- vot_info_fin %>%
  mutate(voter_turnout = if_else(voter_turnout>1 , 1, voter_turnout))

summary(vot_info_fin$voter_turnout)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.4220  0.5763  0.6215  0.6229  0.6675  0.7875
```

```r
dim(vot_info_fin)
```

```
## [1] 200  31
```

Transforming data for modeling Pivot the table so that each county has one record and so that data for each election is in separate columns.

```r
vot_info_fin_pivot <- vot_info_fin %>%
  pivot_wider(
    id_cols = c(state),
    names_from = year,
    values_from = c(totalvotes, cvap_est, voter_turnout, voter_turnout_dem, voter_turnout_gop, pctdiff_d
                    winning_party,winning_party_binary)
  )

dim(vot_info_fin_pivot)
```

```
## [1] 50 37
```

```r
colSums(is.na(vot_info_fin_pivot))
```

```
##                  state         totalvotes_2008         totalvotes_2012
##                      0                       0                       0
##        totalvotes_2016         totalvotes_2020           cvap_est_2008
##                      0                       0                       0
##          cvap_est_2012           cvap_est_2016           cvap_est_2020
##                      0                       0                       0
##     voter_turnout_2008      voter_turnout_2012      voter_turnout_2016
##                      0                       0                       0
##     voter_turnout_2020  voter_turnout_dem_2008  voter_turnout_dem_2012
##                      0                       0                       0
## voter_turnout_dem_2016  voter_turnout_dem_2020  voter_turnout_gop_2008
##                      0                       0                       0
## voter_turnout_gop_2012  voter_turnout_gop_2016  voter_turnout_gop_2020
##                      0                       0                       0
## pctdiff_dem_vs_gop_2008 pctdiff_dem_vs_gop_2012 pctdiff_dem_vs_gop_2016
##                      0                       0                       0
## pctdiff_dem_vs_gop_2020 rawdiff_dem_vs_gop_2008 rawdiff_dem_vs_gop_2012
##                      0                       0                       0
## rawdiff_dem_vs_gop_2016 rawdiff_dem_vs_gop_2020     winning_party_2008
##                      0                       0                       0
##     winning_party_2012      winning_party_2016      winning_party_2020
##                      0                       0                       0
```

```
## winning_party_binary_2008 winning_party_binary_2012 winning_party_binary_2016
##                          0                          0                          0
## winning_party_binary_2020
##                          0
```

```r
vot_info_fin_pivot_na <- vot_info_fin_pivot %>%
  filter(if_any(where(is.numeric), is.na))

vot_info_fin_pivot_na
```

```
## # A tibble: 0 x 37
## # i 37 variables: state <chr>, totalvotes_2008 <dbl>, totalvotes_2012 <dbl>,
## #   totalvotes_2016 <dbl>, totalvotes_2020 <dbl>, cvap_est_2008 <dbl>,
## #   cvap_est_2012 <dbl>, cvap_est_2016 <dbl>, cvap_est_2020 <dbl>,
## #   voter_turnout_2008 <dbl>, voter_turnout_2012 <dbl>,
## #   voter_turnout_2016 <dbl>, voter_turnout_2020 <dbl>,
## #   voter_turnout_dem_2008 <dbl>, voter_turnout_dem_2012 <dbl>,
## #   voter_turnout_dem_2016 <dbl>, voter_turnout_dem_2020 <dbl>, ...
```

Exploratory Data Analysis

```r
glimpse(vot_info_fin_pivot)
```

```
## Rows: 50
## Columns: 37
## $ state                   <chr> "ALABAMA", "ARIZONA", "ARKANSAS", "CALIFORNI~
## $ totalvotes_2008         <dbl> 2099819, 2293475, 1086617, 13561900, 2401361~
## $ totalvotes_2012         <dbl> 2070353, 2299254, 1069468, 13038547, 2569217~
## $ totalvotes_2016         <dbl> 2123367, 2604277, 1129896, 14181595, 2780220~
## $ totalvotes_2020         <dbl> 2323282, 3385294, 1219069, 17500881, 3256980~
## $ cvap_est_2008           <dbl> 3481380, 4110885, 2090155, 22329310, 3403825~
## $ cvap_est_2012           <dbl> 3600120, 4444230, 2152350, 23881285, 3679115~
## $ cvap_est_2016           <dbl> 3671115, 4812760, 2195865, 25232630, 3979310~
## $ cvap_est_2020           <dbl> 3782980, 5000090, 2211560, 25916215, 4194465~
## $ voter_turnout_2008      <dbl> 0.6031571, 0.5579030, 0.5198739, 0.6073587, ~
## $ voter_turnout_2012      <dbl> 0.5750789, 0.5173571, 0.4968839, 0.5459734, ~
## $ voter_turnout_2016      <dbl> 0.5783984, 0.5411192, 0.5145562, 0.5620340, ~
## $ voter_turnout_2020      <dbl> 0.6141407, 0.6770466, 0.5512258, 0.6752869, ~
## $ voter_turnout_dem_2008  <dbl> 0.2336657, 0.2516993, 0.2020472, 0.3705655, ~
## $ voter_turnout_dem_2012  <dbl> 0.2210193, 0.2306883, 0.1832458, 0.3288887, ~
## $ voter_turnout_dem_2016  <dbl> 0.1987263, 0.2412684, 0.1732775, 0.3469233, ~
## $ voter_turnout_dem_2020  <dbl> 0.2245912, 0.3344226, 0.1916891, 0.4286988, ~
## $ voter_turnout_gop_2008  <dbl> 0.36380573, 0.29923265, 0.30524865, 0.224448~
## $ voter_turnout_gop_2012  <dbl> 0.34885643, 0.27758554, 0.30094734, 0.202667~
## $ voter_turnout_gop_2016  <dbl> 0.35908709, 0.26022511, 0.31189167, 0.177698~
## $ voter_turnout_gop_2020  <dbl> 0.38096157, 0.33233122, 0.34394138, 0.231763~
## $ pctdiff_dem_vs_gop_2008 <dbl> -0.215764787, -0.085199969, -0.198512447, 0.~
## $ pctdiff_dem_vs_gop_2012 <dbl> -0.222294942, -0.090647662, -0.236879458, 0.~
## $ pctdiff_dem_vs_gop_2016 <dbl> -0.277249764, -0.035032372, -0.269385855, 0.~
## $ pctdiff_dem_vs_gop_2020 <dbl> -0.254616530, 0.003088949, -0.276206679, 0.2~
## $ rawdiff_dem_vs_gop_2008 <dbl> -453067, -195404, -215707, 3262692, 214987, ~
## $ rawdiff_dem_vs_gop_2012 <dbl> -460229, -208422, -253335, 3014327, 137948, ~
## $ rawdiff_dem_vs_gop_2016 <dbl> -588703, -91234, -304378, 4269978, 136386, 2~
## $ rawdiff_dem_vs_gop_2020 <dbl> -591546, 10457, -336715, 5103821, 439745, 36~
```

```
## $ winning_party_2008        <chr> "Republican Party", "Republican Party", "Rep~
## $ winning_party_2012        <chr> "Republican Party", "Republican Party", "Rep~
## $ winning_party_2016        <chr> "Republican Party", "Republican Party", "Rep~
## $ winning_party_2020        <chr> "Republican Party", "Democratic Party", "Rep~
## $ winning_party_binary_2008 <dbl> 1, 1, 1, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0,~
## $ winning_party_binary_2012 <dbl> 1, 1, 1, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 1, 0,~
## $ winning_party_binary_2016 <dbl> 1, 1, 1, 0, 0, 0, 0, 0, 1, 1, 0, 1, 0, 1, 1,~
## $ winning_party_binary_2020 <dbl> 1, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 1, 1,~
```

```
#identify empty and NA values
colSums(vot_info_fin_pivot == "" | is.na(vot_info_fin_pivot))
```

```
##                     state           totalvotes_2008           totalvotes_2012
##                         0                         0                         0
##           totalvotes_2016           totalvotes_2020             cvap_est_2008
##                         0                         0                         0
##             cvap_est_2012             cvap_est_2016             cvap_est_2020
##                         0                         0                         0
##         voter_turnout_2008        voter_turnout_2012        voter_turnout_2016
##                         0                         0                         0
##         voter_turnout_2020    voter_turnout_dem_2008    voter_turnout_dem_2012
##                         0                         0                         0
##     voter_turnout_dem_2016    voter_turnout_dem_2020    voter_turnout_gop_2008
##                         0                         0                         0
##     voter_turnout_gop_2012    voter_turnout_gop_2016    voter_turnout_gop_2020
##                         0                         0                         0
##    pctdiff_dem_vs_gop_2008   pctdiff_dem_vs_gop_2012   pctdiff_dem_vs_gop_2016
##                         0                         0                         0
##    pctdiff_dem_vs_gop_2020   rawdiff_dem_vs_gop_2008   rawdiff_dem_vs_gop_2012
##                         0                         0                         0
##    rawdiff_dem_vs_gop_2016   rawdiff_dem_vs_gop_2020        winning_party_2008
##                         0                         0                         0
##         winning_party_2012        winning_party_2016        winning_party_2020
##                         0                         0                         0
## winning_party_binary_2008 winning_party_binary_2012 winning_party_binary_2016
##                         0                         0                         0
## winning_party_binary_2020
##                         0
```

After cleaning, our dataset includes election data by county for 49 states and the District of Columbia for elections since 2008.

```
vot_info_fin_pivot %>%
  group_by(state) %>%
  summarise(count = n()) %>%
  arrange(desc(count))
```

```
## # A tibble: 50 x 2
##    state                count
##    <chr>                <int>
##  1 ALABAMA                  1
##  2 ARIZONA                  1
##  3 ARKANSAS                 1
##  4 CALIFORNIA               1
##  5 COLORADO                 1
##  6 CONNECTICUT              1
```

```
##   7 DELAWARE                   1
##   8 DISTRICT OF COLUMBIA       1
##   9 FLORIDA                    1
##  10 GEORGIA                    1
## # i 40 more rows
```

Summary Statistics

```r
vot_info_fin_pivot %>%
  # keep(is.numeric) %>%
  Hmisc::describe()
```

```
## .
##
##  37  Variables      50  Observations
## --------------------------------------------------------------------------------
## state
##        n  missing distinct
##       50        0       50
##
## lowest : ALABAMA       ARIZONA       ARKANSAS      CALIFORNIA    COLORADO
## highest: VIRGINIA      WASHINGTON    WEST VIRGINIA WISCONSIN     WYOMING
## --------------------------------------------------------------------------------
## totalvotes_2008
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##       50        0       50        1  2617223  2593224   320412   408942
##      .25      .50      .75      .90      .95
##   748693  1874417  3070222  5726041  7858842
##
## lowest :   256035   265853   316621   325046   377708
## highest:  5977981  7591233  8077795  8391639 13561900
## --------------------------------------------------------------------------------
## totalvotes_2012
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##       50        0       50        1  2574882  2536660   309929   408925
##      .25      .50      .75      .90      .95
##   729138  1880665  3157204  5596944  7574484
##
## lowest :   249061   293764   299290   322932   363815
## highest:  5742040  7061925  7993851  8474179 13038547
## --------------------------------------------------------------------------------
## totalvotes_2016
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##       50        0       50        1  2723449  2714469   328254   423053
##      .25      .50      .75      .90      .95
##   757802  2015184  3258220  5614377  8401388
##
## lowest :   255849   311268   315077   344360   370093
## highest:  6115402  7707363  8969226  9420039 14181595
## --------------------------------------------------------------------------------
## totalvotes_2020
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##       50        0       50        1  3162375  3195109   365872   495870
##      .25      .50      .75      .90      .95
```

```
##   881512  2235672  3980225  6121898  9984882
##
## lowest :   278503   344356   361819   370826   422609
## highest:  6915283  8661735 11067456 11315056 17500881
## ---------------------------------------------------------------------------
## cvap_est_2008
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##       50        0       50        1  4195044  4170941   491625   633410
##      .25      .50      .75      .90      .95
##  1309551  3215518  4637568  8793148 12918299
##
## lowest :   405095   435875   481700   503755   590660
## highest:  9475240 12812550 13004820 15277005 22329310
## ---------------------------------------------------------------------------
## cvap_est_2012
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##       50        0       50        1  4390725  4384967   511357   668502
##      .25      .50      .75      .90      .95
##  1355374  3333563  4850173  9013607 13561701
##
## lowest :   427305   475400   491550   535565   616000
## highest:  9676880 13425020 13673530 16529510 23881285
## ---------------------------------------------------------------------------
## cvap_est_2016
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##       50        0       50        1  4567752  4587293   534347   697236
##      .25      .50      .75      .90      .95
##  1379610  3395468  5121699  9124464 14257276
##
## lowest :   432285   494675   511190   562650   635415
## highest:  9748290 13686695 14724115 17859500 25232630
## ---------------------------------------------------------------------------
## cvap_est_2020
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##       50        0       50        1  4702691  4732570   538750   724965
##      .25      .50      .75      .90      .95
##  1399374  3417013  5344791  9209789 14848718
##
## lowest :   431010   512080   512335   571035   645585
## highest:  9893015 14182055 15394170 18729795 25916215
## ---------------------------------------------------------------------------
## voter_turnout_2008
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##       50        0       50        1   0.6266  0.06688   0.5239   0.5574
##      .25      .50      .75      .90      .95
##   0.5935   0.6297   0.6671   0.6928   0.7140
##
## lowest : 0.48086  0.49529  0.519874 0.528755 0.552552
## highest: 0.705489 0.710384 0.716994 0.719984 0.769177
## ---------------------------------------------------------------------------
## voter_turnout_2012
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##       50        0       50        1    0.594  0.07498   0.4845   0.5121
##      .25      .50      .75      .90      .95
```

```
##   0.5548   0.5920   0.6397   0.6816   0.7000
##
## lowest : 0.438971 0.460157 0.483611 0.485549 0.496884
## highest: 0.695838 0.698325 0.701361 0.719345 0.749026
## ---------------------------------------------------------------------------
## voter_turnout_2016
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##       50        0       50        1   0.6006  0.06956   0.5035   0.5153
##      .25      .50      .75      .90      .95
##   0.5645   0.6105   0.6389   0.6779   0.7006
##
## lowest : 0.421981 0.494479 0.50221  0.505152 0.514556
## highest: 0.68449  0.698669 0.702133 0.710067 0.729406
## ---------------------------------------------------------------------------
## voter_turnout_2020
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##       50        0       50        1   0.6704  0.06978   0.5546   0.5939
##      .25      .50      .75      .90      .95
##   0.6306   0.6707   0.7183   0.7456   0.7586
##
## lowest : 0.547172 0.54962  0.551226 0.558778 0.590313
## highest: 0.755092 0.757333 0.759601 0.776495 0.787542
## ---------------------------------------------------------------------------
## voter_turnout_dem_2008
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##       50        0       50        1   0.3251  0.09138   0.2032   0.2226
##      .25      .50      .75      .90      .95
##   0.2585   0.3378   0.3883   0.4100   0.4149
##
## lowest : 0.189829 0.193195 0.202047 0.204564 0.210943
## highest: 0.411041 0.413738 0.415819 0.455184 0.563923
## ---------------------------------------------------------------------------
## voter_turnout_dem_2012
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##       50        0       50        1   0.2941  0.09445   0.1628   0.1898
##      .25      .50      .75      .90      .95
##   0.2313   0.3093   0.3582   0.3835   0.4029
##
## lowest : 0.137509 0.161337 0.162146 0.163536 0.183246
## highest: 0.39438  0.40028  0.405035 0.405328 0.56178
## ---------------------------------------------------------------------------
## voter_turnout_dem_2016
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##       50        0       50        1   0.2715  0.09328   0.1525   0.1659
##      .25      .50      .75      .90      .95
##   0.2100   0.2727   0.3344   0.3473   0.3790
##
## lowest : 0.129482 0.130923 0.149112 0.156677 0.1591
## highest: 0.351163 0.360991 0.393659 0.401878 0.553278
## ---------------------------------------------------------------------------
## voter_turnout_dem_2020
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##       50        0       50        1   0.3292   0.1069   0.1834   0.2191
##      .25      .50      .75      .90      .95
```

```
##   0.2530   0.3347   0.3966   0.4309   0.4602
##
## lowest : 0.165938 0.170509 0.176661 0.191689 0.201217
## highest: 0.437729 0.452357 0.466635 0.474195 0.619366
## ---------------------------------------------------------------------------
## voter_turnout_gop_2008
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##       50        0       50        1   0.2916  0.06684   0.2079   0.2237
##      .25      .50      .75      .90      .95
##   0.2558   0.3061   0.3295   0.3527   0.3633
##
## lowest : 0.039844 0.127908 0.205468 0.210868 0.217141
## highest: 0.354197 0.362723 0.363806 0.381638 0.407208
## ---------------------------------------------------------------------------
## voter_turnout_gop_2012
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##       50        0       50        1   0.2879  0.07141   0.1760   0.2031
##      .25      .50      .75      .90      .95
##   0.2490   0.3032   0.3301   0.3491   0.3687
##
## lowest : 0.0449748 0.12226   0.165616  0.188583  0.202667
## highest: 0.351629  0.358678  0.376924  0.400094  0.404423
## ---------------------------------------------------------------------------
## voter_turnout_gop_2016
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##       50        0       50        1   0.2886   0.0729   0.1845   0.2142
##      .25      .50      .75      .90      .95
##   0.2595   0.3082   0.3350   0.3585   0.3629
##
## lowest : 0.024889 0.126757 0.177699 0.192791 0.205644
## highest: 0.359087 0.360367 0.364998 0.385309 0.403481
## ---------------------------------------------------------------------------
## voter_turnout_gop_2020
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##       50        0       50        1   0.3265  0.07788   0.2212   0.2288
##      .25      .50      .75      .90      .95
##   0.2933   0.3427   0.3676   0.4037   0.4120
##
## lowest : 0.036277 0.188344 0.220098 0.22251  0.228636
## highest: 0.404351 0.411243 0.412575 0.426769 0.449082
## ---------------------------------------------------------------------------
## pctdiff_dem_vs_gop_2008
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##       50        0       50        1  0.04804   0.2418 -0.26941 -0.20024
##      .25      .50      .75      .90      .95
## -0.12783  0.05421  0.17001  0.25898  0.32866
##
## lowest : -0.32062  -0.312902 -0.281781 -0.254296 -0.215765
## highest: 0.267072  0.278062  0.370065  0.452293  0.859246
## ---------------------------------------------------------------------------
## pctdiff_dem_vs_gop_2012
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##       50        0       50        1  0.00334   0.2623 -0.32808 -0.23995
##      .25      .50      .75      .90      .95
```

```
## -0.17819  0.03426  0.15104  0.26212  0.32966
##
## lowest : -0.480409 -0.408237 -0.335446 -0.319074 -0.267565
## highest: 0.274294  0.297487  0.355979  0.426808  0.836348
## --------------------------------------------------------------------------------
## pctdiff_dem_vs_gop_2016
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##       50        0       50        1 -0.03438   0.2638 -0.36093 -0.30030
##      .25      .50      .75      .90      .95
## -0.20227 -0.02351  0.11290  0.26408  0.28987
##
## lowest : -0.462953 -0.421536 -0.363912 -0.357289 -0.317612
## highest: 0.264164  0.276161  0.301093  0.321828  0.867763
## --------------------------------------------------------------------------------
## pctdiff_dem_vs_gop_2020
##         n   missing  distinct      Info      Mean       Gmd       .05       .10
##        50         0        50         1 -0.004123    0.2685 -0.332357 -0.279380
##       .25       .50       .75       .90       .95
## -0.180934  0.002812  0.160490  0.291935  0.332128
##
## lowest : -0.431119 -0.38935  -0.333573 -0.330871 -0.307943
## highest: 0.294664  0.332104  0.332148  0.350887  0.867524
## --------------------------------------------------------------------------------
## rawdiff_dem_vs_gop_2008
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##       50        0       50        1   191797   594627  -425470  -303473
##      .25      .50      .75      .90      .95
##  -169019   111687   288183   682166  1134253
##
## lowest : -950695 -457669 -453067 -391741 -366441
## highest:  795218  823940 1388146 2027402 3262692
## --------------------------------------------------------------------------------
## rawdiff_dem_vs_gop_2012
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##       50        0       50        1   102545   576196  -475936  -411816
##      .25      .50      .75      .90      .95
##  -208348    71058   214740   653377   816265
##
## lowest : -1261719  -501621  -488787  -460229  -447778
## highest:   705975   732976   884410  2100831  3014327
## --------------------------------------------------------------------------------
## rawdiff_dem_vs_gop_2016
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##       50        0       50        1    58184   618106  -582139  -524620
##      .25      .50      .75      .90      .95
##  -237832   -96383   123091   565186   926529
##
## lowest : -807179 -652230 -588703 -574117 -528761
## highest:  734759  904303  944714 1732973 4269978
## --------------------------------------------------------------------------------
## rawdiff_dem_vs_gop_2020
##        n  missing distinct     Info     Mean      Gmd      .05      .10
##       50        0       50        1   141613   727935  -574710  -490032
##      .25      .50      .75      .90      .95
```

```
##  -302033    11564   217077   807326  1129511
##
## lowest : -708764 -631221 -591546 -554133 -516390
## highest: 1008609 1025024 1215000 1986187 5103821
## --------------------------------------------------------------------------------
## winning_party_2008
##        n  missing distinct
##       50        0        2
##
## Value      Democratic Party Republican Party
## Frequency                29               21
## Proportion             0.58             0.42
## --------------------------------------------------------------------------------
## winning_party_2012
##        n  missing distinct
##       50        0        2
##
## Value      Democratic Party Republican Party
## Frequency                27               23
## Proportion             0.54             0.46
## --------------------------------------------------------------------------------
## winning_party_2016
##        n  missing distinct
##       50        0        2
##
## Value      Democratic Party Republican Party
## Frequency                21               29
## Proportion             0.42             0.58
## --------------------------------------------------------------------------------
## winning_party_2020
##        n  missing distinct
##       50        0        2
##
## Value      Democratic Party Republican Party
## Frequency                26               24
## Proportion             0.52             0.48
## --------------------------------------------------------------------------------
## winning_party_binary_2008
##        n  missing distinct     Info      Sum     Mean      Gmd
##       50        0        2    0.731       21     0.42   0.4971
##
## --------------------------------------------------------------------------------
## winning_party_binary_2012
##        n  missing distinct     Info      Sum     Mean      Gmd
##       50        0        2    0.745       23     0.46   0.5069
##
## --------------------------------------------------------------------------------
## winning_party_binary_2016
##        n  missing distinct     Info      Sum     Mean      Gmd
##       50        0        2    0.731       29     0.58   0.4971
##
## --------------------------------------------------------------------------------
## winning_party_binary_2020
##        n  missing distinct     Info      Sum     Mean      Gmd
```

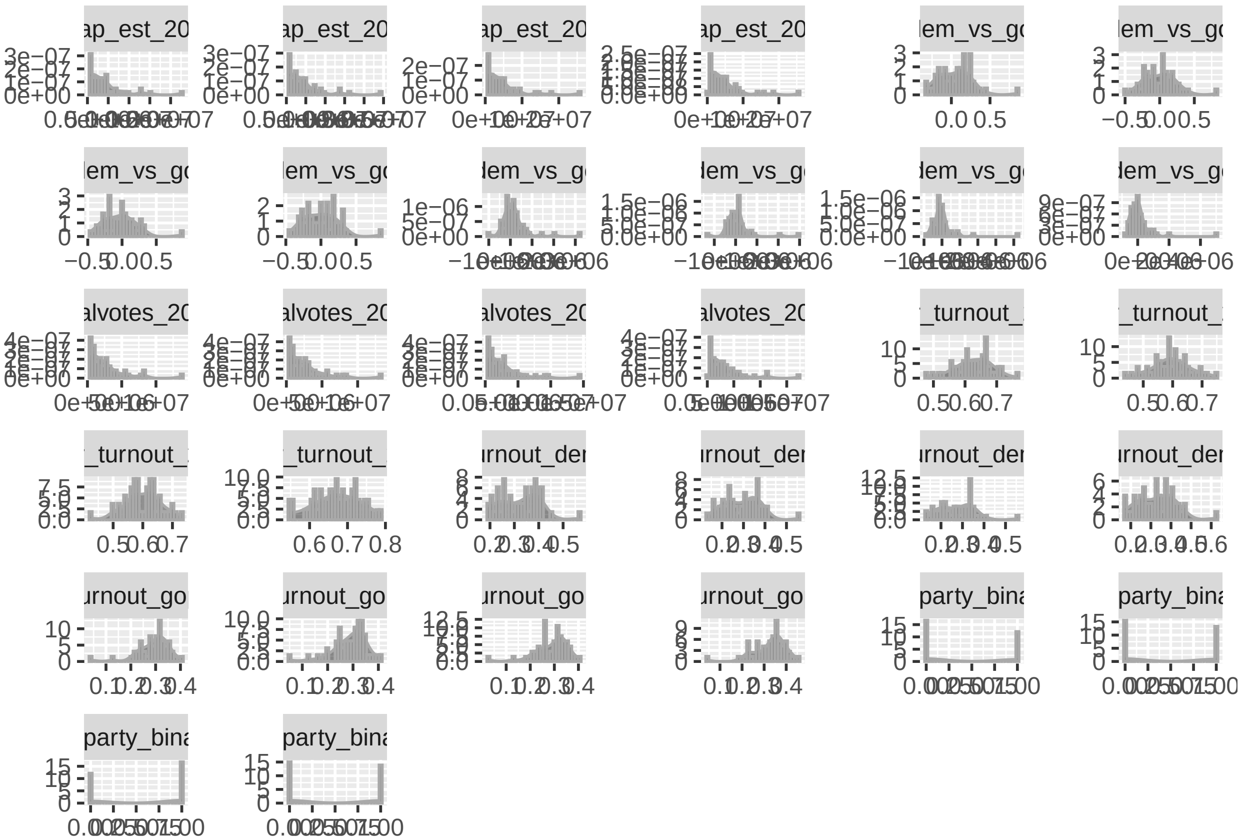
```
##        50         0         2     0.749        24      0.48    0.5094
##
## --------------------------------------------------------------------------------
```

Distribution of variables

```
# Histograms
vot_info_fin_pivot %>%
  keep(is.numeric) %>%
  gather() %>%
  ggplot(aes(value)) +
    facet_wrap(~ key, scales = "free") +
    geom_density(fill = "#222222", alpha = 0.5, color = "darkgray") +
    geom_histogram(aes(y=..density..), alpha=0.5, fill = "#222222", color="darkgray", position="identity
  theme(axis.title = element_blank())
```

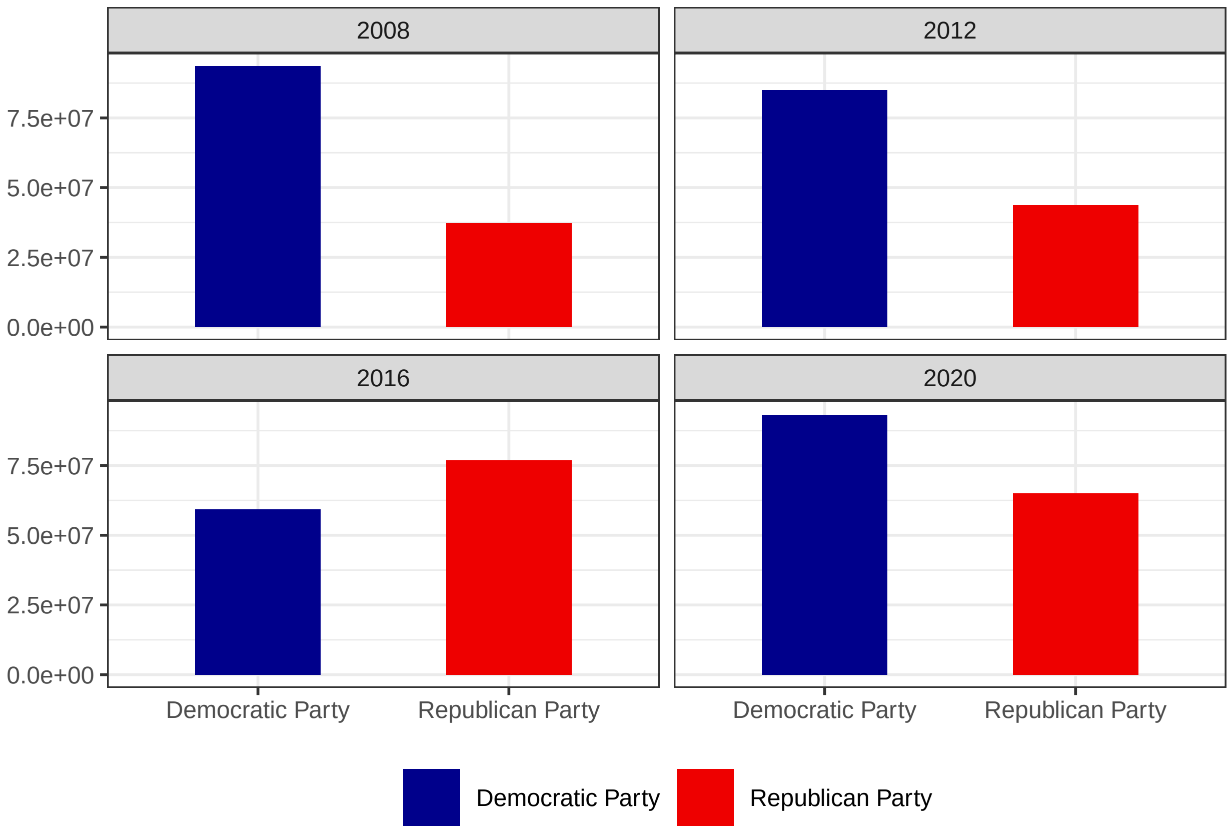
```
## Warning: The dot-dot notation (`..density..`) was deprecated in ggplot2 3.4.0.
## i Please use `after_stat(density)` instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



```
vot_info_fin %>%
  group_by(year, winning_party) %>%
  summarise(count = sum(totalvotes)) %>%
  ggplot(aes(x = winning_party, y = count, fill = winning_party)) +
```

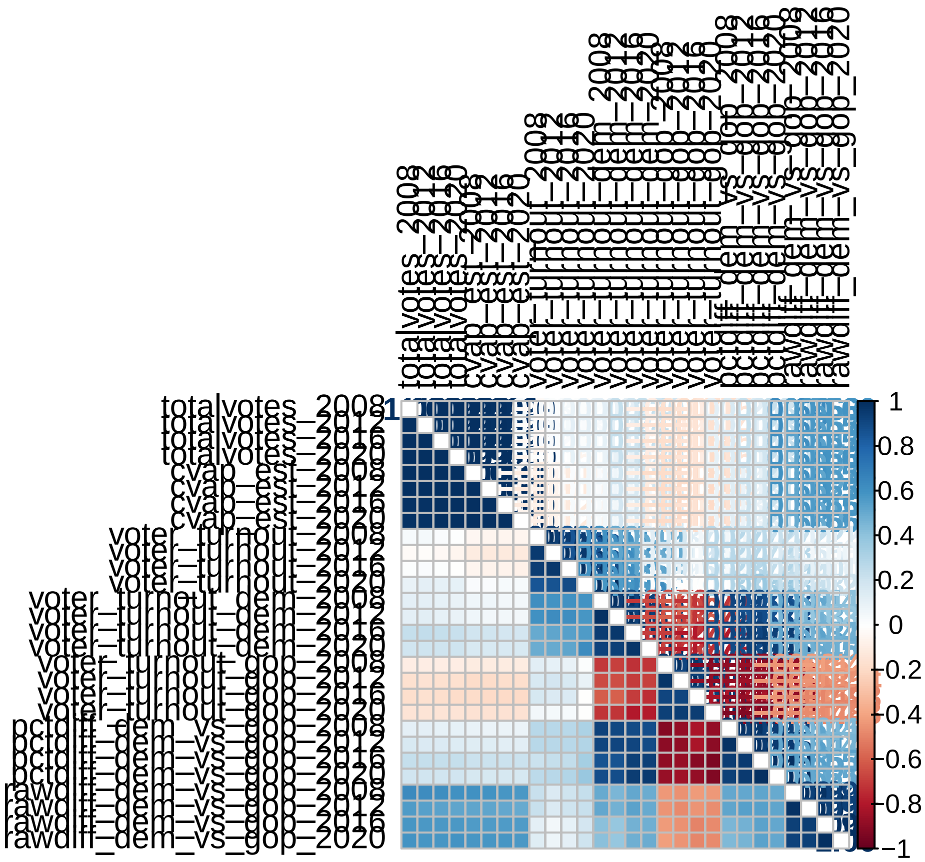
```r
  # Map fill to winning_party
  scale_fill_manual(values = c("darkblue","red2"))+
  geom_col(width = 0.5) +  #adjust the width as needed
  facet_wrap(~year) +
  theme_bw() + # Setting background as blank
  theme(legend.position = "bottom",
        #legend.position = c(0.11, 0.1), #puts legend inside the plot
        # legend.text = element_text(size = 6), #, family = "Arial"
        legend.key.size = unit(8, "mm"), #changes the size of the legend symbol
        legend.title = element_blank(), #removes legend title
        legend.spacing.x = unit(.25, 'cm'),
        axis.title = element_blank()
        )
```



Detect Multicollinearity Using Correlation Matrix

```r
cor_df <- vot_info_fin_pivot %>%
  select(-c(state, starts_with("winning"))) %>%
  keep(is.numeric)

cor_matrix <- cor(cor_df)

# Create a heatmap for the correlation matrix
# Visualize correlation between variables
corrplot.mixed(cor(cor_df %>% keep(is.numeric)),
               tl.col = 'black', tl.pos = 'lt',
               upper = "number", lower="shade",
```

```r
                shade.col=NA, tl.srt=90 )
```



Detect Multicollinearity Using VIF

The Variance Inflation Factor (VIF) helps quantify how much multicollinearity exists by showing how much the variance of a coefficient is inflated due to linear dependence with other predictors.

VIF Interpretation:
VIF = 1: No correlation between the predictor and other variables.
VIF between 1 and 5: Moderate correlation.
VIF > 5 (or sometimes > 10): High multicollinearity, and you may want to consider removing this variable.

```r
vif_data <- vif(lm(totalvotes_2020 ~ ., data=cor_df))
# Fit a linear model and calculate VIF
print(vif_data)
```

```
##         totalvotes_2008        totalvotes_2012        totalvotes_2016
##              12668.3908             12694.3444              7599.7554
##           cvap_est_2008          cvap_est_2012          cvap_est_2016
##             148251.5428            359757.1275            134479.5925
##           cvap_est_2020      voter_turnout_2008      voter_turnout_2012
##              29345.9999               731.9125               989.6403
##      voter_turnout_2016      voter_turnout_2020  voter_turnout_dem_2008
##                174.6884               823.5184              2021.3224
##  voter_turnout_dem_2012  voter_turnout_dem_2016  voter_turnout_dem_2020
##               2140.8185              1248.5868              4274.2918
##  voter_turnout_gop_2008  voter_turnout_gop_2012  voter_turnout_gop_2016
```

```
##               1046.6863              1622.7741              1075.2029
##   voter_turnout_gop_2020 pctdiff_dem_vs_gop_2008 pctdiff_dem_vs_gop_2012
##                926.9023              1768.3352              2541.5297
## pctdiff_dem_vs_gop_2016 pctdiff_dem_vs_gop_2020 rawdiff_dem_vs_gop_2008
##               3328.2442              2357.2987               379.9912
## rawdiff_dem_vs_gop_2012 rawdiff_dem_vs_gop_2016 rawdiff_dem_vs_gop_2020
##                427.1657               998.3352               655.8737
```

```
# Convert VIF values to a dataframe for visualization
vif_df <- as.data.frame(vif_data)
vif_df$variables <- rownames(vif_df)
```

Build Model

Based on the VIF values shown in our exploratory data analysis, it is evident there is high multicollinearity in our data. Multicollinearity, can cause problems in some models (like linear regression) but may not be as critical for tree-based methods like Random Forests. As such, we will build a Random Forest Model.

Before modelling, we will exclude non-predictive columns like 'FIPS', 'county', and 'state' from the model and subset the data to only include relevant columns. The columns "FIPS", "county", and "state" are identifiers or categorical labels, not numerical values that contribute directly to predicting totalvotes_2020. Including categorical variables like "county" or "state" without encoding them properly can lead to high dimensionality when creating dummy variables.

Base model

Train

```
#train
df_subset <- vot_info_fin_pivot %>%
  select(-c("winning_party_2008",
            "winning_party_2012",
            "winning_party_2020",
            "winning_party_2016")) %>%
  mutate(across(starts_with("winning"), as.factor),
         state = as.factor(state))

# Split the data into training and testing sets (70% train, 30% test)
set.seed(123)  # for reproducibility
train_indices <- sample(seq_len(nrow(df_subset)),
                        size = 0.7 * nrow(df_subset))
train_data <- df_subset[train_indices, ]
test_data <- df_subset[-train_indices, ]

rf_model <- randomForest(winning_party_binary_2020 ~ .,
                         data = train_data, ntree = 500,
                         mtry = 5, importance = TRUE)

# View the model summary
print(rf_model)
```

```
##
## Call:
##  randomForest(formula = winning_party_binary_2020 ~ ., data = train_data,      ntree = 500, mtry = 5
##               Type of random forest: classification
```

```
##                      Number of trees: 500
## No. of variables tried at each split: 5
##
##         OOB estimate of  error rate: 2.86%
## Confusion matrix:
##    0  1 class.error
## 0 16  1  0.05882353
## 1  0 18  0.00000000
```

This is the out-of-bag (OOB) error estimate, which is an internal error estimate in random forests. In this case, the OOB error rate is 2.86%, meaning that the model predicts strongly on the training data based on the OOB observations. Overall, the model proves to be highly accurate with almost perfect results and minimal overfitting.

Evaluate

```r
#evaluate
# Predictions on the test data
predictions <- predict(rf_model, test_data)

table(predictions)
```

```
## predictions
## 0 1
## 8 7
```

```r
# Confusion matrix to evaluate accuracy
conf_matrix <- confusionMatrix(predictions,
                               test_data$winning_party_binary_2020)
print(conf_matrix)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction 0 1
##          0 8 0
##          1 1 6
##
##                Accuracy : 0.9333
##                  95% CI : (0.6805, 0.9983)
##     No Information Rate : 0.6
##     P-Value [Acc > NIR] : 0.005172
##
##                   Kappa : 0.8649
##
##  Mcnemar's Test P-Value : 1.000000
##
##             Sensitivity : 0.8889
##             Specificity : 1.0000
##          Pos Pred Value : 1.0000
##          Neg Pred Value : 0.8571
##              Prevalence : 0.6000
##          Detection Rate : 0.5333
##    Detection Prevalence : 0.5333
##       Balanced Accuracy : 0.9444
```

```
##
##        'Positive' Class : 0
##
```

The test data correctly predicts Democrat Party for the 2020 election.

8 samples were correctly classified as 0 (True Negatives). 6 samples were correctly classified as 1 (True Positives). 1 sample was misclassified as 1 instead of 0 (False Positive). 0 samples were misclassified as 0 instead of 1 (False Negative).

Accuracy is the proportion of correct predictions over the total number of predictions: Accuracy $=8+6/(8+6+1+0) = 0.9333$ or 93.33% This indicates the model correctly classified 93.33% of the test data.

Checking for Overfitting

```r
rf_cv <- train(winning_party_binary_2020 ~ .,
               data = train_data, method = "rf",
               trControl = trainControl(method = "cv",
                                        number = 10))

print(rf_cv)
```

```
## Random Forest
##
## 35 samples
## 32 predictors
##  2 classes: '0', '1'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 32, 31, 31, 32, 32, 31, ...
## Resampling results across tuning parameters:
##
##   mtry  Accuracy   Kappa
##    2    0.9416667  0.89
##   41    0.9750000  0.95
##   80    0.9750000  0.95
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was mtry = 41.
```

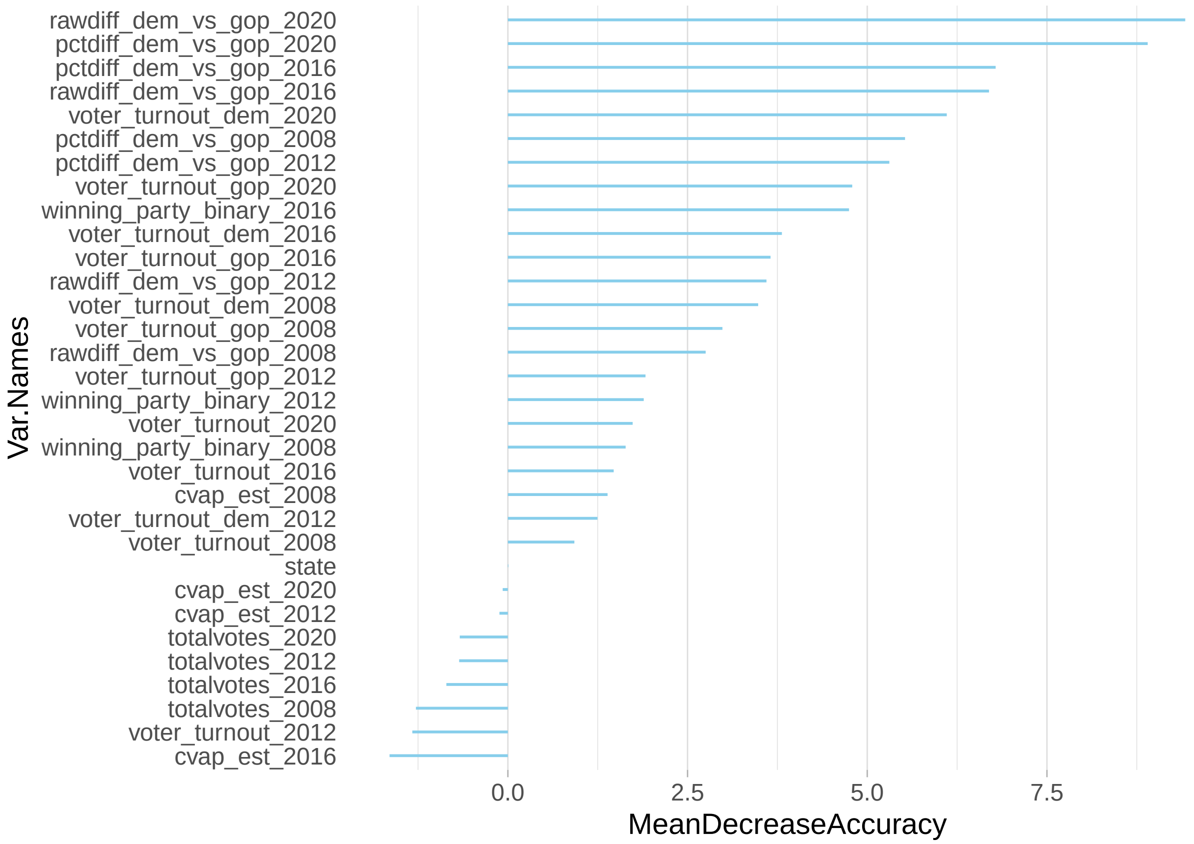
This Random Forest model shows good performance on the dataset (up to 93.3% accuracy). The tuning process optimized the mtry parameter to balance model complexity and predictive performance. With mtry = 41, the model uses a significant portion of the predictors for splitting, which is likely appropriate given the relatively small number of samples.

If deployed, the model should generalize well given the robustness of Random Forest and the cross-validation methodology used.

Feature Importance

```r
# Variable importance
#varImpPlot(rf_model)
ImpData <- as.data.frame(importance(rf_model))
ImpData$Var.Names <- row.names(ImpData)
```

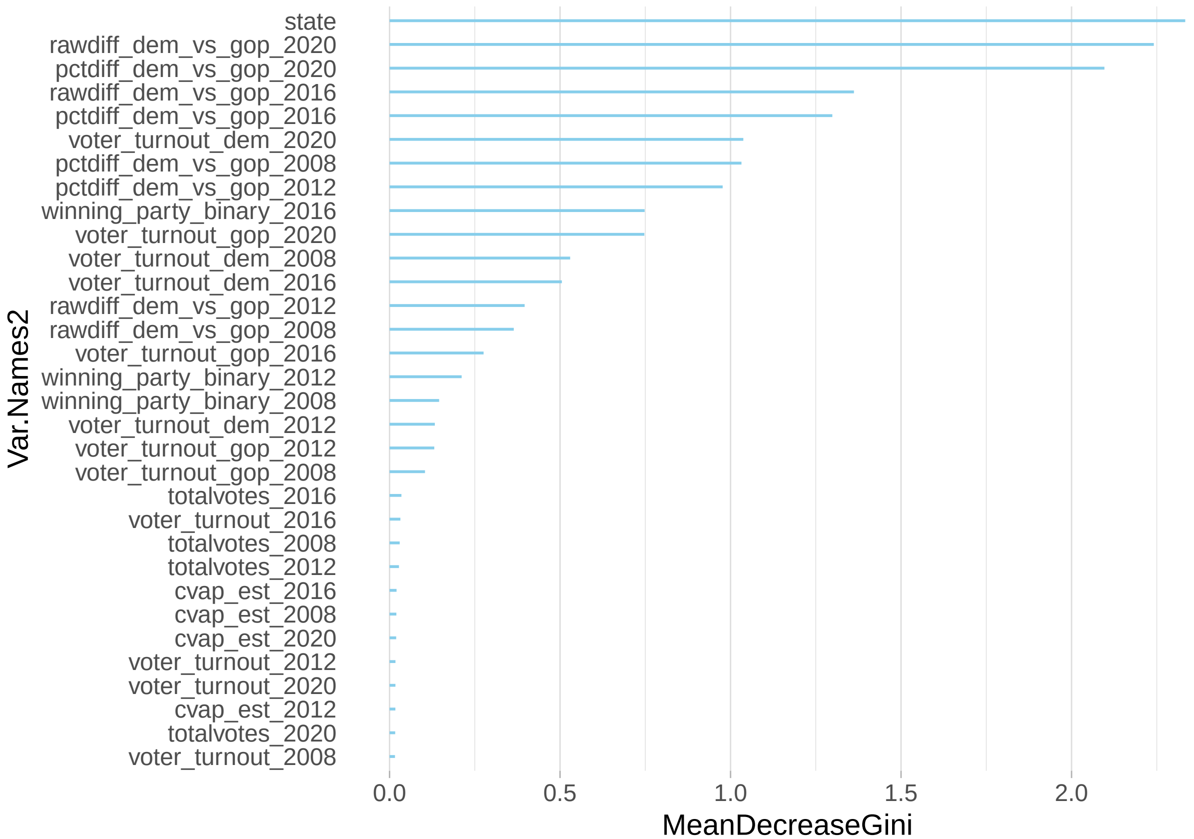
```r
#reorder variables based on MeanDecreaseAccuracy to display in descending order
ImpData$Var.Names <- factor(ImpData$Var.Names, levels = ImpData$Var.Names[order(ImpData$MeanDecreaseAccu

ggplot(ImpData, aes(x=Var.Names, y=MeanDecreaseAccuracy)) +
  geom_segment(aes(x=Var.Names, xend=Var.Names, y=0, yend=MeanDecreaseAccuracy), color="skyblue") +
  #geom_point(aes(size = IncNodePurity), color="steelblue", alpha=1) +
  theme_light() +
  coord_flip() +
  theme(
    legend.position = "bottom",
    panel.grid.major.y = element_blank(),
    panel.border = element_blank(),
    axis.ticks.y = element_blank()
  )
```



Mean Decrease Accuracy (MDA) is another metric used in Random Forest models to measure the importance of attributes. It quantifies how much the model's predictive accuracy decreases when a particular attribute's values are randomly permuted. The attributes with the lowest mean decrease accuracy are cvap_est_2016, voter_turnout_2012, totalvotes_2008, totalvotes_2016, and totalvotes_2012.

```r
#reorder variables based on MeanDecreaseGini to display in descending order
ImpData$Var.Names2 <- factor(ImpData$Var.Names, levels = ImpData$Var.Names[order(ImpData$MeanDecreaseGi

ggplot(ImpData, aes(x=Var.Names2, y=MeanDecreaseGini)) +
  geom_segment(aes(x=Var.Names2, xend=Var.Names2, y=0, yend=MeanDecreaseGini), color="skyblue") +
  #geom_point(aes(size = IncNodePurity), color="steelblue", alpha=1) +
  theme_light() +
  coord_flip() +
```

```r
  theme(
    legend.position = "bottom",
    panel.grid.major.y = element_blank(),
    panel.border = element_blank(),
    axis.ticks.y = element_blank()
  )
```



A high Mean Decrease Gini value for a variable indicates that it is an important attribute in the model. It allows for feature ranking and selection, helping to identify which variables most significantly impact the model's output. In our model, the top 5 attributes are state, rawdiff_dem_vs_gop_2020, pctdiff_dem_vs_gop_2020, rawdiff_dem_vs_gop_2016, pctdiff_dem_vs_gop_2016.

Demographic data

```r
# To obtain data for the 2008 population from the American Community
# Survey (ACS), you should use the 2006-2008 ACS 3-Year Estimates.
# This dataset aggregates data collected over those three years,
#  providing insights for the population during that period. 5
# year ACS data unavailable for 2008. 3 year ACS data was discontinued
# after 2009.

#load 2008 data using API
ed_attain2008 <- get_acs(
  geography = "county",
  variables = c(paste0("B15001_00",
                       seq(01,09),"E"),
                paste0("B15001_0",
```

```r
                        seq(10,83),"E")),
  year = 2008,
  survey = "acs3",
  cache_table = TRUE) %>%
  mutate(year=2008)

#2012 data and onward uses the 5 year ACS data
#load 2012 data using API
ed_attain2012 <- get_acs(
  geography = "county",
  variables = c(paste0("B15001_00",
                        seq(01,09),"E"),
                paste0("B15001_0",
                        seq(10,83),"E")),
  year = 2012,
  survey = "acs5",
  cache_table = TRUE) %>%
  mutate(year=2012)

#load 2016 data using API
ed_attain2016 <- get_acs(
  geography = "county",
  variables = c(paste0("B15001_00",
                        seq(01,09),"E"),
                paste0("B15001_0",
                        seq(10,83),"E")),
  year = 2016,
  survey = "acs5",
  cache_table = TRUE) %>%
  mutate(year=2016)

#load 2020 data using API
ed_attain2020 <- get_acs(
  geography = "county",
  variables = c(paste0("B15001_00",
                        seq(01,09),"E"),
                paste0("B15001_0",
                        seq(10,83),"E")),
  year = 2020,
  survey = "acs5",
  cache_table = TRUE) %>%
  mutate(year=2020)
```

```r
#check column names
#get column names 2008
url08 <- "https://api.census.gov/data/2008/acs/acs3/groups/B15001.html"

webpage08 <- read_html(url08)

table08 <- webpage08 %>%
  html_node("table") %>%  # Adjust the selector if necessary
  html_table() %>%
```

```r
  select(c("Name","Label","Concept","Required","Attributes",
           "Limit","Predicate Type","Group"))

filteredtable08 <- table08 %>%
  # filter(!is.na(Name) & Name != "") %>%
  # Remove rows with NA or empty names
  filter(Name %in% c(paste0("B15001_00", seq(01,09),"E"),
                     paste0("B15001_0", seq(10,83),"E")))
# %>%
#    mutate(Label = str_replace_all(Label,", GED, or alternative",
# ' (includes equivalency)'))

#get column names  2012
url12 <- "https://api.census.gov/data/2012/acs/acs5/groups/B15001.html"

webpage12 <- read_html(url12)

table12 <- webpage12 %>%
  html_node("table") %>%  # Adjust the selector if necessary
  html_table() %>%
  select(c("Name","Label","Concept","Required","Attributes",
           "Limit","Predicate Type","Group"))

filteredtable12 <- table12 %>%
  # filter(!is.na(Name) & Name != "") %>%
  # Remove rows with NA or empty names
  filter(Name %in% c(paste0("B15001_00", seq(01,09),"E"),
                     paste0("B15001_0", seq(10,83),"E")))
# %>%
#    mutate(Label = str_replace_all(Label,", GED, or alternative",
#' (includes equivalency)'))

#get column names 2016
url16 <- "https://api.census.gov/data/2016/acs/acs5/groups/B15001.html"

webpage16 <- read_html(url16)

table16 <- webpage16 %>%
  html_node("table") %>%  # Adjust the selector if necessary
  html_table() %>%
  select(c("Name","Label","Concept","Required","Attributes",
           "Limit","Predicate Type","Group"))

filteredtable16 <- table16 %>%
  # filter(!is.na(Name) & Name != "") %>%  # Remove rows with NA or empty names
  filter(Name %in% c(paste0("B15001_00", seq(01,09),"E"),
                     paste0("B15001_0", seq(10,83),"E")))

#get columnn names 2020
url20 <- "https://api.census.gov/data/2020/acs/acs5/groups/B15001.html"

webpage20 <- read_html(url20)
```

```r
table20 <- webpage20 %>%
  html_node("table") %>%  # Adjust the selector if necessary
  html_table() %>%
  select(c("Name","Label","Concept","Required","Attributes",
           "Limit","Predicate Type","Group"))

filteredtable20 <- table20 %>%
  # filter(!is.na(Name) & Name != "") %>%  # Remove rows with NA or empty names
  filter(Name %in% c(paste0("B15001_00", seq(01,09),"E"),
                     paste0("B15001_0", seq(10,83),"E"))) %>%
  mutate(Label = str_replace_all(Label,":",""))
```

```r
#update the mismatches
filteredtable08 <- filteredtable08 %>%
   mutate(Label = str_replace_all(Label,", GED, or alternative",
                                  ' (includes equivalency)'))

filteredtable12 <- filteredtable12 %>%
  mutate(Label = str_replace_all(Label,", GED, or alternative",
                                 ' (includes equivalency)'))
```

Get column names All column names are the same across all 4 election year Educational Attainment data.

```r
ed_attain <- rbind(ed_attain2008, ed_attain2012, ed_attain2016, ed_attain2020)
```

```r
ed_colnames <- filteredtable20 %>%
  mutate(Name = str_replace_all(Name,"E","")) %>%
  select(c(Name, Label))

table(sort(unique(ed_colnames$Name))==sort(unique(ed_attain$variable)))
```

Combine and merge education data

```
##
## TRUE
##   83
```

```r
ed_attain2a <- left_join(ed_attain, ed_colnames, by = c("variable"="Name"))

glimpse(ed_attain2a)
```

```
## Rows: 958,567
## Columns: 7
## $ GEOID    <chr> "01001", "01001", "01001", "01001", "01001", "01001", "01001"~
## $ NAME     <chr> "Autauga County, Alabama", "Autauga County, Alabama", "Autaug~
## $ variable <chr> "B15001_001", "B15001_002", "B15001_003", "B15001_004", "B150~
## $ estimate <dbl> 36493, 17387, 2160, 0, 543, 913, 567, 14, 123, 0, 3157, 64, 3~
## $ moe      <dbl> 132, 127, 182, 154, 260, 286, 177, 24, 89, 154, 244, 76, 222,~
## $ year     <dbl> 2008, 2008, 2008, 2008, 2008, 2008, 2008, 2008, 2008, 2008, 2~
## $ Label    <chr> "Estimate!!Total", "Estimate!!Total!!Male", "Estimate!!Total!~
```

```r
#identify empty and NA values
colSums(ed_attain2a == "" | is.na(ed_attain2a))
```

```
##    GEOID     NAME variable estimate      moe     year    Label
##        0        0        0        0     8584        0        0
```

```r
# voteFIPS <- unique(voting_info_final_pivot$FIPS)
demoFIPS <- unique(ed_attain2a$GEOID)

ed_attain2 <- ed_attain2a %>%
  filter(!GEOID %in% setdiff(demoFIPS, ls_FIPS)) %>%
  #keep only the fips we have in the voting dataset
  separate(col="NAME", into=c("county", "state"), sep=",") %>%
  mutate(county = str_remove(county, " County"),
         county = if_else(county == "Doña Ana", "Dona Ana", county)
         )

ed_attain3 <- ed_attain2 %>%
  group_by(state, year, variable, Label) %>%
  summarise(estimate = sum(estimate),
            moe = sum(moe)) %>%
  mutate(Label2 = Label) %>%
  separate(Label2, into = c("type","value","gender", "age_group",
                            "education"), sep = "!!")
```

Clean and reshape data

```
## `summarise()` has grouped output by 'state', 'year', 'variable'. You can
## override using the `.groups` argument.
```

```
## Warning: Expected 5 pieces. Missing pieces filled with `NA` in 2600 rows [1, 2, 3, 11,
## 19, 27, 35, 43, 44, 52, 60, 68, 76, 84, 85, 86, 94, 102, 110, 118, ...].
```

```r
length(unique(ed_attain3$GEOID))
```

```
## Warning: Unknown or uninitialised column: `GEOID`.
```

```
## [1] 0
```

```r
# edcountystate <- ed_attain3 %>%
#   select(GEOID,county, state) %>%
#   distinct(GEOID,county,state) %>%
#   group_by(GEOID) %>%
#   summarise(count=n())
```

```r
head(ed_attain3, 10)
```

```
## # A tibble: 10 x 11
## # Groups:   state, year, variable [10]
##    state       year variable   Label estimate   moe type  value gender age_group
##    <chr>      <dbl> <chr>      <chr>    <dbl> <dbl> <chr> <chr> <chr>  <chr>
##  1 " Alabama"  2008 B15001_001 Esti~  3312158  3241 Esti~ Total <NA>   <NA>
##  2 " Alabama"  2008 B15001_002 Esti~  1575413  4947 Esti~ Total Male   <NA>
##  3 " Alabama"  2008 B15001_003 Esti~   216719  7405 Esti~ Total Male   18 to 24~
##  4 " Alabama"  2008 B15001_004 Esti~     5635  5162 Esti~ Total Male   18 to 24~
##  5 " Alabama"  2008 B15001_005 Esti~    43862 12926 Esti~ Total Male   18 to 24~
```

```
##  6 " Alabama"  2008 B15001_006 Esti~    74290 15113 Esti~ Total Male   18 to 24~
##  7 " Alabama"  2008 B15001_007 Esti~    72890 15034 Esti~ Total Male   18 to 24~
##  8 " Alabama"  2008 B15001_008 Esti~     7478  5801 Esti~ Total Male   18 to 24~
##  9 " Alabama"  2008 B15001_009 Esti~    11740  6353 Esti~ Total Male   18 to 24~
## 10 " Alabama"  2008 B15001_010 Esti~      824  6330 Esti~ Total Male   18 to 24~
## # i 1 more variable: education <chr>
```

```r
#identify empty and NA values
colSums(ed_attain3 == "" | is.na(ed_attain3))
```

```
##     state      year  variable     Label  estimate       moe      type     value
##         0         0         0         0         0      1065         0         0
##    gender age_group education
##       200       600      2600
```

```r
ed_attain3_na <- ed_attain3 %>%
  filter(is.na(gender) | is.na(age_group) |
           is.na(education)) #is.na(gender) |

ed_attain3_na %>%
  count(variable, Label)
```

```
## # A tibble: 2,600 x 5
## # Groups:   state, year, variable [2,600]
##    state       year variable   Label                                    n
##    <chr>      <dbl> <chr>      <chr>                                <int>
##  1 " Alabama"  2008 B15001_001 Estimate!!Total                          1
##  2 " Alabama"  2008 B15001_002 Estimate!!Total!!Male                    1
##  3 " Alabama"  2008 B15001_003 Estimate!!Total!!Male!!18 to 24 years    1
##  4 " Alabama"  2008 B15001_011 Estimate!!Total!!Male!!25 to 34 years    1
##  5 " Alabama"  2008 B15001_019 Estimate!!Total!!Male!!35 to 44 years    1
##  6 " Alabama"  2008 B15001_027 Estimate!!Total!!Male!!45 to 64 years    1
##  7 " Alabama"  2008 B15001_035 Estimate!!Total!!Male!!65 years and over     1
##  8 " Alabama"  2008 B15001_043 Estimate!!Total!!Female                  1
##  9 " Alabama"  2008 B15001_044 Estimate!!Total!!Female!!18 to 24 years  1
## 10 " Alabama"  2008 B15001_052 Estimate!!Total!!Female!!25 to 34 years  1
## # i 2,590 more rows
```

```r
unique(ed_attain3_na$variable)
```

```
##  [1] "B15001_001" "B15001_002" "B15001_003" "B15001_011" "B15001_019"
##  [6] "B15001_027" "B15001_035" "B15001_043" "B15001_044" "B15001_052"
## [11] "B15001_060" "B15001_068" "B15001_076"
```

```r
#total county population
tot_pop <- ed_attain3 %>%
  filter(is.na(gender)) %>%
  select(state,  estimate, year, value)
```

```
## Adding missing grouping variables: `variable`
```

```r
#value is the column name that will be used to spread/pivot_wider

#male/female county population
gen <- ed_attain3 %>%
  filter(is.na(age_group), !is.na(gender)) %>%
  select(state,  estimate, year, gender)
```

```
## Adding missing grouping variables: `variable`
```

```r
#gender and age grp population
age_gen_pop <- ed_attain3_na %>%
  filter(!is.na(age_group)) %>%
  select(state,  estimate, year, gender, age_group)
```

```
## Adding missing grouping variables: `variable`
```

```r
#gender, age, education
ed_pop <- ed_attain3 %>%
  filter(!is.na(education)) %>%
  select(state, estimate, year, gender, age_group, education)
```

```
## Adding missing grouping variables: `variable`
```

```r
#age, education
age <- ed_pop %>%
  group_by(state,  year, age_group) %>%
  summarise(estimate = sum(estimate))
```

```
## `summarise()` has grouped output by 'state', 'year'. You can override using the
## `.groups` argument.
```

```r
#gender, education
ed_pop2 <- ed_pop %>%
  group_by(state,  year, gender, education) %>%
  summarise(estimate = sum(estimate))
```

```
## `summarise()` has grouped output by 'state', 'year', 'gender'. You can override
## using the `.groups` argument.
```

```r
#age, education
ed_pop3 <- ed_pop %>%
  group_by(state,  year, age_group,  education) %>%
  summarise(estimate = sum(estimate))
```

```
## `summarise()` has grouped output by 'state', 'year', 'age_group'. You can
## override using the `.groups` argument.
```

```r
#education
ed_pop4 <- ed_pop %>%
  group_by(state,  year, education) %>%
  summarise(estimate = sum(estimate))
```

```
## `summarise()` has grouped output by 'state', 'year'. You can override using the
## `.groups` argument.
```

Age, Gender, Education

```r
#need to spread/pivot_wider and then merge with main dataset for modelling
#age
age <- ed_pop %>%
  group_by(state,  year, age_group) %>%
  summarise(estimate = sum(estimate))
```

```
## `summarise()` has grouped output by 'state', 'year'. You can override using the
## `.groups` argument.
```

```r
#gender
gen <- ed_attain3 %>%
  filter(is.na(age_group), !is.na(gender)) %>%
  select(state,  estimate, year, gender)
```

```
## Adding missing grouping variables: `variable`
```

```r
#education level
edu <- ed_pop %>%
  group_by(state,  year, education) %>%
  summarise(estimate = sum(estimate))
```

```
## `summarise()` has grouped output by 'state', 'year'. You can override using the
## `.groups` argument.
```

```r
#age pivoted
age2 <- age %>%
  pivot_wider(id_cols = c(state),
              names_from = c(year,age_group),
              values_from = estimate)

colSums(age2 == "" | is.na(age2))
```

```
##                  state    2008_18 to 24 years    2008_25 to 34 years
##                      0                      0                      0
##    2008_35 to 44 years    2008_45 to 64 years 2008_65 years and over
##                      0                      0                      0
##    2012_18 to 24 years    2012_25 to 34 years    2012_35 to 44 years
##                      0                      0                      0
##    2012_45 to 64 years 2012_65 years and over    2016_18 to 24 years
##                      0                      0                      0
##    2016_25 to 34 years    2016_35 to 44 years    2016_45 to 64 years
##                      0                      0                      0
## 2016_65 years and over    2020_18 to 24 years    2020_25 to 34 years
##                      0                      0                      0
##    2020_35 to 44 years    2020_45 to 64 years 2020_65 years and over
##                      0                      0                      0
```

```r
#gender pivoted
gen2 <- gen %>%
  pivot_wider(id_cols = c(state),
              names_from = c(year, gender),
              values_from = estimate)

colSums(gen2 == "" | is.na(gen2))
```

```
##       state   2008_Male 2008_Female   2012_Male 2012_Female   2016_Male
##           0           0           0           0           0           0
## 2016_Female   2020_Male 2020_Female
##           0           0           0
```

```r
#edu pivoted
edu2 <- edu %>%
  pivot_wider(id_cols = c(state),
              names_from = c(year, education),
              values_from = estimate)
```

```r
colSums(edu2 == "" | is.na(edu2))
```

```
##                                            state
##                                                0
##              2008_9th to 12th grade, no diploma
##                                                0
##                         2008_Associate's degree
##                                                0
##                          2008_Bachelor's degree
##                                                0
##            2008_Graduate or professional degree
##                                                0
## 2008_High school graduate (includes equivalency)
##                                                0
##                         2008_Less than 9th grade
##                                                0
##                     2008_Some college, no degree
##                                                0
##              2012_9th to 12th grade, no diploma
##                                                0
##                         2012_Associate's degree
##                                                0
##                          2012_Bachelor's degree
##                                                0
##            2012_Graduate or professional degree
##                                                0
## 2012_High school graduate (includes equivalency)
##                                                0
##                         2012_Less than 9th grade
##                                                0
##                     2012_Some college, no degree
##                                                0
##              2016_9th to 12th grade, no diploma
##                                                0
##                         2016_Associate's degree
##                                                0
##                          2016_Bachelor's degree
##                                                0
##            2016_Graduate or professional degree
##                                                0
## 2016_High school graduate (includes equivalency)
##                                                0
##                         2016_Less than 9th grade
##                                                0
##                     2016_Some college, no degree
##                                                0
##              2020_9th to 12th grade, no diploma
##                                                0
##                         2020_Associate's degree
##                                                0
##                          2020_Bachelor's degree
##                                                0
##            2020_Graduate or professional degree
##                                                0
```

```
## 2020_High school graduate (includes equivalency)
##                                                 0
##                          2020_Less than 9th grade
##                                                 0
##                      2020_Some college, no degree
##                                                 0
```

```r
age2 <- age2 %>%
  select(-starts_with("2008"))

gen2 <- gen2 %>%
  select(-starts_with("2008"))

edu2 <- edu2 %>%
  select(-starts_with("2008"))
```

```r
dem0 <- left_join(age2, gen2, by = c("state"))

dem <- left_join(dem0, edu2, by = c("state")) %>%
  ungroup()

#check dimensions, there is an extra state now
dim(dem)
```

```
## [1] 50 43
```

```r
#na / empty cell check
colSums(dem == "" | is.na(dem))
```

```
##                                             state
##                                                 0
##                               2012_18 to 24 years
##                                                 0
##                               2012_25 to 34 years
##                                                 0
##                               2012_35 to 44 years
##                                                 0
##                               2012_45 to 64 years
##                                                 0
##                           2012_65 years and over
##                                                 0
##                               2016_18 to 24 years
##                                                 0
##                               2016_25 to 34 years
##                                                 0
##                               2016_35 to 44 years
##                                                 0
##                               2016_45 to 64 years
##                                                 0
##                           2016_65 years and over
##                                                 0
##                               2020_18 to 24 years
##                                                 0
##                               2020_25 to 34 years
##                                                 0
##                               2020_35 to 44 years
```

```
##                                                0
##                                2020_45 to 64 years
##                                                0
##                              2020_65 years and over
##                                                0
##                                         2012_Male
##                                                0
##                                       2012_Female
##                                                0
##                                         2016_Male
##                                                0
##                                       2016_Female
##                                                0
##                                         2020_Male
##                                                0
##                                       2020_Female
##                                                0
##                 2012_9th to 12th grade, no diploma
##                                                0
##                             2012_Associate's degree
##                                                0
##                              2012_Bachelor's degree
##                                                0
##               2012_Graduate or professional degree
##                                                0
## 2012_High school graduate (includes equivalency)
##                                                0
##                            2012_Less than 9th grade
##                                                0
##                        2012_Some college, no degree
##                                                0
##                 2016_9th to 12th grade, no diploma
##                                                0
##                             2016_Associate's degree
##                                                0
##                              2016_Bachelor's degree
##                                                0
##               2016_Graduate or professional degree
##                                                0
## 2016_High school graduate (includes equivalency)
##                                                0
##                            2016_Less than 9th grade
##                                                0
##                        2016_Some college, no degree
##                                                0
##                 2020_9th to 12th grade, no diploma
##                                                0
##                             2020_Associate's degree
##                                                0
##                              2020_Bachelor's degree
##                                                0
##               2020_Graduate or professional degree
##                                                0
## 2020_High school graduate (includes equivalency)
```

```
##                                                0
##                          2020_Less than 9th grade
##                                                0
##                      2020_Some college, no degree
##                                                0
```

```r
#check for dupe, no dupe, but Puerto Rico needs to be filtered out
unique(dem$state)
```

```
##  [1] " Alabama"               " Arizona"               " Arkansas"
##  [4] " California"            " Colorado"              " Connecticut"
##  [7] " Delaware"              " District of Columbia" " Florida"
## [10] " Georgia"               " Hawaii"                " Idaho"
## [13] " Illinois"              " Indiana"               " Iowa"
## [16] " Kansas"                " Kentucky"              " Louisiana"
## [19] " Maine"                 " Maryland"              " Massachusetts"
## [22] " Michigan"              " Minnesota"             " Mississippi"
## [25] " Missouri"              " Montana"               " Nebraska"
## [28] " Nevada"                " New Hampshire"         " New Jersey"
## [31] " New Mexico"            " New York"              " North Carolina"
## [34] " North Dakota"          " Ohio"                  " Oklahoma"
## [37] " Oregon"                " Pennsylvania"          " Rhode Island"
## [40] " South Carolina"        " South Dakota"          " Tennessee"
## [43] " Texas"                 " Utah"                  " Vermont"
## [46] " Virginia"              " Washington"            " West Virginia"
## [49] " Wisconsin"             " Wyoming"
```

```r
dem <- dem %>%
  filter(!str_detect(state, "Puerto Rico")) %>%
  mutate(state = trimws(state, which="both"))

vot_info_fin_pivot <- vot_info_fin_pivot %>%
  mutate(state = str_to_title(state))
```

Clean up

Merge with model data

```r
model_data <- left_join(vot_info_fin_pivot, dem, join_by(state == state))

dim(model_data)
```

```
## [1] 50 79
```

```r
colSums(model_data == "" | is.na(model_data))
```

```
##                                           state
##                                               0
##                                 totalvotes_2008
##                                               0
##                                 totalvotes_2012
##                                               0
##                                 totalvotes_2016
##                                               0
##                                 totalvotes_2020
```

```
##                                              0
##                                  cvap_est_2008
##                                              0
##                                  cvap_est_2012
##                                              0
##                                  cvap_est_2016
##                                              0
##                                  cvap_est_2020
##                                              0
##                             voter_turnout_2008
##                                              0
##                             voter_turnout_2012
##                                              0
##                             voter_turnout_2016
##                                              0
##                             voter_turnout_2020
##                                              0
##                         voter_turnout_dem_2008
##                                              0
##                         voter_turnout_dem_2012
##                                              0
##                         voter_turnout_dem_2016
##                                              0
##                         voter_turnout_dem_2020
##                                              0
##                         voter_turnout_gop_2008
##                                              0
##                         voter_turnout_gop_2012
##                                              0
##                         voter_turnout_gop_2016
##                                              0
##                         voter_turnout_gop_2020
##                                              0
##                        pctdiff_dem_vs_gop_2008
##                                              0
##                        pctdiff_dem_vs_gop_2012
##                                              0
##                        pctdiff_dem_vs_gop_2016
##                                              0
##                        pctdiff_dem_vs_gop_2020
##                                              0
##                        rawdiff_dem_vs_gop_2008
##                                              0
##                        rawdiff_dem_vs_gop_2012
##                                              0
##                        rawdiff_dem_vs_gop_2016
##                                              0
##                        rawdiff_dem_vs_gop_2020
##                                              0
##                             winning_party_2008
##                                              0
##                             winning_party_2012
##                                              0
##                             winning_party_2016
```

```
##                                                0
##                              winning_party_2020
##                                                0
##                       winning_party_binary_2008
##                                                0
##                       winning_party_binary_2012
##                                                0
##                       winning_party_binary_2016
##                                                0
##                       winning_party_binary_2020
##                                                0
##                             2012_18 to 24 years
##                                                1
##                             2012_25 to 34 years
##                                                1
##                             2012_35 to 44 years
##                                                1
##                             2012_45 to 64 years
##                                                1
##                          2012_65 years and over
##                                                1
##                             2016_18 to 24 years
##                                                1
##                             2016_25 to 34 years
##                                                1
##                             2016_35 to 44 years
##                                                1
##                             2016_45 to 64 years
##                                                1
##                          2016_65 years and over
##                                                1
##                             2020_18 to 24 years
##                                                1
##                             2020_25 to 34 years
##                                                1
##                             2020_35 to 44 years
##                                                1
##                             2020_45 to 64 years
##                                                1
##                          2020_65 years and over
##                                                1
##                                       2012_Male
##                                                1
##                                     2012_Female
##                                                1
##                                       2016_Male
##                                                1
##                                     2016_Female
##                                                1
##                                       2020_Male
##                                                1
##                                     2020_Female
##                                                1
##                2012_9th to 12th grade, no diploma
```

```
##                                                1
##                         2012_Associate's degree
##                                                1
##                          2012_Bachelor's degree
##                                                1
##             2012_Graduate or professional degree
##                                                1
## 2012_High school graduate (includes equivalency)
##                                                1
##                        2012_Less than 9th grade
##                                                1
##                    2012_Some college, no degree
##                                                1
##               2016_9th to 12th grade, no diploma
##                                                1
##                         2016_Associate's degree
##                                                1
##                          2016_Bachelor's degree
##                                                1
##             2016_Graduate or professional degree
##                                                1
## 2016_High school graduate (includes equivalency)
##                                                1
##                        2016_Less than 9th grade
##                                                1
##                    2016_Some college, no degree
##                                                1
##               2020_9th to 12th grade, no diploma
##                                                1
##                         2020_Associate's degree
##                                                1
##                          2020_Bachelor's degree
##                                                1
##             2020_Graduate or professional degree
##                                                1
## 2020_High school graduate (includes equivalency)
##                                                1
##                        2020_Less than 9th grade
##                                                1
##                    2020_Some college, no degree
##                                                1
```

```
model_data2 <- model_data %>%
  drop_na() %>%
  janitor::clean_names()

dim(model_data2)
```

```
## [1] 49 79
```

#Build Second Model ### Train

```
#train
df_subset2 <- model_data2 %>%
  select(-c("winning_party_2008", "winning_party_2012", "winning_party_2020", "winning_party_2016")) %>%
  mutate(across(starts_with("winning"), as.factor),
```

```r
        state = as.factor(state))

# Split the data into training and testing sets (70% train, 30% test)
set.seed(123)  # for reproducibility
train_indices2 <- sample(seq_len(nrow(df_subset2)),
                        size = 0.7 * nrow(df_subset2))
train_data2 <- df_subset2[train_indices2, ]
test_data2 <- df_subset2[-train_indices2, ]

rf_model2 <- randomForest(winning_party_binary_2020 ~ .,
                          data = train_data2,
                          ntree = 500,
                          mtry = 5,
                          importance = TRUE)

# View the model summary
print(rf_model2)
```

```
##
## Call:
##  randomForest(formula = winning_party_binary_2020 ~ ., data = train_data2,      ntree = 500, mtry = 5
##               Type of random forest: classification
##                     Number of trees: 500
## No. of variables tried at each split: 5
##
##         OOB estimate of  error rate: 5.88%
## Confusion matrix:
##    0  1 class.error
## 0 15  1  0.06250000
## 1  1 17  0.05555556
```

True 0 (15): 15 instances of class 0 were correctly classified.

False 0 (1): 1 instance was incorrectly classified as 0.

True 1 (17): 17 instances of class 1 were correctly classified.

False 1 (1): Only 1 instance was incorrectly classified as 1.

Class error:
For class 0: 0.0625% error.
For class 1: 0.0556% error.

Evaluate

```r
#evaluate
# Predictions on the test data
predictions2 <- predict(rf_model2, test_data2)

#0= dem, 1=rep
table(predictions2)
```

```
## predictions2
## 0 1
## 8 7
```

```r
# Confusion matrix to evaluate accuracy
conf_matrix2 <- confusionMatrix(predictions2, test_data2$winning_party_binary_2020)
print(conf_matrix2)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction 0 1
##          0 8 0
##          1 1 6
##
##                Accuracy : 0.9333
##                  95% CI : (0.6805, 0.9983)
##     No Information Rate : 0.6
##     P-Value [Acc > NIR] : 0.005172
##
##                   Kappa : 0.8649
##
##  Mcnemar's Test P-Value : 1.000000
##
##             Sensitivity : 0.8889
##             Specificity : 1.0000
##          Pos Pred Value : 1.0000
##          Neg Pred Value : 0.8571
##              Prevalence : 0.6000
##          Detection Rate : 0.5333
##    Detection Prevalence : 0.5333
##       Balanced Accuracy : 0.9444
##
##        'Positive' Class : 0
##
```

The model performs well overall, with high accuracy (93.33%), excellent sensitivity (88.89%), and perfect specificity (100%). It is also statistically significantly better than random predictions ($p = 0.005172$). It missed only one instance where the true class was 1 but predicted as 0.

Checking for Overfitting

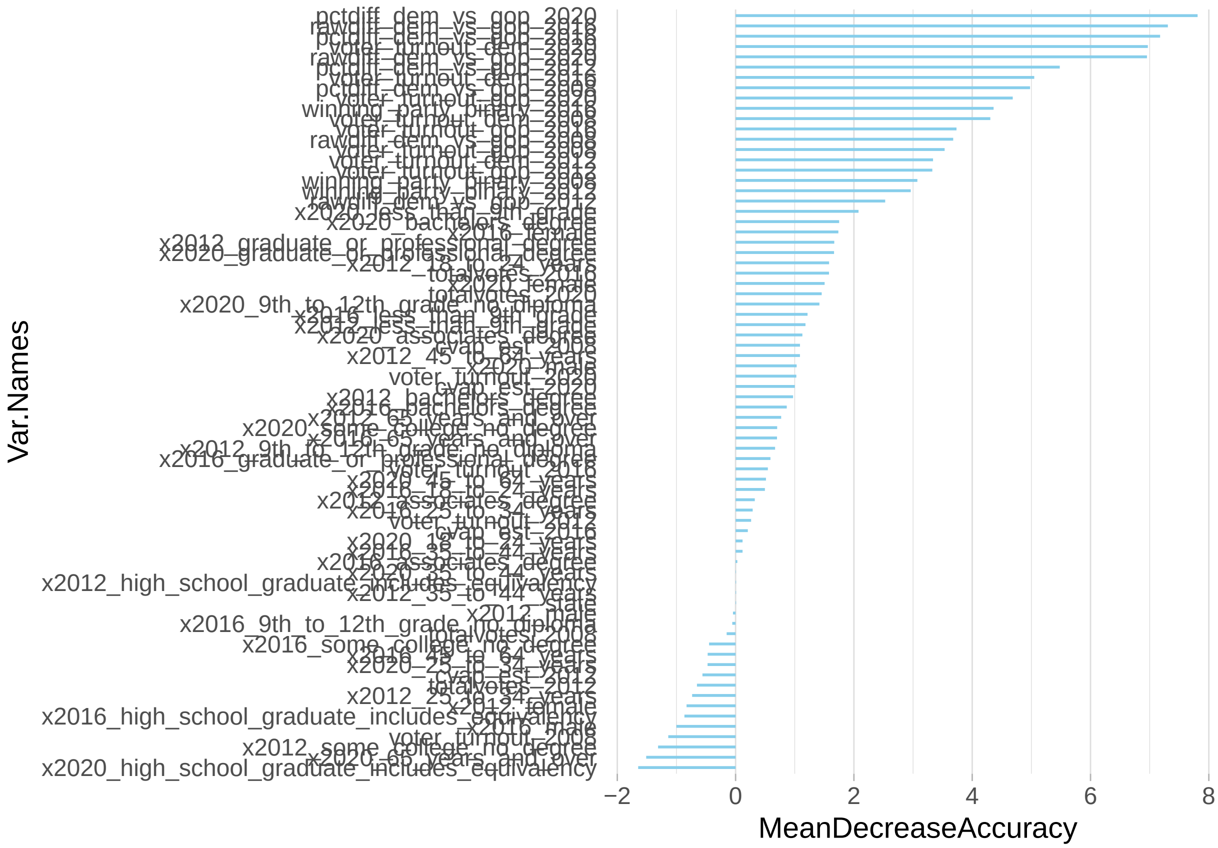
```r
rf_cv2 <- train(winning_party_binary_2020 ~ .,
                data = train_data2,
                method = "rf",
                trControl = trainControl(method = "cv", number = 10))

print(rf_cv2)
```

```
## Random Forest
##
## 34 samples
## 74 predictors
##  2 classes: '0', '1'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 30, 30, 30, 31, 31, 31, ...
```

```
## Resampling results across tuning parameters:
##
##   mtry  Accuracy   Kappa
##     2   0.8500000  0.68
##    61   0.9333333  0.88
##   121   0.9333333  0.88
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was mtry = 61.
```

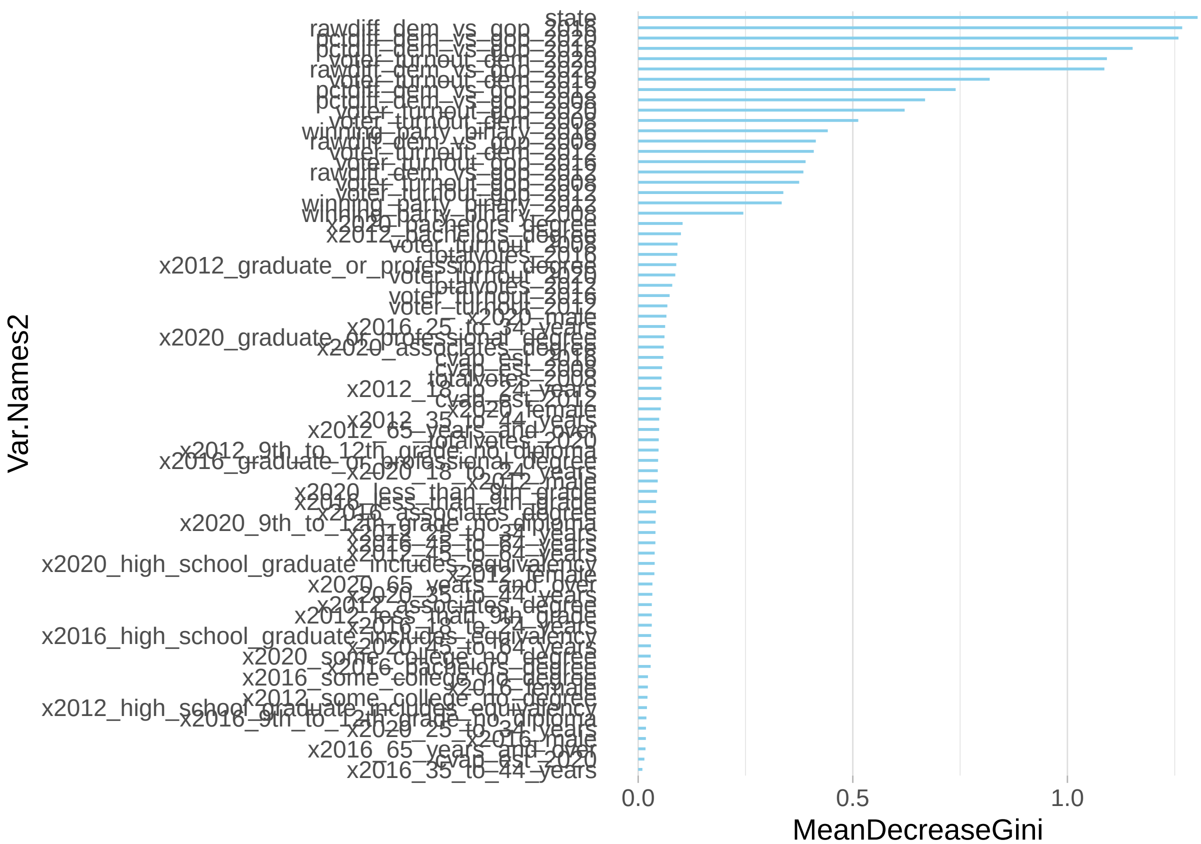
Feature Importance

```r
# Variable importance

ImpData2 <- as.data.frame(importance(rf_model2))
ImpData2$Var.Names <- row.names(ImpData2)

#reorder variables based on MeanDecreaseAccuracy to display in descending order
ImpData2$Var.Names <- factor(ImpData2$Var.Names, levels = ImpData2$Var.Names[order(ImpData2$MeanDecreas

ggplot(ImpData2, aes(x=Var.Names, y=MeanDecreaseAccuracy)) +
  geom_segment(aes(x=Var.Names, xend=Var.Names, y=0, yend=MeanDecreaseAccuracy), color="skyblue") +
  #geom_point(aes(size = IncNodePurity), color="steelblue", alpha=1) +
  theme_light() +
  coord_flip() +
  theme(
    legend.position = "bottom",
    panel.grid.major.y = element_blank(),
    panel.border = element_blank(),
    axis.ticks.y = element_blank()
  )
```



The attributes with the lowest mean decrease accuracy in our second model are x2020_high_school_graduate_includes_equiv
x2020_65_years_and_over, x2012_some_college_no_degree, voter_turnout_2008, and x2016_male.

```
#reorder variables based on MeanDecreaseGini to display in descending order
ImpData2$Var.Names2 <- factor(ImpData2$Var.Names, levels = ImpData2$Var.Names[order(ImpData2$MeanDecrea

ggplot(ImpData2, aes(x=Var.Names2, y=MeanDecreaseGini)) +
  geom_segment(aes(x=Var.Names2, xend=Var.Names2, y=0, yend=MeanDecreaseGini), color="skyblue") +
  #geom_point(aes(size = IncNodePurity), color="steelblue", alpha=1) +
  theme_light() +
  coord_flip() +
  theme(
    legend.position = "bottom",
    panel.grid.major.y = element_blank(),
    panel.border = element_blank(),
    axis.ticks.y = element_blank()
  )
```



In our second model, the top 5 attributes are state, rawdiff_dem_vs_gop_2016, pctdiff_dem_vs_gop_2020, pctdiff_dem_vs_gop_2016, voter_turnout_dem_2020.

Prediction

```r
predictions_2024 <- predict(rf_model2, df_subset2)

#demo = 0, rep = 1
table(predictions_2024) # Republican Party
```

```
## predictions_2024
##  0  1
## 24 25
```

```r
# table(df_subset2$winning_party_binary_2020) #Democratic Party
#
# table(df_subset2$winning_party_binary_2016) #Republican Party
```

The prediction results of the model show that the Republican Party would win the 2024 elections which is true to the outcome of our elections this year.

Model predictions by state

```r
#merge predictions back with original data
model_data3 <- model_data2

model_data3$predicted_values2024 <- predictions_2024
```

```r
model_data3 <- model_data3 %>%
  mutate(prediction_2024 = if_else(predictions_2024 == 0, "Democratic Party", "Republican Party"))

state_predictions <- model_data3 %>%
  select(c(state, prediction_2024))

state_predictions%>%
  kableExtra::kable() %>%
  kableExtra::kable_minimal()
```

state	prediction_2024
Alabama	Republican Party
Arizona	Republican Party
Arkansas	Republican Party
California	Democratic Party
Colorado	Democratic Party
Connecticut	Democratic Party
Delaware	Democratic Party
Florida	Republican Party
Georgia	Democratic Party
Hawaii	Democratic Party
Idaho	Republican Party
Illinois	Democratic Party
Indiana	Republican Party
Iowa	Republican Party
Kansas	Republican Party
Kentucky	Republican Party
Louisiana	Republican Party
Maine	Democratic Party
Maryland	Democratic Party
Massachusetts	Democratic Party
Michigan	Democratic Party
Minnesota	Democratic Party
Mississippi	Republican Party
Missouri	Republican Party
Montana	Republican Party
Nebraska	Republican Party
Nevada	Democratic Party
New Hampshire	Democratic Party
New Jersey	Democratic Party
New Mexico	Democratic Party
New York	Democratic Party
North Carolina	Republican Party
North Dakota	Republican Party
Ohio	Republican Party
Oklahoma	Republican Party
Oregon	Democratic Party
Pennsylvania	Democratic Party
Rhode Island	Democratic Party

South Carolina	Republican Party
South Dakota	Republican Party
Tennessee	Republican Party
Texas	Republican Party
Utah	Republican Party
Vermont	Democratic Party
Virginia	Democratic Party
Washington	Democratic Party
West Virginia	Republican Party
Wisconsin	Democratic Party
Wyoming	Republican Party

Actual election results by state

```
# Specify the URL
url <- "https://www.reuters.com/graphics/USA-ELECTION/RESULTS/zjpqnemxwvx/"

response <- GET(url)

# Parse the webpage content
webpage <- read_html(content(response, as = "text"))
```

```
## No encoding supplied: defaulting to UTF-8.
```

```
# Extract the table(s)
tables <- html_table(webpage, fill = TRUE)

tbl1 <- tables[[1]]
colnames(tbl1)[colnames(tbl1) == ""] <- "st_abbrv"
tbl1 <- tbl1 %>%
  mutate(type="Solid Democrat")

tbl2 <- tables[[2]]
colnames(tbl2)[colnames(tbl2) == ""] <- "st_abbrv"
tbl2 <- tbl2 %>%
  mutate(type="Lean Democrat")

tbl3 <- tables[[3]]
colnames(tbl3)[colnames(tbl3) == ""] <- "st_abbrv"
tbl3 <- tbl3 %>%
  mutate(type="Competitive")

tbl4 <- tables[[4]]
colnames(tbl4)[colnames(tbl4) == ""] <- "st_abbrv"
tbl4 <- tbl4 %>%
  mutate(type="Lean Republican")

tbl5 <- tables[[5]]
colnames(tbl5)[colnames(tbl5) == ""] <- "st_abbrv"
tbl5 <- tbl5 %>%
  mutate(type="Republican")

actual_results2024 <- rbind(tbl1, tbl2, tbl3, tbl4, tbl5)
```

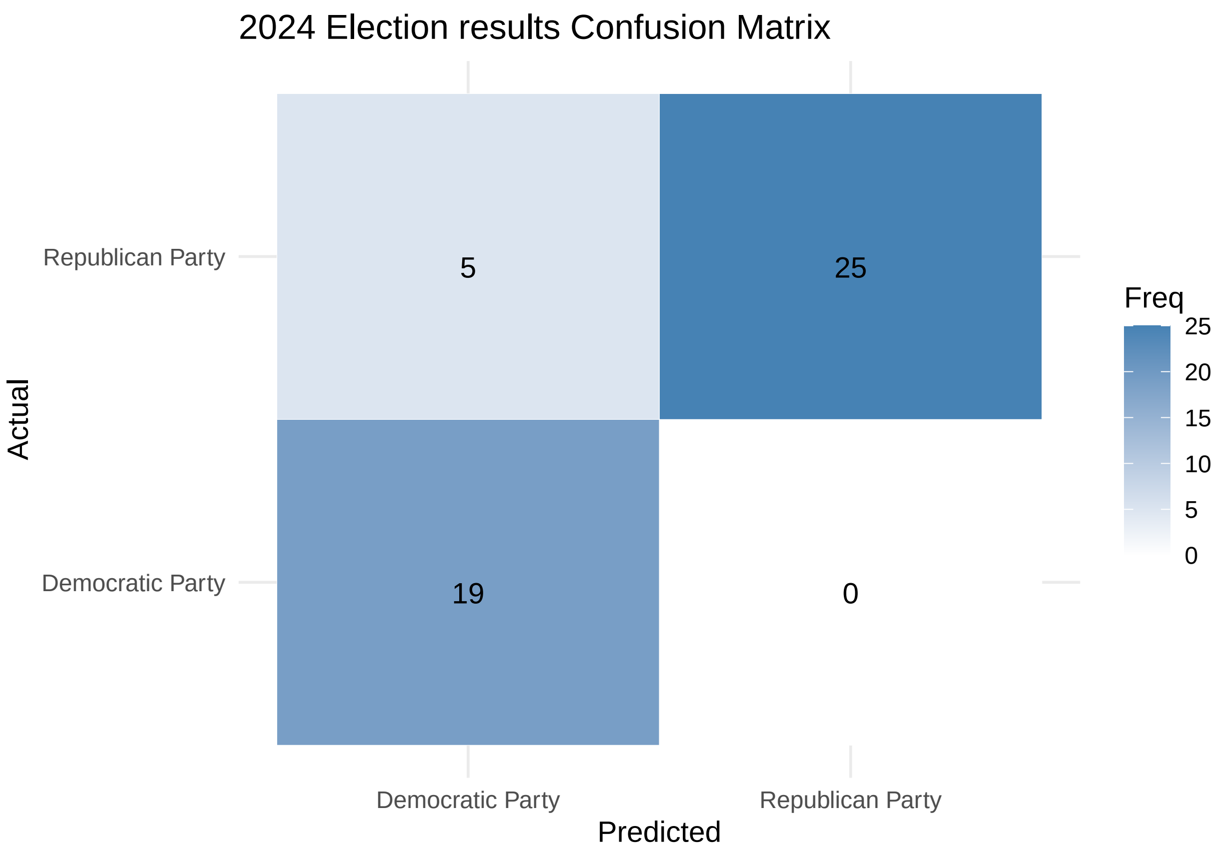
```r
# colnames(actual_results2024)[colnames(actual_results2024) == ""] <- "st_abbrv"

actual_results2024_ <- actual_results2024 %>%
  filter(!st_abbrv == "") %>%
  mutate(st_abbrv2 = case_when(st_abbrv=="D.C." ~ "District Of Columbia",
                               st_abbrv == "Md." ~ "Maryland",
                               st_abbrv == "Neb." ~ "Nebraska",
                               st_abbrv == "N.C." ~ "North Carolina",
                               st_abbrv == "N.D." ~ "North Dakota",
                               st_abbrv == "N.H." ~ "New Hampshire",
                               st_abbrv == "N.J." ~ "New Jersey",
                               st_abbrv == "N.M." ~ "New Mexico",
                               st_abbrv == "N.Y." ~ "New York",
                               st_abbrv == "Nev." ~ "Nevada",
                               st_abbrv == "Va." ~ "Virginia",
                               st_abbrv == "Vt." ~ "Vermont",
                               st_abbrv == "W.Va." ~ "West Virginia",
                               st_abbrv == "Wash." ~ "Washington",
                               TRUE ~ st_abbrv)) %>%
  arrange(st_abbrv2) %>%
  mutate(State = ls_states,
         Democrat = as.numeric(str_remove(Dem., "%"))/100,
         Republican = as.numeric(str_remove(Rep., "%"))/100,
         actual_2024 = if_else(Democrat>Republican, "Democratic Party","Republican Party")
         )

act_res24_tbl <- actual_results2024_ %>%
  select(c(State, Democrat, Republican, type, actual_2024))

act_vs_res <- left_join(act_res24_tbl, state_predictions, join_by(State==state)) %>%
  mutate(correctly_predicted = actual_2024==prediction_2024)

act_vs_res %>%
  kableExtra::kable() %>%
  kableExtra::kable_minimal()
```

State	Democrat	Republican	type	actual_2024	prediction_2024	correctly_p
Alabama	0.34	0.65	Republican	Republican Party	Republican Party	TRUE
Alaska	0.41	0.55	Republican	Republican Party	NA	NA
Arizona	0.47	0.52	Competitive	Republican Party	Republican Party	TRUE
Arkansas	0.34	0.64	Republican	Republican Party	Republican Party	TRUE
California	0.58	0.38	Solid Democrat	Democratic Party	Democratic Party	TRUE
Colorado	0.54	0.43	Solid Democrat	Democratic Party	Democratic Party	TRUE
Connecticut	0.56	0.42	Solid Democrat	Democratic Party	Democratic Party	TRUE
Delaware	0.57	0.42	Solid Democrat	Democratic Party	Democratic Party	TRUE
District Of Columbia	0.90	0.06	Solid Democrat	Democratic Party	NA	NA
Florida	0.43	0.56	Lean Republican	Republican Party	Republican Party	TRUE
Georgia	0.49	0.51	Competitive	Republican Party	Democratic Party	FALSE
Hawaii	0.61	0.37	Solid Democrat	Democratic Party	Democratic Party	TRUE
Idaho	0.30	0.67	Republican	Republican Party	Republican Party	TRUE
Illinois	0.55	0.44	Solid Democrat	Democratic Party	Democratic Party	TRUE

Indiana	0.40	0.59	Republican	Republican Party	Republican Party	TRUE
Iowa	0.43	0.56	Republican	Republican Party	Republican Party	TRUE
Kansas	0.41	0.57	Republican	Republican Party	Republican Party	TRUE
Kentucky	0.34	0.65	Republican	Republican Party	Republican Party	TRUE
Louisiana	0.38	0.60	Republican	Republican Party	Republican Party	TRUE
Maine	0.52	0.45	Lean Democrat	Democratic Party	Democratic Party	TRUE
Maryland	0.63	0.34	Solid Democrat	Democratic Party	Democratic Party	TRUE
Massachusetts	0.61	0.36	Solid Democrat	Democratic Party	Democratic Party	TRUE
Michigan	0.48	0.50	Competitive	Republican Party	Democratic Party	FALSE
Minnesota	0.51	0.47	Competitive	Democratic Party	Democratic Party	TRUE
Mississippi	0.38	0.61	Republican	Republican Party	Republican Party	TRUE
Missouri	0.40	0.58	Republican	Republican Party	Republican Party	TRUE
Montana	0.38	0.58	Republican	Republican Party	Republican Party	TRUE
Nebraska	0.39	0.59	Republican	Republican Party	Republican Party	TRUE
Nevada	0.47	0.51	Competitive	Republican Party	Democratic Party	FALSE
New Hampshire	0.51	0.48	Lean Democrat	Democratic Party	Democratic Party	TRUE
New Jersey	0.52	0.46	Solid Democrat	Democratic Party	Democratic Party	TRUE
New Mexico	0.52	0.46	Lean Democrat	Democratic Party	Democratic Party	TRUE
New York	0.56	0.44	Solid Democrat	Democratic Party	Democratic Party	TRUE
North Carolina	0.48	0.51	Competitive	Republican Party	Republican Party	TRUE
North Dakota	0.31	0.67	Republican	Republican Party	Republican Party	TRUE
Ohio	0.44	0.55	Republican	Republican Party	Republican Party	TRUE
Oklahoma	0.32	0.66	Republican	Republican Party	Republican Party	TRUE
Oregon	0.55	0.41	Solid Democrat	Democratic Party	Democratic Party	TRUE
Pennsylvania	0.49	0.50	Competitive	Republican Party	Democratic Party	FALSE
Rhode Island	0.56	0.42	Solid Democrat	Democratic Party	Democratic Party	TRUE
South Carolina	0.40	0.58	Republican	Republican Party	Republican Party	TRUE
South Dakota	0.34	0.63	Republican	Republican Party	Republican Party	TRUE
Tennessee	0.34	0.64	Republican	Republican Party	Republican Party	TRUE
Texas	0.42	0.56	Lean Republican	Republican Party	Republican Party	TRUE
Utah	0.38	0.59	Republican	Republican Party	Republican Party	TRUE
Vermont	0.64	0.32	Solid Democrat	Democratic Party	Democratic Party	TRUE
Virginia	0.52	0.46	Lean Democrat	Democratic Party	Democratic Party	TRUE
Washington	0.57	0.39	Solid Democrat	Democratic Party	Democratic Party	TRUE
West Virginia	0.28	0.70	Republican	Republican Party	Republican Party	TRUE
Wisconsin	0.49	0.50	Competitive	Republican Party	Democratic Party	FALSE
Wyoming	0.26	0.72	Republican	Republican Party	Republican Party	TRUE

```r
act_vs_res2 <- act_vs_res %>%
  drop_na(prediction_2024) %>%
  mutate(prediction_2024 = as.factor(prediction_2024),
         actual_2024 = as.factor(actual_2024))

# Create confusion matrix
conf_matrix <- confusionMatrix(act_vs_res2$prediction_2024, act_vs_res2$actual_2024)

# Extract the confusion matrix table
cm_table <- as.data.frame(conf_matrix$table)

# Plot confusion matrix using ggplot2
```

```r
ggplot(cm_table, aes(x = Prediction, y = Reference, fill = Freq)) +
  geom_tile(color = "white") +
  scale_fill_gradient(low = "white", high = "steelblue") +
  geom_text(aes(label = Freq), vjust = 1) +
  theme_minimal() +
  labs(
    title = "2024 Election results Confusion Matrix",
    x = "Predicted",
    y = "Actual"
  )
```




```r
#incorrect predictions
act_vs_res %>%
  filter(correctly_predicted== FALSE)%>%
  kableExtra::kable() %>%
  kableExtra::kable_minimal()
```

State	Democrat	Republican	type	actual_2024	prediction_2024	correctly_predicted
Georgia	0.49	0.51	Competitive	Republican Party	Democratic Party	FALSE
Michigan	0.48	0.50	Competitive	Republican Party	Democratic Party	FALSE
Nevada	0.47	0.51	Competitive	Republican Party	Democratic Party	FALSE
Pennsylvania	0.49	0.50	Competitive	Republican Party	Democratic Party	FALSE
Wisconsin	0.49	0.50	Competitive	Republican Party	Democratic Party	FALSE