Prediction Assigment

Khristel Zavaleta

2022-10-30

Contents

1 Part 1: Code and Data used to produce predictions

1.1 Cleaning code and Merging data frames

data sets used can be found in: https://github.com/khristel26/Senate.git

- **Defining the States running**

```
states_running <- read_xlsx("States running.xlsx")
```

1.1.1 Cleaning data: Historical votes from senate

```
senate_hist <- read_csv("1976-2020-senate.csv") # Source: Mit Election Data

senate_hist <- senate_hist %>%
  filter(state_po %in% states_running$state_po, stage != "pre") %>%
  filter(!(state_po == "GA" & year == 2020)) %>%
  filter(party_simplified == "REPUBLICAN" | party_simplified == "DEMOCRAT") %>%
  select(year, state, state_po, stage, special, party_simplified, candidatevotes, totalvotes)

senate_hist <- senate_hist %>%
  group_by(party_simplified, year, state, state_po, special, stage, totalvotes) %>%
  summarise(candidate_votes = sum(candidatevotes))

senate_hist <- senate_hist %>%
  pivot_wider(names_from = "party_simplified", values_from = "candidate_votes")
```

1.1.2 Including presidential approval variable

I Filter the presidential approval values that correspond to the dates closest to the Senate election dates.

```
pres_approval_ratings <- read_xlsx("pres_approval_ratings_1948_2016.xlsx")

pres_approval_ratings <- pres_approval_ratings %>%
  separate(
    col = end,
    into = c("year", "month", "day"),
    sep = "-"
  )

pres_approval_ratings <- pres_approval_ratings %>%
  filter(year %in% senate_hist$year) %>%
  filter(month != 12)


pres_approval_ratings$month <- as.numeric(pres_approval_ratings$month)
pres_approval_ratings$day <- as.numeric(pres_approval_ratings$day)

pres_approval_ratings <- pres_approval_ratings %>%
  group_by(year) %>%
  filter(month == month[which.min(11 - month)]) %>%
  filter(day == day[which.min(abs(8 - day))])

# Erasing duplicates
pres_approval_ratings <- pres_approval_ratings[!duplicated(pres_approval_ratings), ]

# Completing information for data that was not in the dataframe

df <- data_frame(
  year = c("2018", "2020", "2021", "2022"),
  President = c("Donald Trump", "Donald Trump", "Joe Biden", "Joe Biden"),
  approving = c(39, 43, 42, 40),
  disapproving = c(55, 55, 55, 55),
  unsure = c(6, 2, 3, 5)
)

pres_approval_ratings <- rbind(pres_approval_ratings, df[c(1, 2, 3, 4), ])

pres_approval_ratings <- pres_approval_ratings %>%
  select(year, approving, disapproving, unsure)

# Reading a presidential party doc

pres_party <- read_xlsx("pres_party.xlsx")

pres_approval_ratings <- merge(pres_approval_ratings,
  pres_party[, c("year", "pres_party")],
  by = c("year"), all.x = TRUE
)
```

- Merging the data with the main data frame

```
senate_hist <- merge(senate_hist, pres_approval_ratings, by = c("year"), all.x = TRUE)

senate_hist <- senate_hist %>%
  rename(pres_aprov = approving, pres_disapp = disapproving, pres_unsure = unsure)

# Converting the pres party in dummy

senate_hist <- senate_hist %>%
  mutate(party_pres_dem = ifelse(pres_party == "democrat", 1, 0))
```

1.1.3 Including inflation variable

```
inflation <- read_csv("united-states-inflation-rate-cpi.csv", skip = 16)[, -4]

inflation <- inflation %>%
  separate(
    col = date,
    into = c("year", "month", "day"),
    sep = "-"
  ) %>%
  select(-c("month", "day"))

inflation <- inflation %>%
  add_row(year = "2022", `Inflation Rate (%)` = 8.32, `Annual Change` = 3.62)
```

- Merging inflation with historical data

```
senate_hist <- merge(senate_hist, inflation, by = c("year"), all.x = TRUE)
```

1.1.4 Including states with no abortion protection variable

```
senate_hist <- merge(senate_hist, states_running[, c(
  "state_po",
  "State.C", "right_abortion"
)],
by = c("state_po"), all.x = TRUE
)

senate_hist <- senate_hist %>%
  mutate(abortion_issue = ifelse(right_abortion == "Yes", 0, 1)) %>%
  rename(STATE = State.C)
```

1.1.5 Cleaning data: Senate polls historical

```
states_usa_ab <- cbind(as.data.frame(state.abb), as.data.frame(state.name))
```

```
senate_polls_hist <- read_csv("senate_polls_historical.csv")

senate_polls_hist$end_date <- as.Date(senate_polls_hist$end_date, format = "%m/%d/%y")
senate_polls_hist$election_date <- as.Date(senate_polls_hist$election_date, format = "%m/%d/%y")

senate_polls_hist <- merge(senate_polls_hist, states_usa_ab, by.x = c("state"), by.y = c("state.name"))

senate_polls_hist <- senate_polls_hist %>%
  filter(stage != "jungle primary") %>%
  filter(state.abb %in% states_running$state_po) %>%
  arrange(election_date)

senate_polls_hist[2682, 34] <- as.Date(as.character("2020-12-05"), format = "%Y-%m-%d")
senate_polls_hist[2683, 34] <- as.Date(as.character("2020-12-05"), format = "%Y-%m-%d")

senate_polls_hist <- senate_polls_hist %>%
  group_by(state) %>%
  filter(end_date == end_date[which.min(election_date - end_date)])

# Having more than one polling close to the election day, we will summarize the data

senate_polls_hist <- senate_polls_hist %>%
  group_by(state, candidate_name, end_date, stage, party, seat_number) %>%
  summarise(poll_votes = mean(pct))

# Identifying special elections

senate_polls_hist <- senate_polls_hist %>%
  mutate(special = ifelse(seat_number == 2, TRUE, FALSE)) %>%
  ungroup() %>%
  select(-c(seat_number, candidate_name))

senate_polls_hist <- senate_polls_hist %>%
  mutate(
    party_2 =
      ifelse(party == "LIB" | party == "GRE" | party == "IND" | party == "PG",
        "other", party
      )
  )

senate_polls_hist <- senate_polls_hist %>%
  group_by(state, end_date, stage, special, party_2) %>%
  summarise(vote_poll = mean(poll_votes)) %>%
  filter(state != "California")

# Pivot wider
senate_polls_vf <- senate_polls_hist %>%
  pivot_wider(names_from = "party_2", values_from = "vote_poll") %>%
  separate(
```

```
    col = end_date,
    into = c("year", "poll_month", "poll_day"),
    sep = "-"
  ) %>%
  rename(STATE = state)
```

- Merging with the hist data set

```
senate_polls_vf <- merge(senate_hist,
  senate_polls_vf[, c("STATE", "year", "special", "DEM", "REP", "other")],
  by = c("STATE", "year", "special"), all.x = TRUE
)
```

```
#write.csv(senate_polls_vf, "senate_polls_vf.csv", row.names = FALSE)
senate <- read.csv("senate_historical_1978-2020.csv")[, 1:20]
```

1.1.6 Including GDP growth variable

```
gdp <- read_csv("united-states-gdp-growth-rate.csv", skip = 16)[, 1:3]
gdp$date <- as.Date(gdp$date, format = "%Y/%m/%d")
gdp$year <- year(gdp$date)
gdp <- gdp %>%
  rename(gdp_growth = `GDP Growth (%)`)
```

- Merge the data

```
senate_vf <- merge(senate, gdp[, c("year", "gdp_growth")], by = c("year"), all.x = TRUE)
```

1.1.7 Cleaning data of Senate elections 2022 polls

```
senate_polls_2022 <- read_csv("senate_polls.csv")[, 11:42]

senate_polls_2022 <- senate_polls_2022[, -c(4:14)]


senate_polls_2022$end_date <- as.Date(senate_polls_2022$end_date, format = "%m/%d/%y")
senate_polls_2022$election_date <- as.Date(senate_polls_2022$election_date, format = "%m/%d/%y")

senate_polls_2022 <- senate_polls_2022 %>%
  group_by(state, seat_number) %>%
  filter(end_date == end_date[which.min(election_date - end_date)]) %>%
  filter(party == "DEM" | party == "REP") %>%
  filter(election_date == as.Date(as.character("2022-11-08"), format = "%Y-%m-%d"))

senate_polls_2022 <- senate_polls_2022 %>%
  group_by(state, party, seat_number) %>%
  summarise(poll_votes = mean(pct))

senate_polls_2022 <- senate_polls_2022 %>%
  pivot_wider(names_from = "party", values_from = "poll_votes")

senate_polls_2022 <- senate_polls_2022 %>%
  mutate(poll_dem = DEM / (DEM + REP) * 100, poll_rep = REP / (DEM + REP) * 100) %>%
  mutate(poll_favor_dem = ifelse(DEM > REP, 1, 0)) %>%
  mutate(special = ifelse(seat_number == 2, FALSE, TRUE)) %>%
  select(-c(seat_number))

senate_polls_2022 <- merge(senate_polls_2022, states_running,
  by.x = c("state"),
  by.y = c("State.C"), all.y = TRUE
)

senate_polls_2022 <- senate_polls_2022 %>%
  mutate(year = 2022, gdp_growth = -0.13)

senate_polls_2022 <- merge(senate_polls_2022, pres_approval_ratings, by = c("year"))

senate_polls_2022 <- merge(senate_polls_2022, inflation, by = c("year"))

senate_polls_2022 <- senate_polls_2022 %>%
  mutate(
    party_pres_dem = ifelse(pres_party == "democrat", 1, 0),
    abortion_issue = ifelse(right_abortion == "YES", 0, 1)
  )

#Completing the polls that we didnt have in the data
#Hawai
senate_polls_2022[10, 3] <- 65.4
senate_polls_2022[10, 4] <- 30
senate_polls_2022[10, 5] <- 68.553
senate_polls_2022[10, 6] <- 31.447
```

```r
senate_polls_2022[10, 7] <- 1
#Idaho
senate_polls_2022[11, 3] <- 27
senate_polls_2022[11, 4] <- 66.2
senate_polls_2022[11, 5] <- 28.97
senate_polls_2022[11, 6] <- 71.03
senate_polls_2022[11, 7] <- 0

#NH
senate_polls_2022[21, 3] <- 50
senate_polls_2022[21, 4] <- 46
senate_polls_2022[21, 5] <- 52.0833
senate_polls_2022[21, 6] <- 47.9167
senate_polls_2022[21, 7] <- 1

#NY
senate_polls_2022[22, 3] <- 57
senate_polls_2022[22, 4] <- 39
senate_polls_2022[22, 5] <- 59.3750
senate_polls_2022[22, 6] <- 40.6250
senate_polls_2022[22, 7] <- 1

#NC
senate_polls_2022[23, 3] <- 43
senate_polls_2022[23, 4] <- 50
senate_polls_2022[23, 5] <- 46.2366
senate_polls_2022[23, 6] <- 53.7634
senate_polls_2022[23, 7] <- 0

#ND
senate_polls_2022[24, 3] <- 25
senate_polls_2022[24, 4] <- 70.3
senate_polls_2022[24, 5] <- 26.2329
senate_polls_2022[24, 6] <- 73.7671
senate_polls_2022[24, 7] <- 0

#SC
senate_polls_2022[30, 3] <- 37.5
senate_polls_2022[30, 4] <- 62.5
senate_polls_2022[30, 5] <- 37.50
senate_polls_2022[30, 6] <- 62.500
senate_polls_2022[30, 7] <- 0

#SD
senate_polls_2022[31, 3] <- 32.2
senate_polls_2022[31, 4] <- 64.4
senate_polls_2022[31, 5] <- 33.3333
senate_polls_2022[31, 6] <- 66.6667
senate_polls_2022[31, 7] <- 0

senate_polls_2022[32, 7] <- 0
```

1.2 Regressions

1.2.1 Regression 1

```
reg1 <- summary(lm(share_dem ~ poll_dem + poll_favor_dem +
  party_pres_dem * gdp_growth, data = senate_vf))
reg1
```

```
##
## Call:
## lm(formula = share_dem ~ poll_dem + poll_favor_dem + party_pres_dem *
##     gdp_growth, data = senate_vf)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -14.327  -1.404   0.216   1.601   6.786
##
## Coefficients:
##                            Estimate Std. Error t value Pr(>|t|)
## (Intercept)                 0.78731    2.26456   0.348  0.72931
## poll_dem                    0.93183    0.05181  17.986  < 2e-16 ***
## poll_favor_dem              1.91485    1.27252   1.505  0.13763
## party_pres_dem             -1.15148    1.39925  -0.823  0.41381
## gdp_growth                  0.51937    0.17962   2.892  0.00533 **
## party_pres_dem:gdp_growth -0.21309    0.61660  -0.346  0.73086
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.113 on 60 degrees of freedom
##   (457 observations deleted due to missingness)
## Multiple R-squared:  0.9431, Adjusted R-squared:  0.9384
## F-statistic:   199 on 5 and 60 DF,  p-value: < 2.2e-16
```

- Generating the predicting model

```
senate_regressions <- senate_vf %>%
  mutate(predict1 = reg1$coefficients[1] + reg1$coefficients[2] * poll_dem +
    reg1$coefficients[3] * poll_favor_dem + reg1$coefficients[4] * party_pres_dem +
    reg1$coefficients[5] * gdp_growth + reg1$coefficients[6] * party_pres_dem * gdp_growth)
```

1.2.2 Regression 2

```
reg2 <- summary(lm(share_dem ~ poll_dem + poll_favor_dem + abortion_issue +
  party_pres_dem * inflation_rate, data = senate_vf))
reg2
```

```
##
## Call:
## lm(formula = share_dem ~ poll_dem + poll_favor_dem + abortion_issue +
##     party_pres_dem * inflation_rate, data = senate_vf)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -13.3054  -1.5966   0.0907   1.5895   6.7101
##
## Coefficients:
##                                Estimate Std. Error t value Pr(>|t|)
## (Intercept)                    -6.15478    3.22863  -1.906  0.06148 .
## poll_dem                        0.95549    0.05653  16.904  < 2e-16 ***
## poll_favor_dem                  2.17476    1.29580   1.678  0.09858 .
## abortion_issue                  1.14101    1.09438   1.043  0.30138
## party_pres_dem                  4.19875    2.11002   1.990  0.05124 .
## inflation_rate                  2.58326    0.94744   2.727  0.00841 **
## party_pres_dem:inflation_rate -2.42986    1.15351  -2.106  0.03942 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.111 on 59 degrees of freedom
##   (457 observations deleted due to missingness)
## Multiple R-squared:  0.9441, Adjusted R-squared:  0.9385
## F-statistic: 166.2 on 6 and 59 DF,  p-value: < 2.2e-16
```

- Generating the predicting model

```
senate_regressions <- senate_regressions %>%
  mutate(predict2 = reg2$coefficients[1] + reg2$coefficients[2] * poll_dem +
    reg2$coefficients[3] * poll_favor_dem + reg2$coefficients[4] *
      abortion_issue + reg2$coefficients[5] * party_pres_dem +
    reg2$coefficients[6] * inflation_rate +
    reg2$coefficients[7] * party_pres_dem * inflation_rate)
```

1.2.3 Regression 3

```
reg3 <- summary(lm(share_dem ~ poll_dem + pres_disapp + poll_favor_dem +
  Annual.Change, data = senate_vf))
reg3
```

```
##
## Call:
## lm(formula = share_dem ~ poll_dem + pres_disapp + poll_favor_dem +
##     Annual.Change, data = senate_vf)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -14.2057  -1.4185   0.1523   1.8758   6.4627
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)    -6.42569    4.99200  -1.287    0.203
## poll_dem        0.94962    0.05334  17.803   <2e-16 ***
## pres_disapp     0.11280    0.08302   1.359    0.179
## poll_favor_dem  1.72971    1.32096   1.309    0.195
## Annual.Change   0.91744    0.60660   1.512    0.136
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.237 on 61 degrees of freedom
##   (457 observations deleted due to missingness)
## Multiple R-squared:  0.9375, Adjusted R-squared:  0.9334
## F-statistic: 228.6 on 4 and 61 DF,  p-value: < 2.2e-16
```

- Generating the predicting model

```
senate_regressions <- senate_regressions %>%
  mutate(predict3 = reg3$coefficients[1] + reg3$coefficients[2] * poll_dem +
    reg3$coefficients[3] * pres_disapp + reg3$coefficients[4] * poll_favor_dem +
    reg3$coefficients[5] * Annual.Change)
```

1.2.4 In sample errors:

- Finding the residuals

```
senate_regressions <- senate_regressions %>%
  mutate(
    residuals1 = abs(share_dem - predict1),
    residuals2 = abs(share_dem - predict2),
    residuals3 = abs(share_dem - predict3)
  )

senate_regressions %>%
  summarise(
    mean_res1 = mean(residuals1, na.rm = TRUE), mean_res2 =
      mean(residuals2, na.rm = TRUE), mean_res3 = mean(residuals3, na.rm = TRUE)
  )
```

```
##   mean_res1 mean_res2 mean_res3
## 1  2.037883  2.051735  2.132205
```

1.2.5 Calculating MAE: Out of sample validation

```
senate_regression <- senate_regressions %>%
  arrange(residuals1)

# Filtering for non NA values in regressions
senate_regression <- senate_regression[1:66, ] %>%
  arrange(year)

# Subsetting the data for the 3-fold cross validation
senate_regression <- senate_regression %>%
  mutate(random = runif(66)) %>%
  arrange(random)

senate_regression$partition[senate_regression$random < 1 / 3] <- 0
senate_regression$partition[senate_regression$random >= 1 / 3 &
  senate_regression$random < 2 / 3] <- 1
senate_regression$partition[senate_regression$random >= 2 / 3] <- 2

senate_regression <- senate_regression %>%
  mutate(
    test_error_p1 = NA,
    test_error_p2 = NA,
    test_error_p3 = NA
  )

# Train errors for first prediction model

for (i in 0:2) {
  model_p1 <- lm(share_dem ~ poll_dem + poll_favor_dem +
                   party_pres_dem * gdp_growth,
```

```
    data = senate_regression[senate_regression$partition != i, ]
  )
  senate_regression$yhat_1 <- predict(model_p1,
                                      newdata = senate_regression)
  senate_regression[[paste0("train_error_p1_", i)]] <-
    senate_regression$share_dem - senate_regression$yhat_1
  senate_regression$test_error_p1 <-
    ifelse(senate_regression$partition == i,
           senate_regression[[paste0("train_error_p1_", i)]],
           senate_regression$test_error_p1)
}

# Train errors for second prediction model

for (i in 0:2) {
  model_p2 <- lm(share_dem ~ poll_dem + poll_favor_dem + abortion_issue +
    party_pres_dem * inflation_rate,
  data = senate_regression[senate_regression$partition != i, ]
  )
  senate_regression$yhat_2 <- predict(model_p2, newdata = senate_regression)
  senate_regression[[paste0("train_error_p2_", i)]] <-
    senate_regression$share_dem - senate_regression$yhat_2
  senate_regression$test_error_p2 <-
    ifelse(senate_regression$partition == i,
      senate_regression[[paste0("train_error_p2_", i)]],
      senate_regression$test_error_p2
    )
}

# Train errors for third prediction model

for (i in 0:2) {
  model_p3 <- lm(share_dem ~ poll_dem + pres_disapp +
    poll_favor_dem + Annual.Change,
  data = senate_regression[senate_regression$partition != i, ]
  )
  senate_regression$yhat_3 <- predict(model_p3, newdata = senate_regression)
  senate_regression[[paste0("train_error_p3_", i)]] <-
    senate_regression$share_dem - senate_regression$yhat_3
  senate_regression$test_error_p3 <-
    ifelse(senate_regression$partition == i,
      senate_regression[[paste0("train_error_p3_", i)]],
      senate_regression$test_error_p3
    )
}
```

- **Comparing out of sample Performance - Mean of absolute residuals**

```
senate_regression %>%
  summarise(
    model1_MAE = mean(abs(test_error_p1)), model2_MAE = mean(abs(test_error_p2)),
    model3_MAE = mean(abs(test_error_p3))
  )
```

```
##   model1_MAE model2_MAE model3_MAE
## 1   2.152873   2.145575   2.207093
```

1.2.6 Creatting the Senate Election Prediction for 2022

- **Using the fist model**

```
senate_polls_2022 <- senate_polls_2022 %>%
  mutate(prediction_dem = reg1$coefficients[1]
  + reg1$coefficients[2] * poll_dem
    + reg1$coefficients[3] * poll_favor_dem
    + reg1$coefficients[4] * party_pres_dem
    + reg1$coefficients[5] * gdp_growth
    + reg1$coefficients[6] * party_pres_dem * gdp_growth)

senate_polls_2022$prediction_dem[is.na(senate_polls_2022$prediction_dem)] <- 0
```

2 Part 2: Democratic two-party vote share

2.1 List of the Democratic two-party vote share [Dem votes/(Dem Votes + Rep Votes)]

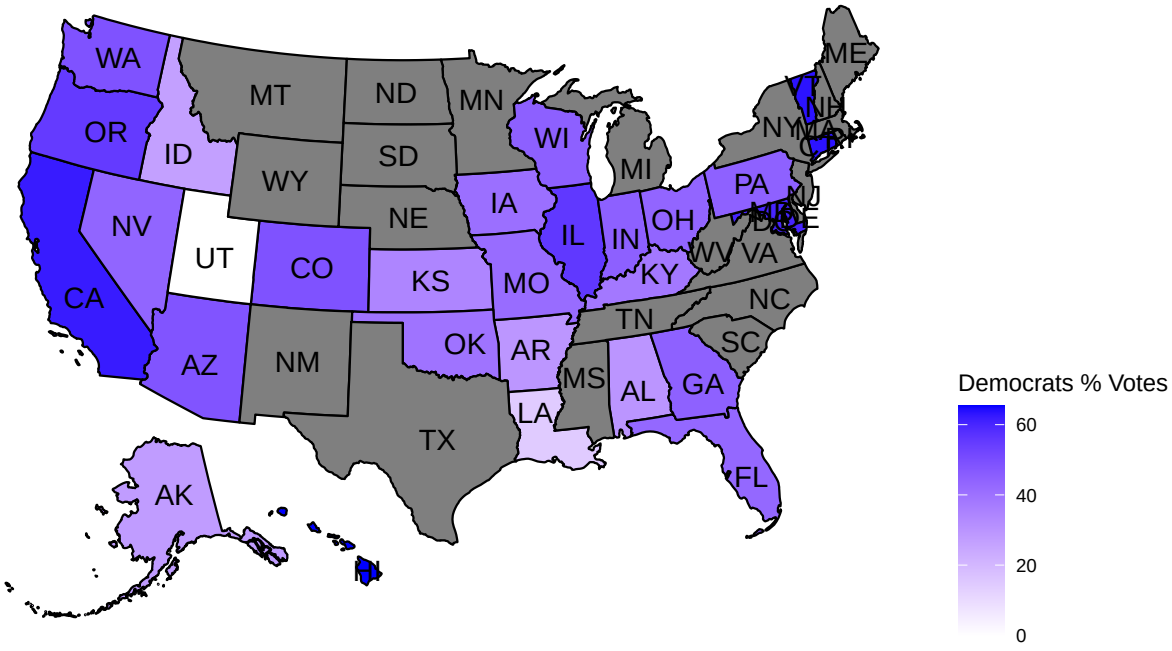
```
senate_polls_2022 %>%
  mutate(pred_share_dem = prediction_dem / 100) %>%
  select(state, pred_share_dem)
```

```
##             state pred_share_dem
## 1         Alabama      0.2988599
## 2          Alaska      0.2792646
## 3         Arizona      0.4839630
## 4        Arkansas      0.2949801
## 5      California      0.6207980
## 6        Colorado      0.4884821
## 7     Connecticut      0.6322946
## 8         Florida      0.4282523
## 9         Georgia      0.4479967
## 10         Hawaii      0.6539059
## 11          Idaho      0.2659112
## 12       Illinois      0.5586760
## 13        Indiana      0.4496141
## 14           Iowa      0.4078559
## 15         Kansas      0.3459330
## 16       Kentucky      0.3825703
## 17      Louisiana      0.1438696
## 18       Maryland      0.6014285
## 19       Missouri      0.4157955
## 20         Nevada      0.4368465
## 21  New Hampshire      0.5004363
## 22       New York      0.5683825
## 23 North Carolina      0.4268065
## 24   North Dakota      0.2404061
## 25           Ohio      0.4362753
## 26       Oklahoma      0.3950747
## 27       Oklahoma      0.3615320
## 28         Oregon      0.5519236
## 29   Pennsylvania      0.4465152
## 30 South Carolina      0.3453963
## 31   South Dakota      0.3065697
## 32           Utah      0.0000000
## 33        Vermont      0.6330589
## 34     Washington      0.4889204
## 35      Wisconsin      0.4496783
```

2.2 Election Maps

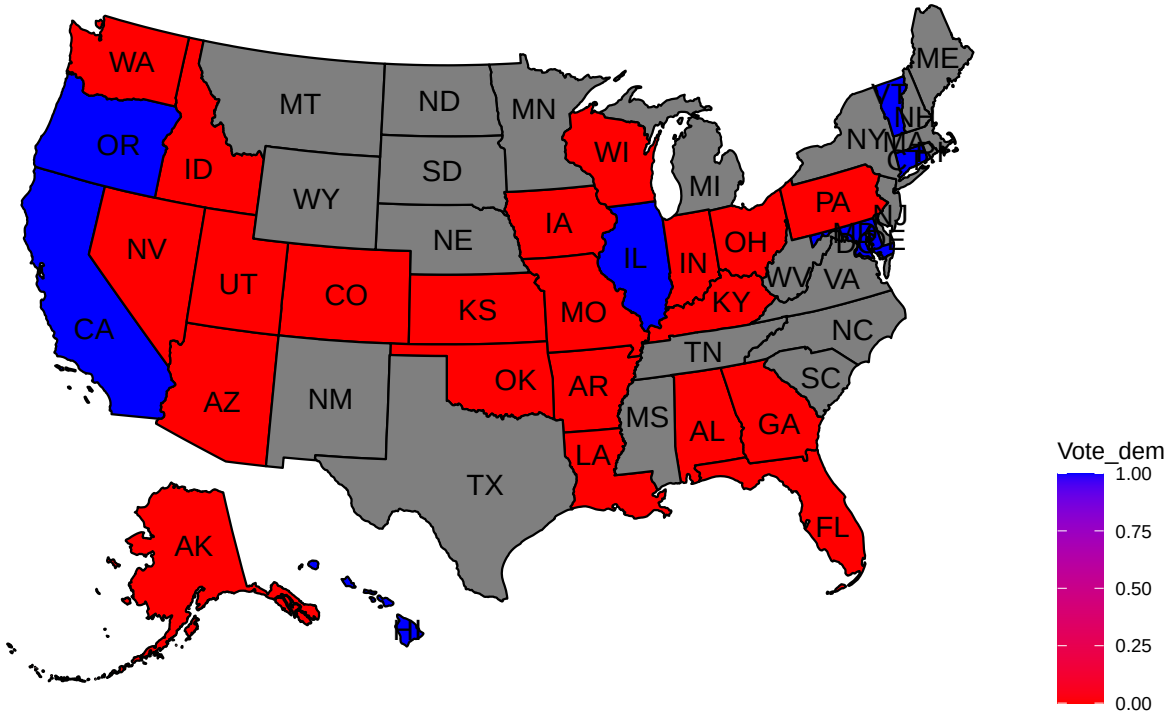
Democratic two-party vote percentage

```
library(usmap)
plot_usmap(data = senate_polls_2022, values = "prediction_dem", labels = TRUE) +
  scale_fill_continuous(low = "white", high = "blue", name = "Democrats % Votes") +
  theme(legend.position = "right")
```



Democratic two-party winning in blue, losing in red *1 = Dem win, 0 = Dem lose*

```
senate_polls_2022 <- senate_polls_2022 %>%
  mutate(dem_win = (ifelse(prediction_dem > 50, 1, 0)))
plot_usmap(data = senate_polls_2022, values = "dem_win", labels = TRUE) +
  scale_fill_continuous(low = "red", high = "blue", name = "Vote_dem", label = scales::comma) +
  theme(legend.position = "right")
```



3 Part 3: Report (max 5 pages)

The estimated models were the following:

Dependent Variable:	share_dem		
Model:	(1)	(2)	(3)
Variables			
(Intercept)	0.7873	-6.155	-6.426
	(2.265)	(3.229)	(4.992)
poll_dem	0.9318	0.9555	0.9496
	(0.0518)	(0.0565)	(0.0533)
poll_favor_dem	1.915	2.175	1.730
	(1.273)	(1.296)	(1.321)
party_pres_dem	-1.151	4.199	
	(1.399)	(2.110)	
gdp_growth	0.5194		
	(0.1796)		
party_pres_dem × gdp_growth	-0.2131		
	(0.6166)		
abortion_issue		1.141	
		(1.094)	
inflation_rate		2.583	
		(0.9474)	
party_pres_dem × inflation_rate		-2.430	
		(1.154)	
pres_disapp			0.1128
			(0.0830)
Annual.Change			0.9174
			(0.6066)
Fit statistics			
R^2	0.94312	0.94415	0.93746

IID standard-errors in parentheses

The model chosen (Model 1) to estimate the predictions 2022 contains 4 variables:

- Poll_dem: historical data from senate election polls has been collected. Likewise, this data has been filtered for the states of interest in the 2022 elections.
- Poll_favor_dem: a dummy variable has been created that indicates 1 if the poll favors the Democratic party and 0 if it does not. This variable gives greater weight to the surveys. This variable has been chosen, since in past elections, despite the fact that the polls showed a very close value of votes between the Republican and Democratic parties, almost always the inclination that was granted towards one of them was correct.
- Party_pres_dem: dummy variable that indicates 1 if the president's party at the time of the election is Democratic and 0 if it is not.
- gdp_growth: this variable interacts with the Party_pres_dem variable. We understand that GDP growth affects the population in many aspects and could reflect their well-being.

Likewise, the chosen model has been compared with two other models, which in addition to taking into account the variables mentioned above, use the following variables:

- abortion_issue: Since it is a topic that has provoked many reactions within the United States, we have incorporated this dummy variable that indicates 1 to the states that have protected the right to abortion (Alaska, California, Colorado, Connecticut, Delaware, Hawaii, Illinois, Kansas, Maine, Maryland, Massachusetts, Minnesota, Nevada, New Hampshire, New Jersey, New Mexico, New York, Oregon, Rhode Island, Vermont, and Washington) and 0 for the other states (with the understanding that although in all the others is not illegal, but abortion is a topic of discussion on the public agenda). Although this topic is expected to have a greater influence in 2022 than in the previous elections, we wanted to see the interaction with the real votes obtained in historical voting.
- Inflation_rate: Like the GDP, inflation impacts people's well-being and will allow us to observe the influence on the vote.
- Inflation annual_change: Like the GDP, inflation impacts people's well-being and will allow us to observe the influence on the vote.
- pres_disapp: The presidential disapproval has been chosen because it allows eliciting the sentiment of the voters towards politics and specifically the government

- **In-sample and out-of-sample performance**

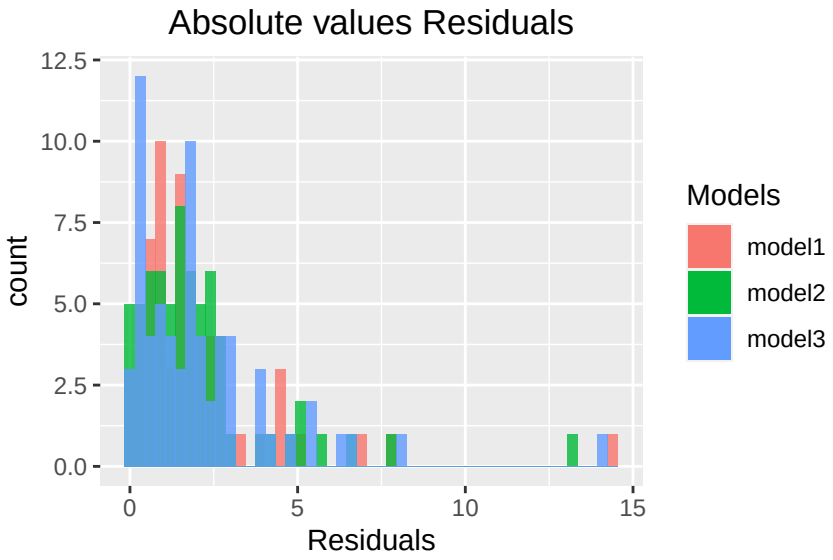
From those models, we performed an in-sample and out-of-sample performance evaluation, which allowed us to conclude that the model that best fit the data was Model 1.

Regarding the in-sample performance, we obtained the following error tests for each of the models:

Mean absolute errors

```
##     Model1   Model2   Model3
## 1 2.037883 2.051735 2.132205
```

Total values absolute errors in graph



Likewise, we perform an out of sample performance through Cross Validation, then we calculate the Mean Absolute Error (MAE) and compare the models.

Mean absolute errors

```
##   Model1_MAE Model2_MAE Model3_MAE
## 1   2.152873   2.145575   2.207093
```

With these data, we can see that the model that minimizes the predictor errors is model 1, since it obtains an error of **2.037883** in the first test and another of **2.298171** in the second test. It should be noted that these numbers are in percentage, since we are working with 100% votes.

The process carried out to find the in sample and out sample performance can be seen in the section 1.2.4 and 1.2.5 of this document

- **Prediction 2022 Senate Elections**

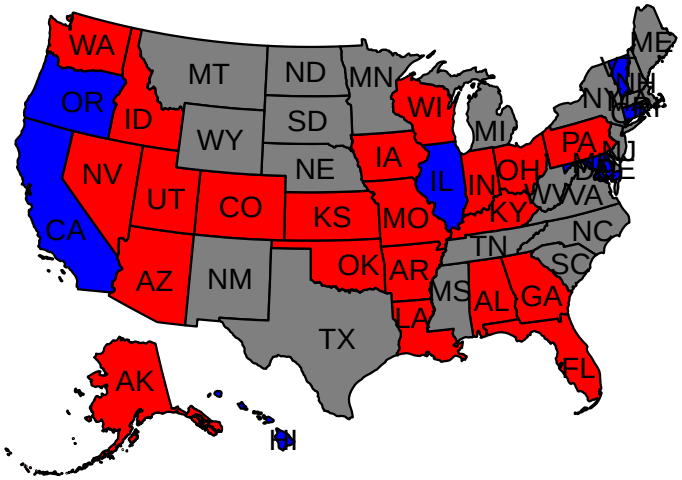
Finally, we use model 1, described in the first table shown in this section, and calculate the Democratic two-party vote share:

```
##             State Dem Share Prediction
## 1         Alabama            0.2988599
## 2          Alaska            0.2792646
## 3         Arizona            0.4839630
## 4        Arkansas            0.2949801
## 5      California            0.6207980
## 6        Colorado            0.4884821
## 7     Connecticut            0.6322946
## 8         Florida            0.4282523
## 9         Georgia            0.4479967
## 10         Hawaii            0.6539059
## 11          Idaho            0.2659112
## 12       Illinois            0.5586760
## 13        Indiana            0.4496141
## 14           Iowa            0.4078559
## 15         Kansas            0.3459330
## 16       Kentucky            0.3825703
## 17      Louisiana            0.1438696
## 18       Maryland            0.6014285
## 19       Missouri            0.4157955
## 20         Nevada            0.4368465
## 21  New Hampshire            0.5004363
## 22       New York            0.5683825
## 23 North Carolina            0.4268065
## 24   North Dakota            0.2404061
## 25           Ohio            0.4362753
## 26       Oklahoma            0.3950747
## 27       Oklahoma            0.3615320
## 28         Oregon            0.5519236
## 29   Pennsylvania            0.4465152
## 30 South Carolina            0.3453963
## 31   South Dakota            0.3065697
## 32           Utah            0.0000000
## 33        Vermont            0.6330589
```

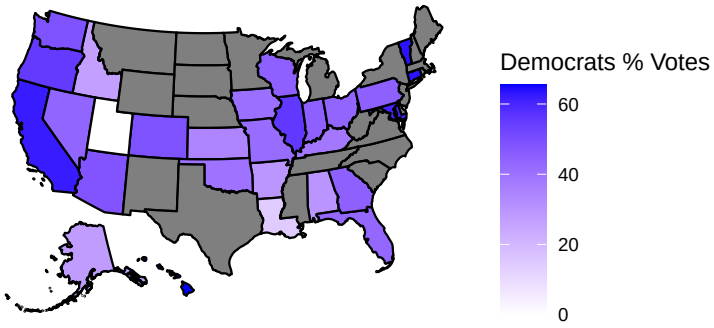
```
## 34      Washington            0.4889204
## 35       Wisconsin            0.4496783
```

In that sense, of the 35 senate seats up for election in the midterms (from 34 states and 1 special election) we have predicted that there would be 9 states that would be winning the Democratic party: California, Connecticut, Hawaii, Illinois, Maryland, New Hampshire (it is within the standard deviation and could also be won by the Republican party), New York, Oregon, Vermont). Also, it should be noted that within the standard deviation of our model, the following three states could also have control of the Democratic party: Arizona, Colorado and Washington, making a possible total of 12 seats for the Democratic Party.

Democratic two-party winning in blue, losing in red *1 = Dem win, 0 = Dem lose*



Democratic two-party vote percentage



- **Analysis of the chosen model**

Missing Data: It is important to note that, despite historical information on the election process dating back to 1976, we were unable to locate polling data that go back to that year. The pollings that were gathered started in 2014, which reduces the total number of observations in the data.

The absence of historical polling data going back to 1976 is not necessarily a bad thing, either, as these numbers may not be accurate predictors of current votes given their age. Also to be mentioned are statistics on inflation, gdp growth, and approval ratings going back to 1976.

Similarly, it was noted that states, like Hawaii, Idaho, South Dakota and North Dakota do not have polling that helps enhance the projections for these states in order to produce the 2022 estimations. Since the base of senate polling 2022 that I used lacked data for these states, polling data from the internet had to be manually imported to R in circumstances like these. We can discuss missing data that compromises the accuracy of our coefficients in that manner.

The relationship between GDP growth and inflation was discovered to be linear, yielding NA in regressions involving both variables. As a result, we could only utilize one of these variables in each model.

Likewise, it was observed that the MAE for the three models was close, being more similar that of model 1 and model 2.

Quality of data: The quality of the polling stations was not considered while selecting them, but rather which ones were closest to the election date. If there were multiple polling locations close to the election day, an average was calculated. We know that this can have an impact on the statistics since the pollsters with the best prediction quality may have been removed. The pollsters' sample size was also not taken into consideration.

The same thing occurred with the presidential approval polls; the results that were closest to the votes were used, regardless of the pollster or sample size.

It is important to note that in some states, more than one candidate from each party may compete, and if no candidate reaches 50% of the votes they will move to a runoff election. The support received by party was added in elections when more than one candidate ran for the Democratic or Republican parties. Similarly, the runoffs in states like Georgia have also been included when analyzing historical polling data. We are aware that this can degrade the accuracy of our predictions.