VoteMatch - preprocessing

Imported dataset

Selecting variables

Before building any predictive model, we begin by selecting a set of variables to explore their relationships with the target variable (`pes21_votechoice2021`), as well as among themselves.

- `feature_vars` contains variables potentially relevant predictors we need to explore.

In addition, two variable sets are included for diagnostic purposes: - `check_disengaged`: used to identify politically disengaged respondents. - `check_low_quality`: used to flag low-quality or problematic responses (e.g., duplicates, speeders, inattentiveness).

We combine all of these into a single dataframe (`ces_selected`) to conduct correlation testing in the next step.

```
vote_choices <- c("pes21_votechoice2021", "cps21_votechoice","pes21_votechoice2021_7_TEXT",
                  "cps21_votechoice_6_TEXT")
mia_vars <- c("pes21_province", "cps21_age", "pes21_follow_pol", "pes21_rural_urban",
              "pes21_inequal", "pes21_abort2", "pes21_contact1", "Region", "cps21_marital",
              "cps21_imm_year", "cps21_bornin_canada", "cps21_rel_imp", "cps21_volunteer")

extra_vars <- c("cps21_education","pes21_lived", "cps21_fed_gov_sat", "Duration__in_seconds_")

# Merge selected variables
feature_vars <- unique(c(mia_vars, extra_vars))


# use for check data quality
check_disengaged <- c("pes21_follow_pol", "cps21_interest_gen_1", "cps21_interest_elxn_1",
                      "cps21_news_cons","cps21_govt_confusing")

check_low_quality <- c("cps21_duplicates_pid_flag", "cps21_duplicate_ip_demo_flag",
                       "pes21_speeder_low_quality","pes21_duplicates_pid_flag",
                       "cps21_inattentive","pes21_inattentive")

selected_var <- unique(c(vote_choices, feature_vars, check_disengaged, check_low_quality))
ces_selected <- ces2021 %>% select(all_of(selected_var))
head(ces_selected)
```

```
## # A tibble: 6 x 31
##   pes21_votechoice2021                    cps21_votechoice pes21_votechoice2021~1
##   <dbl+lbl>                               <dbl+lbl>        <chr>
## 1  2 [Conservative Party]                 NA               "-99"
## 2  3 [ndp]                                 3 [ndp]         "-99"
```

```
## 3  9 [Don’t know / Prefer not to answer]  7 [Don’t know/~ "-99"
## 4  1 [Liberal Party]                      NA              "-99"
## 5 NA                                       3 [ndp]         ""
## 6  4 [Bloc Québécois]                     4 [Bloc Québéc~ "-99"
## # i abbreviated name: 1: pes21_votechoice2021_7_TEXT
## # i 28 more variables: cps21_votechoice_6_TEXT <chr>, pes21_province <dbl+lbl>,
## #   cps21_age <dbl>, pes21_follow_pol <dbl+lbl>, pes21_rural_urban <dbl+lbl>,
## #   pes21_inequal <dbl+lbl>, pes21_abort2 <dbl+lbl>, pes21_contact1 <dbl+lbl>,
## #   Region <chr>, cps21_marital <dbl+lbl>, cps21_imm_year <dbl+lbl>,
## #   cps21_bornin_canada <dbl+lbl>, cps21_rel_imp <dbl+lbl>,
## #   cps21_volunteer <dbl+lbl>, cps21_education <dbl+lbl>, ...
```

Creating disengagement and data quality flags

identify disengagement group To define political disengagement, the following survey items are used:

pes21_follow_pol And how closely do you follow politics on TV, radio, newspapers, or the Internet?

- Very closely (1)
- Fairly closely (2)
- Not very closely (3)
- Not at all (4)

cps21_interest_gen_1 How interested are you in politics generally? Set the slider to a number from 0 to 10, where 0 means no interest at all, and 10 means a great deal of interest.

cps21_interest_elxn_1 How interested are you in this federal election? Set the slider to a number from 0 to 10, where 0 means no interest at all, and 10 means a great deal of interest.

cps21_news_cons On average, how much time do you usually spend watching, reading, and listening to news each day?

- None (1)
- 1-10 minutes (2)
- 11-30 minutes (3)
- 31-60 minutes (4)
- Between 1 and 2 hours (5)
- More than 2 hours (6)
- Don’t know/ Prefer not to answer (7)

cps21_govt_confusing Sometimes, politics and government seem so complicated that a person like me can’t really understand what’s going on.

- Strongly disagree (1)
- Somewhat disagree (2)
- Somewhat agree (3)
- Strongly agree (4)
- Don’t know/ Prefer not to answer (5)

These variables are combined into a simple count (`disengaged_count`) to reflect the number of disengagement indicators present for each respondent.

```r
# Each component reflects low political interest, low media engagement, or confusion.
# Missing values (NA) are treated as disengaged (i.e., score = 1),
# since non-response may reflect a lack of political interest or attentiveness.

ces_selected$disengaged_count <- with(ces_selected,
  as.integer(is.na(pes21_follow_pol)      | pes21_follow_pol >= 3) +
  as.integer(is.na(cps21_interest_gen_1)  | cps21_interest_gen_1 <= 2) +
  as.integer(is.na(cps21_interest_elxn_1) | cps21_interest_elxn_1 <= 2) +
  as.integer(is.na(cps21_news_cons)       | cps21_news_cons %in% c(1, 7)) +
  as.integer(is.na(cps21_govt_confusing)  | cps21_govt_confusing == 5)
)

print("Distribution of disengaged_count:")
```

```
## [1] "Distribution of disengaged_count:"
```

```r
table(ces_selected$disengaged_count)
```

```
##
##    0    1    2    3    4    5
## 8750 9035 1574 1157  383   69
```

We define respondents with disengaged_count >= 3 as politically disengaged. This threshold reflects a combination of at least 3 disengagement indicators,and captures roughly 8% of the sample.

```r
disengaged_threshold <- 3
```

identify low quality group According to the original codebook, a number of severe data quality issues (e.g., incomplete responses, failed attention checks, straightlining) were already removed from the public dataset.

However, some respondents were flagged for less-severe issues and retained. These include: - Inattentive respondents (e.g., those taking unusually long to complete the survey) - Duplicate IP/demo matches - Initial duplicates - PES speeders (respondents who completed the post-election survey unusually fast)

We use the following variables to track these lower-level quality concerns: - `cps21_duplicates_pid_flag` - `cps21_duplicate_ip_demo_flag` - `cps21_inattentive` - `pes21_speeder_low_quality` - `pes21_duplicates_pid_flag` - `pes21_inattentive`

To simplify later filtering or robustness checks, we create a `low_quality_count` variable to count how many of these flags are triggered per respondent.

```r
# Compute a low_quality_count score to summarize how many data quality flags each respondent triggered.
# Each variable is binary (0 = no issue, 1 = issue).
# Missing values (NA) are treated as 0 (i.e., no issue), to avoid excluding respondents.

ces_selected$low_quality_count <- rowSums(ces_selected[check_low_quality], na.rm = TRUE)

print("Distribution of low_quality_count:")
```

```
## [1] "Distribution of low_quality_count:"
```

```
table(ces_selected$low_quality_count)
```

```
##
##     0     1     2     3
## 17847  2808   310     3
```

Based on this distribution, we define low_quality_count >= 3 as low quality.

Note: This is the first round of cleaning. A second round of filtering based on survey duration (e.g., too fast or too slow) will be applied later.

```
low_quality_threshold <- 3
```

remove unnecessary variables We convert labelled variables to readable factor levels using `as_factor()`, making the data easier to interpret and use in further exploring.

```
# convert to readable entry
ces_selected_converted <- ces_selected %>% mutate(across(where(is.labelled), as_factor))

# remove the variables for checking quality
ces_feature <- ces_selected_converted %>%
  select(all_of(c(vote_choices, feature_vars)), disengaged_count, low_quality_count)

head(ces_feature)
```

```
## # A tibble: 6 x 23
##   pes21_votechoice2021             cps21_votechoice      pes21_votechoice2021~1
##   <fct>                            <fct>                 <chr>
## 1 Conservative Party               <NA>                  "-99"
## 2 ndp                              ndp                   "-99"
## 3 Don't know / Prefer not to answer Don't know/ Prefer n~ "-99"
## 4 Liberal Party                    <NA>                  "-99"
## 5 <NA>                             ndp                   ""
## 6 Bloc Québécois                   Bloc Québécois        "-99"
## # i abbreviated name: 1: pes21_votechoice2021_7_TEXT
## # i 20 more variables: cps21_votechoice_6_TEXT <chr>, pes21_province <fct>,
## #   cps21_age <dbl>, pes21_follow_pol <fct>, pes21_rural_urban <fct>,
## #   pes21_inequal <fct>, pes21_abort2 <fct>, pes21_contact1 <fct>,
## #   Region <chr>, cps21_marital <fct>, cps21_imm_year <fct>,
## #   cps21_bornin_canada <fct>, cps21_rel_imp <fct>, cps21_volunteer <fct>,
## #   cps21_education <fct>, pes21_lived <fct>, cps21_fed_gov_sat <fct>, ...
```

Variables used strictly for quality checks (e.g., duplicate flags) are removed from the main feature set, but `disengaged_count` and the raw quality flags are retained for possible use in filtering or exploratory analysis.

Data Cleaning - Set Target Variable

The target variable was created by merging two columns:

pes21_votechoice2021 Which party did you vote for?

- Liberal Party (1)
- Conservative Party (2)
- NDP (3) (Display This Choice: If In which province or territory are you currently living? = Quebec)
- Bloc Québécois (4)
- Green Party (5)
- People's Party (6)
- Another party (specify) (7): pes21_votechoice2021_7_TEXT
- I spoiled my vote (8)
- Don't know / Prefer not to answer (9)

cps21_votechoice Which party do you think you will vote for?

- Liberal Party (1)
- Conservative Party (2)
- NDP (3) (Display This Choice: If In which province or territory are you currently living? = Quebec)
- Bloc Québécois (4)
- Green Party (5)
- Another party (please specify) (6): cps21_votechoice_6_TEXT
- Don't know/ Prefer not to answer (7)

The idea was to prioritize *pes21_votechoice2021* when it was not missing (NA), and if it was missing, use the value from *cps21_votechoice.*

```
ces_feature <- ces_feature %>%
  mutate(
    votechoice = case_when(
      !is.na(pes21_votechoice2021) ~ as.character(pes21_votechoice2021),
      !is.na(cps21_votechoice) ~ as.character(cps21_votechoice),
      TRUE ~ NA_character_
    ),
    vote_source = case_when(
      !is.na(pes21_votechoice2021) ~ "pes",
      !is.na(cps21_votechoice) ~ "cps",
      TRUE ~ NA_character_
    )
  )

unique(ces_feature$votechoice)
```

```
##  [1] "Conservative Party"                "ndp"
##  [3] "Don't know / Prefer not to answer" "Liberal Party"
##  [5] "Bloc Québécois"                    NA
##  [7] "Don't know/ Prefer not to answer"  "People's Party"
##  [9] "Green Party"                       "Another party (specify)"
## [11] "Another party (please specify)"    "I spoiled my vote"
```

We identified there were issues with inconsistent formatting in the data (such as inconsistent capitalization and spacing). For example, there were variations like "Don't know/ Prefer not to answer" and "Don't know / Prefer not to answer".

To ensure consistent formatting, we converted all variations of text to a consistent format, and replaced any inconsistent spaces and symbols

```r
# Standardizing Formatting
ces_feature$votechoice <- recode(ces_feature$votechoice,
                                 "ndp" = "NDP",
                                 "Don't know/ Prefer not to answer" = "Don't know / Prefer not to answe
                                 "Another party (please specify)" = "Another party (specify)",
                                 .default = ces_feature$votechoice)

#ces_feature$votechoice <- factor(ces_feature$votechoice)
table(ces_feature$votechoice, useNA = "ifany")
```

```
##
##          Another party (specify)                    Bloc Québécois
##                              275                              1899
##               Conservative Party Don't know / Prefer not to answer
##                             4740                              1357
##                      Green Party                  I spoiled my vote
##                              401                                71
##                    Liberal Party                               NDP
##                             5415                              3643
##                   People's Party                              <NA>
##                              417                              2750
```

Handling "Another party (specify)" Responses In the raw dataset, vote choices were coded into predefined categories (e.g., Liberal, Conservative, NDP, etc.).
However, some respondents selected **"Another party (specify)"**, which allowed them to write in a custom party name.

Upon inspection, we found **275** such responses, stored in the free-text fields: - `pes21_votechoice2021_7_TEXT` (post-election survey) - `cps21_votechoice_6_TEXT` (pre-election survey)

These responses were initially assigned the label `"Another party"`, making them uninformative for modeling.

While small in number, these write-in responses contain meaningful data: - Some match known parties like **People's Party**, **Bloc Québécois**, or **Green Party** - Others reflect **independent candidates** or **small/obscure parties** - A portion includes **spoiled ballots**, **protest votes**, or **intentional ambiguity** (e.g., "None of your business")

Without processing, they would either be dropped (as NA) or lumped into a generic "Other" class.

```r
map_another_to_main_parties <- function(text) {
  text <- tolower(trimws(text))

  case_when(
    # Spoiled or protest vote
    grepl("spoil|annul|cancel|decline|private|blank|secret ballot|none of your business|don't vote", te

    # People's Party
    grepl("ppc|people.?s party|parti populaire|popular party", text) ~ "People's Party",

    # Bloc Québécois
    grepl("bloc", text) ~ "Bloc Québécois",

    # Green Party
    grepl("green", text) |
```

```
      grepl("protection des animaux|animal protection", text) ~ "Green Party",

    # Liberal
    grepl("liberal", text) ~ "Liberal Party",

    # Conservative
    grepl("conservative|pcc|pcp|cpp", text) ~ "Conservative Party",

    # NDP
    grepl("ndp|new democratic|npd", text) ~ "NDP",

    # Independent, Maverick, Communist, etc.
    grepl("independent|maverick|rhinoceros|communist|libertarian|parti libre|christian heritage|chp", t

    # Uncertain or undecided
    grepl("undecid|indécis|i don't know|incertain|not sure|je vais probablement", text) ~ "Don't know /

    # Anything else
    TRUE ~ "Another party"
  )
}
```

To recover this information, we implemented a custom mapping function to classify free-text responses into 9 categories:

1. Liberal Party
2. Conservative Party
3. NDP
4. Green Party
5. Bloc Québécois
6. People's Party
7. Another party (small or independent parties)
8. Spoiled / Protest vote
9. Don't know / Prefer not to answer

This mapping was applied **after** merging vote choices from both surveys and **after** standardizing vote labels.

```
# Merged another party choice
ces_feature <- ces_feature %>%
  mutate(
    another_text = case_when(
      votechoice == "Another party (specify)" & vote_source == "pes" ~ pes21_votechoice2021_7_TEXT,
      votechoice == "Another party (specify)" & vote_source == "cps" ~ cps21_votechoice_6_TEXT,
      TRUE ~ NA_character_
    )
```

```r
  )

# replace votechoice
ces_feature <- ces_feature %>%
  mutate(
    votechoice = if_else(
      votechoice == "Another party (specify)" ,
      map_another_to_main_parties(another_text),
      votechoice
    )
  )
table(ces_feature$votechoice, useNA = "ifany")
```

```
##
##                    Another party                   Bloc Québécois
##                               91                             1915
##               Conservative Party Don't know / Prefer not to answer
##                             4745                             1362
##                      Green Party                I spoiled my vote
##                              405                               89
##                    Liberal Party                              NDP
##                             5417                             3646
##                    People's Party                            <NA>
##                              548                             2750
```

```r
#unmapped_rows <- ces_feature %>%   filter(votechoice == "Another party")
#unique(unmapped_rows$another_text)
```

After applying the mapping, some records moved from `"Another party"` to more specific classes.

Data Cleaning - Check feature variables

Before modeling, we examine the distribution and potential correlations of these features to decide whether they should be included in the model.

Here is the code used to inspect the unique data entries for each selected feature. By reviewing these values, we can identify issues like missing data, inconsistent formatting,or unexpected categories — which indicates that data cleaning is needed before analysis.

```r
# return unique entry for each feature
get_feature_levels <- function(data, column_name) {
  if (!column_name %in% names(data)) {
    stop("Column not found in dataset.")
  }
  unique_values <- unique(data[[column_name]])
  return(unique_values)
}

for (var in feature_vars) {
  # Skip only "Duration__in_seconds_"
  if (var == "Duration__in_seconds_") next
```

```
  cat("\n---", var, "---\n")
  print(get_feature_levels(ces_feature, var))
}
```

```
##
## --- pes21_province ---
##  [1] Quebec                    British Columbia
##  [3] Ontario                   <NA>
##  [5] Alberta                   Newfoundland and Labrador
##  [7] Saskatchewan              Manitoba
##  [9] New Brunswick             Yukon
## [11] Nova Scotia               Northwest Territories
## [13] Prince Edward Island      Nunavut
## 13 Levels: Alberta British Columbia Manitoba ... Yukon
##
## --- cps21_age ---
##  [1] 57 22 28 29 41 63 52 66 42 92 33 48 65 54 68 44 31 38 45 58 64 77 36 62 78
## [26] 72 81 24 46 60 40 59 56 25 49 30 69 53 26 34 43 76 75 80 27 47 35 82 32 73
## [51] 61 18 79 67 70 21 50 37 88 19 39 55 51 74 23 20 85 83 71 90 84 86 89 87 91
## [76] 95 93 96 97
##
## --- pes21_follow_pol ---
## [1] Fairly closely                   Not very closely
## [3] Not at all                       <NA>
## [5] Very closely                     Don't know/ Prefer not to answer
## 5 Levels: Very closely Fairly closely Not very closely ... Don't know/ Prefer not to answer
##
## --- pes21_rural_urban ---
## [1] A small town (more than 1000 people but less than 15K)
## [2] A suburb of a large town or city
## [3] A large town or city (more than 50K people)
## [4] <NA>
## [5] A rural area or village (less than1000 people)
## [6] Don't know / Prefer not to answer
## [7] A middle-sized town (15K-50K people) not attached to a city
## 6 Levels: A rural area or village (less than1000 people) ...
##
## --- pes21_inequal ---
## [1] Probably yes                     Definitely yes
## [3] <NA>                             Definitely not
## [5] Probably not                     Not sure
## [7] Don't know/ Prefer not to answer
## 6 Levels: Definitely yes Probably yes Not sure Probably not ... Don't know/ Prefer not to answer
##
## --- pes21_abort2 ---
## [1] No                               <NA>
## [3] Yes                              In some circumstances
## [5] Don't know/ Prefer not to answer
## Levels: Yes In some circumstances No Don't know/ Prefer not to answer
##
## --- pes21_contact1 ---
## [1] No                               <NA>
## [3] Yes                              Don't know/ Prefer not to answer
```

```
## Levels: Yes No Don't know/ Prefer not to answer
##
## --- Region ---
## [1] "Quebec"      "West"        "Ontario"     "Atlantic"    "Territories"
##
## --- cps21_marital ---
## [1] Separated                          Never Married
## [3] Married                            Divorced
## [5] Living with a partner              Widowed
## [7] Don't know/ Prefer not to answer
## 7 Levels: Married Living with a partner Divorced Separated ... Don't know/ Prefer not to answer
##
## --- cps21_imm_year ---
##  [1] <NA>                             2001
##  [3] 1965                             1969
##  [5] 2011                             1966
##  [7] 1962                             2003
##  [9] 1996                             2009
## [11] 1957                             2017
## [13] 2012                             1982
## [15] 1994                             1968
## [17] 1964                             1946
## [19] 2013                             1974
## [21] 1963                             1970
## [23] 1999                             2008
## [25] 1988                             1997
## [27] 2010                             2007
## [29] 2005                             1950
## [31] 1992                             1971
## [33] 1979                             1960
## [35] 2019                             1995
## [37] 2002                             1948
## [39] 1984                             1998
## [41] 1973                             1987
## [43] 1976                             1991
## [45] 1981                             2014
## [47] 1967                             2000
## [49] 2016                             1978
## [51] 1975                             1972
## [53] 1990                             2004
## [55] 1989                             2006
## [57] 2015                             2018
## [59] 1955                             1986
## [61] 1983                             1980
## [63] 1977                             1993
## [65] 1985                             1953
## [67] 1952                             1949
## [69] 1959                             1951
## [71] 1961                             1956
## [73] 1958                             2020
## [75] Don't know/ Prefer not to answer 2021
## [77] 1954                             1947
## [79] 1938                             1945
## [81] 1920                             1934
```

```
## [83] 1942                             1931
## 103 Levels: 1920 1921 1922 1923 1924 1925 1926 1927 1928 1929 1930 1931 ... Don't know/ Prefer not t
##
## --- cps21_bornin_canada ---
## [1] Yes                              No
## [3] Don't know/ Prefer not to say
## Levels: Yes No Don't know/ Prefer not to say
##
## --- cps21_rel_imp ---
## [1] Not important at all              Somewhat important
## [3] <NA>                              Very important
## [5] Not very important                Don't know/ Prefer not to answer
## 5 Levels: Very important Somewhat important ... Don't know/ Prefer not to answer
##
## --- cps21_volunteer ---
## [1] Never                             A few times
## [3] More than five times              Don't know/ Prefer not to answer
## [5] Just once
## 5 Levels: Never Just once A few times ... Don't know/ Prefer not to answer
##
## --- cps21_education ---
##  [1] Some technical, community college, CEGEP, College Classique
##  [2] Some university
##  [3] Bachelor's degree
##  [4] Master's degree
##  [5] Completed technical, community college, CEGEP, College Classique
##  [6] Professional degree or doctorate
##  [7] Completed secondary/ high school
##  [8] Some elementary school
##  [9] Some secondary/ high school
## [10] Completed elementary school
## [11] Don't know/ Prefer not to answer
## [12] No schooling
## 12 Levels: No schooling Some elementary school ... Don't know/ Prefer not to answer
##
## --- pes21_lived ---
## [1] More than 10 years                Less than 1 year
## [3] <NA>                              3-10 years
## [5] 1-3 years                         Don't know/ Prefer not to answer
## 5 Levels: Less than 1 year 1-3 years 3-10 years ... Don't know/ Prefer not to answer
##
## --- cps21_fed_gov_sat ---
## [1] Not at all satisfied              Fairly satisfied
## [3] Not very satisfied                Very satisfied
## [5] Don't know/ Prefer not to answer
## 5 Levels: Very satisfied Fairly satisfied ... Don't know/ Prefer not to answer
```

We are beginning with handling ambiguous responses such as NA and "don't know". In parallel, we aim to identify patterns of political apathy, which may be reflected through missing values, neutral responses, or lack of engagement.

```r
replace_dontknow_with_na <- function(col) {
  if (is.character(col) || is.factor(col)) {
    col <- as.character(col)
```

```r
    col[grepl("don.?t\\s*know|prefer not to answer", col, ignore.case = TRUE)] <- NA
    return(as.factor(col))
  } else {
    return(col)
  }
}

# Apply the function to all feature columns
ces_feature <- ces_feature %>%
  mutate(across(all_of(feature_vars), replace_dontknow_with_na))
```

Handling disengaged data

In this step, we identify and remove politically disengaged respondents and those with unclear or missing vote intentions.

```r
# Since We define respondents with disengaged_count >= 3 as politically disengaged
# Separate disengaged respondents based on disengaged_threshold

disengaged_responses <- ces_feature %>%
  filter(disengaged_count >= disengaged_threshold)

ces_feature_cleaned <- ces_feature %>%
  filter(disengaged_count < disengaged_threshold)

# Filter respondents with invalid vote choices (e.g., "Don't know" or "Spoiled vote")
invalid_vote_choices <- c("Don't know / Prefer not to answer", "I spoiled my vote")
invalid_vote_responses <- ces_feature_cleaned %>%
  filter(is.na(votechoice) | votechoice %in% invalid_vote_choices)

ces_feature_cleaned <- ces_feature_cleaned %>%
  filter(!(is.na(votechoice) | votechoice %in% invalid_vote_choices))

# Combine all disengaged rows
disengaged_group <- bind_rows(disengaged_responses, invalid_vote_responses)
```

```r
ces_feature_cleaned$votechoice <- factor(ces_feature_cleaned$votechoice)
disengaged_group$votechoice <- factor(disengaged_group$votechoice) # may not work
```

Initially, votechoice was converted to a character type, which led to issues when inspecting the data later. The values in the column became numeric codes instead of the intended text labels (e.g., "1" for "Liberal Party"). This happened because when a factor column is converted to a character type, the factor levels are lost and replaced by numeric codes.

Since we plan to use votechoice as the target variable in predictive modeling, it is essential to revert it back to factor type. This is because many machine learning algorithms (e.g., logistic regression, random forest) require categorical variables to be factors, as they help the model interpret the categories correctly.

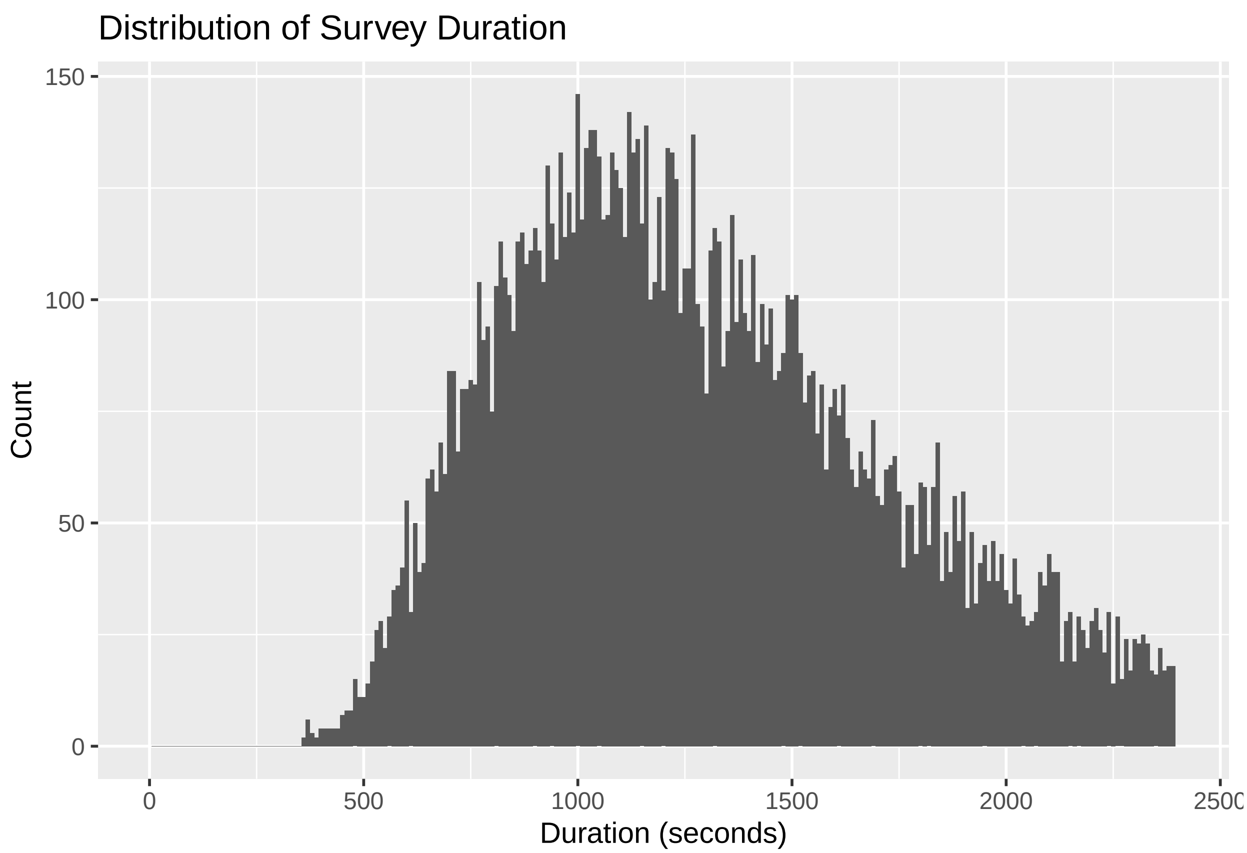
Handling low-quality data

Based on survey duration time, we identified some responses as unreliable. These cases are also labeled as politically disengaged. Since they may bias the model, we temporarily remove them from the dataset before modeling.

```r
library(ggplot2)
ggplot(ces_feature_cleaned, aes(x = Duration__in_seconds_)) +
  geom_histogram(binwidth = 10) +
  xlim(0, 2400) +
  labs(title = "Distribution of Survey Duration",
       x = "Duration (seconds)", y = "Count")
```

```
## Warning: Removed 2398 rows containing non-finite outside the scale range
## (`stat_bin()`).
```

```
## Warning: Removed 2 rows containing missing values or values outside the scale range
## (`geom_bar()`).
```




```r
summary(ces_feature$Duration__in_seconds_)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##     362     995    1325    8710    1875 1575155
```

The summary statistics of the `Duration` variable are as follows:

- **Minimum**: 362 seconds (~6 minutes)

- **1st Quartile (Q1)**: 995 seconds (~16.6 minutes)
- **Median**: 1325 seconds (~22 minutes)
- **Mean**: 8710 seconds (significantly inflated by outliers)
- **3rd Quartile (Q3)**: 1875 seconds (~31.3 minutes)
- **Maximum**: 1,575,155 seconds (> 400 hours)

According to the summary, These values suggest that while most respondents completed the survey in under 30 minutes, there are a few extreme outliers with excessively long durations that strongly distort the mean.

The **minimum value of 362 seconds** and the **Q1 value of 995 seconds** suggest that any respondent completing the survey in under 10 minutes may not have engaged meaningfully with the content. Similarly, values above 48 hour are highly suspicious and may indicate participants who were inactive for long periods.

Therefore, a threshold of **600 seconds (10 minutes)** was chosen to identify "too fast" respondents, while an upper cap of **172800 seconds (48 hour)** was applied to identify "too slow" responses.

```r
filter_by_duration <- function(data, duration_col = "Duration__in_seconds_",
                               fast_threshold = 600, slow_threshold = 172800) {
  data %>%
    filter(
      .data[[duration_col]] >= fast_threshold,
      .data[[duration_col]] <= slow_threshold
    )
}

ces_feature_cleaned <- filter_by_duration(ces_feature_cleaned)
```

Then Remove responses considered low quality: We exclude any row where 'low_quality_count' is greater than low_quality_threshold This helps reduce noise from unreliable responses (e.g., inconsistent answers or other quality issues).

```r
ces_feature_cleaned <- ces_feature_cleaned %>% filter(low_quality_count <= low_quality_threshold)
```

#===

Pre-step for Correlation Test

Since these features are of different types and most of them are non-numeric. We cannot apply a single, unified statistical method. Instead, we need to adopt different analysis strategies based on the nature of each variable.

Next, we divide the selected features into two groups based on their data types: - Categorical features will be tested using Cramér's V - Ordinal or continuous features will be tested using the Kruskal-Wallis test This helps us evaluate the strength of correlation between each feature and the target variable.

```r
# Initialize lists for variable classification
list_chi <- c()
list_kruskal <- c()
```

```r
# Target variable (e.g. party vote choice)
#target_var<-"votechoice"
#target_var<-"pes21_votechoice2021"
#target <- ces_feature_cleaned[[target_var]]


# Target variable (e.g. party vote choice)
target_var <- "votechoice"
target <- ces_feature_cleaned[[target_var]]
#target <- ces_feature_cleaned[["votechoice"]]
is_target_cat <- is.factor(target) || is.character(target)

# Loop through feature variables
if (is_target_cat) {
  for (var in feature_vars) {
    x <- ces_feature_cleaned[[var]]

    if (is.factor(x) || is.character(x)) {
      list_chi <- c(list_chi, var)
    } else if (is.numeric(x) || is.ordered(x)) {
      list_kruskal <- c(list_kruskal, var)
    }
  }
}

# Print results
cat("Variables for Cramér's V (categorical):\n")
```

```
## Variables for Cramér’s V (categorical):
```

```r
print(list_chi)
```

```
##  [1] "pes21_province"      "pes21_follow_pol"    "pes21_rural_urban"
##  [4] "pes21_inequal"       "pes21_abort2"        "pes21_contact1"
##  [7] "Region"              "cps21_marital"       "cps21_imm_year"
## [10] "cps21_bornin_canada" "cps21_rel_imp"       "cps21_volunteer"
## [13] "cps21_education"     "pes21_lived"         "cps21_fed_gov_sat"
```

```r
cat("\nVariables for Kruskal-Wallis (numeric or ordered):\n")
```

```
##
## Variables for Kruskal-Wallis (numeric or ordered):
```

```r
print(list_kruskal)
```

```
## [1] "cps21_age"              "Duration__in_seconds_"
```

Correlation Test

For the features in list_chi, we compute their correlation with the target variable (party vote choice) using Cramér’s V. The calculation uses the cramerV() function from the rcompanion package, which automatically removes observations with missing values (NA).

```r
library(rcompanion)  # cramerV
cramer_results <- data.frame(Variable = character(),
                             CramersV = numeric(),
                             stringsAsFactors = FALSE)

for (var in list_chi) {
  tbl <- table(ces_feature_cleaned[[target_var]], ces_feature_cleaned[[var]])
  if (min(dim(tbl)) > 1) {
    result <- cramerV(tbl, bias.correct = TRUE)
    cramer_results <- rbind(cramer_results, data.frame(Variable = var, CramersV = result))
  }
}

# print result
cramer_results <- cramer_results[order(-cramer_results$CramersV), ]
print(cramer_results, row.names = FALSE)
```

```
##             Variable CramersV
##    cps21_fed_gov_sat  0.45030
##               Region  0.28780
##       pes21_province  0.24490
##        pes21_inequal  0.20110
##         pes21_abort2  0.19670
##   cps21_bornin_canada  0.14350
##        cps21_rel_imp  0.13300
##        cps21_marital  0.10810
##    pes21_rural_urban  0.07643
##     pes21_follow_pol  0.06152
##      cps21_education  0.05568
##          pes21_lived  0.05517
##       pes21_contact1  0.04160
##      cps21_volunteer  0.03118
##       cps21_imm_year      NaN
```

Higher Cramér's V values indicate stronger associations with the target variable. Variables such as Region and Province showed relatively strong correlations with vote choice, while others like Volunteer activity and Immigration year had weaker or missing correlations.

During the Cramér's V analysis, we found that 'cps21_imm_year' returned NaN. It might because the 'cps21_imm_year' variable has many unique values (immigration years). To address this, we converted 'cps21_imm_year' into 'years since immigration' by subtracting it from 2021. This transformed variable is numeric and can be meaningfully analyzed using the Kruskal–Wallis test.

```r
# convert new variable
ces_feature_cleaned$imm_duration <- 2021 - as.numeric(ces_feature_cleaned$cps21_imm_year)

# add to list_kruskal
list_kruskal <- c(list_kruskal, "imm_duration")
```

Then we applied the Kruskal–Wallis test to evaluate whether the distributions of features in list_kruskal differ significantly across vote choice categories.

```r
kruskal_results <- data.frame(Variable = character(),
                              KruskalP = numeric(),
                              stringsAsFactors = FALSE)

for (var in list_kruskal) {
  df <- na.omit(ces_feature_cleaned[, c(var, target_var)])
  formula <- as.formula(paste(var, "~", target_var))
  result <- kruskal.test(formula, data = df)

  kruskal_results <- rbind(kruskal_results,
                           data.frame(Variable = var,
                                      KruskalP = result$p.value))
}

# print result
kruskal_results <- kruskal_results[order(kruskal_results$KruskalP), ]
print(kruskal_results)
```

```
##                Variable      KruskalP
## 1             cps21_age 5.886393e-187
## 2 Duration__in_seconds_  8.153069e-49
## 3          imm_duration  3.900711e-06
```

Since small p-values indicate strong evidence of differences between groups. The features 'cps21_age', 'Duration_*in*_*seconds*', and 'imm_duration' all represented that these features are highly associated with voting behavior and may be valuable for predictive modeling.

We will use the following features in the prediction model:

```r
# filter variables with Cramér's V > 0.1
selected_cramer_vars <- cramer_results %>%
  filter(CramersV > 0.1) %>%
  pull(Variable)

# filter variables with kruskal < 0.05
selected_kruskal_vars <- kruskal_results %>%
  filter(KruskalP < 0.05) %>%
  pull(Variable)

selected_model_vars <- unique(c(selected_cramer_vars, selected_kruskal_vars))
print(selected_model_vars)
```

```
##  [1] "cps21_fed_gov_sat"     "Region"                "pes21_province"
##  [4] "pes21_inequal"         "pes21_abort2"          "cps21_bornin_canada"
##  [7] "cps21_rel_imp"         "cps21_marital"         "cps21_age"
## [10] "Duration__in_seconds_" "imm_duration"
```

Checking Feature Redundancy

To prevent multicollinearity in the model, we calculated pairwise Cramér's V scores among features to identify strongly correlated variables.

Interpretation thresholds: • V > 0.6 — High correlation: likely redundant; consider removing one of the variables. • V > 0.4 — Moderate correlation: possible redundancy; proceed with caution. • V < 0.3 — Low correlation: safe to include both variables.

```r
library(rcompanion)

feature_corr_results <- data.frame(VarA = character(),
                                   VarB = character(),
                                   CramersV = numeric(),
                                   stringsAsFactors = FALSE)

for (i in 1:(length(selected_cramer_vars)-1)) {
  for (j in (i+1):length(selected_cramer_vars)) {
    varA <- selected_cramer_vars[i]
    varB <- selected_cramer_vars[j]

    clean_data <- ces_feature_cleaned %>%
      dplyr::select(all_of(c(varA, varB))) %>%
      dplyr::filter(!is.na(.data[[varA]]), !is.na(.data[[varB]]))

    tbl <- table(clean_data[[varA]], clean_data[[varB]])

    if (min(dim(tbl)) > 1) {
      result <- cramerV(tbl, bias.correct = TRUE)
      feature_corr_results <- rbind(feature_corr_results,
                                    data.frame(VarA = varA, VarB = varB, CramersV = result))
    }
  }
}

# print out
feature_corr_results <- feature_corr_results %>%
  mutate(Explanation = case_when(
    CramersV > 0.6 ~ "High correlation - consider removing one variable",
    CramersV > 0.4 ~ "Moderate correlation - possible redundancy",
    TRUE ~ "Low correlation - likely safe to include both"
  ))

feature_corr_results <- feature_corr_results[order(-feature_corr_results$CramersV), ]
print(feature_corr_results)
```

```
##                             VarA               VarB CramersV
## Cramer V7                 Region     pes21_province 0.999700
## Cramer V23          pes21_abort2      cps21_rel_imp 0.326300
## Cramer V15        pes21_province cps21_bornin_canada 0.165000
## Cramer V2      cps21_fed_gov_sat      pes21_inequal 0.164300
## Cramer V25 cps21_bornin_canada      cps21_rel_imp 0.160200
## Cramer V11                Region      cps21_rel_imp 0.160000
## Cramer V16        pes21_province      cps21_rel_imp 0.156900
## Cramer V10                Region cps21_bornin_canada 0.156100
## Cramer V12                Region      cps21_marital 0.124700
## Cramer V3      cps21_fed_gov_sat       pes21_abort2 0.121000
## Cramer V18         pes21_inequal       pes21_abort2 0.118700
## Cramer V17        pes21_province      cps21_marital 0.116000
```

```
## Cramer V22        pes21_abort2 cps21_bornin_canada 0.112800
## Cramer V26 cps21_bornin_canada       cps21_marital 0.111000
## Cramer V27       cps21_rel_imp       cps21_marital 0.110900
## Cramer V14      pes21_province        pes21_abort2 0.110600
## Cramer V1    cps21_fed_gov_sat      pes21_province 0.110200
## Cramer V9               Region        pes21_abort2 0.102800
## Cramer V24        pes21_abort2       cps21_marital 0.097390
## Cramer V    cps21_fed_gov_sat              Region 0.091060
## Cramer V5    cps21_fed_gov_sat       cps21_rel_imp 0.076400
## Cramer V4    cps21_fed_gov_sat cps21_bornin_canada 0.076200
## Cramer V21       pes21_inequal       cps21_marital 0.067860
## Cramer V6    cps21_fed_gov_sat       cps21_marital 0.067760
## Cramer V13      pes21_province       pes21_inequal 0.065030
## Cramer V8               Region       pes21_inequal 0.050350
## Cramer V20       pes21_inequal       cps21_rel_imp 0.034970
## Cramer V19       pes21_inequal cps21_bornin_canada 0.009949
##                                                 Explanation
## Cramer V7  High correlation - consider removing one variable
## Cramer V23     Low correlation - likely safe to include both
## Cramer V15     Low correlation - likely safe to include both
## Cramer V2      Low correlation - likely safe to include both
## Cramer V25     Low correlation - likely safe to include both
## Cramer V11     Low correlation - likely safe to include both
## Cramer V16     Low correlation - likely safe to include both
## Cramer V10     Low correlation - likely safe to include both
## Cramer V12     Low correlation - likely safe to include both
## Cramer V3      Low correlation - likely safe to include both
## Cramer V18     Low correlation - likely safe to include both
## Cramer V17     Low correlation - likely safe to include both
## Cramer V22     Low correlation - likely safe to include both
## Cramer V26     Low correlation - likely safe to include both
## Cramer V27     Low correlation - likely safe to include both
## Cramer V14     Low correlation - likely safe to include both
## Cramer V1      Low correlation - likely safe to include both
## Cramer V9      Low correlation - likely safe to include both
## Cramer V24     Low correlation - likely safe to include both
## Cramer V       Low correlation - likely safe to include both
## Cramer V5      Low correlation - likely safe to include both
## Cramer V4      Low correlation - likely safe to include both
## Cramer V21     Low correlation - likely safe to include both
## Cramer V6      Low correlation - likely safe to include both
## Cramer V13     Low correlation - likely safe to include both
## Cramer V8      Low correlation - likely safe to include both
## Cramer V20     Low correlation - likely safe to include both
## Cramer V19     Low correlation - likely safe to include both
```

Among all feature pairs, only `Region` and `pes21_province` showed a high Cramér's V (0.9997), indicating near-perfect redundancy. Since both represent geographic information. We will retain only one of them to avoid duplication. In this case, we choose to keep `pes21_province`.

Result

Update the list of features in the prediction model:

```r
selected_cramer_vars <- setdiff(selected_cramer_vars, "Region")
selected_model_vars <- unique(c(selected_cramer_vars, selected_kruskal_vars))
print(selected_model_vars)
```

```
##  [1] "cps21_fed_gov_sat"     "pes21_province"        "pes21_inequal"
##  [4] "pes21_abort2"          "cps21_bornin_canada"   "cps21_rel_imp"
##  [7] "cps21_marital"         "cps21_age"             "Duration__in_seconds_"
## [10] "imm_duration"
```

Export Data for Modeling

```r
ces_Modeling <- ces_feature_cleaned %>% select(all_of(selected_model_vars), target_var)
```

```
## Warning: Using an external vector in selections was deprecated in tidyselect 1.1.0.
## i Please use `all_of()` or `any_of()` instead.
##   # Was:
##   data %>% select(target_var)
##
##   # Now:
##   data %>% select(all_of(target_var))
##
## See <https://tidyselect.r-lib.org/reference/faq-external-vector.html>.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```

```r
save(ces_Modeling, disengaged_group, selected_model_vars, file = "preprocessed_data.RData")
```