Is Betting better than Polls at Predicting Elections?

Claire Yang, Zoe Wu, Fatema Abdulla, Martin Bigil-Rico

November 2024

Question 3: Evaluating Data Quality

This question encourages you to begin the exploratory data analysis (EDA) for your final project. By addressing potential data quality issues early, you can identify and rectify problems promptly. For each important variable in your dataset, assess its quality by creating a table that includes the following:

- **Continuous variables**:
  - The number of non-missing observations.
  - The number of missing observations.
  - Measures of central tendency (e.g., mean, median).
  - Measures of variability (e.g., standard deviation, interquartile range [IQR]).
- **Categorical variables**:
  - The levels of the variable.
  - For each level:
    - The number of non-missing observations.
    - The number of missing observations.

Answer: Comparing Predictive Performance of Betting Markets and Polls

Our project aims to compare the predictive accuracy of betting markets and polls in forecasting the outcome of the 2024 U.S. presidential election by state. We use the following datasets:

1. **Betting Market Data**
   - Sourced from a Kaggle dataset, this dataset scrapes Polymarket's 2024 election results by state and consolidates them into a CSV file (source).
2. **Poll Data**
   - Sourced from FiveThirtyEight, this dataset provides polling averages and raw data from the 2024 U.S. presidential election (source).
3. **Actual Election Results**
   - Sourced from CBS News, this dataset reports the official 2024 presidential election results (source).

Exploratory Data Analysis (EDA)   Dataset 1: Betting Market Data

The dataset provides separate files for each state, containing the probability of a Republican win under the column "Donald Trump" and a Democratic win under "Kamala Harris." These probabilities are based on the amount bet on Polymarket for each candidate. Data was aggregated by month, resulting in a final dataset of state-level probabilities for Republican and Democratic wins from April 17, 2024, to November 4, 2024.

Columns Included "Date (UTC)","Timestamp (UTC)","Donald Trump","Kamala Harris", and "Other". All values of the table were continuous, enumerating the date, time, percentage probability for Trump, percentage probability for Kamala, and percentage probability for a different candidate. No cateogrical variables existed.

First non-missing and missing observations were quantified for each probability column. Mean probabilities and standard deviations for Republican and Democratic wins nationwide were calculated.

```r
install.packages("readr")
```

```
## Installing package into '/cloud/lib/x86_64-pc-linux-gnu-library/4.4'
## (as 'lib' is unspecified)
```

```r
install.packages("dplyr")
```

```
## Installing package into '/cloud/lib/x86_64-pc-linux-gnu-library/4.4'
## (as 'lib' is unspecified)
```

```r
install.packages("ggplot2")
```

```
## Installing package into '/cloud/lib/x86_64-pc-linux-gnu-library/4.4'
## (as 'lib' is unspecified)
```

```r
install.packages("caret")
```

```
## Installing package into '/cloud/lib/x86_64-pc-linux-gnu-library/4.4'
## (as 'lib' is unspecified)
```

```r
# Load necessary libraries
library(readr)
library(dplyr)
```

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```r
library(ggplot2)
library(caret)
```

```
## Loading required package: lattice
```

```r
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v forcats   1.0.0     v stringr   1.5.1
## v lubridate 1.9.3     v tibble    3.2.1
## v purrr     1.0.2     v tidyr     1.3.1

## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## x purrr::lift()   masks caret::lift()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become error
```

```r
#Access csv files in directory
file_path <- "data/betting_data/polymarket/csv_month/"

file_list <- list.files(path = file_path, pattern = "*.csv", full.names = TRUE)

# Data frames to store results
final_data <- data.frame()
```

```r
missing_data_summary <- data.frame()

for (file in file_list) {
  state_data <- read_csv(file, show_col_types = FALSE)

  # Extract the state abbreviation from the file name
  state_abbrev <- tools::file_path_sans_ext(basename(file)) %>%
    stringr::str_extract("^[A-Z]{2}")

  # Calculate averages for  Trump and Harris
  avg_trump <- mean(state_data$`Donald Trump`, na.rm = TRUE)
  avg_harris <- mean(state_data$`Kamala Harris`, na.rm = TRUE)

  # Count missing and non-missing observations for each candidate
  non_missing_trump <- sum(!is.na(state_data$`Donald Trump`))
  missing_trump <- sum(is.na(state_data$`Donald Trump`))
  non_missing_harris <- sum(!is.na(state_data$`Kamala Harris`))
  missing_harris <- sum(is.na(state_data$`Kamala Harris`))

  # Append missing data summary for this state
  missing_data_summary <- bind_rows(
    missing_data_summary,
    data.frame(
      state = state_abbrev,
      candidate = "Donald Trump",
      non_missing = non_missing_trump,
      missing = missing_trump
    ),
    data.frame(
      state = state_abbrev,
      candidate = "Kamala Harris",
      non_missing = non_missing_harris,
      missing = missing_harris
    )
  )

  # Create a new data structure for average percentages
  state_results <- data.frame(
    candidate = c("Donald Trump", "Kamala Harris"),
    percentage = c(avg_trump, avg_harris),
    state = c(state_abbrev, state_abbrev)
  )

  final_data <- bind_rows(final_data, state_results)
}

# Calculate nationwide statistics
nationwide_stats <- final_data %>%
  group_by(candidate) %>%
  summarise(
    nationwide_mean = mean(percentage, na.rm = TRUE),
    nationwide_sd = sd(percentage, na.rm = TRUE),
    total_non_missing = sum(!is.na(percentage)),
```

```r
    total_missing = sum(is.na(percentage))
  )

#print(missing_data_summary)

#print(nationwide_stats)
```

While we will be aggregating over time, we nevertheless ran a missing information test on data and timestamp.

```r
missing_data_summary <- data.frame()

for (file in file_list) {
  state_data <- read_csv(file, show_col_types = FALSE)

  # Extract the state abbreviation from the file name
  state_abbrev <- tools::file_path_sans_ext(basename(file)) %>%
    stringr::str_extract("^[A-Z]{2}")

  # Count missing and non-missing observations for each candidate
  non_missing_date <- sum(!is.na(state_data$`Date (UTC)`))
  missing_date <- sum(is.na(state_data$`Date (UTC)`))
  non_missing_timestamp <- sum(!is.na(state_data$`Timestamp (UTC)`))
  missing_timestamp <- sum(is.na(state_data$`Timestamp (UTC)`))

  # Append missing data summary for this state
  missing_data_summary <- bind_rows(
    missing_data_summary,
    data.frame(
      state = state_abbrev,
      variable = "Date",
      non_missing = non_missing_date,
      missing = missing_date
    ),
    data.frame(
      state = state_abbrev,
      variable = "Timestamp",
      non_missing = non_missing_timestamp,
      missing = missing_timestamp
    )
  )
}

#print(missing_data_summary)
```

We have no missing data in the csv files for each state, suggesting this dataset is good to go.

```r
install.packages("sf")
```

```
## Installing package into '/cloud/lib/x86_64-pc-linux-gnu-library/4.4'
## (as 'lib' is unspecified)
```

```r
install.packages("maps")
```

```
## Installing package into '/cloud/lib/x86_64-pc-linux-gnu-library/4.4'
## (as 'lib' is unspecified)
```

```r
install.packages("tigris")
```

```
## Installing package into '/cloud/lib/x86_64-pc-linux-gnu-library/4.4'
## (as 'lib' is unspecified)
```

```r
# Calculate the blue percentage (Kamala Harris's share of the total percentage)
single_percent <- final_data %>%
  filter(candidate %in% c("Kamala Harris", "Donald Trump")) %>% # Keep only relevant candidates
  pivot_wider(names_from = candidate, values_from = percentage) %>% # Reshape to wide format
  mutate(
    blue_percentage = `Kamala Harris` / (`Kamala Harris` + `Donald Trump`) # Calculate ratio
  ) %>%
  select(state, blue_percentage) # Keep only state and blue_percentage

# Print result
print(single_percent)
```

```
## # A tibble: 50 x 2
##    state blue_percentage
##    <chr>           <dbl>
##  1 AK             0.0698
##  2 AL             0.0314
##  3 AR             0.0391
##  4 AZ             0.383
##  5 CA             0.960
##  6 CO             0.934
##  7 CT             0.954
##  8 DE             0.954
##  9 FL             0.104
## 10 GA             0.376
## # i 40 more rows
```

```r
library(sf)
```

```
## Linking to GEOS 3.8.0, GDAL 3.0.4, PROJ 6.3.1; sf_use_s2() is TRUE
```

```r
library(dplyr)
library(ggplot2)

# Calculate blue_percentage as a percentage (0 to 100)
single_percent <- final_data %>%
  filter(candidate %in% c("Kamala Harris", "Donald Trump")) %>%
  pivot_wider(names_from = candidate, values_from = percentage) %>%
  mutate(
    blue_percentage = 100 * `Kamala Harris` / (`Kamala Harris` + `Donald Trump`)
  ) %>%
  select(state, blue_percentage)

# Download U.S. state geometries
us_states <- tigris::states(cb = TRUE, year = 2021) %>%
  filter(!STUSPS %in% c("PR", "VI", "GU", "MP", "AS")) %>% # Exclude territories
  select(state = STUSPS, geometry)
```

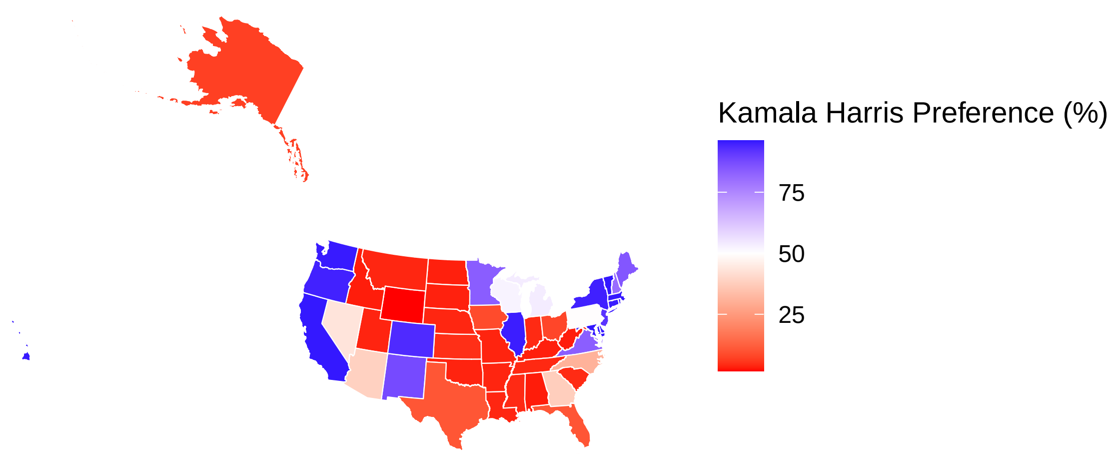
```
##   |                                                                              |
```

```r
# Merge state geometries with single_percent
map_data <- us_states %>%
  left_join(single_percent, by = "state")


# Plot the fixed map
ggplot(map_data) +
  geom_sf(aes(fill = blue_percentage), color = "white", lwd = 0.2) +
  scale_fill_gradient2(
    low = "red", mid = "white", high = "blue",
    midpoint = 50,
    name = "Kamala Harris Preference (%)"
  ) +
  labs(
    title = "U.S. Map of Preferences for Presidential Candidates",
    subtitle = "Based on Average Percentages",
    caption = "Data Source: Polymarket"
  ) +
  theme_minimal() +
  theme(
    axis.text = element_blank(),
    axis.ticks = element_blank(),
    panel.grid = element_blank()
  ) +
  coord_sf(crs = "+proj=aea +lat_1=29.5 +lat_2=45.5 +lat_0=37.5 +lon_0=-96") # Albers projection
```

U.S. Map of Preferences for Presidential Candidates

Based on Average Percentages



Data Source: Polymarket

Dataset 2: Poll Data

The dataset contains average polling data by candidate, date, and their adjusted percentage of being favored. We started out by exploring how many missing and non-missing data point there were for Harris and Trump. We then aggregated data over time, such that we are left with poll percentages for Harris/Trump per state. We take the mean to find the average poll percentage for Harris and Trump nationwide. We then take the standard deviation.

```r
polling_data <- read.csv("data/polls_data/538_data/polls_average_2024.csv")
```

```r
#isolate 2024 polling data (there are many rwos from 2020 in here with no values)
polling2024 <- polling_data %>%
  filter(cycle == 2024) %>%
  #take out the pct_trend_adjusted column (it's NA everywhere)
  select(- pct_trend_adjusted)
```

```r
#categorical variables - checking for missing data
# **Categorical variables**:
#     -   The levels of the variable.
#     -   For each level:
#         -   The number of non-missing observations.
#         -   The number of missing observations.

#cat variables - candidate, date, state, cycle, party

cat_vars <- c("candidate", "date", "state", "cycle", "party")

#function which calculates number of levels, and missing observations for each column

cat_eda <- function(data, vars) {

  levels <- numeric(length(cat_vars))
  nas <- numeric(length(cat_vars))
  non_nas <- numeric(length(cat_vars))

  #iterate through cat varaibles
  for (i in 1:5) {
    col <- vars[i]
    #update values appropriately
    levels[i] <- length(unique(data[[col]]))
    nas[i] <- sum(is.na(data[[col]]))
    non_nas[i] <- length(data[[col]]) - nas
  }

   #put results into table
   results <- data.frame(
    Variable = vars,
    Levels = levels,
    Missing = nas,
    NonMissing = non_nas)

   return(results)
}

cat_eda(polling2024, cat_vars)
```

```
## Warning in non_nas[i] <- length(data[[col]]) - nas: number of items to replace
## is not a multiple of replacement length
## Warning in non_nas[i] <- length(data[[col]]) - nas: number of items to replace
## is not a multiple of replacement length
## Warning in non_nas[i] <- length(data[[col]]) - nas: number of items to replace
## is not a multiple of replacement length
```

```
## Warning in non_nas[i] <- length(data[[col]]) - nas: number of items to replace
## is not a multiple of replacement length
## Warning in non_nas[i] <- length(data[[col]]) - nas: number of items to replace
## is not a multiple of replacement length
```

```
##    Variable Levels Missing NonMissing
## 1 candidate      4       0       8191
## 2      date    248       0       8191
## 3     state     30       0       8191
## 4     cycle      1       0       8191
## 5     party      3       0       8191
```

Notable things - not all 50 states are represented, since we only have 30 levels for the state. Only one cycle is represented here - further analysis on this might require addition of other cycles to corroborate results. Also, there are no missing data values (Except for the column that we dropped which was entirely empty) - very clear data set… and there are 4 candidates represented, although in later terms there are only 2 as people dropped out of the race.

Interesting things to potentially look at - could Biden's initial run be understood as a separate "mini" race within this election? Can we use that as a different case study?

```
  # **Continuous variables**:
  #   -   The number of non-missing observations.
  #   -   The number of missing observations.
  #   -   Measures of central tendency (e.g., mean, median).
  #   -   Measures of variability (e.g., standard deviation, interquartile range [IQR]).

cont_vars <- c("pct_estimate", "hi", "lo")

cont_eda <- function(data, vars) {
  l <- length(cont_vars)
  mean <- numeric(l)
  median <- numeric(l)
  sd <- numeric(l)
  iqr <- numeric(l)
  nas <- numeric(l)
  non_nas <- numeric(l)


  #iterate through cont varaibles
  for (i in 1:l) {
    col <- vars[i]
    #update values appropriately
    x <- data[[col]]
    nas[i] <- sum(is.na(x))
    non_nas[i] <- length(x) - nas
    mean[i] <- mean(x)
    median[i] <- median(x)
    sd[i] <- sd(x)
    iqr[i] <- IQR(x)
  }

   #put results into table
   results <- data.frame(
    Variable = vars,
    Missing = nas,
```

```r
    NonMissing = non_nas,
    Mean = mean,
    Median = median,
    IQR = iqr,
    StandardDeviation = sd)

   return(results)
}

cont_eda(polling2024, cont_vars)
```

```
## Warning in non_nas[i] <- length(x) - nas: number of items to replace is not a
## multiple of replacement length
## Warning in non_nas[i] <- length(x) - nas: number of items to replace is not a
## multiple of replacement length
## Warning in non_nas[i] <- length(x) - nas: number of items to replace is not a
## multiple of replacement length
```

```
##       Variable Missing NonMissing     Mean   Median       IQR StandardDeviation
## 1 pct_estimate       0       8191 36.50075 42.41610 10.748800          15.42268
## 2           hi       0       8191 38.71783 44.63370  9.842059          15.26338
## 3           lo       0       8191 34.28866 40.23686 11.625625          15.60190
```

```r
#polling information by state
polling_bystate <- polling2024 %>%
  filter(candidate == "Trump" | candidate == "Harris") %>%
  group_by(state, candidate) %>%
  summarize(mean = mean(pct_estimate),
          median = median(pct_estimate),
          sd = sd(pct_estimate),
          iqr = IQR(pct_estimate))
```

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

```r
polling_bystate
```

```
## # A tibble: 60 x 6
## # Groups:   state [30]
##    state      candidate  mean median    sd   iqr
##    <chr>      <chr>     <dbl>  <dbl> <dbl> <dbl>
##  1 Arizona    Harris     46.2   46.7 0.913 1.01
##  2 Arizona    Trump      44.9   44.3 2.11  4.03
##  3 California Harris     59.4   59.4 0.371 0.659
##  4 California Trump      32.5   31.6 2.23  4.39
##  5 Colorado   Harris     53.0   52.9 0.247 0.372
##  6 Colorado   Trump      41.6   41.7 0.215 0.304
##  7 Florida    Harris     44.8   45.0 0.793 1.07
##  8 Florida    Trump      47.8   47.3 1.92  3.54
##  9 Georgia    Harris     46.6   47.0 0.863 1.24
## 10 Georgia    Trump      45.9   45.6 1.59  2.59
## # i 50 more rows
```

Things to think about - again, we have no missing values, so there are no missing values throughout this entire dataset. This gives us a good idea for around where the range hovers (i.e. there was no clear winner), which gives evidence that this is an interesting question (since if there was an obvious winner, the additional granularity of the betting data might be useless because the answer is so obvious.) It might be interesting that the range of the the highs is actually lower than the lows, but the standard deviations are all relatively similar. Looking by state also gives us some interesting information!

Okay. now to start making visualizations!!

```r
#preliminary wrangling
results <- read.csv("data/actual_results_data/state_results_2024.csv")

results <- results %>%
  mutate(trump_win = case_when(Trump_Votes > Harris_Votes ~ 1,
                               Trump_Votes < Harris_Votes ~ 0))

polling_bystate_rshape <- polling_bystate %>%
  select(state, candidate, mean) %>%
  pivot_wider(names_from = candidate, values_from = mean) %>%
  mutate(pred_twin = case_when(Trump > Harris ~ 1,
                               Trump < Harris ~ 0))

state_polls <- filter(polling_bystate, candidate == "Trump")
#below has a few NAs because of missing cd2s in nebraska, maine, and national reuslt
polling_bystate_tr <- left_join(polling_bystate_rshape, results, by = c("state" = "State"))

matched_results <- mutate(polling_bystate_tr,
                          match = case_when(trump_win == pred_twin ~ 1,
                                            trump_win != pred_twin ~ 0),
                          state = tolower(state))
```

```r
library(maps)
```

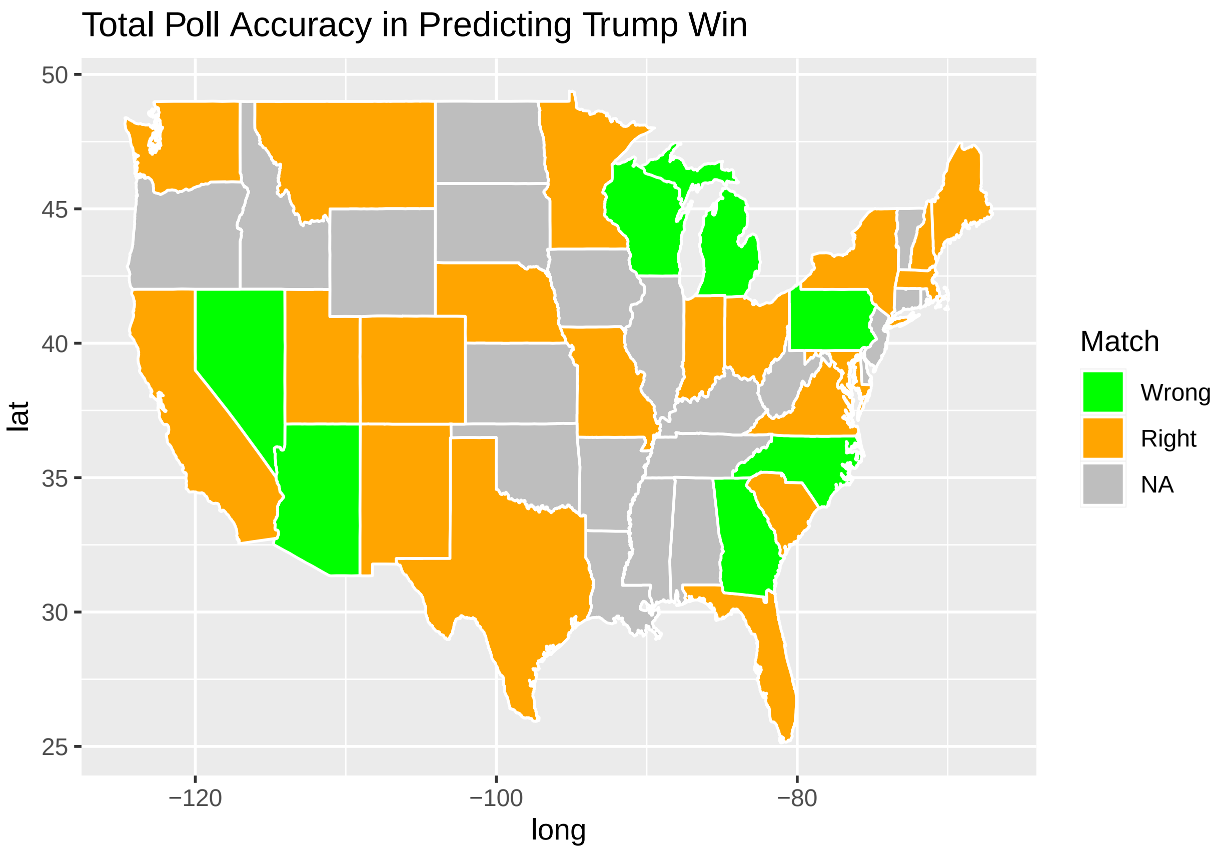
```
##
## Attaching package: 'maps'

## The following object is masked from 'package:purrr':
##
##     map
```

```r
us_states <- map_data("state")

us_states <- us_states %>%
  left_join(matched_results, by = c("region" = "state")) %>%
  mutate(match = coalesce(match, 2),
         match = as.factor(match))

ggplot(us_states, aes(long, lat, group = group, fill = match)) +
  geom_polygon(color = "white") +  # Draw state boundaries
  scale_fill_manual(
    values = c("0" = "green", "1" = "orange", "2" = "grey"),
    name = "Match",  # Legend title
    labels = c("Wrong", "Right", "NA"))  +
  labs(title = "Total Poll Accuracy in Predicting Trump Win")
```



Dataset 3: Election Results

The dataset contains the number of votes for Harris and Trump, separated by state. We find the percentage who voted for Harris and Trump, the amount of missing and non-missing data, and the mean and variance.

```
actual_results <- read_csv("data/actual_results_data/state_results_2024.csv")
```

```
## Rows: 51 Columns: 3
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (1): State
## dbl (2): Harris_Votes, Trump_Votes
##
## 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.
```

```
# Add percentage columns for Harris and Trump
actual_results <- actual_results %>%
  mutate(
    percent_harris = (Harris_Votes / (Harris_Votes + Trump_Votes)) * 100,
    percent_trump = (Trump_Votes / (Harris_Votes + Trump_Votes)) * 100
  )

# Calculate the number of missing and non-missing observations for each column
missing_summary <- actual_results %>%
  summarise(
    missing_harris = sum(is.na(Harris_Votes)),
    non_missing_harris = sum(!is.na(Harris_Votes)),
    missing_trump = sum(is.na(Trump_Votes)),
```

```
    non_missing_trump = sum(!is.na(Trump_Votes)),
    missing_state = sum(is.na(State)),
    non_missing_trump = sum(!is.na(State))
  )

# Calculate the mean and variance for votes and percentages
stats_summary <- actual_results %>%
  summarise(
    mean_percent_harris = mean(percent_harris, na.rm = TRUE),
    var_percent_harris = var(percent_harris, na.rm = TRUE),
    mean_percent_trump = mean(percent_trump, na.rm = TRUE),
    var_percent_trump = var(percent_trump, na.rm = TRUE)
  )

#print(missing_summary)
#print(stats_summary)
```

This dataset does not contain any missing data.

Final Thoughts

The analysis highlights that all three datasets—betting market data, polling data, and actual election results—are well-structured with minimal to no missing values, giving us reliable and comprehensive insights. The lack of missing data facilitates robust comparisons across datasets, allowing us to evaluate the predictive accuracy of betting markets and polls effectively. Interestingly, the variability observed in polling data and betting market probabilities reflects the competitiveness of the 2024 U.S. presidential election, showing the value of nuanced data sources for prediction. This data can provide a glimpse into how markets can be telltales of current political trends.

Baseline Model

Let's start by reading in the actual data and manipulating it such that we have a column titles 'trump_win' where a 1 denotes a state he won and a 0 denotes a state he lost.

```
results <- read.csv("data/actual_results_data/state_results_2024.csv")
results <- results %>%
mutate(trump_win = case_when(Trump_Votes > Harris_Votes ~ 1,
Trump_Votes < Harris_Votes ~ 0))

results <- results %>%
mutate(
percent_harris = (Harris_Votes / (Harris_Votes + Trump_Votes)) * 100,
percent_trump = (Trump_Votes / (Harris_Votes + Trump_Votes)) * 100
)
```

```
colnames(final_data)[colnames(final_data) == "state"] <- "State"
final_data$State <- state.name[match(final_data$State, state.abb)]
final_data <- left_join(final_data, results, by = "State")
```

```
library(tidyverse)
betting_data <- final_data %>%
  pivot_wider(
    names_from = candidate,
    values_from = percentage,
    names_prefix = "percentage_"
  )
```

```r
colnames(betting_data) <- gsub(" ", "_", colnames(betting_data))

head(betting_data)
```

```
## # A tibble: 6 x 8
##   State      Harris_Votes Trump_Votes trump_win percent_harris percent_trump
##   <chr>             <int>       <int>     <dbl>          <dbl>         <dbl>
## 1 Alaska           130763      175489         1           42.7          57.3
## 2 Alabama          769391     1457704         1           34.5          65.5
## 3 Arkansas         396077      758393         1           34.3          65.7
## 4 Arizona         1577729     1764862         1           47.2          52.8
## 5 California      8879713     5747751         0           60.7          39.3
## 6 Colorado        1727576     1377040         0           55.6          44.4
## # i 2 more variables: percentage_Donald_Trump <dbl>,
## #   percentage_Kamala_Harris <dbl>
```

```r
betting_model <- lm(trump_win ~ `percentage_Donald_Trump` + `percentage_Kamala_Harris`, data = betting_
summary(betting_model)
```

```
##
## Call:
## lm(formula = trump_win ~ percentage_Donald_Trump + percentage_Kamala_Harris,
##     data = betting_data)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.19720 -0.06911 -0.04999 -0.02562  0.50662
##
## Coefficients:
##                          Estimate Std. Error t value Pr(>|t|)
## (Intercept)                 3.324      4.595   0.723    0.473
## percentage_Donald_Trump    -2.243      4.656  -0.482    0.632
## percentage_Kamala_Harris   -3.367      4.629  -0.727    0.471
##
## Residual standard error: 0.167 on 47 degrees of freedom
## Multiple R-squared:  0.8887, Adjusted R-squared:  0.884
## F-statistic: 187.7 on 2 and 47 DF,  p-value: < 2.2e-16
```

```r
polling_bystate <- polling2024 %>%
filter(candidate == "Trump" | candidate == "Harris") %>%
group_by(state, candidate) %>%
summarize(mean = mean(pct_estimate),
median = median(pct_estimate),
sd = sd(pct_estimate),
iqr = IQR(pct_estimate))
```

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

```r
polls_data <- polling_bystate %>%
  pivot_wider(
    names_from = candidate,
    values_from = c(median, sd, iqr, mean)
  )
```

```r
head(polls_data)
```

```
## # A tibble: 6 x 9
## # Groups:   state [6]
##   state      median_Harris median_Trump sd_Harris sd_Trump iqr_Harris iqr_Trump
##   <chr>              <dbl>        <dbl>     <dbl>    <dbl>      <dbl>     <dbl>
## 1 Arizona             46.7         44.3    0.913    2.11        1.01      4.03
## 2 California          59.4         31.6    0.371    2.23        0.659     4.39
## 3 Colorado            52.9         41.7    0.247    0.215       0.372     0.304
## 4 Florida             45.0         47.3    0.793    1.92        1.07      3.54
## 5 Georgia             47.0         45.6    0.863    1.59        1.24      2.59
## 6 Indiana             39.2         56.1    0.0969   0.0853      0.158     0.0549
## # i 2 more variables: mean_Harris <dbl>, mean_Trump <dbl>
```

```r
colnames(polls_data)[colnames(polls_data) == "state"] <- "State"
polls_data <- left_join(polls_data, results, by = "State")

polling_model <- lm(trump_win ~ mean_Harris + mean_Trump, data = polls_data)

summary(polling_model)
```

```
##
## Call:
## lm(formula = trump_win ~ mean_Harris + mean_Trump, data = polls_data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -0.5371 -0.2102  0.1142  0.2358  0.3531
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  8.27215    3.80187   2.176   0.0396 *
## mean_Harris -0.11203    0.04221  -2.654   0.0139 *
## mean_Trump  -0.05149    0.03999  -1.288   0.2101
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.3065 on 24 degrees of freedom
##   (3 observations deleted due to missingness)
## Multiple R-squared:  0.6542, Adjusted R-squared:  0.6253
## F-statistic:  22.7 on 2 and 24 DF,  p-value: 2.928e-06
```

```r
betting_data$predictions <- predict(betting_model, betting_data)

polls_data$predictions <- predict(polling_model, polls_data)
```

```r
colnames(polls_data)[colnames(polls_data) == "state"] <- "State"

polling_model <- lm(trump_win ~ mean_Harris + mean_Trump, data = polls_data)

summary(polling_model)
```

```
##
## Call:
## lm(formula = trump_win ~ mean_Harris + mean_Trump, data = polls_data)
```

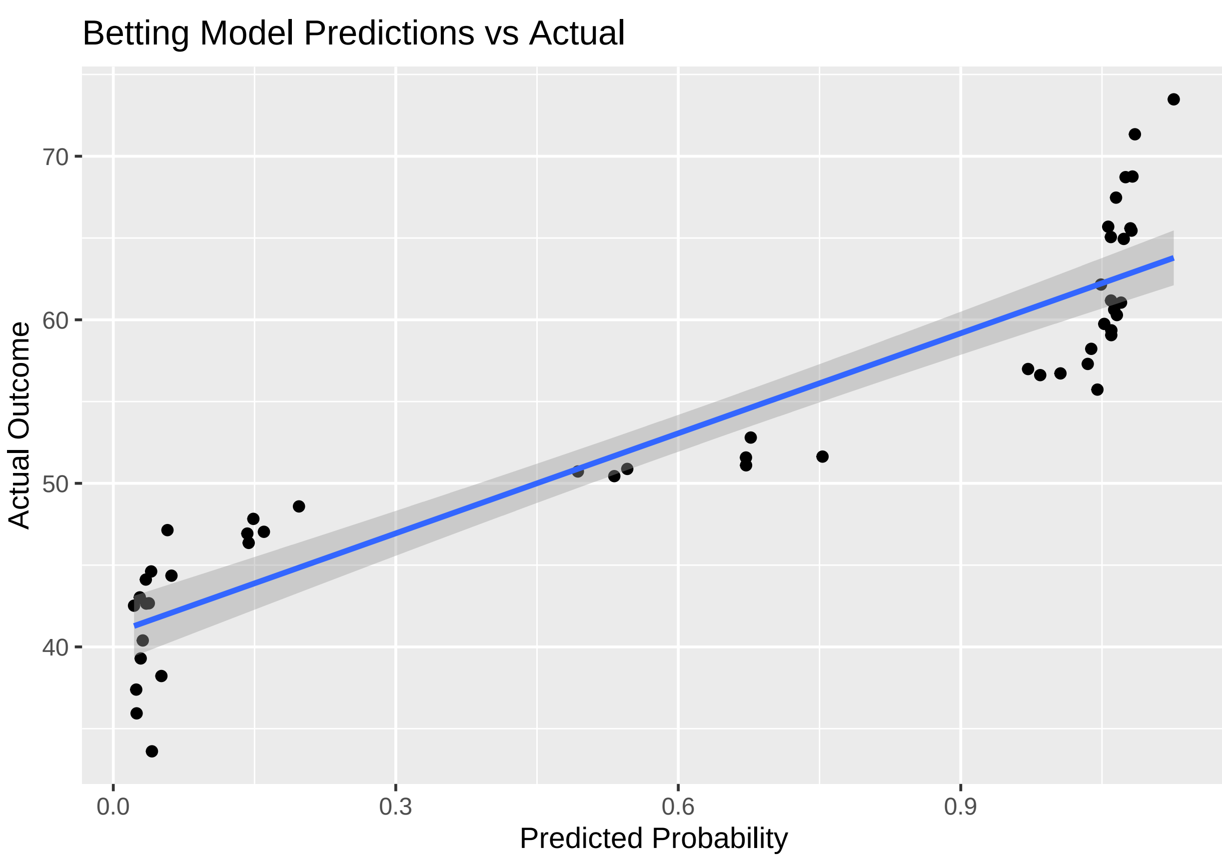
```
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -0.5371 -0.2102  0.1142  0.2358  0.3531
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  8.27215    3.80187   2.176   0.0396 *
## mean_Harris -0.11203    0.04221  -2.654   0.0139 *
## mean_Trump  -0.05149    0.03999  -1.288   0.2101
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.3065 on 24 degrees of freedom
##   (3 observations deleted due to missingness)
## Multiple R-squared:  0.6542, Adjusted R-squared:  0.6253
## F-statistic:  22.7 on 2 and 24 DF,  p-value: 2.928e-06
```

```
betting_data$predictions <- predict(betting_model, betting_data)

polls_data$predictions <- predict(polling_model, polls_data)
```

```
ggplot(betting_data, aes(x = predictions, y = percent_trump)) +
  geom_point() +
  geom_smooth(method = "lm") +
  labs(title = "Betting Model Predictions vs Actual", x = "Predicted Probability", y = "Actual Outcome")
```

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

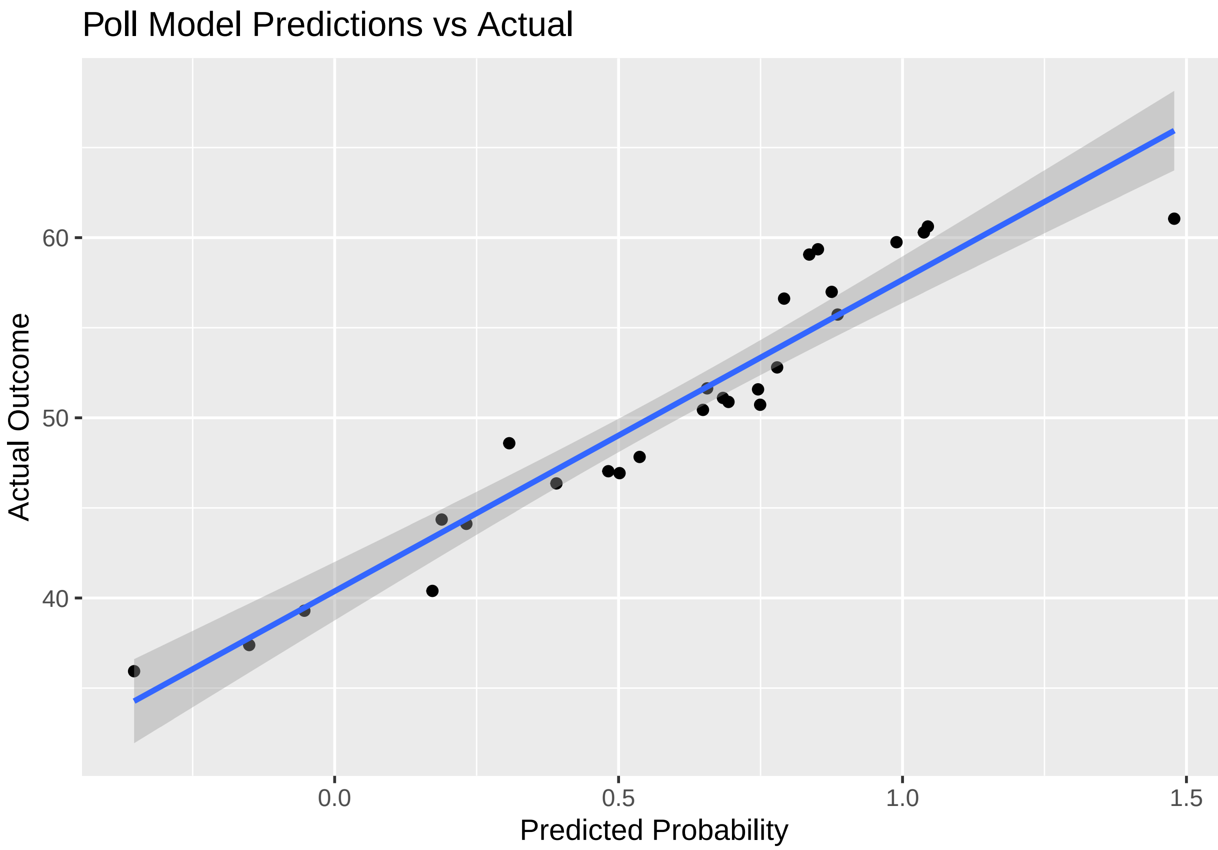


```r
ggplot(polls_data, aes(x = predictions, y = percent_trump)) +
  geom_point() +
  geom_smooth(method = "lm") +
  labs(title = "Poll Model Predictions vs Actual", x = "Predicted Probability", y = "Actual Outcome")
```

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

## Warning: Removed 3 rows containing non-finite outside the scale range
## (`stat_smooth()`).

## Warning: Removed 3 rows containing missing values or values outside the scale range
## (`geom_point()`).
```



We see from the figures above that comparing the baseline models from the polling data and the betting data, that the polling data currently seems to be much more accurate. However, we hope to explore if that statement stills holds once we adjust our models and improve their accuracy.

Bibliography

- Kaggle. *Polymarket 2024 US Election State Data*. Retrieved from: https://www.kaggle.com/datasets/pbizil/polymarket-2024-us-election-state-data

- FiveThirtyEight. *2024 Presidential Polls*. Retrieved from: https://projects.fivethirtyeight.com/polls/president-general/2024/national/

- CBS News. *2024 Presidential Election Results*. Retrieved from: https://www.cbsnews.com/elections/2024/president/