3_odds_us_race_not_eligible

Maiko Hata

```
library(tidyverse)
```

```
-- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
v dplyr     1.1.4     v readr     2.1.5
v forcats   1.0.0     v stringr   1.5.1
v ggplot2   3.5.2     v tibble    3.3.0
v lubridate 1.9.4     v tidyr     1.3.1
v purrr     1.0.4
-- Conflicts ------------------------------------------ tidyverse_conflicts() --
x dplyr::filter() masks stats::filter()
x dplyr::lag()    masks stats::lag()
i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to becom
```

```
library(here)
```

```
here() starts at /Users/hata/Desktop/EDLD652_Diss
```

```
library(rio)
library(knitr)
library(gt)
library(DT)
library(reactable)
library(gtsummary)
library(kableExtra)
```

```

Attaching package: 'kableExtra'

The following object is masked from 'package:dplyr':

    group_rows
```

```r
library(tinytex)
library(janitor)
```

```
Attaching package: 'janitor'

The following objects are masked from 'package:stats':

    chisq.test, fisher.test
```

```r
library(tidylog)
```

```
Attaching package: 'tidylog'

The following objects are masked from 'package:gtsummary':

    mutate, select

The following objects are masked from 'package:dplyr':

    add_count, add_tally, anti_join, count, distinct, distinct_all,
    distinct_at, distinct_if, filter, filter_all, filter_at, filter_if,
    full_join, group_by, group_by_all, group_by_at, group_by_if,
    inner_join, left_join, mutate, mutate_all, mutate_at, mutate_if,
    relocate, rename, rename_all, rename_at, rename_if, rename_with,
    right_join, sample_frac, sample_n, select, select_all, select_at,
    select_if, semi_join, slice, slice_head, slice_max, slice_min,
    slice_sample, slice_tail, summarise, summarise_all, summarise_at,
    summarise_if, summarize, summarize_all, summarize_at, summarize_if,
    tally, top_frac, top_n, transmute, transmute_all, transmute_at,
    transmute_if, ungroup

The following objects are masked from 'package:tidyr':

    drop_na, fill, gather, pivot_longer, pivot_wider, replace_na,
    separate_wider_delim, separate_wider_position,
    separate_wider_regex, spread, uncount

The following object is masked from 'package:stats':
```

```
    filter
```

```
library(sjPlot)
```

```
#refugeeswelcome
```

```
library(lme4)
```

```
Loading required package: Matrix
```

```
Attaching package: 'Matrix'
```

```
The following objects are masked from 'package:tidyr':

    expand, pack, unpack
```

```
Attaching package: 'lme4'
```

```
The following object is masked from 'package:rio':

    factorize
```

```
library(tibble)
library(dplyr)
library(epitools)
library(readxl)
library(pwr)
library(rcompanion)
library(grateful)
library(distill)
library(readxl)
library(scales)
```

```
Attaching package: 'scales'
```

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

    discard
```

```
The following object is masked from 'package:readr':

    col_factor
```

```
library(tidyr)
library(patchwork)
library(corrplot)
```

```
corrplot 0.95 loaded
```

```
library(distill)
library(tibble)
library(rcartocolor)
library(ggplot2)
library(quarto)
library(descr)
```

```
Attaching package: 'descr'

The following object is masked from 'package:janitor':

    crosstab
```

```
race_us_2 <- read.csv("/Users/hata/Desktop/EDLD652_Diss/Data/race_us_2.csv")
```

```
colnames(race_us_2) <- c(
  "Race",
  "area",
  "Total Exits",
  "Withdrawn",
  "Dismissed",
  "Moved Out",
  "Part B Eligible",
  "Not Eligible",
  "Not Determined"
)
```

```r
# PREPPING race_us_2 for odds ratio for NOT DETERMINED and NOT ELIGIBLE in 3_odds_us_race.qmd
# Step 3 (Revised): Clean and rename the dataset to prepare for odds ratio analysis

race_us_2 <- race_us_2 %>%
  select(-`Total Exits`, -`area`) %>%
  rename(
    `Withdrawn` = `Withdrawn`,
    `Dismissed` = `Dismissed`,
    `Moved Out` = `Moved Out`,
    `Part B Eligible` = `Part B Eligible`,
    `Not Eligible` = `Not Eligible`,
    `Not Determined` = `Not Determined`
  )
```

```
select: dropped 2 variables (area, Total Exits)
```

1) Odds Ratio for NOT ELIGIBLE by Race (National)

```r
# Step 4: Define a function to compute the odds ratio for being 'Not Eligible' using Chi-Squa

get_not_eligible_odds <- function(target_group) {
  # Step 4.1: Get 'N' count for target group
  a <- race_us_2 %>%
    filter(Race == target_group) %>%
    pull(`Not Eligible`)

  # Step 4.2: Get other group's count for target group (sum of other categories)
  b <- race_us_2 %>%
    filter(Race == target_group) %>%
    select(-Race, -`Not Eligible`) %>%
    unlist() %>%
    sum()

  # Step 4.3: Get total 'Not Eligible' count for all other groups
  c <- race_us_2 %>%
    filter(Race != target_group) %>%
    pull(`Not Eligible`) %>%
    sum()

  # Step 4.4: Get total non-'Not Eligible' count for all other groups
```

```
  d <- race_us_2 %>%
    filter(Race != target_group) %>%
    select(-Race, -`Not Eligible`) %>%
    unlist() %>%
    sum()

  # Step 4.5: Create 2x2 matrix for Chi-Square test
  mat <- matrix(c(a, b, c, d), nrow = 2, byrow = TRUE)

  # Step 4.6: Run Chi-Square test and return result
  result <- chisq.test(mat)

  # Optional: Manually calculate odds ratio from matrix
  odds_ratio <- (a / b) / (c / d)
  ci_log <- log(odds_ratio) + c(-1, 1) * 1.96 * sqrt(1/a + 1/b + 1/c + 1/d)

  tibble(
    `Odds Ratio` = odds_ratio,
    `P Value` = result$p.value,
    Lower = round(exp(ci_log[1]), 2),
    Upper = round(exp(ci_log[2]), 2)
  )
}
```

```
# Step 5: Apply the function to compute odds ratios for all race groups (Not Eligible)

# Step 5.1: Get the list of unique race categories
race_list <- unique(race_us_2$Race)

# Step 5.2: Use purrr::map to apply get_not_eligible_odds to each race
library(purrr)

summary_df_odds_noteligible <- map_dfr(race_list, function(race) {
  get_not_eligible_odds(race) %>%
    mutate(Race = race)
})
```

```
filter: removed 6 rows (86%), one row remaining
filter: removed 6 rows (86%), one row remaining
select: dropped 2 variables (Race, Not Eligible)
filter: removed one row (14%), 6 rows remaining
filter: removed one row (14%), 6 rows remaining
```

```
select: dropped 2 variables (Race, Not Eligible)
mutate: new variable 'Race' (character) with one unique value and 0% NA
filter: removed 6 rows (86%), one row remaining
filter: removed 6 rows (86%), one row remaining
select: dropped 2 variables (Race, Not Eligible)
filter: removed one row (14%), 6 rows remaining
filter: removed one row (14%), 6 rows remaining
select: dropped 2 variables (Race, Not Eligible)
mutate: new variable 'Race' (character) with one unique value and 0% NA
filter: removed 6 rows (86%), one row remaining
filter: removed 6 rows (86%), one row remaining
select: dropped 2 variables (Race, Not Eligible)
filter: removed one row (14%), 6 rows remaining
filter: removed one row (14%), 6 rows remaining
select: dropped 2 variables (Race, Not Eligible)
mutate: new variable 'Race' (character) with one unique value and 0% NA
filter: removed 6 rows (86%), one row remaining
filter: removed 6 rows (86%), one row remaining
select: dropped 2 variables (Race, Not Eligible)
filter: removed one row (14%), 6 rows remaining
filter: removed one row (14%), 6 rows remaining
select: dropped 2 variables (Race, Not Eligible)
mutate: new variable 'Race' (character) with one unique value and 0% NA
filter: removed 6 rows (86%), one row remaining
filter: removed 6 rows (86%), one row remaining
select: dropped 2 variables (Race, Not Eligible)
filter: removed one row (14%), 6 rows remaining
filter: removed one row (14%), 6 rows remaining
select: dropped 2 variables (Race, Not Eligible)
mutate: new variable 'Race' (character) with one unique value and 0% NA
filter: removed 6 rows (86%), one row remaining
filter: removed 6 rows (86%), one row remaining
select: dropped 2 variables (Race, Not Eligible)
filter: removed one row (14%), 6 rows remaining
filter: removed one row (14%), 6 rows remaining
select: dropped 2 variables (Race, Not Eligible)
mutate: new variable 'Race' (character) with one unique value and 0% NA
filter: removed 6 rows (86%), one row remaining
filter: removed 6 rows (86%), one row remaining
select: dropped 2 variables (Race, Not Eligible)
filter: removed one row (14%), 6 rows remaining
filter: removed one row (14%), 6 rows remaining
select: dropped 2 variables (Race, Not Eligible)
```

```
mutate: new variable 'Race' (character) with one unique value and 0% NA
```

```r
colnames(summary_df_odds_noteligible)
```

```
[1] "Odds Ratio" "P Value"    "Lower"      "Upper"      "Race"
```

```r
summary_df_odds_noteligible <- summary_df_odds_noteligible %>%
  bind_rows(tibble(
    Race = "National Average",
    `Odds Ratio` = 1,
    `P Value` = NA,
    Lower = NA,
    Upper = NA
  )) %>%
  mutate(
    `Odds Ratio` = round(`Odds Ratio`, 2),
    Lower = round(Lower, 2),
    Upper = round(Upper, 2),
    `P Value` = ifelse(is.na(`P Value`), NA, ifelse(`P Value` < 0.001, "< .001", round(`P Val
    Race = ifelse(Race == "National Average", "zzz_National Average", Race)
  ) %>%
  arrange(Race) %>%
  mutate(Race = ifelse(Race == "zzz_National Average", "National Average", Race))
```

```
mutate: changed 7 values (88%) of 'Odds Ratio' (0 new NAs)
        converted 'P Value' from double to character (0 new NA)
        changed one value (12%) of 'Race' (0 new NAs)
mutate: changed one value (12%) of 'Race' (0 new NAs)
```

```r
library(knitr)
library(kableExtra)

kable(summary_df_odds_noteligible, align = "lcccc") %>%
  kable_styling(full_width = FALSE, position = "center") %>%
  kable_classic(full_width = FALSE, html_font = "Cambria")
```

Odds Ratio	P Value	Lower	Upper	Race
0.83	< .001	0.80	0.86	Alaska Native or American Indian
0.85	< .001	0.84	0.86	Asian

0.68	< .001	0.68	0.69	Black or African American
0.79	< .001	0.78	0.79	Hispanic or Latino
0.84	< .001	0.80	0.89	Pacific Islander
1.00	0.96	0.99	1.01	Two or More Races
1.45	< .001	1.45	1.46	White
1.00	NA	NA	NA	National Average

```
library(knitr)
library(kableExtra)

kable(summary_df_odds_noteligible, align = "lcccc") %>%
  kable_styling(full_width = FALSE, position = "center") %>%
  kable_classic(full_width = FALSE, html_font = "Cambria")
```

Odds Ratio	P Value	Lower	Upper	Race
0.83	< .001	0.80	0.86	Alaska Native or American Indian
0.85	< .001	0.84	0.86	Asian
0.68	< .001	0.68	0.69	Black or African American
0.79	< .001	0.78	0.79	Hispanic or Latino
0.84	< .001	0.80	0.89	Pacific Islander
1.00	0.96	0.99	1.01	Two or More Races
1.45	< .001	1.45	1.46	White
1.00	NA	NA	NA	National Average

Not Eligible Cohen's h

```
# Step 1: Load necessary libraries
library(dplyr)
library(knitr)
library(kableExtra)

# Step 2: Define Cohen's h function
compute_cohens_h <- function(p1, p2) {
  2 * asin(sqrt(p1)) - 2 * asin(sqrt(p2))
}

# Step 3: Read and prepare the data
race_us_2_clean2 <- read.csv("/Users/hata/Desktop/EDLD652_Diss/Data/race_us_2.csv")
```

```
colnames(race_us_2_clean2)
```

```
[1] "Race"            "area"            "Total.Exits"     "Withdrawn"
[5] "Dismissed"       "Moved.Out"       "Part.B.Eligible" "Not.Eligible"
[9] "Not.Determined"
```

```
### Preparing clean data
race_us_2_clean2 <- race_us_2_clean2 %>%
  mutate(
    Withdrawn = as.numeric(Withdrawn),
    Dismissed = as.numeric(Dismissed),
    Moved.Out = as.numeric(Moved.Out),
    Part.B.Eligible = as.numeric(Part.B.Eligible),
    Not.Eligible = as.numeric(Not.Eligible),
    Not.Determined = as.numeric(Not.Determined),
    Total = Withdrawn + Dismissed + Moved.Out +
            Part.B.Eligible + Not.Eligible + Not.Determined,
    Proportion = Not.Eligible / Total
  )
```

```
mutate: converted 'Withdrawn' from integer to double (0 new NA)
        converted 'Dismissed' from integer to double (0 new NA)
        converted 'Moved.Out' from integer to double (0 new NA)
        converted 'Part.B.Eligible' from integer to double (0 new NA)
        converted 'Not.Eligible' from integer to double (0 new NA)
        converted 'Not.Determined' from integer to double (0 new NA)
        new variable 'Total' (double) with 7 unique values and 0% NA
        new variable 'Proportion' (double) with 7 unique values and 0% NA
```

```
# Compute national row
national_row <- race_us_2_clean2 %>%
  summarise(
    Race = "National Average",
    Withdrawn = sum(Withdrawn),
    Dismissed = sum(Dismissed),
    Moved.Out = sum(Moved.Out),
    Part.B.Eligible = sum(Part.B.Eligible),
    Not.Eligible = sum(Not.Eligible),
    Not.Determined = sum(Not.Determined)
  ) %>%
```

```
  mutate(
    Total = Withdrawn + Dismissed + Moved.Out + Part.B.Eligible + Not.Eligible + Not.Determi
    Proportion = Not.Eligible / Total
  )
```

```
summarise: now one row and 7 columns, ungrouped
mutate: new variable 'Total' (double) with one unique value and 0% NA
        new variable 'Proportion' (double) with one unique value and 0% NA
```

```
# Combine
race_us_2_clean2 <- bind_rows(race_us_2_clean2, national_row)
```

```
get_not_eligible_or <- function(group) {
  df <- race_us_2_clean2

  a <- df %>% filter(Race == group) %>% pull(Not.Eligible)
  b <- df %>% filter(Race == group) %>% pull(Total) - a
  c <- df %>% filter(Race == "National Average") %>% pull(Not.Eligible)
  d <- df %>% filter(Race == "National Average") %>% pull(Total) - c

  mat <- matrix(c(a, b, c, d), nrow = 2, byrow = TRUE)
  result <- chisq.test(mat)

  tibble(
    `Odds Ratio` = oddsratio.wald(mat)$measure[2, 1],
    `P Value` = result$p.value,
    Lower = oddsratio.wald(mat)$measure[2, 2],
    Upper = oddsratio.wald(mat)$measure[2, 3]
  )
}
```

```
races <- race_us_2_clean2$Race
races <- races[races != "National Average"]  #  Exclude National Average

summary_df_not_eligible <- map_dfr(races, function(race) {
  get_not_eligible_or(race) %>% mutate(Race = race)
})
```

```
filter: removed 7 rows (88%), one row remaining
filter: removed 7 rows (88%), one row remaining
filter: removed 7 rows (88%), one row remaining
```

```
filter: removed 7 rows (88%), one row remaining
mutate: new variable 'Race' (character) with one unique value and 0% NA
filter: removed 7 rows (88%), one row remaining
filter: removed 7 rows (88%), one row remaining
filter: removed 7 rows (88%), one row remaining
filter: removed 7 rows (88%), one row remaining
mutate: new variable 'Race' (character) with one unique value and 0% NA
filter: removed 7 rows (88%), one row remaining
filter: removed 7 rows (88%), one row remaining
filter: removed 7 rows (88%), one row remaining
filter: removed 7 rows (88%), one row remaining
mutate: new variable 'Race' (character) with one unique value and 0% NA
filter: removed 7 rows (88%), one row remaining
filter: removed 7 rows (88%), one row remaining
filter: removed 7 rows (88%), one row remaining
filter: removed 7 rows (88%), one row remaining
mutate: new variable 'Race' (character) with one unique value and 0% NA
filter: removed 7 rows (88%), one row remaining
filter: removed 7 rows (88%), one row remaining
filter: removed 7 rows (88%), one row remaining
filter: removed 7 rows (88%), one row remaining
mutate: new variable 'Race' (character) with one unique value and 0% NA
filter: removed 7 rows (88%), one row remaining
filter: removed 7 rows (88%), one row remaining
filter: removed 7 rows (88%), one row remaining
filter: removed 7 rows (88%), one row remaining
mutate: new variable 'Race' (character) with one unique value and 0% NA
filter: removed 7 rows (88%), one row remaining
filter: removed 7 rows (88%), one row remaining
filter: removed 7 rows (88%), one row remaining
filter: removed 7 rows (88%), one row remaining
mutate: new variable 'Race' (character) with one unique value and 0% NA
```

```r
races <- race_us_2_clean2$Race
races <- races[races != "National Average"]  #  Exclude National Average

summary_df_not_eligible <- map_dfr(races, function(race) {
  get_not_eligible_or(race) %>% mutate(Race = race)
})
```

```
filter: removed 7 rows (88%), one row remaining
filter: removed 7 rows (88%), one row remaining
```

```
filter: removed 7 rows (88%), one row remaining
filter: removed 7 rows (88%), one row remaining
mutate: new variable 'Race' (character) with one unique value and 0% NA
filter: removed 7 rows (88%), one row remaining
filter: removed 7 rows (88%), one row remaining
filter: removed 7 rows (88%), one row remaining
filter: removed 7 rows (88%), one row remaining
mutate: new variable 'Race' (character) with one unique value and 0% NA
filter: removed 7 rows (88%), one row remaining
filter: removed 7 rows (88%), one row remaining
filter: removed 7 rows (88%), one row remaining
filter: removed 7 rows (88%), one row remaining
mutate: new variable 'Race' (character) with one unique value and 0% NA
filter: removed 7 rows (88%), one row remaining
filter: removed 7 rows (88%), one row remaining
filter: removed 7 rows (88%), one row remaining
filter: removed 7 rows (88%), one row remaining
mutate: new variable 'Race' (character) with one unique value and 0% NA
filter: removed 7 rows (88%), one row remaining
filter: removed 7 rows (88%), one row remaining
filter: removed 7 rows (88%), one row remaining
filter: removed 7 rows (88%), one row remaining
mutate: new variable 'Race' (character) with one unique value and 0% NA
filter: removed 7 rows (88%), one row remaining
filter: removed 7 rows (88%), one row remaining
filter: removed 7 rows (88%), one row remaining
filter: removed 7 rows (88%), one row remaining
mutate: new variable 'Race' (character) with one unique value and 0% NA
filter: removed 7 rows (88%), one row remaining
filter: removed 7 rows (88%), one row remaining
filter: removed 7 rows (88%), one row remaining
filter: removed 7 rows (88%), one row remaining
mutate: new variable 'Race' (character) with one unique value and 0% NA
```

```r
summary_df_not_eligible <- bind_rows(
  summary_df_not_eligible,
  tibble(
    Race = "National Average",
    `Odds Ratio` = 1,
    `P Value` = NA,
    Lower = NA,
    Upper = NA
```

```
  )
)
```

```
summary_df_not_eligible <- summary_df_not_eligible %>%
  mutate(
    `Odds Ratio` = round(`Odds Ratio`, 2),
    Lower = round(Lower, 2),
    Upper = round(Upper, 2),
    `P Value` = ifelse(is.na(`P Value`), NA, ifelse(`P Value` < 0.001, "< .001", round(`P Va
  )
```

```
mutate: changed 7 values (88%) of 'Odds Ratio' (0 new NAs)
        converted 'P Value' from double to character (0 new NA)
        changed 7 values (88%) of 'Lower' (0 new NAs)
        changed 7 values (88%) of 'Upper' (0 new NAs)
```

```
summary_df_not_eligible$Race <- factor(
  summary_df_not_eligible$Race,
  levels = rev(c(sort(setdiff(summary_df_not_eligible$Race, "National Average")), "National A
)
```

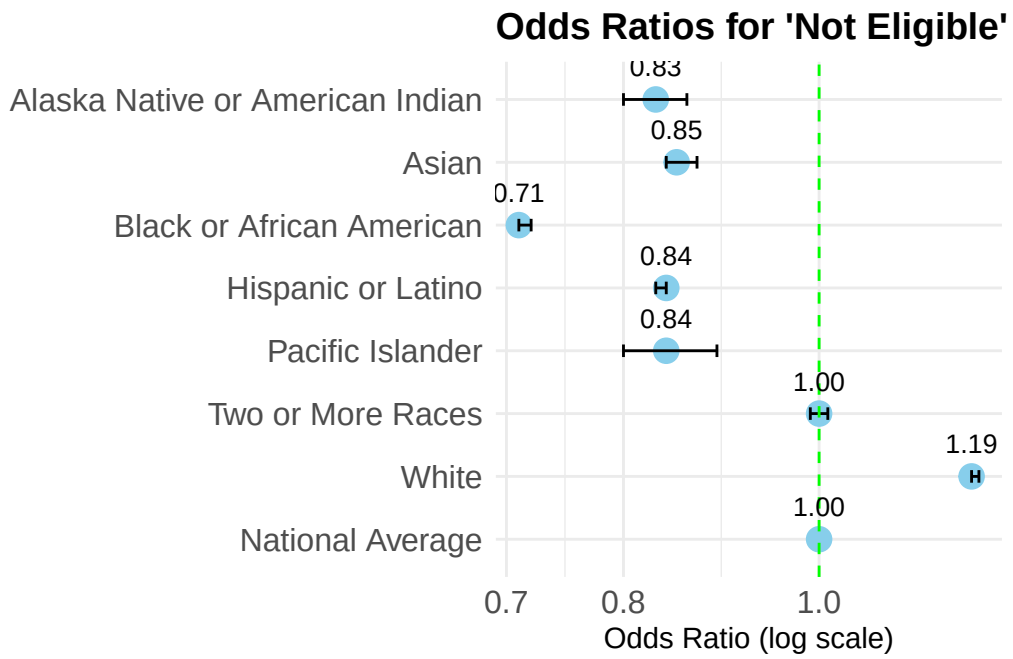
```
summary_df_not_eligible_plot <- summary_df_not_eligible %>%
  filter(Race != "National Average")
```

```
filter: removed one row (12%), 7 rows remaining
```

```
ggplot(summary_df_not_eligible, aes(x = `Odds Ratio`, y = Race)) +
  geom_point(color = "skyblue", size = 4) +
  geom_errorbarh(aes(xmin = Lower, xmax = Upper), height = 0.2) +
  geom_vline(xintercept = 1, linetype = "dashed", color = "green") +
  geom_text(aes(label = sprintf("%.2f", `Odds Ratio`)), vjust = -1.2, size = 3.5) +
  scale_x_log10() +
  labs(
    x = "Odds Ratio (log scale)",
    y = "",
    title = "Odds Ratios for 'Not Eligible' Exit Category by Race"
  ) +
  theme_minimal() +
  theme(
    axis.text.y = element_text(size = 12),
```

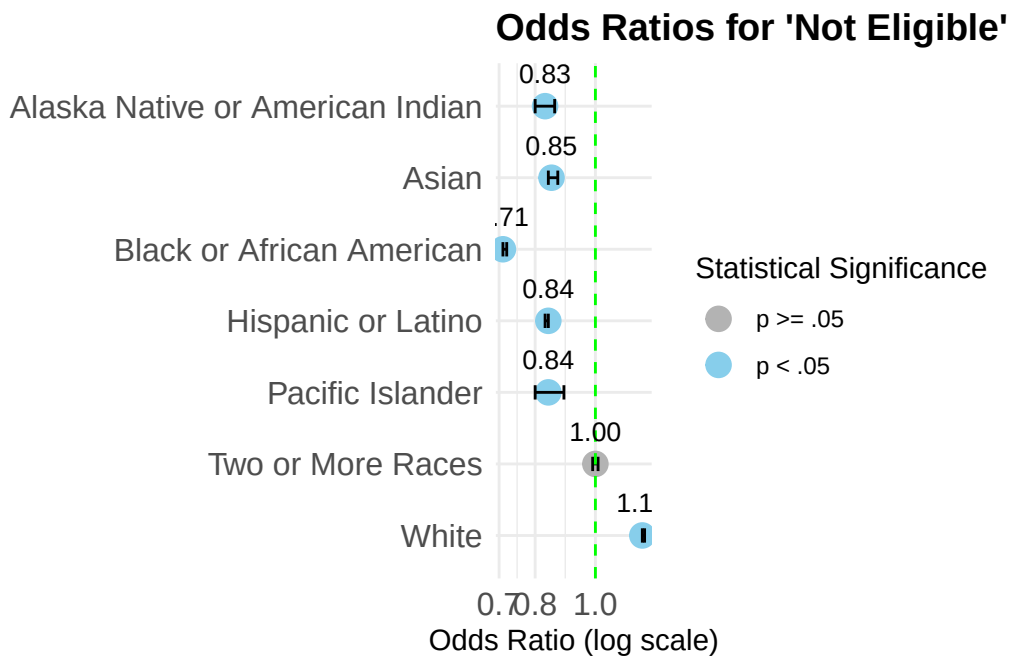
```
    axis.text.x = element_text(size = 12),
    plot.title = element_text(size = 14, face = "bold")
  )
```

```
Warning: Removed 1 row containing missing values or values outside the scale range
(`geom_errorbarh()`).
```




```
ggplot(summary_df_not_eligible_plot, aes(x = `Odds Ratio`, y = Race)) +
  geom_point(aes(color = `P Value` < 0.05), size = 4) +
  geom_errorbarh(aes(xmin = Lower, xmax = Upper), height = 0.2) +
  geom_vline(xintercept = 1, linetype = "dashed", color = "green") +  # keep this!
  geom_text(aes(label = sprintf("%.2f", `Odds Ratio`)), vjust = -1.2, size = 3.5) +
  scale_x_log10() +
  scale_color_manual(
    values = c("TRUE" = "skyblue", "FALSE" = "gray70"),
    labels = c("TRUE" = "p < .05", "FALSE" = "p   .05")
  ) +
  labs(
    x = "Odds Ratio (log scale)",
    y = "",
    color = "Statistical Significance",
```

```
    title = "Odds Ratios for 'Not Eligible' Exit Category by Race"
  ) +
  theme_minimal() +
  theme(
    axis.text.y = element_text(size = 12),
    axis.text.x = element_text(size = 12),
    plot.title = element_text(size = 14, face = "bold")
  )
```




```
# races <- race_us_2_clean2$Race
# races <- races[races != "National Average"]  #   Exclude National Average
#
# summary_df_not_eligible <- map_dfr(races, function(race) {
#   get_not_eligible_or(race) %>% mutate(Race = race)
# })
```

```
summary_df_not_eligible <- bind_rows(
  summary_df_not_eligible,
  tibble(
    Race = "National Average",
    `Odds Ratio` = 1,
    `P Value` = NA,
```

```
    Lower = NA,
    Upper = NA
  )
)
```

```
# Step 2: Compute national Not Eligible rate
 national_ne <- sum(race_us_2_clean2$Not.Eligible, na.rm = TRUE)
 national_total <- sum(race_us_2_clean2$Total, na.rm = TRUE)
 national_rate <- national_ne / national_total

# # Step 3: Compute Cohen's h
 summary_df_h <- race_us_2_clean2 %>%
   mutate(
     `National Avg` = national_rate,
     `Cohen's h` = round(compute_cohens_h(Proportion, national_rate), 3)
   ) %>%
   select(
     Race,
     `Not Eligible N` = Not.Eligible,
     `Total N` = Total,
     `Not Eligible %` = Proportion,
     `National Avg %` = `National Avg`,
     `Cohen's h`
   ) %>%
   mutate(
     `Not Eligible %` = sprintf("%.2f%%", 100 * `Not Eligible %`),
     `National Avg %` = sprintf("%.2f%%", 100 * `National Avg %`)
   )
```

```
mutate: new variable 'National Avg' (double) with one unique value and 0% NA
        new variable 'Cohen's h' (double) with 7 unique values and 0% NA
select: renamed 4 variables (Not Eligible N, Total N, Not Eligible %, National Avg %) and dr
mutate: converted 'Not Eligible %' from double to character (0 new NA)
        converted 'National Avg %' from double to character (0 new NA)
```

```
 # Step 4: Display with kable
 kable(summary_df_h, caption = "Cohen's h for 'Not Eligible' Exit Rates by Race") %>%
   kable_styling(full_width = FALSE, position = "center") %>%
   kable_classic(full_width = FALSE, html_font = "Cambria")
```

Table 3: Cohen's h for 'Not Eligible' Exit Rates by Race

Race	Not Eligible N	Total N	Not Eligible %	National Avg %	Cohen's h
Alaska Native or American Indian	4395	23372	18.80%	21.83%	-0.075
Asian	26712	138727	19.26%	21.83%	-0.064
Black or African American	65353	394149	16.58%	21.83%	-0.134
Hispanic or Latino	165654	874188	18.95%	21.83%	-0.072
Two or More Races	28996	132856	21.83%	21.83%	0.000
Pacific Islander	1911	10033	19.05%	21.83%	-0.069
White	402568	1612911	24.96%	21.83%	0.074
National Average	695589	3186236	21.83%	21.83%	0.000

```
#  Step: Write Cohen's h summary for 'Not Eligible' exit category by race to CSV
write.csv(
  summary_df_h,
  file = "/Users/hata/Desktop/EDLD652_Diss/Data/cohens_h_noteligible_by_race.csv",
  row.names = FALSE
)
```

ODDS RATIO FORREST PLOT for NOT ELIGIBLE

```
# get_not_eligible_or <- function(group) {
#   df <- race_us_2_clean2 %>%
#     mutate(
#       Not.Eligible = as.numeric(Not.Eligible),
#       Total = Withdrawn + Dismissed + Moved.Out + Part.B.Eligible + Not.Eligible + Not.Dete
#     )
#
#   # Group of interest
#   a <- df %>% filter(Race == group) %>% pull(Not.Eligible)
#   b <- df %>% filter(Race == group) %>% pull(Total) - a
#
#   # Everyone else
#   c <- df %>% filter(Race != group) %>% summarise(sum(Not.Eligible, na.rm = TRUE)) %>% pull
#   d <- df %>% filter(Race != group) %>% summarise(sum(Total, na.rm = TRUE)) %>% pull() - c
#
#   # 2x2 table
#   mat <- matrix(c(a, b, c, d), nrow = 2, byrow = TRUE)
#   result <- chisq.test(mat)  # Use chi-square test instead of Fisher's test
```

```
#
#   tibble(
#     `Odds Ratio` = oddsratio.wald(mat)$measure[2, 1],
#     `P Value` = result$p.value,
#     Lower = oddsratio.wald(mat)$measure[2, 2],
#     Upper = oddsratio.wald(mat)$measure[2, 3]
#   )
# }
```

```
# Step 2: Apply it to all racial groups for 'Not Eligible'

# races <- race_us_2_clean2$Race

# summary_df_not_eligible <- purrr::map_dfr(races, function(race) {
#   get_not_eligible_or(race) %>%
#     mutate(Race = race)
# })
```

```
# Step 4: Clean and prepare for forest plot

# summary_df_not_eligible <- summary_df_not_eligible %>%
#   mutate(
#     `Odds Ratio` = round(`Odds Ratio`, 2),
#     Lower = round(Lower, 2),
#     Upper = round(Upper, 2),
#     `P Value` = ifelse(is.na(`P Value`), NA, ifelse(`P Value` < 0.001, "< .001", round(`P V
#   )
```

```
# Step 5: Make Race a factor so National Average is last

# summary_df_not_eligible$Race <- factor(summary_df_not_eligible$Race, levels = c(
#   setdiff(summary_df_not_eligible$Race, "National Average"), "National Average"
# ))
```

```
# # Step 6: Reverse order: top = A, bottom = Z (with National Average last)
#
# summary_df_not_eligible$Race <- factor(
#   summary_df_not_eligible$Race,
#   levels = rev(c(sort(setdiff(summary_df_not_eligible$Race, "National Average")), "National
# )
```

```
# Step 7: Forest plot for Not Eligible odds ratios

# ggplot(summary_df_not_eligible, aes(x = `Odds Ratio`, y = Race)) +
#   geom_point(color = "skyblue", size = 4) +
#   geom_errorbarh(aes(xmin = Lower, xmax = Upper), height = 0.2) +
#   geom_vline(xintercept = 1, linetype = "dashed", color = "green") +
#   geom_text(aes(label = sprintf("%.2f", `Odds Ratio`)), vjust = -1.2, size = 3.5) +
#   scale_x_log10() +
#   labs(
#     x = "Odds Ratio (log scale)",
#     y = "",
#     title = "Odds Ratios for 'Not Eligible' Exit Category by Race"
#   ) +
#   theme_minimal() +
#   theme(
#     axis.text.y = element_text(size = 12),
#     axis.text.x = element_text(size = 12),
#     plot.title = element_text(size = 14, face = "bold")
#   )
```