Group Assignment

2024-01-23

Group members:

- Tashfeen Ahmed
- Adrian Alarcon
- Yvan Kammelu
- Zhicheng Zhong

```
#install.packages(c("arrow","gender", "wru", "lubridate", "gtsummary"))
# Load required libraries
library(gender)
```

```
## Warning: package 'gender' was built under R version 4.2.3
```

```
library(wru)
```

```
## Warning: package 'wru' was built under R version 4.2.3
```

```
library(lubridate)
```

```
## Warning: package 'lubridate' was built under R version 4.2.3

##
## Attaching package: 'lubridate'

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

```
library(dplyr)
```

```
## Warning: package 'dplyr' was built under R version 4.2.3

##
## 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(gtsummary)
```

```
## Warning: package 'gtsummary' was built under R version 4.2.3
```

```r
library(arrow)
```

```
## Warning: package 'arrow' was built under R version 4.2.3

##
## Attaching package: 'arrow'

## The following object is masked from 'package:lubridate':
##
##     duration

## The following object is masked from 'package:utils':
##
##     timestamp
```

```r
library(tidyr)
```

```
## Warning: package 'tidyr' was built under R version 4.2.3
```

```r
library(zoo)
```

```
## Warning: package 'zoo' was built under R version 4.2.3

##
## Attaching package: 'zoo'

## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric
```

```r
library(purrr)
```

```
## Warning: package 'purrr' was built under R version 4.2.3
```

```r
data<- read_feather("app_data_starter.feather")
```

```r
# Task 1: Create individual-level variables
examiner_names <- data %>% distinct(examiner_name_first)
```

Obtaining gender of the examiner

Using the `gender` package, we identify the gender of the examiner based on the first name, according to the documentation.

```r
# get a table of names and gender

examiner_names_gender <- examiner_names %>%
  do(results = gender(.$examiner_name_first, method = "ssa")) %>%
  unnest(cols = c(results), keep_empty = TRUE) %>%
  select(
    examiner_name_first = name,
    gender,
    proportion_female
  )
```

In this part, we joined the gender data obtained in the previous step into the main dataset.

```r
# remove extra colums from the gender table
examiner_names_gender <- examiner_names_gender %>%
  select(examiner_name_first, gender)

# joining gender back to the dataset
data <- data %>%
  left_join(examiner_names_gender, by = "examiner_name_first")

# cleaning up
rm(examiner_names)
rm(examiner_names_gender)
gc()
```

```
##            used  (Mb) gc trigger  (Mb)  max used  (Mb)
## Ncells  4432047 236.7    8033454 429.1   4453825 237.9
## Vcells 59358952 452.9  124197188 947.6 103804676 792.0
```

Obtaining the race of the examiner

Based on the last name, and using the `wru` package, we identified the probability of the examiner to be of an specific race among Asian, Black, Hispanic and other.

```r
library(wru)

examiner_surnames <- data %>%
  select(surname = examiner_name_last) %>%
  distinct()

examiner_surnames
```

```
## # A tibble: 3,806 x 1
##    surname
##    <chr>
##  1 HOWARD
##  2 YILDIRIM
##  3 HAMILTON
##  4 MOSHER
##  5 BARR
##  6 GRAY
##  7 MCMILLIAN
##  8 FORD
##  9 STRZELECKA
## 10 KIM
## # i 3,796 more rows
```

```
examiner_race <- predict_race(voter.file = examiner_surnames, surname.only = T) %>%
  as_tibble()
```

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

```
## Proceeding with last name predictions...
```

```
## i All local files already up-to-date!
```

```
## 701 (18.4%) individuals' last names were not matched.
```

```
examiner_race
```

```
## # A tibble: 3,806 x 6
##    surname    pred.whi pred.bla pred.his pred.asi pred.oth
##    <chr>         <dbl>    <dbl>    <dbl>    <dbl>    <dbl>
##  1 HOWARD       0.597   0.295    0.0275   0.00690   0.0741
##  2 YILDIRIM     0.807   0.0273   0.0694   0.0165    0.0798
##  3 HAMILTON     0.656   0.239    0.0286   0.00750   0.0692
##  4 MOSHER       0.915   0.00425  0.0291   0.00917   0.0427
##  5 BARR         0.784   0.120    0.0268   0.00830   0.0615
##  6 GRAY         0.640   0.252    0.0281   0.00748   0.0724
##  7 MCMILLIAN    0.322   0.554    0.0212   0.00340   0.0995
##  8 FORD         0.576   0.320    0.0275   0.00621   0.0697
##  9 STRZELECKA   0.472   0.171    0.220    0.0825    0.0543
## 10 KIM          0.0169  0.00282  0.00546  0.943     0.0319
## # i 3,796 more rows
```

```
examiner_race <- examiner_race %>%
  mutate(max_race_p = pmax(pred.asi, pred.bla, pred.his, pred.oth, pred.whi)) %>%
  mutate(race = case_when(
    max_race_p == pred.asi ~ "Asian",
    max_race_p == pred.bla ~ "black",
    max_race_p == pred.his ~ "Hispanic",
    max_race_p == pred.oth ~ "other",
    max_race_p == pred.whi ~ "white",
    TRUE ~ NA_character_
  ))

examiner_race
```

```
## # A tibble: 3,806 x 8
##    surname    pred.whi pred.bla pred.his pred.asi pred.oth max_race_p race
##    <chr>         <dbl>    <dbl>    <dbl>    <dbl>    <dbl>      <dbl> <chr>
##  1 HOWARD       0.597   0.295    0.0275   0.00690   0.0741      0.597 white
##  2 YILDIRIM     0.807   0.0273   0.0694   0.0165    0.0798      0.807 white
##  3 HAMILTON     0.656   0.239    0.0286   0.00750   0.0692      0.656 white
##  4 MOSHER       0.915   0.00425  0.0291   0.00917   0.0427      0.915 white
##  5 BARR         0.784   0.120    0.0268   0.00830   0.0615      0.784 white
##  6 GRAY         0.640   0.252    0.0281   0.00748   0.0724      0.640 white
##  7 MCMILLIAN    0.322   0.554    0.0212   0.00340   0.0995      0.554 black
##  8 FORD         0.576   0.320    0.0275   0.00621   0.0697      0.576 white
##  9 STRZELECKA   0.472   0.171    0.220    0.0825    0.0543      0.472 white
## 10 KIM          0.0169  0.00282  0.00546  0.943     0.0319      0.943 Asian
## # i 3,796 more rows
```

On this step, we cleaned the dataset removing extra columns

```
# removing extra columns
examiner_race <- examiner_race %>%
  select(surname,race)

data <- data %>%
  left_join(examiner_race, by = c("examiner_name_last" = "surname"))

rm(examiner_race)
rm(examiner_surnames)
gc()
```

```
##            used  (Mb) gc trigger  (Mb)  max used  (Mb)
## Ncells  4578955 244.6    8033454 429.1   6606674 352.9
## Vcells 61664259 470.5  124197188 947.6 123787764 944.5
```

```
library(lubridate) # to work with dates

examiner_dates <- data %>%
  select(examiner_id, filing_date, appl_status_date)

examiner_dates
```

```
## # A tibble: 2,018,477 x 3
##    examiner_id filing_date appl_status_date
##          <dbl> <date>      <chr>
##  1       96082 2000-01-26  30jan2003 00:00:00
##  2       87678 2000-10-11  27sep2010 00:00:00
##  3       63213 2000-05-17  30mar2009 00:00:00
##  4       73788 2001-07-20  07sep2009 00:00:00
##  5       77294 2000-04-10  19apr2001 00:00:00
##  6       68606 2000-04-28  16jul2001 00:00:00
##  7       89557 2004-01-26  15may2017 00:00:00
##  8       97543 2000-06-23  03apr2002 00:00:00
##  9       98714 2000-02-04  27nov2002 00:00:00
## 10       65530 2002-02-20  23mar2009 00:00:00
## # i 2,018,467 more rows
```

```
examiner_dates <- examiner_dates %>%
  mutate(start_date = ymd(filing_date), end_date = as_date(dmy_hms(appl_status_date)))
```

After the cleaning and preprocessing steps, we grouped the data at a examiner level. This would allow us to perform a regression models

```
examiner_dates <- examiner_dates %>%
  group_by(examiner_id) %>%
  summarise(
    earliest_date = min(start_date, na.rm = TRUE),
    latest_date = max(end_date, na.rm = TRUE),
    tenure_days = interval(earliest_date, latest_date) %/% days(1)
    ) %>%
  filter(year(latest_date)<2018)

examiner_dates
```

```
## # A tibble: 5,625 x 4
##    examiner_id earliest_date latest_date tenure_days
##          <dbl> <date>        <date>            <dbl>
##  1       59012 2004-07-28    2015-07-24         4013
##  2       59025 2009-10-26    2017-05-18         2761
##  3       59030 2005-12-12    2017-05-22         4179
##  4       59040 2007-09-11    2017-05-23         3542
##  5       59052 2001-08-21    2007-02-28         2017
##  6       59054 2000-11-10    2016-12-23         5887
##  7       59055 2004-11-02    2007-12-26         1149
##  8       59056 2000-03-24    2017-05-22         6268
##  9       59074 2000-01-31    2017-03-17         6255
## 10       59081 2011-04-21    2017-05-19         2220
## # i 5,615 more rows
```

```
data <- data %>%
  left_join(examiner_dates, by = "examiner_id")

rm(examiner_dates)
gc()
```

```
##            used  (Mb) gc trigger   (Mb)  max used   (Mb)
## Ncells  4585573 244.9   14555648  777.4  14555648  777.4
## Vcells 74026253 564.8  149116625 1137.7 149090804 1137.5
```

```
data
```

```
## # A tibble: 2,018,477 x 26
##    application_number filing_date examiner_name_last examiner_name_first
##    <chr>              <date>      <chr>              <chr>
##  1 08284457           2000-01-26  HOWARD             JACQUELINE
##  2 08413193           2000-10-11  YILDIRIM           BEKIR
##  3 08531853           2000-05-17  HAMILTON           CYNTHIA
##  4 08637752           2001-07-20  MOSHER             MARY
##  5 08682726           2000-04-10  BARR                MICHAEL
##  6 08687412           2000-04-28  GRAY                LINDA
##  7 08716371           2004-01-26  MCMILLIAN           KARA
##  8 08765941           2000-06-23  FORD                VANESSA
##  9 08776818           2000-02-04  STRZELECKA          TERESA
## 10 08809677           2002-02-20  KIM                 SUN
## # i 2,018,467 more rows
## # i 22 more variables: examiner_name_middle <chr>, examiner_id <dbl>,
## #   examiner_art_unit <dbl>, uspc_class <chr>, uspc_subclass <chr>,
## #   patent_number <chr>, patent_issue_date <date>, abandon_date <date>,
## #   disposal_type <chr>, appl_status_code <dbl>, appl_status_date <chr>,
## #   tc <dbl>, gender.x <chr>, race.x <chr>, earliest_date.x <date>,
## #   latest_date.x <date>, tenure_days.x <dbl>, gender.y <chr>, ...
```

```
data <- data %>%
  select(
    application_number,
    filing_date,
    examiner_name_last,
    examiner_name_first,
    examiner_name_middle,
    examiner_id,
    examiner_art_unit,
    uspc_class,
    uspc_subclass,
    patent_number,
    patent_issue_date,
    abandon_date,
    disposal_type,
    appl_status_code,
    appl_status_date,
    tc,
    gender = gender.y,  # Renaming the column to remove the suffix
    race = race.y,      # Renaming the column to remove the suffix
    earliest_date = earliest_date.y,  # Renaming the column to remove the suffix
    latest_date = latest_date.y,  # Renaming the column to remove the suffix
    tenure_days = tenure_days.y   # Renaming the column to remove the suffix
  )
data
```

```
## # A tibble: 2,018,477 x 21
##    application_number filing_date examiner_name_last examiner_name_first
##    <chr>              <date>      <chr>              <chr>
##  1 08284457           2000-01-26  HOWARD              JACQUELINE
##  2 08413193           2000-10-11  YILDIRIM            BEKIR
##  3 08531853           2000-05-17  HAMILTON            CYNTHIA
##  4 08637752           2001-07-20  MOSHER              MARY
##  5 08682726           2000-04-10  BARR                MICHAEL
##  6 08687412           2000-04-28  GRAY                LINDA
##  7 08716371           2004-01-26  MCMILLIAN           KARA
##  8 08765941           2000-06-23  FORD                VANESSA
##  9 08776818           2000-02-04  STRZELECKA          TERESA
## 10 08809677           2002-02-20  KIM                 SUN
## # i 2,018,467 more rows
## # i 17 more variables: examiner_name_middle <chr>, examiner_id <dbl>,
## #   examiner_art_unit <dbl>, uspc_class <chr>, uspc_subclass <chr>,
## #   patent_number <chr>, patent_issue_date <date>, abandon_date <date>,
## #   disposal_type <chr>, appl_status_code <dbl>, appl_status_date <chr>,
## #   tc <dbl>, gender <chr>, race <chr>, earliest_date <date>,
## #   latest_date <date>, tenure_days <dbl>
```

Task 2: Create a panel dataset

```r
library(dplyr)
library(lubridate)
library(zoo)

# Convert dates to quarters
data <- data %>%
  mutate(
    filing_year_quarter = as.yearqtr(filing_date),
    abandon_year_quarter = as.yearqtr(abandon_date),
    issue_year_quarter = as.yearqtr(patent_issue_date)
  )

# Aggregate applications data by quarter
panel_data <- data %>%
  group_by(examiner_id, filing_year_quarter) %>%
  summarise(
    num_new_applications = n_distinct(application_number),
    num_abandoned_applications = sum(disposal_type == "ABN", na.rm = TRUE),
    num_issued_patents = sum(disposal_type == "ISS", na.rm = TRUE),
    num_in_process_applications = sum(disposal_type == "PEND", na.rm = TRUE),
    current_art_unit = first(examiner_art_unit),
    .groups = 'drop'
  )

# Add the count of people and women in each art unit per quarter
art_unit_info <- data %>%
  group_by(filing_year_quarter, examiner_art_unit) %>%
  summarise(
    num_people_in_art_unit = n_distinct(examiner_id),
    num_women_in_art_unit = sum(gender == "female", na.rm = TRUE),
    .groups = 'drop'
  )

# Join the art unit info with the main panel data
panel_data <- panel_data %>%
  left_join(art_unit_info, by = c("filing_year_quarter", "current_art_unit" = "examiner_art_unit"))

# Mark the last five quarters for each examiner
panel_data <- panel_data %>%
  group_by(examiner_id) %>%
  mutate(
    # Get a list of the last five quarters of activity for each examiner
    last_five_quarters = list(tail(sort(unique(filing_year_quarter)), 5))
  ) %>%
  ungroup() %>%
  mutate(
    # Check if the current quarter is in the last five quarters of activity
    separation_indicator = if_else(map_lgl(filing_year_quarter, ~ .x %in% last_five_quarters[[1]]), 1, 
  )


# Detect changes in current_art_unit
panel_data <- panel_data %>%
  group_by(examiner_id) %>%
  mutate(
    # If the current art unit is different from the previous one, it's a move (1), otherwise, it's not
    # For the first row of each examiner (where there is no "previous" art unit), use NA as the default
    AU_move_indicator = if_else(current_art_unit != lag(current_art_unit, default = NA), 1, 0)
  ) %>%
  mutate(
    # Replace NA with 0 - assumes that the first observation is not a move.
    AU_move_indicator = replace_na(AU_move_indicator, 0)
  ) %>%
  ungroup()
```

```
table(panel_data$separation_indicator)
```

```
##
##      0      1
## 175481  15400
```

```
table(panel_data$AU_move_indicator)
```

```
##
##      0      1
## 168875  22006
```

Task 3: Estimate predictors for turnover and mobility

```
# Prepare the data for regression
regression_data <- panel_data %>%
  filter(num_new_applications > 0)

if (!requireNamespace("xgboost", quietly = TRUE)) {
  install.packages("xgboost")
}
library(xgboost)
```

```
## Warning: package 'xgboost' was built under R version 4.2.3

##
## Attaching package: 'xgboost'

## The following object is masked from 'package:dplyr':
##
##     slice
```

```r
library(caret)
```

```
## Warning: package 'caret' was built under R version 4.2.3

## Loading required package: ggplot2

## Warning: package 'ggplot2' was built under R version 4.2.3

## Loading required package: lattice

##
## Attaching package: 'caret'

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

```r
# Convert data to matrix format
X <- as.matrix(regression_data[, c("num_new_applications",
                                    "num_abandoned_applications",
                                    "num_issued_patents",
                                    "num_in_process_applications",
                                    "num_people_in_art_unit",
                                    "num_women_in_art_unit")])
y <- regression_data$AU_move_indicator

set.seed(123)
params <- list(
  objective = "binary:logistic",
  eval_metric = "auc",  # You can also use 'recall' as the evaluation metric
  scale_pos_weight = 7.67)


# Train the XGBoost model
xgb_model <- xgboost(data = X,
                     label = y,
                     objective = params$objective,
                     eval_metric = params$eval_metric,
                     scale_pos_weight = params$scale_pos_weight,
                     nrounds = 100)
```

```
## [1]  train-auc:0.758162
## [2]  train-auc:0.765967
## [3]  train-auc:0.768144
## [4]  train-auc:0.772081
## [5]  train-auc:0.773715
## [6]  train-auc:0.775339
## [7]  train-auc:0.776616
## [8]  train-auc:0.777500
## [9]  train-auc:0.778648
## [10] train-auc:0.779267
## [11] train-auc:0.779877
## [12] train-auc:0.780455
## [13] train-auc:0.780928
## [14] train-auc:0.781428
## [15] train-auc:0.781877
## [16] train-auc:0.782573
## [17] train-auc:0.782857
## [18] train-auc:0.783227
## [19] train-auc:0.783863
## [20] train-auc:0.784117
## [21] train-auc:0.784829
## [22] train-auc:0.785486
## [23] train-auc:0.785547
## [24] train-auc:0.785828
## [25] train-auc:0.786395
## [26] train-auc:0.786522
## [27] train-auc:0.786810
## [28] train-auc:0.787312
## [29] train-auc:0.787720
## [30] train-auc:0.788475
## [31] train-auc:0.788664
## [32] train-auc:0.789086
## [33] train-auc:0.789864
## [34] train-auc:0.790139
## [35] train-auc:0.790590
## [36] train-auc:0.791037
## [37] train-auc:0.791374
## [38] train-auc:0.791664
## [39] train-auc:0.791904
## [40] train-auc:0.792278
## [41] train-auc:0.792395
## [42] train-auc:0.792677
## [43] train-auc:0.792777
## [44] train-auc:0.793379
## [45] train-auc:0.793945
## [46] train-auc:0.794035
## [47] train-auc:0.794385
## [48] train-auc:0.794721
## [49] train-auc:0.795072
## [50] train-auc:0.795653
## [51] train-auc:0.795805
## [52] train-auc:0.796199
## [53] train-auc:0.796717
## [54] train-auc:0.796993
## [55] train-auc:0.797624
## [56] train-auc:0.797924
## [57] train-auc:0.798329
## [58] train-auc:0.798380
## [59] train-auc:0.798498
## [60] train-auc:0.799022
## [61] train-auc:0.799323
## [62] train-auc:0.799585
## [63] train-auc:0.799773
## [64] train-auc:0.800238
## [65] train-auc:0.800676
## [66] train-auc:0.800973
## [67] train-auc:0.801407
## [68] train-auc:0.801890
## [69] train-auc:0.802418
## [70] train-auc:0.802789
## [71] train-auc:0.802959
## [72] train-auc:0.803471
## [73] train-auc:0.804002
## [74] train-auc:0.804161
## [75] train-auc:0.804273
## [76] train-auc:0.804771
## [77] train-auc:0.804847
## [78] train-auc:0.804864
## [79] train-auc:0.804877
## [80] train-auc:0.805340
## [81] train-auc:0.805511
## [82] train-auc:0.805588
## [83] train-auc:0.805943
## [84] train-auc:0.806009
## [85] train-auc:0.806410
## [86] train-auc:0.806984
## [87] train-auc:0.807390
## [88] train-auc:0.807826
## [89] train-auc:0.808117
## [90] train-auc:0.808308
## [91] train-auc:0.808959
## [92] train-auc:0.809183
## [93] train-auc:0.809271
## [94] train-auc:0.809560
## [95] train-auc:0.809608
## [96] train-auc:0.809844
## [97] train-auc:0.810259
## [98] train-auc:0.810435
## [99] train-auc:0.810577
## [100]    train-auc:0.810845
```

```r
# Make predictions
predictions <- predict(xgb_model, X, type = "response")

# Convert probabilities to binary predictions
binary_predictions <- ifelse(predictions > 0.5, 1, 0)

# Evaluate the classifier using a classification report
```

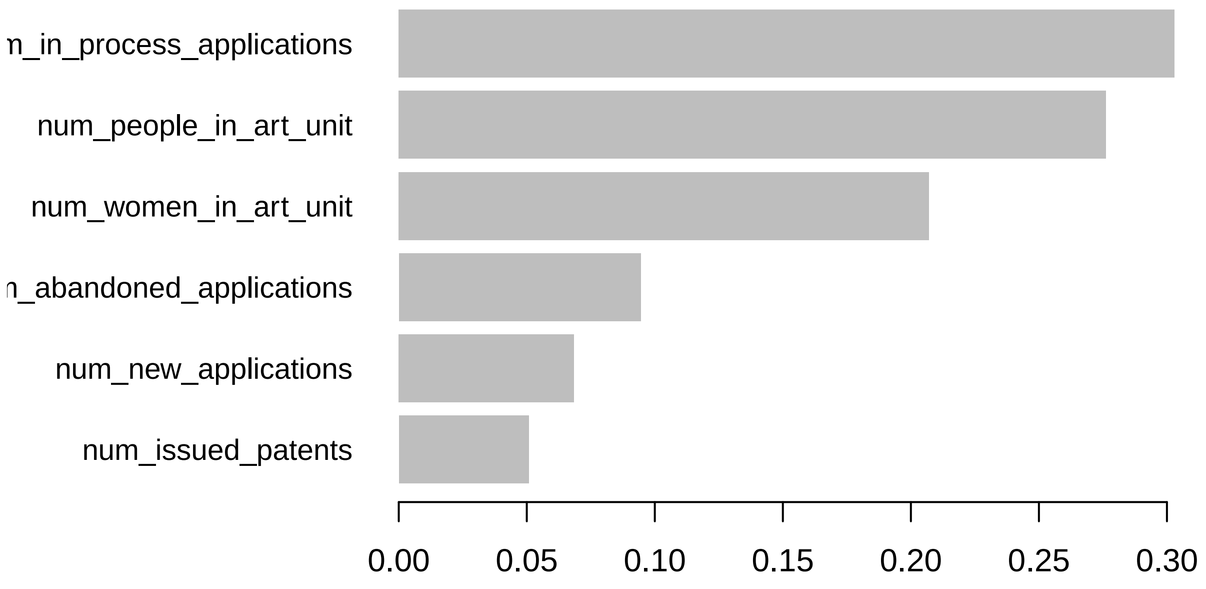
```
confusion_matrix <- confusionMatrix(data = as.factor(binary_predictions),
                                    reference = as.factor(y))
print(confusion_matrix)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction      0      1
##          0 121063   6005
##          1  47812  16001
##
##                Accuracy : 0.7181
##                  95% CI : (0.716, 0.7201)
##     No Information Rate : 0.8847
##     P-Value [Acc > NIR] : 1
##
##                   Kappa : 0.2431
##
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 0.7169
##             Specificity : 0.7271
##          Pos Pred Value : 0.9527
##          Neg Pred Value : 0.2507
##              Prevalence : 0.8847
##          Detection Rate : 0.6342
##    Detection Prevalence : 0.6657
##       Balanced Accuracy : 0.7220
##
##        'Positive' Class : 0
##
```

```
# Full classification report including precision, recall, accuracy


# Calculate recall (sensitivity)
recall <- sensitivity(factor(binary_predictions), factor(y), positive = "1")
cat("Recall:", recall, "\n")
```

```
## Recall: 0.7271199
```

```
if (!requireNamespace("xgboost", quietly = TRUE)) {
  install.packages("xgboost")
}
library(xgboost)


# Extract feature importance
importance <- xgb.importance(model = xgb_model)

# Plot feature importance
xgb.plot.importance(importance_matrix = importance)
```

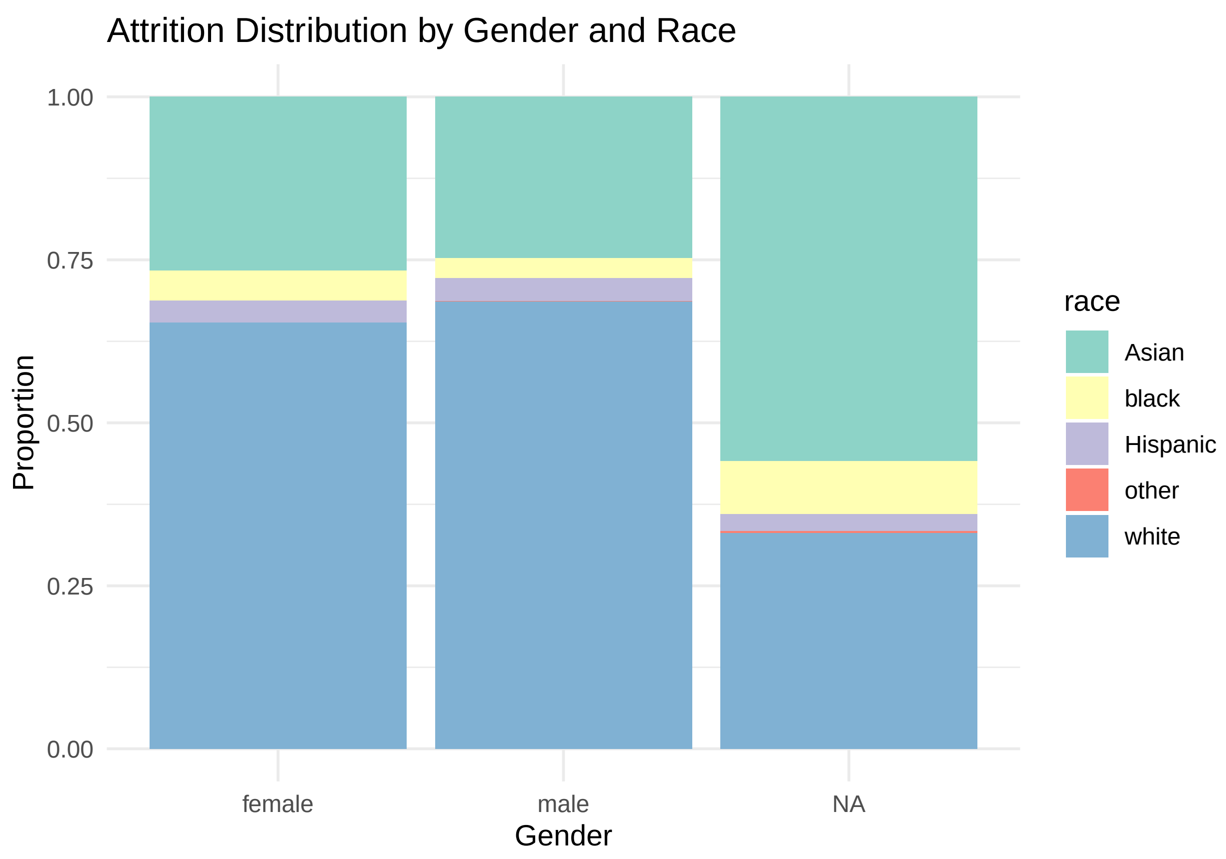


Descriptive Analysis: Initially, perform a descriptive analysis to understand the distribution of attrition across different demographic groups and the general characteristics of examiners who leave vs. those who stay.

```
enhanced_panel_data <- panel_data %>%
  left_join(data %>%
              select(examiner_id, gender, race, examiner_art_unit, filing_year_quarter) %>%
              distinct(),
            by = c("examiner_id", "filing_year_quarter"))
```

Plots

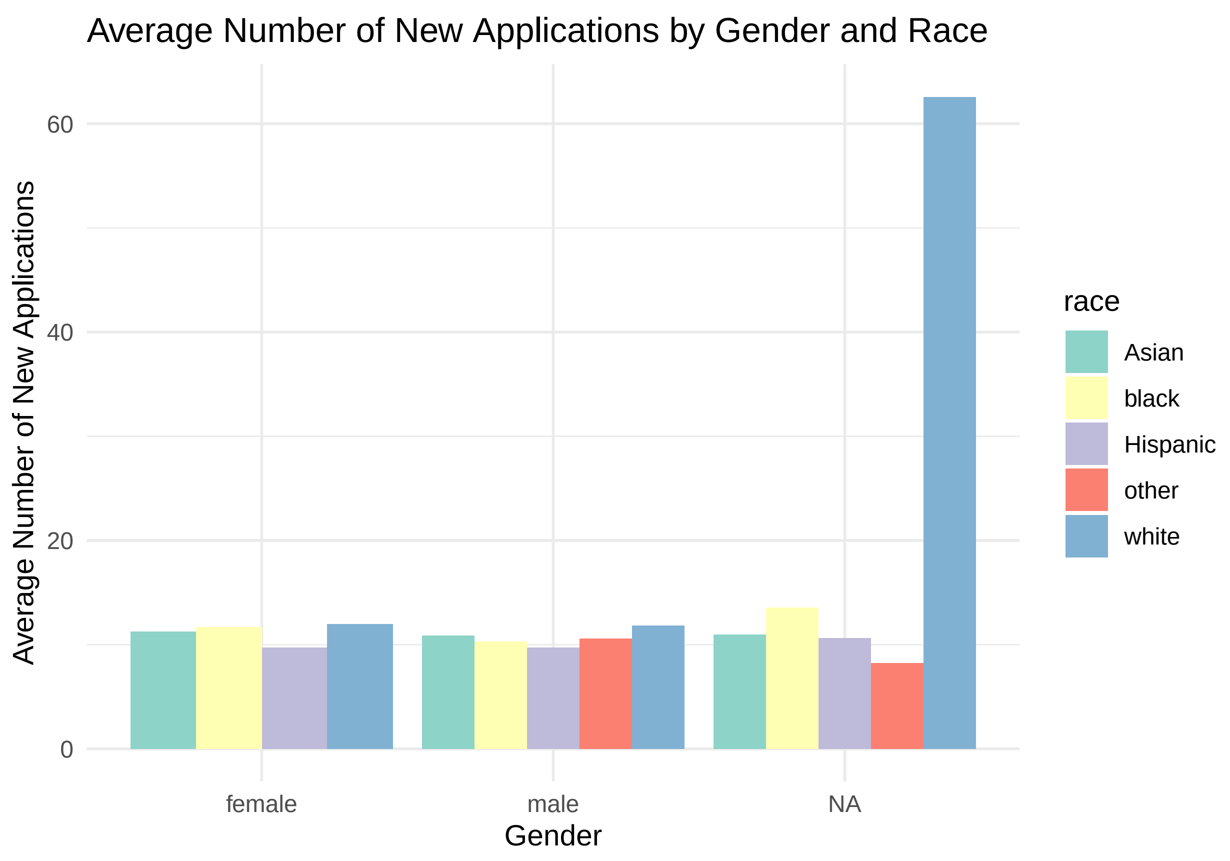
```
#Plot 1: Attrition Rates by Gender and Race
#This plot will help visualize attrition rates across different genders and races, providing insights i

library(ggplot2)

ggplot(enhanced_panel_data, aes(x = gender, fill = race)) +
  geom_bar(position = "fill") +
  labs(title = "Attrition Distribution by Gender and Race", x = "Gender", y = "Proportion") +
  scale_fill_brewer(palette = "Set3") +
  theme_minimal()
```



```r
# plot visualizes the average number of new patent applications handled by USPTO examiners, broken down

library(ggplot2)

# Calculating the average number of new applications by gender and race
avg_new_applications <- enhanced_panel_data %>%
  group_by(gender, race) %>%
  summarise(Avg_Num_New_Applications = mean(num_new_applications, na.rm = TRUE)) %>%
  ungroup()
```

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

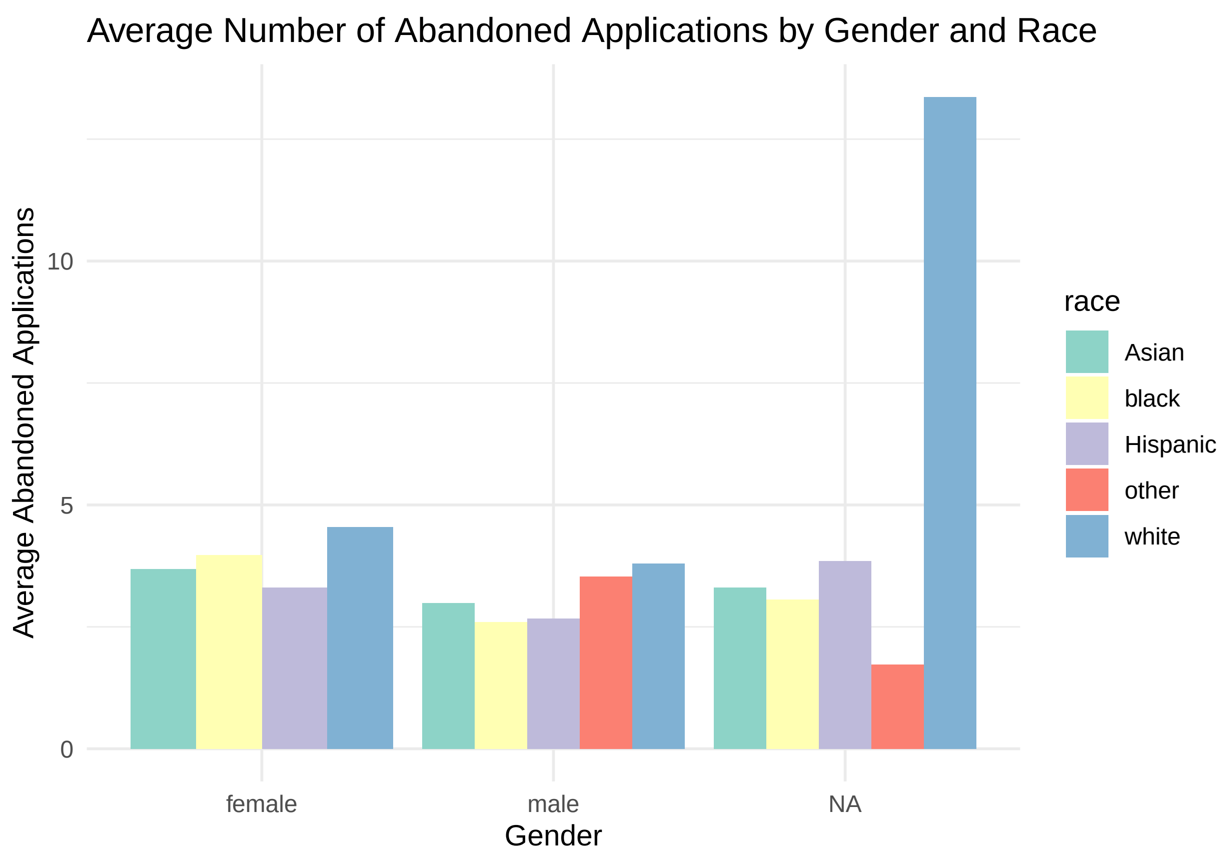
```r
# Plotting
ggplot(avg_new_applications, aes(x = gender, y = Avg_Num_New_Applications, fill = race)) +
  geom_bar(stat = "identity", position = "dodge") +
  labs(title = "Average Number of New Applications by Gender and Race", x = "Gender", y = "Average Numbe
  scale_fill_brewer(palette = "Set3") +
  theme_minimal()
```



```r
# Average Number of Abandoned Applications by Gender and Race

avg_abandoned_applications <- enhanced_panel_data %>%
  group_by(gender, race) %>%
  summarise(Avg_Abandoned_Applications = mean(num_abandoned_applications, na.rm = TRUE)) %>%
  ungroup()
```

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

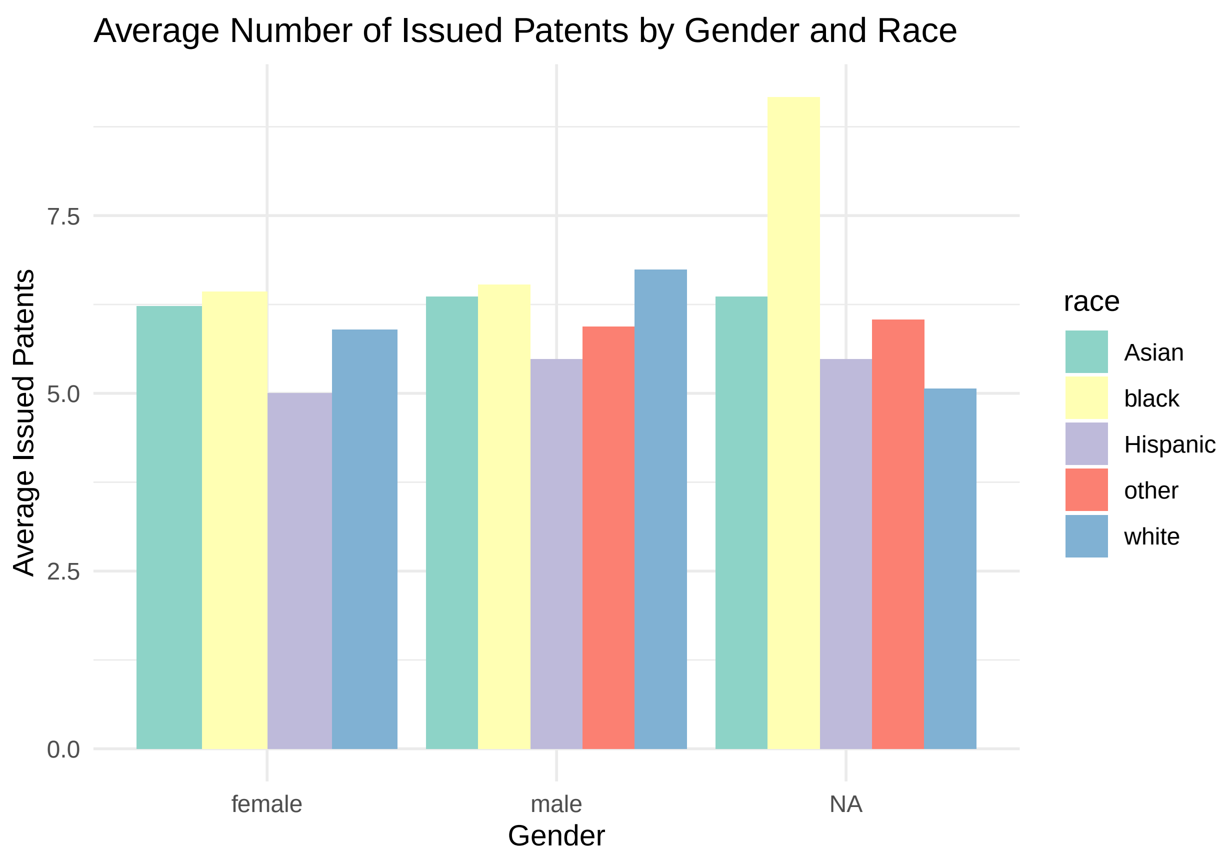
```r
ggplot(avg_abandoned_applications, aes(x = gender, y = Avg_Abandoned_Applications, fill = race)) +
  geom_bar(stat = "identity", position = "dodge") +
  labs(title = "Average Number of Abandoned Applications by Gender and Race", x = "Gender", y = "Averag
  scale_fill_brewer(palette = "Set3") +
  theme_minimal()
```



```
#Average Number of Issued Patents by Gender and Race

avg_issued_patents <- enhanced_panel_data %>%
  group_by(gender, race) %>%
  summarise(Avg_Issued_Patents = mean(num_issued_patents, na.rm = TRUE)) %>%
  ungroup()
```

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

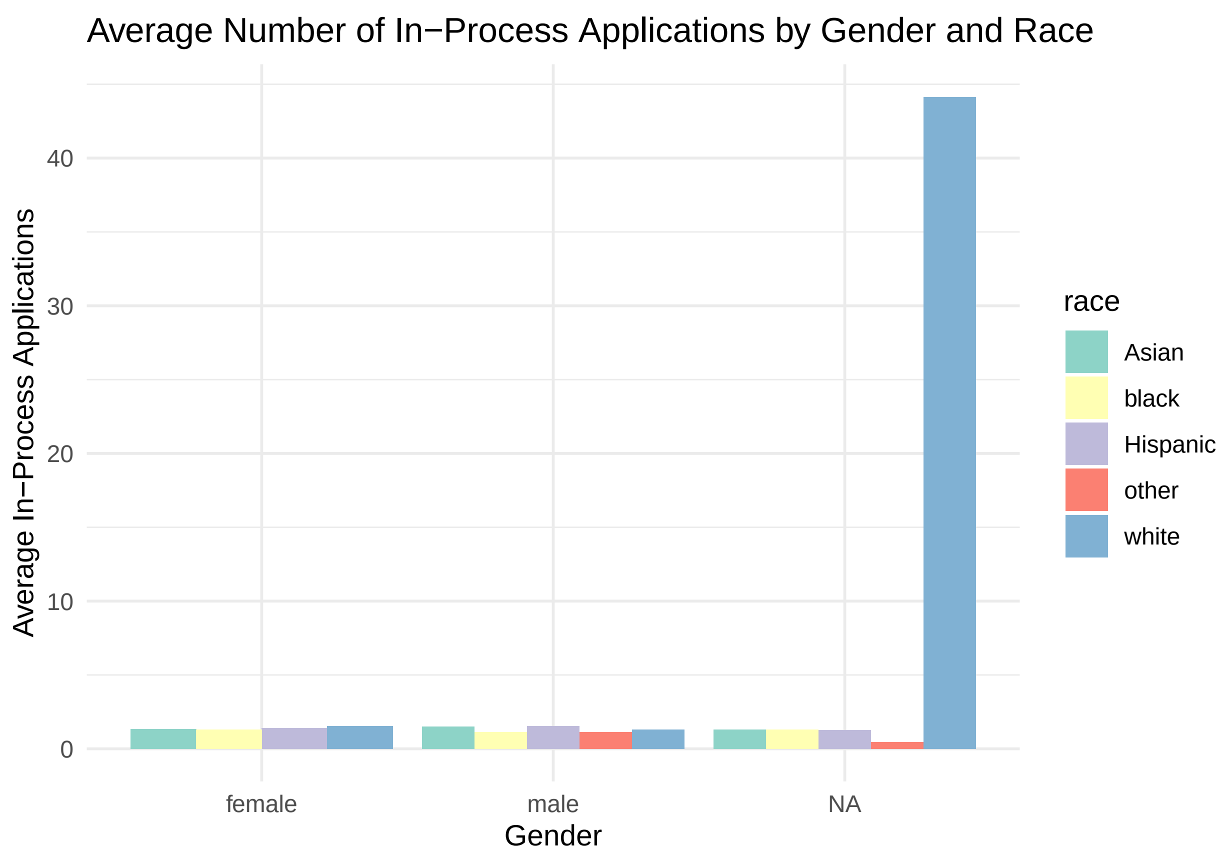
```
ggplot(avg_issued_patents, aes(x = gender, y = Avg_Issued_Patents, fill = race)) +
  geom_bar(stat = "identity", position = "dodge") +
  labs(title = "Average Number of Issued Patents by Gender and Race", x = "Gender", y = "Average Issued
  scale_fill_brewer(palette = "Set3") +
  theme_minimal()
```



```r
#Average Number of In-Process Applications by Gender and Race
avg_in_process_applications <- enhanced_panel_data %>%
  group_by(gender, race) %>%
  summarise(Avg_In_Process_Applications = mean(num_in_process_applications, na.rm = TRUE)) %>%
  ungroup()
```

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

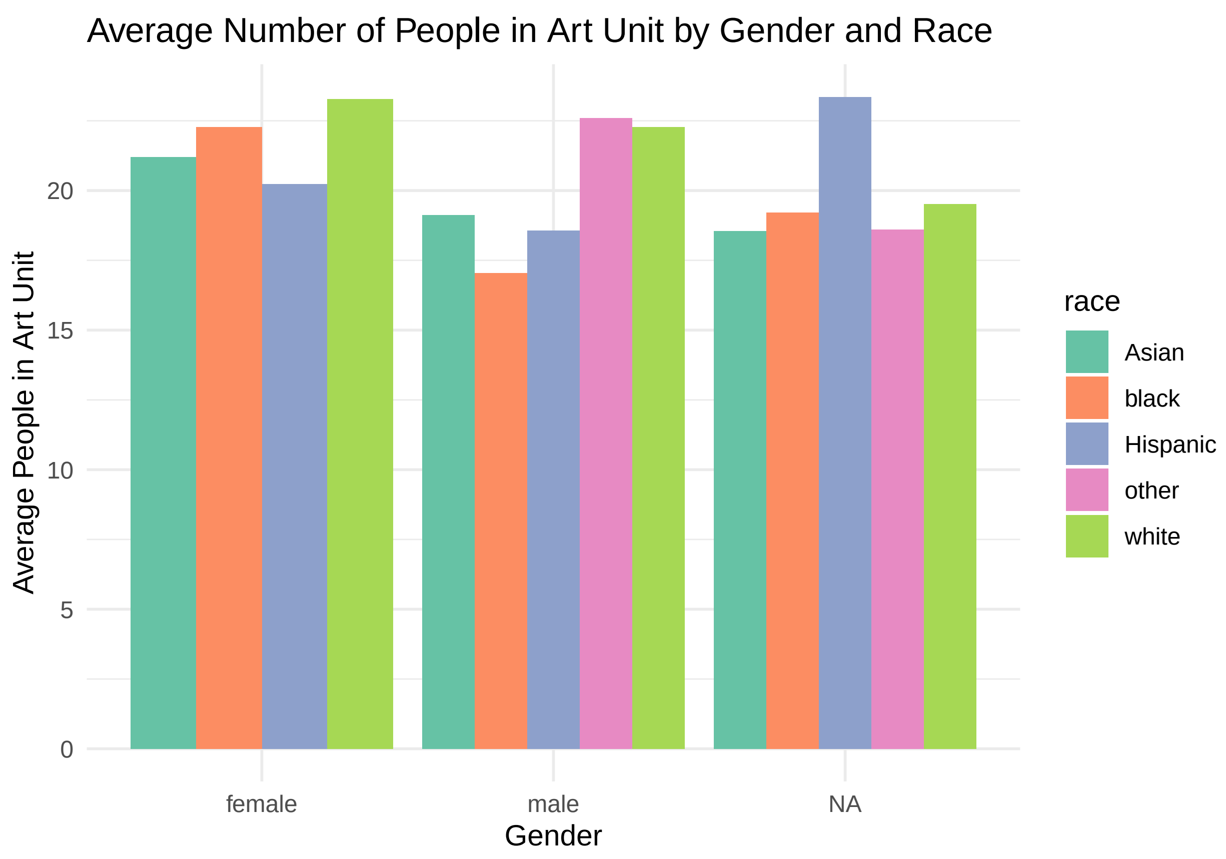
```r
ggplot(avg_in_process_applications, aes(x = gender, y = Avg_In_Process_Applications, fill = race)) +
  geom_bar(stat = "identity", position = "dodge") +
  labs(title = "Average Number of In-Process Applications by Gender and Race", x = "Gender", y = "Averag
  scale_fill_brewer(palette = "Set3") +
  theme_minimal()
```



```r
#Average Number of People in an Art Unit by Gender and Race
avg_people_in_art_unit <- enhanced_panel_data %>%
  group_by(gender, race) %>%
  summarise(Avg_People_in_Art_Unit = mean(num_people_in_art_unit, na.rm = TRUE)) %>%
  ungroup()
```

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

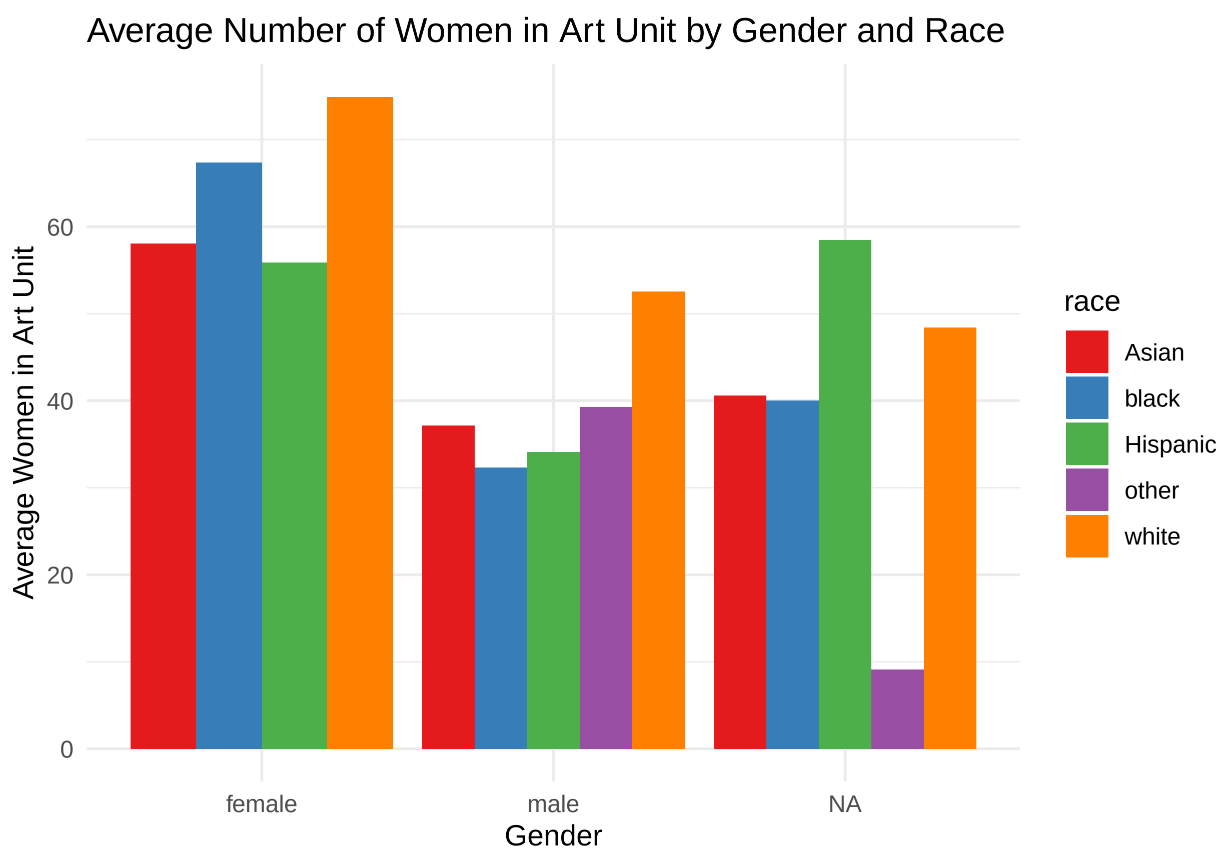
```r
ggplot(avg_people_in_art_unit, aes(x = gender, y = Avg_People_in_Art_Unit, fill = race)) +
  geom_bar(stat = "identity", position = "dodge") +
  labs(title = "Average Number of People in Art Unit by Gender and Race", x = "Gender", y = "Average Pe
  scale_fill_brewer(palette = "Set2") +
  theme_minimal()
```



```r
#Average Number of Women in an Art Unit by Gender and Race
avg_women_in_art_unit <- enhanced_panel_data %>%
  group_by(gender, race) %>%
  summarise(Avg_Women_in_Art_Unit = mean(num_women_in_art_unit, na.rm = TRUE)) %>%
  ungroup()
```

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

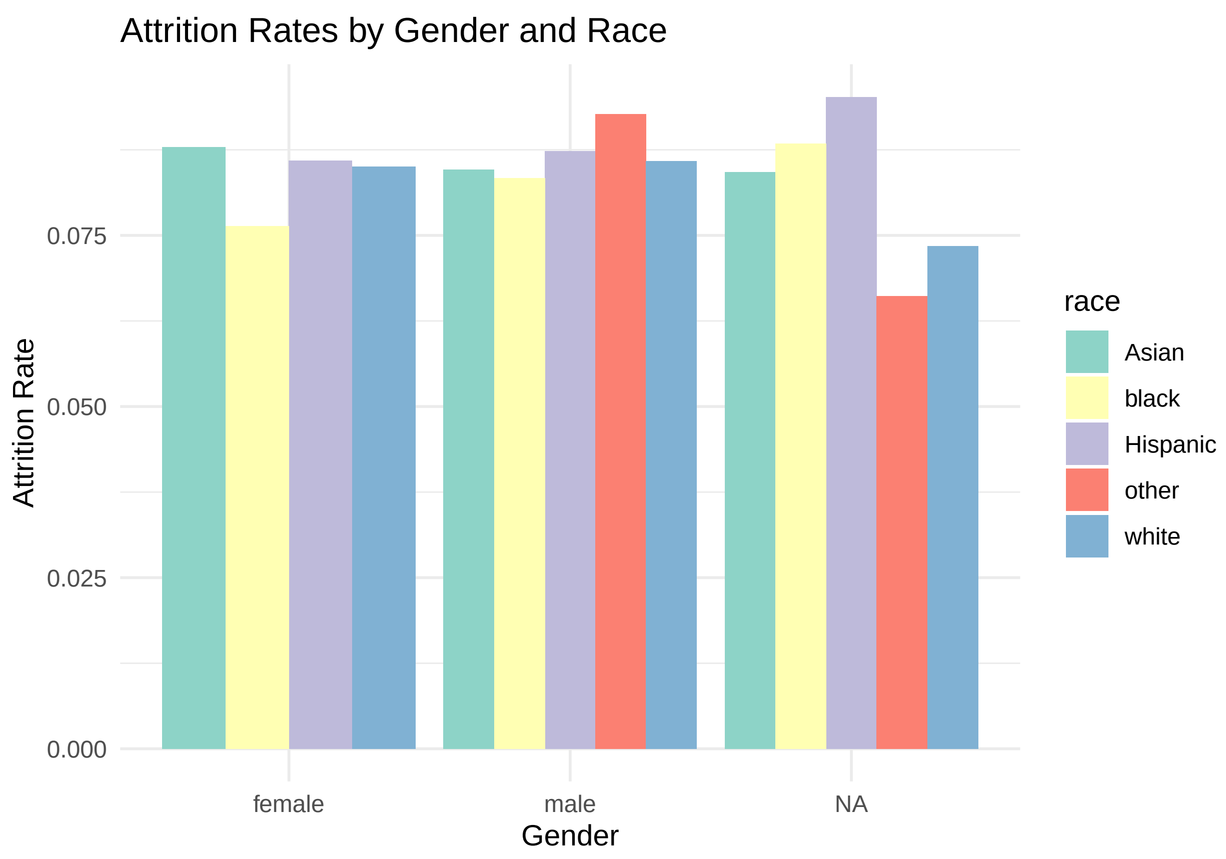
```r
ggplot(avg_women_in_art_unit, aes(x = gender, y = Avg_Women_in_Art_Unit, fill = race)) +
  geom_bar(stat = "identity", position = "dodge") +
  labs(title = "Average Number of Women in Art Unit by Gender and Race", x = "Gender", y = "Average Wome
  scale_fill_brewer(palette = "Set1") +
  theme_minimal()
```



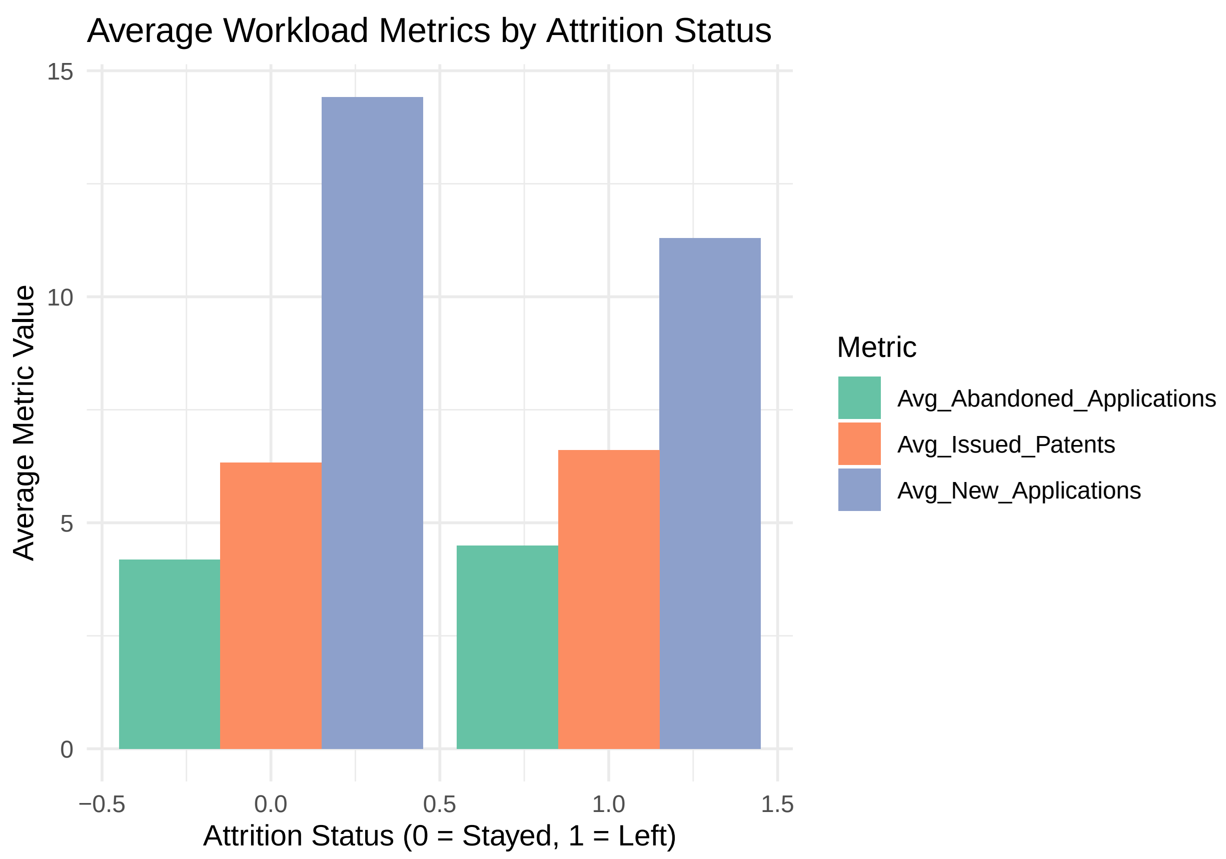
```r
#Attrition Rates by Demographic Group

attrition_rates <- enhanced_panel_data %>%
  group_by(gender, race) %>%
  summarise(Attrition_Count = sum(separation_indicator == 1, na.rm = TRUE),
            Total_Count = n(),
            Attrition_Rate = Attrition_Count / Total_Count) %>%
  ungroup()
```

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

```r
ggplot(attrition_rates, aes(x = gender, y = Attrition_Rate, fill = race)) +
  geom_bar(stat = "identity", position = "dodge") +
  labs(title = "Attrition Rates by Gender and Race", x = "Gender", y = "Attrition Rate") +
  scale_fill_brewer(palette = "Set3") +
  theme_minimal()
```



```r
# Average Workload Metrics by Attrition Status

library(ggplot2)
library(dplyr)

# Calculate the average workload metrics by attrition status
workload_by_attrition <- enhanced_panel_data %>%
  group_by(separation_indicator) %>%
  summarise(
    Avg_New_Applications = mean(num_new_applications, na.rm = TRUE),
    Avg_Abandoned_Applications = mean(num_abandoned_applications, na.rm = TRUE),
    Avg_Issued_Patents = mean(num_issued_patents, na.rm = TRUE)
  ) %>%
  gather(key = "Metric", value = "Average", -separation_indicator)

# Plotting
ggplot(workload_by_attrition, aes(x = separation_indicator, y = Average, fill = Metric)) +
  geom_bar(stat = "identity", position = "dodge") +
  scale_fill_brewer(palette = "Set2") +
  labs(title = "Average Workload Metrics by Attrition Status",
       x = "Attrition Status (0 = Stayed, 1 = Left)",
       y = "Average Metric Value") +
  theme_minimal()
```



Modelling Analysis

```
library(xgboost)
library(caret)
library(dplyr)
library(tidyr) # for pivot_longer and pivot_wider

enhanced_panel_data$gender <- as.factor(enhanced_panel_data$gender)
enhanced_panel_data$race <- as.factor(enhanced_panel_data$race)

# Prepare the dataset for XGBoost by removing unwanted columns
data <- enhanced_panel_data %>%
  select(-examiner_id, -filing_year_quarter, -current_art_unit, -examiner_art_unit, -last_five_quarters
  na.omit()  # Remove rows with NA values

# Create a model matrix for the features, automatically one-hot encoding factor variables
# Note: The '-1' removes the intercept term which is not needed for XGBoost
features <- model.matrix(~ . -1 -separation_indicator, data = data)
labels <- data$separation_indicator

# Split the data into training and testing sets
set.seed(123) # For reproducibility
index <- createDataPartition(labels, p = .8, list = FALSE)
train_features <- features[index,]
test_features <- features[-index,]
train_labels <- labels[index]
```

```r
test_labels <- labels[-index]

# Prepare matrices for xgboost
dtrain <- xgb.DMatrix(data = train_features, label = train_labels)
dtest <- xgb.DMatrix(data = test_features, label = test_labels)


# Train the XGBoost model
set.seed(123) # for reproducibility
params <- list(
  objective = "binary:logistic",
  eval_metric = "auc",  # You can also use 'recall' as the evaluation metric
  scale_pos_weight = sum(train_labels == 0) / sum(train_labels == 1))  # Adjust based on class imbalanc


# Train the XGBoost model
xgb_model <- xgb.train(params = params, data = dtrain, nrounds = 100, verbose = 0)

# Evaluate the model
xgb_pred <- predict(xgb_model, dtest)
xgb_pred_label <- ifelse(xgb_pred > 0.5, 1, 0)
confusion_matrix <- table(Predicted = xgb_pred_label, Actual = test_labels)
confusion_matrix
```

```
##          Actual
## Predicted     0     1
##         0 26238   543
##         1 12020  3089
```

```r
# Full classification report including precision, recall, accuracy
precision <- posPredValue(factor(xgb_pred_label), factor(test_labels), positive = "1")
recall <- sensitivity(factor(xgb_pred_label), factor(test_labels), positive = "1")
cat("Recall:", recall, "\n")
```

```
## Recall: 0.8504956
```

```r
cat("Precision:", precision, "\n")
```

```
## Precision: 0.2044477
```

```r
# Calculate accuracy
accuracy <- sum(diag(confusion_matrix)) / sum(confusion_matrix)
cat("Accuracy:", accuracy, "\n")
```

```
## Accuracy: 0.7000955
```

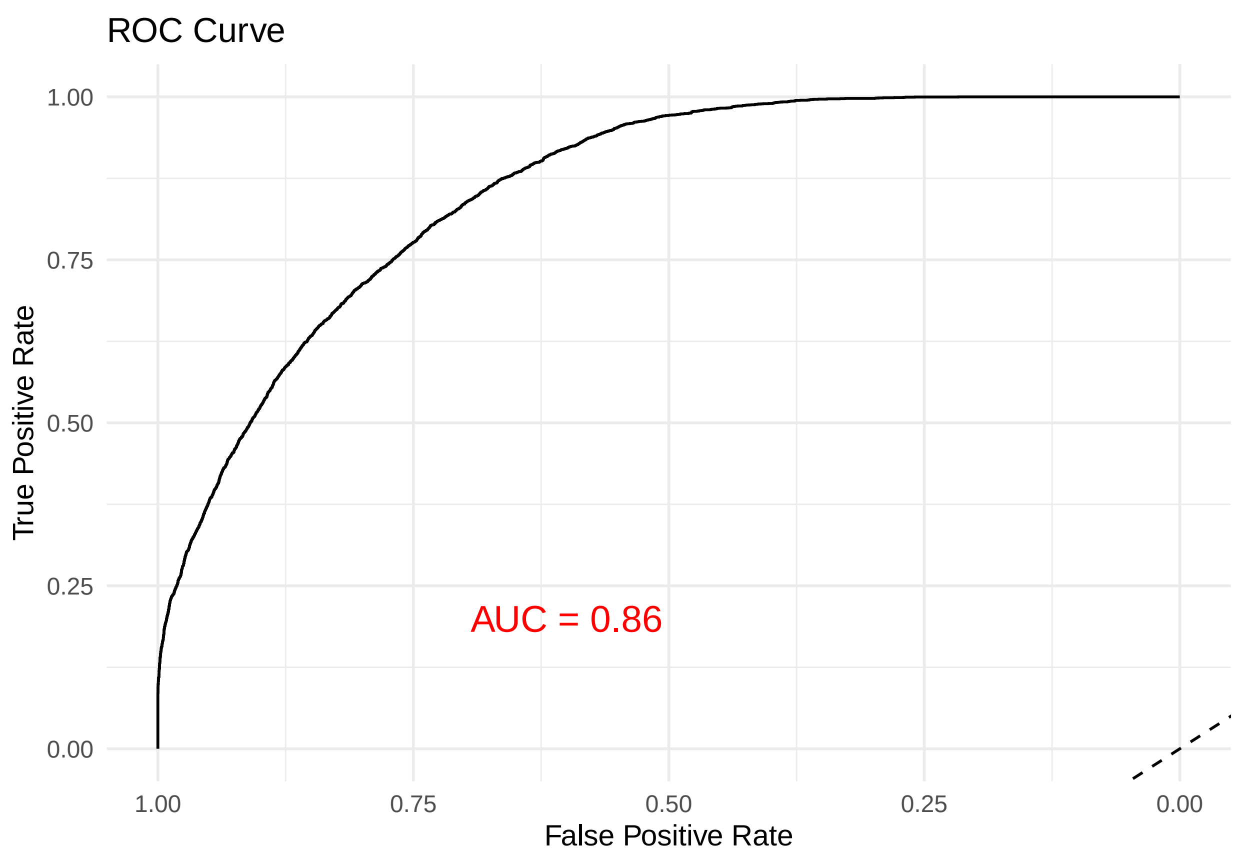
```r
# Calculate the ROC curve
# Ensure the pROC package is installed and loaded
if (!requireNamespace("pROC", quietly = TRUE)) {
  install.packages("pROC")
}
library(pROC)
```

```
## Warning: package 'pROC' was built under R version 4.2.3

## Type 'citation("pROC")' for a citation.

##
## Attaching package: 'pROC'

## The following objects are masked from 'package:stats':
##
##     cov, smooth, var
```

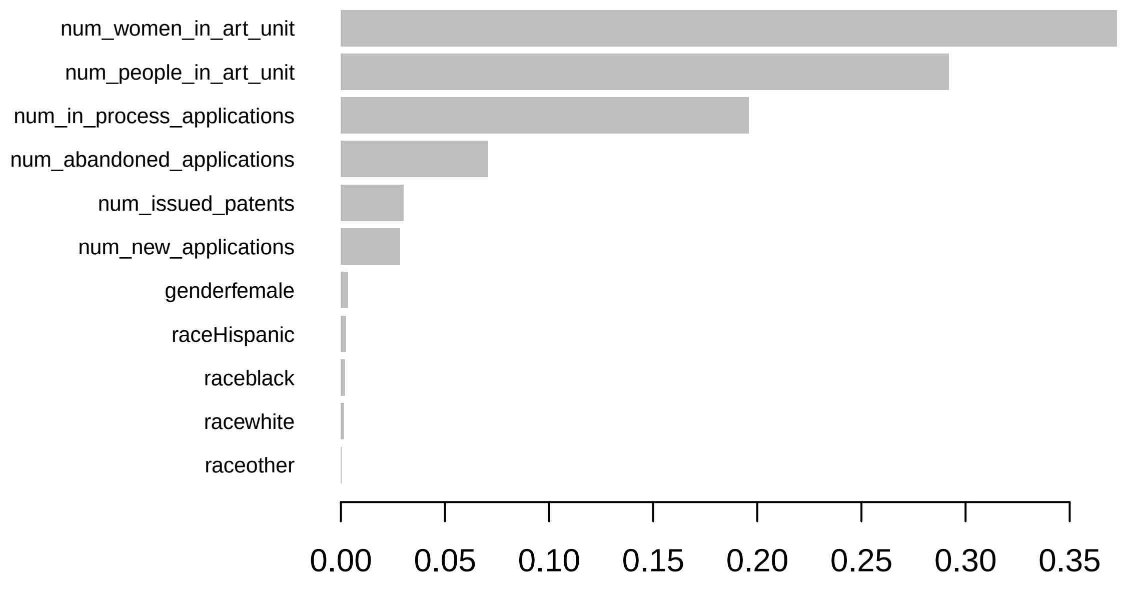
```
# Calculate the ROC curve and AUC
roc_obj <- roc(response = test_labels, predictor = as.numeric(xgb_pred))
```

```
## Setting levels: control = 0, case = 1

## Setting direction: controls < cases
```

```
roc_plot <- ggroc(roc_obj)

# Calculate AUC value
auc_value <- auc(roc_obj)

# Plotting the ROC curve with AUC value annotated on the plot
roc_plot_with_auc <- roc_plot +
  geom_abline(linetype = 'dashed') +
  labs(title = 'ROC Curve',
       x = 'False Positive Rate',
       y = 'True Positive Rate') +
  theme_minimal() +
  annotate("text", x = 0.6, y = 0.2, label = paste("AUC =", round(auc_value, 2)), color = "red", size =

print(roc_plot_with_auc)
```



```r
# Feature importance
importance_matrix <- xgb.importance(model = xgb_model)


# Plot Feature Importance
xgb.plot.importance(importance_matrix)
```



Calculating effect

```r
# Install 'grf' package if not already installed
if (!requireNamespace("grf", quietly = TRUE)) {
  install.packages("grf")
}

# Load required libraries
library(grf)
```

```
## Warning: package 'grf' was built under R version 4.2.3
```

```r
# Filter data to keep only rows where gender is male or female
data <- data[data$gender %in% c("male", "female"), ]
data$gender <- factor(data$gender)

# Convert gender to binary (0 for male, 1 for female)
data$gender_binary <- as.integer(data$gender == "female")

# Fit the uplift classifier
uplift_model <- causal_forest(
  Y = data$separation_indicator,
  W = data$gender_binary,  # Treatment variable (0 for male, 1 for female)
  X = data[, c("num_new_applications",
               "num_abandoned_applications",
               "num_issued_patents",
                "num_in_process_applications",
                "num_people_in_art_unit",
                "num_women_in_art_unit"
                )]
)

# Calculate ATE
ate_estimate <- predict(uplift_model, estimate.variance = TRUE)$predictions
```

```r
# Print ITE
print(ate_estimate[1:10])
```

```
##  [1] -0.103886032 -0.014578934 -0.028441583 -0.010356551 -0.007019132
##  [6] -0.009023220  0.007741406  0.006980482  0.007510394 -0.024356226
```

```r
# Calculate ATE
ate_estimate <- mean(ate_estimate)

# Print ATE estimate
print(ate_estimate)
```

```
## [1] -0.001849595
```

A negative ATE suggests that, on average, being in the treated group (female) leads to a decrease in the outcome variable compared to being in the control group (being male). This could imply that, on average, being female is associated with a lower likelihood of separation