# Exercise 3

2024-03-28

The first few sections of this markdown document reiterates the processing steps highlighted in Exercise 2 with some improvments to the code. For details on Exercise 3, skip to page 8 for the regression analysis.

# Initialisation of libraries and dataset

## Import Libraries

```
knitr::opts_chunk$set(tidy.opts=list(width.cutoff=60), format='latex', echo=TRUE)
library(tidyverse)
```

```
## ── Attaching core tidyverse packages ──────────────────────── tidyverse 2.0.0 ──
## ✔ dplyr     1.1.4     ✔ readr     2.1.4
## ✔ forcats   1.0.0     ✔ stringr   1.5.1
## ✔ ggplot2   3.5.0     ✔ tibble    3.2.1
## ✔ lubridate 1.9.3     ✔ tidyr     1.3.1
## ✔ purrr     1.0.2
## ── Conflicts ────────────────────────────────────────── tidyverse_conflicts() ──
## ✖ dplyr::filter() masks stats::filter()
## ✖ dplyr::lag()    masks stats::lag()
## ℹ Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflic
ts to become errors
```

```
library(lubridate)
library(arrow)
```

```
## Warning: package 'arrow' was built under R version 4.3.3
```

```
##
## Attaching package: 'arrow'
##
## The following object is masked from 'package:lubridate':
##
##     duration
##
## The following object is masked from 'package:utils':
##
##     timestamp
```

# Import Dataset

```
data_path = "/Users/lauray/Documents/GitHub/2024-ona-assignments/Exercise 3/672_proje
ct_data/app_data_sample.parquet" # change this to your path
applications = arrow::read_parquet(data_path)
```

# Processing of dataset to include gender and race for examiners

## Adding gender.y to dataset based on surnames library

```
library(gender)

# get a list of first names without repetitions
examiner_names = applications %>%
  distinct(examiner_name_first)


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
  )

examiner_names_gender
```

| examiner_name_first<br><chr> | gender<br><chr> | proportion_female<br><dbl> |
|---|---|---|
| AARON | male | 0.0082 |
| ABDEL | male | 0.0000 |
| ABDOU | male | 0.0000 |
| ABDUL | male | 0.0000 |
| ABDULHAKIM | male | 0.0000 |
| ABDULLAH | male | 0.0000 |
| ABDULLAHI | male | 0.0000 |

| | | |
|---|---|---|
| ABIGAIL | female | 0.9982 |
| ABIMBOLA | female | 0.9436 |
| ABRAHAM | male | 0.0031 |

1-10 of 1,822 rows Previous **1** 2 3 4 5 6 … 183 Next

```
gc()
```

```
##            used  (Mb) gc trigger  (Mb) limit (Mb) max used  (Mb)
## Ncells  1389649  74.3    2451032 130.9         NA  1655396  88.5
## Vcells 16150573 123.3   31151144 237.7      16384 31057111 237.0
```

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

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

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

```
##            used  (Mb) gc trigger  (Mb) limit (Mb) max used  (Mb)
## Ncells  4511671 241.0    8112600 433.3         NA  4531210 242.0
## Vcells 49508733 377.8   92999694 709.6      16384 79823558 609.1
```

# Adding race.y to dataset using surnames library

```
library(wru)
```

```
##
## Please cite as:
##
## Khanna K, Bertelsen B, Olivella S, Rosenman E, Rossell Hayes A, Imai K
## (2024). _wru: Who are You? Bayesian Prediction of Racial Category Using
## Surname, First Name, Middle Name, and Geolocation_. R package version
## 3.0.1, <https://CRAN.R-project.org/package=wru>.
##
## Note that wru 2.0.0 uses 2020 census data by default.
## Use the argument `year = "2010"`, to replicate analyses produced with earlier pack
age versions.
```

```
examiner_surnames = applications %>%
  select(surname = examiner_name_last) %>%
  distinct()


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

```
## Predicting race for 2020
```

```
## 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 = 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 = examiner_race %>%
  select(surname,race)

applications = applications %>%
  left_join(examiner_race, by = c("examiner_name_last" = "surname"))

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

```
##            used  (Mb) gc trigger  (Mb) limit (Mb) max used  (Mb)
## Ncells  4703636 251.3    8112600 433.3         NA  6857319 366.3
## Vcells 51880295 395.9   92999694 709.6      16384 91953627 701.6
```

# Adding dates-related data to calculate tenure days

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

examiner_dates = applications %>%
  select(examiner_id, filing_date, appl_status_date)
examiner_dates = examiner_dates %>%
  mutate(start_date = ymd(filing_date), end_date = as_date(dmy_hms(appl_status_date))
)

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)
applications = applications %>%
  left_join(examiner_dates, by = "examiner_id")

rm(examiner_dates)
gc()
```

```
##            used  (Mb) gc trigger  (Mb) limit (Mb)  max used  (Mb)
## Ncells  4712246 251.7    8112600 433.3         NA   8112600 433.3
## Vcells 57955761 442.2  111679632 852.1      16384 111382473 849.8
```

# Creating panel data

# Cleaning noisy data

```
distinct_dataset = applications %>%
  select(examiner_art_unit, examiner_id, gender, race, tenure_days) %>%
  distinct()
distinct_dataset = distinct_dataset %>%
  mutate(first_three_digits = str_sub(examiner_art_unit, 1, 3))

filtered_df = distinct_dataset %>%
  filter(str_sub(examiner_art_unit, 1, 3) %in% c("161", "162"))
```

```r
# Summary statistics for tenure
tenure_summary = filtered_df %>%
  group_by(workgroup = str_sub(examiner_art_unit, 1, 3)) %>%
  summarise(
    count = n(),
    mean_tenure = mean(tenure_days, na.rm = TRUE),
    sd_tenure = sd(tenure_days, na.rm = TRUE),
    min_tenure = min(tenure_days, na.rm = TRUE),
    max_tenure = max(tenure_days, na.rm = TRUE)
  )

print(tenure_summary)
```

```
## # A tibble: 2 × 6
##   workgroup count mean_tenure sd_tenure min_tenure max_tenure
##   <chr>     <int>       <dbl>     <dbl>      <dbl>      <dbl>
## 1 161         362       5185.     1474.        330       6350
## 2 162         339       5397.     1182.        614       6518
```

```r
# Frequency tables for gender and race
gender_table = table(filtered_df$first_three_digits, filtered_df$gender)
race_table = table(filtered_df$first_three_digits, filtered_df$race)

print(gender_table)
```

```
##
##       female male
##   161    146  157
##   162    112  149
```

```r
print(race_table)
```

```
##
##       Asian black Hispanic white
##   161    70    11        7   274
##   162    72    18       15   234
```

```r
library(dplyr)
library(stringr)

distinct_dataset = distinct_dataset %>%
  mutate(group = case_when(
    str_sub(examiner_art_unit, 1, 3) == "161" ~ "161",
    str_sub(examiner_art_unit, 1, 3) == "162" ~ "162",
    TRUE ~ "Other"
  ))

# Proportion of female employees
female_prop = distinct_dataset %>%
  group_by(group) %>%
  summarise(female_count = sum(gender == "female", na.rm = TRUE),
            total_count = n(),
            prop_female = female_count / total_count)

# Proportion of white employees
white_prop = distinct_dataset %>%
  group_by(group) %>%
  summarise(white_count = sum(race == "white", na.rm = TRUE),
            total_count = n(),
            prop_white = white_count / total_count)

# Proportion of asian employees
asian_prop = distinct_dataset %>%
  group_by(group) %>%
  summarise(asian_count = sum(race == "Asian", na.rm = TRUE),
            total_count = n(),
            prop_asian = asian_count / total_count)

# Proportion of white employees
black_prop = distinct_dataset %>%
  group_by(group) %>%
  summarise(black_count = sum(race == "black", na.rm = TRUE),
            total_count = n(),
            prop_black = black_count / total_count)

print(female_prop)
```

```
## # A tibble: 3 × 4
##   group female_count total_count prop_female
##   <chr>        <int>       <int>       <dbl>
## 1 161            146         362       0.403
## 2 162            112         339       0.330
## 3 Other         2504        9812       0.255
```

```r
print(white_prop)
```

```
## # A tibble: 3 × 4
##   group white_count total_count prop_white
##   <chr>       <int>       <int>      <dbl>
## 1 161           274         362      0.757
## 2 162           234         339      0.690
## 3 Other        6152        9812      0.627
```

```
print(asian_prop)
```

```
## # A tibble: 3 × 4
##   group asian_count total_count prop_asian
##   <chr>       <int>       <int>      <dbl>
## 1 161            70         362      0.193
## 2 162            72         339      0.212
## 3 Other        2903        9812      0.296
```

```
print(black_prop)
```

```
## # A tibble: 3 × 4
##   group black_count total_count prop_black
##   <chr>       <int>       <int>      <dbl>
## 1 161            11         362     0.0304
## 2 162            18         339     0.0531
## 3 Other         388        9812     0.0395
```

```
# Create a contingency table for gender
gender_table = table(distinct_dataset$first_three_digits, distinct_dataset$gender)

# Perform chi-squared test
gender_chi_sq = chisq.test(gender_table)
```

```
## Warning in chisq.test(gender_table): Chi-squared approximation may be incorrect
```

```
# Print the results
print(gender_chi_sq)
```

```
##
##  Pearson's Chi-squared test
##
## data:  gender_table
## X-squared = 530.76, df = 37, p-value < 2.2e-16
```

```r
# Create a contingency table for race
race_table = table(distinct_dataset$first_three_digits, distinct_dataset$race)

# Perform chi-squared test
race_chi_sq = chisq.test(race_table)
```

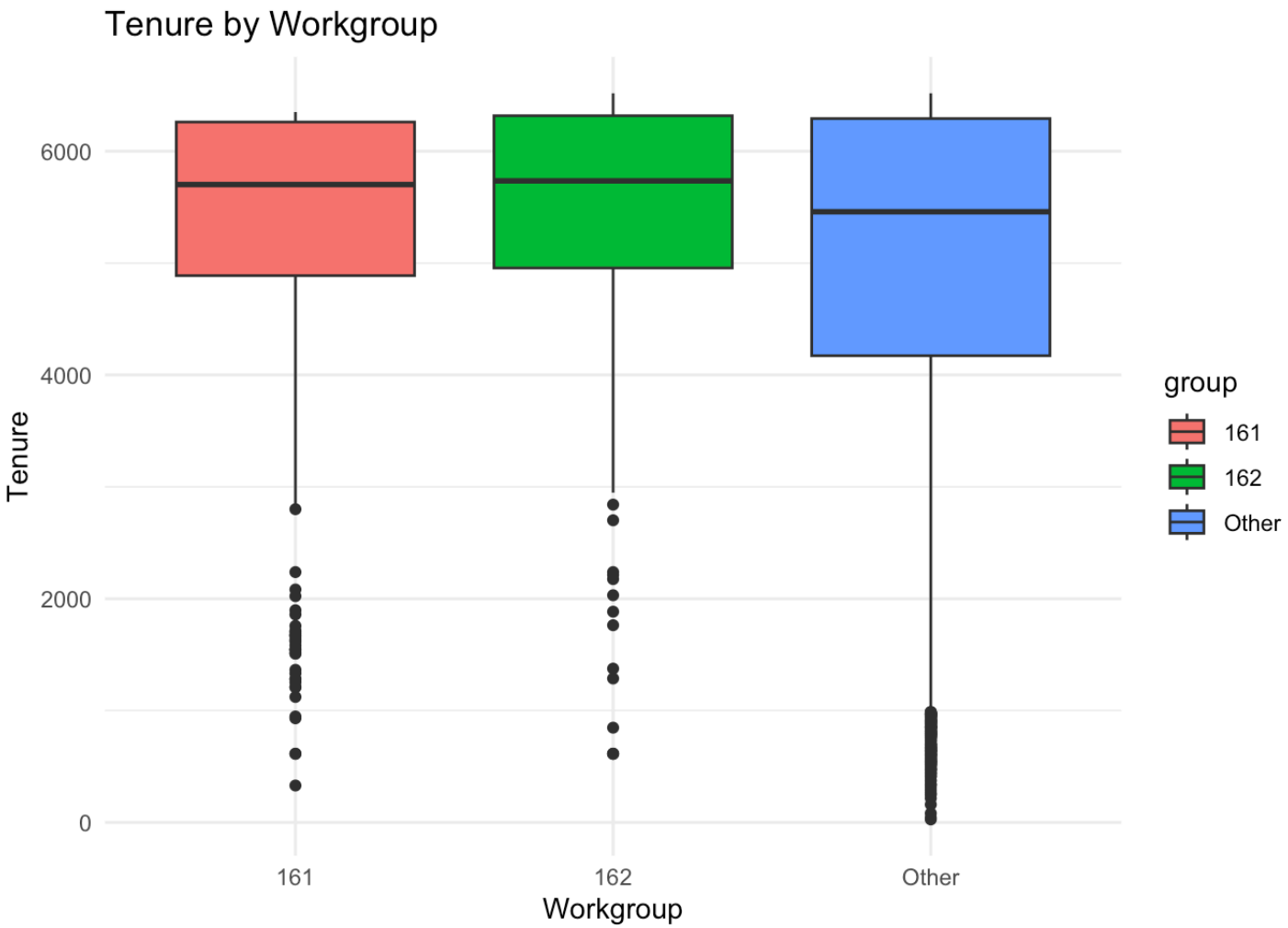
```
## Warning in chisq.test(race_table): Chi-squared approximation may be incorrect
```

```r
# Print the results
print(race_chi_sq)
```

```
##
##  Pearson's Chi-squared test
##
## data:  race_table
## X-squared = 807.51, df = 148, p-value < 2.2e-16
```

```r
library(ggplot2)

# Boxplot for tenure by workgroup
ggplot(distinct_dataset, aes(x = group, y = tenure_days, fill = group)) +
  geom_boxplot() +
  labs(x = "Workgroup", y = "Tenure", title = "Tenure by Workgroup") +
  theme_minimal()
```

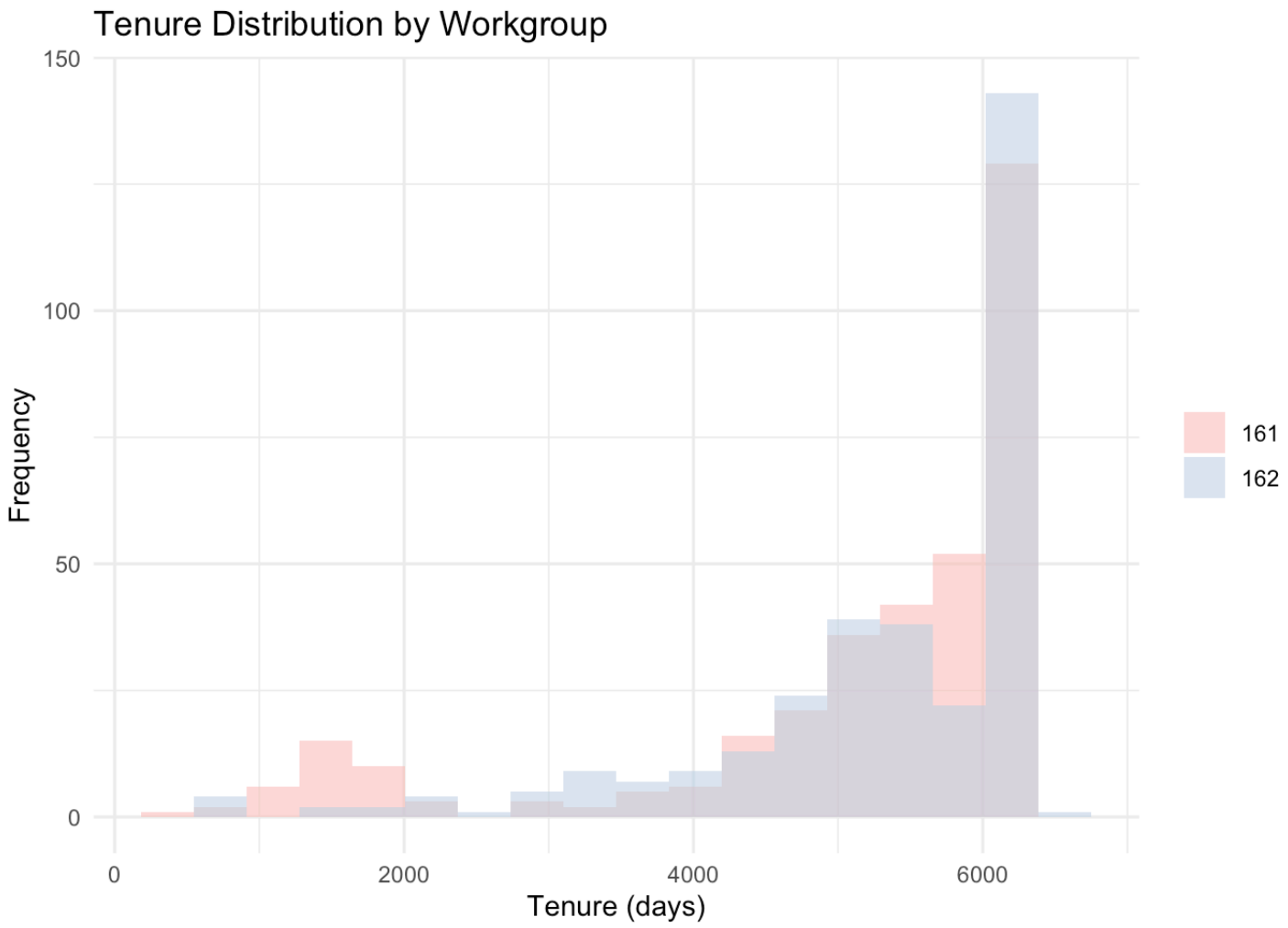
```
## Warning: Removed 247 rows containing non-finite outside the scale range
## (`stat_boxplot()`).
```



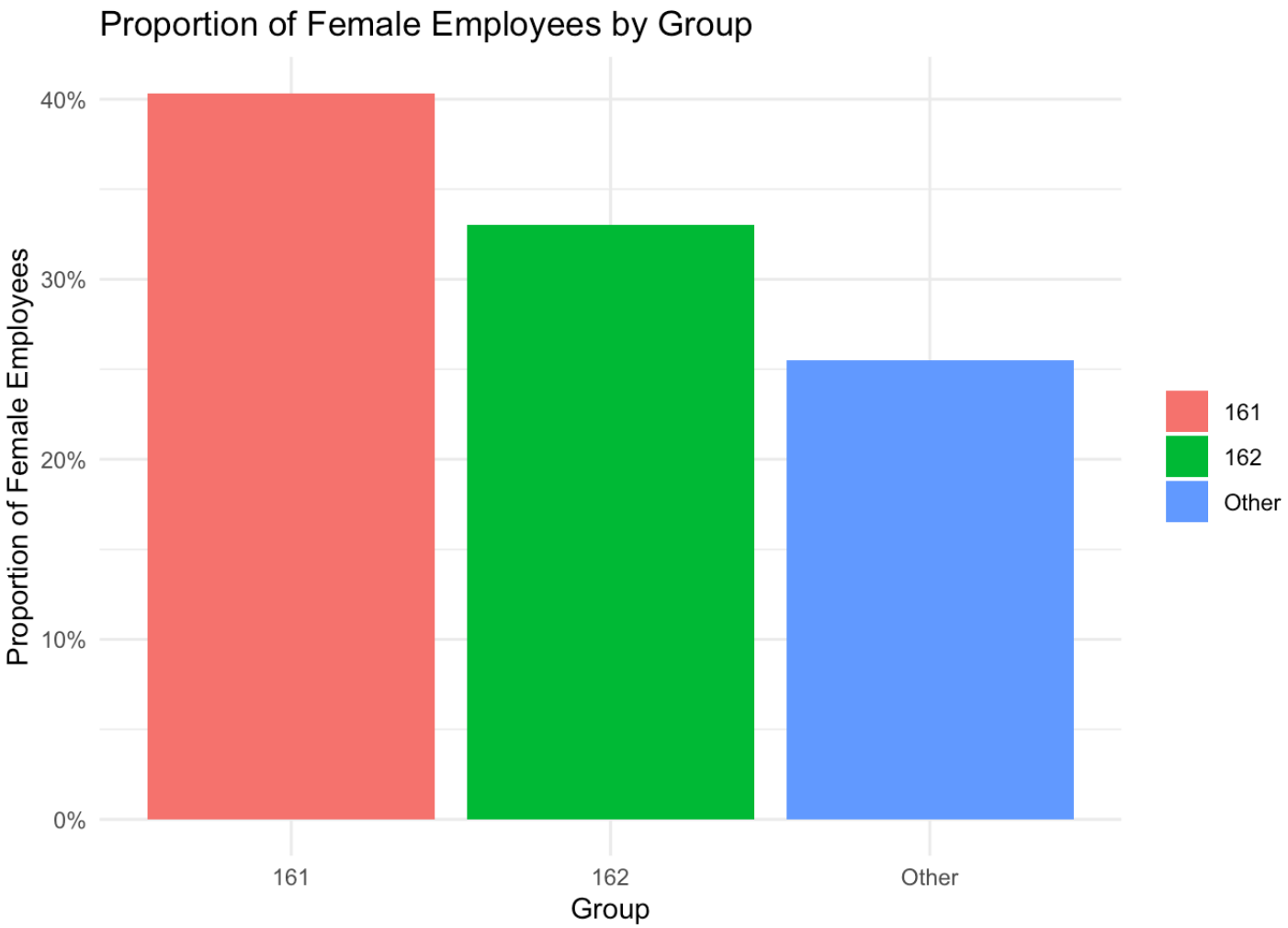

```
distinct_dataset %>%
  mutate(workgroup = substring(examiner_art_unit, 1, 3)) %>%
  ggplot(aes(x = tenure_days, fill = workgroup)) +
  geom_histogram(data = . %>% filter(workgroup == "161"), binwidth = 365, alpha = 0.5
) +
  geom_histogram(data = . %>% filter(workgroup == "162"), binwidth = 365, alpha = 0.5
) +
  labs(title = "Tenure Distribution by Workgroup", x = "Tenure (days)", y = "Frequenc
y") +
  scale_fill_brewer(palette = "Pastel1") +
  theme_minimal() +
  theme(legend.title = element_blank())
```

```
## Warning: Removed 13 rows containing non-finite outside the scale range
## (`stat_bin()`).
```

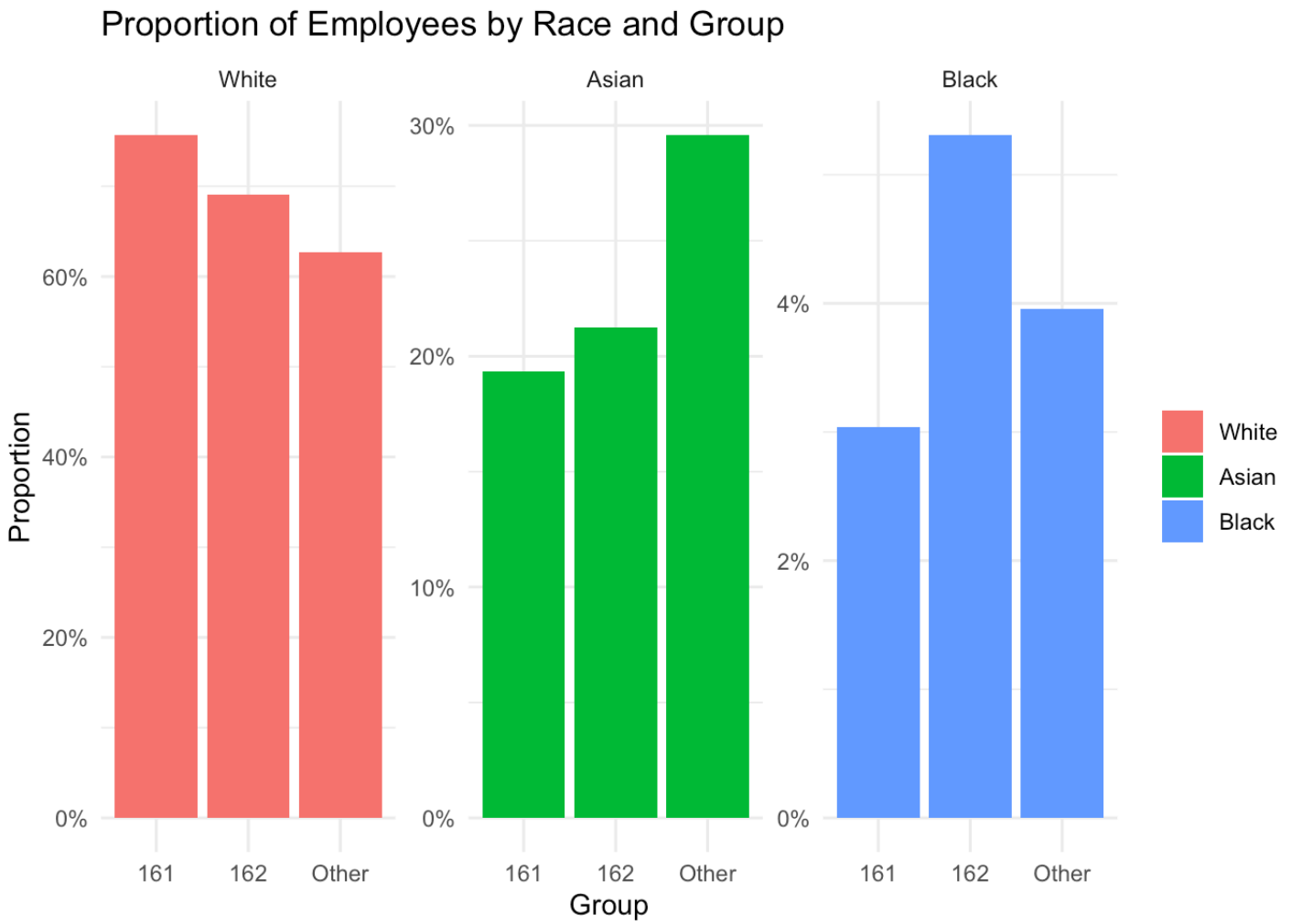
```
## Warning: Removed 16 rows containing non-finite outside the scale range
## (`stat_bin()`).
```




```
df_prop = distinct_dataset %>%
  group_by(group) %>%
  summarise(total_count = n(), # Total employees in each group
            female_count = sum(gender == "female", na.rm = TRUE)) %>%
  mutate(proportion_female = female_count / total_count) %>%
  select(group, proportion_female)


ggplot(df_prop, aes(x = group, y = proportion_female, fill = group)) +
  geom_col() +
  scale_y_continuous(labels = scales::percent_format()) +
  labs(x = "Group", y = "Proportion of Female Employees", title = "Proportion of Fema
le Employees by Group") +
  theme_minimal() +
  theme(legend.title = element_blank())
```


Proportion of Female Employees by Group
40%
30%
20%
10%
0%
Proportion of Female Employees
161
162
Other
Group
161
162
Other

```
df_prop = distinct_dataset %>%
  group_by(group) %>%
  summarise(
    total_count = n(),
    white_count = sum(race == "white", na.rm = TRUE),
    asian_count = sum(race == "Asian", na.rm = TRUE),
    black_count = sum(race == "black", na.rm = TRUE)
  ) %>%
  mutate(
    proportion_white = white_count / total_count,
    proportion_asian = asian_count / total_count,
    proportion_black = black_count / total_count
  ) %>%
  pivot_longer(
    cols = starts_with("proportion"),
    names_to = "race",
    values_to = "proportion"
  ) %>%
  mutate(
    race = factor(race, levels = c("proportion_white", "proportion_asian", "proportio
n_black"),
                  labels = c("White", "Asian", "Black"))
  )

library(ggplot2)

ggplot(df_prop, aes(x = group, y = proportion, fill = race)) +
  geom_col() +
  facet_wrap(~ race, scales = "free_y") +
  scale_y_continuous(labels = scales::percent_format()) +
  labs(x = "Group", y = "Proportion", title = "Proportion of Employees by Race and Gr
oup") +
  theme_minimal() +
  theme(legend.title = element_blank())
```




```
edges_sample = read_csv("/Users/lauray/Documents/GitHub/2024-ona-assignments/Exercise 3/672_project_data/edges_sample.csv")
```

```
## Rows: 32906 Columns: 4
## ── Column specification ────────────────────────────────────────────────────────
## Delimiter: ","
## chr  (1): application_number
## dbl  (2): ego_examiner_id, alter_examiner_id
## date (1): advice_date
##
## 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.
```

```
edges_sample = drop_na(edges_sample)
edges_sample = select(edges_sample, ego_examiner_id, alter_examiner_id)
filtered_examiners = distinct_dataset %>%
  filter(group %in% c("161","162"))
```

```
library(igraph)
```

```
##
## Attaching package: 'igraph'
```

```
## The following objects are masked from 'package:lubridate':
##
##     %--%, union
```

```
## The following objects are masked from 'package:dplyr':
##
##     as_data_frame, groups, union
```

```
## The following objects are masked from 'package:purrr':
##
##     compose, simplify
```

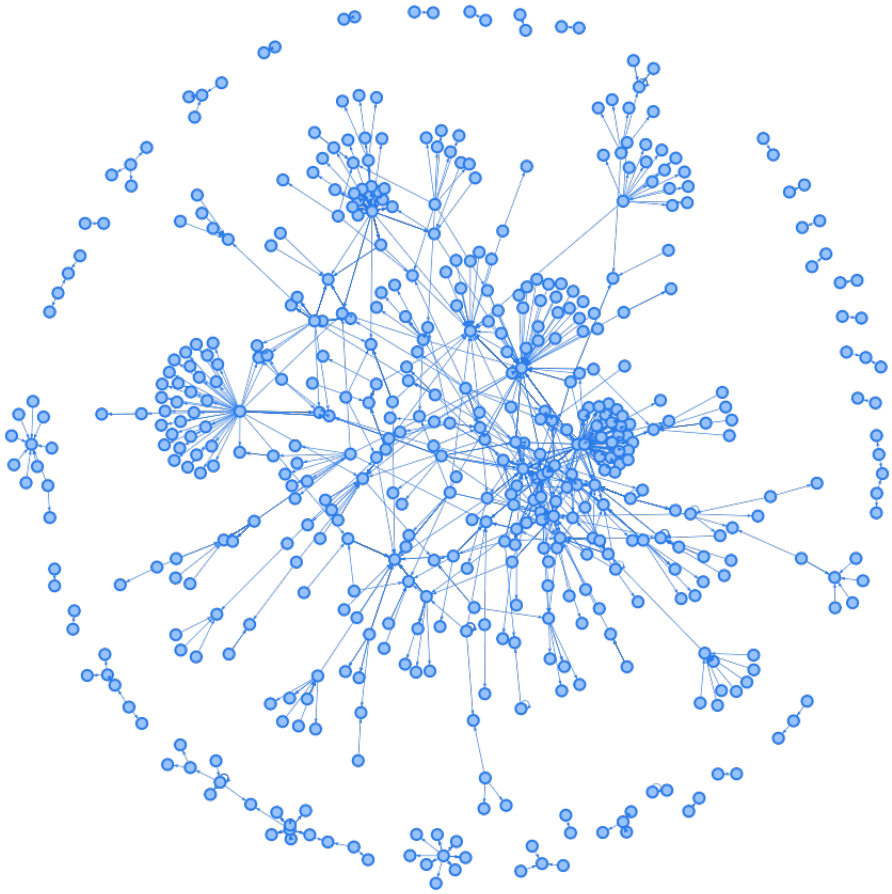
```
## The following object is masked from 'package:tidyr':
##
##     crossing
```

```
## The following object is masked from 'package:tibble':
##
##     as_data_frame
```

```
## The following objects are masked from 'package:stats':
##
##     decompose, spectrum
```

```
## The following object is masked from 'package:base':
##
##     union
```

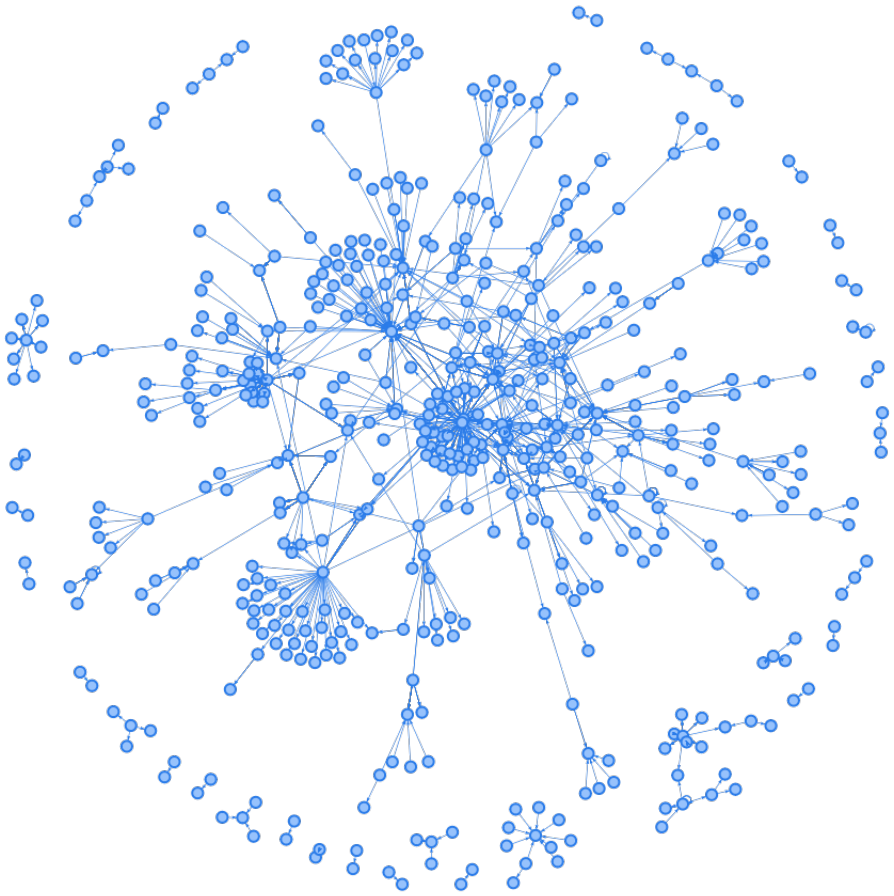
```
library(readr)
edges_for_network = edges_sample %>%
  filter(ego_examiner_id %in% filtered_examiners$examiner_id | alter_examiner_id %in%
filtered_examiners$examiner_id)
network = graph_from_data_frame(edges_for_network, directed = TRUE)


library(visNetwork)
visNetwork::visIgraph(network)
```



```
edges_sample = read_csv("/Users/lauray/Documents/GitHub/2024-ona-assignments/Exercise 3/672_project_data/edges_sample.csv")
```

```
## Rows: 32906 Columns: 4
## ── Column specification ────────────────────────────────────────────────────────
## Delimiter: ","
## chr  (1): application_number
## dbl  (2): ego_examiner_id, alter_examiner_id
## date (1): advice_date
##
## 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.
```

```
edges_sample = drop_na(edges_sample)
edges_sample = select(edges_sample, ego_examiner_id, alter_examiner_id)
filtered_examiners = distinct_dataset %>%
  filter(group %in% c("161"))
filtered_examiners = filtered_examiners %>%
  mutate(examiner_id = as.character(examiner_id))
library(igraph)
library(readr)
edges_for_network = edges_sample %>%
  filter(ego_examiner_id %in% filtered_examiners$examiner_id | alter_examiner_id %in%
filtered_examiners$examiner_id)
network_161 = graph_from_data_frame(edges_for_network, directed = TRUE)


library(visNetwork)
visNetwork::visIgraph(network)
```



```
# Color nodes by gender
gender_df = unique(select(applications, examiner_id, gender))
library(data.table)
```

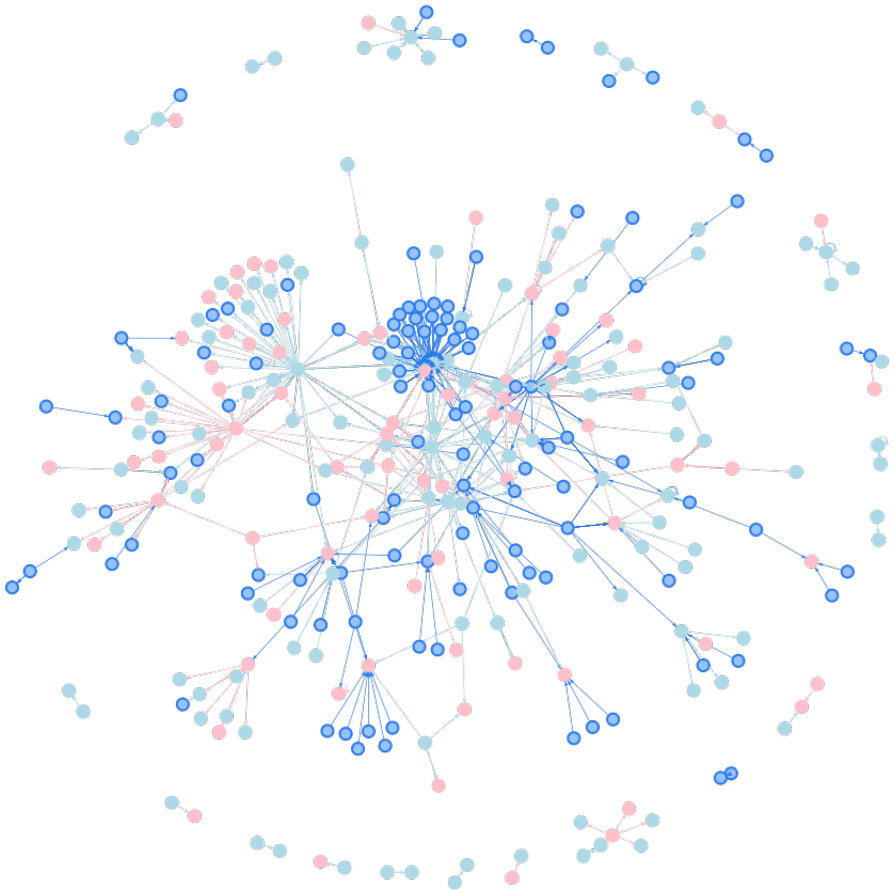
```
##
## Attaching package: 'data.table'
```

```
## The following objects are masked from 'package:lubridate':
##
##     hour, isoweek, mday, minute, month, quarter, second, wday, week,
##     yday, year
```

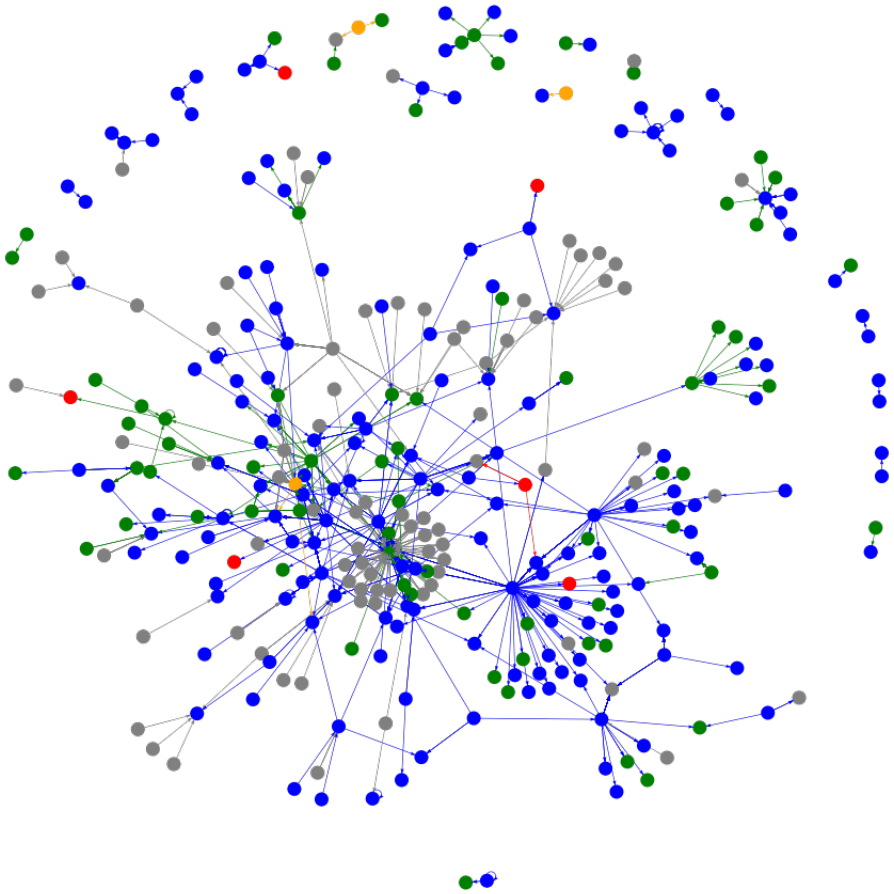
```
## The following objects are masked from 'package:dplyr':
##
##     between, first, last
```

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

```
setDT(edges_for_network)
setDT(gender_df)
edges_for_network = merge(edges_for_network, gender_df, by.x = "ego_examiner_id", by.
y = "examiner_id", all.x = TRUE)
edges_for_network = merge(edges_for_network, gender_df, by.x = "alter_examiner_id", b
y.y = "examiner_id", all.x = TRUE, suffixes = c("_ego", "_alter"))

network_161 <- graph_from_data_frame(edges_for_network, directed = TRUE)

gender_vector <- c(setNames(edges_for_network$gender_ego, edges_for_network$ego_exami
ner_id),
                   setNames(edges_for_network$gender_alter, edges_for_network$alter_e
xaminer_id))

gender_vector <- gender_vector[!duplicated(names(gender_vector))]

V(network_161)$gender <- gender_vector[V(network_161)$name]

getGenderColor <- function(gender) {
  ifelse(gender == "male", "lightblue",
         ifelse(gender == "female", "pink", "black"))
}

V(network_161)$color <- sapply(V(network_161)$gender, getGenderColor)

visNetwork::visIgraph(network_161) %>%
  visNodes(color = list(background = V(network_161)$color))
```



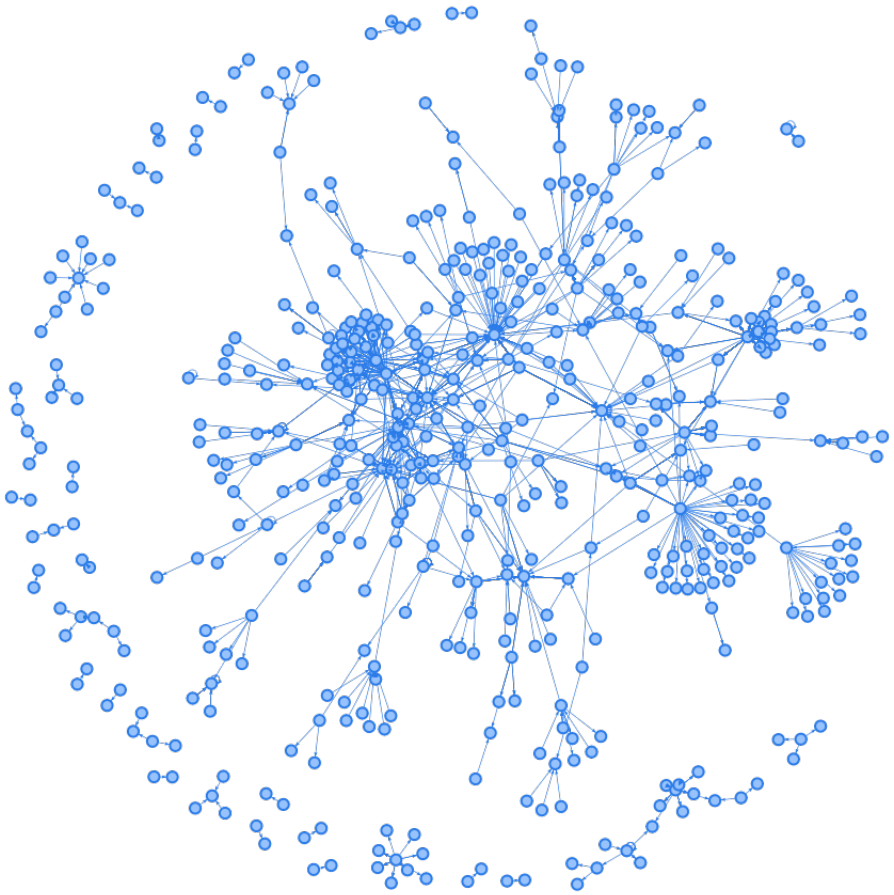
```
race_df = unique(select(applications, examiner_id, race))
setDT(edges_for_network)
setDT(race_df)
edges_for_network = merge(edges_for_network, race_df, by.x = "ego_examiner_id", by.y
= "examiner_id", all.x = TRUE)
edges_for_network = merge(edges_for_network, race_df, by.x = "alter_examiner_id", by.
y = "examiner_id", all.x = TRUE, suffixes = c("_ego", "_alter"))
network_161 <- graph_from_data_frame(edges_for_network, directed = TRUE)

race_vector <- c(setNames(edges_for_network$race_ego, edges_for_network$ego_examiner_
id),
                 setNames(edges_for_network$race_alter, edges_for_network$alter_exami
ner_id))

race_vector <- race_vector[!duplicated(names(race_vector))]

V(network_161)$race <- race_vector[V(network_161)$name]

race_colors <- c("Asian" = "green", "black" = "red", "Hispanic" = "orange", "white" =
"blue", "other" = "purple")

V(network_161)$color <- ifelse(V(network_161)$race %in% names(race_colors), race_colo
rs[V(network_161)$race], "gray")

visNetwork::visIgraph(network_161) %>%
  visNodes(color = list(background = V(network_161)$color,
                        border = "#2b2b2b", highlight = "#d9d9d9"))
```



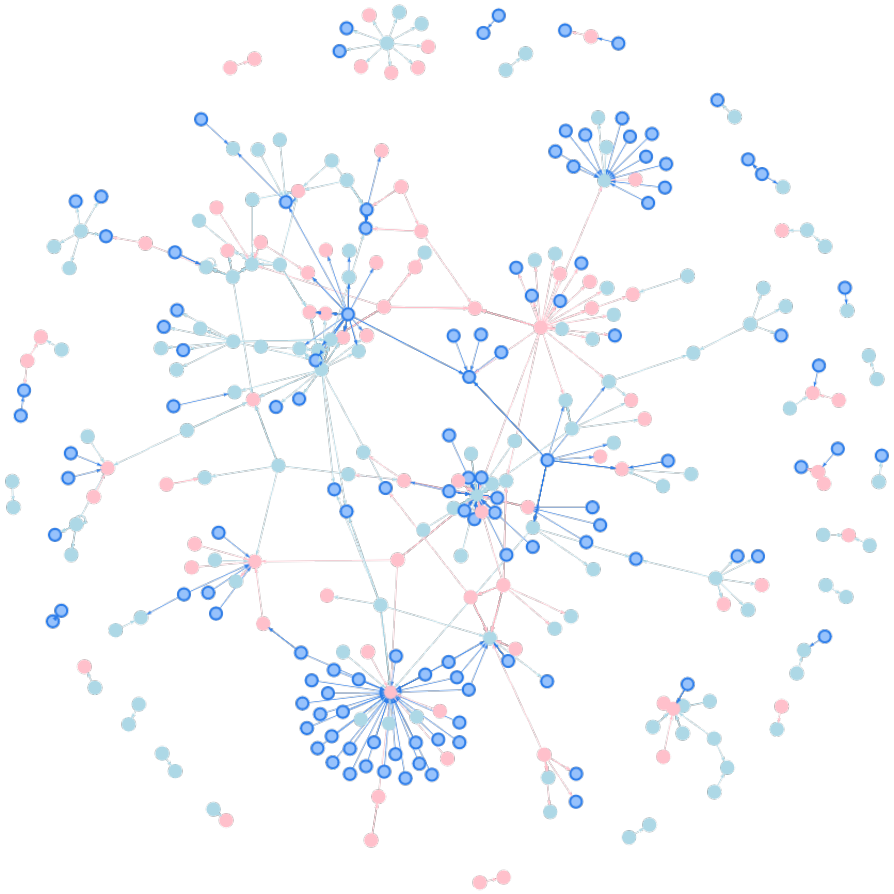
```
edges_sample = read_csv("/Users/lauray/Documents/GitHub/2024-ona-assignments/Exercise 3/672_project_data/edges_sample.csv")
```

```
## Rows: 32906 Columns: 4
## ── Column specification ────────────────────────────────────────────────────────
## Delimiter: ","
## chr  (1): application_number
## dbl  (2): ego_examiner_id, alter_examiner_id
## date (1): advice_date
##
## 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.
```

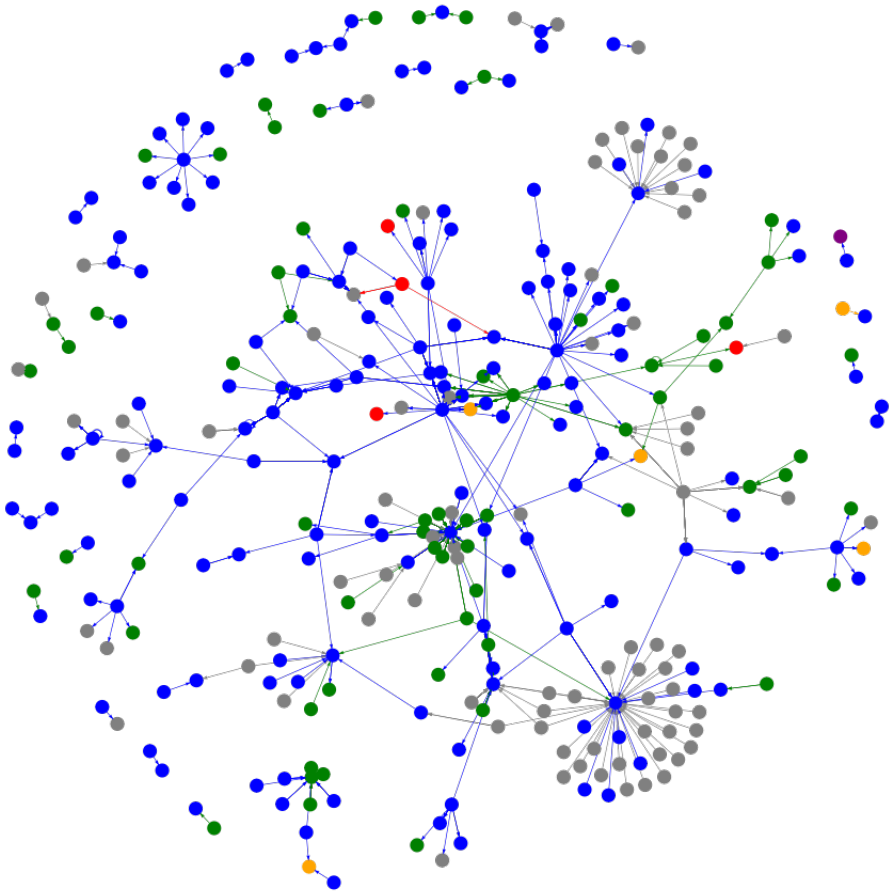
```
edges_sample = drop_na(edges_sample)
edges_sample = select(edges_sample, ego_examiner_id, alter_examiner_id)
filtered_examiners = distinct_dataset %>%
  filter(group %in% c("162"))
library(igraph)
library(readr)
edges_for_network = edges_sample %>%
  filter(ego_examiner_id %in% filtered_examiners$examiner_id | alter_examiner_id %in%
filtered_examiners$examiner_id)
network_162 = graph_from_data_frame(edges_for_network, directed = TRUE)

visNetwork::visIgraph(network)
```



```
# Color nodes by gender
gender_df = unique(select(applications, examiner_id, gender))
library(data.table)
setDT(edges_for_network)
setDT(gender_df)
edges_for_network = merge(edges_for_network, gender_df, by.x = "ego_examiner_id", by.
y = "examiner_id", all.x = TRUE)
edges_for_network = merge(edges_for_network, gender_df, by.x = "alter_examiner_id", b
y.y = "examiner_id", all.x = TRUE, suffixes = c("_ego", "_alter"))

network_162 <- graph_from_data_frame(edges_for_network, directed = TRUE)

gender_vector <- c(setNames(edges_for_network$gender_ego, edges_for_network$ego_exami
ner_id),
                   setNames(edges_for_network$gender_alter, edges_for_network$alter_e
xaminer_id))

gender_vector <- gender_vector[!duplicated(names(gender_vector))]

V(network_162)$gender <- gender_vector[V(network_162)$name]

getGenderColor <- function(gender) {
  ifelse(gender == "male", "lightblue",
         ifelse(gender == "female", "pink", "black"))
}

V(network_162)$color <- sapply(V(network_162)$gender, getGenderColor)

visNetwork::visIgraph(network_162) %>%
  visNodes(color = list(background = V(network_162)$color))
```



```
race_df = unique(select(applications, examiner_id, race))
setDT(edges_for_network)
setDT(race_df)
edges_for_network = merge(edges_for_network, race_df, by.x = "ego_examiner_id", by.y
= "examiner_id", all.x = TRUE)
edges_for_network = merge(edges_for_network, race_df, by.x = "alter_examiner_id", by.
y = "examiner_id", all.x = TRUE, suffixes = c("_ego", "_alter"))
network_162 <- graph_from_data_frame(edges_for_network, directed = TRUE)

# Create a named vector of races with examiner IDs as names
race_vector <- c(setNames(edges_for_network$race_ego, edges_for_network$ego_examiner_
id),
                 setNames(edges_for_network$race_alter, edges_for_network$alter_exami
ner_id))

# Remove potential duplicates, keeping the first occurrence
race_vector <- race_vector[!duplicated(names(race_vector))]

# Assign race to vertices based on the vector created above
V(network_162)$race <- race_vector[V(network_162)$name]

# Define a color palette for the races, adjust according to your race categories
race_colors <- c("Asian" = "green", "black" = "red", "Hispanic" = "orange", "white" =
"blue", "other" = "purple")

# Assign colors to nodes based on race, default to 'gray' if race is not in the palet
te
V(network_162)$color <- ifelse(V(network_162)$race %in% names(race_colors), race_colo
rs[V(network_162)$race], "gray")

# Visualize the network with visNetwork
visNetwork::visIgraph(network_162) %>%
  visNodes(color = list(background = V(network_162)$color,
                        border = "#2b2b2b", highlight = "#d9d9d9"))
```



```
closeness_centrality = closeness(network_161, mode = "out")
centrality_scores = data.frame(examiner_id = V(network_161)$name, closeness_centralit
y = closeness_centrality) %>%
  mutate(examiner_id = as.numeric(examiner_id))
final_dataset_161 = distinct_dataset %>%
  left_join(centrality_scores, by = "examiner_id")
average_by_gender_161 = final_dataset_161 %>%
  group_by(gender) %>%
  summarise(average_closeness_centrality = mean(closeness_centrality, na.rm = TRUE))

closeness_centrality = closeness(network_162, mode = "out")
centrality_scores = data.frame(examiner_id = V(network_162)$name, closeness_centralit
y = closeness_centrality) %>%
  mutate(examiner_id = as.numeric(examiner_id))
final_dataset_162 = distinct_dataset %>%
  left_join(centrality_scores, by = "examiner_id")

average_by_gender_162 = final_dataset_162 %>%
  group_by(gender) %>%
  summarise(average_closeness_centrality = mean(closeness_centrality, na.rm = TRUE))

# Display the table
print(average_by_gender_161)
```

```
## # A tibble: 3 × 2
##   gender average_closeness_centrality
##   <chr>                         <dbl>
## 1 female                        0.430
## 2 male                          0.567
## 3 <NA>                          0.587
```

```
print(average_by_gender_162)
```

```
## # A tibble: 3 × 2
##   gender average_closeness_centrality
##   <chr>                         <dbl>
## 1 female                        0.598
## 2 male                          0.641
## 3 <NA>                          0.582
```

```
# Calculate average closeness centrality score grouped by race
average_by_race_161 = final_dataset_161 %>%
  group_by(race) %>%
  summarise(average_closeness_centrality = mean(closeness_centrality, na.rm = TRUE))

average_by_race_162 = final_dataset_162 %>%
  group_by(race) %>%
  summarise(average_closeness_centrality = mean(closeness_centrality, na.rm = TRUE))


# Display the table
print(average_by_race_161)
```

```
## # A tibble: 5 × 2
##   race     average_closeness_centrality
##   <chr>                           <dbl>
## 1 Asian                           0.652
## 2 Hispanic                        0.571
## 3 black                           0.5
## 4 other                         NaN
## 5 white                           0.484
```

```
print(average_by_race_162)
```

```
## # A tibble: 5 × 2
##   race     average_closeness_centrality
##   <chr>                           <dbl>
## 1 Asian                           0.666
## 2 Hispanic                        1
## 3 black                           0.5
## 4 other                         NaN
## 5 white                           0.597
```

```
library(dplyr)

# Define bins for tenure days. Adjust the breaks as needed for your dataset.
tenure_bins = c(0, 365, 730, 1095, 1460, Inf) # Example bins: <1 year, 1-2 years, 2-3 years, 3-4 years, >4 years
labels = c("<1 year", "1-2 years", "2-3 years", "3-4 years", ">4 years") # Labels for the bins

# Create a new column for tenure categories
final_dataset_161 = final_dataset_161 %>%
  mutate(tenure_category = cut(tenure_days, breaks = tenure_bins, labels = labels, ri
ght = FALSE))

# Calculate average closeness centrality score grouped by tenure category
average_by_tenure_category_161 = final_dataset_161 %>%
  group_by(tenure_category) %>%
  summarise(average_closeness_centrality = mean(closeness_centrality, na.rm = TRUE))


final_dataset_162 = final_dataset_162 %>%
  mutate(tenure_category = cut(tenure_days, breaks = tenure_bins, labels = labels, ri
ght = FALSE))

average_by_tenure_category_162 = final_dataset_162 %>%
  group_by(tenure_category) %>%
  summarise(average_closeness_centrality = mean(closeness_centrality, na.rm = TRUE))


# Display the table
print(average_by_tenure_category_161)
```

```
## # A tibble: 6 × 2
##   tenure_category average_closeness_centrality
##   <fct>                                  <dbl>
## 1 <1 year                              NaN
## 2 1-2 years                              0.5
## 3 2-3 years                            NaN
## 4 3-4 years                              0.833
## 5 >4 years                               0.531
## 6 <NA>                                 NaN
```

```
print(average_by_tenure_category_162)
```

```
## # A tibble: 6 × 2
##   tenure_category average_closeness_centrality
##   <fct>                                  <dbl>
## 1 <1 year                              NaN
## 2 1-2 years                              0.5
## 3 2-3 years                              0.528
## 4 3-4 years                            NaN
## 5 >4 years                               0.624
## 6 <NA>                                 NaN
```

```
library(igraph)
# Calculate betweenness centrality
betweenness_161 = betweenness(network_161, directed = TRUE)
centrality_161 = data.frame(examiner_id = V(network_161)$name, betweenness = betweenn
ess_161)

betweenness_162 = betweenness(network_162, directed = TRUE)
centrality_162 = data.frame(examiner_id = V(network_162)$name, betweenness = betweenn
ess_162)

centrality_161 = centrality_161 %>%
  mutate(examiner_id = as.numeric(examiner_id))

centrality_162 = centrality_162 %>%
  mutate(examiner_id = as.numeric(examiner_id))

final_data_161 = distinct_dataset %>%
  filter(group == "161") %>%
  left_join(centrality_161, by = "examiner_id")

final_data_162 = distinct_dataset %>%
  filter(group == "162") %>%
  left_join(centrality_162, by = "examiner_id")

# Group by gender and calculate average betweenness
average_betweenness_gender_161 = final_data_161 %>%
```

```
  group_by(gender) %>%
  summarise(average_betweenness = mean(betweenness, na.rm = TRUE))

average_betweenness_gender_162 = final_data_162 %>%
  group_by(gender) %>%
  summarise(average_betweenness = mean(betweenness, na.rm = TRUE))

average_betweenness_race_161 = final_data_161 %>%
  group_by(race) %>%
  summarise(average_betweenness = mean(betweenness, na.rm = TRUE))

average_betweenness_race_162 = final_data_162 %>%
  group_by(race) %>%
  summarise(average_betweenness = mean(betweenness, na.rm = TRUE))

final_data_161 = final_data_161 %>%
  mutate(tenure_category = cut(tenure_days, breaks = tenure_bins, labels = labels, ri
ght = FALSE))

average_betweenness_tenure_161 = final_data_161 %>%
  group_by(tenure_category) %>%
  summarise(average_betweenness = mean(betweenness, na.rm = TRUE))

final_data_162 = final_data_162 %>%
  mutate(tenure_category = cut(tenure_days, breaks = tenure_bins, labels = labels, ri
ght = FALSE))

average_betweenness_tenure_162 = final_data_162 %>%
  group_by(tenure_category) %>%
  summarise(average_betweenness = mean(betweenness, na.rm = TRUE))

print(average_betweenness_gender_161)
```

```
## # A tibble: 3 × 2
##   gender average_betweenness
##   <chr>                <dbl>
## 1 female                4.23
## 2 male                  3.54
## 3 <NA>                  3.43
```

```
print(average_betweenness_gender_162)
```

```
## # A tibble: 3 × 2
##   gender average_betweenness
##   <chr>                <dbl>
## 1 female               0.927
## 2 male                 0.607
## 3 <NA>                 1.10
```

```
print(average_betweenness_race_161)
```

```
## # A tibble: 4 × 2
##   race     average_betweenness
##   <chr>                  <dbl>
## 1 Asian                  3.74
## 2 Hispanic              30.6
## 3 black                  0.511
## 4 white                  3.45
```

```
print(average_betweenness_race_162)
```

```
## # A tibble: 4 × 2
##   race     average_betweenness
##   <chr>                  <dbl>
## 1 Asian                  0.7
## 2 Hispanic               0
## 3 black                  0.256
## 4 white                  0.909
```

```
print(average_betweenness_tenure_161)
```

```
## # A tibble: 6 × 2
##   tenure_category average_betweenness
##   <fct>                         <dbl>
## 1 <1 year                      NaN
## 2 1-2 years                      0
## 3 2-3 years                    NaN
## 4 3-4 years                      0
## 5 >4 years                       3.91
## 6 <NA>                           0
```

```
print(average_betweenness_tenure_162)
```

```
## # A tibble: 5 × 2
##   tenure_category average_betweenness
##   <fct>                         <dbl>
## 1 1-2 years                      0
## 2 2-3 years                      0
## 3 3-4 years                    NaN
## 4 >4 years                       0.845
## 5 <NA>                           0
```