Exercise 3: Examiners' Demographics and Advice Networks

Liliana Tretyakova

March 27, 2023

Introduction

In this assignment, we analyze the demographics of examiners in two selected workgroups and explore the advice networks within those workgroups. Specifically, we will:

1. Load the data files and add the following variables for examiners:

- Gender
- Race
- Tenure

2. Choose two workgroups and compare their demographics through summary statistics and plots.
3. Create advice networks from edges_sample and calculate centrality scores for examiners in the selected workgroups.

```
# Load required libraries
library(tidyverse)
```

```
## Warning: package 'tidyverse' was built under R version 4.2.2
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.2 --
## v ggplot2 3.3.6      v purrr   0.3.4
## v tibble  3.1.8      v dplyr   1.0.10
## v tidyr   1.2.1      v stringr 1.4.1
## v readr   2.1.3      v forcats 0.5.2
```

```
## Warning: package 'readr' was built under R version 4.2.2
```

```
## Warning: package 'dplyr' was built under R version 4.2.2
```

```
## Warning: package 'forcats' was built under R version 4.2.2
```

```
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(lubridate)
```

```
## Warning: package 'lubridate' was built under R version 4.2.2
```

```
## Loading required package: timechange
```

```
## Warning: package 'timechange' was built under R version 4.2.2
```

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

```
library(arrow)
```

```
## Warning: package 'arrow' was built under R version 4.2.2
```

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

```
library(gender)
```

```
## Warning: package 'gender' was built under R version 4.2.2
```

```
library(wru)
```

```
## Warning: package 'wru' was built under R version 4.2.2
```

```
library(igraph)
```

```
## Warning: package 'igraph' was built under R version 4.2.2
```

```
##
## 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(ggplot2)

# Load data
applications <- read_parquet("C:/Users/ulyan/OneDrive - McGill University/Documents/MMA/Winter II 2023/
edges_sample <- read_csv("C:/Users/ulyan/OneDrive - McGill University/Documents/MMA/Winter II 2023/Org
```

```
## 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.
```

Add Gender, Race, and Tenure Variables for Examiners

First, we will add the gender, race, and tenure variables to the examiners' data.

Here's how we can add gender variable:

```
# Get unique examiner first names
examiner_names <- applications %>% distinct(examiner_name_first)

# Predict gender based on first names
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)

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

Now, let's add race variable:

```
# Get unique examiner last names
examiner_surnames <- applications %>% select(surname = examiner_name_last) %>% distinct()

# Predict race based on last names
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.
```

```
# Select the race with the highest probability for each last name
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_
  ))

# Join race data back to the main applications dataset
applications <- applications %>%
  left_join(examiner_race, by = c("examiner_name_last" = "surname"))
```

Finally, we estimate and add tenure variable:

```
# Extract examiner IDs and application dates
examiner_dates <- applications %>%
  select(examiner_id, filing_date, appl_status_date)

# Convert dates to a consistent format
examiner_dates <- examiner_dates %>%
  mutate(start_date = ymd(filing_date), end_date = as_date(dmy_hms(appl_status_date)))

# Calculate the earliest and latest dates for each examiner and their tenure in days
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)

# Join tenure data back to the main applications dataset
applications <- applications %>% left_join(examiner_dates, by = "examiner_id")
```

Now that we have added gender, race, and tenure variables to the examiners' data, let's proceed with the analysis.

Select and Compare Demographics of Two Workgroups

First, we will select two workgroups and analyze their demographics by generating summary statistics and plots.

```
# Choose workgroups
workgroup1 <- "161"
workgroup2 <- "162"
```

We will start with summary statistics:

```
# Filter the applications dataset for the chosen workgroups
workgroups_data <- applications %>%
  filter(substr(examiner_art_unit, 1, 3) %in% c(workgroup1, workgroup2))

# Summary statistics for demographics
summary_stats <- workgroups_data %>%
  group_by(workgroup = substr(examiner_art_unit, 1, 3)) %>%
  summarise(
    avg_tenure_days = mean(tenure_days, na.rm = TRUE),
    proportion_female = mean(proportion_female, na.rm = TRUE),
    count = n()
  ) %>%
  mutate(across(c(avg_tenure_days, proportion_female), round, 2))

# Print summary statistics
print(summary_stats)
```

```
## # A tibble: 2 x 4
##   workgroup avg_tenure_days proportion_female  count
##   <chr>               <dbl>             <dbl>  <int>
## 1 161                 5679.              0.49  89795
## 2 162                 5806.              0.48 141390
```

Next, let's take a look at demographics in workgroups:

```r
# Filter the applications dataset for the chosen workgroups
workgroups_data <- applications %>%
  filter(substr(examiner_art_unit, 1, 3) %in% c(workgroup1, workgroup2))

# Gender distribution
gender_distribution <- workgroups_data %>%
  group_by(workgroup = substr(examiner_art_unit, 1, 3), gender) %>%
  summarise(count = n()) %>%
  arrange(workgroup, count, .by_group = TRUE)
```

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

```r
# Race distribution
race_distribution <- workgroups_data %>%
  group_by(workgroup = substr(examiner_art_unit, 1, 3), race) %>%
  summarise(count = n()) %>%
  arrange(workgroup, count, .by_group = TRUE)
```

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

```r
# Tenure distribution (grouped by years)
tenure_distribution <- workgroups_data %>%
  mutate(tenure_years = floor(tenure_days / 365)) %>%
  group_by(workgroup = substr(examiner_art_unit, 1, 3), tenure_years) %>%
  summarise(count = n()) %>%
  arrange(workgroup, tenure_years)
```

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

```r
# Display summary tables
print(gender_distribution)
```

```
## # A tibble: 6 x 3
## # Groups:   workgroup [2]
##   workgroup gender count
##   <chr>     <chr>  <int>
## 1 161       <NA>   12966
## 2 161       female 37275
## 3 161       male   39554
## 4 162       <NA>   34598
## 5 162       female 51412
## 6 162       male   55380
```

```r
print(race_distribution)
```

```
## # A tibble: 8 x 3
## # Groups:   workgroup [2]
```

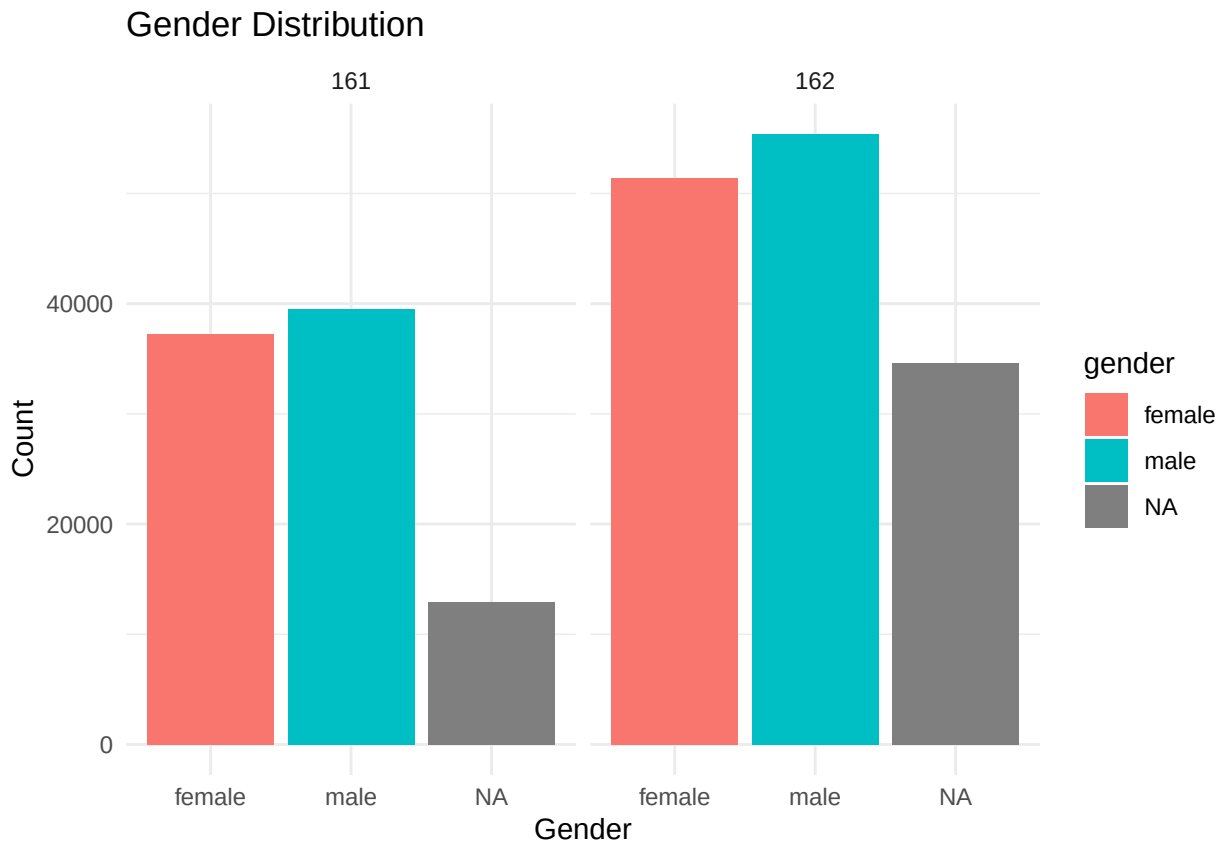
```
##   workgroup race     count
##   <chr>     <chr>    <int>
## 1 161       Hispanic  1843
## 2 161       black     2452
## 3 161       Asian    19528
## 4 161       white    65972
## 5 162       Hispanic  3884
## 6 162       black    11023
## 7 162       Asian    35442
## 8 162       white    91041
```

```
print(tenure_distribution, n=37)
```

```
## # A tibble: 37 x 3
## # Groups:   workgroup [2]
##    workgroup tenure_years count
##    <chr>            <dbl> <int>
##  1 161                  0     1
##  2 161                  1     2
##  3 161                  2     4
##  4 161                  3   168
##  5 161                  4   233
##  6 161                  5   118
##  7 161                  6    12
##  8 161                  7   141
##  9 161                  8     4
## 10 161                  9   961
## 11 161                 10  1382
## 12 161                 11  2717
## 13 161                 12  4762
## 14 161                 13  7204
## 15 161                 14 10448
## 16 161                 15 11042
## 17 161                 16 14499
## 18 161                 17 32366
## 19 161                 NA  3731
## 20 162                  1     6
## 21 162                  2     1
## 22 162                  3     6
## 23 162                  4   203
## 24 162                  5    26
## 25 162                  6     6
## 26 162                  7   479
## 27 162                  8   759
## 28 162                  9  1308
## 29 162                 10  2412
## 30 162                 11  1594
## 31 162                 12  5134
## 32 162                 13 10135
## 33 162                 14 16627
## 34 162                 15 10559
## 35 162                 16 15547
## 36 162                 17 72199
## 37 162                 NA  4389
```

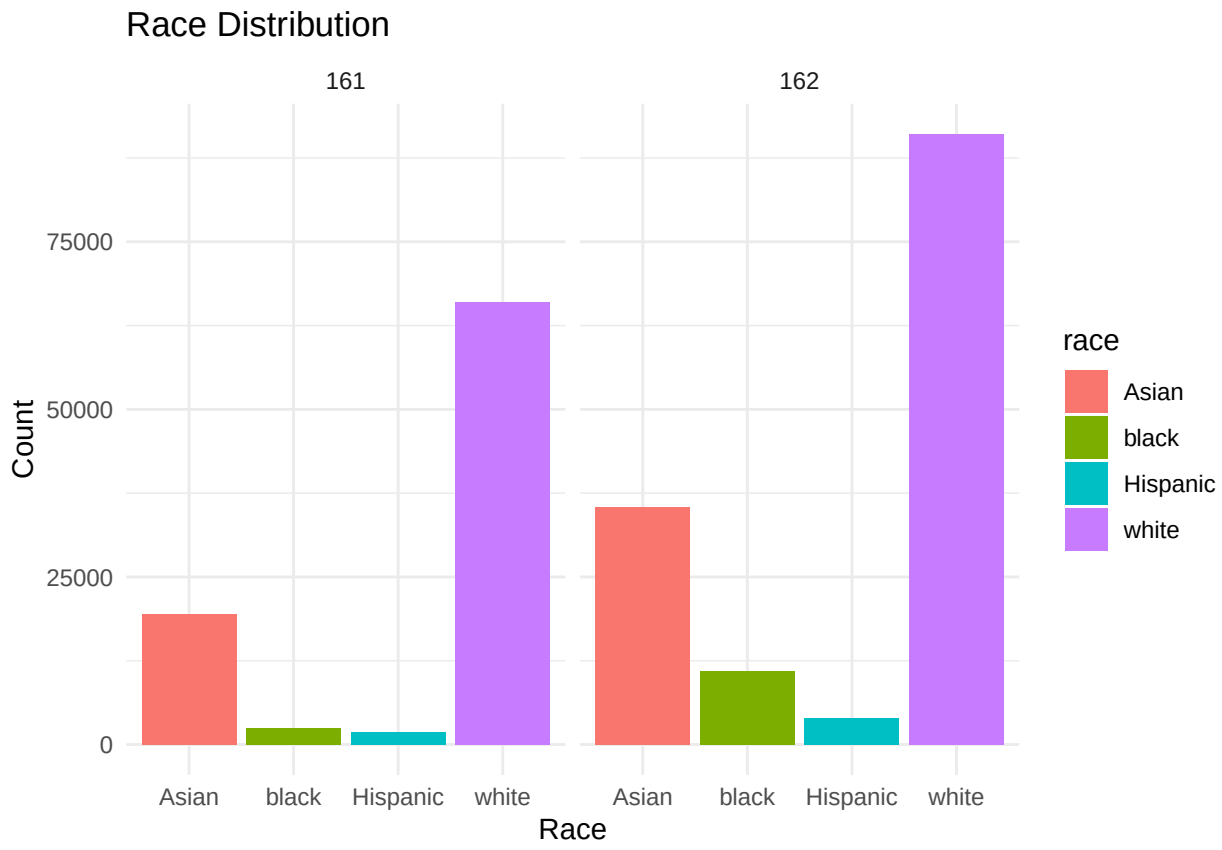
Now, let's visualize the demographics of the selected workgroups using bar plots.

```
# Plot for gender distribution
gender_plot <- workgroups_data %>%
  ggplot(aes(x = gender, fill = gender)) +
  geom_bar() +
  facet_wrap(~substr(examiner_art_unit, 1, 3)) +
  labs(title = "Gender Distribution",
       x = "Gender",
       y = "Count") +
  theme_minimal()

# Display plot
print(gender_plot)
```



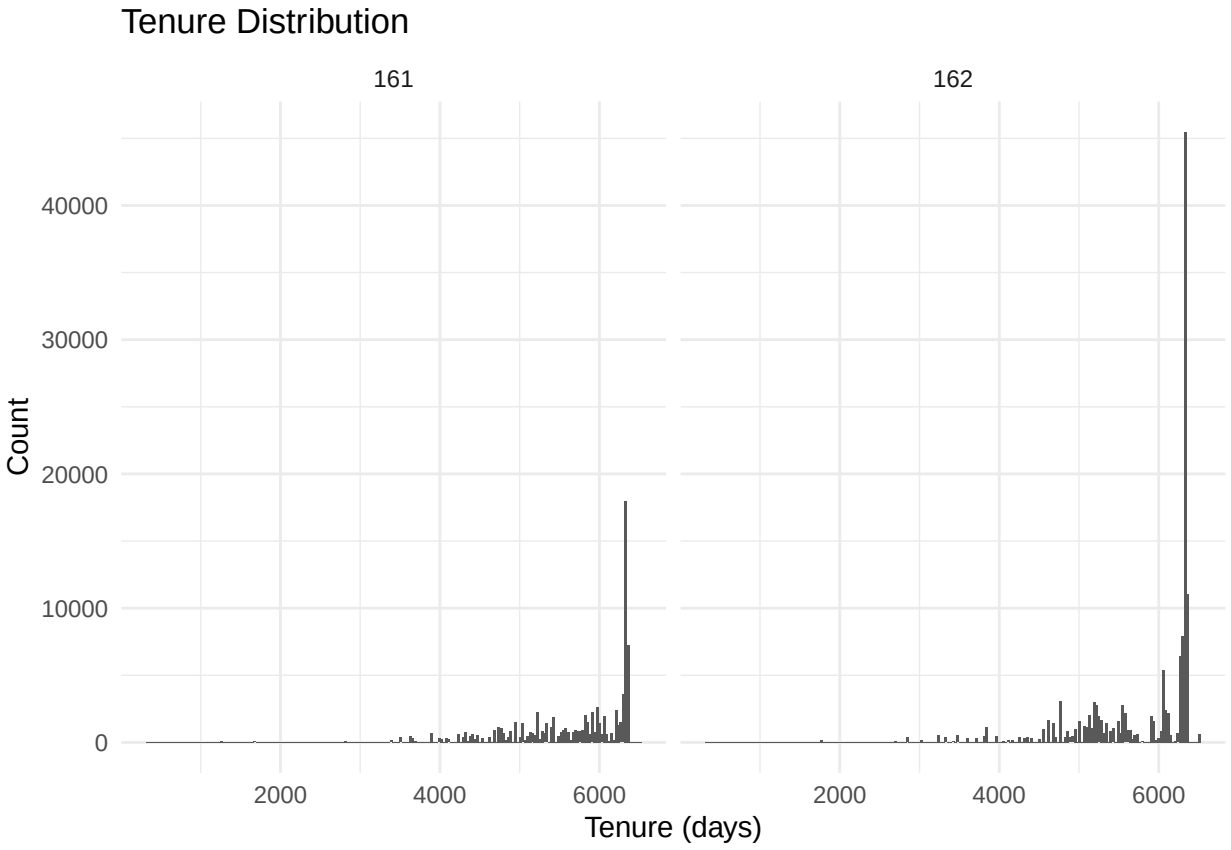

```
# Plot for race distribution
race_plot <- workgroups_data %>%
  ggplot(aes(x = race, fill = race)) +
  geom_bar() +
  facet_wrap(~substr(examiner_art_unit, 1, 3)) +
  labs(title = "Race Distribution",
       x = "Race",
       y = "Count") +
  theme_minimal()
```

```
# Display plot
print(race_plot)
```




```
# Plot for tenure distribution
tenure_plot <- workgroups_data %>%
  ggplot(aes(x = tenure_days)) +
  geom_histogram(binwidth = 30) +
  facet_wrap(~substr(examiner_art_unit, 1, 3)) +
  labs(title = "Tenure Distribution",
       x = "Tenure (days)",
       y = "Count") +
  theme_minimal()

# Display plot
print(tenure_plot)
```

```
## Warning: Removed 8120 rows containing non-finite values (stat_bin).
```



We can make the following observations about the two workgroups:

Gender Distribution: - Workgroup 161 has 37,275 female and 39,554 male examiners, with 12,966 examiners having unknown gender. - Workgroup 162 has 51,412 female and 55,380 male examiners, with 34,598 examiners having unknown gender.

In both workgroups, the number of male examiners is slightly higher than the number of female examiners. However, there are also a considerable number of examiners with unknown gender in both workgroups.

Race Distribution: - Workgroup 161 has 1,843 Hispanic, 2,452 Black, 19,528 Asian, and 65,972 White examiners. - Workgroup 162 has 3,884 Hispanic, 11,023 Black, 35,442 Asian, and 91,041 White examiners.

In both workgroups, the majority of examiners are White, followed by Asian, Black, and Hispanic examiners. Workgroup 162 has a larger number of examiners for each race compared to Workgroup 161.

Tenure Distribution (grouped by years): - Both workgroups show a similar trend in tenure distribution, with the number of examiners generally increasing as the tenure in years increases. - For both workgroups, the largest number of examiners fall into the 17-year tenure category. There are 32,366 examiners in Workgroup 161 and 72,199 examiners in Workgroup 162 with a 17-year tenure. - A considerable number of examiners in both workgroups have unknown tenure, 3,731 in Workgroup 161 and 4,389 in Workgroup 162.

In summary, the two workgroups have similar trends in terms of examiners' demographics. Both workgroups have slightly more male than female examiners, a majority of White examiners followed by Asian, Black, and Hispanic examiners, and a similar distribution of tenure years with the largest number of examiners having a 17-year tenure. Workgroup 162 has a larger number of examiners for each demographic category compared to Workgroup 161.

Create Advice Networks and Calculate Centrality Scores

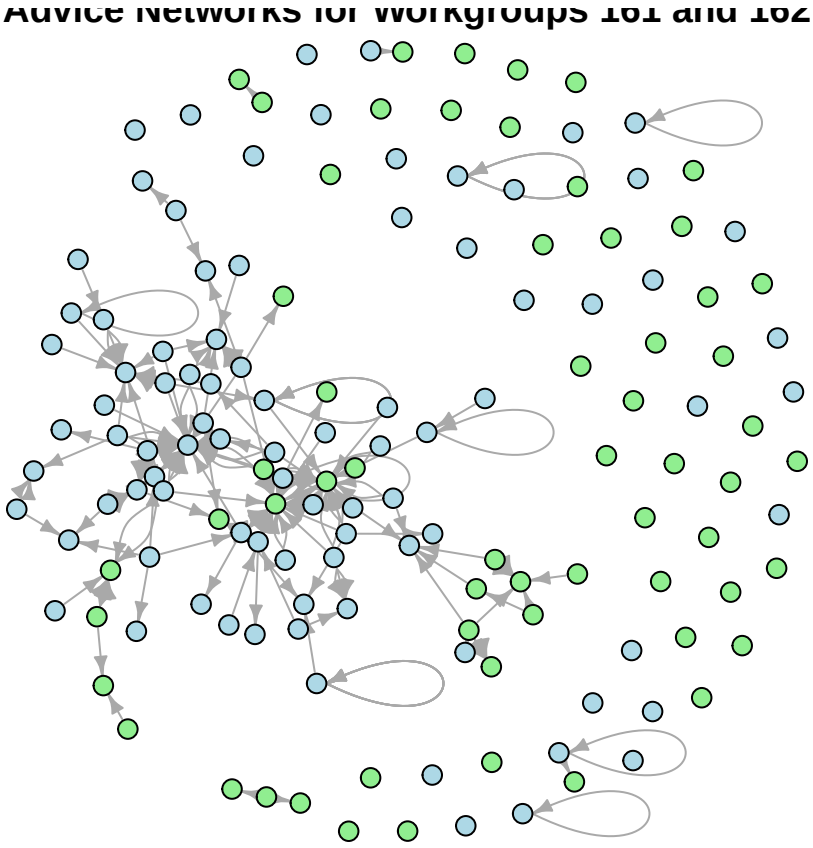
Next, we will create advice networks for the selected workgroups using the edges_sample dataset and calculate centrality scores for the examiners.

We will start with creating and plotting advice networks:

```
# Create an igraph object from the edges_sample data
g <- graph_from_data_frame(edges_sample[, c("ego_examiner_id", "alter_examiner_id")], directed = TRUE)
```

```
## Warning in graph_from_data_frame(edges_sample[, c("ego_examiner_id",
## "alter_examiner_id")], : In 'd' 'NA' elements were replaced with string "NA"
```

```
# Extract the first 3 digits of examiner_art_unit values
applications$workgroup <- substr(applications$examiner_art_unit, 1, 3)

# Create a mapping between examiner_id and workgroup in the applications dataset
examiner_workgroup_mapping <- applications %>%
  select(examiner_id, workgroup) %>%
  distinct()

# Add attributes to vertices in the network
V(g)$workgroup <- examiner_workgroup_mapping$workgroup[match(V(g)$name, examiner_workgroup_mapping$exami

# Filter the network to only include the two selected workgroups
g_filtered <- g %>%
  induced_subgraph(V(g)[V(g)$workgroup %in% c(workgroup1, workgroup2)])

# Set plot options
par(mar = c(0, 0, 0, 0))
set.seed(123)

# Create the plot
plot(g_filtered,
     vertex.color = ifelse(V(g_filtered)$workgroup == workgroup1, "lightblue", "lightgreen"),
     vertex.label = NA,
     vertex.size = 5,
     edge.arrow.size = 0.5,
     main = "Advice Networks for Workgroups 161 and 162")
```



Now, we will calculate centrality scores for examiners in selected workgroups.

Since we need to ensure examiner_id has the same data type in both data frames, we will convert examiner_id to numeric in examiner_workgroup_mapping.

```
# Create a mapping between examiner_id and workgroup in the applications dataset
examiner_workgroup_mapping <- applications %>%
  select(examiner_id, workgroup) %>%
  mutate(examiner_id = as.numeric(examiner_id)) %>% # Convert examiner_id to numeric
  distinct()
```

For this exercise, let's use Degree Centrality and Betweenness Centrality. Degree Centrality measures the number of direct connections an examiner has, while Betweenness Centrality measures the extent to which an examiner lies on the shortest paths between other examiners in the network. Both measures can help us identify influential or well-connected examiners in the network.

```
# Calculate Degree Centrality and Betweenness Centrality
degree_centrality <- degree(g_filtered, mode = "all")
betweenness_centrality <- betweenness(g_filtered, directed = TRUE)

# Add the centrality scores to the vertex attributes
V(g_filtered)$degree_centrality <- degree_centrality
V(g_filtered)$betweenness_centrality <- betweenness_centrality

# Merge centrality scores with the examiners' characteristics
centrality_scores <- data.frame(
  examiner_id = as.numeric(V(g_filtered)$name), # Convert examiner_id to numeric
```

```
  workgroup = V(g_filtered)$workgroup,
  degree_centrality = V(g_filtered)$degree_centrality,
  betweenness_centrality = V(g_filtered)$betweenness_centrality
)

applications_centrality <- applications %>%
  select(examiner_id, gender, race, tenure_days) %>%
  mutate(examiner_id = as.numeric(examiner_id)) %>% # Convert examiner_id to numeric
  inner_join(centrality_scores, by = "examiner_id")

# Examine the results
print(applications_centrality)
```

```
## # A tibble: 100,951 x 7
##    examiner_id gender race  tenure_days workgroup degree_centrality betweennes~1
##          <dbl> <chr>  <chr>       <dbl> <chr>                 <dbl>        <dbl>
##  1       70017 female Asian        6283 162                       0            0
##  2       69138 <NA>   white        6348 161                       4            1
##  3       64839 male   white        6254 161                       0            0
##  4       94939 <NA>   Asian        6336 162                       0            0
##  5       65737 female white        6129 162                       9            0
##  6       95225 male   white        6332 162                       0            0
##  7       68694 male   white        6350 161                       2            0
##  8       90588 female white        6343 161                       1            0
##  9       65536 female Asian        6345 162                       1            0
## 10       59399 male   white        6339 161                       1            0
## # ... with 100,941 more rows, and abbreviated variable name
## #   1: betweenness_centrality
```

In this analysis, we have successfully loaded the data, added demographic variables, selected two workgroups, compared their demographics, and created advice networks with centrality scores for the examiners. This information can be used to explore the relationships between examiners' demographics and their advice networks.