Exercise_3

Hai Nguyen

2023-03-28

Load data

Load the following data: + applications from `app_data_sample.parquet` + edges from `edges_sample.csv`

```
applications <- read_parquet("C:/Users/nguye/OneDrive/Documents/MMA/Winter 
2023/Network 
analysis/672_project_data/app_data_sample.parquet",as_data_frame=TRUE)
edges <- read_csv("C:/Users/nguye/OneDrive/Documents/MMA/Winter 2023/Network 
analysis/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.
```

```
applications
```

```
## # A tibble: 2,018,477 × 16
##    applicat…¹ filing_d…² exami…³ exami…⁴ exami…⁵ exami…⁶ exami…⁷ uspc_…⁸ 
uspc_…⁹
##    <chr>      <date>     <chr>   <chr>   <chr>     <dbl>   <dbl> <chr>   
<chr>
##  1 08284457   2000-01-26 HOWARD  JACQUE… V         96082    1764 508     
273000
##  2 08413193   2000-10-11 YILDIR… BEKIR   L         87678    1764 208     
179000
##  3 08531853   2000-05-17 HAMILT… CYNTHIA <NA>      63213    1752 430     
271100
##  4 08637752   2001-07-20 MOSHER  MARY    <NA>      73788    1648 530     
388300
##  5 08682726   2000-04-10 BARR    MICHAEL E         77294    1762 427     
430100
##  6 08687412   2000-04-28 GRAY    LINDA   LAMEY     68606    1734 156     
204000
```

```
## 7 08716371   2004-01-26 MCMILL… KARA    RENITA    89557    1627 424
401000
## 8 08765941   2000-06-23 FORD    VANESSA L         97543    1645 424
001210
## 9 08776818   2000-02-04 STRZEL… TERESA  E         98714    1637 435
006000
## 10 08809677   2002-02-20 KIM     SUN     U         65530    1723 210
645000
## # … with 2,018,467 more rows, 7 more variables: patent_number <chr>,
## #   patent_issue_date <date>, abandon_date <date>, disposal_type <chr>,
## #   appl_status_code <dbl>, appl_status_date <chr>, tc <dbl>, and
abbreviated
## #   variable names ¹application_number, ²filing_date, ³examiner_name_last,
## #   ⁴examiner_name_first, ⁵examiner_name_middle, ⁶examiner_id,
## #   ⁷examiner_art_unit, ⁸uspc_class, ⁹uspc_subclass
```

```
edges
```

```
## # A tibble: 32,906 × 4
##    application_number advice_date ego_examiner_id alter_examiner_id
##    <chr>              <date>                <dbl>             <dbl>
##  1 09402488           2008-11-17            84356             66266
##  2 09402488           2008-11-17            84356             63519
##  3 09402488           2008-11-17            84356             98531
##  4 09445135           2008-08-21            92953             71313
##  5 09445135           2008-08-21            92953             93865
##  6 09445135           2008-08-21            92953             91818
##  7 09479304           2008-12-15            61767             69277
##  8 09479304           2008-12-15            61767             92446
##  9 09479304           2008-12-15            61767             66805
## 10 09479304           2008-12-15            61767             70919
## # … with 32,896 more rows
```

Get gender for examiners

We'll get gender based on the first name of the examiner, which is recorded in the field `examiner_name_first`. We'll use library `gender` for that, relying on a modified version of their own example.

Note that there are over 2 million records in the applications table – that's because there are many records for each examiner, as many as the number of applications that examiner worked on during this time frame. Our first step therefore is to get all *unique* names in a separate list `examiner_names`. We will then guess gender for each one and will join this table back to the original dataset. So, let's get names without repetition:

```
library(gender)
#install_genderdata_package() # only run this line the first time you use the
package, to get data for it
# get a list of first names without repetitions
examiner_names <- applications %>%
```

```
  distinct(examiner_name_first)
examiner_names
```

```
## # A tibble: 2,595 × 1
##    examiner_name_first
##    <chr>
##  1 JACQUELINE
##  2 BEKIR
##  3 CYNTHIA
##  4 MARY
##  5 MICHAEL
##  6 LINDA
##  7 KARA
##  8 VANESSA
##  9 TERESA
## 10 SUN
## # … with 2,585 more rows
```

Now let's use function `gender()` as shown in the example for the package to attach a gender and probability to each name and put the results into the table `examiner_names_gender`

```
# 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
  )
examiner_names_gender
```

```
## # A tibble: 1,822 × 3
##    examiner_name_first gender proportion_female
##    <chr>               <chr>              <dbl>
##  1 AARON               male              0.0082
##  2 ABDEL               male              0
##  3 ABDOU               male              0
##  4 ABDUL               male              0
##  5 ABDULHAKIM          male              0
##  6 ABDULLAH            male              0
##  7 ABDULLAHI           male              0
##  8 ABIGAIL             female            0.998
##  9 ABIMBOLA            female            0.944
## 10 ABRAHAM             male              0.0031
## # … with 1,812 more rows
```

Finally, let's join that table back to our original applications data and discard the temporary tables we have just created to reduce clutter in our environment.

```
# 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) max used  (Mb)
## Ncells  4671496 249.5    8229292 439.5  5114501 273.2
## Vcells 49878864 380.6   93055462 710.0 80194619 611.9
```

Guess the examiner’s race

We’ll now use package wru to estimate likely race of an examiner. Just like with gender, we’ll get a list of unique names first, only now we are using surnames.

```
library(wru)
examiner_surnames <- applications %>%
  select(surname = examiner_name_last) %>%
  distinct()
examiner_surnames
```

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

We’ll follow the instructions for the package outlined here https://github.com/kosukeimai/wru.

```
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 × 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
## # … with 3,796 more rows
```

As you can see, we get probabilities across five broad US Census categories: white, black, Hispanic, Asian and other. (Some of you may correctly point out that Hispanic is not a race category in the US Census, but these are the limitations of this package.)

Our final step here is to pick the race category that has the highest probability for each last name and then join the table back to the main applications table. See this example for comparing values across columns: https://www.tidyverse.org/blog/2020/04/dplyr-1-0-0-rowwise/. And this one for `case_when()` function: https://dplyr.tidyverse.org/reference/case_when.html.

```
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 × 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
## # … with 3,796 more rows
```

Let's join the data back to the applications table.

```
# removing extra columns
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) max used  (Mb)
## Ncells  4806713 256.8    8229292 439.5  8229292 439.5
## Vcells 54213238 413.7  111746554 852.6 92935921 709.1
```

Examiner's tenure

To figure out the timespan for which we observe each examiner in the applications data, let's find the first and the last observed date for each examiner. We'll first get examiner IDs and application dates in a separate table, for ease of manipulation. We'll keep examiner ID (the field `examiner_id`), and earliest and latest dates for each application (`filing_date` and `appl_status_date` respectively). We'll use functions in package `lubridate` to work with date and time values.

```
library(lubridate) # to work with dates
examiner_dates <- applications %>%
  select(examiner_id, filing_date, appl_status_date)
examiner_dates
```

```
## # A tibble: 2,018,477 × 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
## # … with 2,018,467 more rows
```

The dates look inconsistent in terms of formatting. Let's make them consistent. We'll create new variables `start_date` and `end_date`.

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

Let's now identify the earliest and the latest date for each examiner and calculate the difference in days, which is their tenure in the organization.

```
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 × 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
## # … with 5,615 more rows
```

Joining back to the applications data.

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

```
rm(examiner_dates)
gc()
```

```
##            used  (Mb) gc trigger   (Mb)  max used   (Mb)
## Ncells  4813327 257.1   14938879  797.9  14938879  797.9
## Vcells 64477940 492.0  134175864 1023.7 133649394 1019.7
```

Examiner's workgroup

Let's create workgroup for each examiner in the applications data

```
applications$workgroups <- substr(applications$examiner_art_unit, 1, 3)
```

Count the number of examiners in each workgroup

```
workgroups_count = applications %>%
  distinct(examiner_id, workgroups) %>%
  group_by(workgroups) %>%
  summarize(count = n()) %>%
  arrange(desc(count))
```

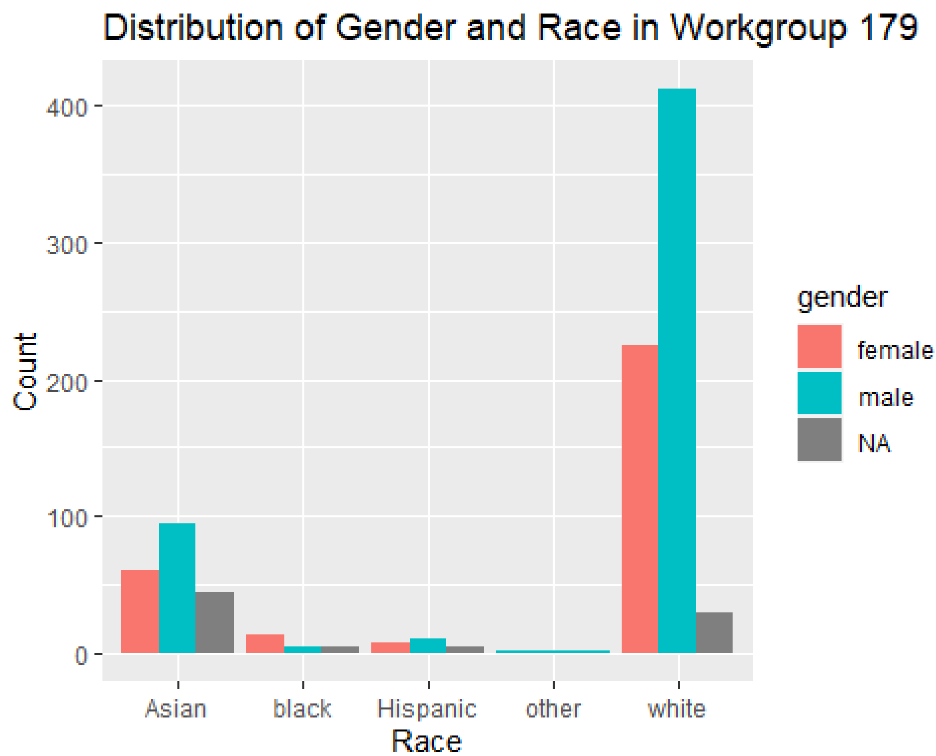
We will choose the top 2 workgroups with the largest number of examiner, which is 179 and 216

```
# Create a new table for applications with `workgroups` equal to 179
workgroup_179 <- applications %>%
  filter( workgroups == "179")

workgroup_179 <- distinct(workgroup_179, examiner_id, .keep_all = TRUE)

# Create a new table for applications with `workgroups` equal to 216
workgroup_216 <- applications %>%
  filter( workgroups == "216")

workgroup_216 <- distinct(workgroup_216, examiner_id, .keep_all = TRUE)
```

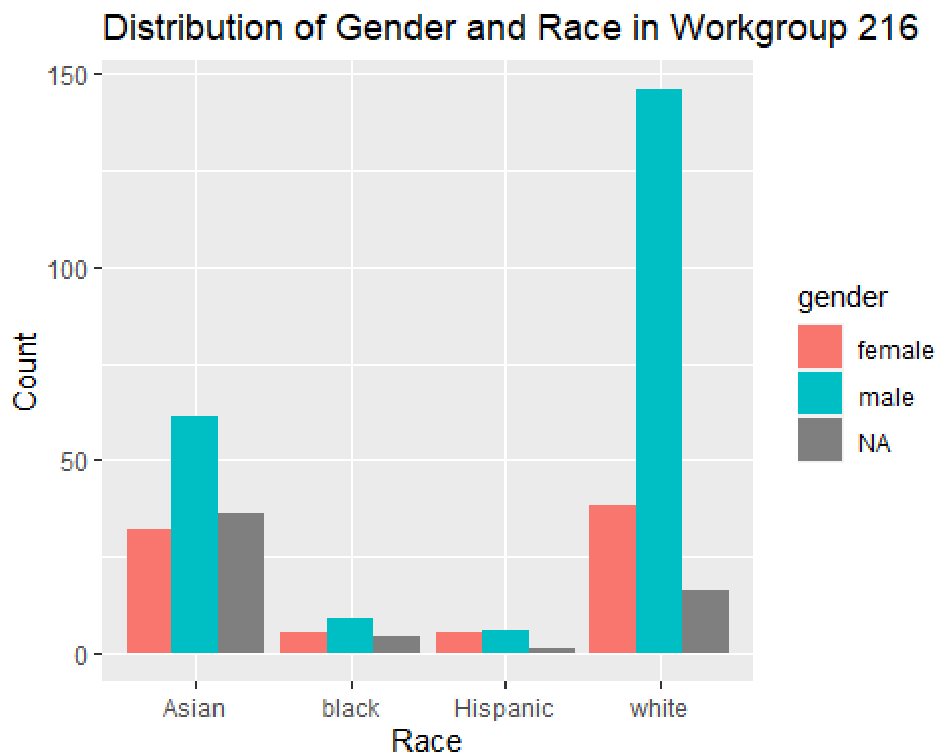
We will examine the demography of both workgroup

Workgroup 179:

```
ggplot(workgroup_179, aes(x = race, fill = gender)) +
  geom_bar(position = "dodge") +
  xlab("Race") +
  ylab("Count") +
  ggtitle("Distribution of Gender and Race in Workgroup 179")
```




```
table(workgroup_179$gender)

##
## female   male
##    308    523

table(workgroup_179$race)

##
##    Asian    black Hispanic    other    white
##      200       24       23        1      666

ggplot(workgroup_216, aes(x = race, fill = gender)) +
  geom_bar(position = "dodge") +
  xlab("Race") +
  ylab("Count") +
  ggtitle("Distribution of Gender and Race in Workgroup 216")
```




```
table(workgroup_216$gender)

##
## female   male
##     80    222

table(workgroup_216$race)

##
##    Asian    black Hispanic    white
##      129       18       12      200
```

In terms of gender, both groups have more male than female. In addition, if we look at the composition of gender within a specific race, it is indicated that for both Asian and White, male examiners outnumber their female peers.

In terms of race, Asian and White are the two dominating race in both workgroup. For 179, the number of white examiner is approximately 3.3 more than that of Asian examiner. Similarly, in 216, white examiners outnumber their Asian peers by 71 examiners. For both group, Black and Hispanic are underrepresented.

Network between examiner

We need to create a node dataframe with the examiner ID

```
#Create the nodes dataframe
nodes <- applications %>%
```

```
  distinct(examiner_id) %>%
  select(examiner_id)

nodes
```

```
## # A tibble: 5,649 × 1
##    examiner_id
##          <dbl>
##  1       96082
##  2       87678
##  3       63213
##  4       73788
##  5       77294
##  6       68606
##  7       89557
##  8       97543
##  9       98714
## 10       65530
## # … with 5,639 more rows
```

We then proceed to create a new edges dataframe by cleaning the original edges dataframe

```
#Subset examiner_id and workgroup columns from applications
examiner_workgroup <- subset(applications, select = c("examiner_id",
"workgroups"))

# Look up the workgroup corresponding to each ego_examiner_id in the
examiner_workgroup dataframe
edges$ego_examiner_workgroup <-
examiner_workgroup$workgroups[match(edges$ego_examiner_id,
examiner_workgroup$examiner_id)]

# Look up the workgroup corresponding to each alter_examiner_id in the
examiner_workgroup dataframe
edges$alter_examiner_workgroup <-
examiner_workgroup$workgroups[match(edges$alter_examiner_id,
examiner_workgroup$examiner_id)]

# Filter out observations where either ego_examiner_workgroup or
alter_examiner_workgroup is NA
edges_filtered <- edges[complete.cases(edges[, c("ego_examiner_id",
"alter_examiner_id", "ego_examiner_workgroup", "alter_examiner_workgroup")]),
]

#Create the edges dataframe

edges_1 <- edges_filtered %>%
  distinct(ego_examiner_id, alter_examiner_id) %>%
  select(ego_examiner_id, alter_examiner_id)
```

```
edges_1
```

```
## # A tibble: 6,385 × 2
##    ego_examiner_id alter_examiner_id
##              <dbl>             <dbl>
##  1           84356             66266
##  2           84356             63519
##  3           84356             98531
##  4           92953             93865
##  5           92953             91818
##  6           72253             61519
##  7           72253             72253
##  8           72253             67515
##  9           67078             75772
## 10           67078             97328
## # … with 6,375 more rows
```

Let's create a network graph between the examiners

```
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
```

```
# Create network graph
g <- graph_from_data_frame(d = edges_1, directed = FALSE, vertices = nodes)

## Warning in graph_from_data_frame(d = edges_1, directed = FALSE, vertices =
## nodes): In `vertices[,1]' `NA' elements were replaced with string "NA"
```

For the purpose of this project, I will measure degree centrality. With the large number of employees in the organization, the person with the most connections will be influential as their connection could help them in their projects. Looking from a project perspective, knowing a lot of people will give you an advantage.

Let's calculate the degree centrality for all examiner_id

```
# Calculate the degree centrality of each node (examiner ID)
centrality <- degree(g, mode = "all", normalized = FALSE)

# Combine the centrality scores and node IDs into a table
centrality_table <- data.frame(examiner_id = V(g)$name, centrality =
as.vector(centrality))
centrality_table$examiner_id <- as.numeric(centrality_table$examiner_id)

## Warning: NAs introduced by coercion
```

Let's calculate the degree centrality for each examiner in group 179

```
workgroup_179 <- left_join(workgroup_179, centrality_table, by =
"examiner_id")
workgroup_179 <- arrange(workgroup_179, desc(centrality))
workgroup_179

## # A tibble: 914 × 23
##    applicat…¹ filing_d…² exami…³ exami…⁴ exami…⁵ exami…⁶ exami…⁷ uspc_…⁸
uspc_…⁹
##    <chr>      <date>     <chr>   <chr>   <chr>     <dbl>   <dbl> <chr>
<chr>
##  1 09902475   2001-07-09 DAVIS   ROBERT  B         63176    1791 425
195000
##  2 09717374   2000-11-22 HENDRI… STUART  L         67698    1793 502
417000
##  3 10531525   2005-11-02 INYARD  APRIL   C         79847    1794 428
221000
##  4 09727516   2000-12-04 ALANKO  ANITA   KAREN     71119    1792 216
083000
##  5 09849065   2001-05-04 PADGETT MARIAN… L         75341    1792 427
569000
##  6 09731945   2000-12-07 DANIELS MATTHEW J         59771    1791 264
257000
##  7 10399396   2003-04-21 JOY     DAVID   J         80730    1794 428
292100
##  8 10362305   2003-05-21 PARVINI PEGAH   <NA>      66436    1793 106
487000
```

```
## 9 10402745 2003-03-28 DAHIME… MAHMOUD <NA>      93432    1792 438
710000
## 10 10523571 2006-02-27 BELL    WILLIAM P          76154    1791 264
243000
## # … with 904 more rows, 14 more variables: 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>,
## #   workgroups <chr>, centrality <dbl>, and abbreviated variable names
## #   ¹application_number, ²filing_date, ³examiner_name_last,
## #   ⁴examiner_name_first, ⁵examiner_name_middle, ⁶examiner_id, …
```

Let's calculate the degree centrality for each examiner in group 216

```
workgroup_216 <- left_join(workgroup_216, centrality_table, by =
"examiner_id")
workgroup_216 <- arrange(workgroup_216, desc(centrality))
workgroup_216
```

```
## # A tibble: 359 × 23
##    applicat…¹ filing_d…² exami…³ exami…⁴ exami…⁵ exami…⁶ exami…⁷ uspc_…⁸
uspc_…⁹
##    <chr>      <date>     <chr>   <chr>   <chr>     <dbl>   <dbl> <chr>
<chr>
##  1 09564248   2000-05-04 PHAM    KHANH   B         78326    2166 707
204000
##  2 09580327   2000-05-26 WASSUM  LUKE    S         99902    2167 707
010000
##  3 10481715   2004-06-01 WONG    JOSEPH  D         93865    2166 707
104100
##  4 09602441   2000-06-23 ALAM    HOSAIN  T         92060    2166 707
101000
##  5 09507064   2000-02-18 ABEL J… NEVEEN  <NA>      83222    2165 707
100000
##  6 11071937   2005-03-04 LIAO    JASON   G         70672    2169 707
002000
##  7 09782596   2001-02-12 STEVENS ROBERT  <NA>      71230    2162 715
517000
##  8 09479669   2000-01-10 COBY    FRANTZ  <NA>      60130    2161 707
007000
##  9 09656484   2000-09-07 LU      KUEN    S         83313    2167 707
010000
## 10 09505165   2000-02-16 ALI     MOHAMM… <NA>      67404    2167 707
001000
## # … with 349 more rows, 14 more variables: 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>,
## #   workgroups <chr>, centrality <dbl>, and abbreviated variable names
## #   ¹application_number, ²filing_date, ³examiner_name_last,
## #   ⁴examiner_name_first, ⁵examiner_name_middle, ⁶examiner_id, …
```

In group 179, the top 10 examiners with the highest centrality score are all white, with 4 females and 6 males. This could be explained by the fact that white is the dominating race in the group, and subconsciously people would make connection with people with similar backgrounds, in this case, of white race. On the other hand, in group 216, the top 10 examiners with the highest centrality score is mostly minorities (6 Asians, 1 black). This is interesting as Asian is not the major race in the group. This could be due to the difference in the topic that each workgroup deal with, and the people with the highest centrality score are those that are good at these topics.