# Exercise 3 - Ruhi Mahendra

## Load data

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

```
# change to your own path!
data_path <- "~/Desktop/MMA/2023-ona-assignments/"
applications <- read_parquet(paste0(data_path,"app_data_sample.parquet"))
edges <- read_csv(paste0(data_path,"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) limit (Mb) max used  (Mb)
## Ncells  4675327 249.7    8232162 439.7         NA  5124738 273.7
## Vcells 49926905 381.0   95923350 731.9      16384 80242732 612.3
```

## 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) limit (Mb) max used  (Mb)
## Ncells   4829520 258.0    8232162 439.7         NA  8232162 439.7
## Vcells 54322818 414.5   95923350 731.9      16384 95288723 727.0
```

## 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) limit (Mb)  max used   (Mb)
## Ncells   4835200 258.3   14941738  798.0         NA  14941738  798.0
## Vcells 64585682 492.8  138305624 1055.2      16384 138240253 1054.7
```

Pick two workgroups you want to focus on (remember that a workgroup is represented by the first 3 digits of `examiner_art_unit` value) How do they compare on examiners' demographics? Show summary statistics and plots.

```
New  <- applications %>% select(examiner_art_unit, gender, race, tenure_days)
New$examiner_art_unit = substr(New$examiner_art_unit, 1, 3)

workgroup_171 <- New %>% filter(examiner_art_unit == 171)

workgroup_174 <- New %>% filter(examiner_art_unit == 174)
```

summary statistics workgroup_171

```
library(vtable)
```

```
## Loading required package: kableExtra

## Warning in !is.null(rmarkdown::metadata$output) && rmarkdown::metadata$output
## %in% : 'length(x) = 2 > 1' in coercion to 'logical(1)'

##
## Attaching package: 'kableExtra'

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

```
st(workgroup_171)
```

Summary Statistics

| Variable | N | Mean | Std. Dev. | Min | Pctl. 25 | Pctl. 75 | Max |
|---|---|---|---|---|---|---|---|

| Variable | N | Mean | Std. Dev. | Min | Pctl. 25 | Pctl. 75 | Max |
|---|---|---|---|---|---|---|---|
| examiner_art_unit | 76544 | | | | | | |
| … 171 | 76544 | 100% | | | | | |
| gender | 69484 | | | | | | |
| … female | 22175 | 32% | | | | | |
| … male | 47309 | 68% | | | | | |
| race | 76544 | | | | | | |
| … Asian | 17182 | 22% | | | | | |
| … black | 3889 | 5% | | | | | |
| … Hispanic | 2081 | 3% | | | | | |
| … white | 53392 | 70% | | | | | |
| tenure_days | 75847 | 5530 | 1139 | 550 | 4863 | 6342 | 6350 |

plot for workgroup_171

```
ggplot(workgroup_171, aes(x = race, fill = gender)) +
  geom_bar(position = "dodge") +
  xlab("Race") +
  ylab("Count") +
  ggtitle("Distribution of Gender and Race in Workgroup 171")
```


summary statistics workgroup_174

```
library(vtable)
st(workgroup_174)
```

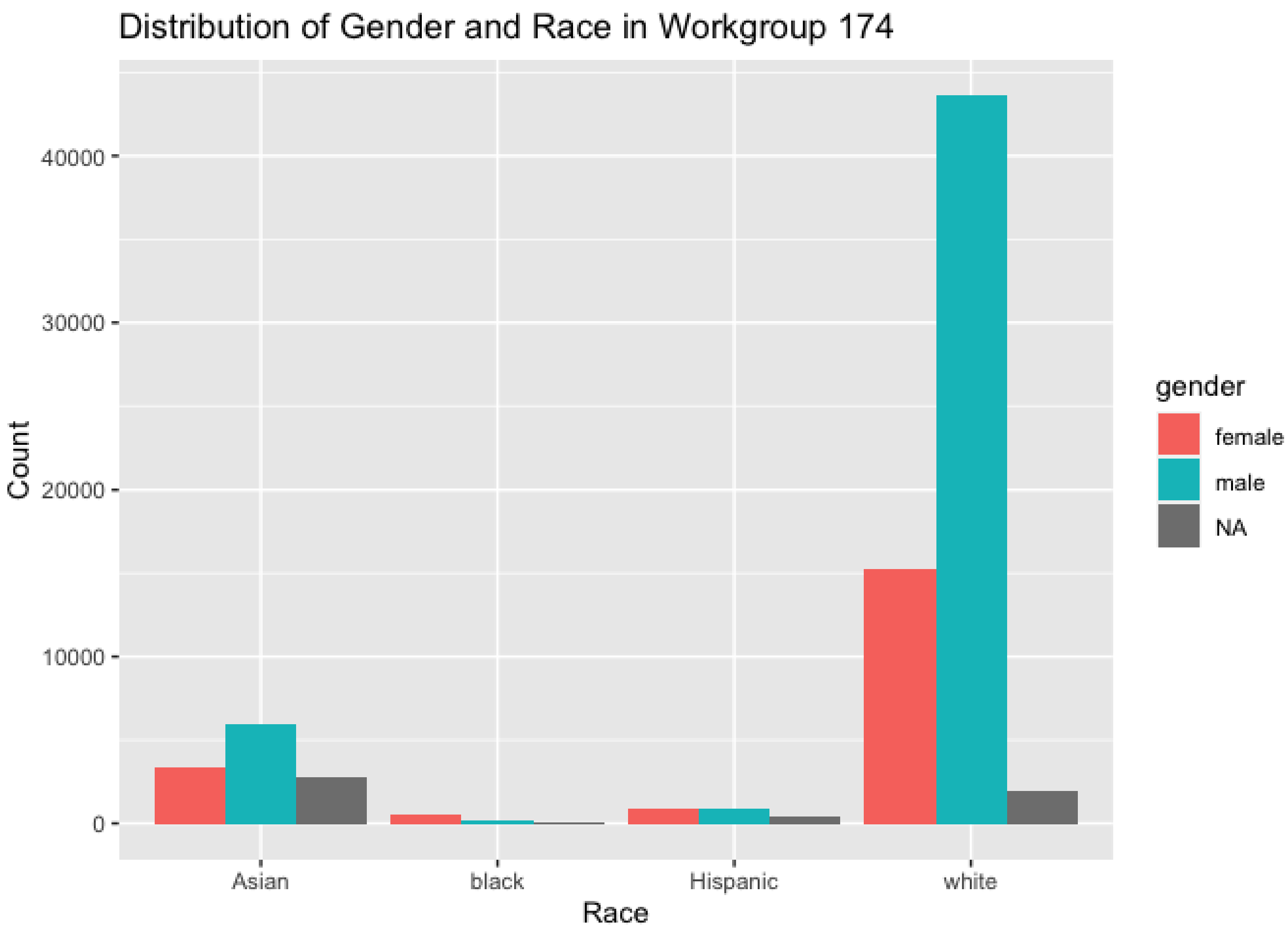
Summary Statistics

| Variable | N | Mean | Std. Dev. | Min | Pctl. 25 | Pctl. 75 | Max |
|---|---|---|---|---|---|---|---|
| examiner_art_unit | 75598 | | | | | | |
| … 174 | 75598 | 100% | | | | | |

| Variable | N | Mean | Std. Dev. | Min | Pctl. 25 | Pctl. 75 | Max |
|---|---|---|---|---|---|---|---|
| gender | 70583 | | | | | | |
| ... female | 20012 | 28% | | | | | |
| ... male | 50571 | 72% | | | | | |
| race | 75598 | | | | | | |
| ... Asian | 12111 | 16% | | | | | |
| ... black | 627 | 1% | | | | | |
| ... Hispanic | 2137 | 3% | | | | | |
| ... white | 60723 | 80% | | | | | |
| tenure_days | 74199 | 5645 | 1026 | 464 | 4879 | 6342 | 6350 |

plot for workgroup_174

```
ggplot(workgroup_174, aes(x = race, fill = gender)) +
  geom_bar(position = "dodge") +
  xlab("Race") +
  ylab("Count") +
  ggtitle("Distribution of Gender and Race in Workgroup 174")
```




Create advice networks from `edges_sample` and calculate centrality scores for examiners in your selected workgroups

Create nodes

```
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
```

create edges

```
applications$workgroups <- substr(applications$examiner_art_unit, 1, 3)
examiner_workgroup <- subset(applications, select = c("examiner_id", "workgroups"))
edges$ego_examiner_workgroup <- examiner_workgroup$workgroups[match(edges$ego_examiner_id, examiner_workgrc
# Look up the workgroup corresponding to each alter_examiner_id in the examiner_workgroup dataframe
edges$alter_examiner_workgroup <- examiner_workgroup$workgroups[match(edges$alter_examiner_id, examiner_wor
# 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_work
#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
```

create network graph

```
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"
```

Get all centralities for network

```
# Calculate the degree centrality of each node (examiner ID)
degree_centrality <- degree(g, mode = "all", normalized = FALSE)
betweeness_centrality <- betweenness(g)
closeness_centrality <- closeness(g, mode = "all", normalized = FALSE)
# Combine the centrality scores and node IDs into a table
centrality_table <- data.frame(examiner_id = V(g)$name, degree = as.vector(degree_centrality), betweeness =
closeness = as.vector(closeness_centrality))

centrality_table$examiner_id <- as.numeric(centrality_table$examiner_id)
```

```
## Warning: NAs introduced by coercion
```

```
New  <- applications %>% select(examiner_id,examiner_art_unit, gender, race, tenure_days, filing_date, pate

all_centrality <- left_join(New, centrality_table, by = "examiner_id")
```

```r
all_centrality <- all_centrality %>% mutate(app_proc_time =
  case_when(!is.na(patent_issue_date) ~ patent_issue_date - filing_date,
            !is.na(abandon_date) ~ abandon_date - filing_date
              )
)



#degree
model_1 <- lm(app_proc_time ~ degree + race + gender + tenure_days, data = all_centrality)

#betweeness
model_2 <- lm(app_proc_time ~ betweeness + race + gender + tenure_days, data = all_centrality)

#closeness
model_3 <- lm(app_proc_time ~ closeness + race + gender + tenure_days, data = all_centrality)

model_1
```

```
##
## Call:
## lm(formula = app_proc_time ~ degree + race + gender + tenure_days,
##     data = all_centrality)
##
## Coefficients:
##  (Intercept)        degree     raceblack  raceHispanic     raceother
##   1296.53464       6.23462     -12.00902     -22.36250     192.87115
##    racewhite    gendermale   tenure_days
##    -64.21158      11.03222      -0.01566
```

```r
model_2
```

```
##
## Call:
```

```
## lm(formula = app_proc_time ~ betweeness + race + gender + tenure_days,
##     data = all_centrality)
##
## Coefficients:
##  (Intercept)    betweeness     raceblack  raceHispanic     raceother
##    1.284e+03     1.842e-03    -1.436e+01    -2.632e+01     1.987e+02
##    racewhite    gendermale   tenure_days
##   -6.725e+01     1.471e+01    -1.083e-02
```

```
model_3
```

```
##
## Call:
## lm(formula = app_proc_time ~ closeness + race + gender + tenure_days,
##     data = all_centrality)
##
## Coefficients:
##  (Intercept)     closeness     raceblack  raceHispanic     raceother
##   1534.18048     -45.09759      -2.06510      -4.43287     169.66668
##    racewhite    gendermale   tenure_days
##    -70.22125      16.17924      -0.04794
```

Degree Centrality: Degree centrality increases processing time as the centrality is higher. Being black, Hispanic or white decreases the processing time. Being a male increases the processing time.

Betweeness Centrality: Betweeness centrality increases processing time as the centrality is higher. Being black, Hispanic or white decreases the processing time. Being a male increases the processing time.

Closeness Centrality: Closeness centrality decreases processing time as the centrality gets higher. Being black, Hispanic or white decreases the processing time. Being a male increases the processing time.

```
#degree
model_4 <- lm(app_proc_time ~ degree*gender + race + tenure_days, data = all_centrality)
```

```r
#betweeness
model_5 <- lm(app_proc_time ~ betweeness*gender + race + tenure_days, data = all_centrality)

#closeness
model_6 <- lm(app_proc_time ~ closeness*gender + race + tenure_days, data = all_centrality)

model_4
```

```
##
## Call:
## lm(formula = app_proc_time ~ degree * gender + race + tenure_days,
##     data = all_centrality)
##
## Coefficients:
##       (Intercept)             degree          gendermale          raceblack
##        1300.30362            4.89221             5.67112          -12.34964
##      raceHispanic          raceother           racewhite        tenure_days
##         -23.21556          191.95084           -64.53331           -0.01563
## degree:gendermale
##           1.81804
```

```r
model_5
```

```
##
## Call:
## lm(formula = app_proc_time ~ betweeness * gender + race + tenure_days,
##     data = all_centrality)
##
## Coefficients:
##           (Intercept)             betweeness             gendermale
##             1.286e+03              8.995e-04              1.119e+01
##             raceblack           raceHispanic              raceother
##            -1.469e+01             -2.662e+01              1.990e+02
```

```
##             racewhite            tenure_days  betweeness:gendermale
##            -6.713e+01             -1.082e-02             1.258e-03
```

```
model_6
```

```
##
## Call:
## lm(formula = app_proc_time ~ closeness * gender + race + tenure_days,
##     data = all_centrality)
##
## Coefficients:
##          (Intercept)              closeness             gendermale
##           1534.00283              -70.42145               15.67776
##            raceblack           raceHispanic              raceother
##             -2.28232               -4.53611              169.67887
##            racewhite            tenure_days  closeness:gendermale
##            -70.41362               -0.04783               38.00754
```

Degree Centrality: Degree centrality increases processing time as the centrality is higher. Being black, Hispanic or white decreases the processing time. Being a male increases the processing time. Having interactions between centrality and gender, lowers the processing time as centrality becomes higher and gender is male versus not having any interactions.

Betweeness Centrality: Betweeness centrality increases processing time as the centrality is higher. Being black, Hispanic or white decreases the processing time. Being a male increases the processing time. Having interactions between centrality and gender, increases the processing time as the centrality becomes higher versus not having any interactions.

Closeness Centrality: Closeness centrality decreases processing time as the centrality gets higher. Being black, Hispanic or white decreases the processing time. Being a male increases the processing time. Having interactions between centrality and gender increases the processing time as centrality becomes higher and gender is male versus not having any interactions.