Exercise 4

Load data

```
# change to your own path!
data_path <- "C:/Users/mattl/OneDrive/Documents/Education/Masters/Courses/Spring Semester/Organizational
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 x 16
##    application_number filing_date examiner_name_last examiner_name_first
##    <chr>              <date>      <chr>              <chr>
##  1 08284457           2000-01-26  HOWARD             JACQUELINE
##  2 08413193           2000-10-11  YILDIRIM           BEKIR
##  3 08531853           2000-05-17  HAMILTON           CYNTHIA
##  4 08637752           2001-07-20  MOSHER             MARY
##  5 08682726           2000-04-10  BARR               MICHAEL
##  6 08687412           2000-04-28  GRAY               LINDA
##  7 08716371           2004-01-26  MCMILLIAN          KARA
##  8 08765941           2000-06-23  FORD               VANESSA
##  9 08776818           2000-02-04  STRZELECKA         TERESA
## 10 08809677           2002-02-20  KIM                SUN
## # ... with 2,018,467 more rows, and 12 more variables:
## #   examiner_name_middle <chr>, examiner_id <dbl>, examiner_art_unit <dbl>,
## #   uspc_class <chr>, uspc_subclass <chr>, patent_number <chr>,
## #   patent_issue_date <date>, abandon_date <date>, disposal_type <chr>,
## #   appl_status_code <dbl>, appl_status_date <chr>, tc <dbl>
```

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:

```
#install_genderdata_package() # only run this line the first time you use the package, to get data for

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

#Now let's use function `gender()` as shown in the example for the package to attach a gender and proba

# 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 x 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 table


# remove extra columns 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  4815461 257.2    8039874 429.4  5177008 276.5
## Vcells 50061480 382.0   93358455 712.3 80377241 613.3
```

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.

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

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

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

```
## [1] "Proceeding with surname-only predictions..."

## Warning in merge_surnames(voter.file): Probabilities were imputed for 698
## surnames that could not be matched to Census list.
```

```
examiner_race
```

```
## # A tibble: 3,806 x 6
##    surname    pred.whi pred.bla pred.his pred.asi pred.oth
##    <chr>         <dbl>    <dbl>    <dbl>    <dbl>    <dbl>
##  1 HOWARD       0.643   0.295    0.0237   0.005    0.0333
##  2 YILDIRIM     0.861   0.0271   0.0609   0.0135   0.0372
##  3 HAMILTON     0.702   0.237    0.0245   0.0054   0.0309
##  4 MOSHER       0.947   0.00410  0.0241   0.00640  0.0185
##  5 BARR         0.827   0.117    0.0226   0.00590  0.0271
##  6 GRAY         0.687   0.251    0.0241   0.0054   0.0324
##  7 MCMILLIAN    0.359   0.574    0.0189   0.00260  0.0463
##  8 FORD         0.620   0.32     0.0237   0.0045   0.0313
##  9 STRZELECKA   0.666   0.0853   0.137    0.0797   0.0318
## 10 KIM          0.0252  0.00390  0.00650  0.945    0.0198
## # ... with 3,796 more rows
```

```
#As you can see, we get probabilities across five broad US Census categories: white, black, Hispanic, A

#Our final step here is to pick the race category that has the highest probability for each last name a

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

```
## # A tibble: 3,806 x 8
##    surname    pred.whi pred.bla pred.his pred.asi pred.oth max_race_p race
##    <chr>         <dbl>    <dbl>    <dbl>    <dbl>    <dbl>      <dbl> <chr>
##  1 HOWARD       0.643   0.295    0.0237   0.005    0.0333      0.643 white
##  2 YILDIRIM     0.861   0.0271   0.0609   0.0135   0.0372      0.861 white
##  3 HAMILTON     0.702   0.237    0.0245   0.0054   0.0309      0.702 white
##  4 MOSHER       0.947   0.00410  0.0241   0.00640  0.0185      0.947 white
##  5 BARR         0.827   0.117    0.0226   0.00590  0.0271      0.827 white
##  6 GRAY         0.687   0.251    0.0241   0.0054   0.0324      0.687 white
##  7 MCMILLIAN    0.359   0.574    0.0189   0.00260  0.0463      0.574 black
##  8 FORD         0.620   0.32     0.0237   0.0045   0.0313      0.620 white
##  9 STRZELECKA   0.666   0.0853   0.137    0.0797   0.0318      0.666 white
## 10 KIM          0.0252  0.00390  0.00650  0.945    0.0198      0.945 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  5155084 275.4    8039874 429.4  6183262 330.3
## Vcells 53747594 410.1   93358455 712.3 92712276 707.4
```

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.

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

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

```
# The dates look inconsistent in terms of formatting. Let's make them consistent. We'll create new vari


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


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

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

```
##  9       59074 2000-01-31    2017-03-17         6255
## 10       59081 2011-04-21    2017-05-19         2220
## # ... 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  5169252 276.1   14332328  765.5  14332328  765.5
## Vcells 66126675 504.6  134612175 1027.1 134358658 1025.1
```

Application Processing Time

```
# Since an application can only be issued or abandoned, one or the other will always be NA, therefore I

applications$appl_end_date <- paste(applications$patent_issue_date, applications$abandon_date, sep=',')

# Then I will clean up the column by removing instances of commas and NA's
applications$appl_end_date <- gsub('NA', "", as.character(applications$appl_end_date))
applications$appl_end_date <- gsub(',', "", as.character(applications$appl_end_date))

# Ensure date format is consistent for both columns
applications$appl_end_date <- as.Date(applications$appl_end_date, format="%Y-%m-%d")
applications$filing_date <- as.Date(applications$filing_date, format="%Y-%m-%d")

# Finding the difference in days between the application end date and the filing date
applications$appl_proc_days <- as.numeric(difftime(applications$appl_end_date, applications$filing_date

# Remove instances where the filing date happens after the issue or abandon dates (these must be mistak
applications <- applications %>% filter(appl_proc_days >=0 | appl_proc_days != NA)
```

Additional Data Cleaning Prior to Modelling

```
# Find the count of missing values in each column
sapply(applications, function(x) sum(is.na(x)))
```

```
##   application_number          filing_date   examiner_name_last
##                    0                    0                    0
##  examiner_name_first examiner_name_middle          examiner_id
##                    0               390396                 3746
##    examiner_art_unit           uspc_class       uspc_subclass
##                    0                    4                 1555
##        patent_number    patent_issue_date         abandon_date
##               601857               601383              1087295
##        disposal_type     appl_status_code     appl_status_date
```

```
##                   0                 355                 356
##                  tc              gender                race
##                   0              253871                   0
##        earliest_date         latest_date         tenure_days
##                18240               18240               18240
##        appl_end_date      appl_proc_days
##                   0                   0
```

```
# Remove unnecessary columns for modelling
applications_mod <- subset(applications, select = -c(filing_date, abandon_date, earliest_date, appl_end

sapply(applications_mod, function(x) sum(is.na(x)))
```

```
##  application_number  examiner_name_last examiner_name_first         examiner_id
##                   0                   0                   0                3746
##   examiner_art_unit          uspc_class       uspc_subclass       disposal_type
##                   0                   4                1555                   0
##    appl_status_code                  tc              gender                race
##                 355                   0              253871                   0
##         tenure_days      appl_proc_days
##               18240                   0
```

```
applications_mod <- applications_mod %>% drop_na(examiner_id)
```

Use mice to impute missing values

```
applications_mod$gender <- as.factor(applications_mod$gender)

applications_mod_imp <- complete(mice(applications_mod, m=3, maxit=3))
```

```
##
##  iter imp variable
##   1   1  appl_status_code  gender  tenure_days
##   1   2  appl_status_code  gender  tenure_days
##   1   3  appl_status_code  gender  tenure_days
##   2   1  appl_status_code  gender  tenure_days
##   2   2  appl_status_code  gender  tenure_days
##   2   3  appl_status_code  gender  tenure_days
##   3   1  appl_status_code  gender  tenure_days
##   3   2  appl_status_code  gender  tenure_days
##   3   3  appl_status_code  gender  tenure_days

## Warning: Number of logged events: 7
```

```
rm(applications_mod)
```

Generating Network Information

```r
# first get work group for each examiner and limit to our two wgs of interest
examiner_aus = distinct(subset(applications_mod_imp, select=c(examiner_art_unit, examiner_id)))

# we eventually want to make a network with nodes colored by work group, so lets add that indicator
examiner_aus$wg = substr(examiner_aus$examiner_art_unit, 1,3)

# restrict down to our selected art units to reduce merging complexity later on
examiner_aus = examiner_aus[examiner_aus$wg==162 | examiner_aus$wg==219,]

# now we will merge in the aus df on applications
adv_network = merge(x=edges, y=examiner_aus, by.x="ego_examiner_id", by.y="examiner_id", all.x=TRUE)
adv_network = adv_network %>% rename(ego_art_unit=examiner_art_unit, ego_wg=wg)

adv_network = drop_na(adv_network)

# now repeat for the alter examiners
adv_network = merge(x=adv_network, y=examiner_aus, by.x="alter_examiner_id", by.y="examiner_id", all.x="
adv_network = adv_network %>% rename(alter_art_unit=examiner_art_unit, alter_wg=wg)
adv_network = drop_na(adv_network)
```

```r
# we are left with 870 edges corresponding to instances of examiners in wg173 or wg175 asking for advic
egoNodes = subset(adv_network, select=c(ego_examiner_id,ego_art_unit, ego_wg)) %>% rename(examiner_id=eg
alterNodes = subset(adv_network, select=c(alter_examiner_id,alter_art_unit, alter_wg))%>% rename(examin
nodes = rbind(egoNodes, alterNodes)
nodes = distinct(nodes)

# problem: when we reduce to the list of distinct nodes, we actually have more than we should, since so
nodes = nodes %>% group_by(examiner_id) %>% summarise(examiner_id=first(examiner_id), art_unit=first(ar
```

```r
network <- graph_from_data_frame(d=adv_network, vertices=nodes, directed=TRUE)
network
```

```
## IGRAPH 761341c DN-- 98 916 --
## + attr: name (v/c), art_unit (v/n), wg (v/c), application_number (e/c),
## | advice_date (e/n), ego_art_unit (e/n), ego_wg (e/c), alter_art_unit
## | (e/n), alter_wg (e/c)
## + edges from 761341c (vertex names):
##  [1] 59491->76935 59491->76935 59491->76935 59491->76935 59491->76935
##  [6] 59491->76935 59491->76935 59491->76935 59491->76935 61889->88905
## [11] 61889->88905 62114->72495 62114->66534 62114->72495 62114->66534
## [16] 62253->67690 62253->67690 62253->67690 62253->67690 63657->73150
## [21] 63657->73150 63657->73150 63657->73150 63657->73150 63657->73150
## [26] 63822->61417 64904->96780 65111->65111 65111->65111 65111->65111
## + ... omitted several edges
```

```r
# Calculate the node metrics
Degree <- degree(network)
Closeness <- closeness(network)
Betweenness <- betweenness(network)
Eig <- evcent(network)$vector
```

```
comp <- data.frame(nodes, Degree, Eig, Closeness, Betweenness)
comp
```

```
##       examiner_id art_unit  wg Degree          Eig  Closeness Betweenness
## 59491       59491     2196 219     48 2.644753e-02 1.00000000        0.90
## 59664       59664     2193 219      3 1.875528e-03        NaN        0.00
## 60302       60302     1626 162      6 0.000000e+00        NaN        0.00
## 60465       60465     2193 219      5 1.897166e-03        NaN        0.00
## 60768       60768     2191 219      2 8.131513e-05        NaN        0.00
## 61064       61064     2192 219      5 1.221863e-04        NaN        0.00
## 61417       61417     1626 162      2 0.000000e+00        NaN        0.00
## 61529       61529     1628 162      3 0.000000e+00        NaN        0.00
## 61889       61889     2193 219     24 9.429415e-03 1.00000000        4.00
## 61980       61980     2195 219      1 1.856425e-05        NaN        0.00
## 62114       62114     2195 219      4 3.187399e-05 0.50000000        0.00
## 62253       62253     1623 162      5 0.000000e+00 1.00000000        0.00
## 62661       62661     1627 162     13 0.000000e+00        NaN        0.00
## 63657       63657     2196 219     87 2.804044e-03 1.00000000        0.00
## 63822       63822     1624 162      1 0.000000e+00 1.00000000        0.00
## 63971       63971     2192 219      2 5.780753e-05        NaN        0.00
## 64904       64904     2192 219      1 1.592129e-05 1.00000000        0.00
## 65111       65111     1623 162     20 0.000000e+00 1.00000000        0.00
## 65353       65353     2193 219      9 2.027090e-04        NaN        0.00
## 65536       65536     1625 162      1 0.000000e+00 1.00000000        0.00
## 65537       65537     1624 162      3 0.000000e+00 1.00000000        0.00
## 65554       65554     2194 219      4 3.320903e-02        NaN        0.00
## 65713       65713     1623 162     12 0.000000e+00 0.50000000        0.00
## 65737       65737     1621 162      1 0.000000e+00 1.00000000        0.00
## 66030       66030     2193 219     19 3.703110e-04 0.10000000        0.00
## 66359       66359     2194 219      6 1.844651e-05        NaN        0.00
## 66534       66534     2194 219     80 1.919068e-03        NaN        0.00
## 67078       67078     2196 219     12 5.682276e-03        NaN        0.00
## 67208       67208     1624 162    120 9.962709e-01 0.20000000        0.00
## 67226       67226     2197 219    136 1.000000e+00 0.50000000        3.10
## 67256       67256     1627 162     22 0.000000e+00 0.14285714        0.00
## 67581       67581     1621 162      7 0.000000e+00 0.25000000        0.00
## 67690       67690     1623 162     31 0.000000e+00        NaN        0.00
## 67731       67731     1621 162      1 0.000000e+00 1.00000000        0.00
## 67753       67753     1623 162      1 0.000000e+00        NaN        0.00
## 68166       68166     1625 162     11 0.000000e+00 1.00000000        0.00
## 68339       68339     1626 162      1 0.000000e+00 1.00000000        0.00
## 68695       68695     1627 162      7 0.000000e+00 1.00000000        1.00
## 68752       68752     2192 219     12 3.924478e-03        NaN        0.00
## 69896       69896     1628 162      1 0.000000e+00        NaN        0.00
## 70026       70026     2192 219     90 3.481434e-03 0.04000000        0.00
## 70206       70206     1627 162      4 0.000000e+00        NaN        0.00
## 70767       70767     1624 162     10 0.000000e+00        NaN        0.00
## 71175       71175     2193 219      6 1.896603e-03 0.08333333        2.25
## 71558       71558     2191 219      2 8.131513e-05        NaN        0.00
## 71996       71996     2192 219     52 1.618863e-03        NaN        0.00
## 72089       72089     2195 219     12 2.236048e-03 0.16666667        0.00
## 72495       72495     2193 219      2 5.292522e-07        NaN        0.00
## 72941       72941     1626 162      1 0.000000e+00        NaN        0.00
```

```
## 73150       73150     2192 219     44 1.151042e-03        NaN        0.00
## 73364       73364     1629 162      1 0.000000e+00        NaN        0.00
## 73777       73777     1623 162      7 0.000000e+00 1.00000000        0.00
## 75034       75034     1626 162      3 0.000000e+00 1.00000000        1.00
## 75431       75431     2195 219      6 3.751055e-03        NaN        0.00
## 75940       75940     2191 219     52 8.568931e-03        NaN        0.00
## 76141       76141     2191 219     30 7.733290e-03 0.50000000        4.00
## 76935       76935     2199 219     24 5.200072e-02        NaN        0.00
## 77348       77348     1626 162      1 0.000000e+00 1.00000000        0.00
## 81211       81211     1628 162      3 0.000000e+00        NaN        0.00
## 81865       81865     1621 162      6 0.000000e+00 1.00000000        0.00
## 81959       81959     1629 162      3 0.000000e+00        NaN        0.00
## 82386       82386     2191 219      8 0.000000e+00        NaN        0.00
## 83552       83552     2194 219      8 1.338727e-03 0.25000000        0.00
## 84460       84460     2193 219     15 8.461901e-04 0.16666667       12.00
## 85216       85216     1627 162      5 0.000000e+00        NaN        0.00
## 87028       87028     2191 219    186 4.897170e-03 0.09090909       16.00
## 87486       87486     1621 162      5 0.000000e+00 0.14285714        0.00
## 87994       87994     2191 219      5 1.020354e-05 1.00000000        0.00
## 88077       88077     2191 219     80 2.649855e-03        NaN        0.00
## 88508       88508     1621 162      3 0.000000e+00        NaN        0.00
## 88905       88905     2199 219      2 1.565709e-04        NaN        0.00
## 89882       89882     1623 162      4 0.000000e+00 0.50000000        0.00
## 91747       91747     1627 162      1 0.000000e+00 1.00000000        0.00
## 91956       91956     1627 162      3 0.000000e+00        NaN        0.00
## 92902       92902     1621 162      1 0.000000e+00        NaN        0.00
## 93403       93403     1626 162      9 0.000000e+00 0.25000000        0.00
## 93677       93677     1623 162      1 0.000000e+00        NaN        0.00
## 94070       94070     1623 162     10 0.000000e+00 0.50000000        0.00
## 94513       94513     2193 219      5 1.927300e-03        NaN        0.00
## 94925       94925     1626 162      4 0.000000e+00        NaN        0.00
## 95339       95339     2193 219     47 2.814301e-02        NaN        0.00
## 95446       95446     1625 162     17 0.000000e+00        NaN        0.00
## 95769       95769     2193 219      4 1.882553e-03 0.08333333        0.75
## 95997       95997     2192 219      4 0.000000e+00        NaN        0.00
## 96206       96206     2194 219      8 4.067542e-05        NaN        0.00
## 96780       96780     2192 219      7 1.917706e-03        NaN        0.00
## 96898       96898     1628 162      2 0.000000e+00        NaN        0.00
## 97328       97328     2199 219    195 7.530191e-02 0.02857143        0.00
## 97520       97520     1624 162      1 0.000000e+00 1.00000000        0.00
## 97590       97590     2193 219      2 6.956133e-05        NaN        0.00
## 97673       97673     2193 219      6 3.751055e-03        NaN        0.00
## 98228       98228     2195 219     25 1.502279e-02        NaN        0.00
## 98700       98700     1625 162     13 0.000000e+00 0.25000000        0.00
## 98717       98717     2192 219     31 7.914968e-04 0.04347826        0.00
## 99047       99047     1627 162      4 0.000000e+00        NaN        0.00
## 99346       99346     2192 219     14 3.453124e-04 1.00000000        0.00
## 99424       99424     1625 162      1 0.000000e+00        NaN        0.00
## 99514       99514     2192 219      8 9.570456e-05        NaN        0.00
```

Now, we will merge the centrality measurements back into our imputed dataset

```
applications_final <- merge(x=applications_mod_imp, y=comp, by='examiner_id', all.x=TRUE)
```

```
# Filter to only include selected work groups
applications_final = applications_final %>% filter(wg==162 | wg==219)

# Remove NaN values for instances where examiner didnt ask for any advice
applications_final <- drop_na(applications_final)
```

Modelling

First we will do a simple model with no interaction variables.

```
lm1 <- lm(appl_proc_days~Eig + Degree + Closeness + Betweenness + gender + tenure_days, data=applicatio
summary(lm1)
```

```
##
## Call:
## lm(formula = appl_proc_days ~ Eig + Degree + Closeness + Betweenness +
##     gender + tenure_days, data = applications_final)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1964.6  -409.7   -82.6   304.0  4233.2
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  1.603e+03  6.311e+01  25.401  < 2e-16 ***
## Eig         -2.302e+02  3.026e+01  -7.606 2.90e-14 ***
## Degree       2.941e+00  1.334e-01  22.042  < 2e-16 ***
## Closeness   -1.359e+02  1.144e+01 -11.878  < 2e-16 ***
## Betweenness  2.651e+01  1.767e+00  15.007  < 2e-16 ***
## gendermale   4.114e+00  7.472e+00   0.551    0.582
## tenure_days -6.778e-02  9.973e-03  -6.796 1.09e-11 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 572.5 on 28881 degrees of freedom
## Multiple R-squared:  0.05473,    Adjusted R-squared:  0.05454
## F-statistic: 278.7 on 6 and 28881 DF,  p-value: < 2.2e-16
```

Some notes/insights/hypothesis based on these results: - Stat significant model and most variables are significant, except for any influence by gender

- Baseline processing time is 1604 days (Female, 0 days of tenure, never sought advice)
- Increasing an examiners eigenvector importance from 0 to 1 would be expected to decrease processing time by 236 days (i.e, the more important). This would make sense, if a particular examiner had a lot of influence over an advice network, with many other examiners seeking them for advice, they would likely be a subject matter expert and would need to take less time seeking out information online or from other individuals to get the answers they needed to process the application

- An examiner seeking advice one additional time from another examiner (an increase in degree by 1) results in an increase in processing time of about 3 days. This could make sense, as seeking out additional advice, or having those come seek advice from you, is time consuming and could take time away from processing the applications
- An increase in closeness centrality from 0 to 1 would be expected to correspond to a 138 day decrease in processing time. This could also make sense, as a high closeness centrality would correspond to an examiner being well connected within the network. Even if they don't know someone who is an SME on a topic, they likely know someone who knows someone. This may make the process of finding the information they are looking for easier, and speed up the time taken to process the application
- A 1 unit increase in betweenness centrality corresponds to a 27 day increase in processing time. Similar to degree centrality, if an examiner is a main gate for information to pass in the network, this could be time consuming and take time away from them processing applications themselves.
- Finally, an increase in tenure by 1 day corresponds to a slight decrease in processing time. This seems intuitive that more experience would result in quicker processing times.

Adding Interaction Variables

```
lm2 <- lm(appl_proc_days~Eig + Degree + Closeness + Betweenness + gender + tenure_days + Degree*gender +
summary(lm2)
```

```
##
## Call:
## lm(formula = appl_proc_days ~ Eig + Degree + Closeness + Betweenness +
##     gender + tenure_days + Degree * gender + Eig * gender + Closeness *
##     gender + Betweenness * gender, data = applications_final)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1639.7  -405.3   -84.1   299.9  4236.0
##
## Coefficients:
##                           Estimate Std. Error t value Pr(>|t|)
## (Intercept)            1398.46470   66.48970  21.033  < 2e-16 ***
## Eig                    -456.03124   66.39166  -6.869 6.61e-12 ***
## Degree                    4.47189    0.53982   8.284  < 2e-16 ***
## Closeness                 9.41931   29.22512   0.322    0.747
## Betweenness             105.37691    6.86764  15.344  < 2e-16 ***
## gendermale              174.52531   30.97343   5.635 1.77e-08 ***
## tenure_days              -0.06056    0.01002  -6.043 1.53e-09 ***
## Degree:gendermale        -3.53856    0.58185  -6.082 1.20e-09 ***
## Eig:gendermale         6863.53189  638.03872  10.757  < 2e-16 ***
## Closeness:gendermale   -144.21974   31.85002  -4.528 5.98e-06 ***
## Betweenness:gendermale  -78.60021    7.12814 -11.027  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 569.4 on 28877 degrees of freedom
## Multiple R-squared:  0.06525,    Adjusted R-squared:  0.06493
## F-statistic: 201.6 on 10 and 28877 DF,  p-value: < 2.2e-16
```

Some notes/insights/hypothesis based on these results: - Stat significant model and all variables are significant with at least 90% confidence

- Relationships for variables from previous model are similar, except closeness now has a more positive relationship with processing time (a unit increase in closeness centrality correlates with a 48 day increase in processing time)
- Tenure still has a reducing effect on processing time
- Unit increase in degree for male examiners reduces processing time by 4.5 days
- Unit increase in eig importance for male examiners increases processing time by 7270 days
- Unit increase in closeness for males decreases processing time by 191 days
- Unit increase in betweenness for males decreases processing time by 78 days

It should be noted that it is difficult to infer how a change by one or many examiners would impact the processing times since each of the centrality scores are interrelated.