ex4

All the preposessing steps are the same as EX3. ## Load the data

```
library(arrow)
```

```
## Warning: package 'arrow' was built under R version 4.1.2
```

```
##
## Attaching package: 'arrow'
```

```
## The following object is masked from 'package:utils':
##
##     timestamp
```

```
df = read_parquet("~/Desktop/McGill/ORGB/2022-ona-assignments/ex3/app_data_sample.parquet")
```

Predicting examiners' gender based on first name:

The gender package attempts to infer gender (or more precisely, sex assigned at birth) based on first names using historical data, typically data that was gathered by the state.

```
library(gender)
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

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

```
first_name = df %>%distinct(examiner_name_first)
gender_probability = gender(first_name$examiner_name_first)
gender_dictionary = gender_probability %>% select(name,gender)
df <- df %>% left_join(gender_dictionary, by = c("examiner_name_first" = "name"))
head(df$gender)
```

```
## [1] "female" NA       "female" "female" "male"   "female"
```

The gender package assign gender based on historical data. Some of the name is not in the data set, thus there are some missing gender information. I filled those values by distribution.

```
table(is.na(df$gender))
```

```
##
##   FALSE    TRUE
## 1714618  303859
```

```
gender_na = is.na(df$gender)
gender_fill = sample(df$gender[!gender_na], size = sum(gender_na), replace = TRUE)
df$gender[is.na(df$gender)] <- gender_fill
table(is.na(df$gender))
```

```
##
##   FALSE
## 2018477
```

All the missing value has been filled.

Predicting examiners' race based on last name:

The "predictrace" package predict the race of a surname using U.S. Census data which says how many people of each race has a certain surname.

```
library(predictrace)
race = predict_race(df$examiner_name_last, probability = FALSE)
df$race = race$likely_race
head(df$race,10)
```

```
##  [1] "white" "white" "white" "white" "white" "white" "black" "white" NA
## [10] "asian"
```

Again, fill the missing values based on distribution.

```
table(is.na(df$race))
```

```
##
##   FALSE    TRUE
## 1704131  314346
```

```
race_na = is.na(df$race)
race_fill = sample(df$race[!race_na], size = sum(race_na), replace = TRUE)
df$race[is.na(df$race)] <- race_fill
table(is.na(df$race))
```

```
##
##   FALSE
## 2018477
```

Calculate Tenure

To calculate tenure, I need to calculate the time the application stay in the system.

For most applications, the filing date is the date on which PTO received the application.

The appl_status_date variable indicates the date that the application entered its most recent status (or status as of the end of 2014).

```
tenure_info <- df %>% select(examiner_id, filing_date, appl_status_date)

library(lubridate)
```

```
##
## Attaching package: 'lubridate'

## The following object is masked from 'package:arrow':
##
##     duration

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

```
tenure_info = tenure_info %>% mutate(appl_status_date = as_date(dmy_hms(appl_status_date)))
tenure_info$tenure_days = as.numeric(difftime(tenure_info$appl_status_date,tenure_info$filing_date,units

## detect missing values
table(is.na(tenure_info$tenure_days))
```

```
##
##   FALSE    TRUE
## 2013867    4610
```

```
## fill missing values
tenure_na = is.na(tenure_info$tenure_days)
tenure_fill = sample(tenure_info$tenure_days[!tenure_na], size = sum(tenure_na), replace = TRUE)
tenure_info$tenure_days[is.na(tenure_info$tenure_days)] <- tenure_fill
table(is.na(tenure_info$tenure_days))
```

```
##
##   FALSE
## 2018477
```

```
## join with df
df$tenure = tenure_info$tenure_days
```

Create variable 'app_proc_time'

```
app_proc_time <- c()
for (i in 1:length(df$application_number)){
  if (is.na(df$abandon_date[i])){
    app_proc_time[i] = as.numeric(difftime(df$patent_issue_date[i],df$filing_date[i],units="days"))
  }
  else{
    app_proc_time[i] = as.numeric(difftime(df$abandon_date[i],df$filing_date[i],units="days"))

  }
}

df$app_proc_time = app_proc_time
```

Pick two workgroup

The two group I pick is 164 and 172. 1600 – Biotechnology 1700 – Chemical and Materials Engineering

```
wg = as.numeric(substr(df$examiner_art_unit, 1, 3))
df$wg = wg
group_164 = df %>% filter(df$wg == 164)
group_172 = df %>% filter(df$wg == 172)
```

Create advice networks from edges-sample

```
library(tidyverse)
net = read_csv("~/Desktop/McGill/ORGB/2022-ona-assignments/ex3/edges_sample.csv")

edges_164 = inner_join(df %>% filter(wg == 164),net,by = c("application_number" = "application_number"))
edges_164 = drop_na(edges_164)

edges_172 = inner_join(df %>% filter(wg == 172),net,by = c("application_number" = "application_number"))
edges_172 = drop_na(edges_172)
```

Create Nodes

```
edges = rbind(edges_164,edges_172)
node_ego = edges %>% select(ego_examiner_id) %>%rename(id=ego_examiner_id)
node_alter = edges %>% select(alter_examiner_id)%>%rename(id=alter_examiner_id)
nodes_all <-rbind.data.frame(node_ego, node_alter)

nodes = nodes_all %>% distinct(id)
```

Create Graph

```
library(igraph)
```

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

## 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:lubridate':
##
##     %--%, union

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

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

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

```
net_164 = graph_from_data_frame(d=edges_164, vertices=nodes, directed=TRUE)
net_164
```

```
## IGRAPH 59848fb DN-- 381 1318 --
## + attr: name (v/c), examiner_art_unit (e/n), app_proc_time (e/n), race
## | (e/c), gender (e/c), tenure (e/n)
## + edges from 59848fb (vertex names):
##  [1] 91688->71059 91688->67669 97910->59738 97910->99004 97910->67669
##  [6] 75775->69583 75775->83794 75775->70306 75775->91151 75775->71534
## [11] 70204->72882 70204->94911 71120->65790 59338->72882 61757->65024
## [16] 61757->72882 60067->91747 60067->71087 60067->73722 60067->81365
## [21] 96963->72882 97910->65790 97910->59738 97910->99004 93839->71946
## [26] 74224->65024 74224->94911 96963->67657 87897->69583 87897->72882
## [31] 75775->69583 75775->83794 75775->70306 93839->67669 93839->71946
## + ... omitted several edges
```

```
net_172 = graph_from_data_frame(d=edges_172, vertices=nodes, directed=TRUE)
```

Pick the mesure of centrality

1. Degree centrality is defined as the number of links incident upon a node

2. Eigenvector Centrality is an algorithm that measures the transitive influence of nodes. A high eigenvector score means that a node is connected to many nodes who themselves have high scores.
3. Closeness centrality is a measure of the average shortest distance from each vertex to each other vertex
4. Betweenness centrality is a way of detecting the amount of influence a node has over the flow of information in a graph.

```
## Degree Centrality
nodes$dc <- degree(net_164)
nodes$dc <- degree(net_172)

## Eigenvector Centrality
nodes$ec <- evcent(net_164)$vector
nodes$ec <- evcent(net_172)$vector

## Closeness Centrality
nodes$cc <- closeness(net_164)
nodes$cc <- closeness(net_172)

## Betweenness Centrality
nodes$bc <- betweenness(net_164)
nodes$bc <- betweenness(net_172)
```

Join the node dataset with original dataset to do the linear regression

```
nodes = left_join(nodes,df%>%filter(wg== 164|wg == 172),by = c("id" = "examiner_id"))
## select the variable I want to examine
nodes = nodes %>% select (id,dc,ec,bc,cc,examiner_art_unit,app_proc_time,race,gender,tenure)

## drop na
nodes = drop_na(nodes)
## change categorical variables into factors
nodes = nodes %>% mutate_at(vars(gender, race),
            as.factor)
```

Linear Regression

```
# create linear regression
lr <- lm(app_proc_time ~ dc+ec+bc+cc+examiner_art_unit+race+gender+tenure,nodes)
# view model summary
summary(lr)
```

```
##
## Call:
## lm(formula = app_proc_time ~ dc + ec + bc + cc + examiner_art_unit +
##     race + gender + tenure, data = nodes)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1411.07   -54.61    29.00   105.49   558.22
##
## Coefficients:
```

```
##                      Estimate Std. Error t value Pr(>|t|)
## (Intercept)       -1.710e+04  2.509e+03  -6.815 1.03e-11 ***
## dc                -1.676e+00  1.009e+00  -1.661   0.0968 .
## ec                 5.259e+01  2.170e+01   2.424   0.0154 *
## bc                -4.488e+00  8.714e+00  -0.515   0.6065
## cc                -2.576e+01  1.345e+01  -1.915   0.0555 .
## examiner_art_unit  1.002e+01  1.456e+00   6.878 6.63e-12 ***
## raceblack          1.305e+01  3.212e+01   0.406   0.6846
## racehispanic      -1.978e+01  2.279e+01  -0.868   0.3854
## racewhite         -1.430e+01  7.702e+00  -1.857   0.0634 .
## gendermale         9.985e+00  6.864e+00   1.455   0.1458
## tenure             8.365e-01  4.496e-03 186.031  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 244.6 on 6754 degrees of freedom
## Multiple R-squared:  0.8569, Adjusted R-squared:  0.8567
## F-statistic:  4046 on 10 and 6754 DF,  p-value: < 2.2e-16
```

Analyze Result

Based on the linear regression result, application process time is significantly affected by tenure and examiner's art unit. Longer tenure time results in longer application process time. Also, among the two art unit that I examine, 164 and 172, 172's examiners took longer time to process the application.

Gender did not affect the relationship. Some other variable that are statistically significant for application processing time are closeness centrality and eigenvector centrality. A examiner with high closeness centrality will have shorter processing time. This make senses because higher closeness centrality indicate they has shorter "path" to other examiners. They can find the help they need easily.