A4

2024-04-08

```
library(tidyverse)
```

```
## Warning: package 'ggplot2' was built under R version 4.3.2
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.3     v readr     2.1.4
## v forcats   1.0.0     v stringr   1.5.0
## v ggplot2   3.5.0     v tibble    3.2.1
## v lubridate 1.9.3     v tidyr     1.3.0
## v purrr     1.0.2
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```
library(lubridate)
library(arrow)
```

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

```
data_path <- "/Users/apple/Desktop/Network Analysis/A4/672_project_data/"
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.
```

Adding gender to 'applications'based on first name

```
library(gender)
examiner_names <- applications %>%
```

```
        distinct(examiner_name_first)

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)

# 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  4497814 240.3    7985808 426.5         NA  4517300 241.3
## Vcells 49628030 378.7   93142584 710.7      16384 79943845 610.0
```

Adding Race based on last name

```
library(wru)
```

```
## Warning: package 'wru' was built under R version 4.3.2
```

```
##
## Please cite as:
##
## Khanna K, Bertelsen B, Olivella S, Rosenman E, Rossell Hayes A, Imai K
## (2024). _wru: Who are You? Bayesian Prediction of Racial Category Using
## Surname, First Name, Middle Name, and Geolocation_. R package version
## 3.0.1, <https://CRAN.R-project.org/package=wru>.
##
## Note that wru 2.0.0 uses 2020 census data by default.
## Use the argument `year = "2010"`, to replicate analyses produced with earlier package versions.
```

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

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

```
## Predicting race for 2020
```

```
## 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 <- 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_
        ))

# 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  4706093 251.4    7985808 426.5         NA  6930477 370.2
## Vcells 52017165 396.9   93142584 710.7      16384 92256505 703.9
```

Adding tenure days

```
library(lubridate) # to work with dates

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

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

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)

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

rm(examiner_dates)
gc()
```

```
##            used  (Mb) gc trigger  (Mb) limit (Mb)  max used  (Mb)
## Ncells  4714714 251.8    7985808 426.5         NA   7985808 426.5
## Vcells 58092647 443.3  111851100 853.4      16384 111798179 853.0
```

The objective is to use linear regression models to explore the impact of network centrality (degree, betweenness, closeness) on the application processing time by USPTO examiners in art unit "161." This focus accounts for the potential variance in processing times across different units due to the unique nature of applications they handle.

```
library(tidyverse)
# Filter the data for workgroup
workgroup <- applications %>%
  filter(substr(examiner_art_unit, 1, 3) == "161")
```

###Calculate the Application Process Time First, it filters out rows where both the patent_issue_date and abandon_date are missing (NA). This ensures that the remaining dataset only includes records where at least one of these dates is known, indicating that a final decision has been made on the application.

Next, it combines the patent_issue_date and abandon_date into a new column, decision_date, using the coalesce function. coalesce takes the first non-NA value among its arguments for each row, effectively choosing the patent issue date if available, or the abandon date otherwise.

Finally,it calculates the difference between the decision_date and filing_date in days for each record, creating a new column app_proc_time to hold this value. This final step provides a numeric representation of the processing time for each patent application, from filing to final decision.

```
# first filter out rows without NA in both patent_issue_date and abandon_date

filtered_workgroup <- workgroup %>%
  filter(!(is.na(patent_issue_date) & is.na(abandon_date)))

# next, combine both the columns to form a new column called final_decision_date

filtered_workgroup <- filtered_workgroup %>%
  mutate(
    decision_date = coalesce(patent_issue_date, abandon_date)
  )

# calculate the differences in application processing days

filtered_workgroup <- filtered_workgroup %>%
  mutate(
    filing_date = ymd(filing_date), # Convert filing_date to Date format if necessary
   decision_date = ymd(decision_date) # Convert final_decision_date to Date format if necessary
  ) %>%
  mutate(
    app_proc_time = as.numeric(difftime(decision_date, filing_date, units = "days"))
  )
```

Prepare the edge dataset

The code removes any rows where ego_examiner_id or alter_examiner_id is missin, refining the dataset to only include rows where both ego_examiner_id and alter_examiner_id are present.

Following the cleaning steps, the dataset undergoes a transformation. It's grouped by both ego_examiner_id and alter_examiner_id, creating subsets of data that share these identifiers. Within each subset, the code

calculates the number of unique application_number entries, thereby quantifying the number of connections between each pair of ego and alter examiners.

```
library(dplyr)

# Drop rows with NA ego_examiner_id
edges_cleaned <- edges %>%
  filter(!is.na(ego_examiner_id))

# Drop rows with NA alter_examiner_id
edges_cleaned <- edges_cleaned %>%
  filter(!is.na(alter_examiner_id))

# Transform the data to calculate number of connections
edges_transformed <- edges_cleaned %>%
  group_by(ego_examiner_id, alter_examiner_id) %>%
  summarise(connections = n_distinct(application_number), .groups = "drop")
```

Calculate centrality measures for each examiner

```
library(igraph)
```

```
## Warning: package 'igraph' was built under R version 4.3.2
```

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

```
## The following objects are masked from 'package:lubridate':
##
##     %--%, union
```

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

```
## The following objects are masked from 'package:purrr':
##
##     compose, simplify
```

```
## The following object is masked from 'package:tidyr':
##
##     crossing
```

```
## The following object is masked from 'package:tibble':
##
##     as_data_frame
```

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

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

```
g <- graph_from_data_frame(edges_transformed, directed = FALSE, vertices = NULL)

# Calculate degree centrality
degree_centrality <- degree(g, mode = "all")
```

```r
# Calculate betweenness centrality
betweenness_centrality <- betweenness(g, directed = FALSE)

# Calculate closeness centrality
closeness_centrality <- closeness(g, mode = "all")

# Combining all centrality measures into a dataframe
centrality_measures <- data.frame(
  examiner_id = V(g)$name,
  degree = degree_centrality,
  betweenness = betweenness_centrality,
  closeness = closeness_centrality
)

# Viewing the first few rows of the centrality measures
head(centrality_measures)
```

```
##       examiner_id degree betweenness    closeness
## 59030       59030      5  16094.0000 5.276210e-05
## 59108       59108     30  66582.9126 6.916586e-05
## 59141       59141      4    482.9573 5.576934e-05
## 59156       59156      1      0.0000 4.435377e-05
## 59165       59165      8   3657.6868 5.384740e-05
## 59166       59166      5    290.8125 5.307011e-05
```

Create a Unique examiner Dataset

By extracting unique examiner attributes and calculating average processing times beforehand, I try to minimize data redundancy. This ensures that each examiner is represented just once in the dataset, making subsequent analyses more straightforward and efficient.

```r
# Extract unique examiner attributes
unique_examiners <- filtered_workgroup %>%
  select(examiner_id, examiner_art_unit, race, gender, tenure_days) %>%
  distinct(examiner_id, .keep_all = TRUE)

# Calculate the average application processing time for each unique examiner
average_processing_time <- filtered_workgroup %>%
  group_by(examiner_id) %>%
  summarise(avg_app_proc_time = mean(app_proc_time, na.rm = TRUE)) %>%
  ungroup()

# Join the average processing time with the unique examiners data
unique_examiners <- unique_examiners %>%
  left_join(average_processing_time, by = "examiner_id")

# View the unique examiners list with average processing time
print(head(unique_examiners))
```

```
## # A tibble: 6 x 6
##   examiner_id examiner_art_unit race  gender tenure_days avg_app_proc_time
##         <dbl>             <dbl> <chr> <chr>        <dbl>             <dbl>
## 1       73198              1615 Asian female        6335             1265.
## 2       64054              1616 white male          5504              500.
```

```
## 3       75736              1617 white female        6323              850.
## 4       69138              1614 white <NA>          6348             1058.
## 5       64839              1616 white male          6254              957.
## 6       95415              1614 white male          6197              526.
```

Drop rows with missing values because they pertain to variables that are predictors in a regression analysis.

```
na_count_by_column <- colSums(is.na(unique_examiners))
print(na_count_by_column)
```

```
##       examiner_id examiner_art_unit              race            gender
##                 1                 0                 0                39
##       tenure_days avg_app_proc_time
##                 5                 0
```

```
unique_examiners <- na.omit(unique_examiners)
na_count_by_column <- colSums(is.na(unique_examiners))
print(na_count_by_column)
```

```
##       examiner_id examiner_art_unit              race            gender
##                 0                 0                 0                 0
##       tenure_days avg_app_proc_time
##                 0                 0
```

Merge unique examiners and centrality measures based on examiner id

```
# Convert examiner_id in unique_examiners_clean to character if it's numeric
unique_examiners <- unique_examiners %>%
  mutate(examiner_id = as.character(examiner_id))

# Ensure examiner_id in centrality_measures is also character
centrality_measures <- centrality_measures %>%
  mutate(examiner_id = as.character(examiner_id))

# Now perform the inner join
data <- unique_examiners %>%
  inner_join(centrality_measures, by = "examiner_id")

# View the first few rows of the combined dataset
head(data)
```

```
## # A tibble: 6 x 9
##   examiner_id examiner_art_unit race  gender tenure_days avg_app_proc_time
##   <chr>                   <dbl> <chr> <chr>        <dbl>             <dbl>
## 1 64839                    1616 white male          6254              957.
## 2 68694                    1619 white male          6350             1240.
## 3 90588                    1615 white female        6343             1287.
## 4 59399                    1615 white male          6339             1282.
## 5 98070                    1616 white female        6338              794.
## 6 74829                    1615 white female        6055              734.
## # i 3 more variables: degree <dbl>, betweenness <dbl>, closeness <dbl>
```

Use linear regression models to estimate the relationship between centrality and 'app_proc_time'

This linear regression model attempts to predict the average processing time (in days) it takes for a U.S. Patent and Trademark Office (USPTO) examiner to process a patent application, using three predictor variables: degree centrality, betweenness centrality, and closeness centrality of the examiners in a network graph.

```
linear_model <- lm(avg_app_proc_time ~ degree + betweenness + closeness, data = data)

summary(linear_model)
```

```
##
## Call:
## lm(formula = avg_app_proc_time ~ degree + betweenness + closeness,
##     data = data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -898.50 -175.41   52.58  214.24  673.65
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  1.123e+03  4.733e+01  23.726   <2e-16 ***
## degree       8.853e+00  9.552e+00   0.927    0.357
## betweenness -6.761e-04  4.206e-03  -0.161    0.873
## closeness   -7.297e+01  2.906e+02  -0.251    0.802
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 296.6 on 76 degrees of freedom
## Multiple R-squared:  0.03087,    Adjusted R-squared:  -0.007382
## F-statistic: 0.807 on 3 and 76 DF,  p-value: 0.4938
```

Coefficients: Intercept (1.123e+03): The model estimates that the baseline average processing time, when all predictor variables are zero, is approximately 1123 days. This is a theoretical scenario, given that centrality measures cannot be zero in practice.

Degree (8.853e+00): The coefficient for degree centrality suggests that for each unit increase in degree centrality, the average processing time is expected to increase by about 8.85 days. However, this effect is not statistically significant (p-value = 0.357), meaning we do not have sufficient evidence to confidently say that degree centrality impacts processing time.

Betweenness (-6.761e-04): The negative coefficient for betweenness centrality implies that as an examiner's betweenness centrality increases, their processing time decreases. However, the effect is very small (a unit increase in betweenness centrality decreases processing time by approximately 0.0006761 days), and it is not statistically significant (p-value = 0.873).

Closeness (-7.297e+01): The coefficient for closeness centrality indicates that an increase in closeness centrality is associated with a decrease in average processing time by about 72.97 days. Yet, this relationship is not statistically significant (p-value = 0.802), suggesting uncertainty about the impact of closeness centrality on processing time.

Conclusion: The model suggests that, based on the data and variables chosen, centrality measures (degree, betweenness, closeness) do not significantly predict the average processing time for patent applications by USPTO examiners. This could mean that other factors not included in the model might be more influential in determining processing times, or that the relationships between these particular network centrality measures

and processing times are complex and not linear. Further investigation with additional variables, different model specifications, or non-linear modeling might provide more insights.

Does this relationship differ by examiner gender?

This linear regression model extends the previous analysis by including gender as a predictor and by examining the interactions between gender and each of the centrality measures (degree, betweenness, and closeness) to understand how the relationship between these centrality measures and average application processing time might differ by gender.

```
linear_model_2 <- lm(avg_app_proc_time ~ gender+degree + betweenness + closeness+gender*degree+gender*b

summary(linear_model_2)
```

```
##
## Call:
## lm(formula = avg_app_proc_time ~ gender + degree + betweenness +
##     closeness + gender * degree + gender * betweenness + gender *
##     closeness, data = data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -919.36 -168.82   58.12  217.73  697.25
##
## Coefficients:
##                           Estimate Std. Error t value Pr(>|t|)
## (Intercept)              1.139e+03  7.042e+01  16.182   <2e-16 ***
## gendermale              -2.062e+01  9.926e+01  -0.208    0.836
## degree                   1.274e+01  1.160e+01   1.098    0.276
## betweenness             -5.384e-03  6.186e-03  -0.870    0.387
## closeness                1.913e+03  1.525e+03   1.255    0.214
## gendermale:degree       -1.113e+01  2.087e+01  -0.533    0.596
## gendermale:betweenness   8.762e-03  9.328e-03   0.939    0.351
## gendermale:closeness    -2.053e+03  1.554e+03  -1.321    0.191
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 298 on 72 degrees of freedom
## Multiple R-squared:  0.07325,    Adjusted R-squared:  -0.01686
## F-statistic: 0.8129 on 7 and 72 DF,  p-value: 0.5795
```

Coefficients: Intercept (1.139e+03): For the baseline category of gender (assuming female if "male" is specified), the average processing time is estimated to be around 1139 days when all other predictors are zero.

Gendermale (-2.062e+01): Being male is associated with a decrease in average processing time by approximately 20.62 days, compared to being female. However, this difference is not statistically significant (p-value = 0.836).

Degree (1.274e+01): Each unit increase in degree centrality is associated with an increase in average processing time by about 12.74 days, although this effect is not statistically significant (p-value = 0.276).

Betweenness (-5.384e-03): The coefficient for betweenness centrality suggests a small decrease in processing time (around 0.0054 days) for each unit increase in betweenness centrality, but this effect is also not statistically significant (p-value = 0.387).

Closeness (1.913e+03): The positive coefficient for closeness centrality suggests an increase in average processing time by about 1913 days for each unit increase in closeness centrality, which is a large effect but

not statistically significant (p-value = 0.214).

Interaction Terms: Gendermale:Degree (-1.113e+01): The negative interaction term suggests that the effect of degree centrality on processing time is smaller for males by about 11.13 days compared to females, but this interaction is not statistically significant (p-value = 0.596).

Gendermale:Betweenness (8.762e-03): This positive interaction term indicates that the effect of betweenness centrality on processing time is slightly more positive for males, increasing processing time by approximately 0.0088 days, but this is not significant (p-value = 0.351).

Gendermale:Closeness (-2.053e+03): The negative interaction term suggests that the effect of closeness centrality on processing time decreases for males by about 2053 days compared to females, yet this effect is not statistically significant (p-value = 0.191).

Conclusion: The model attempts to explore whether the relationships between centrality measures and average processing time vary by gender. However, none of the main effects or interaction terms are statistically significant, suggesting that, within the limitations of this dataset and model specification, there's no clear evidence that gender modifies the impact of these network centrality measures on processing time. This outcome could mean that either gender does not play a significant role in this context, or the model and available data do not capture the complexities of such relationships effectively. Further investigation with additional data, alternative model specifications, or different variables might provide more insights.