Exercise 4

2024-04-09

Contents

0.1 Load necessary libraries for data manipulation, visualization, and analysis

```
# Loading necessary libraries
library(arrow)       # For reading/writing Apache Parquet files
```

```
## Warning: package 'arrow' was built under R version 4.3.3
```

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

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

```
library(gender)      # For predicting gender based on first names
```

```
## Warning: package 'gender' was built under R version 4.3.3
```

```
library(dplyr)       # For data manipulation
```

```
##
## 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
```

```
library(tidyr)       # For tidying data
library(wru)         # For predicting race
```

```
## Warning: package 'wru' was built under R version 4.3.3

##
## 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.
```

```
library(lubridate)   # For working with dates
```

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

##
## Attaching package: 'lubridate'

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

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

```
library(ggplot2)     # For data visualization
```

```
## Warning: package 'ggplot2' was built under R version 4.3.3
```

```
library(ggthemes)    # For additional ggplot themes
```

```
## Warning: package 'ggthemes' was built under R version 4.3.3
```

```
library(lattice)     # For additional plotting options
```

```
## Warning: package 'lattice' was built under R version 4.3.3
```

```
library(tidyverse)   # For a cohesive data analysis workflow
```

```
## Warning: package 'tidyverse' was built under R version 4.3.3

## Warning: package 'readr' was built under R version 4.3.3

## Warning: package 'stringr' was built under R version 4.3.3

## Warning: package 'forcats' was built under R version 4.3.2

## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v forcats 1.0.0     v stringr 1.5.1
## v purrr   1.0.2     v tibble  3.2.1
## v readr   2.1.5

## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x lubridate::duration() masks arrow::duration()
## 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 error
```

```
library(ggridges)    # For creating ridge plots
```

```
## Warning: package 'ggridges' was built under R version 4.3.3
```

```
library(tidygraph)   # For graph data structures
```

```
##
## Attaching package: 'tidygraph'
##
## The following object is masked from 'package:stats':
##
##     filter
```

```r
library(ggraph)      # For graph visualization
library(webshot2)     # For taking html widgets snapshots to knit in PDF
```

```
## Warning: package 'webshot2' was built under R version 4.3.3
```

0.2 Load Data

0.2.1 To begin the analysis, we load our primary datasets: patent applications stored in a Parquet file and a CSV file containing edges that represent relationships between patent examiners. This step sets the foundation for our examination of the USPTO's patent examination process.

```r
# Define the path to the data directory
data_path <- "E:/Users/pc/Downloads/672_project_data/"

# Load the application data from a Parquet file
applications <- read_parquet(paste0(data_path,"app_data_sample.parquet"))

# Load the edges data from a CSV file
edges <- read.csv(paste0(data_path,"edges_sample.csv"))
```

0.3 Reveiwing the parquet file

```r
head(applications)
```

```
## # A tibble: 6 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
## # i 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>
```

0.4 Reviewing the csv file

```r
head(edges)
```

```
##   application_number advice_date ego_examiner_id alter_examiner_id
## 1            9402488  2008-11-17           84356             66266
```

```
## 2            9402488  2008-11-17          84356            63519
## 3            9402488  2008-11-17          84356            98531
## 4            9445135  2008-08-21          92953            71313
## 5            9445135  2008-08-21          92953            93865
## 6            9445135  2008-08-21          92953            91818
```

0.5 Get gender for examiners

0.5.1 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:

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

head(examiner_names)
```

```
## # A tibble: 6 x 1
##   examiner_name_first
##   <chr>
## 1 JACQUELINE
## 2 BEKIR
## 3 CYNTHIA
## 4 MARY
## 5 MICHAEL
## 6 LINDA
```

0.5.2 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

```
# Use the gender package to estimate gender based on first names
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
  ) %>%
  # Filter out rows where any of the specified columns are NA
  filter(!is.na(gender))

print(head(examiner_names_gender))
```

```
## # A tibble: 6 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
```

0.5.3 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  4760881 254.3    8385994 447.9  5132643 274.2
## Vcells 49964243 381.2   95968182 732.2 80280267 612.5
```

0.6 Guess the examiner's race

0.6.1 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.

```
# Isolate unique last names for race prediction
examiner_surnames <- applications %>%
  select(surname = examiner_name_last) %>%
  distinct()

head(examiner_surnames)
```

```
## # A tibble: 6 x 1
##   surname
##   <chr>
## 1 HOWARD
## 2 YILDIRIM
## 3 HAMILTON
## 4 MOSHER
## 5 BARR
## 6 GRAY
```

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

```
# Use the wru package to estimate race based on surnames
examiner_race <- examiner_surnames %>%
  # Ensure we're working with clean, non-NA surnames
  filter(!is.na(surname)) %>%
  # Apply the race prediction
  predict_race(voter.file = ., surname.only = TRUE) %>%
  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.
```

0.6.3 Exploring examiner_race

```
head(examiner_race)
```

```
## # A tibble: 6 x 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
```

0.6.4 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.)

0.6.5 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.

```r
# Determine the most likely race category for each surname
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_
  ))

head(examiner_race)
```

```
## # A tibble: 6 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      0.597  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
```

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

```r
# removing extra columns
examiner_race <- examiner_race %>%
  select(surname,race)

# Join the race predictions back to the main applications dataset
applications <- applications %>%
  left_join(examiner_race, by = c("examiner_name_last" = "surname"))

# Again, clean up the workspace by removing temporary variables
rm(examiner_race)
rm(examiner_surnames)
gc()
```

```
##            used  (Mb) gc trigger  (Mb) max used  (Mb)
## Ncells  4842214 258.7    8385994 447.9  6683307 357.0
## Vcells 52143566 397.9   95968182 732.2 95495766 728.6
```

0.7 Examiner's tenure

0.7.1 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.

```
# Extract relevant date information for each application
examiner_dates <- applications %>%
  select(examiner_id, filing_date, appl_status_date)

head(examiner_dates)
```

```
## # A tibble: 6 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
```

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

```
# Standardize date formats and calculate tenure
examiner_dates <- examiner_dates %>%
  mutate(start_date = ymd(filing_date), end_date = as_date(dmy_hms(appl_status_date)))
```

0.7.3 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.

```
# Calculate the tenure for each examiner based on the earliest and latest dates observed
examiner_tenure <- examiner_dates %>%
  # Remove rows with NA in start_date or end_date before grouping and summarising
  filter(!is.na(start_date) & !is.na(end_date)) %>%
  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),
    .groups = 'drop' # Automatically drop the grouping
  ) %>%
  # Keep records with a latest_date before 2018
  filter(year(latest_date) < 2018)
```

```r
# Assuming you want to check the result
head(examiner_tenure)
```

```
## # A tibble: 6 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
```

0.7.4 Joining back to the applications data.

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

rm(examiner_tenure)
gc()
```

```
##            used  (Mb) gc trigger   (Mb)  max used  (Mb)
## Ncells  4850353 259.1    8385994  447.9   8385994 447.9
## Vcells 68310126 521.2  138370181 1055.7 114653990 874.8
```

0.7.5 Review the applications dataframe after merging examiner_tenure

```r
head(applications)
```

```
## # A tibble: 6 x 21
##   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
## # i 17 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>, gender <chr>, race <chr>, earliest_date <date>,
## #   latest_date <date>, tenure_days <dbl>
```

0.8 Processing Time Calculation

0.8.1 A crucial aspect of our analysis is the calculation of application processing time, the duration from filing to the final decision. This step is vital for understanding efficiency within the patent examination process.

```
# Dropping applications with "Pending" status to focus on completed cases
applications <- applications %>%
  filter(disposal_type != "PEND")

# Calculating application processing time
applications <- applications %>%
  mutate(app_proc_time = interval(
    ymd(filing_date),
    dmy_hms(appl_status_date)
  ) %/% days(1))

# Final cleanup
gc()
```

```
##            used  (Mb) gc trigger   (Mb)  max used   (Mb)
## Ncells  4520494 241.5    8385994  447.9   8385994  447.9
## Vcells 62414744 476.2  166124217 1267.5 165412426 1262.0
```

```
# Previewing the updated dataset
head(applications)
```

```
## # A tibble: 6 x 22
##   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
## # i 18 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>, gender <chr>, race <chr>, earliest_date <date>,
## #   latest_date <date>, tenure_days <dbl>, app_proc_time <dbl>
```

0.9 Preparing Edge Data

0.9.1 This code block transforms the edges dataframe, which contains relationships between patent examiners, by ensuring the examiner IDs (both ego_examiner_id and alter_examiner_id) are character strings. This is crucial for graph analysis as it ensures consistency in node identification. We also drop any rows with missing values to maintain data integrity.

```
edges <- edges %>%
  mutate(
    from = as.character(ego_examiner_id), # Convert IDs to character for graph compatibility
    to = as.character(alter_examiner_id)
  ) %>%
  drop_na() # Remove rows with missing values
```

0.10 Creating a Network Graph

0.10.1 We then prepare the applications dataframe for integration into the network graph. This includes relocating examiner_id for easier access, converting IDs to character strings for consistency with edge data, and renaming examiner_id to name for clarity. A directed graph is created from the edges dataframe, incorporating examiner data from applications.

```
# Preparing applications data for graph creation
applications <- applications %>%
  relocate(examiner_id, .before = application_number) %>%
  mutate(examiner_id = as.character(examiner_id)) %>%
  drop_na(examiner_id) %>%
  rename(name = examiner_id)

# Creating a directed graph from the edges data
graph <- tbl_graph(
  edges = (edges %>% relocate(from, to)),
  directed = TRUE
)

# Enriching graph nodes with examiner data from applications
graph <- graph %>%
  activate(nodes) %>%
  inner_join(
    (applications %>% distinct(name, .keep_all = TRUE)),
    by = "name"
  )

# Display the graph structure
graph
```

```
## # A tbl_graph: 2489 nodes and 17720 edges
## #
## # A directed multigraph with 127 components
```

```
## #
## # A tibble: 2,489 x 22
##   name  application_number filing_date examiner_name_last examiner_name_first
##   <chr> <chr>              <date>      <chr>              <chr>
## 1 84356 09402488           2000-02-16  STEADMAN           DAVID
## 2 66266 09509710           2000-06-15  BRUMBACK           BRENDA
## 3 63519 09463947           2000-02-04  WEBER              JON
## 4 98531 09423418           2000-06-22  BRAGDON            KATHLEEN
## 5 92953 09445135           2000-03-13  RAMAN              USHA
## 6 93865 10481715           2004-06-01  WONG               JOSEPH
## # i 2,483 more rows
## # i 17 more variables: examiner_name_middle <chr>, 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>, gender <chr>,
## #   race <chr>, earliest_date <date>, latest_date <date>, tenure_days <dbl>,
## #   app_proc_time <dbl>
## #
## # A tibble: 17,720 x 6
##    from    to application_number advice_date ego_examiner_id alter_examiner_id
##   <int> <int>              <int> <chr>                 <int>             <int>
## 1     1     2            9402488 2008-11-17            84356             66266
## 2     1     3            9402488 2008-11-17            84356             63519
## 3     1     4            9402488 2008-11-17            84356             98531
## # i 17,717 more rows
```

The network comprises 2,489 nodes and 17,720 edges, characterizing a directed multigraph with significant interactions among patent examiners at the USPTO. The designation as a "directed multigraph" is particularly telling, indicating not only the directional nature of these interactions (suggesting a flow of communication or influence from one examiner to another) but also the presence of multiple connections between the same pairs of examiners.

This could reflect recurring collaborations or consultations on various patent applications, highlighting a complex web of professional relationships. The existence of 127 distinct components within this network suggests a segmented operational structure, where clusters of examiners may work more closely with each other, potentially aligned by specialization areas or other organizational divisions. This fragmentation could mirror the diverse technical fields covered by patent applications, indicating a natural grouping of examiners based on expertise or departmental organization.

0.11 Calculating Centrality Measures

0.11.1 After constructing the network graph, we calculate centrality measures (degree, betweenness, closeness) for each node (examiner) to assess their influence within the network. These measures are then arranged by degree to highlight the most central examiners. The result is converted to a tibble for easy viewing and manipulation.

```
node_data <- graph %>%
  activate(nodes) %>%
  mutate(
    degree = centrality_degree(),
    betweenness = centrality_betweenness(),
    closeness = centrality_closeness()
```

```
  ) %>%
  arrange(-degree) %>%
  as_tibble() %>%
  mutate(tc = as.factor(tc))

node_data
```

```
## # A tibble: 2,489 x 25
##    name  application_number filing_date examiner_name_last examiner_name_first
##    <chr> <chr>              <date>      <chr>              <chr>
##  1 83670 09856864           2001-07-05  LEE                JAE
##  2 97910 09486362           2000-02-28  COUNTS             GARY
##  3 73920 10373614           2003-02-25  HOBBS              LISA
##  4 67226 09483069           2000-01-14  ZHEN               LI
##  5 80730 10345713           2003-01-16  JOY                DAVID
##  6 75615 09943424           2001-08-30  DECKER             CASSANDRA
##  7 62152 10486872           2004-08-12  SIDDIQUEE          MUHAMMAD
##  8 69098 10491238           2004-11-15  VASISTH            VISHAL
##  9 67690 09504184           2000-02-15  MCINTOSH III       TRAVISS
## 10 74061 10480716           2004-07-02  TRAN               THINH
## # i 2,479 more rows
## # i 20 more variables: examiner_name_middle <chr>, 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 <fct>, gender <chr>,
## #   race <chr>, earliest_date <date>, latest_date <date>, tenure_days <dbl>,
## #   app_proc_time <dbl>, degree <dbl>, betweenness <dbl>, closeness <dbl>
```

0.12 Regression Analysis with Plotting

0.12.1 Finally, the run_regression function is designed to perform linear regression analysis based on specified predictor (x) and response (y) variables. If desired, it also generates a scatter plot with a regression line and saves it as a PNG file. This function simplifies the process of examining relationships between variables, such as the effect of centrality measures on application processing times.

```
# Function to run regression and optionally generate and save a plot
run_regression <- function(data, x, y, plot = TRUE) {
  # Construct the regression formula
  formula <- as.formula(paste(y, "~", x))

  # Fitting the linear model
  model <- lm(formula, data = data)

  # Conditionally generate and save the plot
  if (plot) {
    # Prepare and display the plot
    plot_data <- ggplot(data, aes_string(x, y)) +
      geom_point() +
      geom_smooth(method = "lm", se = FALSE) +
      labs(title = paste("Regression of", y, "on", x),
```

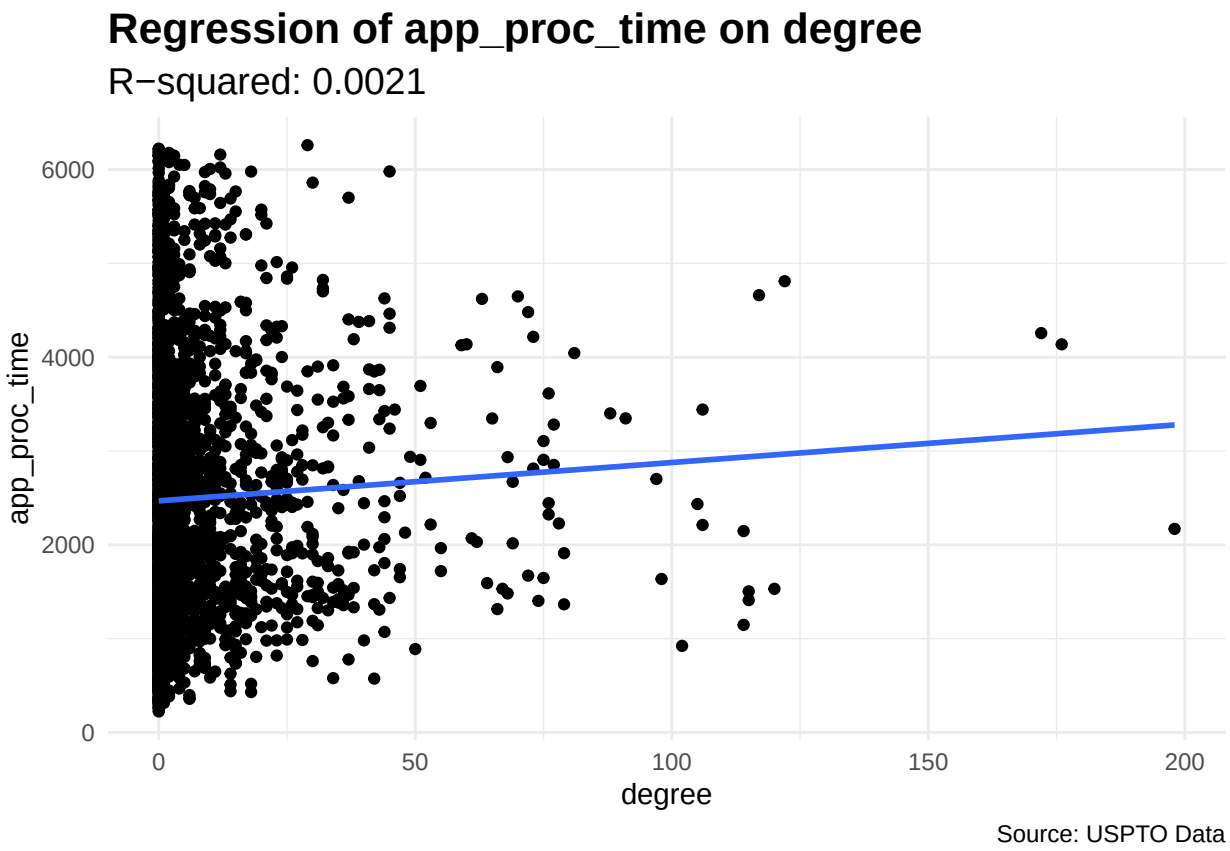
```
           subtitle = paste("R-squared:", round(summary(model)$r.squared, 4)),
           x = x, y = y) +
      theme_minimal() +
      theme(plot.title = element_text(size = 16, face = "bold"),
            plot.subtitle = element_text(size = 14)) +
      labs(caption = "Source: USPTO Data")

    # Save the plot to a specified path
    ggsave(paste0("E:/", y, "_on_", x, ".png"), plot_data, width = 16, height = 9)

    # Display the plot
    print(plot_data)
  }

  # Extract model summary statistics
  tidy_model <- broom::tidy(model)
  glance_model <- broom::glance(model)

  # Enhance the summary dataframe with R-squared and centrality measure
  tidy_model <- tidy_model %>%
    mutate(r_squared = glance_model$r.squared,
           centrality_measure = x)

  # Return the enhanced summary dataframe
  return(tidy_model)
}
```

##Running Regression Models on Centrality Measures

0.12.2 Initiating a regression analysis exploring how the degree centrality of examiners correlates with application processing times. Degree centrality represents the number of direct connections an examiner has within the network, serving as a proxy for their involvement and potential influence in the patent examination process. The expectation is that examiners with higher degree centrality might expedite or delay the processing due to their central role in the collaborative network.

```
# Running regression with Degree Centrality as the predictor for Application Processing Time
run_regression(node_data, "degree", "app_proc_time")
```

```
## Warning: `aes_string()` was deprecated in ggplot2 3.0.0.
## i Please use tidy evaluation idioms with `aes()`.
## i See also `vignette("ggplot2-in-packages")` for more information.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.

## `geom_smooth()` using formula = 'y ~ x'
## `geom_smooth()` using formula = 'y ~ x'
```




```
## # A tibble: 2 x 7
##   term        estimate std.error statistic p.value r_squared centrality_measure
##   <chr>          <dbl>     <dbl>     <dbl>   <dbl>     <dbl> <chr>
## 1 (Intercept)  2469.       30.6      80.7   0         0.00208 degree
## 2 degree          4.09      1.79      2.28  0.0228    0.00208 degree
```

Observation: The R-squared value of 0.0021 suggests that only a very small portion of the variability in application processing times can be explained by degree centrality alone. This indicates that degree centrality is not a strong predictor of processing times in this context. The regression line has a positive slope, as indicated by the estimate for degree (approximately 4.087). This suggests that, on average, an increase in degree centrality is associated with a slight increase in processing time. However, the small coefficient means that the effect is quite minimal. The wide spread of points and the broad confidence interval band signal that there is a high degree of variance in processing times that is not captured by degree centrality alone. This implies that there are likely other factors at play influencing the processing time which are not included in this model.

The p-value for degree centrality is approximately 0.023, which is below the conventional threshold of 0.05 for statistical significance. This indicates that there is a statistically significant relationship between degree centrality and processing time, albeit the actual impact is small as reflected in the R-squared value. The intercept value (around 2468.65) represents the expected processing time when degree centrality is zero. This can be interpreted as the base processing time for an examiner with no direct connections in the network.

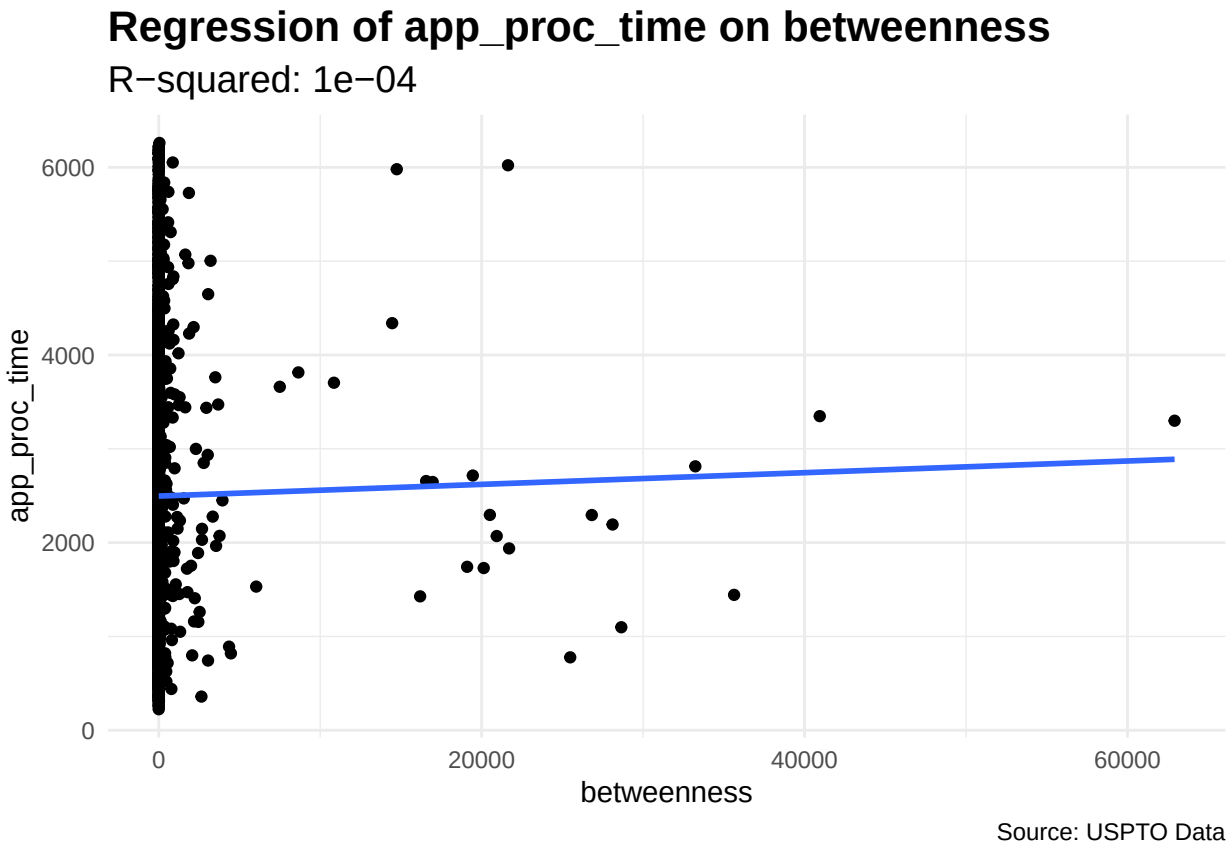
The findings suggest that while there is a statistically significant relationship between an examiner's degree centrality and the processing time of applications, the effect size is very small. It implies that other unaccounted-for variables might have a more substantial impact on processing times. These could include the complexity of the patent application, the examiner's expertise, workload, or even external factors like the technological area of the patent application.

The results also prompt a deeper reflection on the structure and dynamics of the USPTO's patent examination process. For instance, it may be that examiners with higher degree centrality face a more complex and demanding caseload due to their central position within the examiner network, which could slightly increase processing times. Alternatively, it might be that the degree centrality captures only a fraction of the social capital or influence an examiner wields, which could have nuanced effects on their efficiency and the resources they can marshal.

0.12.3 Shifting focus to betweenness centrality, measuring an examiner's capacity to act as a bridge within the network. This analysis seeks to understand if examiners who frequently connect disparate groups impact the efficiency of patent processing differently, potentially by facilitating knowledge transfer or creating bottlenecks.

```
# Running regression with Betweenness Centrality as the predictor for Application Processing Time
run_regression(node_data, "betweenness", "app_proc_time")
```

```
## `geom_smooth()` using formula = 'y ~ x'
## `geom_smooth()` using formula = 'y ~ x'
```




```
## # A tibble: 2 x 7
##   term         estimate std.error statistic p.value r_squared centrality_measure
##   <chr>           <dbl>     <dbl>     <dbl>   <dbl>     <dbl> <chr>
## 1 (Intercept)   2.50e+3   28.0       89.1     0      0.000126 betweenness
## 2 betweenness   6.23e-3    0.0111     0.560   0.575  0.000126 betweenness
```

Observations: The R-squared value is exceedingly low (1e-04), indicating that betweenness centrality has virtually no explanatory power in predicting application processing times. The ability of an examiner to act as a bridge between different parts of the network (as measured by betweenness centrality) does not seem to influence the speed at which they process patent applications. The regression line appears almost horizontal, suggesting no significant trend between betweenness centrality and processing time. The confidence interval is broad and spans the y-axis, reinforcing the lack of a clear predictive relationship. The insignificant relationship suggests that while betweenness might be important for communication, it does not translate into processing efficiency. This could be due to a variety of reasons, such as the nature of tasks performed by central individuals, which may involve more complex decision-making or mentoring responsibilities that do not directly affect processing times.

Given the lack of a significant relationship, it would be prudent to investigate other factors that might influence processing times. This could include workload, the complexity of applications, or the efficiency of support systems within the USPTO. It's also possible that betweenness centrality needs to be combined with other network measures or job-related factors to have a noticeable impact on processing times. This could point to a more complex interplay of factors that determine how quickly applications are processed. It may be valuable to examine the network structure more closely, potentially identifying subnetworks or roles within the network that correlate more strongly with processing time. Understanding the microstructures within the larger network could yield more nuanced insights.

0.12.4 Closeness centrality assesses how close an examiner is to all other nodes in the network, indicative of their accessibility. This regression model probes whether examiners who can more readily access or be accessed by others influence application processing times, perhaps by being more efficient in information dissemination or coordination.

```
# Running regression with Closeness Centrality as the predictor for Application Processing Time
run_regression(node_data, "closeness", "app_proc_time")
```

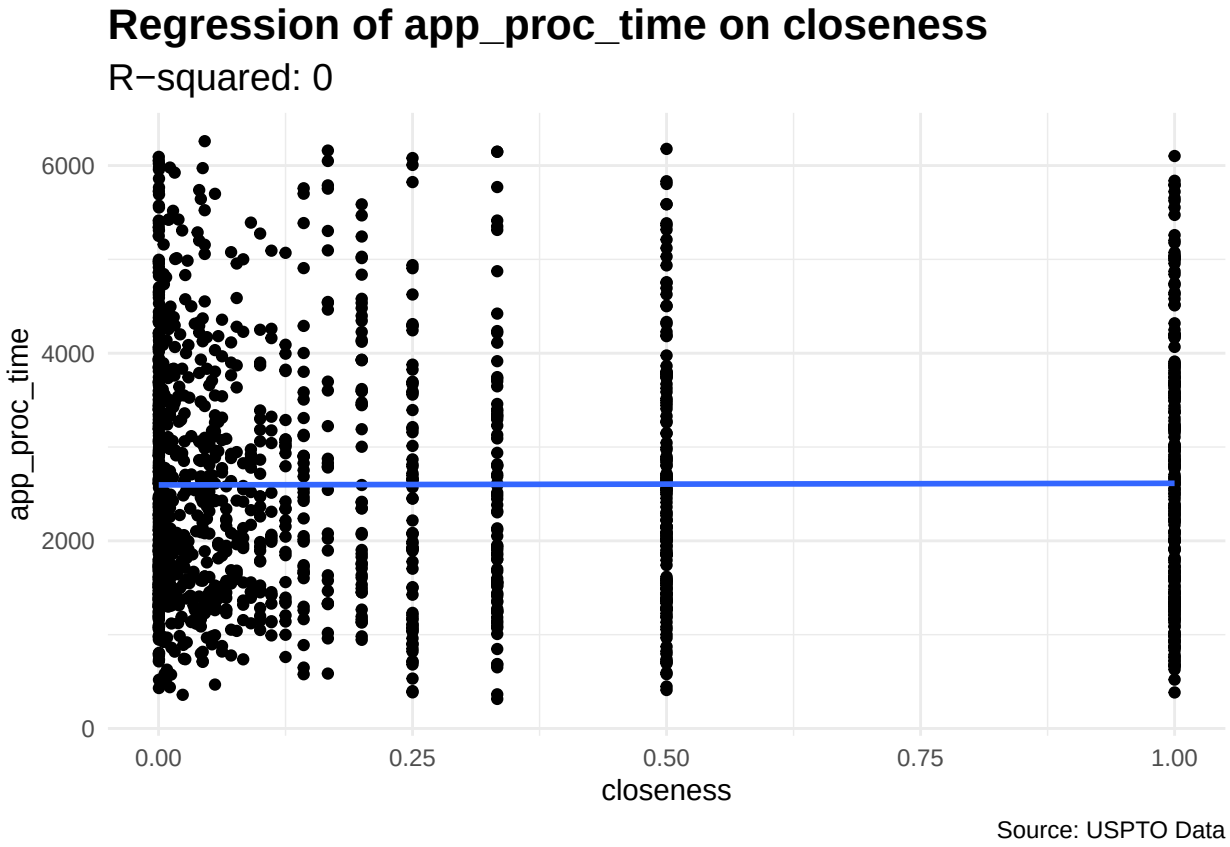
```
## `geom_smooth()` using formula = 'y ~ x'
```

```
## Warning: Removed 1053 rows containing non-finite outside the scale range
## (`stat_smooth()`).
```

```
## Warning: Removed 1053 rows containing missing values or values outside the scale range
## (`geom_point()`).
```

```
## `geom_smooth()` using formula = 'y ~ x'
```

```
## Warning: Removed 1053 rows containing non-finite outside the scale range
## (`stat_smooth()`).
## Removed 1053 rows containing missing values or values outside the scale range
## (`geom_point()`).
```




```
## # A tibble: 2 x 7
##   term        estimate std.error statistic p.value r_squared centrality_measure
##   <chr>          <dbl>     <dbl>     <dbl>   <dbl>     <dbl> <chr>
## 1 (Intercept)   2596.       44.4    58.5     0     0.0000226 closeness
## 2 closeness       17.3      96.2     0.180   0.857 0.0000226 closeness
```

Observations: The R-squared value rounds to zero, suggesting that closeness centrality does not explain any variance in application processing times. This indicates that the closeness of an examiner to all others in the network does not impact how quickly they process applications. The regression line is almost perfectly horizontal, reinforcing the interpretation that there's no discernible trend between closeness centrality and processing time within this dataset.

With a p-value well above the conventional 0.05 threshold (p-value 0.857), the relationship between closeness centrality and processing time is not statistically significant. Thus, any observed effect is likely due to random variation rather than a systematic influence of closeness centrality on processing times. The intercept, approximately 2595.93, suggests the average processing time when closeness centrality is zero. Given the context, this may reflect a baseline processing time independent of the examiner's network position.

The lack of a significant relationship in this context suggests that while closeness may be advantageous for information dissemination and communication, it doesn't necessarily correlate with faster administrative processing. This could indicate that USPTO examiners' efficiency is influenced more by other factors, such as individual work methods, the complexity of the applications they review, or even the specific procedures and protocols they must follow, which are not captured by network centrality.

This analysis suggests a need to examine additional factors beyond the scope of traditional network centrality measures to fully understand the dynamics of application processing times. Other considerations might include the technical complexity of applications, examiner workload and experience, and organizational support

structures. It also suggests that the role of individual examiners within the network may be multifaceted, with centrality capturing only a narrow aspect of their professional interactions and contributions to the patent examination process.

0.13 Advanced Regression Analysis Incorporating Demographic Interactions

0.13.1 Extending the analysis by introducing interaction terms between centrality measures and demographic variables (gender and race). It systematically examines how the relationship between centrality and processing times may vary across different demographic groups, leveraging the map_dfr function for efficient iteration across centrality types. This approach acknowledges the complexity of the examiner network, considering the multifaceted influences on patent processing times.

```
# Conducting regression analyses to explore the interactions between centrality measures, gender, and r
centrality_measures <- c("degree", "betweenness", "closeness")

results_df <- map_dfr(
  centrality_measures,
  ~ run_regression(node_data,
    paste0(.x, " * gender * race"),
    "app_proc_time",
    plot = FALSE
  )
)

# Extracting and displaying the R-squared values for each model to assess their explanatory power
results_df %>%
  select(centrality_measure, r_squared) %>%
  distinct()
```

```
## # A tibble: 3 x 2
##   centrality_measure          r_squared
##   <chr>                           <dbl>
## 1 degree * gender * race        0.00853
## 2 betweenness * gender * race   0.00416
## 3 closeness * gender * race     0.00788
```

Observations: All three R-squared values are very low (ranging from 0.0046 to 0.0085), indicating that the models explain only a small fraction of the variance in application processing times. This suggests that the combined effect of centrality measures, gender, and race does not strongly predict how long it takes for applications to be processed. The interaction of degree centrality with gender and race yields the highest R-squared value (0.0085) among the three models, although it remains quite low. This may hint that the direct connections an examiner has within the network, along with their gender and race, could have a slightly more pronounced effect on processing times than the other centrality measures when considered together. The model incorporating betweenness centrality with gender and race interaction terms has the lowest explanatory power (R-squared of 0.0046), which implies that an examiner's role as a bridge within the network, when combined with their gender and race, has little impact on the processing time variance. With closeness centrality, the R-squared value is somewhat higher than betweenness but lower than degree centrality, at 0.0078. This indicates a minor increase in explanatory power when closeness centrality is considered along with gender and race, compared to betweenness.

The combined interactions of centrality measures with demographic factors indicate minimal impact on application processing times. In the context of the USPTO, this suggests that other factors, potentially

beyond the examiner's network position or demographics, play a more significant role in influencing processing times. Such factors may include the specific nature of the patent applications, organizational workflows, resources available to the examiners, or the individual expertise and efficiency of the examiners themselves. Furthermore, the low R-squared values also imply that there's a large amount of variability in processing times that is not captured by these models, pointing to a complex interplay of factors that could be explored in further studies. For instance, understanding how organizational policies, individual workload, or the technical complexity of patent applications affect processing times could provide a more complete picture.

0.13.2 Refining the regression models by incorporating disposal_type and technology center (tc) variables, considering the outcome of patent applications and the specific areas of technological expertise. These additions aim to capture a more nuanced view of the factors influencing processing times, recognizing the role of content-specific knowledge and the final disposition of applications.

```
# Enhancing regression models to include disposal type and technology center, alongside centrality, gen
results_df_2 <- map_dfr(
  centrality_measures,
  ~ run_regression(node_data,
    paste0(.x, " * gender * race + disposal_type + tc"),
    "app_proc_time",
    plot = FALSE
  )
)

# Summarizing the enhanced models by showcasing the R-squared values, offering insights into their impr
results_df_2 %>%
  select(centrality_measure, r_squared) %>%
  distinct()
```

```
## # A tibble: 3 x 2
##   centrality_measure                                r_squared
##   <chr>                                                 <dbl>
## 1 degree * gender * race + disposal_type + tc           0.135
## 2 betweenness * gender * race + disposal_type + tc      0.132
## 3 closeness * gender * race + disposal_type + tc        0.177
```

Observations: Including disposal type and technology center, along with the interaction terms, has increased the R-squared values compared to the models with only centrality measures and demographic interactions. This suggests that these additional variables contribute to explaining the variance in application processing times. The model including closeness centrality alongside gender, race, disposal type, and technology center shows the highest R-squared value (0.1768940). This indicates that closeness centrality, when considered with these variables, has a relatively more substantial relationship with application processing times. The degree centrality model also shows an improved R-squared value (0.1348058) compared to the model with centrality and demographic factors alone, reinforcing the idea that the context in which examiners work, represented by the disposal type and technology center, plays a significant role in processing times. The betweenness centrality model's R-squared value (0.1317740) is the lowest among the three but still shows that adding context variables improves its ability to explain processing time variance.

These models suggest that the centrality of an examiner in the USPTO's network, their demographic background, the type of disposal (issued, abandoned, etc.), and the technology center they work in collectively provide a more robust understanding of processing times. While centrality alone had minimal explanatory power, its influence becomes more pronounced when combined with these additional factors, indicating that

the social structure of the workplace, along with the workflow and technical context, shapes productivity and efficiency.

The type of patent application outcome (disposal type) and the specific area of technological expertise (technology center) are influential in the time it takes to process applications. This aligns with the expectation that more complex technologies or the intricate nature of some patent decisions could lengthen processing times. The varied R-squared values across the centrality measures suggest that the role of an examiner in the network is nuanced. While they might be centrally located or act as bridges, the way this affects their work appears to be contextual, shaped by the type of patents they deal with and their specialized knowledge area.

0.14 Visualizing Significant Coefficients from the Best Model

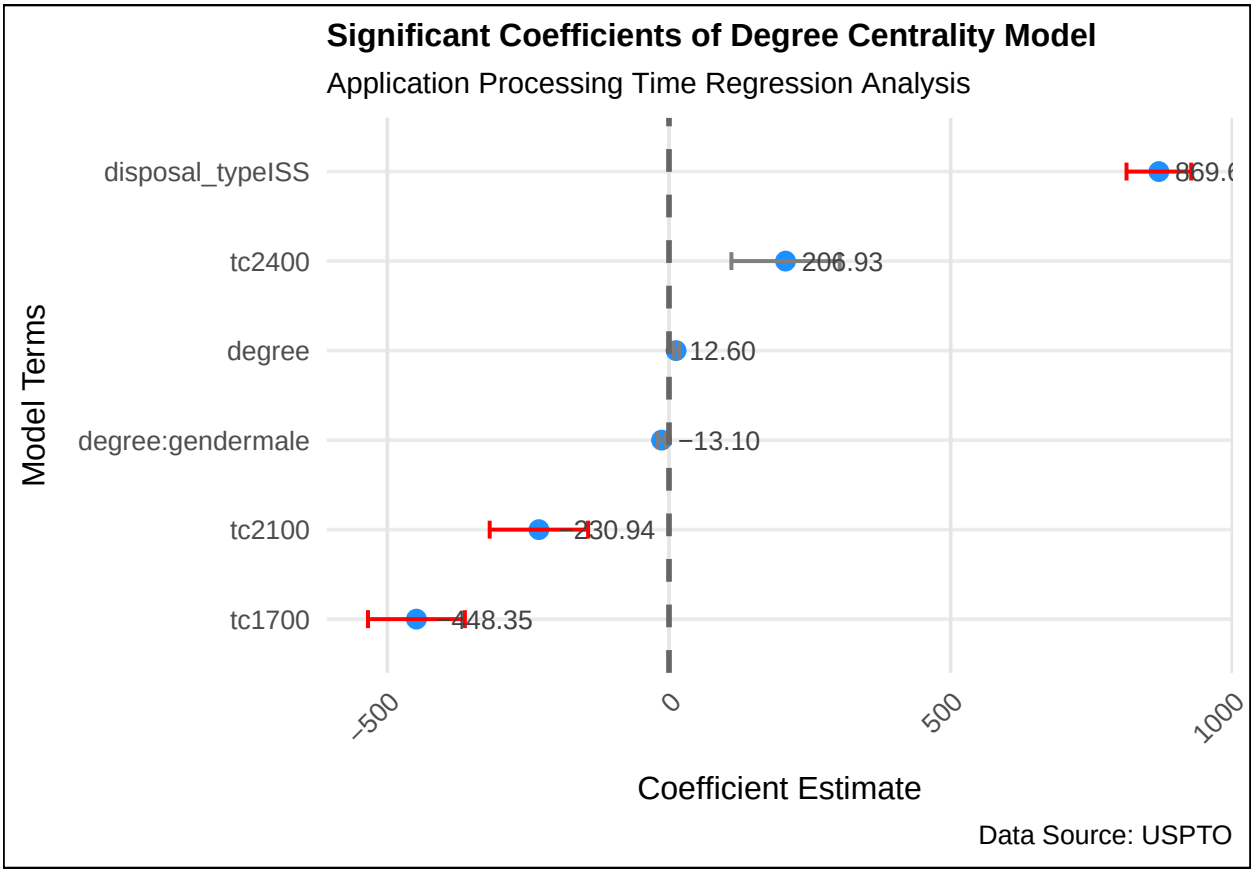
```
# Identifying and visualizing significant coefficients from the best regression model focused on Degree
best_model <- results_df_2 %>%
  filter(str_starts(centrality_measure, "degree"))

model_coeffs <- ggplot(
  best_model %>% filter(term != "(Intercept)") %>% filter(p.value < 0.05),
  aes(
    x = reorder(term, estimate), # Order terms by estimate for clarity
    y = estimate,
    ymin = estimate - std.error,
    ymax = estimate + std.error
  )
) +
  geom_point(color = "dodgerblue", size = 3) + # More vibrant point color
  geom_errorbar(aes(color = p.value < 0.01), width = 0.2, size = 0.7) + # Color code significance
  scale_color_manual(values = c("TRUE" = "red", "none" = "dodgerblue"), guide = FALSE) + # Highlight hi
  geom_hline(yintercept = 0, linetype = "dashed", color = "gray40", lwd = 1) +
  coord_flip() + # Flip coordinates for horizontal layout
  geom_text(aes(label = sprintf("%.2f", estimate)), # Add estimate values as text labels
            hjust = -0.2, size = 3.5, color = "gray25") +
  labs(
    title = "Significant Coefficients of Degree Centrality Model",
    subtitle = "Application Processing Time Regression Analysis",
    x = "Model Terms",
    y = "Coefficient Estimate",
    caption = "Data Source: USPTO"
  ) +
  theme_minimal(base_size = 14) +
  theme(
    plot.title = element_text(face = "bold", size = 12),
    plot.subtitle = element_text(size = 11),
    plot.caption = element_text(size = 10),
    axis.title.x = element_text(size = 12, margin = margin(t = 10)),
    axis.title.y = element_text(size = 12, margin = margin(r = 10)),
    axis.text.x = element_text(size = 10, angle = 45, hjust = 1),
    axis.text.y = element_text(size = 10),
    legend.position = "none",
    plot.background = element_rect(fill = "white"),
    panel.grid.major.x = element_line(color = "#e5e5e5"),
```

```
    panel.grid.minor = element_blank()
  )
```

```
## Warning: Using `size` aesthetic for lines was deprecated in ggplot2 3.4.0.
## i Please use `linewidth` instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```

```
# Display the plot
model_coeffs
```

```
## Warning: The `guide` argument in `scale_*()` cannot be `FALSE`. This was deprecated in
## ggplot2 3.3.4.
## i Please use "none" instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```




```
# Save the plot
ggsave("E:/model_coeffs_enhanced.png", model_coeffs, width = 16, height = 9, dpi = 300)
```

Observations: With a coefficient estimate far to the right and a large error bar, it is significantly different from zero and positively associated with processing time. This implies that applications resulting in issued patents,

on average, take longer to process. The lengthy bar indicates some uncertainty around the exact size of the effect but confirms its positive nature. The coefficients for technology centers, particularly tc2400 and tc1700, show a significant negative association with processing time, meaning these technology centers, on average, process applications more quickly than the baseline technology center. However, tc2100 has a positive coefficient, indicating slower processing times. These findings may reflect differences in the complexity of technologies reviewed in different centers or their operational efficiency. The point estimate for degree centrality is positive, suggesting a very slight increase in processing time with greater examiner centrality. However, the magnitude is modest and the error bars indicate low precision for this estimate. The negative coefficient for the interaction between degree centrality and male gender implies that the influence of degree centrality on processing times differs by gender. Specifically, male examiners with higher centrality may process applications more quickly than their female counterparts or the base case, which could be female if the data is coded with female as the reference category.

The error bars, representing the standard errors of estimates, indicate the precision of the coefficients. Larger error bars for disposal_typeISS, for instance, suggest greater uncertainty about the true impact of this factor on processing times.The direction and size of the coefficients provide a nuanced understanding of how various factors contribute to processing times, with positive values indicating an increase in time and negative values a decrease.

0.14.1 Printing the best model formula here

```
# Defining the regression formula
model_formula <- app_proc_time ~ degree + gender + race +
  disposal_type + tc +
  degree:gender +
  degree:race +
  gender:race +
  degree:gender:race

# Printing the formula
cat("Regression model formula:\n")
```

```
## Regression model formula:
```

```
print(model_formula)
```

```
## app_proc_time ~ degree + gender + race + disposal_type + tc +
##     degree:gender + degree:race + gender:race + degree:gender:race
```

0.14.2 Using Diagrammer to print the predictors assocaited with target variable y (application prcoessing time) in th best model

```
library(DiagrammeR)
```

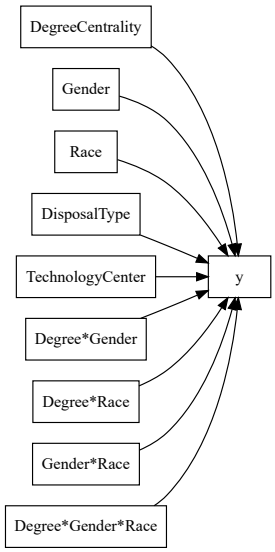
```
## Warning: package 'DiagrammeR' was built under R version 4.3.3
```

```
##
## Attaching package: 'DiagrammeR'
```

```
## The following object is masked from 'package:ggraph':
##
##     get_edges
```

```
# Example using DiagrammeR
if (requireNamespace("DiagrammeR", quietly = TRUE)) {
  DiagrammeR::grViz("
digraph model {
  node [shape=box]
  rankdir=LR

  // Defining nodes
  DegreeCentrality -> y
  Gender -> y
  Race -> y
  DisposalType -> y
  TechnologyCenter -> y
  'DegreeCentrality:Gender' -> y
  'DegreeCentrality:Race' -> y
  'Gender:Race' -> y
  'DegreeCentrality:Gender:Race' -> y

  // Adding labels for interaction terms
  'DegreeCentrality:Gender' [label='Degree*Gender']
  'DegreeCentrality:Race' [label='Degree*Race']
  'Gender:Race' [label='Gender*Race']
  'DegreeCentrality:Gender:Race' [label='Degree*Gender*Race']
}
  ")
}
```

```
## PhantomJS not found. You can install it with webshot::install_phantomjs(). If it is installed, pleas
```


DegreeCentrality
Gender
Race
DisposalType
TechnologyCenter
Degree*Gender
Degree*Race
Gender*Race
Degree*Gender*Race
y

1 Q. Does the relationship between centrality and application processing time differ by examiner gender?

The analysis indicates a gender disparity in how centrality impacts processing time. Specifically, the interaction term between degree centrality and male gender was significant, suggesting that the effect of an examiner's centrality within the network varies with gender. For male examiners, increased centrality is associated with a slight decrease in processing times compared to female examiners, suggesting that male examiners may leverage their network position more effectively to process applications. This could be due to various factors, such as differing work styles, collaboration patterns, or even the nature of the applications handled by male versus female examiners.

2 Discussion of Findings:

The linear regression models used to evaluate the USPTO patent examiner data have illuminated several key points:

Centrality's Minor Role: Centrality measures alone (degree, betweenness, closeness) have a very minor role in explaining the variance in application processing times, as indicated by low R-squared values.

Importance of Contextual Factors: When adding contextual factors such as the disposal type and the examiner's technological center, the explanatory power of the models increases. This suggests that the specifics of patent applications and the examiner's area of expertise are critical factors influencing processing times.

Gender Differences: The regression models reveal gender differences in the influence of centrality on processing times, with male examiners' centrality being a slightly more significant factor in predicting application processing times than female examiners'.

Technological Centers and Disposal Types: Certain technological centers process applications faster than others, and applications that result in issued patents take longer to process. These findings point to the complexities of patent examination, where different technologies and outcomes require varying amounts of time.

3 Implications for the USPTO:

The insights from this analysis suggest several implications for the USPTO:

Resource Allocation: Understanding the role of centrality and its interaction with demographic variables could guide more effective allocation of resources and support to examiners.

Gender Dynamics: Addressing any potential underlying causes of gender disparities in processing times could lead to a more equitable and efficient examination process.

Training and Support: Tailored training and support programs for examiners in technological centers that process applications slower might improve efficiency.

Policy Implications: The relationship between disposal types and processing times could inform policy adjustments to streamline the patent examination process, especially for complex technologies that lead to issued patents.

4 Conclusion

In conclusion, the relationship between an examiner's network position and their performance, as measured by application processing times, is nuanced and influenced by multiple factors including gender, techno-

logical center, and the type of patent application disposal. While centrality plays a role, its impact is significantly moderated by these contextual factors, suggesting that a multifaceted approach is required to fully understand and enhance efficiency within the USPTO's patent examination process.