Assignment 3

USPTO patent examiner data

Loading the dataset and extracting the gender,race and tenure

```
library(arrow)
```

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

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

```
library(readr)
data_path <- "D:/ORGB/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.
```

```
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
```

```
library(tidyr)
library(gender)
#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)
examiner_names
```

Extracting the gender

```
# get a table of names and gender
examiner_names_gender <- examiner_names %>%
  do(results = gender(.$examiner_name_first, method = "ssa")) %>%
  unnest(cols = c(results), keep_empty = TRUE) %>%
  select(
    examiner_name_first = name,
    gender,
    proportion_female
  )

examiner_names_gender
```

```
## # A tibble: 1,822 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
```

```
# 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
```

```
# 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  4375181 233.7    7989887 426.8  4395283 234.8
## Vcells 49439270 377.2   92915974 708.9 79755010 608.5
```

Extracting the surnames

```
library(wru)
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
```

Extracting the race

```
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       0.597   0.295    0.0275   0.00690   0.0741
##  2 YILDIRIM     0.807   0.0273   0.0694   0.0165    0.0798
##  3 HAMILTON     0.656   0.239    0.0286   0.00750   0.0692
##  4 MOSHER       0.915   0.00425  0.0291   0.00917   0.0427
##  5 BARR         0.784   0.120    0.0268   0.00830   0.0615
##  6 GRAY         0.640   0.252    0.0281   0.00748   0.0724
##  7 MCMILLIAN    0.322   0.554    0.0212   0.00340   0.0995
##  8 FORD         0.576   0.320    0.0275   0.00621   0.0697
##  9 STRZELECKA   0.472   0.171    0.220    0.0825    0.0543
## 10 KIM          0.0169  0.00282  0.00546  0.943     0.0319
## # ... with 3,796 more rows
```

```
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       0.597   0.295    0.0275   0.00690   0.0741      0.597 white
##  2 YILDIRIM     0.807   0.0273   0.0694   0.0165    0.0798      0.807 white
##  3 HAMILTON     0.656   0.239    0.0286   0.00750   0.0692      0.656 white
##  4 MOSHER       0.915   0.00425  0.0291   0.00917   0.0427      0.915 white
##  5 BARR         0.784   0.120    0.0268   0.00830   0.0615      0.784 white
##  6 GRAY         0.640   0.252    0.0281   0.00748   0.0724      0.640 white
##  7 MCMILLIAN    0.322   0.554    0.0212   0.00340   0.0995      0.554 black
##  8 FORD         0.576   0.320    0.0275   0.00621   0.0697      0.576 white
##  9 STRZELECKA   0.472   0.171    0.220    0.0825    0.0543      0.472 white
## 10 KIM          0.0169  0.00282  0.00546  0.943     0.0319      0.943 Asian
## # ... with 3,796 more rows
```

```
# 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  4606873 246.1    7989887 426.8  7989887 426.8
## Vcells 53895053 411.2  111579168 851.3 92717093 707.4
```

Extracting the examiner tenure date

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

```
## Loading required package: timechange
```

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

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

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

```
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
```

```
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)

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
```

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

rm(examiner_dates)
gc()
```

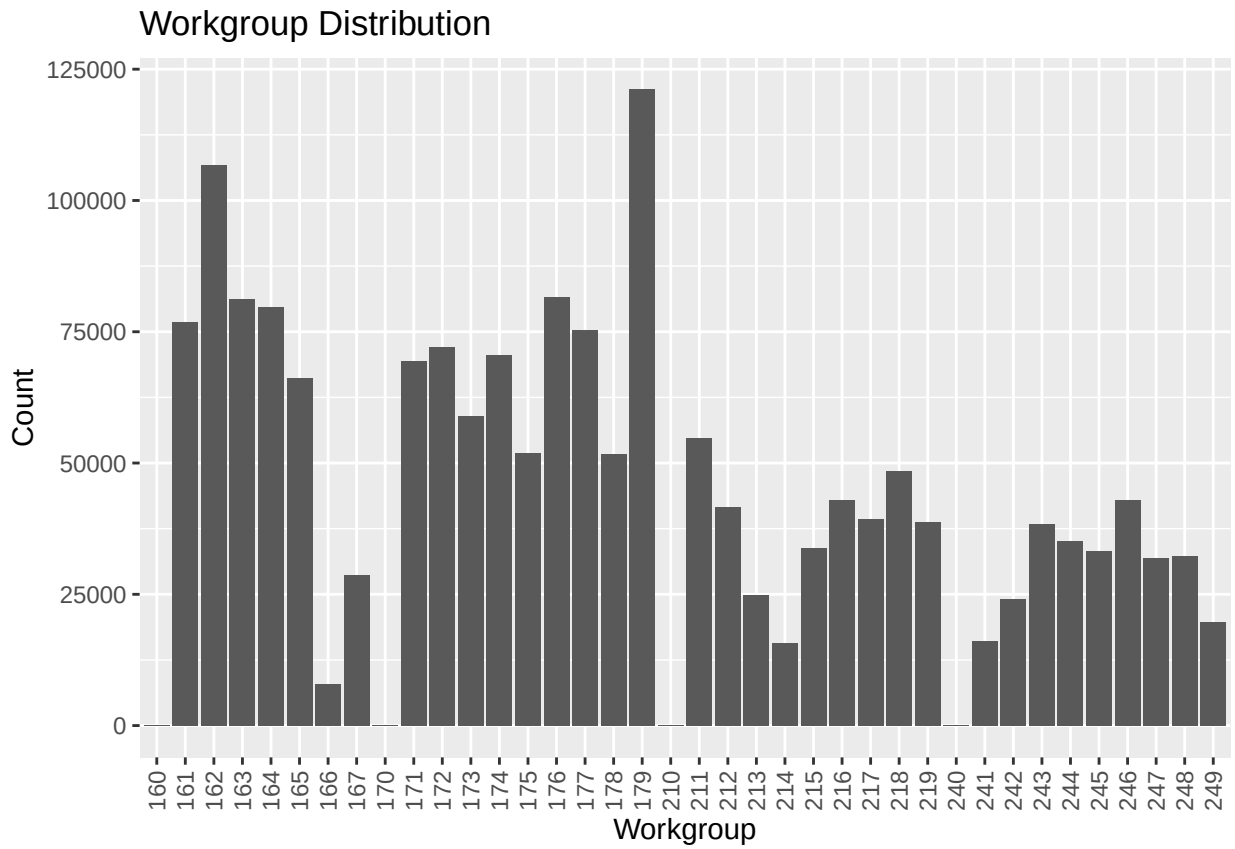
```
##            used  (Mb) gc trigger   (Mb)  max used   (Mb)
## Ncells  4614399 246.5   14521318  775.6  14521318  775.6
## Vcells 64161499 489.6  133975001 1022.2 133920406 1021.8
```

```
applications$work_group <-  substr(applications$examiner_art_unit, 1, 3)
```

```
applications1 <- applications[complete.cases(applications[c("gender", "race")]),]
```

Visualising the various workgroups

```
library(ggplot2)
ggplot(applications1, aes(x = work_group)) +
  geom_bar() +
  xlab("Workgroup") +
  ylab("Count") +
  ggtitle("Workgroup Distribution")+
  theme(axis.text.x = element_text(angle = 90, vjust = 0.5))
```



###Filtering out the workgroups 161, 164 and 179

```
applications_161 <- applications1 %>%  filter(work_group == 161)
applications_164 <- applications1 %>%  filter(work_group == 164)
applications_179 <- applications1 %>%  filter(work_group == 179)
```

###Demographics of workgroup 161

```
applications_161 %>% select(tenure_days,race,gender) %>% summary()
```

```
##   tenure_days       race              gender
##  Min.   : 330   Length:76829       Length:76829
##  1st Qu.:5211   Class :character   Class :character
##  Median :5911   Mode  :character   Mode  :character
##  Mean   :5648
##  3rd Qu.:6324
##  Max.   :6350
##  NA's   :2519
```

```
library(gridExtra)
```

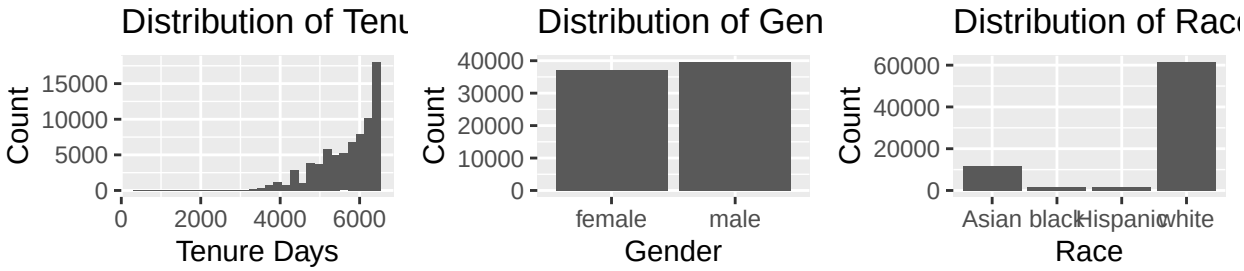
```
## Warning: package 'gridExtra' was built under R version 4.2.3
```

```
##
## Attaching package: 'gridExtra'
```

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

```r
# create histogram of tenure days
plot1 <- ggplot(applications_161, aes(x = tenure_days)) +
  geom_histogram() +
  xlab("Tenure Days") +
  ylab("Count") +
  ggtitle("Distribution of Tenure Days")

# create bar plot of gender
plot2 <- ggplot(applications_161, aes(x = gender)) +
  geom_bar() +
  xlab("Gender") +
  ylab("Count") +
  ggtitle("Distribution of Gender")

# create bar plot of race
plot3 <- ggplot(applications_161, aes(x = race)) +
  geom_bar() +
  xlab("Race") +
  ylab("Count") +
  ggtitle("Distribution of Race")

# vertically stack the plots using grid.arrange
grid.arrange(plot1, plot2, plot3, ncol = 3, widths = c(6, 6, 6), heights = c(8, 8, 8))
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.

## Warning: Removed 2519 rows containing non-finite values (`stat_bin()`).
```

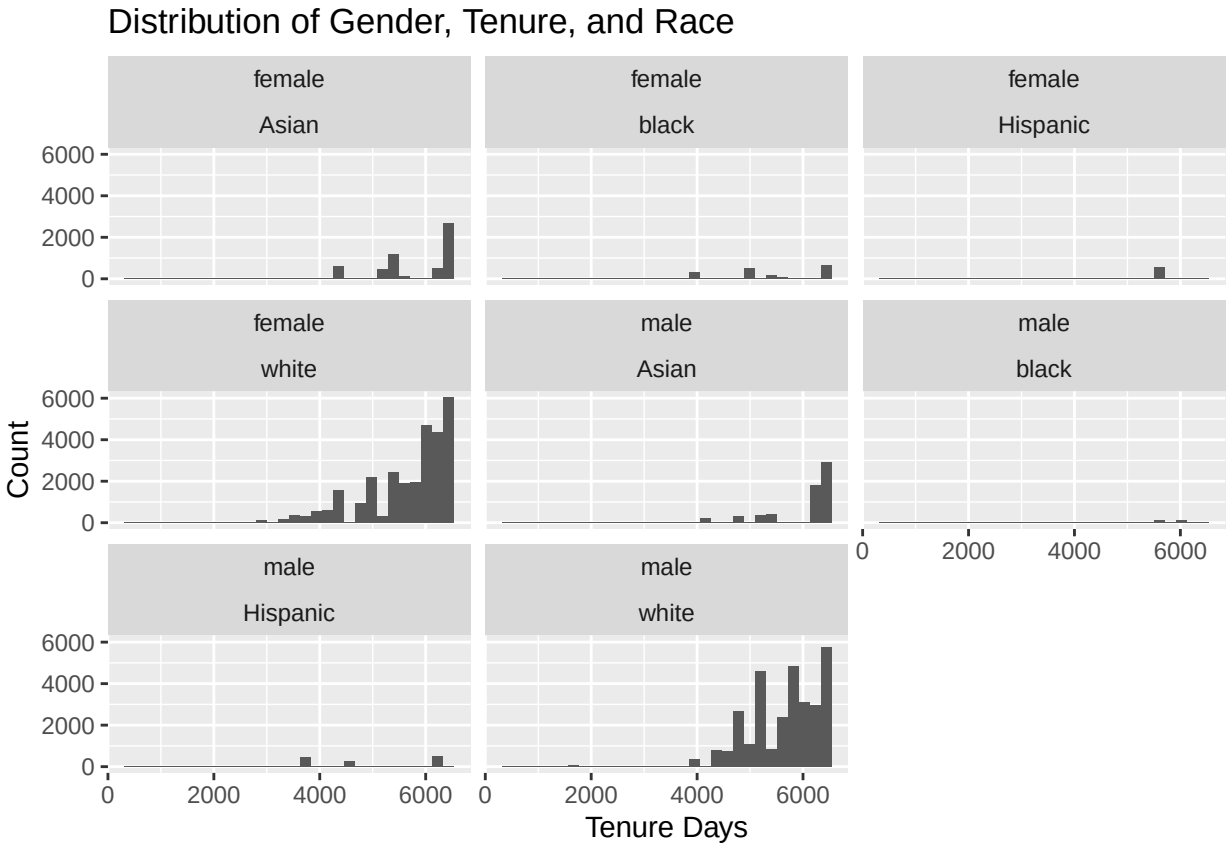


Workgroup 161

```
ggplot(applications_161, aes(x = tenure_days)) +
  geom_histogram() +
  facet_wrap(~gender+race) +
  xlab("Tenure Days") +
  ylab("Count") +
  ggtitle("Distribution of Gender, Tenure, and Race")
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

```
## Warning: Removed 2519 rows containing non-finite values (`stat_bin()`).
```



###Demographics of workgroup 164

```
applications_164 %>% select(tenure_days,race,gender) %>% summary()
```

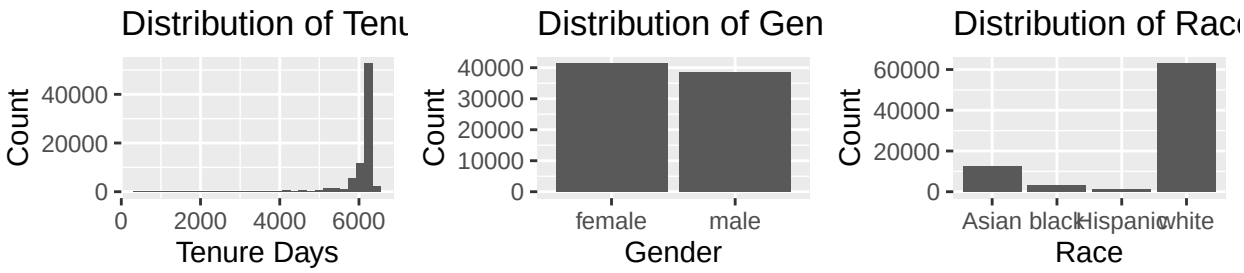
```
##   tenure_days       race              gender
##  Min.   : 314   Length:79791       Length:79791
##  1st Qu.:6072   Class :character   Class :character
##  Median :6317   Mode  :character   Mode  :character
##  Mean   :6126
##  3rd Qu.:6341
##  Max.   :6350
##  NA's   :1073
```

Workgroup 164

```
library(gridExtra)
# create histogram of tenure days
plot1 <- ggplot(applications_164, aes(x = tenure_days)) +
  geom_histogram() +
  xlab("Tenure Days") +
  ylab("Count") +
  ggtitle("Distribution of Tenure Days")

# create bar plot of gender
plot2 <- ggplot(applications_164, aes(x = gender)) +
  geom_bar() +
```

```
  xlab("Gender") +
  ylab("Count") +
  ggtitle("Distribution of Gender")

# create bar plot of race
plot3 <- ggplot(applications_164, aes(x = race)) +
  geom_bar() +
  xlab("Race") +
  ylab("Count") +
  ggtitle("Distribution of Race")

# vertically stack the plots using grid.arrange
grid.arrange(plot1, plot2, plot3, ncol = 3, widths = c(6, 6, 6), heights = c(8, 8, 8))
```

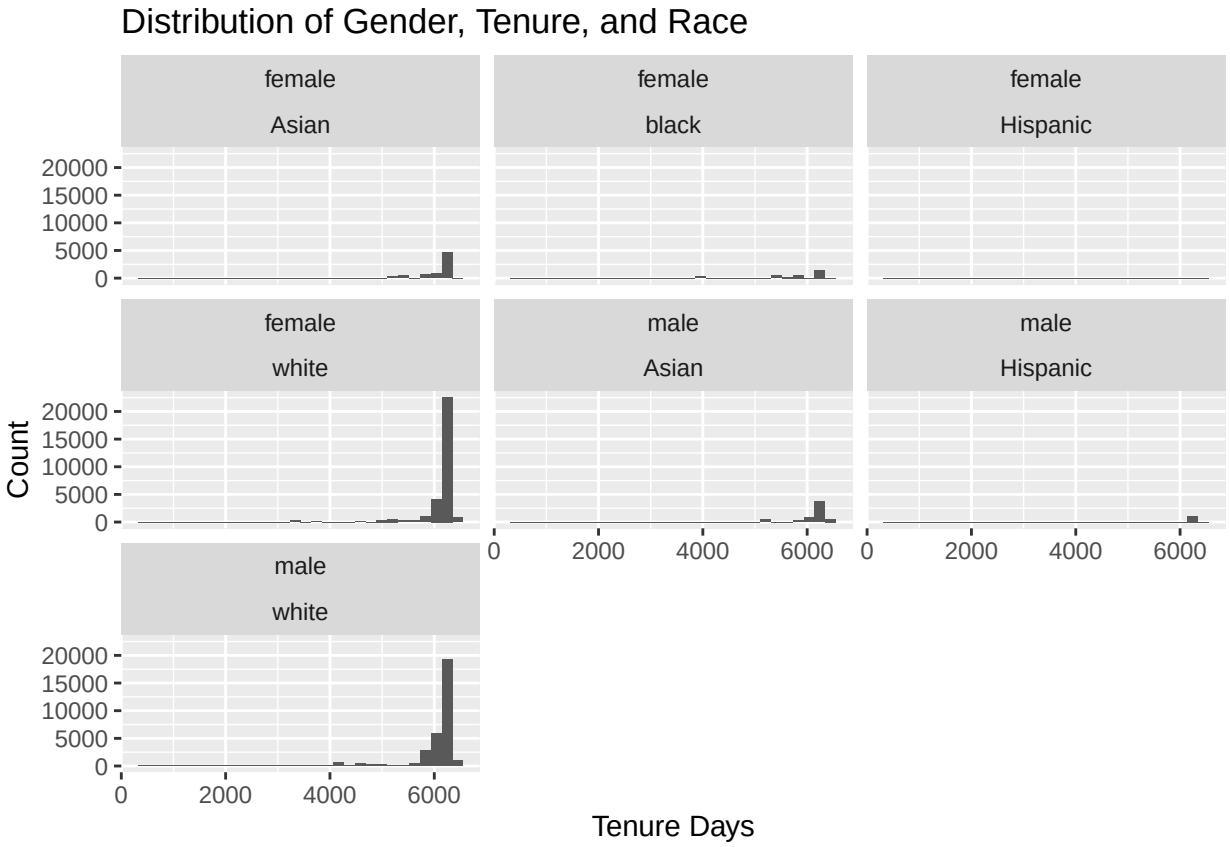
```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.

## Warning: Removed 1073 rows containing non-finite values (`stat_bin()`).
```




```
ggplot(applications_164, aes(x = tenure_days)) +
  geom_histogram() +
  facet_wrap(~gender+race) +
  xlab("Tenure Days") +
  ylab("Count") +
  ggtitle("Distribution of Gender, Tenure, and Race")
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

```
## Warning: Removed 1073 rows containing non-finite values (`stat_bin()`).
```



Workgroup 179

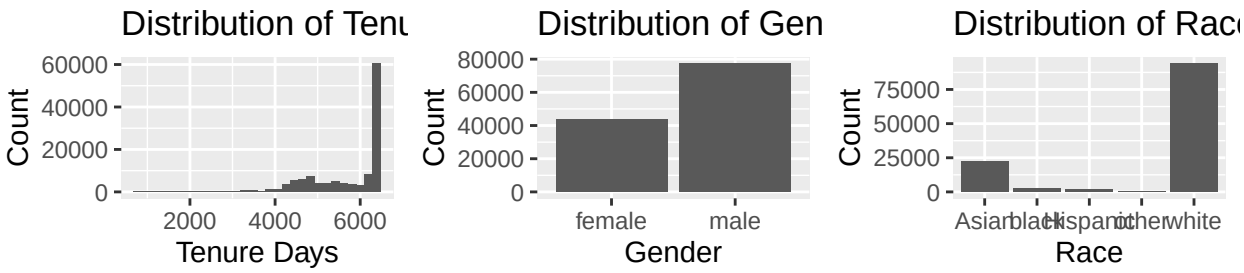
```
applications_179 %>% select(tenure_days,race,gender) %>% summary()
```

```
##   tenure_days       race              gender
##  Min.   : 774   Length:121127      Length:121127
##  1st Qu.:5080   Class :character   Class :character
##  Median :6296   Mode  :character   Mode  :character
##  Mean   :5706
##  3rd Qu.:6342
##  Max.   :6391
##  NA's   :490
```

```
library(gridExtra)
# create histogram of tenure days
plot1 <- ggplot(applications_179, aes(x = tenure_days)) +
  geom_histogram() +
  xlab("Tenure Days") +
  ylab("Count") +
  ggtitle("Distribution of Tenure Days")
```

```
# create bar plot of gender
plot2 <- ggplot(applications_179, aes(x = gender)) +
  geom_bar() +
  xlab("Gender") +
  ylab("Count") +
  ggtitle("Distribution of Gender")

# create bar plot of race
plot3 <- ggplot(applications_179, aes(x = race)) +
  geom_bar() +
  xlab("Race") +
  ylab("Count") +
  ggtitle("Distribution of Race")

# vertically stack the plots using grid.arrange
grid.arrange(plot1, plot2, plot3, ncol = 3, widths = c(6, 6, 6), heights = c(8, 8, 8))
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.

## Warning: Removed 490 rows containing non-finite values (`stat_bin()`).
```

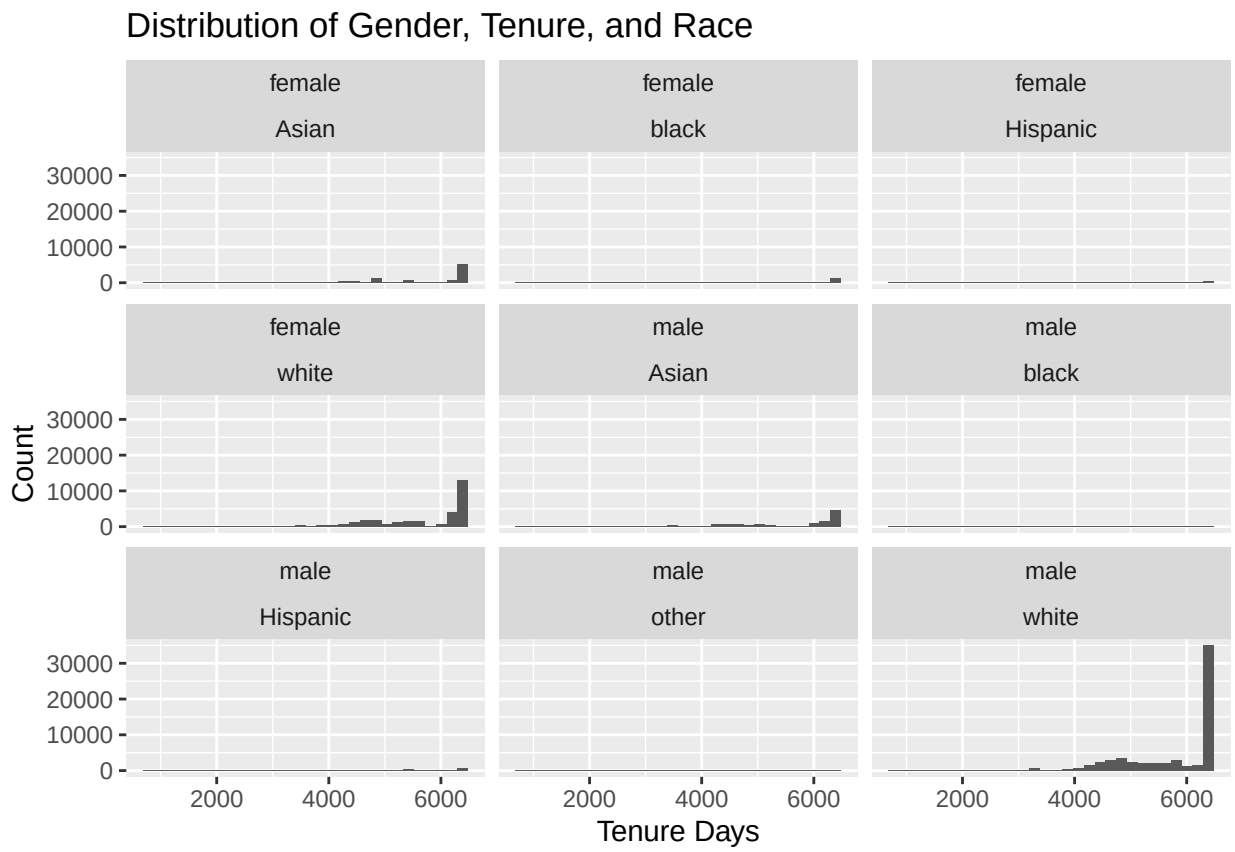



```
ggplot(applications_179, aes(x = tenure_days)) +
  geom_histogram() +
  facet_wrap(~gender+race) +
```

```
  xlab("Tenure Days") +
  ylab("Count") +
  ggtitle("Distribution of Gender, Tenure, and Race")
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

```
## Warning: Removed 490 rows containing non-finite values (`stat_bin()`).
```



Creating the advice networks

```
edges <- edges[complete.cases(edges[c("ego_examiner_id", "alter_examiner_id")]),]
```

```
# load the igraph package
library(igraph)
```

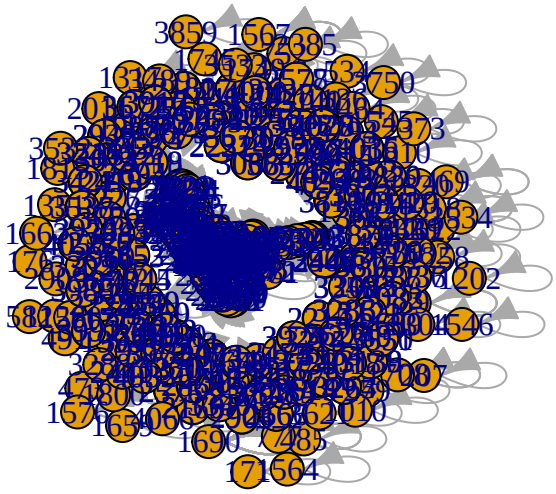
```
## Warning: package 'igraph' was built under R version 4.2.3
```

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

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

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

## 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
```

```
# create an empty graph object
g <- graph.empty()
# add vertices to the graph
vertices <- unique(c(edges$ego_examiner_id, edges$alter_examiner_id))
g <- add.vertices(g, length(vertices), names=vertices)
# add edges to the graph
g <- add.edges(g, cbind(match(edges$ego_examiner_id, vertices), match(edges$alter_examiner_id, vertices)
# plot the network
plot(g)
```



```r
# calculate closeness centrality
cc <- closeness(g)

# sort nodes by closeness centrality in descending order
top_nodes <- order(cc, decreasing = TRUE)[1:100]

# print top 100 nodes and their closeness centrality
cat("Top 100 nodes by closeness centrality:\n")
```

```
## Top 100 nodes by closeness centrality:
```

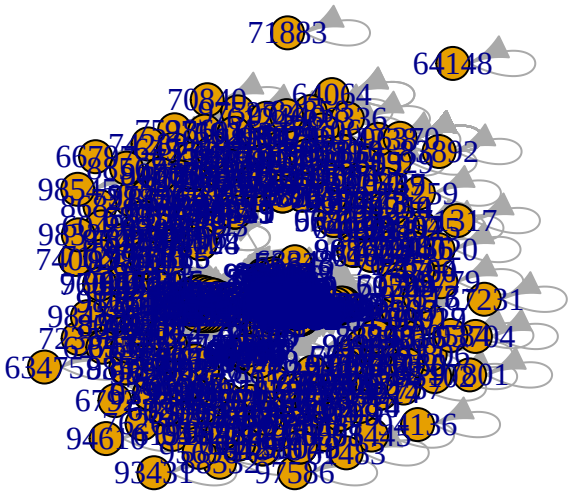
```r
cat(sprintf("%d. %s: %.6f\n", 1:100, V(g)[top_nodes]$name, cc[top_nodes]))
```

```r
detailed_edges <-edges
# Add columns for ego_examiner_id
detailed_edges$ego_examiner_id_gender <- applications1$gender[match(detailed_edges$ego_examiner_id, appl
detailed_edges$ego_examiner_id_race <- applications1$race[match(detailed_edges$ego_examiner_id, applica
detailed_edges$ego_examiner_id_tenure_days <- applications1$tenure_days[match(detailed_edges$ego_examin
detailed_edges$ego_examiner_id_work_group <- applications1$work_group[match(detailed_edges$ego_examiner

# Add columns for alter_examiner_id
detailed_edges$alter_examiner_id_gender <- applications1$gender[match(detailed_edges$alter_examiner_id,
detailed_edges$alter_examiner_id_race <- applications1$race[match(detailed_edges$alter_examiner_id, appl
detailed_edges$alter_examiner_id_tenure_days <- applications1$tenure_days[match(detailed_edges$alter_ex
detailed_edges$alter_examiner_id_work_group <- applications1$work_group[match(detailed_edges$alter_exam
```

Removing the NULL Values

```r
detailed_edges <- detailed_edges %>%
  filter(!is.na(ego_examiner_id_gender) & !is.na(ego_examiner_id_race) &
         !is.na(ego_examiner_id_tenure_days) & !is.na(ego_examiner_id_work_group) &
         !is.na(alter_examiner_id_gender) & !is.na(alter_examiner_id_race) &
         !is.na(alter_examiner_id_tenure_days) & !is.na(alter_examiner_id_work_group))
```

```r
# load the igraph package
library(igraph)
# create an empty graph object
g <- graph.empty()
# add vertices to the graph and set their names
vertices <- unique(c(edges$ego_examiner_id, edges$alter_examiner_id))
g <- add.vertices(g, length(vertices))
V(g)$name <- vertices
# add edges to the graph
g <- add.edges(g, cbind(match(edges$ego_examiner_id, vertices), match(edges$alter_examiner_id, vertices
# plot the network
plot(g)
```



Closeness centrality

Closeness centrality is based on the average length of the shortest path between a node and all other nodes in the graph. If a node has a high closeness centrality, it means that it can reach other nodes in the graph with fewer steps, and other nodes can reach it with fewer steps as well. These are the most reachable nodes or examiners. They would be most accessible for the advice

```
# calculate closeness centrality
cc <- closeness(g)

# sort nodes by closeness centrality in descending order
top_nodes <- order(cc, decreasing = TRUE)[1:25]

# print top 100 nodes and their closeness centrality
cat("Top 25 nodes by closeness centrality:\n")
```

```
## Top 25 nodes by closeness centrality:
```

```
cat(sprintf("%d. %s: %.6f\n", 1:25, V(g)[top_nodes]$name, cc[top_nodes]))
```

```
## 1. 94500: 1.000000
##  2. 69849: 1.000000
##  3. 68377: 1.000000
##  4. 61558: 1.000000
##  5. 69293: 1.000000
##  6. 86115: 1.000000
```

```
##  7. 84925: 1.000000
##  8. 60593: 1.000000
##  9. 97072: 1.000000
##  10. 64078: 1.000000
##  11. 79977: 1.000000
##  12. 80742: 1.000000
##  13. 68912: 1.000000
##  14. 80167: 1.000000
##  15. 96168: 1.000000
##  16. 74313: 1.000000
##  17. 81876: 1.000000
##  18. 99771: 1.000000
##  19. 60567: 1.000000
##  20. 81838: 1.000000
##  21. 95234: 1.000000
##  22. 68618: 1.000000
##  23. 73807: 1.000000
##  24. 99277: 1.000000
##  25. 71237: 1.000000
```

Betweenness centrality

Betweenness centrality is a measure of node importance in a graph that is based on the number of shortest paths between all pairs of nodes that pass through a particular node. Nodes with high betweenness centrality are those that lie on many of the shortest paths between other nodes in the graph. These nodes act as a bridge or bottleneck, facilitating the flow of information or resources between different parts of the network.

```
# calculate closeness centrality
cc <- betweenness(g)

# sort nodes by closeness centrality in descending order
top_nodes <- order(cc, decreasing = TRUE)[1:25]

# print top 100 nodes and their closeness centrality
cat("Top 25 nodes by closeness centrality:\n")
```

```
## Top 25 nodes by closeness centrality:
```

```
cat(sprintf("%d. %s: %.6f\n", 1:25, V(g)[top_nodes]$name, cc[top_nodes]))
```

```
## 1. 92569: 1854213.237937
##  2. 91499: 464692.815663
##  3. 62242: 423511.515183
##  4. 62528: 368205.921965
##  5. 92078: 361584.860710
##  6. 62919: 332976.568830
##  7. 90956: 292291.869746
##  8. 92028: 285095.454483
##  9. 65231: 228021.201204
##  10. 67389: 227232.068797
##  11. 69098: 222551.269620
##  12. 64053: 221894.560032
##  13. 72882: 220392.918222
```

```
##  14. 91824: 216787.495441
##  15. 94987: 215914.693232
##  16. 77648: 199950.032628
##  17. 61485: 192078.569696
##  18. 83222: 190549.951235
##  19. 70835: 171696.574052
##  20. 68532: 164997.517183
##  21. 71353: 161881.031081
##  22. 78350: 153487.629104
##  23. 95398: 152698.304392
##  24. 99395: 152658.183158
##  25. 76536: 149634.547513
```

Degree Centrality

Degree centrality is a measure of node importance in a graph. It is based on the number of edges that are incident on a node, i.e., the number of connections that a node has with other nodes in the graph. Nodes with high degree centrality are those that are well-connected to other nodes in the graph and are therefore important in terms of information flow, communication, or influence. These are the examiners whom a lot of people take advice from.

```
# calculate degree  centrality
cc <- degree(g)

# sort nodes by closeness centrality in descending order
top_nodes <- order(cc, decreasing = TRUE)[1:25]

# print top 100 nodes and their closeness centrality
cat("Top 25 nodes by closeness centrality:\n")
```

```
## Top 25 nodes by closeness centrality:
```

```
cat(sprintf("%d. %s: %.6f\n", 1:25, V(g)[top_nodes]$name, cc[top_nodes]))
```

```
## 1. 83670: 699.000000
##  2. 92569: 644.000000
##  3. 62919: 335.000000
##  4. 67226: 315.000000
##  5. 91499: 315.000000
##  6. 61485: 288.000000
##  7. 62528: 258.000000
##  8. 91824: 252.000000
##  9. 92078: 245.000000
##  10. 62242: 244.000000
##  11. 71764: 233.000000
##  12. 90967: 232.000000
##  13. 73920: 210.000000
##  14. 60023: 203.000000
##  15. 93896: 194.000000
##  16. 97910: 193.000000
##  17. 83222: 183.000000
##  18. 77648: 180.000000
##  19. 95183: 176.000000
```

```
##  20. 92028: 176.000000
##  21. 65231: 171.000000
##  22. 71353: 167.000000
##  23. 60897: 167.000000
##  24. 98690: 164.000000
##  25. 67389: 163.000000
```