Lab 9: Graphs and Social Networks in Pride & Prejudice

Alex Ptacek

Overview

In the this lab assignment, you are going to construct a social network from the characters in the book "Pride & Prejudice", a novel written by Jane Austen and available in the `janeaustenr` package. The social network will be a weighted graph connecting the characters, where the weight is equal to the number of times the names of each character appeared in each 10 line section of the book. Once you create the graph, you will load it into tidygraph, make a visualization of the graph, and rank the most connected characters by a measure called `degree centrality`

Problem 1

Load the text of Pride & Prejudice into R using the `janeaustenr` library. Then download and read pride_prejudice_characters.csv, the csv file from my github page containing a list of characters in `Pride & Prejudice` and their aliases. Here aliases refers to the different names that the characters go by in the books, for example "Darcy" also goes by the names "Mr. Darcy", and "Mr. Fitzwilliam Darcy" (not to be confused with his cousin "Colonel Fitzwilliam").

Process the text of Pride & Prejudice to replace instances where an alias occurs with the full name of the character- I recommend using the iteration techniques you learned ealier, I arranged the order of names in the `csv` file to minimize misidentifications if you replace names in the order that they appear in the file. Making this perfect would require a bit of effort but we are ok if there are some misidentifications. Here the final name of each character will be a single word.

```
library(tidyverse)
library(tidytext)
library(janeaustenr)
library(widyr)
library(tidygraph)
library(ggraph)
```

```
base_text <- austen_books() |>
                      filter(book == "Pride & Prejudice")
pride_and_prej_text <- austen_books() |>
                      filter(book == "Pride & Prejudice")

pp_chars <- read_csv("https://raw.githubusercontent.com/georgehagstrom/DATA607/refs/heads/ma


adj_chars <- pp_chars |>
  mutate(check = ifelse(unique_name == alias, TRUE, FALSE)) |>
  filter(check == FALSE) |>
  select(1:3)


fun_chars <- str_flatten(adj_chars$alias, "|")
fun_chars <- str_c('"', "^", fun_chars, "$", '"')



text_match_df <- tibble(x = ifelse(str_detect(pride_and_prej_text$text, "Jane Austen") == TR
                            pride_and_prej_text$text,
                            str_match_all(pride_and_prej_text$text, fun_chars))) |>
                 mutate(x = as.character(x),
                        x = str_remove_all(x, "c\\(|\\)"))



text_match_df <- text_match_df |>
  separate_wider_delim(x, ", ", names = c("x1", "x2", "x3", "x4"), too_few = "align_start" )
  mutate(across(x1:x4, \(x) str_remove_all(x, '"'))) |>
  left_join(pp_chars, by = join_by(x1 == alias)) |>
  left_join(pp_chars, by = join_by(x2 == alias)) |>
  left_join(pp_chars, by = join_by(x3 == alias)) |>
  left_join(pp_chars, by = join_by(x4 == alias)) |>
  select(6,8,10,12) |>
  rename_at(vars(unique_name.x:unique_name.y.y), ~c("one", "two", "three", "four"))



pride_and_prej_text <- pride_and_prej_text |>
  mutate(text = ifelse(str_detect(pride_and_prej_text$text, "Jane Austen") == TRUE,
                       pride_and_prej_text$text,
```

```
                        str_replace(pride_and_prej_text$text, fun_chars, text_match_df$one)))
```

Problem 2

Following the example in chapter 4 of the text mining with R book, create a new column in the data frame corresponding to the Pride & Prejudice text that divides the text into sections of 10 lines each. Then use the `pairwise_count` function from `widyr` to determine the number of times each name occurs with each other name in the same 10 line section.

```
ten_lines_text <- pride_and_prej_text |>
  mutate(section = row_number() %/% 10) |>
  filter(section > 0) |>
  unnest_tokens(word, text) |>
  filter(!word %in% stop_words$word)

word_pairs <- ten_lines_text |>
  pairwise_count(word, section, sort = TRUE, upper = FALSE)
```

Problem 3

Create a dataframe of nodes which contains the id and unique names of each character, and create a dataframe of edges which contains three columns: a column named `from`, a column named `to`, and a column named `weight`, where the `from` and `to` are the id numbers of each character and `weight` is the number of co-occurences you found in Problem 2. Each pair should only appear once in the edge list (i.e. Elizabeth and MrDarcy but not MrDarcy and then Elizabeth). Create a tidygraph object using `tbl_graph` that contains the social network data that we just constructed.

```
adj_chars <- adj_chars |>
  mutate(unique_name = str_to_lower(unique_name))

word_pairs <-  word_pairs |>
  filter(item1 %in% adj_chars$unique_name,
         item2 %in% adj_chars$unique_name)

word_pairs <- word_pairs |>
  rename(from = item1,
         to = item2,
         weight = n)

name_graph <- as_tbl_graph(word_pairs)
```

```
name_graph <- name_graph |>
  activate(nodes) |>
  mutate(title = str_to_title(name),
         label = str_replace_all(title, " ", "\n"))
```

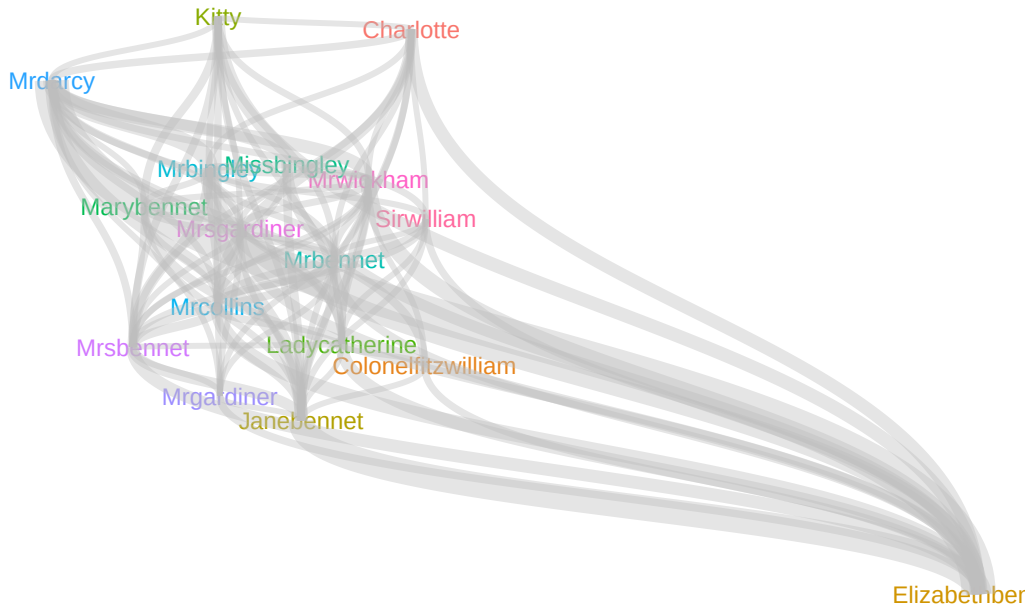
Problem 4

Using `ggraph`, graph the connections between the characters. Make sure that each node is labeled by the character name, and make sure that the weight is represented by the thickness of the edge plotted between the two nodes. Then use the `centrality_degree` function to calculate the weighted degree centrality of each character, and make a plot which shows the degree centrality of each character where the characters are arranged in order of degree centrality.

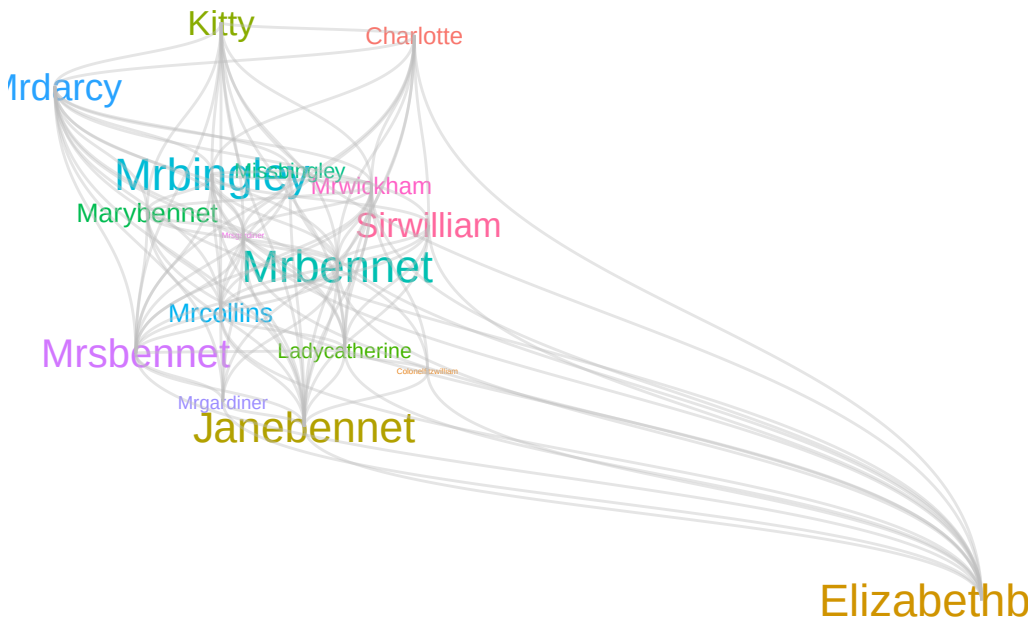
```
thm <- theme_minimal() +
  theme(
    legend.position = "none",
     axis.title = element_blank(),
     axis.text = element_blank(),
     panel.grid = element_blank(),
     panel.grid.major = element_blank(),
  )
theme_set(thm)


name_graph |>
  ggraph(layout = "kk") +
    geom_node_text(aes(label = label, color = name), size = 3) +
    geom_edge_diagonal(aes(width = weight), color = "gray", alpha = .4)
```

```
Warning: The `trans` argument of `continuous_scale()` is deprecated as of ggplot2 3.5.0.
i Please use the `transform` argument instead.
```




```
name_graph |>
  activate(nodes) |>
  mutate(deg_centrality = centrality_degree()) |>
  ggraph(layout = "kk") +
    geom_node_text(aes(label = label, color = name, size = deg_centrality)) +
    geom_edge_diagonal(color = "gray", alpha = 0.4)
```


Kitty
Charlotte
Mrdarcy
Mrbingley
Missbingley
Mrwickham
Marybennet
Sirwilliam
Mrbennet
Mrcollins
Mrsbennet
Ladycatherine
Mrgardiner
Janebennet
Elizabethb