Romeo & Juliet

Get data:

The `gutenberg_download` function retrieves texts from Project Gutenberg but it's good practice not to keep hitting the server, so I've downloaded the files and saved them locally.

```
#RomeoJuliet <- gutenberg_download(1513)
#RomeoJuliet <- gutenberg_download(1532)
RomeoJuliet <- read_csv("TextData/RandJText.csv")
```

Explore

Let's look at the scenes.

```
scenes <- str_subset(RomeoJuliet$text, "Scene")
scenes
```

```
##  [1] "Scene I. A public place."
##  [2] "Scene II. A Street."
##  [3] "Scene III. Room in Capulet's House."
##  [4] "Scene IV. A Street."
##  [5] "Scene V. A Hall in Capulet's House."
##  [6] "Scene I. An open place adjoining Capulet's Garden."
##  [7] "Scene II. Capulet's Garden."
##  [8] "Scene III. Friar Lawrence's Cell."
##  [9] "Scene IV. A Street."
## [10] "Scene V. Capulet's Garden."
## [11] "Scene VI. Friar Lawrence's Cell."
## [12] "Scene I. A public Place."
## [13] "Scene II. A Room in Capulet's House."
## [14] "Scene III. Friar Lawrence's cell."
## [15] "Scene IV. A Room in Capulet's House."
## [16] "Scene V. An open Gallery to Juliet's Chamber, overlooking the"
## [17] "Scene I. Friar Lawrence's Cell."
## [18] "Scene II. Hall in Capulet's House."
## [19] "Scene III. Juliet's Chamber."
## [20] "Scene IV. Hall in Capulet's House."
## [21] "Scene V. Juliet's Chamber; Juliet on the bed."
## [22] "Scene I. Mantua. A Street."
## [23] "Scene II. Friar Lawrence's Cell."
## [24] "Scene III. A churchyard; in it a Monument belonging to the"
```

Let's look at the stage directions.

We are going to be using regular expressions a lot. There's loads of information out there, but my favourite site is: https://www.rexegg.com/ (https://www.rexegg.com/)

```
stage_directions <- str_subset(RomeoJuliet$text, "\\[")
head(stage_directions, 20)
```

```
##  [1] "[Enter Chorus.]"
##  [2] "[Enter Sampson and Gregory armed with swords and bucklers.]"
##  [3] "[Enter Abraham and Balthasar.]"
##  [4] "[They fight.]"
##  [5] "[Enter Benvolio.]"
##  [6] "[Beats down their swords.]"
##  [7] "[Enter Tybalt.]"
##  [8] "[They fight.]"
##  [9] "[Enter several of both Houses, who join the fray; then enter"
## [10] "[Enter Capulet in his gown, and Lady Capulet.]"
## [11] "[Enter Montague and his Lady  Montague.]"
## [12] "[Enter Prince, with Attendants.]"
## [13] "[Exeunt Prince and Attendants; Capulet, Lady Capulet, Tybalt,"
## [14] "[Exeunt Montague and Lady.]"
## [15] "[Enter Romeo.]"
## [16] "[Going.]"
## [17] "[Exeunt.]"
## [18] "[Enter Capulet, Paris, and Servant.]"
## [19] "Whose names are written there, [gives a paper] and to them say,"
## [20] "[Exeunt Capulet and Paris]."
```

Speaker first lines

```
they_said <- function(speaker){
  pattern <- paste("^", speaker, "\\.", sep = "")
  RomeoJuliet$text[str_which(RomeoJuliet$text, pattern) + 1]
}

head(they_said("Romeo"), 20)
```

```
##  [1] "Is the day so young?"
##  [2] "Ay me! sad hours seem long."
##  [3] "Not having that which, having, makes them short."
##  [4] "Out,--"
##  [5] "Out of her favour where I am in love."
##  [6] "Alas that love, whose view is muffled still,"
##  [7] "Good heart, at what?"
##  [8] "Why, such is love's transgression.--"
##  [9] "Tut! I have lost myself; I am not here:"
## [10] "What, shall I groan and tell thee?"
## [11] "Bid a sick man in sadness make his will,--"
## [12] "A right good markman!--And she's fair I love."
## [13] "Well, in that hit you miss: she'll not be hit"
## [14] "She hath, and in that sparing makes huge waste;"
## [15] "O, teach me how I should forget to think."
## [16] "'Tis the way"
## [17] "Your plantain-leaf is excellent for that."
## [18] "For your broken shin."
## [19] "Not mad, but bound more than a madman is;"
## [20] "Ay, mine own fortune in my misery."
```

Can we detect speakers?

```
speakers <- str_subset(RomeoJuliet$text, "^[A-Z]\\w+\\.$")
unique(speakers)
```

```
##  [1] "Chorus."     "Chor."       "Sampson."    "Gregory."    "Abraham."
##  [6] "No."         "Benvolio."   "Tybalt."     "Capulet."    "Montague."
## [11] "Prince."     "Romeo."      "Paris."      "Servant."    "Up."
## [16] "Nurse."      "Juliet."     "Mercutio."   "Friar."      "Right."
## [21] "Peter."      "Anon."       "Farewell."   "Garden."     "Balthasar."
## [26] "Apothecary." "Capulets."   "Page."       "Boy."
```

```
speakers <- str_subset(RomeoJuliet$text, "^[A-Z]\\w+\\.$|^[A-Z]\\w+\\s+[A-Z]\\w+\\
.$")
unique(speakers)
```

```
##  [1] "An Apothecary."   "Three Musicians." "Chorus."
##  [4] "An Officer."      "Chor."            "Sampson."
##  [7] "Gregory."         "Abraham."         "No."
## [10] "Benvolio."        "Tybalt."          "Capulet."
## [13] "Lady Capulet."    "Montague."        "Lady  Montague."
## [16] "Prince."          "Lady Montague."   "Romeo."
## [19] "Paris."           "Servant."         "Up."
## [22] "Nurse."           "Juliet."          "Mercutio."
## [25] "ACT II."          "Friar."           "Right."
## [28] "Peter."           "Anon."            "ACT III."
## [31] "Farewell."        "Garden."          "ACT IV."
## [34] "Balthasar."       "Apothecary."      "Friar John."
## [37] "Friar Lawrence."  "Capulets."        "Page."
## [40] "Boy."
```

Question - Are Romeo & Juliet compatible?

Maybe they are if their speech is similar.

Restructuring the data

```
RJ_processed <-
  RomeoJuliet %>%
  select(-gutenberg_id) %>%
# Remove the ID column
  filter(row_number() >= str_which(RomeoJuliet$text, "^ACT")[1]) %>%
# Remove all the lines before Act 1
  filter(!str_detect(text, "^ACT")) %>%
# Remove all the lines starting with ACT
  filter(!str_detect(text, "^Scene")) %>%
# Remove all the lines starting with Scene
  filter(!str_detect(text, "^\\[.+\\]")) %>%
# Remove all the lines that look like [...]
  filter(text != "") %>%
# Remove all the blank lines
  mutate(Change = str_detect(text, "^[A-Z]\\w+\\.$|^[A-Z]\\w+\\s+[A-Z]\\w+\\.$" ))
%>%     # Add a column to show when there is a new speaker
  mutate(Speaker = "")
# Create a column for the Speaker's name
```

```r
# Add the speaker's name to each column
speaker <- ""
for (i in seq_along(RJ_processed$Change)){
 if (RJ_processed$Change[i]){
   speaker <- RJ_processed$text[i]
 }
 else {
   RJ_processed$Speaker[i] <-  speaker
 }
}

RJ_processed <-
  RJ_processed %>%
  select(-Change) %>%
  filter(Speaker != "")
```

RJ_processed is our key data structure, we have the lines in one column and the speaker in another:

```r
head(RJ_processed,20)
```

```
## # A tibble: 20 x 2
##    text                                                              Speaker
##    <chr>                                                             <chr>
##  1 Gregory, o' my word, we'll not carry coals.                       Sampso~
##  2 No, for then we should be colliers.                               Gregor~
##  3 I mean, an we be in choler we'll draw.                            Sampso~
##  4 Ay, while you live, draw your neck out o' the collar.             Gregor~
##  5 I strike quickly, being moved.                                    Sampso~
##  6 But thou art not quickly moved to strike.                         Gregor~
##  7 A dog of the house of Montague moves me.                          Sampso~
##  8 To move is to stir; and to be valiant is to stand:                Gregor~
##  9 therefore, if thou art moved, thou runn'st away.                  Gregor~
## 10 A dog of that house shall move me to stand:                       Sampso~
## 11 I will take the wall of any man or maid of Montague's.            Sampso~
## 12 That shows thee a weak slave; for the weakest goes to the         Gregor~
## 13 wall.                                                             Gregor~
## 14 True; and therefore women, being the weaker vessels,              Sampso~
## 15 are ever thrust to the wall: therefore I will push Montague's m~ Sampso~
## 16 from the wall and thrust his maids to the wall.                   Sampso~
## 17 The quarrel is between our masters and us their men.             Gregor~
## 18 'Tis all one, I will show myself a tyrant:                        Sampso~
## 19 when I have fought with the men I will be cruel with the maids,   Sampso~
## 20 I will cut off their heads.                                       Sampso~
```

Who has most lines?

```
RJ_lines <-
RJ_processed %>%
  group_by(Speaker) %>%
  summarise(Lines = n()) %>%
  arrange(desc(Lines))
RJ_lines
```

```
## # A tibble: 26 x 2
##    Speaker        Lines
##    <chr>          <int>
##  1 Romeo.           605
##  2 Juliet.          543
##  3 Friar.           336
##  4 Capulet.         294
##  5 Nurse.           281
##  6 Mercutio.        240
##  7 Benvolio.        185
##  8 Lady Capulet.    112
##  9 Prince.           85
## 10 Paris.            69
## # ... with 16 more rows
```

```
RJ_top_speakers <-
  RJ_lines %>%
  filter(Lines > 100)

RJ_top_speakers <- RJ_top_speakers$Speaker
```

Splitting into words

The reference for this next bit is: https://www.tidytextmining.com/ (https://www.tidytextmining.com/)

```
RJ_tidy <-
  RJ_processed %>%
  unnest_tokens(word, text) %>%
  anti_join(stop_words)
```

```
RJ_word_counts <-
  RJ_tidy %>%
  group_by(Speaker, word) %>%
  summarise(Count = n())
```

Commonest words

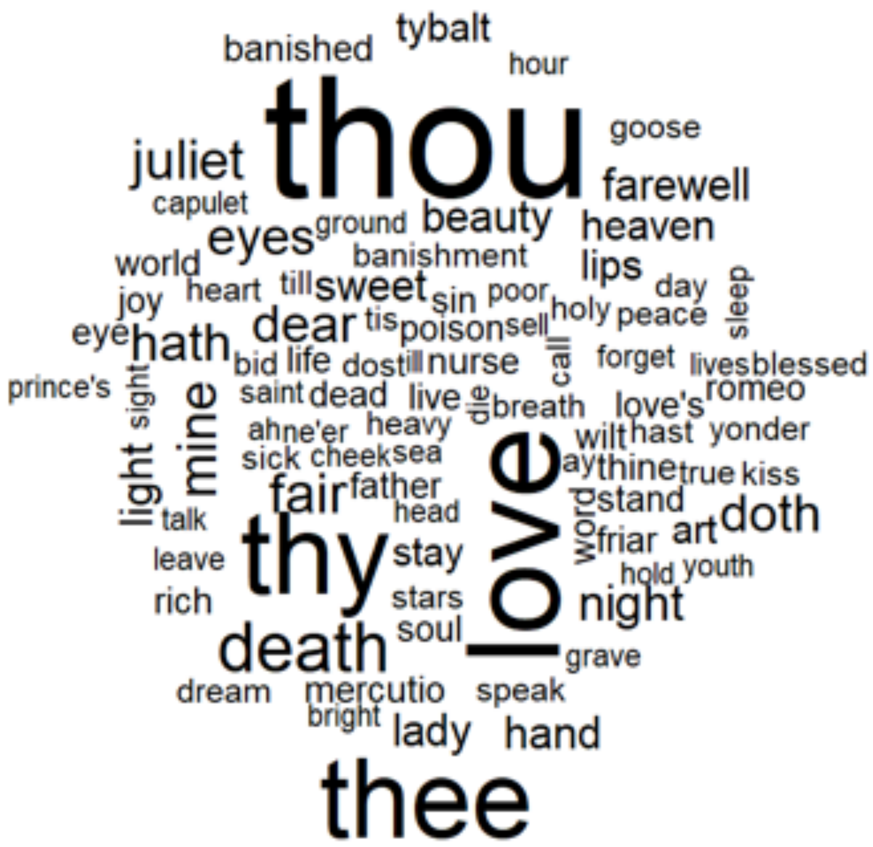
```
RJ_word_counts %>%
  group_by(word) %>%
  summarize(number = sum(Count)) %>%
  arrange(desc(number))
```

```
## # A tibble: 3,281 x 2
##    word  number
##    <chr>  <int>
##  1 thou     276
##  2 thy      165
##  3 love     139
##  4 thee     139
##  5 romeo    114
##  6 night     83
##  7 death     69
##  8 hath      64
##  9 sir       58
## 10 art       55
## # ... with 3,271 more rows
```

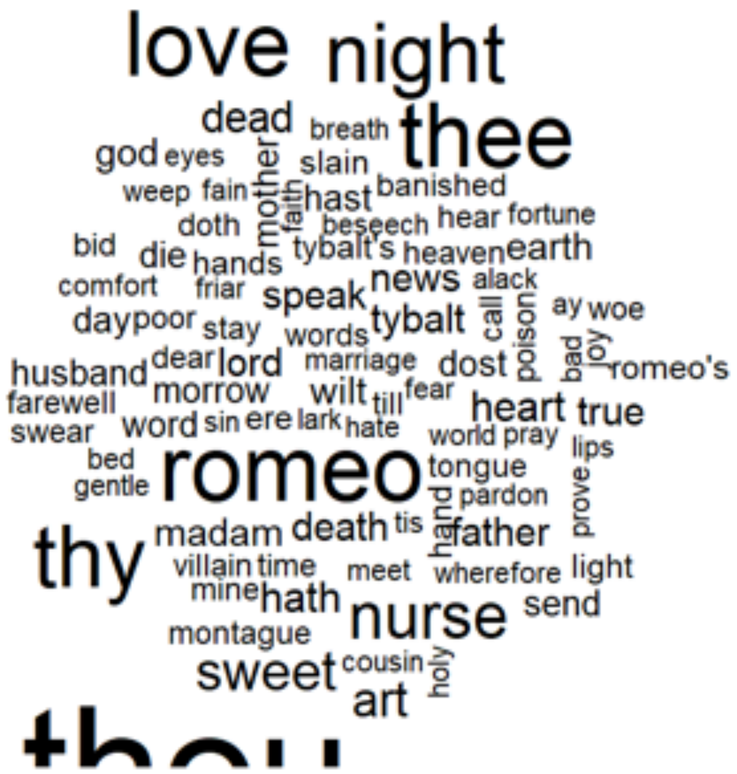
Word clouds

```
# Romeo_words <-
#   RJ_word_counts %>%
#   filter(Speaker == "Romeo.") %>%
#   arrange(desc(Count))
#
# Juliet_words <-
#   RJ_word_counts %>%
#   filter(Speaker == "Juliet.") %>%
#   arrange(desc(Count))
#
# Merc_words <-
#   RJ_word_counts %>%
#   filter(Speaker == "Mercutio.") %>%
#   arrange(desc(Count))
```

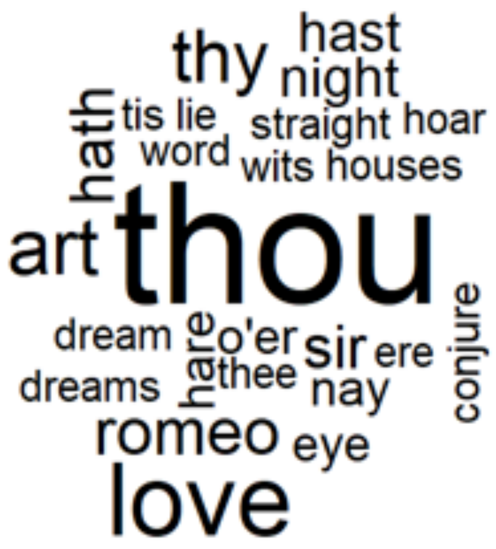
```
wcloud <- function(wcounts, speaker, min_freq = 4){
  words <- filter(wcounts, Speaker == speaker)$word
  counts <- filter(wcounts, Speaker == speaker)$Count
  wordcloud(words, counts, min.freq = min_freq)
}
wcloud(RJ_word_counts, "Romeo.")
```



```
wcloud(RJ_word_counts, "Juliet.")
```



```
wcloud(RJ_word_counts, "Mercutio.")
```



Hmm. Lots of thous, thees and thys. Looks like we need a Shakesperian stop words list methinks.

How can we measure the closeness of two speeches?

```
dtm <-
  RJ_word_counts %>%
  filter(Speaker %in% RJ_top_speakers) %>%
  cast_dtm(Speaker, word, Count)
```

```
dtm <- dtm/sqrt(row_sums(dtm^2))
euc_dist <- tcrossapply_simple_triplet_matrix(dtm, FUN = function(x,y) sqrt(sum((x
-y)^2)))
euc_dist[upper.tri(euc_dist, diag = TRUE)] <- 1
lovers <- which(euc_dist == min(euc_dist), arr.ind = TRUE)
```

This section needs a lot more explanation and work, but…

…The most compatible characters are…

Romeo.

and

Juliet.