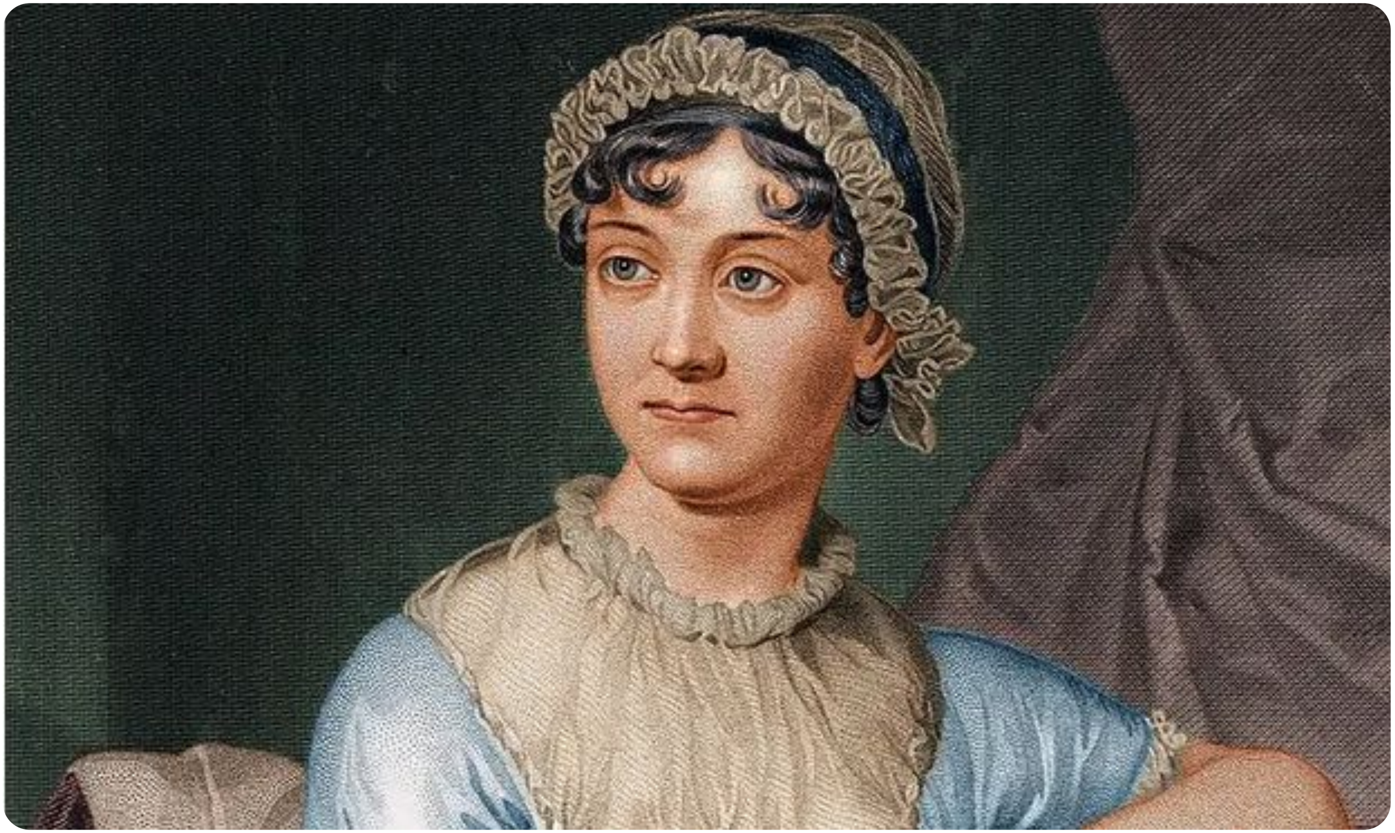
# Text Mining 2

Stat 133 with Gaston Sanchez

# “tidytext” package

```
library(tidyverse)      # base tidy data tools

library(tidytext)       # text mining package that plays
                        # very well with tidyverse

library(janeaustenr)    # Jane Austen’s novels

library(wordcloud)      # for plotting word-clouds

library(igraph)         # for computing networks

library(ggraph)         # graphing networks
                        # “a la” ggplot2
```

# Portrait of Jane Austen



https://en.wikipedia.org/wiki/Jane_Austen

```
# “Pride and Prejudice” by Jane Austen
head(prideprejudice, 11)
```

```
 [1] "PRIDE AND PREJUDICE"
 [2] ""
 [3] "By Jane Austen"
 [4] ""
 [5] ""
 [6] ""
 [7] "Chapter 1"
 [8] ""
 [9] ""
[10] "It is a truth universally acknowledged, that a
single man in possession"
[11] "of a good fortune, must be in want of a wife."
```

```
# tokenization
pride = data.frame(text = prideprejudice)
tidy_pride = unnest_tokens(pride, word, text)
tidy_pride
```

```
        word
1      pride
2        and
3  prejudice
4         by
5       jane
6     austen
7    chapter
8          1
9         it
10        is
...
```

# Default behavior of unnest_tokens()

- Each row is split so that there is one token in the output data frame (or tibble).
- Other columns, such as the line number each word came from, are retained.
- Punctuation has been stripped.
- Converts the tokens to lowercase.
- See `?unnest_tokens` for more info.

# Some common text transformations

Convert to lower case

Remove punctuation symbols

Remove extra spaces

Remove digits

# Word Frequencies

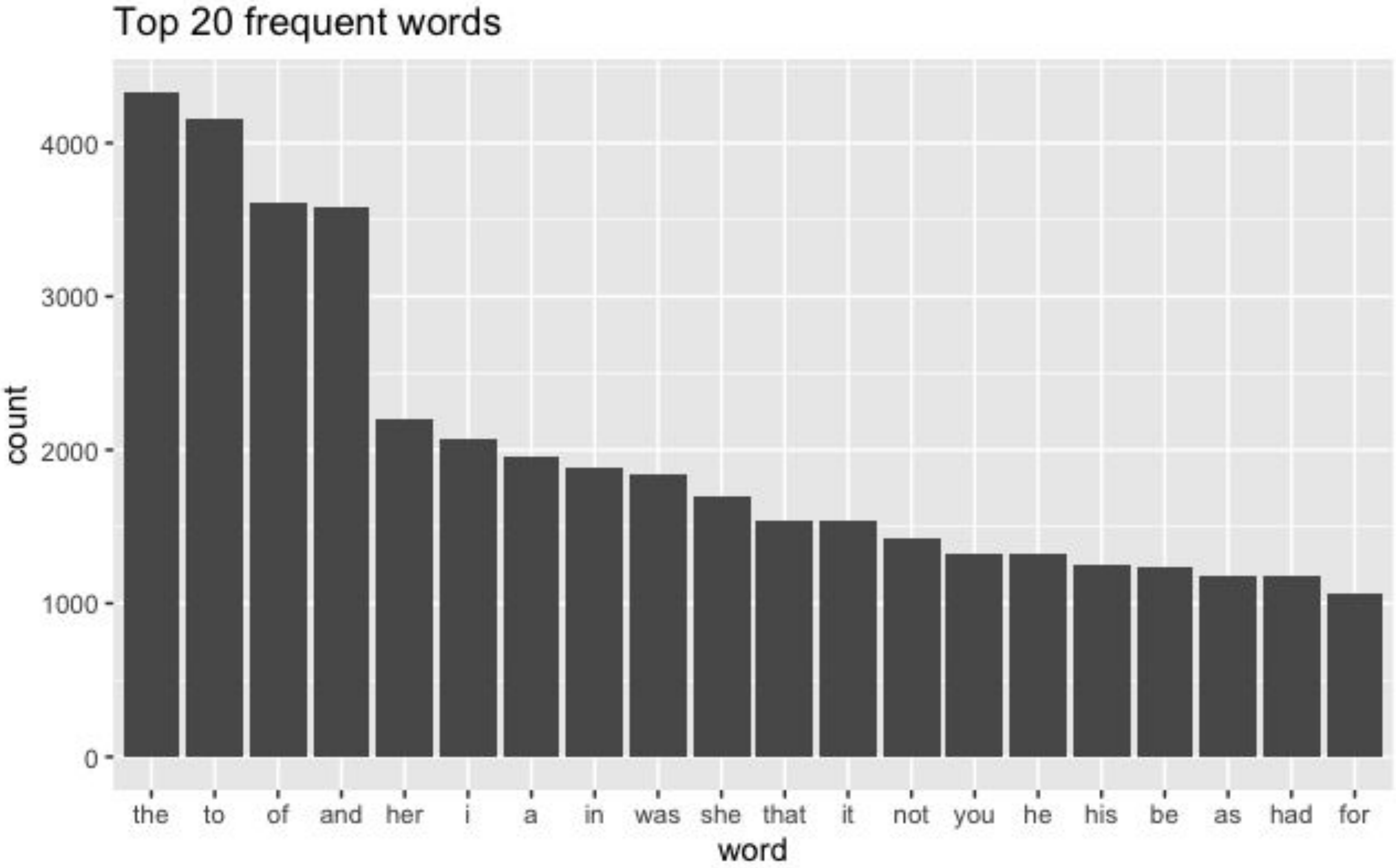
## Word Frequencies

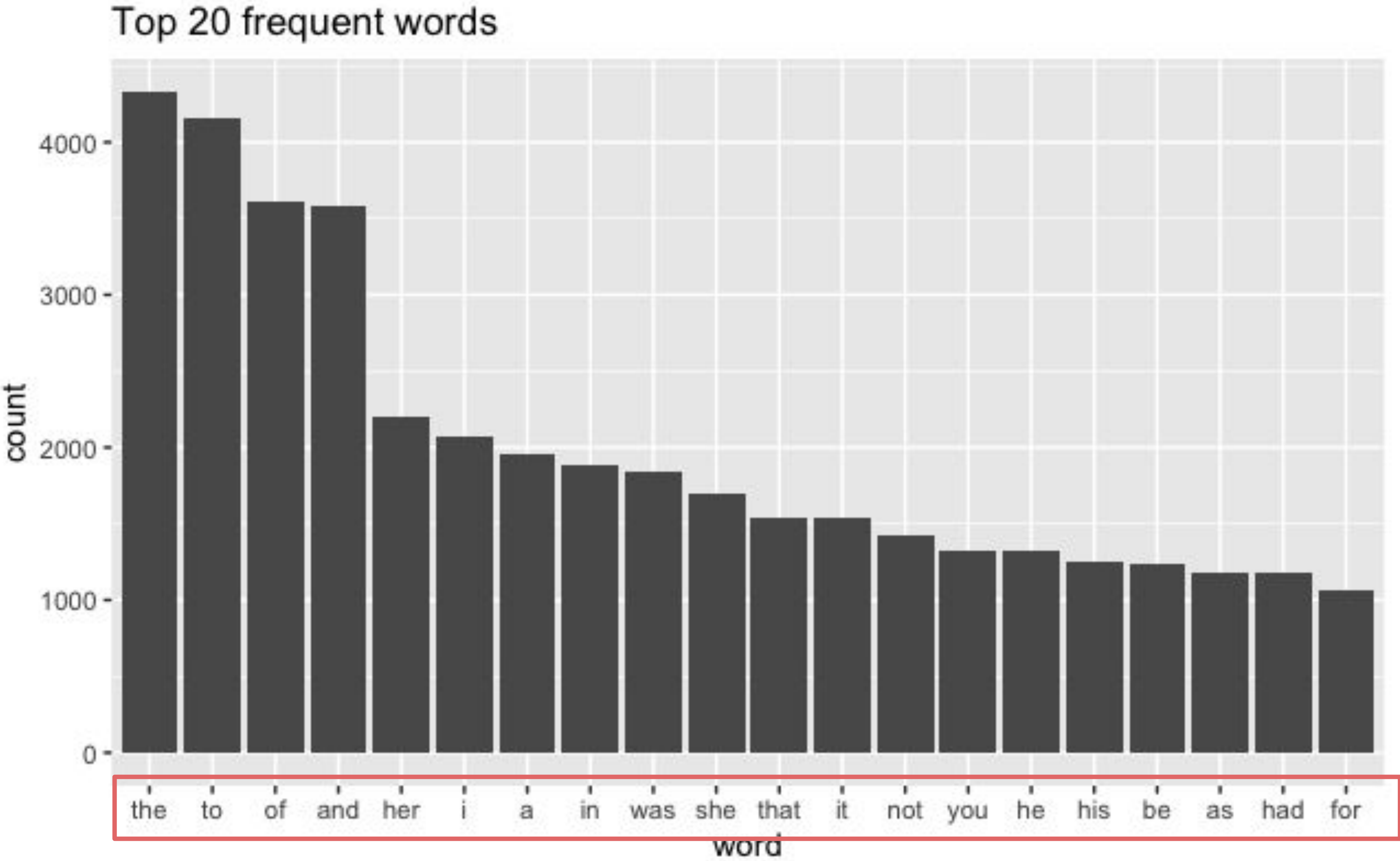
A common task in text mining is to look at word frequencies, which can then be used to compare frequencies across different texts.

```
# counting frequencies
freqs = tidy_pride %>% count(word)
head(freqs, 10)
```

```
            word  n
1     _accident_  1
2   _advantages_  1
3       _affect_  1
4          _all_  4
5           _am_  1
6       _another  1
7          _any_  1
8    _anybody's_  1
9   _appearance_  3
10         _are_  2
```

```r
# top-20 frequent words
top20_freqs = freqs %>%
  arrange(desc(n)) %>%
  slice_head(n = 20)

ggplot(data = top20_freqs,
       aes(x = reorder(word, -n), y = n)) +
  geom_col() +
  labs(title = "Top 20 frequent words") +
  xlab("word") +
  ylab("count")
```


Top 20 frequent words
count
4000
3000
2000
1000
0
the
to
of
and
her
i
a
in
was
she
that
it
not
you
he
his
be
as
had
for
word



*Very common words (not always interesting)*

# Stop Words

# Stop Words (stopwords)

Stop words are a set of commonly used words in a language.

Examples of stop words in English are **“a”**, **“the”**, **“is”**, **“are”**, etc.

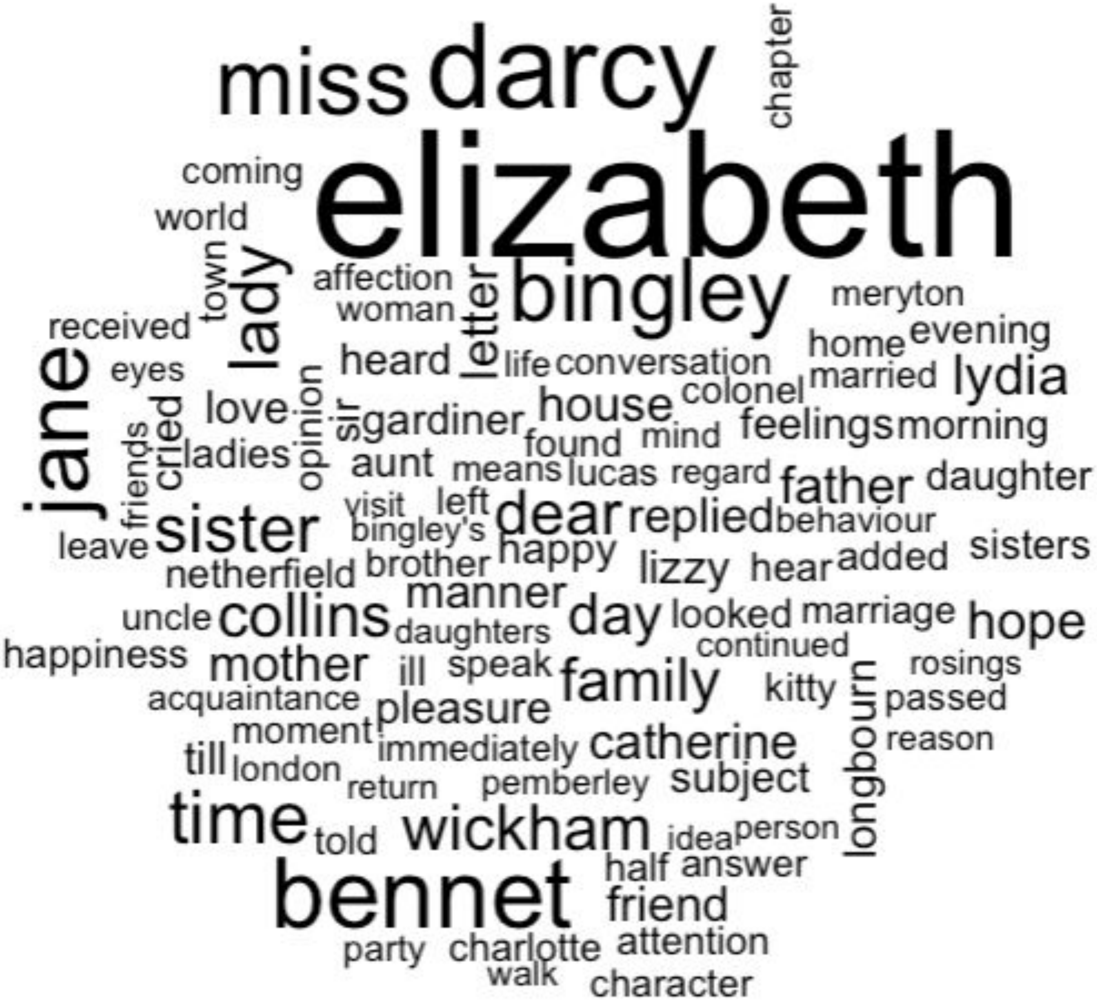
**Stop-words** are commonly used to eliminate words that are so commonly used that they carry very little useful information.

```
# stop_words tibble from tidyverse
head(stop_words, 10);  tail(stop_words, 10)
```

```
# A tibble: 10 x 2
   word        lexicon
   <chr>       <chr>
 1 a           SMART
 2 a's         SMART
 3 able        SMART
 4 about       SMART
 5 above       SMART
 6 according   SMART
 7 accordingly SMART
 8 across      SMART
 9 actually    SMART
10 after       SMART
```

```
# A tibble: 10 x 2
   word     lexicon
   <chr>    <chr>
 1 would    onix
 2 year     onix
 3 years    onix
 4 yet      onix
 5 you      onix
 6 young    onix
 7 younger  onix
 8 youngest onix
 9 your     onix
10 yours    onix
```

```
# removing stopwords and
# graphing wordcloud
tidy_pride = tidy_pride %>%
  anti_join(stop_words) %>%
  count(word) %>%
  with(wordcloud(word, n, max.words = 100))
```


miss
darcy
chapter
coming
elizabeth
world
town
lady
affection
letter
bingley
meryton
woman
received
home
evening
jane
eyes
heard
life
conversation
colonel
married
lydia
love
opinion
sir
gardiner
house
friends
cried
found
mind
feelings
morning
ladies
aunt
means
lucas
regard
father
daughter
visit
left
dear
replied
behaviour
bingley's
sister
leave
happy
sisters
added
lizzy
hear
netherfield
brother
manner
uncle
collins
day
looked
marriage
hope
daughters
continued
happiness
mother
ill
speak
family
rosings
kitty
longbourn
passed
acquaintance
pleasure
moment
reason
catherine
immediately
till
london
subject
return
pemberley
time
wickham
idea
person
told
half
answer
bennet
friend
party
charlotte
attention
walk
character

# N-grams
# e.g. bigrams

# n-grams

We’ve used `unnest_tokens()` to tokenize by word.

We can also use `unnest_tokens()` to tokenize into consecutive sequences of words, called **n-grams**.

By seeing how often word X is followed by word Y, we can then build a model of the relationships between them.

```
# “Pride and Prejudice” by Jane Austen
head(prideprejudice, 11)
```

```
 [1] "PRIDE AND PREJUDICE"
 [2] ""
 [3] "By Jane Austen"
 [4] ""
 [5] ""
 [6] ""
 [7] "Chapter 1"
 [8] ""
 [9] ""
[10] "It is a truth universally acknowledged, that a
single man in possession"
[11] "of a good fortune, must be in want of a wife."
```

```r
# tokenization with bigrams
pride_bigrams <- pride %>%
  unnest_tokens(bigram, text,
    token = "ngrams", n = 2) %>%
  filter(!is.na(bigram))
```

```
                     bigram
1                 pride and
2             and prejudice
3                   by jane
4               jane austen
5                 chapter 1
6                     it is
7                      is a
8                   a truth
9         truth universally
10 universally acknowledged
```

```
# counting and filtering bigrams
count_bigrams <- pride_bigrams %>%
  count(bigram, sort = TRUE)
head(count_bigrams, 10)
```

```
        bigram    n
1       of the  439
2        to be  422
3       in the  365
4         i am  291
5       of her  245
6       it was  235
7       to the  231
8     mr darcy  230
9       of his  219
10     she was  204
```

# bi-grams

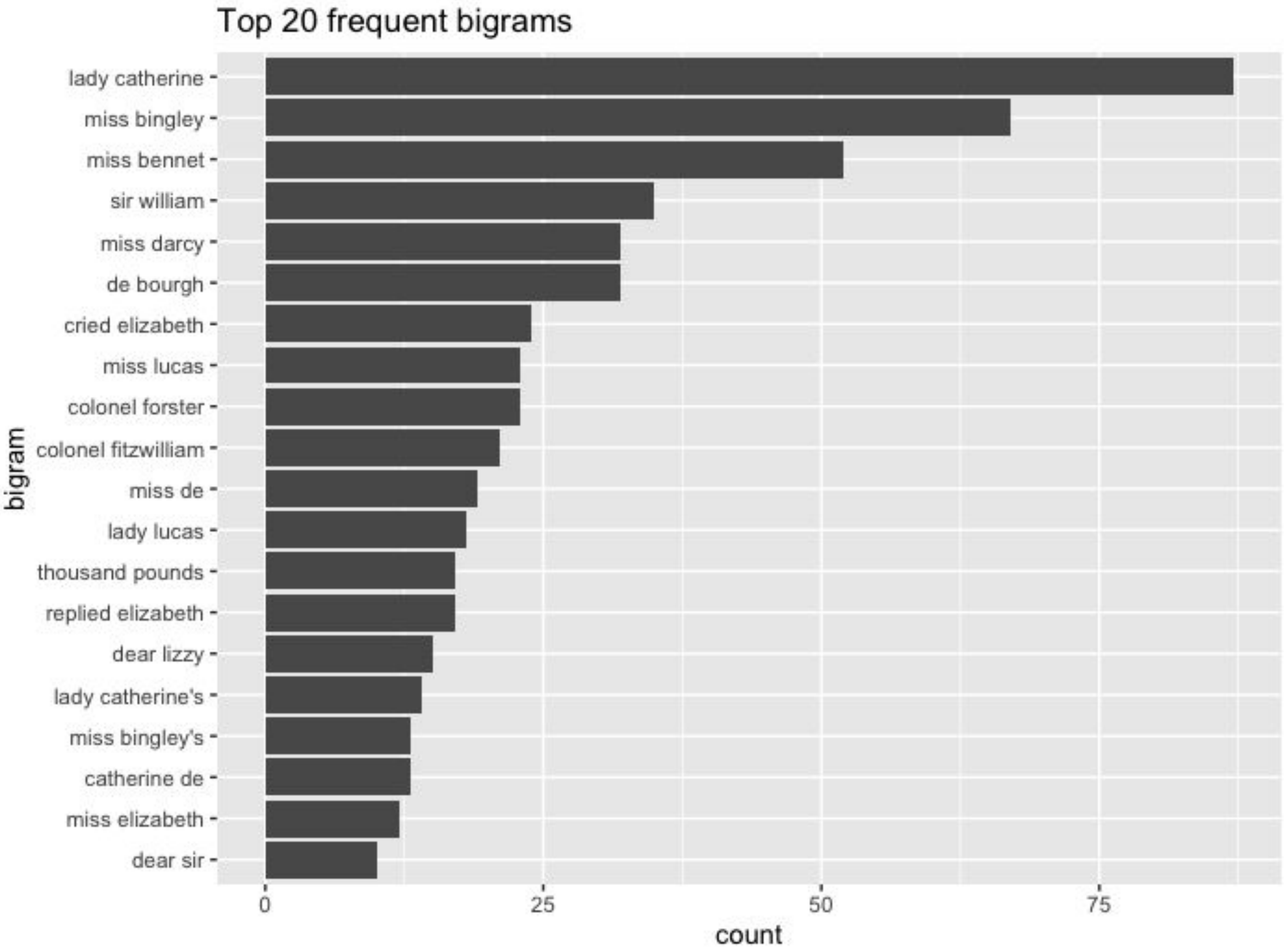
Most common bigrams are pairs of common (uninteresting) words, such as “of the” and “to be”, which can be regarded as stopwords (stop-bigrams)

```
# counting and filtering bigrams
bigrams_separated <- pride_bigrams %>%
  separate(bigram, c("word1", "word2"), sep = " ")

bigrams_filtered <- bigrams_separated %>%
  filter(!word1 %in% stop_words$word) %>%
  filter(!word2 %in% stop_words$word)

count_bigrams <- bigrams_filtered %>%
  count(word1, word2, sort = TRUE)

head(count_bigrams, 10)
```

```
# counting and filtering bigrams
head(count_bigrams, 15)
```

```
      word1       word2  n
1      lady   catherine 87
2      miss     bingley 67
3      miss      bennet 52
4       sir     william 35
5        de      bourgh 32
6      miss       darcy 32
7     cried   elizabeth 24
8   colonel     forster 23
9      miss       lucas 23
10  colonel fitzwilliam 21
11     miss          de 19
12     lady       lucas 18
13  replied   elizabeth 17
14 thousand      pounds 17
15     dear       lizzy 15
```

```
# joining separated words into bigrams
bigrams_united = count_bigrams %>%
  unite(bigram, word1, word2, sep = " ")

ggplot(data = bigrams20) +
  geom_col(aes(x = reorder(bigram, n), y = n)) +
  coord_flip() +
  labs(title = "Top 20 frequent bigrams") +
  xlab("bigram") +
  ylab("count")
```


Top 20 frequent bigrams
lady catherine
miss bingley
miss bennet
sir william
miss darcy
de bourgh
cried elizabeth
miss lucas
colonel forster
colonel fitzwilliam
miss de
lady lucas
thousand pounds
replied elizabeth
dear lizzy
lady catherine's
miss bingley's
catherine de
miss elizabeth
dear sir
bigram
0
25
50
75
count

# Network of n-grams

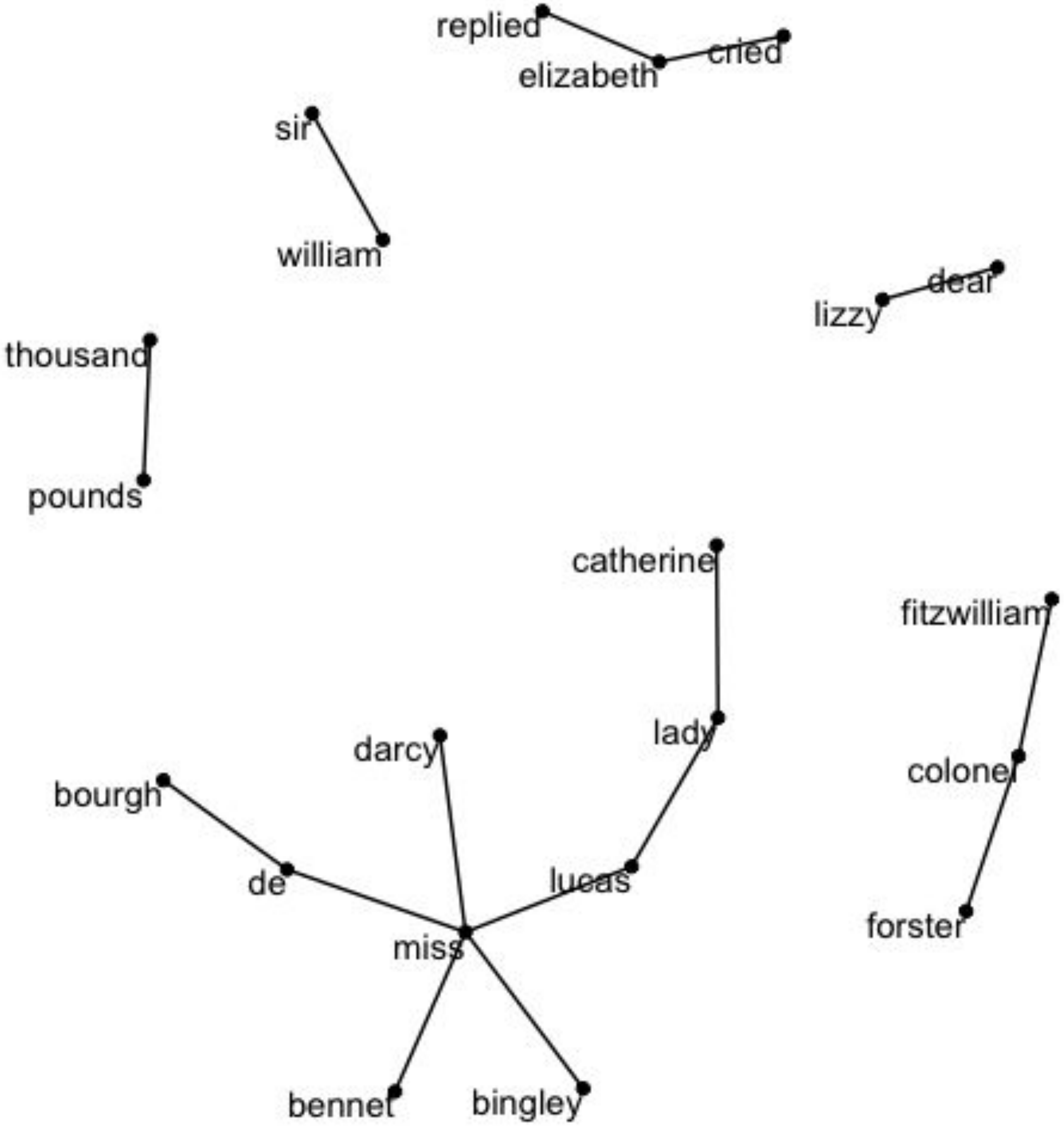
## bi-grams

To visualize the relationships among words simultaneously, rather than just the top few at a time, we can arrange the words into a network or graph.

```
library(igraph)  # make the network

library(ggraph)  # plot network “a la” ggplot
```

```
# joining separated words into bigrams
bigrams_graph <- count_bigrams %>%
  filter(n > 14) %>%
  graph_from_data_frame()

set.seed(1234)
ggraph(bigrams_graph, layout = "fr") +
  geom_edge_link() +
  geom_node_point() +
  geom_node_text(aes(label = name),
                 vjust = 1, hjust = 1) +
  labs(title = "Common bigrams in Pride
                and Prejudice")
```

Common bigrams in Pride and Prejudice


replied
elizabeth
cried
sir
william
dear
lizzy
thousand
pounds
catherine
fitzwilliam
lady
darcy
bourgh
colonel
de
lucas
forster
miss
bennet
bingley