

# Solution slides Part 4

## Introduction to R & Data for Humanities

Afternoon session  
*Text-mining with Tidyverse*

# Exercise 11

11a.

```
## {r}

# Exercise 11a. Can you set the number of words in each n-gram to 2 in the code below? This allows us to examine pairs of two
# consecutive words, called 'bigrams' in Austen's novels

library(dplyr)
library(tidytext)
library(janeaustenr)

austen_bigrams <- austen_books() %>%
  unnest_tokens(bigram, text, token = "ngrams", n = 2)

austen_bigrams
```

Notice that these bigrams overlap: “norland park” is one token, while “park in” is another.

book <fctr>	bigram <chr>
Sense & Sensibility	was at
Sense & Sensibility	at norland
Sense & Sensibility	norland park
Sense & Sensibility	park in
Sense & Sensibility	in the
Sense & Sensibility	the centre
Sense & Sensibility	centre of
Sense & Sensibility	their property
Sense & Sensibility	property where
Sense & Sensibility	where for

31-40 of 675,025 rows

Previous 1 2 3 4 5 6 ... 100 Next

# 11b.

The output here is based on running the piece of code in blue.

```
##{r}

# Exercise 11b. When we count our bigrams using dplyr's count(), we see that a lot of the most common bigrams are pairs of
common words, like stop words. Run this code and you'll see...

austen_bigrams %>%
  count(bigram, sort = TRUE)

# We are of course not only interested in the stop word bigrams. So let's filter our n-grams with tidyr's separate() and remove
cases where either word is a stop word. Run it!

library(tidyr)

bigrams_separated <- austen_bigrams %>%
  separate(bigram, c("word1", "word2"), sep = " ")

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

# new bigram counts:
bigram_counts <- bigrams_filtered %>%
  count(word1, word2, sort = TRUE)

bigram_counts
```

bigram <chr>	n <int>
NA	12242
of the	2853
to be	2670
in the	2221
it was	1691
i am	1485
she had	1405
of her	1363
to the	1315
she was	1309

1-10 of 193,210 rows

Previous 1 2 3 4 5 6 ... 100 Next

# 11b. (resumed)

```
# We are of course not only interested in the stop word bigrams. So let's filter our n-grams with tidyr's separate() and remove cases where either word is a stop word. Run it!

library(tidyr)

bigrams_separated <- austen_bigrams %>%
  separate(bigram, c("word1", "word2"), sep = " ")

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

# new bigram counts:
bigram_counts <- bigrams_filtered %>%
  count(word1, word2, sort = TRUE)

bigram_counts
```

Looks familiar?  
Again with the  
proper nouns,  
that is: names!

word1 <chr>	word2 <chr>	n <int>
NA	NA	12242
sir	thomas	266
miss	crawford	196
captain	wentworth	143
miss	woodhouse	143
frank	churchill	114
lady	russell	110
sir	walter	108
lady	bertram	101
miss	fairfax	98

1-10 of 28,975 rows

Previous 1 2 3 4 5 6 ... 100 Next

# 11c.

# Exercise 11c. We will now use tidyr's unite() function to recombine the columns into one. Using the "separate/filter/count/unite" combination lets us find the most common bigrams not containing stop-words. Run the code below.

```
bigrams_united <- bigrams_filtered %>%  
  unite(bigram, word1, word2, sep = " ")
```

```
bigrams_united
```

book <fctr>	bigram <chr>
Sense & Sensibility	fortune independent
Sense & Sensibility	father's inheriting
Sense & Sensibility	thousand pounds
Sense & Sensibility	remaining moiety
Sense & Sensibility	wife's fortune
Sense & Sensibility	NA NA
Sense & Sensibility	gentleman died
Sense & Sensibility	destroyed half
Sense & Sensibility	son's son
Sense & Sensibility	valuable woods

31-40 of 51,155 rows

Previous 1 2 3 4 5 6 ... 100 Next

# 11d.

# Exercise 11d. we can look at the tf-idf of bigrams across Austen's novels. These tf-idf values can be visualized within each book, just as we did for words. Can you complete the code below and produce a tibble and visualization of your results all at once?

```
bigram_tf_idf <- bigrams_united %>%  
  count(book, bigram) %>%  
  bind_tf_idf(bigram, book, n) %>%  
  arrange(desc(tf_idf))  
|  
bigram_tf_idf
```

book <fctr>	bigram <chr>	n <int>	tf <dbl>	idf <dbl>	tf_idf <dbl>
Mansfield Park	sir thomas	266	0.0244238362	1.7917595	0.0437616398
Persuasion	captain wentworth	143	0.0232142857	1.7917595	0.0415944162
Mansfield Park	miss crawford	196	0.0179965109	1.7917595	0.0322454188
Persuasion	lady russell	110	0.0178571429	1.7917595	0.0319957048
Persuasion	sir walter	108	0.0175324675	1.7917595	0.0314139647
Emma	miss woodhouse	143	0.0128817224	1.7917595	0.0230809480
Northanger Abbey	miss tilney	74	0.0127828641	1.7917595	0.0229038177
Sense & Sensibility	colonel brandon	96	0.0114572145	1.7917595	0.0205285725
Sense & Sensibility	sir john	94	0.0112185225	1.7917595	0.0201008939
Emma	frank churchill	114	0.0102693451	1.7917595	0.0184001963

1-10 of 31,397 rows

Previous **1** 2 3 4 5 6 ... 100 Next

# Exercise 12

12a.

```
```{r}

# Exercise 12a. We want to find out what words tend to appear within each 10-line section of Austen's Pride and Prejudice.
Let's first find the Most Common words, filtering out stop words. Can you complete the code?

austen_section_words <- austen_books() %>%
  filter(book == "Pride & Prejudice") %>%
  mutate(section = row_number() %/% 10) %>%
  filter(section > 0) %>%
  unnest_tokens(word, text) %>%
  filter(!word %in% stop_words$word)

austen_section_words
```
```

| book<br><fctr>    | section<br><dbl> | word<br><chr> |
|-------------------|------------------|---------------|
| Pride & Prejudice | 1                | truth         |
| Pride & Prejudice | 1                | universally   |
| Pride & Prejudice | 1                | acknowledged  |
| Pride & Prejudice | 1                | single        |
| Pride & Prejudice | 1                | possession    |
| Pride & Prejudice | 1                | fortune       |
| Pride & Prejudice | 1                | wife          |
| Pride & Prejudice | 1                | feelings      |
| Pride & Prejudice | 1                | views         |
| Pride & Prejudice | 1                | entering      |

1-10 of 37,240 rows

Previous **1** 2 3 4 5 6 ... 100 Next

# 12b.

```

```{r}

# Exercise 12b. Can you complete the count by using the function mentioned above and providing it with the information on what to count?

library(widyr)

# count words co-occurring within sections
word_pairs <- austen_section_words %>%
  pairwise_count(word, section, sort = TRUE)

word_pairs
```

```

R Console

tbl\_df  
796008 x 3

| item1<br><chr> | item2<br><chr> | n<br><dbl> |
|----------------|----------------|------------|
| darcy          | elizabeth      | 144        |
| elizabeth      | darcy          | 144        |
| miss           | elizabeth      | 110        |
| elizabeth      | miss           | 110        |
| elizabeth      | jane           | 106        |
| jane           | elizabeth      | 106        |
| miss           | darcy          | 92         |
| darcy          | miss           | 92         |
| elizabeth      | bingley        | 91         |
| bingley        | elizabeth      | 91         |

1-10 of 796,008 rows

Previous 1 2 3 4 5 6 ... 100 Next



# 12c.

```
```{r}

# Exercise 12c. The syntax of the pairwise_corr() function is similar to that of pairwise_count(). Just run it!

# we need to filter for at least relatively common words first
word_cors <- austen_section_words %>%
  group_by(word) %>%
  filter(n() >= 20) %>%
  pairwise_cor(word, section, sort = TRUE)

word_cors
```
```

| item1<br><chr> | item2<br><chr> | correlation<br><dbl> |
|----------------|----------------|----------------------|
| bourgh         | de             | 0.9508501            |
| de             | bourgh         | 0.9508501            |
| pounds         | thousand       | 0.7005808            |
| thousand       | pounds         | 0.7005808            |
| william        | sir            | 0.6644719            |
| sir            | william        | 0.6644719            |
| catherine      | lady           | 0.6633048            |
| lady           | catherine      | 0.6633048            |
| forster        | colonel        | 0.6220950            |
| colonel        | forster        | 0.6220950            |

1-10 of 154,842 rows

Previous 1 2 3 4 5 6 ... 100 Next

# 12d.

One of the word correlations that stand out (both in the tibble in exercise 12c and in this visualization, is the one between “marry” and “money”. This could be a semantic relationship to explore in further analyses.

```
```{r}

# Exercise 12d. Let's pick some interesting words and find the other words most associated with them! You can pick your own and
add them to the code below. And do you remember the function we have used a few times to plot your results? Fill it in as well!

word_cors %>%
  filter(item1 %in% c("lady", "colonel", "carriage", "marry")) %>%
  group_by(item1) %>%
  slice_max(correlation, n = 6) %>%
  ungroup() %>%
  mutate(item2 = reorder(item2, correlation)) %>%
  ggplot(aes(item2, correlation)) +
  geom_bar(stat = "identity") +
  facet_wrap(~ item1, scales = "free") +
  coord_flip()
```
```

