

# Health Data Science Practical 2

Jenna Hepburn

June 4, 2023

```
knitr::opts_chunk$set(echo = TRUE)
library(readr)
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(ggplot2)
library(scales)

##
## Attaching package: 'scales'

## The following object is masked from 'package:readr':
##
##     col_factor

library(tidytext)
library(textstem)

## Loading required package: koRpus.lang.en

## Loading required package: koRpus

## Loading required package: syll

## For information on available language packages for 'koRpus', run
##
##     available.koRpus.lang()
##
## and see ?install.koRpus.lang()
```

```
##
```

```
## Attaching package: 'koRpus'
```

```
## The following object is masked from 'package:readr':
```

```
##
```

```
##     tokenize
```

```
library(clinspacy)
```

```
## Welcome to clinspacy.
```

```
## By default, this package will install and use miniconda and create a "clinspacy" conda environment.
```

```
## If you want to override this behavior, use clinspacy_init(miniconda = FALSE) and specify an alternat
```

```
library(topicmodels)
```

```
library('reshape2')
```

```
##
```

```
## Attaching package: 'reshape2'
```

```
## The following object is masked from 'package:tidyর':
```

```
##
```

```
##     smths
```

```
library(stringr)
```

This practical is based on exploratory data analysis, named entity recognition, and topic modelling of unstructured medical note free-text data derived from electronic medical records (EMR). Real EMR data is very difficult to access without a specific need/request so this data set is derived from medical transcription data instead. I'll also caveat that the options of natural language processing (NLP) in R are far inferior to those available in Python.

First, install the packages in the setup block (`install.packages(c("readr", "dplyr", "tidyর", "ggplot2", "tidetext", "textstem", "clinspacy", "topicmodels", "reshape2"))`).

Note: To try and make it clearer which library certain functions are coming from clearer, I'll try to do explicit imports throughout this notebook.

## Data Parsing

After that we can grab the dataset directly from the `clinspacy` library.

```
raw.data <- clinspacy::dataset_mtsamples()  
dplyr::glimpse(raw.data)
```

```
## Rows: 4,999
```

```
## Columns: 6
```

```
## $ note_id      <int> 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 1~  
## $ description  <chr> "A 23-year-old white female presents with complaint ~  
## $ medical_specialty <chr> "Allergy / Immunology", "Bariatrics", "Bariatrics", ~  
## $ sample_name   <chr> "Allergic Rhinitis", "Laparoscopic Gastric Bypass Co~  
## $ transcription <chr> "SUBJECTIVE:, This 23-year-old white female present~  
## $ keywords      <chr> "allergy / immunology, allergic rhinitis, allergies,~
```

There is no explanation or data dictionary with this dataset, which is a surprisingly common and frustrating turn of events!

1 Using the output of dplyr's `glimpse` command (or rstudio's data viewer by clicking on `raw.data` in the Environment pane) provide a description of what you think each variable in this dataset contains.

`note_id` - unique ID number for each unique note

`description` - note containing a description of the medical problem the patient is experiencing

`medical_specialty` - the medical specialty that the patient's medical issue falls under (ex. heart attack would be cardiovascular/pulmonary)

`sample_name` - name of diagnosis or procedure the patient was seeking medical attention for

`transcription` - details on the patient and their condition such as medication, patient history, family history, presenting conditions, etc.

`keywords` - terms to explain the patient's condition including synonyms for their symptoms, diagnosis, or procedure.

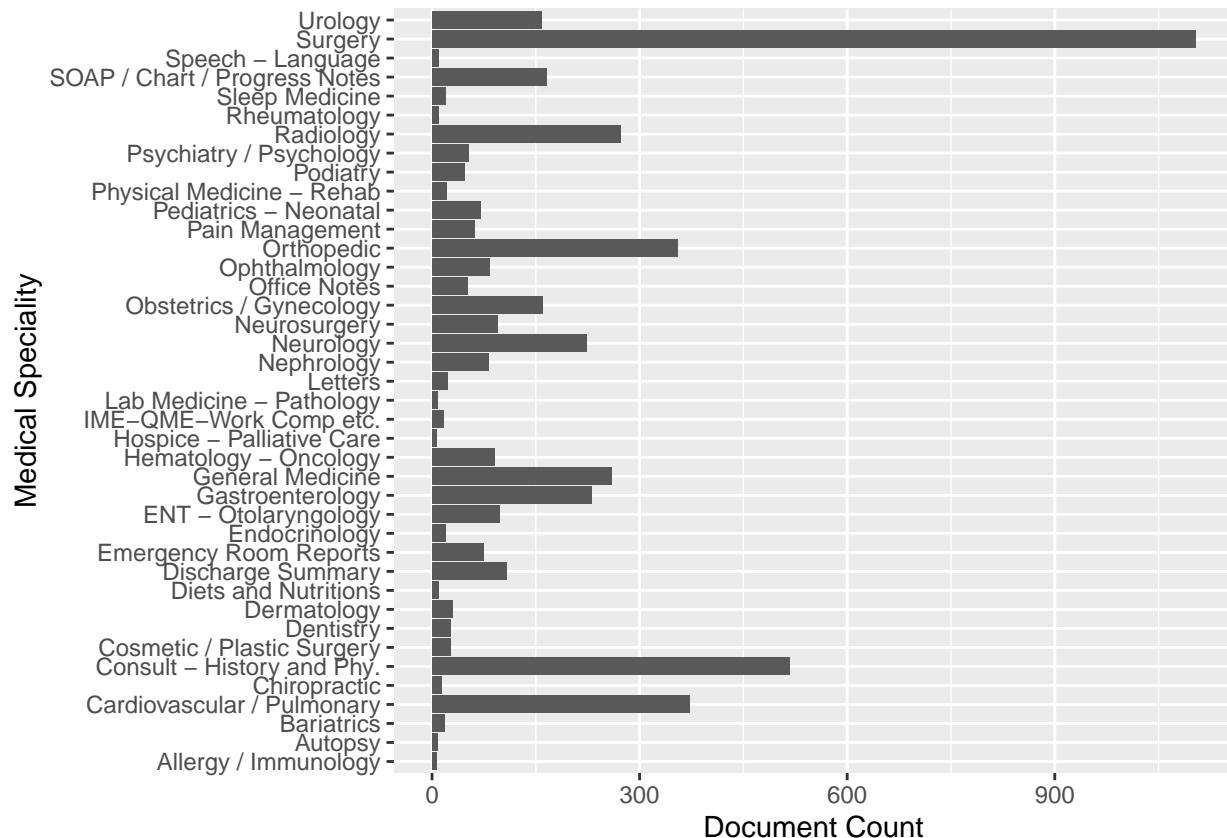
Let's see how many different medical specialties are featured in these notes:

```
raw.data %>% dplyr::select(medical_specialty) %>% dplyr::n_distinct()
```

```
## [1] 40
```

So, how many transcripts are there from each specialty:

```
ggplot2::ggplot(raw.data, ggplot2::aes(y=medical_specialty)) + ggplot2::geom_bar() + labs(x="Document C
```



Let's make our life easier and filter down to 3 specialties: a diagnostic/lab, a medical, and a surgical specialty

```
filtered.data <- raw.data %>% dplyr::filter(medical_specialty %in% c("Orthopedic", "Radiology", "Surgery"))
```

## Text Processing

Let's now apply our standard pre-processing to the transcripts from these specialties.

We are going to use the `tidytext` package to tokenise the transcript free-text.

Let's remove stop words first. e.g., "the", "of", "to", and so forth. These are known as stop words and we can remove them relative easily using a list from `tidytext::stop_words` and `dplyr::anti_join()`

```
analysis.data <- filtered.data %>%
  unnest_tokens(word, transcription) %>%
  mutate(word = str_replace_all(word, "[^[:alnum:]]", "")) %>%
  filter(!str_detect(word, "[0-9]")) %>%
  anti_join(stop_words) %>%
  group_by(note_id) %>%
  summarise(transcription = paste(word, collapse = " ")) %>%
  left_join(select(filtered.data, -transcription), by = "note_id")
```

```
## Joining with 'by = join_by(word)'
```

Now let's tokenize the `transcription` to words (unigram). By default this tokenises to words but other options include characters, n-grams, sentences, lines, paragraphs, or separation around a regular expression.

```
tokenized.data.unigram <- analysis.data %>% tidytext::unnest_tokens(word, transcription, to_lower=TRUE)
```

You can also do bi-grams

```
tokenized.data <- analysis.data %>% tidytext::unnest_tokens(ngram, transcription, token = "ngrams", n=2)
```

**2** How many unique tokens are there in the transcripts from each specialty:

```
tokenized.data.unigram %>% dplyr::group_by(medical_specialty) %>% dplyr::distinct(word) %>% dplyr::summa
```

```
## # A tibble: 3 x 2
##   medical_specialty     n
##   <chr>                 <int>
## 1 Orthopedic            7682
## 2 Radiology             5935
## 3 Surgery               11977
```

Orthopedic - 7,682 Radiology - 5,935 Surgery - 11,977

For bigram:

```
tokenized.data %>% dplyr::group_by(medical_specialty) %>% dplyr::distinct(ngram) %>% dplyr::summarise(n=
```

```

## # A tibble: 3 x 2
##   medical_specialty     n
##   <chr>                 <int>
## 1 Orthopedic            55732
## 2 Radiology             28297
## 3 Surgery               130404

```

Orthopedic - 55,732 Radiology - 28,297 Surgery - 130,404

Let's plot some distribution of unigram tokens (words)

```

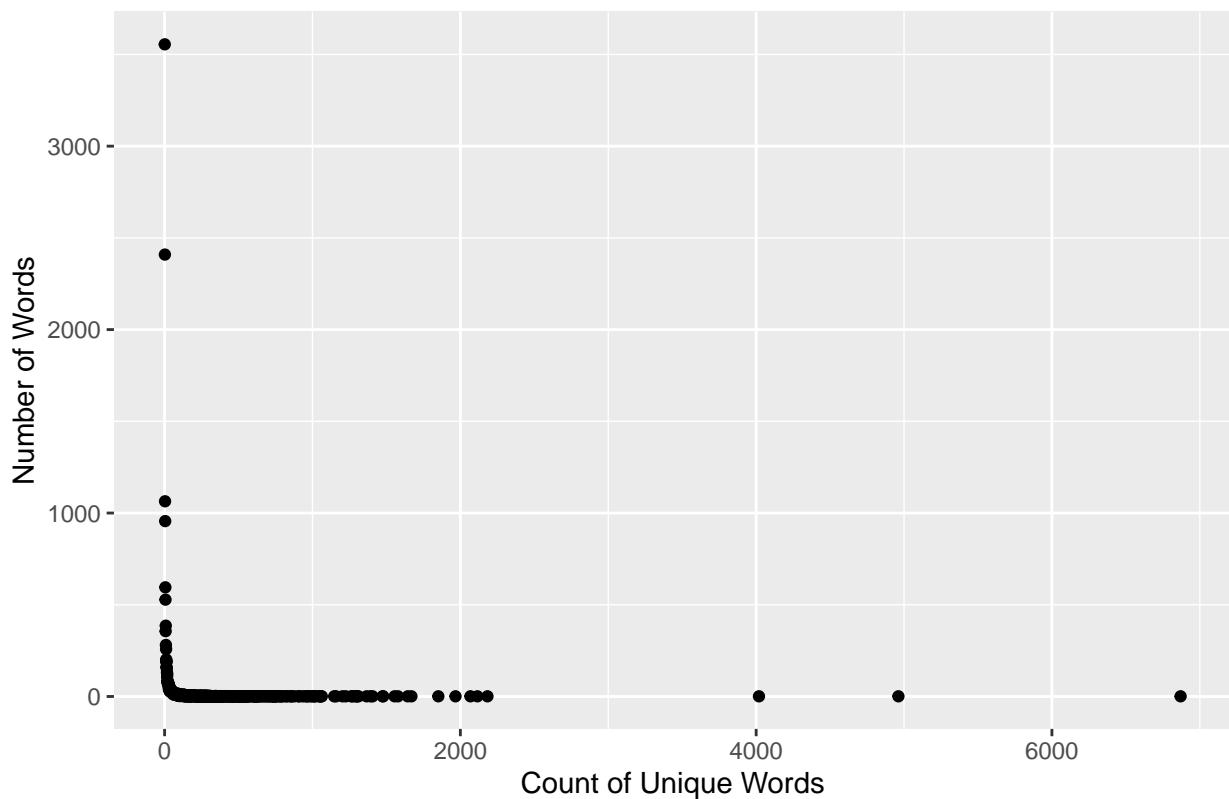
word_counts <- tokenized.data.unigram %>%
  group_by(word) %>%
  summarise(count = n()) %>%
  ungroup() %>%
  arrange(desc(count))

count_distribution <- word_counts %>%
  group_by(count) %>%
  summarise(num_words = n()) %>%
  ungroup()

ggplot2::ggplot(count_distribution, aes(x = count, y = num_words)) +
  geom_point() +
  labs(title = "Scatter Plot of Count Distribution",
       x = "Count of Unique Words",
       y = "Number of Words")

```

Scatter Plot of Count Distribution

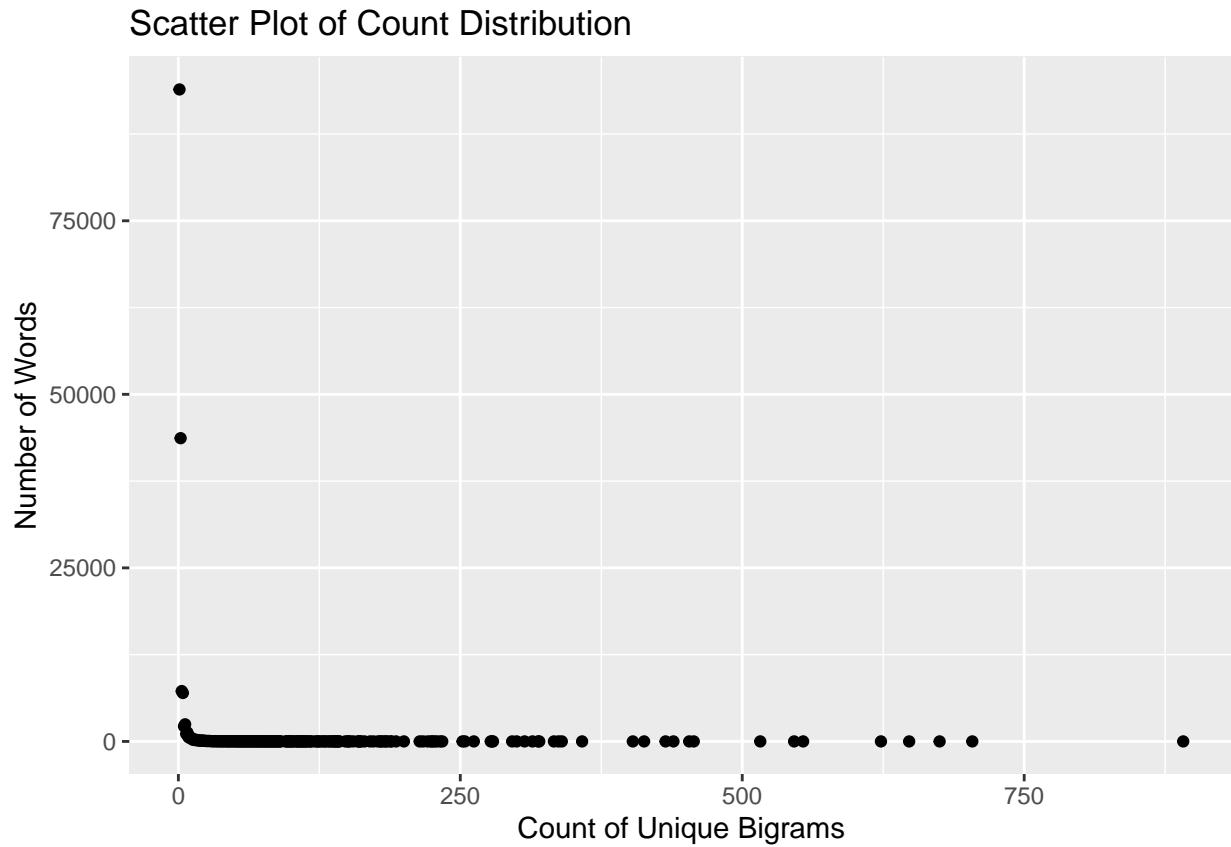


Let's plot some distribution of bigram tokens (words)

```
word_counts <- tokenized.data %>%
  group_by(ngram) %>%
  summarise(count = n()) %>%
  ungroup() %>%
  arrange(desc(count))

count_distribution <- word_counts %>%
  group_by(count) %>%
  summarise(num_words = n()) %>%
  ungroup()

ggplot2::ggplot(count_distribution, aes(x = count, y = num_words)) +
  geom_point() +
  labs(title = "Scatter Plot of Count Distribution",
       x = "Count of Unique Bigrams",
       y = "Number of Words")
```



3 How many unique bigrams are there in each category without stop words and numbers? \*Fin confirmed we should use bigrams instead of words

```
tokenized.data %>% dplyr::group_by(medical_specialty) %>% dplyr::distinct(ngram) %>% dplyr::summarise(n=
```

```
## # A tibble: 3 x 2
```

```

##   medical_specialty      n
##   <chr>              <int>
## 1 Orthopedic          55732
## 2 Radiology           28297
## 3 Surgery             130404

```

Orthopedic - 55,732 Radiology - 28,297 Surgery - 130,404

Sometimes we are interested in tokenising/segmenting things other than words like whole sentences or paragraphs.

4 How many unique sentences are there in each category? Hint: use `?tidytext::unnest_tokens` to see the documentation for this function.

```

sentence.data <- analysis.data %>% tidytext::unnest_tokens(sentences, transcription, token = "sentences")

sentence.data %>% dplyr::group_by(medical_specialty) %>% dplyr::distinct(sentences) %>% dplyr::summarise

## # A tibble: 3 x 2
##   medical_specialty      n
##   <chr>              <int>
## 1 Orthopedic          354
## 2 Radiology           273
## 3 Surgery             1087

```

Orthopedic - 354 Radiology - 273 Surgery - 1,087

Now that we've tokenized to words and removed stop words, we can find the most commonly word used within each category:

```

tokenized.data %>%
  dplyr::group_by(medical_specialty) %>%
  dplyr::count(ngram, sort = TRUE) %>%
  dplyr::top_n(5)

## Selecting by n

## # A tibble: 16 x 3
## # Groups:   medical_specialty [3]
##   medical_specialty ngram                  n
##   <chr>            <chr>              <int>
## 1 Surgery          prepped draped     696
## 2 Surgery          preoperative diagnosis 555
## 3 Surgery          procedure patient    551
## 4 Surgery          postoperative diagnosis 518
## 5 Surgery          tolerated procedure  515
## 6 Orthopedic       prepped draped     183
## 7 Orthopedic       preoperative diagnosis 141
## 8 Orthopedic       lower extremity    139
## 9 Orthopedic       range motion      139
## 10 Orthopedic      postoperative diagnosis 124
## 11 Radiology       carotid artery     59
## 12 Radiology       heart rate        52

```

```

## 13 Radiology      reason exam      51
## 14 Radiology      left ventricular 50
## 15 Radiology      coronary artery   43
## 16 Radiology      exam unremarkable 43

```

We should lemmatize the tokenized words to prevent over counting of similar words before further analyses. Annoyingly, `tidytext` doesn't have a built-in lemmatizer.

5 Do you think a general purpose lemmatizer will work well for medical data? Why not?

No, I do not think the general purpose of a lemmatizer will work well for medical data. This is because words can mean very different things in medicine, but sound the same. For example, hypertension and hypotension sound very similar, but are complete opposites in reality, making the distinction very important. Also, the context of what part of the body something is happening in is very important. For example, a blood clot in the brain is a stroke, and a blood clot in the heart is a heart attack. While these are both blood clots, they have completely different effects and outcomes, and require different specialists. Therefore the general purpose of lematization may not work well for medical data.

Unfortunately, a specialised lemmatizer like in `clinspacy` is going to be very painful to install so we will just use a simple lemmatizer for now:

```
lemmatized.data <- tokenized.data %>% dplyr::mutate(lemma=textstem::lemmatize_words(ngram))
```

We can now calculate the frequency of lemmas within each specialty and note.

```

lemma.freq <- lemmatized.data %>%
  dplyr::count(medical_specialty, lemma) %>%
  dplyr::group_by(medical_specialty) %>%
  dplyr::mutate(proportion = n / sum(n)) %>%
  tidyr::pivot_wider(names_from = medical_specialty, values_from = proportion) %>%
  tidyr::pivot_longer(`Surgery`:`Radiology`,
    names_to = "medical_specialty", values_to = "proportion")

```

And plot the relative proportions

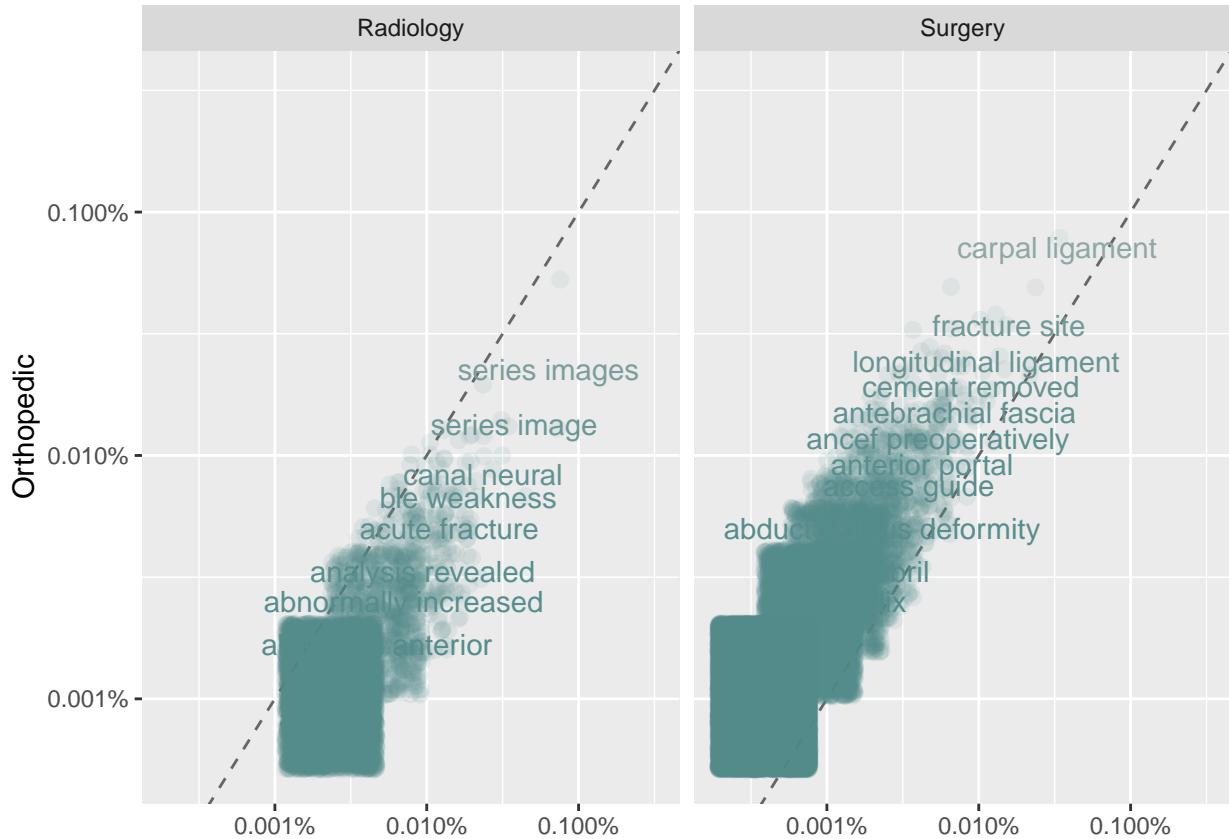
```

ggplot2::ggplot(lemma.freq, ggplot2::aes(x=proportion,
                                             y="Orthopedic",
                                             color=abs(`Orthopedic` - proportion))) +
  ggplot2::geom_abline(color="gray40", lty=2) +
  ggplot2::geom_jitter(alpha=0.1, size=2.5, width=0.3, height=0.3) +
  ggplot2::geom_text(ggplot2::aes(label=lemma), check_overlap=TRUE, vjust=1.5) +
  ggplot2::scale_x_log10(labels=scales::percent_format()) +
  ggplot2::scale_y_log10(labels=scales::percent_format()) +
  ggplot2::scale_color_gradient(limits=c(0, 0.001), low="darkslategray4", high="gray75") +
  ggplot2::facet_wrap(~medical_specialty, ncol = 2) +
  ggplot2::theme(legend.position="none") +
  ggplot2:: labs(y="Orthopedic", x = NULL)

## Warning: Removed 314404 rows containing missing values ('geom_point()').

## Warning: Removed 314404 rows containing missing values ('geom_text()').

```



**6** What does this plot tell you about the relative similarity of lemma frequencies between Surgery and Orthopedic and between radiology and Surgery? Based on what these specialties involve, is this what you would expect?

Based on the plot, the similarity between surgery and orthopedic terms is low for the most part. However, there are some higher lemma frequencies but they are more infrequent than the lower frequencies. Overall, the values for surgery majorly fall above our predictive line, meaning that they have higher proportionality than expected.

Comparing surgery and radiology, many of the points for surgery are above the line (higher than expected), while many of the points for radiology are below the line (lower than expected). This means that we have higher proportionality between surgery and orthopedics than radiology and orthopedics. This is what I would expect as orthopedics is a type of surgery, so I would expect quite a bit of overlap.

**7** Modify the above plotting code to do a direct comparison of Surgery and Radiology (i.e., have Surgery or Radiology on the Y-axis and the other 2 specialties as the X facets)

```
lemma2.freq <- lemmatized.data %>%
  dplyr::count(medical_specialty, lemma) %>%
  dplyr::group_by(medical_specialty) %>%
  dplyr::mutate(proportion = n / sum(n)) %>%
  tidyr::pivot_wider(names_from = medical_specialty, values_from = proportion) %>%
  tidyr::pivot_longer(`Orthopedic`:`Radiology`,
    names_to = "medical_specialty", values_to = "proportion")

ggplot2::ggplot(lemma2.freq, ggplot2::aes(x=proportion,
  y=`Surgery`,
  color=abs(`Surgery` - proportion))) +
```

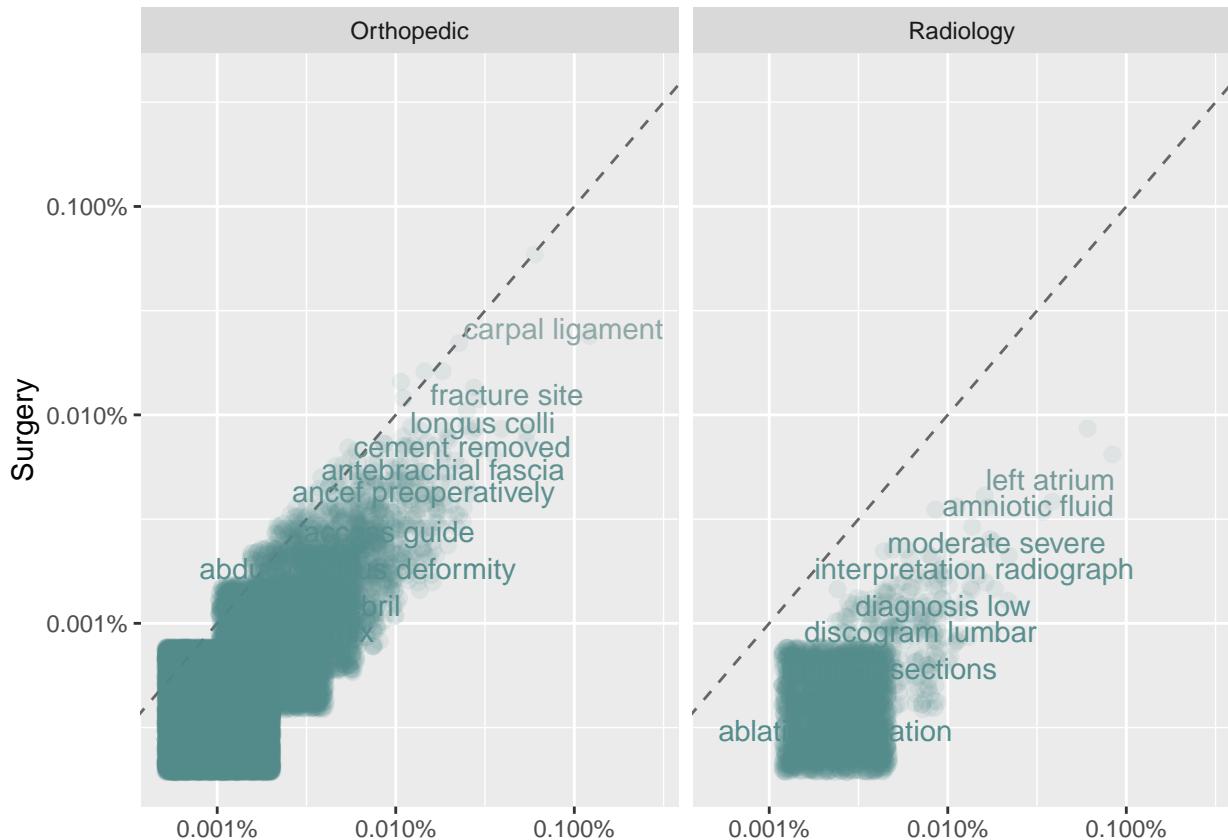
```

ggplot2::geom_abline(color="gray40", lty=2) +
ggplot2::geom_jitter(alpha=0.1, size=2.5, width=0.3, height=0.3) +
ggplot2::geom_text(ggplot2::aes(label=lemma), check_overlap=TRUE, vjust=1.5) +
ggplot2::scale_x_log10(labels=scales::percent_format()) +
ggplot2::scale_y_log10(labels=scales::percent_format()) +
ggplot2::scale_color_gradient(limits=c(0, 0.001), low="darkslategray4", high="gray75") +
ggplot2::facet_wrap(~medical_specialty, ncol = 2) +
ggplot2::theme(legend.position="none") +
ggplot2:: labs(y="Surgery", x = NULL)

```

## Warning: Removed 317819 rows containing missing values ('geom\_point()').

## Warning: Removed 317820 rows containing missing values ('geom\_text()').



### TF-IDF Normalisation

Maybe looking at lemmas across all notes in a specialty is misleading, what if we look at lemma frequencies across a specialty.

```

lemma.counts <- lemmatized.data %>% dplyr::count(medical_specialty, lemma)
total.counts <- lemma.counts %>%
  dplyr::group_by(medical_specialty) %>%
  dplyr::summarise(total=sum(n))

```

```
all.counts <- dplyr::left_join(lemma.counts, total.counts)
```

```
## Joining with 'by = join_by(medical_specialty)'
```

Now we can calculate the term frequency / invariant document frequency (tf-idf):

```
all.counts.tfidf <- tidytext::bind_tf_idf(all.counts, lemma, medical_specialty, n)
```

We can then look at the top 10 lemma by tf-idf within each specialty:

```
all.counts.tfidf %>% dplyr::group_by(medical_specialty) %>% dplyr::slice_max(order_by=tf_idf, n=10)
```

```
## # A tibble: 30 x 7
## # Groups:   medical_specialty [3]
##   medical_specialty lemma      n total      tf     idf    tf_idf
##   <chr>          <chr> <int> <int> <dbl> <dbl> <dbl>
## 1 Orthopedic    range motion 139 97774 0.00142 0.405 0.000576
## 2 Orthopedic    carpal ligament 85 97774 0.000869 0.405 0.000352
## 3 Orthopedic    transverse carpal 81 97774 0.000828 0.405 0.000336
## 4 Orthopedic    extremity prepped 79 97774 0.000808 0.405 0.000328
## 5 Orthopedic    proximal phalanx 75 97774 0.000767 0.405 0.000311
## 6 Orthopedic    department anesthesia 63 97774 0.000644 0.405 0.000261
## 7 Orthopedic    dissection carried 63 97774 0.000644 0.405 0.000261
## 8 Orthopedic    steri strips 59 97774 0.000603 0.405 0.000245
## 9 Orthopedic    closed vicryl 58 97774 0.000593 0.405 0.000241
## 10 Orthopedic   dressing applied 58 97774 0.000593 0.405 0.000241
## # i 20 more rows
```

8 Are there any lemmas that stand out in these lists? Why?

“range motion” and “left ventricular” lemmas have the highest term frequency. We also see for surgery that there are 2 lemmas that mean the same thing, “anesthesia endotracheal” and “endotracheal anesthesia”. If they had been grouped together, this may have resulted in higher (and more accurate) values. Finally, “steri strips” shows up in both orthopedic and surgery specialties.

We can look at transcriptions using these unusual lemmas to check how they are used with `stringr::str_detect`

```
analysis.data %>% dplyr::select(medical_specialty, transcription) %>% dplyr::filter(stringr::str_detect
```

```
## # A tibble: 1 x 2
##   medical_specialty transcription
##   <chr>          <chr>
## 1 Surgery        preoperative diagnoses thrombosed left forearm loop fistula~
```

9 Extract an example of one of the other unusual “top lemmas” by modifying the above code

```
analysis.data %>% dplyr::select(medical_specialty, transcription) %>% dplyr::filter(stringr::str_detect
```

```
## # A tibble: 1 x 2
##   medical_specialty transcription
##   <chr>          <chr>
## 1 Surgery        preoperative diagnosis hallux abductovalgus deformity posto~
```

## Topic Modelling

In NLP, we often have collections of documents (in our case EMR transcriptions) that we'd like to divide into groups so that we can understand them separately. Topic modeling is a method for unsupervised classification of such documents, similar to clustering on numeric data.

Latent Dirichlet allocation (LDA) is a particularly popular method for fitting a topic model. It treats each document as a mixture of topics, and each topic as a mixture of words. This allows documents to "overlap" each other in terms of content, rather than being separated into discrete groups, in a way that mirrors typical use of natural language.

- Every document is a mixture of topics. We imagine that each document may contain words from several topics in particular proportions. For example, in a two-topic model we could say "Document 1 is 90% topic A and 10% topic B, while Document 2 is 30% topic A and 70% topic B."
- Every topic is a mixture of words. For example, we could imagine a two-topic model of American news, with one topic for "politics" and one for "entertainment." The most common words in the politics topic might be "President", "Congress", and "government", while the entertainment topic may be made up of words such as "movies", "television", and "actor". Importantly, words can be shared between topics; a word like "budget" might appear in both equally.

LDA is a mathematical method for estimating both of these at the same time: finding the mixture of words that is associated with each topic, while also determining the mixture of topics that describes each document. There are a number of existing implementations of this algorithm, and we'll explore one of them in depth.

First lets calculate a term frequency matrix for each transcription:

```
lemma.counts <- lemmatized.data %>% dplyr::count(note_id, lemma)
total.counts <- lemma.counts %>%
  dplyr::group_by(note_id) %>%
  dplyr::summarise(total=sum(n))

all.counts <- dplyr::left_join(lemma.counts, total.counts)

## Joining with `by = join_by(note_id)`

emr.dcm <- all.counts %>% tidytext::cast_dtm(note_id, lemma, n)
```

Then we can use LDA function to fit a 5 topic ( $k=5$ ) LDA-model

```
emr.lda <- topicmodels::LDA(emr.dcm, k=5, control=list(seed=42))
emr.topics <- tidytext::tidy(emr.lda, matrix='beta')
```

Then we can extract the top terms per assigned topic:

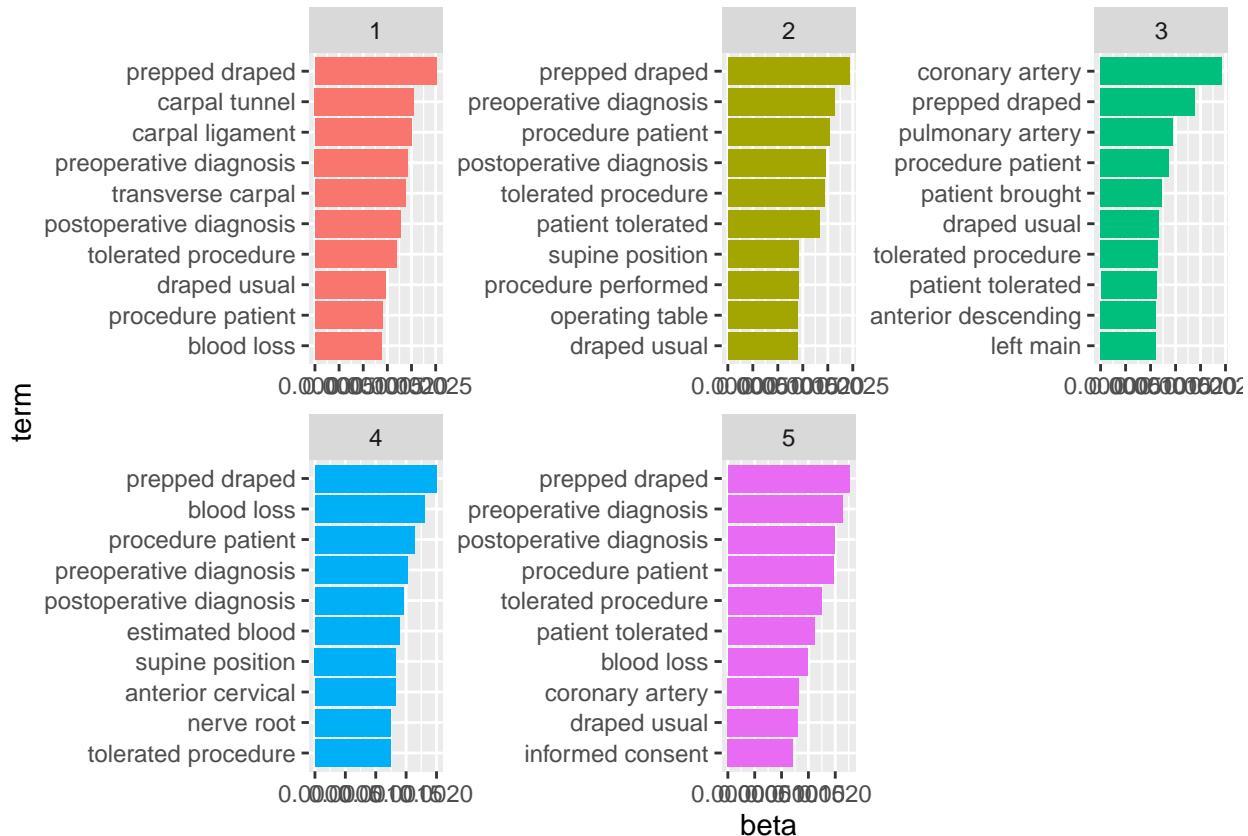
```
top.terms <- emr.topics %>% dplyr::group_by(topic) %>%
  dplyr::slice_max(beta, n=10) %>%
  dplyr::ungroup() %>%
  dplyr::arrange(topic, -beta)

top.terms %>%
  dplyr::mutate(term=tidytext::reorder_within(term, beta, topic)) %>%
```

```

ggplot2::ggplot(ggplot2::aes(beta, term, fill=factor(topic))) +
  ggplot2::geom_col(show.legend=FALSE) +
  ggplot2::facet_wrap(~ topic, scales='free') +
  tidytext::scale_y_reordered()

```



Now we can ask how well do these assigned topics match up to the medical specialties from which each of these transcripts was derived.

```

specialty_gamma <- tidytext::tidy(emr.lda, matrix='gamma')

# we need to join in the specialty from the note_id
note_id_specialty_mapping <- lemmatized.data %>%
  dplyr::mutate(document=as.character(note_id)) %>%
  dplyr::select(document, medical_specialty) %>%
  dplyr::distinct()

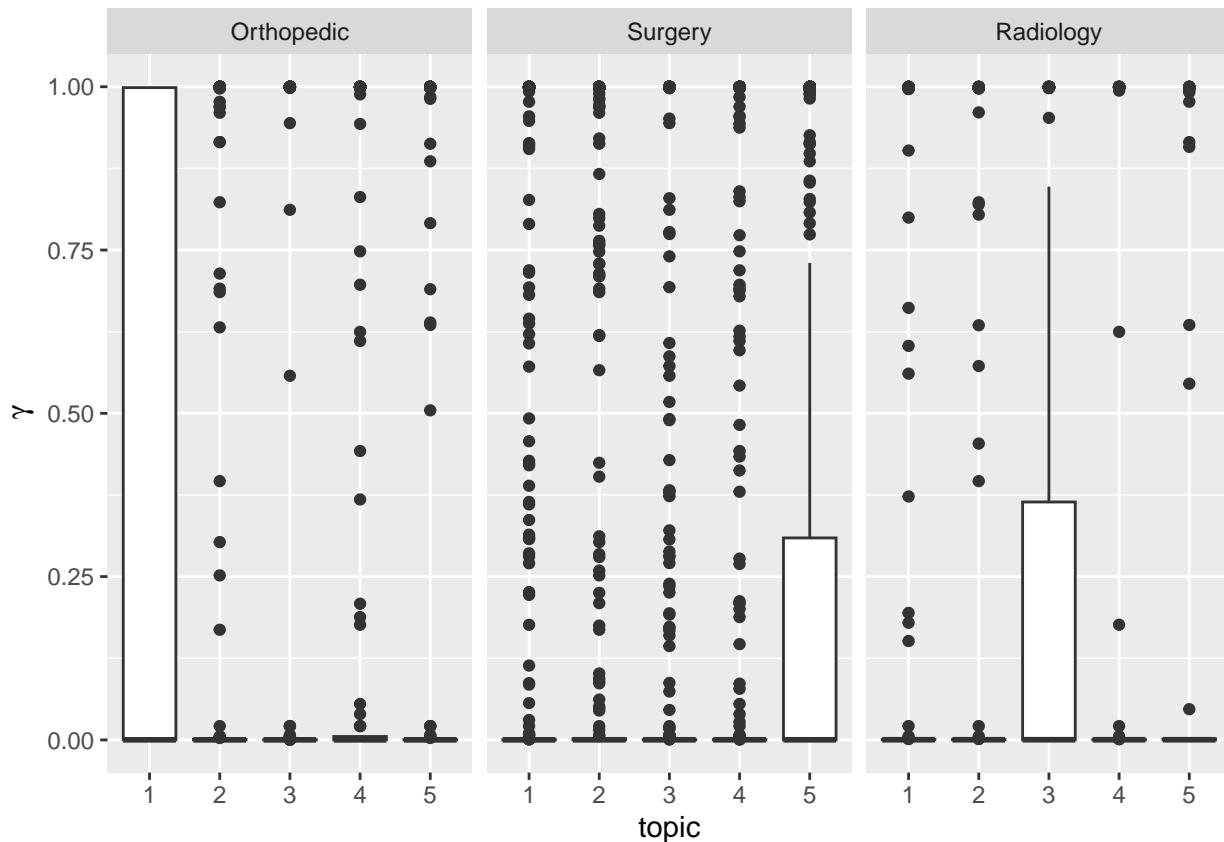
specialty_gamma <- dplyr::left_join(specialty_gamma, note_id_specialty_mapping)

## Joining with 'by = join_by(document)'

specialty_gamma %>%
  dplyr::mutate(medical_specialty = reorder(medical_specialty, gamma * topic)) %>%
  ggplot2::ggplot(ggplot2::aes(factor(topic), gamma)) +
  ggplot2::geom_boxplot()

```

```
ggplot2::facet_wrap(~ medical_specialty) +
  ggplot2::labs(x = "topic", y = expression(gamma))
```



Interestingly, Surgery assigns mostly to a single topic but radiology and Orthopedic are both more diverse in transcriptions. We'd possibly expect this from radiology due to referring to imaging for many different diagnoses/reasons. However, this may all just reflect we are using too few topics in our LDA to capture the range of possible assignments.

**10** Repeat this with a 6 topic LDA, do the top terms from the 5 topic LDA still turn up? How do the specialties get split into sub-topics?

```
lemma.counts2 <- lemmatized.data %>% dplyr::count(note_id, lemma)
total.counts2 <- lemma.counts %>%
  dplyr::group_by(note_id) %>%
  dplyr::summarise(total=sum(n))

all.counts <- dplyr::left_join(lemma.counts2, total.counts2)
```

```
## Joining with 'by = join_by(note_id)'
```

```
emr.dcm <- all.counts %>% tidytext::cast_dtm(note_id, lemma, n)

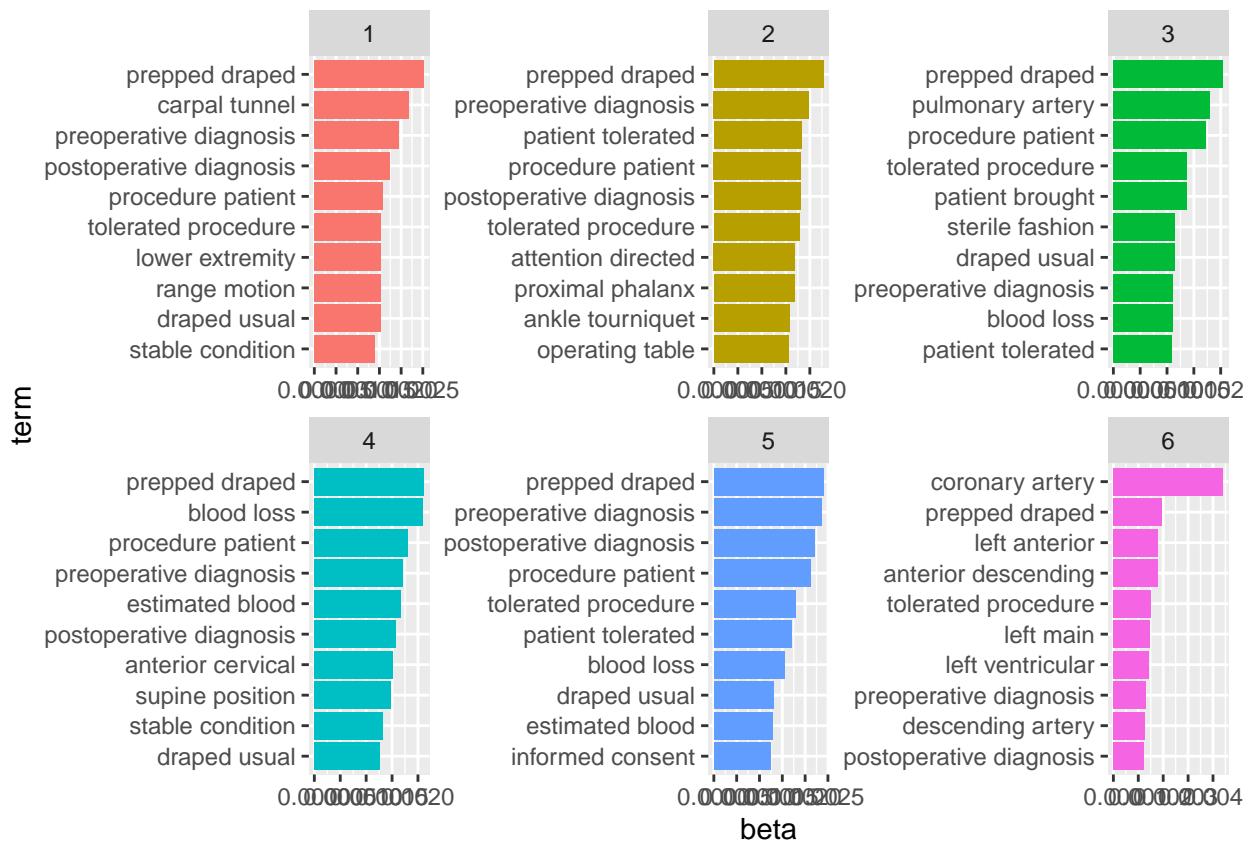
emr.lda <- topicmodels::LDA(emr.dcm, k=6, control=list(seed=42))
emr.topics <- tidytext::tidy(emr.lda, matrix='beta')
```

```

top.terms <- emr.topics %>% dplyr::group_by(topic) %>%
  dplyr::slice_max(beta, n=10) %>%
  dplyr::ungroup() %>%
  dplyr::arrange(topic, -beta)

top.terms %>%
  dplyr::mutate(term=tidytext::reorder_within(term, beta, topic)) %>%
  ggplot2::ggplot(ggplot2::aes(beta, term, fill=factor(topic))) +
  ggplot2::geom_col(show.legend=FALSE) +
  ggplot2::facet_wrap(~ topic, scales='free') +
  tidytext::scale_y_reordered()

```



```

specialty_gamma <- tidytext::tidy(emr.lda, matrix='gamma')

# we need to join in the specialty from the note_id
note_id_specialty_mapping <- lemmatized.data %>%
  dplyr::mutate(document=as.character(note_id)) %>%
  dplyr::select(document, medical_specialty) %>%
  dplyr::distinct()

specialty_gamma <- dplyr::left_join(specialty_gamma, note_id_specialty_mapping)

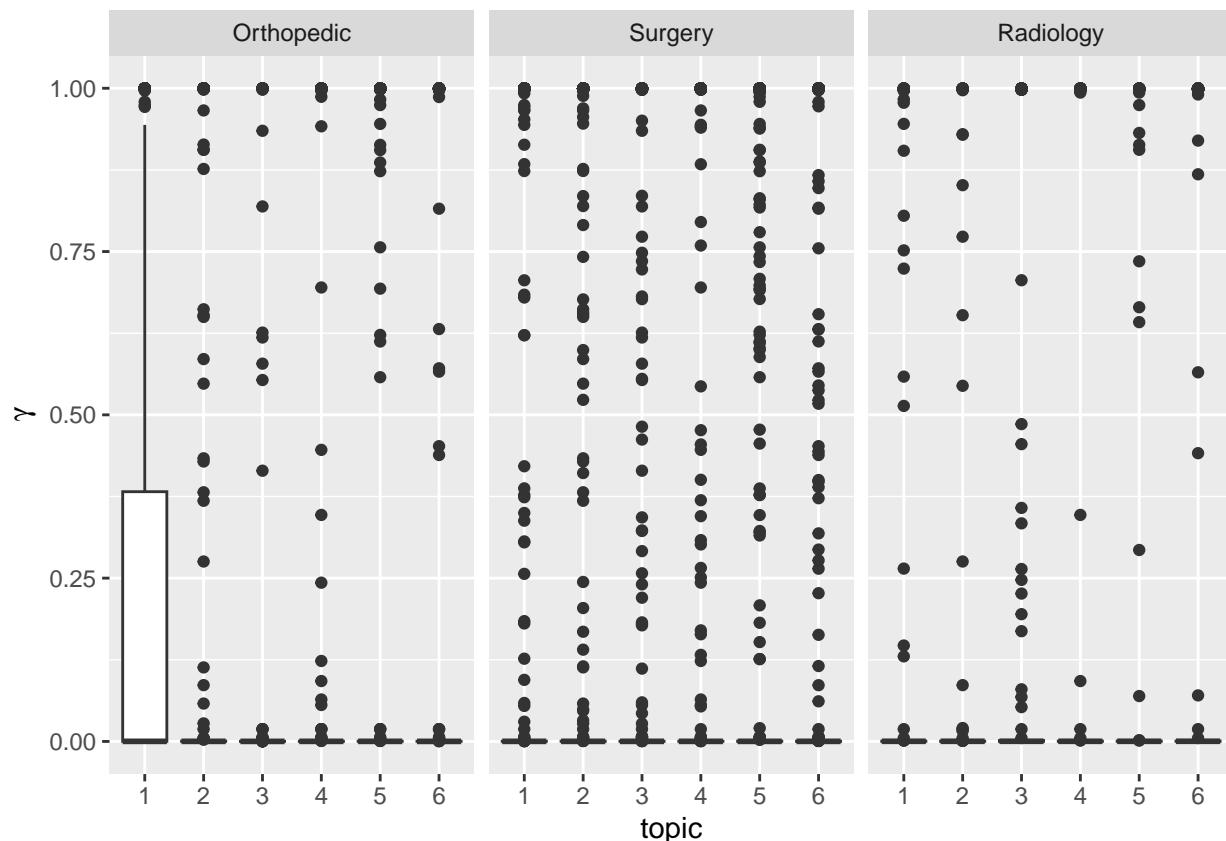
## Joining with 'by = join_by(document)'

```

```

specialty_gamma %>%
  dplyr::mutate(medical_specialty = reorder(medical_specialty, gamma * topic)) %>%
  ggplot2::ggplot(ggplot2::aes(factor(topic), gamma)) +
  ggplot2::geom_boxplot() +
  ggplot2::facet_wrap(~ medical_specialty) +
  ggplot2::labs(x = "topic", y = expression(gamma))

```



The top terms from the 5 top LDA still turned up in the 6 topic LDA. There were some minor differences, however, it appears that most all top terms were included in both. For example, carpal ligament, transverse carpal, and nerve root were included in LDA 5 and did not appear in LDA 6.

The specialties are split into sub-topics in a similar way as in the last LDA. Surgery is less diverse in topics of transcriptions, which makes sense as it is a specialized field, and therefore more likely to stick to similar topics. Radiology appears to be the most diverse, as there are many reasons/diagnoses that someone may receive radiological imaging, and for many parts of the body. Orthopedic also has a fair bit of diversity, but not as much as radiology.

## Credits

Examples draw heavily on material (and directly quotes/copies text) from Julia Slige's `tidytext` textbook.