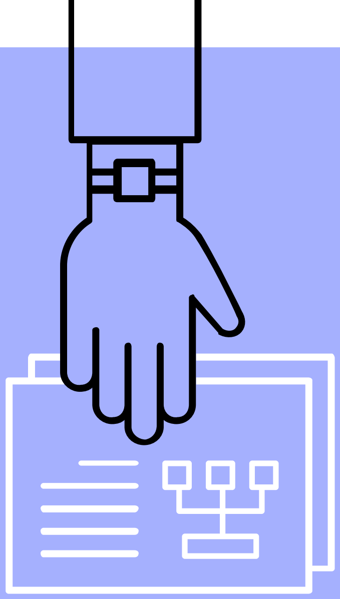
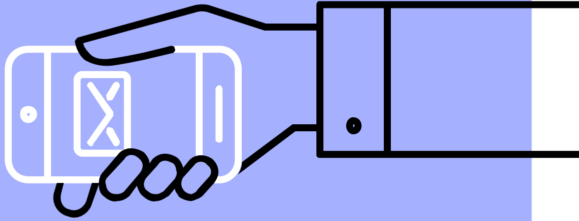
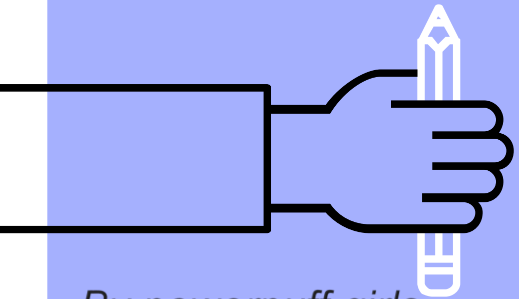
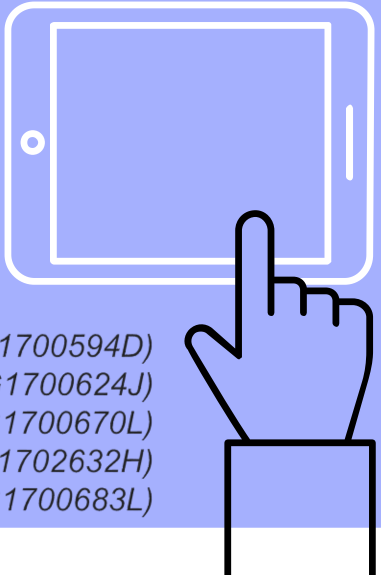




Text Mining

Program Demo





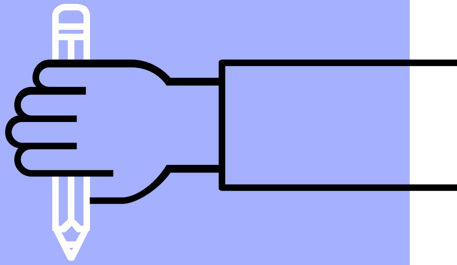
Yang Zhou	*(G1700594D)*
Zeng Jiameng	*(G1700624J)*
Sun Qianqian	*(G1700670L)*
Saumya Agarwal	*(G1702632H)*
Wang Yanhang	*(G1700683L)*

By powerpuff girls

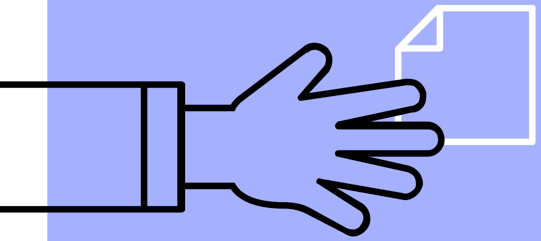
❝ ▹ Outline

- Introduction
- Pre-processing
- Sentiment Analysis
- Topic Modelling

[redacted]



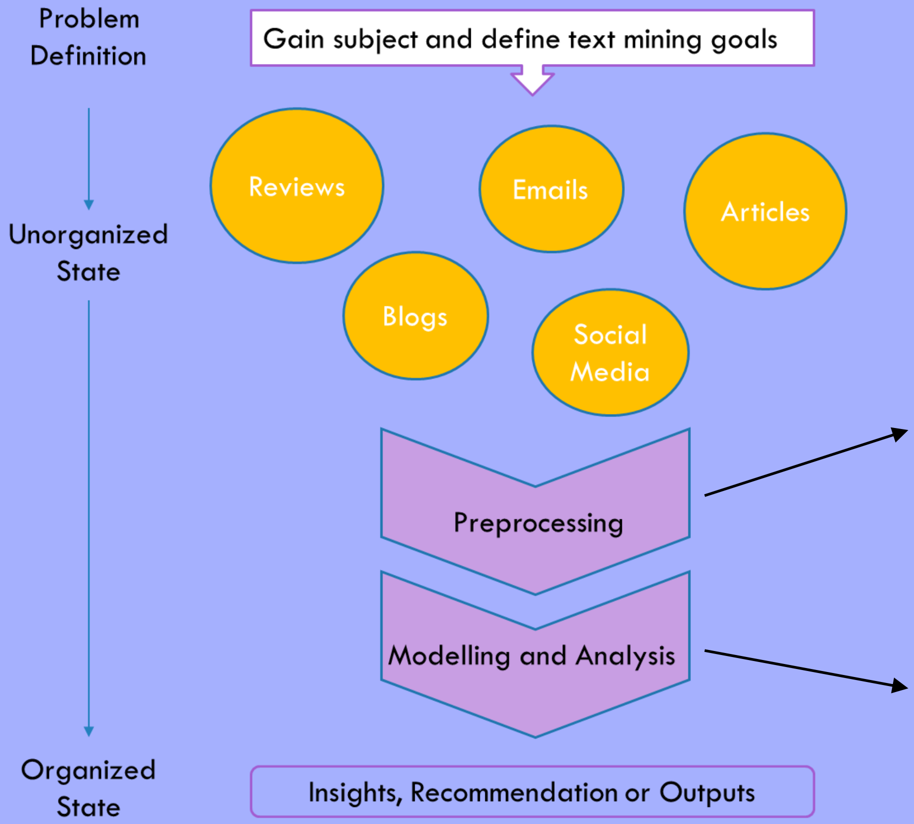
1.
Introduction



What is text mining ?

- Ability to approach the unstructured text
- Process of understanding information and find out valuable knowledges

Overview of process



- Importing
- String operations and pre-processing
- Creating the document-term matrix
- Filter and weight the terms in the DTM

- Sentiment analysis
- Topic Modelling

[redacted]

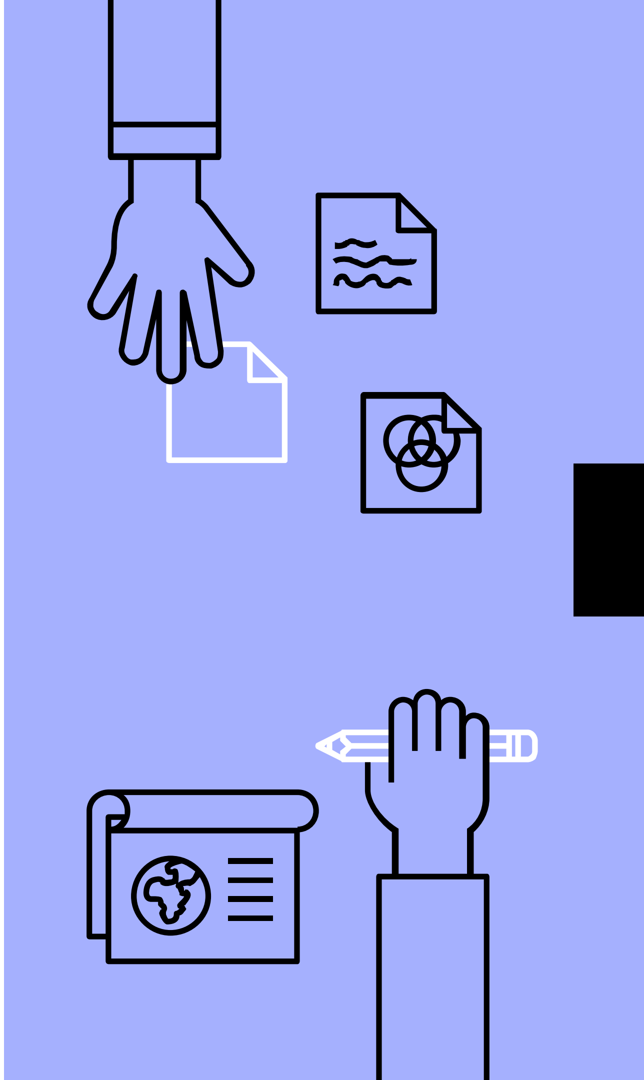
Two approaches

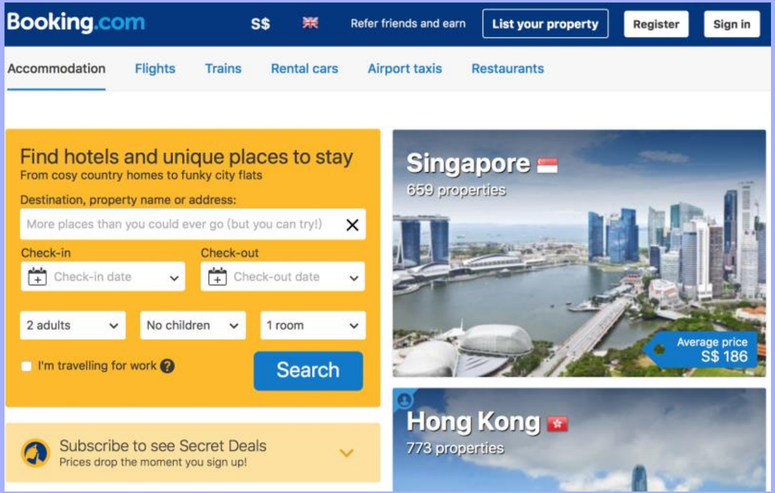
Sentiment Analysis

- identifying and determine whether the writer's attitude towards a particular topic or product is positive, negative, or neutral.
- Supervised (Classification model)
- Unsupervised (Dictionary-based)

Topic Modelling

- discovering the abstract "topics" that occur in a collection of documents.





Booking.com S$ Refer friends and earn List your property Register Sign in

Accommodation Flights Trains Rental cars Airport taxis Restaurants

Find hotels and unique places to stay

From cosy country homes to funky city flats

Destination, property name or address:

More places than you could ever go (but you can try!)

Check-in: Check-in date

Check-out: Check-out date

2 adults | No children | 1 room

□ I'm travelling for work

Search

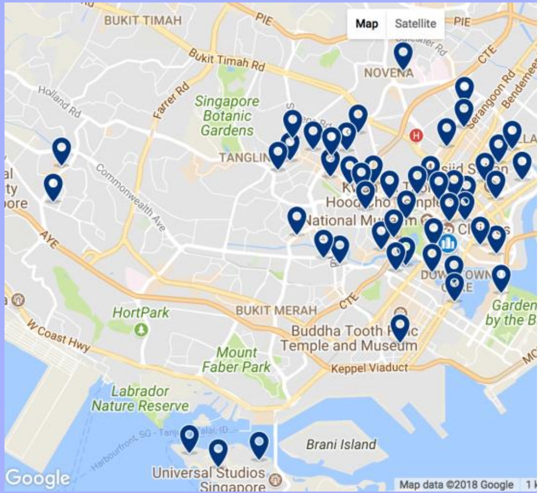
Subscribe to see Secret Deals

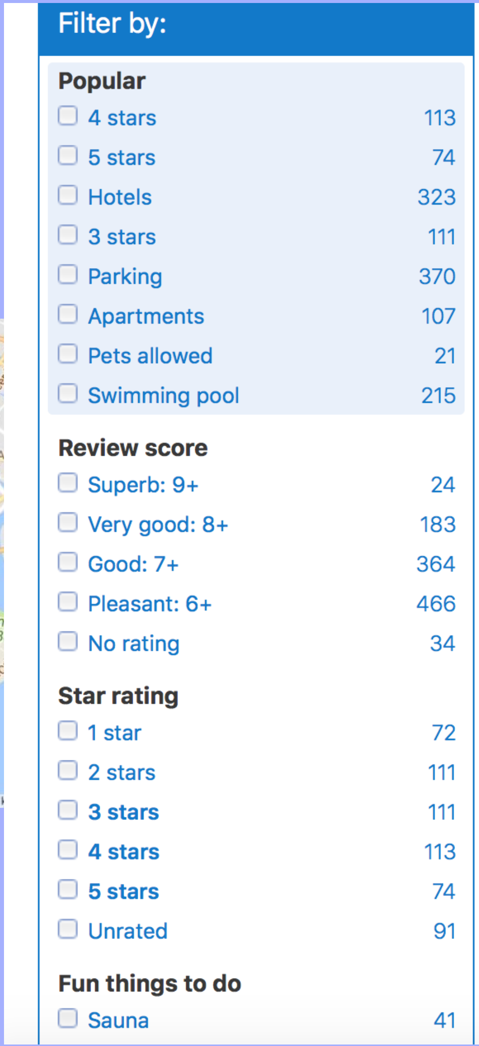
Prices drop the moment you sign up!

Singapore 659 properties

Average price S$ 186

Hong Kong 773 properties





Filter by:

Popular

□ 4 stars	113
□ 5 stars	74
□ Hotels	323
□ 3 stars	111
□ Parking	370
□ Apartments	107
□ Pets allowed	21
□ Swimming pool	215

Review score

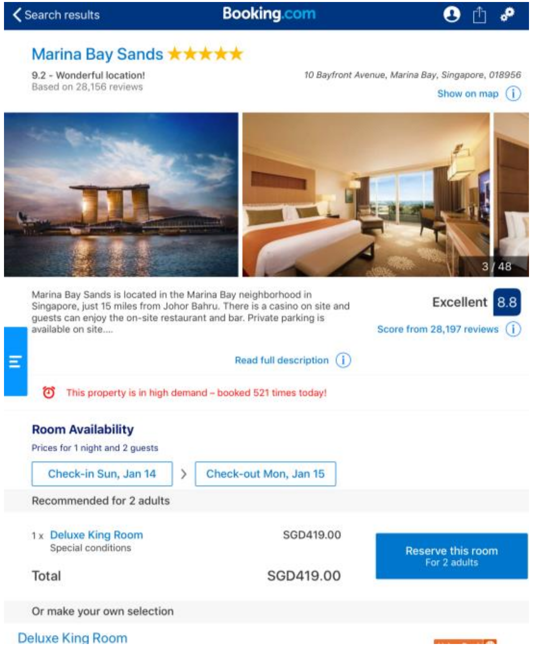
□ Superb: 9+	24
□ Very good: 8+	183
□ Good: 7+	364
□ Pleasant: 6+	466
□ No rating	34

Star rating

□ 1 star	72
□ 2 stars	111
□ 3 stars	111
□ 4 stars	113
□ 5 stars	74
□ Unrated	91

Fun things to do

□ Sauna	41



Marina Bay Sands ★★★★★

9.2 - Wonderful location!
Based on 28,156 reviews

10 Bayfront Avenue, Marina Bay, Singapore, 018956

Show on map

3 / 48

Marina Bay Sands is located in the Marina Bay neighborhood in Singapore, just 15 miles from Johor Bahru. There is a casino on site and guests can enjoy the on-site restaurant and bar. Private parking is available on site....

Excellent 8.8

Score from 28,197 reviews

Read full description

This property is in high demand – booked 521 times today!

Room Availability

Prices for 1 night and 2 guests

Check-in Sun, Jan 14 › Check-out Mon, Jan 15

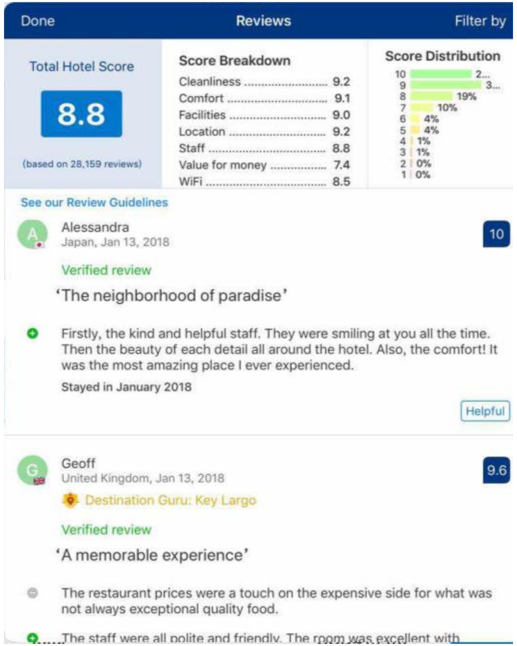
Recommended for 2 adults

1 x Deluxe King Room Special conditions	SGD419.00
Total	SGD419.00

Reserve this room
For 2 adults

Or make your own selection

Deluxe King Room



Total Hotel Score

8.8

(based on 28,159 reviews)

Score Breakdown

Cleanliness	9.2
Comfort	9.1
Facilities	9.0
Location	9.2
Staff	8.8
Value for money	7.4
WiFi	8.5

Score Distribution

10	2...
9	3...
8	19%
7	10%
6	4%
5	4%
4	1%
3	1%
2	0%
1	0%

See our Review Guidelines

Alessandra
Japan, Jan 13, 2018
10

Verified review

'The neighborhood of paradise'

Firstly, the kind and helpful staff. They were smiling at you all the time. Then the beauty of each detail all around the hotel. Also, the comfort! It was the most amazing place I ever experienced.

Stayed in January 2018

Helpful

Geoff
United Kingdom, Jan 13, 2018
9.6

Destination Guru: Key Largo

Verified review

'A memorable experience'

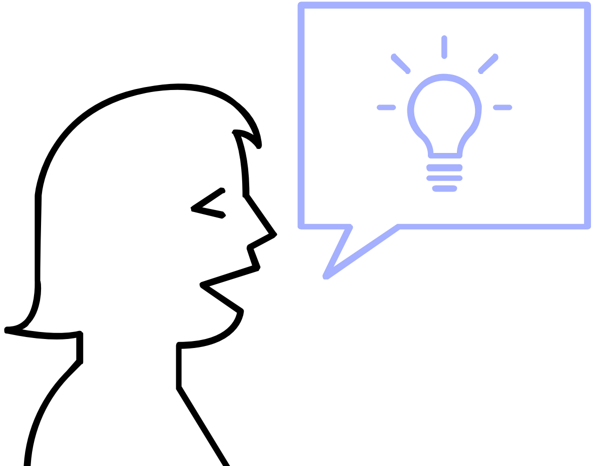
The restaurant prices were a touch on the expensive side for what was not always exceptional quality food.

The staff were all polite and friendly. The room was excellent with

Booking.com

TEXT PRE-PROCESSING
TEXT ANALYSIS

DATASET



Total 6,300 reviews covering 6 hotels:

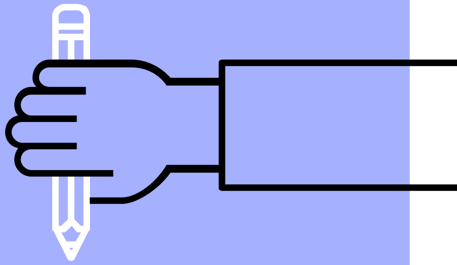
5 STAR: Marina Bay Sands, Crowne Plaza Changi Airport

4 STAR: PARKROYAL on Beach Road, Siloso Beach Resort Sentosa

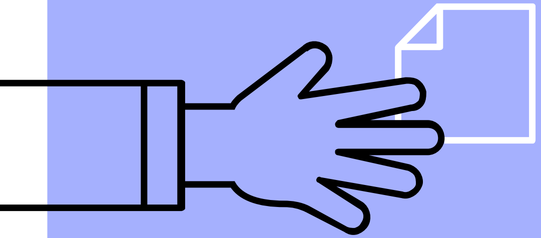
3 STAR: Ibis Singapore on Bencoolen, Destination Singapore Beach Road

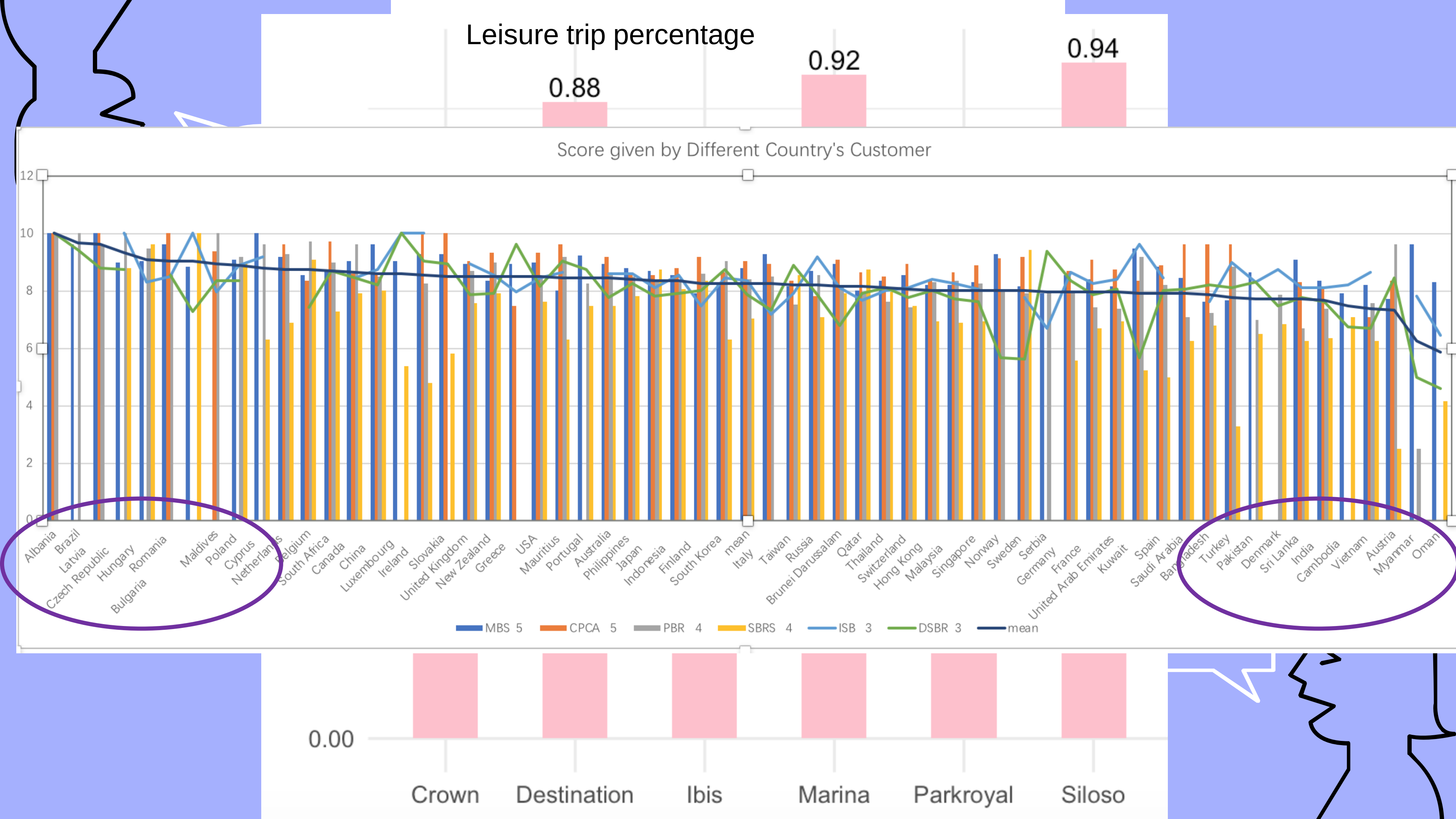
Customer_ID	nationality	tags	score	Content	Content_detail
1	Canada	<U+2022> Leisure trip<U+2022> Couple<U+2022> The Grand Club King Room<U+2022> Stayed 2 nights	9.2	“So, amazing in many ways but did not receive a personal treatment, however club lounge was good”	<U+B198>I was really surprised there was no turn down service, which is normally standard …..Stayed in December 2017

Trip type	#customer	Room type	#night	Submit way
leisure	couple	The Grand club king room	2	mobile
business	solo	Deluxe king room	1	



2.
Data Preparation




Leisure trip percentage
0.88
0.92
0.94
0.00
Crown
Destination
Ibis
Marina
Parkroyal
Siloso
Score given by Different Country's Customer
12
10
8
6
4
2
0
Albania
Brazil
Latvia
Czech Republic
Hungary
Bulgaria
Romania
Maldives
Poland
Cyprus
Netherlands
Belgium
South Africa
Canada
China
Luxembourg
Ireland
Slovakia
United Kingdom
New Zealand
Greece
USA
Mauritius
Portugal
Australia
Philippines
Japan
Indonesia
Finland
South Korea
mean
Italy
Taiwan
Russia
Brunei Darussalam
Qatar
Thailand
Switzerland
Hong Kong
Malaysia
Singapore
Norway
Sweden
Serbia
Germany
France
United Arab Emirates
Kuwait
Spain
Saudi Arabia
Bangladesh
Turkey
Pakistan
Denmark
Sri Lanka
India
Cambodia
Vietnam
Austria
Myanmar
Oman
MBS 5
CPCA 5
PBR 4
SBRS 4
ISB 3
DSBR 3
mean

Importing text

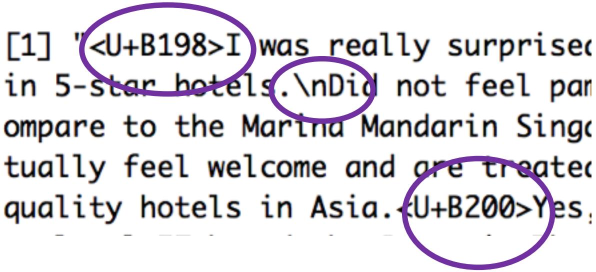
Mining text data from website by R.

Write the data into csv text files which is easy to re-read into R for the later operation.

Customer_ID	nationality	tags	score	Content	Content_detail
1	Canada	<U+2022> Leisure trip<U+2022> Couple<U+2022> The Grand Club King Room<U+2022> Stayed 2 nights	9.2	“So, amazing in many ways but did not receive a personal treatment, however club lounge was good”	<U+B198>I was really surprised there was no turn down service, which is normally standard ⋯..Stayed in December 2017
2	Qatar	<U+2022> Leisure trip<U+2022> Couple<U+2022> Deluxe King Sky View<U+2022> Stayed 1 night<U+2022> Submitted via mobile	7.5	“Nice but missing quite a few small details youa<U+0080><U+0099>d expect with a five star.”	<U+B198>Overpriced in general<U+B200>Amazing view as expected and paid forStayed in January 2018
3	…	…	…	…	…

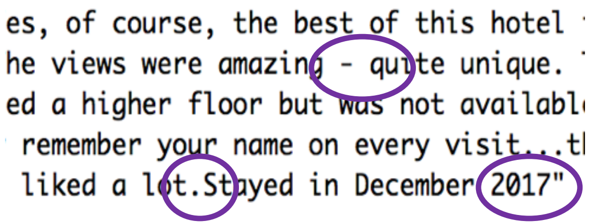
String operations

Remove html tags

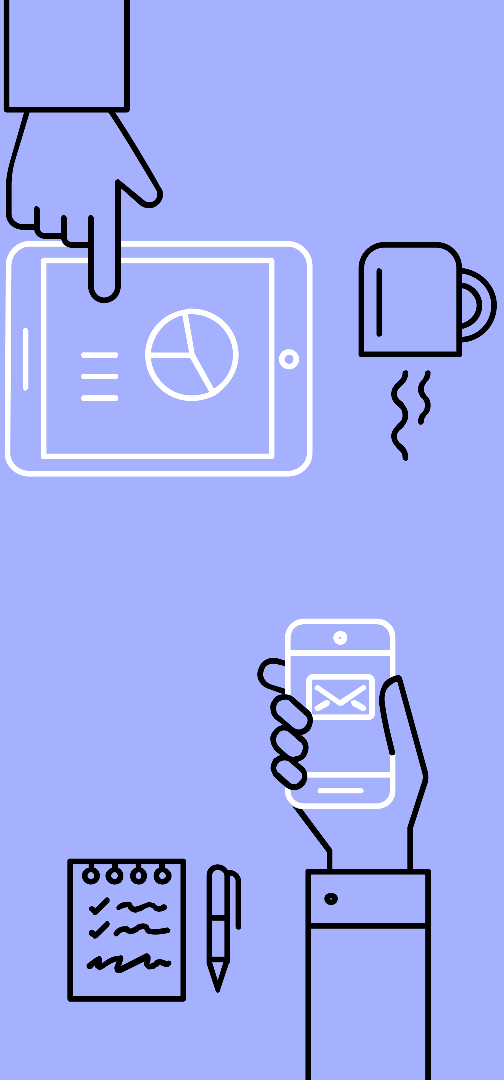


```
[1] "<U+B198>I was really surprisec
in 5-star hotels.\nDid not feel pan
ompare to the Marina Mandarin Singc
tually feel welcome and are treatec
quality hotels in Asia.<U+B200>Yes,
```

Remove digit and punctuation



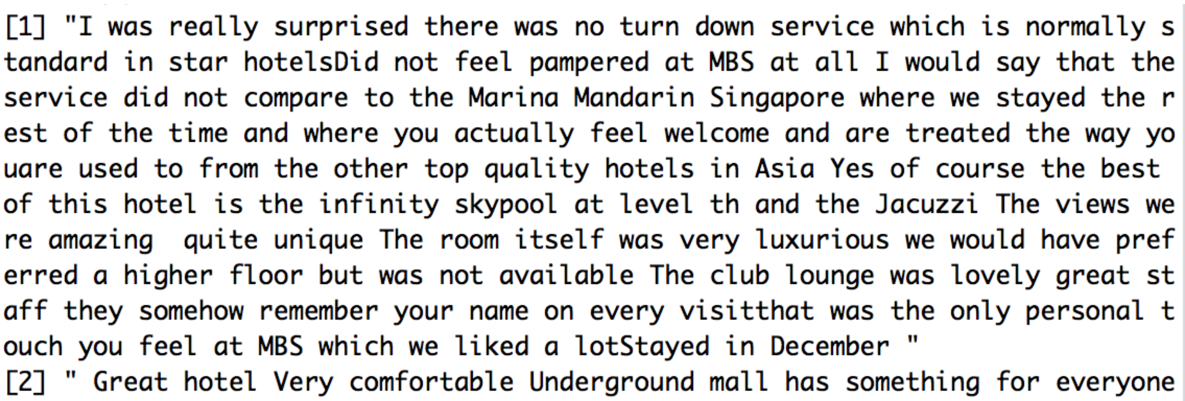
```
es, of course, the best of this hotel
he views were amazing - quite unique.
ed a higher floor but was not availabl
 remember your name on every visit...tl
 liked a lot.Stayed in December 2017"
```



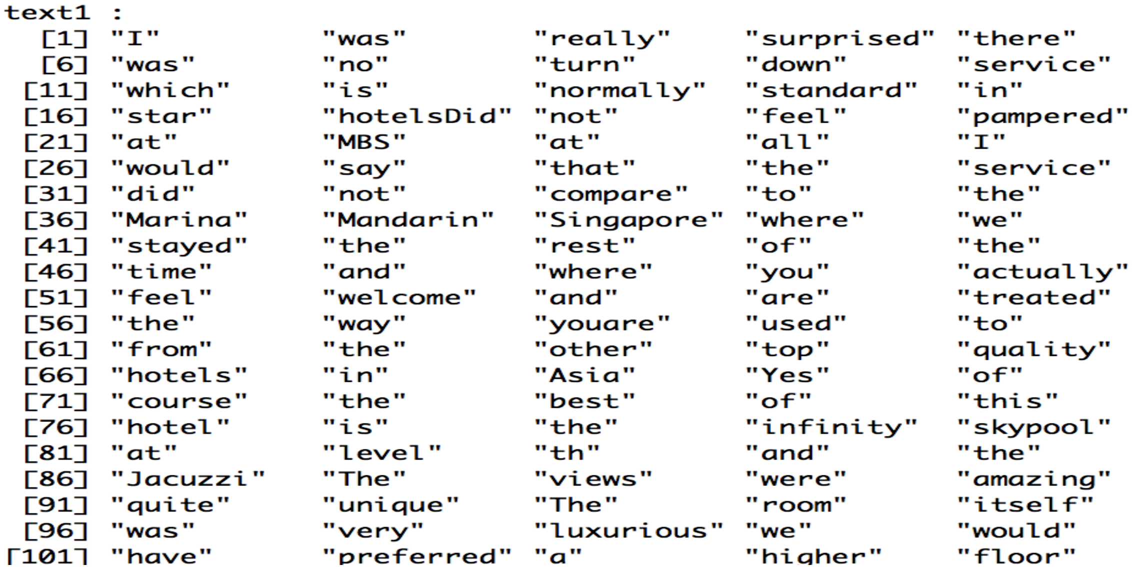
preprocessing

tokenization

- Tokenization is the process of splitting a text into tokens.
- splitting texts by words can mostly be done by word boundaries, such as white spaces, dots and commas.



```
[1] "I was really surprised there was no turn down service which is normally s
tandard in star hotelsDid not feel pampered at MBS at all I would say that the
service did not compare to the Marina Mandarin Singapore where we stayed the r
est of the time and where you actually feel welcome and are treated the way yo
uare used to from the other top quality hotels in Asia Yes of course the best
of this hotel is the infinity skypool at level th and the Jacuzzi The views we
re amazing  quite unique The room itself was very luxurious we would have pref
erred a higher floor but was not available The club lounge was lovely great st
aff they somehow remember your name on every visitthat was the only personal t
ouch you feel at MBS which we liked a lotStayed in December "
[2] " Great hotel Very comfortable Underground mall has something for everyone
```



```
text1 :
  [1] "I"          "was"        "really"     "surprised"  "there"
  [6] "was"        "no"         "turn"       "down"       "service"
 [11] "which"      "is"         "normally"   "standard"   "in"
 [16] "star"       "hotelsDid"  "not"        "feel"       "pampered"
 [21] "at"         "MBS"        "at"         "all"        "I"
 [26] "would"      "say"        "that"       "the"        "service"
 [31] "did"        "not"        "compare"    "to"         "the"
 [36] "Marina"     "Mandarin"   "Singapore"  "where"      "we"
 [41] "stayed"     "the"        "rest"       "of"         "the"
 [46] "time"       "and"        "where"      "you"        "actually"
 [51] "feel"       "welcome"    "and"        "are"        "treated"
 [56] "the"        "way"        "youare"     "used"       "to"
 [61] "from"       "the"        "other"      "top"        "quality"
 [66] "hotels"     "in"         "Asia"       "Yes"        "of"
 [71] "course"     "the"        "best"       "of"         "this"
 [76] "hotel"      "is"         "the"        "infinity"   "skypool"
 [81] "at"         "level"      "th"         "and"        "the"
 [86] "Jacuzzi"    "The"        "views"      "were"       "amazing"
 [91] "quite"      "unique"     "The"        "room"       "itself"
 [96] "was"        "very"       "luxurious"  "we"         "would"
[101] "have"       "preferred"  "a"          "higher"     "floor"
```

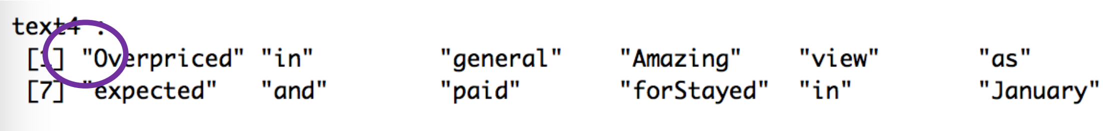
preprocessing

Normalization

transformation of words into a more uniform form.

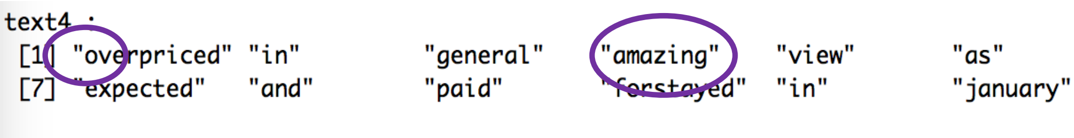
- Lowercasing

- stemming



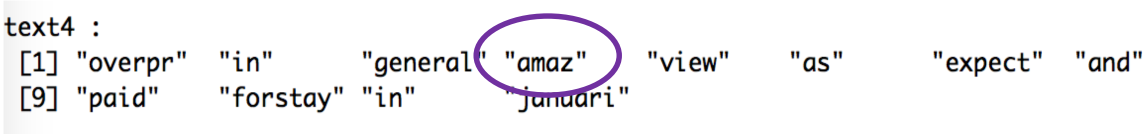
```
text4 :
 [1] "Overpriced" "in"         "general"  "Amazing"    "view"      "as"
 [7] "expected"   "and"        "paid"     "forStayed"  "in"        "January"
```

Transform it in to lower



```
text4 :
 [1] "overpriced" "in"         "general"  "amazing"    "view"      "as"
 [7] "expected"   "and"        "paid"     "forstayed"  "in"        "january"
```

Stemming to reduce the feature space

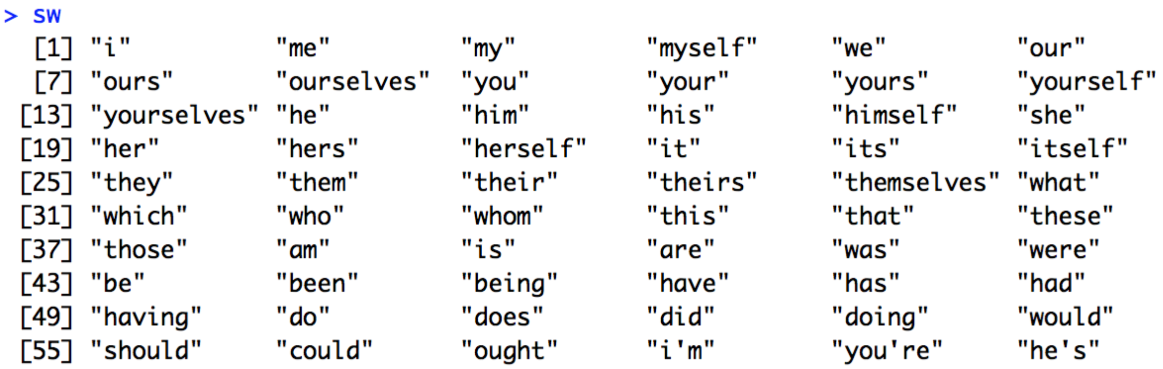


```
text4 :
 [1] "overpr"  "in"      "general" "amaz"    "view"    "as"      "expect"  "and"
 [9] "paid"    "forstay" "in"      "januari"
```

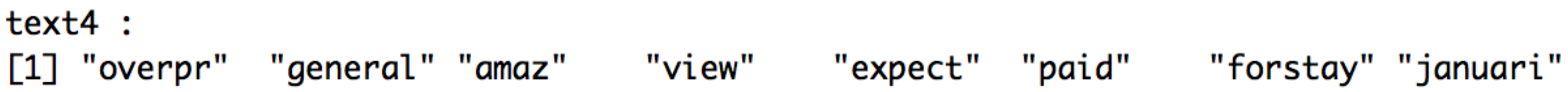
preprocessing

Removing stopwords

- reduce the size of the data
- reduce computational load
- improve accuracy.



```
> sw
 [1] "i"          "me"         "my"         "myself"     "we"         "our"
 [7] "ours"       "ourselves"  "you"        "your"       "yours"      "yourself"
[13] "yourselves" "he"         "him"        "his"        "himself"    "she"
[19] "her"        "hers"       "herself"    "it"         "its"        "itself"
[25] "they"       "them"       "their"      "theirs"     "themselves" "what"
[31] "which"      "who"        "whom"       "this"       "that"       "these"
[37] "those"      "am"         "is"         "are"        "was"        "were"
[43] "be"         "been"       "being"      "have"       "has"        "had"
[49] "having"     "do"         "does"       "did"        "doing"      "would"
[55] "should"     "could"      "ought"      "i'm"        "you're"     "he's"
```

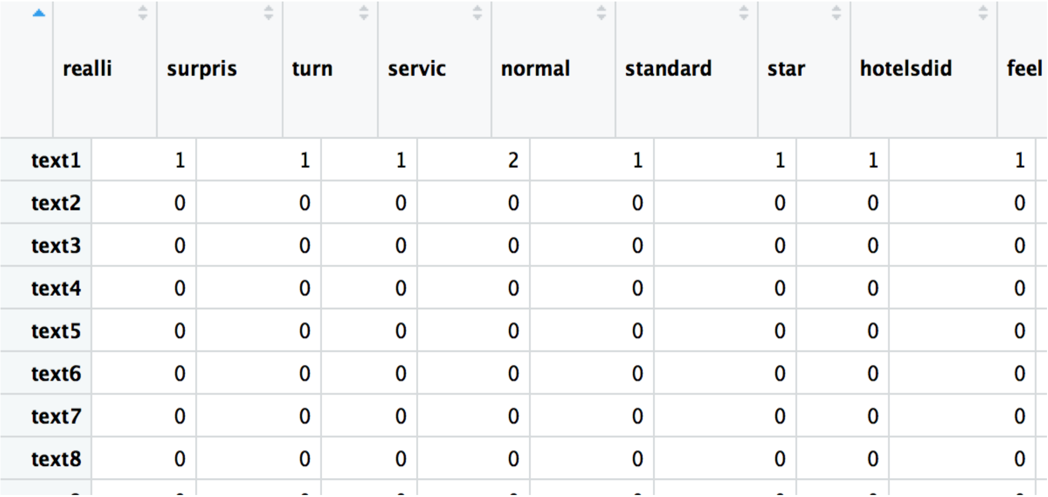


```
text4 :
[1] "overpr"   "general" "amaz"     "view"     "expect"   "paid"     "forstay" "januari"
```

preprocessing

Document-term matrix

- A DTM is a matrix in which rows aredocuments, columns are terms, and cells indicate how often each term occurred in each document.



	realli	surpris	turn	servic	normal	standard	star	hotelsdid	feel
text1	1	1	1	2	1	1	1	1	
text2	0	0	0	0	0	0	0	0	
text3	0	0	0	0	0	0	0	0	
text4	0	0	0	0	0	0	0	0	
text5	0	0	0	0	0	0	0	0	
text6	0	0	0	0	0	0	0	0	
text7	0	0	0	0	0	0	0	0	
text8	0	0	0	0	0	0	0	0	

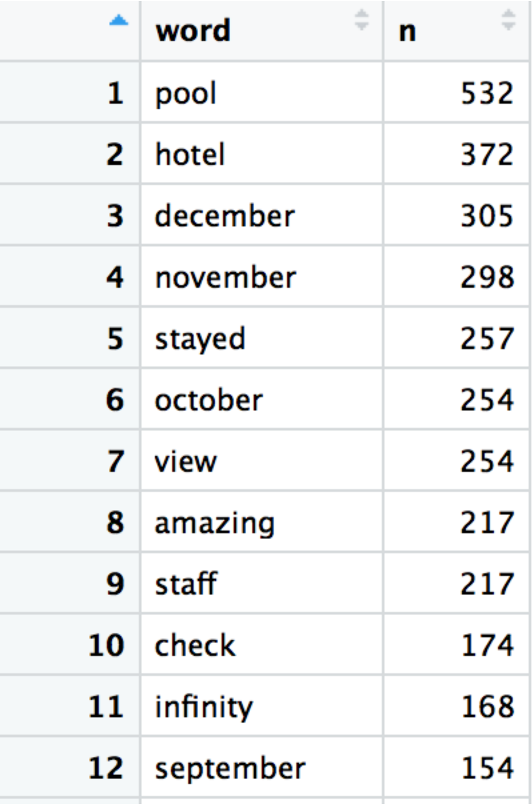
Document-feature matrix of: 1,050 documents, 2,806 features (99.4% sparse).

preprocessing

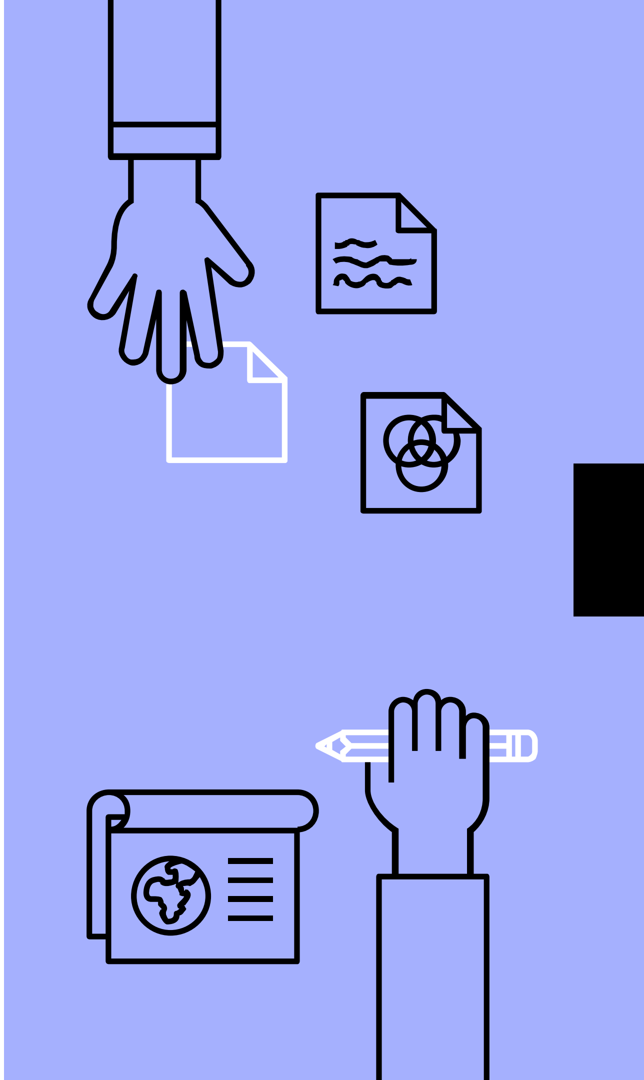
Calculate word count

- Rank words according to frequency

- using a threshold for minimum and maximum number (or proportion) of documents

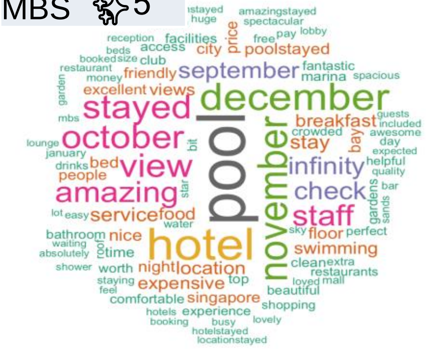


	word	n
1	pool	532
2	hotel	372
3	december	305
4	november	298
5	stayed	257
6	october	254
7	view	254
8	amazing	217
9	staff	217
10	check	174
11	infinity	168
12	september	154



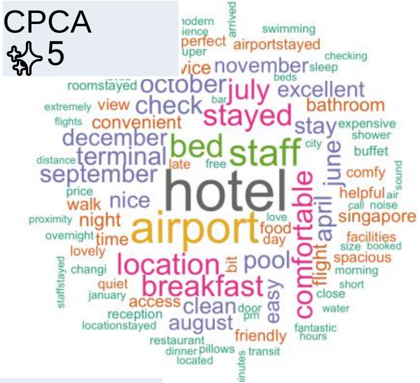
wordclouds

MBS ✨5

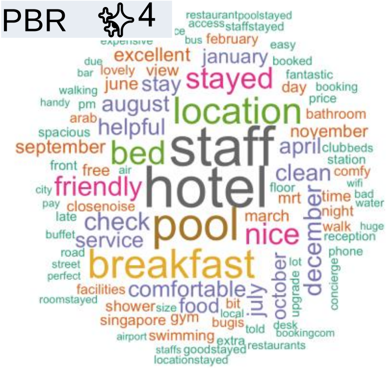


CPCA

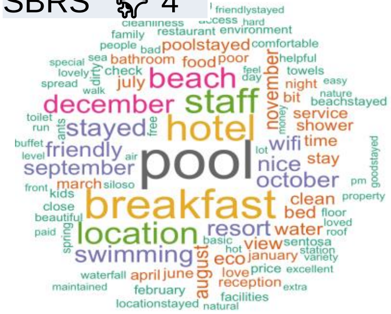
✨5



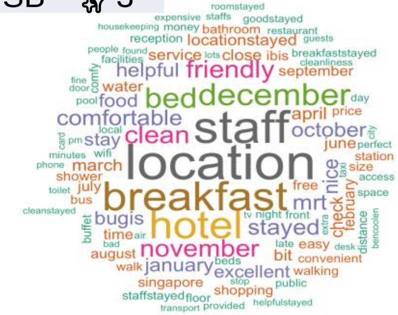
PBR ✨4



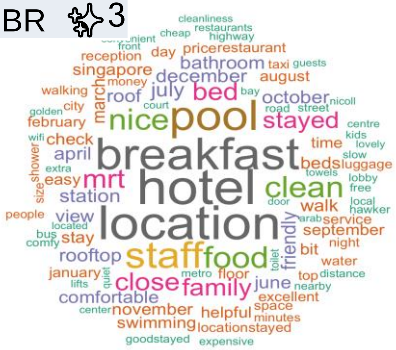
SBRS ✨ 4

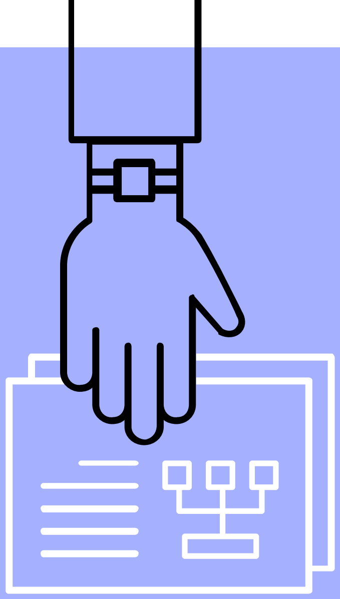


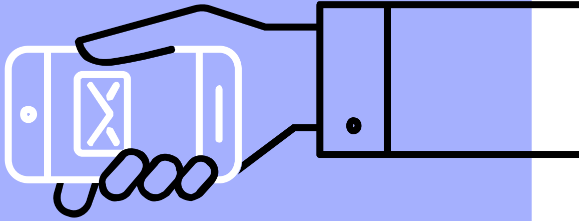
ISB ✨3



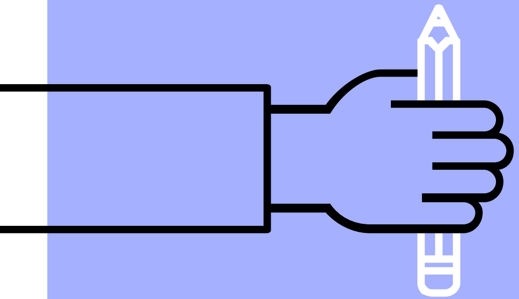
DSBR ✨3

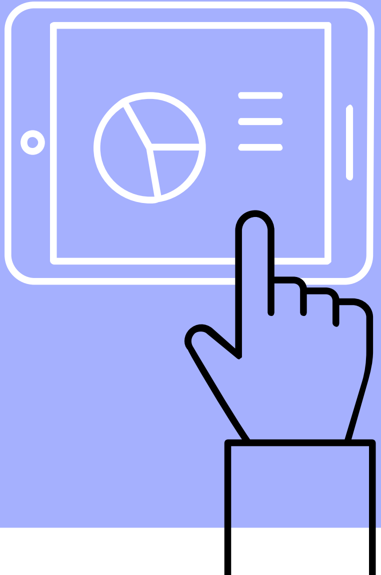






Lexicon based Sentiment Analysis





What are Lexicons?

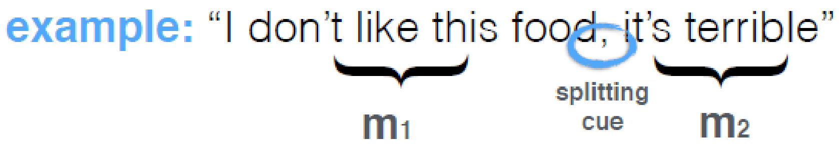
A lexicon is a collection of lexemes or a word database. A lexeme roughly corresponds to a set of words that are different forms of "the same word".

For example, English run, runs, ran and running are forms of the same lexeme.

The sentiment of a textual content depends on the sentiment of each microphase or lexemes which compose it

A microphase is built whenever a splitting cue is found in the text

Conjunctions, Adverbs and punctuations are used as Splitting cues.


example: "I don't like this food, it's terrible"

Lexical Resources

SentiWordNet

http://sentiwordnet.isti.cnr.it

WordNet Affect

http://wndomains.fbk.eu/wnaffect.html

SenticNet

http://sentic.net

MPQA

http://mpqa.cs.pitt.edu

Each word is provided a discrete sentiment score.

Packages in Sentiment Analysis

library(tm)

The main structure for managing documents in **tm** is a so-called Corpus, representing a collection of text documents

library(wordcloud)

library(SnowballC)

library(rJava)

library(Rwordseg)

This package can be used on English as well as Chinese text.

library(plyr)

library(wordcloud2)

Other Packages

library(tidytetxt)

library(sentimentalanalysis)

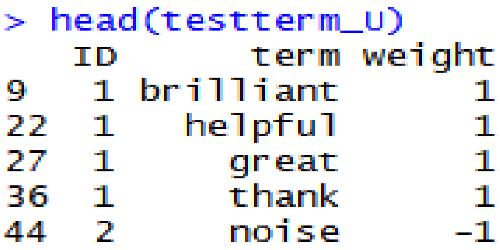
library(syuzhet)

library(sentimentr)

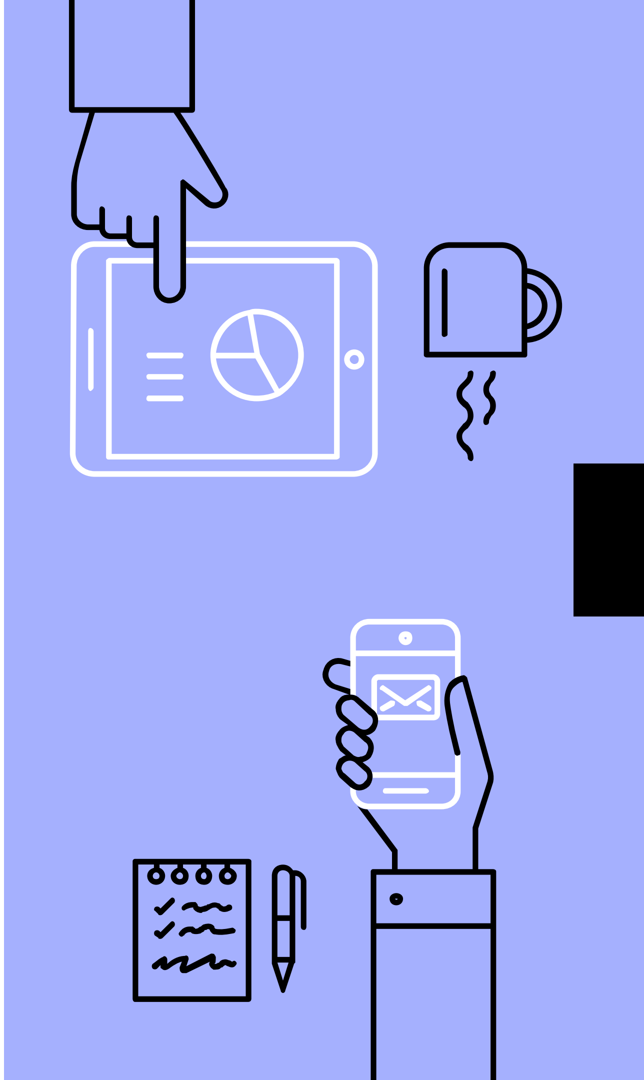
Filtering and Weighting

Each Word is provided with three different sentiment scores (positivity, negativity, objectivity). For Simplicity we have assigned a +1 to positive words, -1 to negative words and 0 to the neutral words.

Positive ▲	**Negative** ▲	**Neutral** ▲
Amazing	Alarming	Again
Eventful	Contradiction	Instead
Great	Hideousness	Provided
Nice	Provocative	Somebody
Trustworthy	Wasteful	Whatever



```
> head(testterm_U)
   ID        term weight
9   1 brilliant      1
22  1   helpful      1
27  1     great      1
36  1     thank      1
44  2     noise     -1
```



Counting and Sentiment Scoring

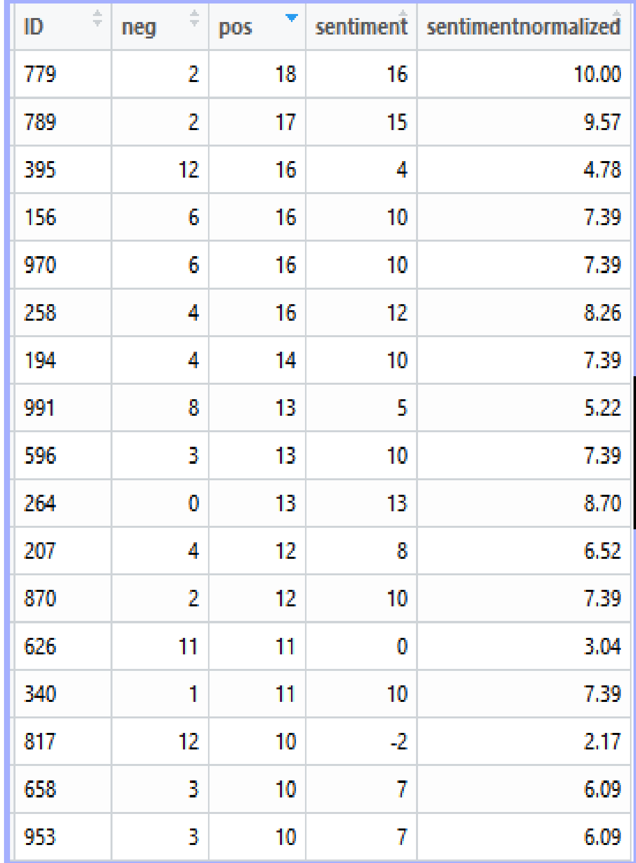
The sentiment of a **Review** depends on the sentiment of the **terms** which compose it.

$$pol(\underset{\uparrow\ \text{Tweet}}{\mathbf{T}}) = \sum_{i=1}^{k} pol(\underset{\uparrow\ \text{microphrase}}{\mathbf{m_i}}) \qquad \mathbf{T=\{m_1 \ldots m_k\}}$$

$$pol(\mathbf{m_i}) = \sum_{j=1}^{n} score(\underset{\uparrow\ \text{term}}{\mathbf{t_j}}) \qquad \mathbf{M_i=\{t_1 \ldots t_n\}}$$

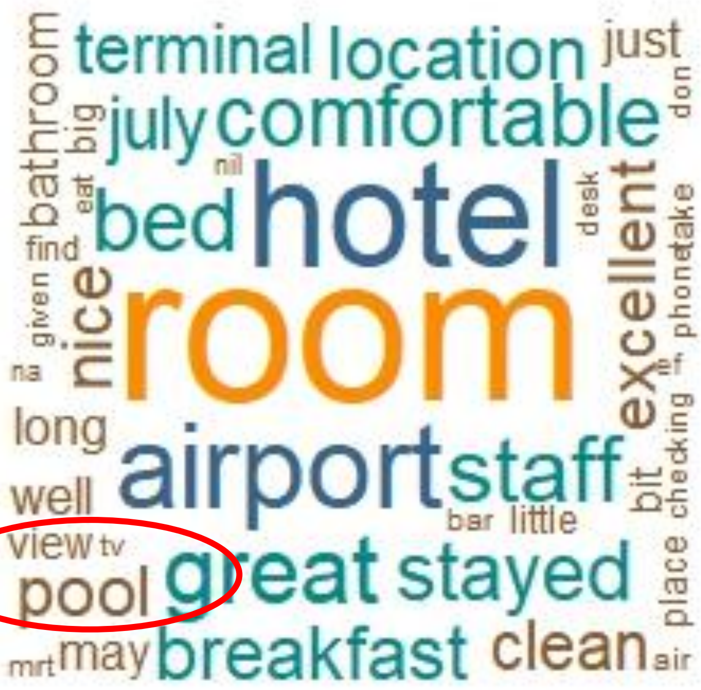
floor is dusty, there are ants everywhere The hotel is eco friendly which is lovely!

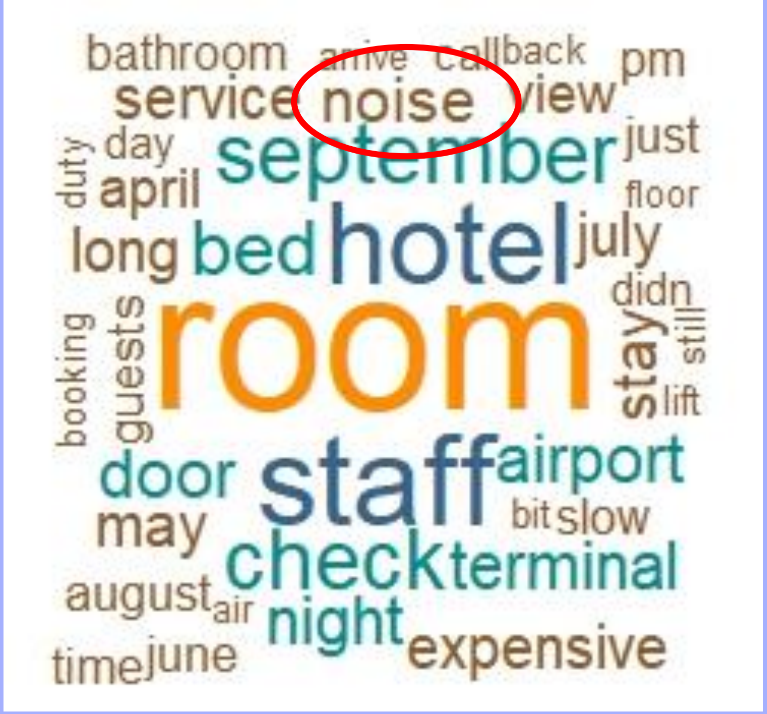
ID	Review	Weight
2	dusty	-1
2	everywhere	0
2	friendly	1
2	lovely	1



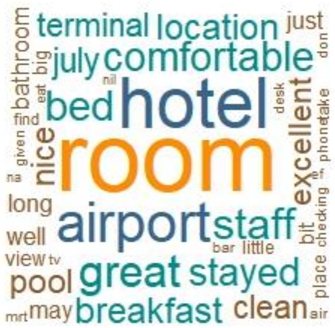
ID	neg	pos	sentiment	sentimentnormalized
779	2	18	16	10.00
789	2	17	15	9.57
395	12	16	4	4.78
156	6	16	10	7.39
970	6	16	10	7.39
258	4	16	12	8.26
194	4	14	10	7.39
991	8	13	5	5.22
596	3	13	10	7.39
264	0	13	13	8.70
207	4	12	8	6.52
870	2	12	10	7.39
626	11	11	0	3.04
340	1	11	10	7.39
817	12	10	-2	2.17
658	3	10	7	6.09
953	3	10	7	6.09

Positive vs Negative Reviews

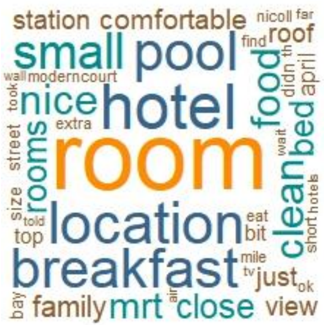




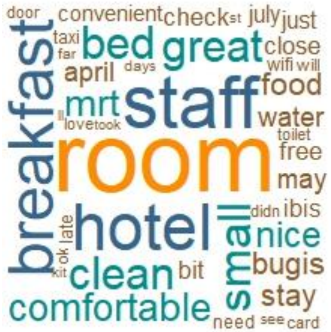
Let's look at Positive Reviews



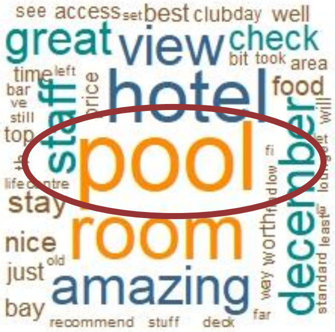
Crowne Plaza



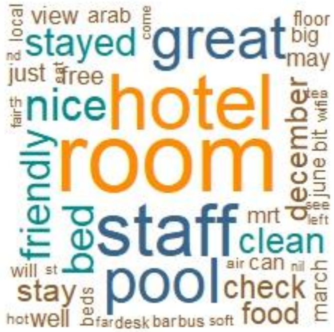
Destination Singapore



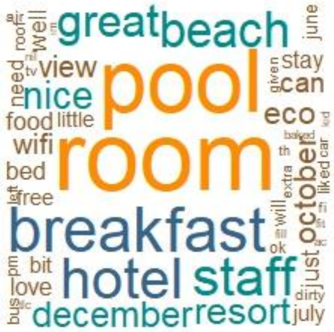
Ibis Singapore



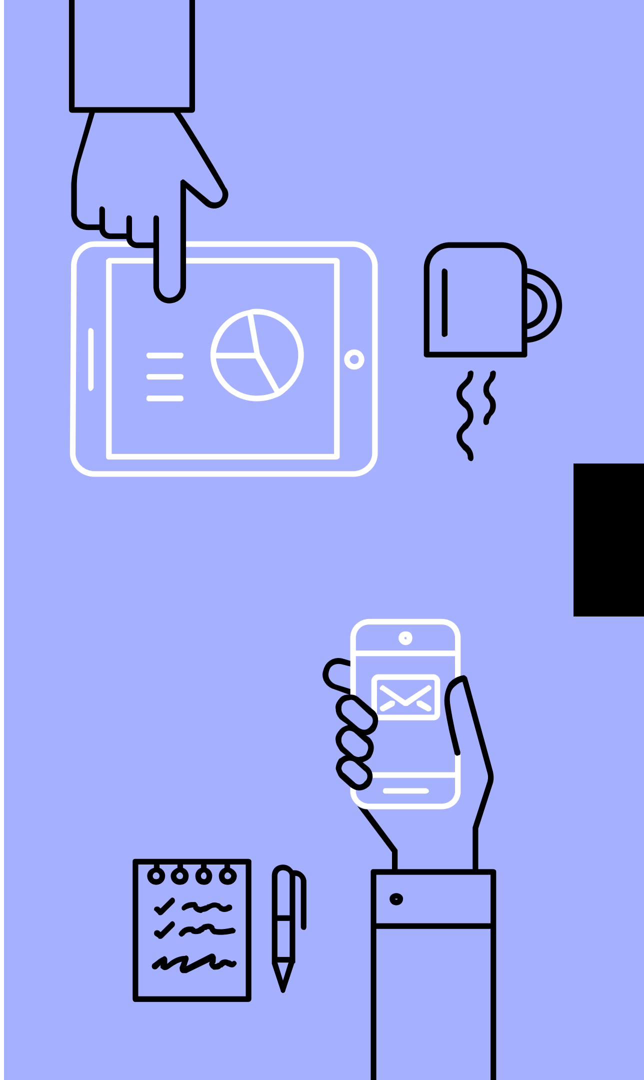
Marina Bay



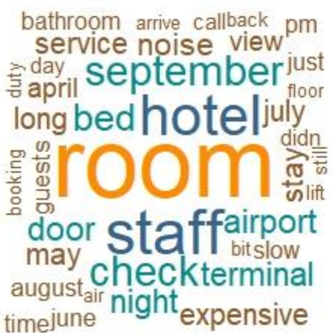
ParkRoyal



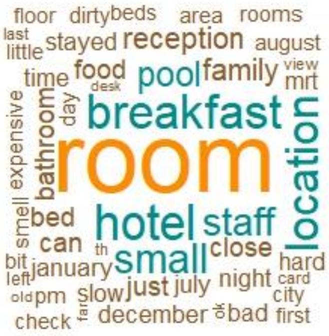
Siloso Beach Resort



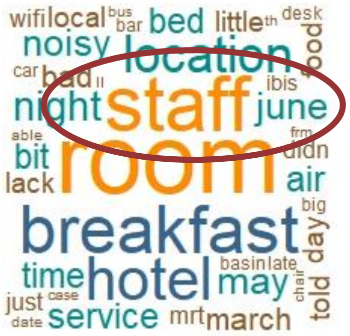
Let’s look at Negative Reviews



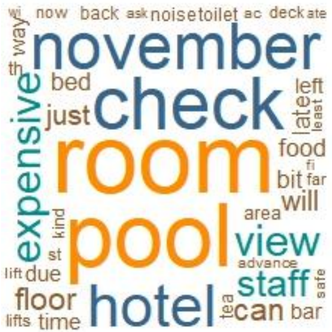
Crowne Plaza



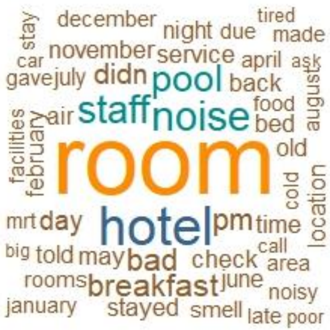
Destination Singapore



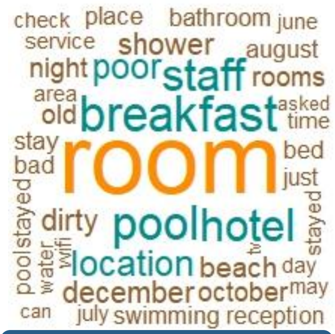
Ibis Singapore



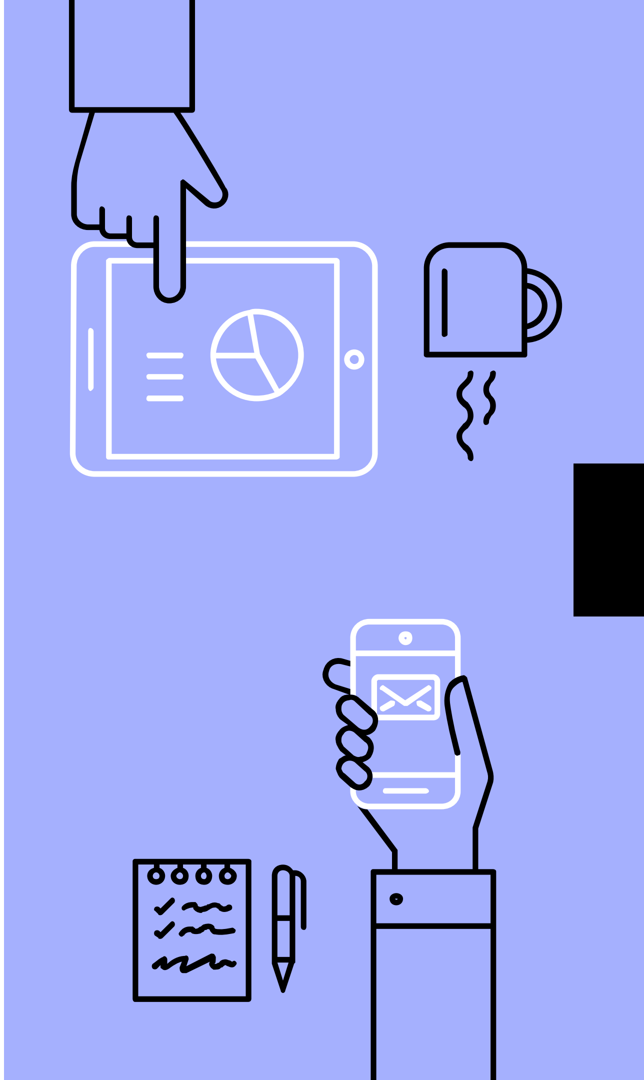
Marina Bay



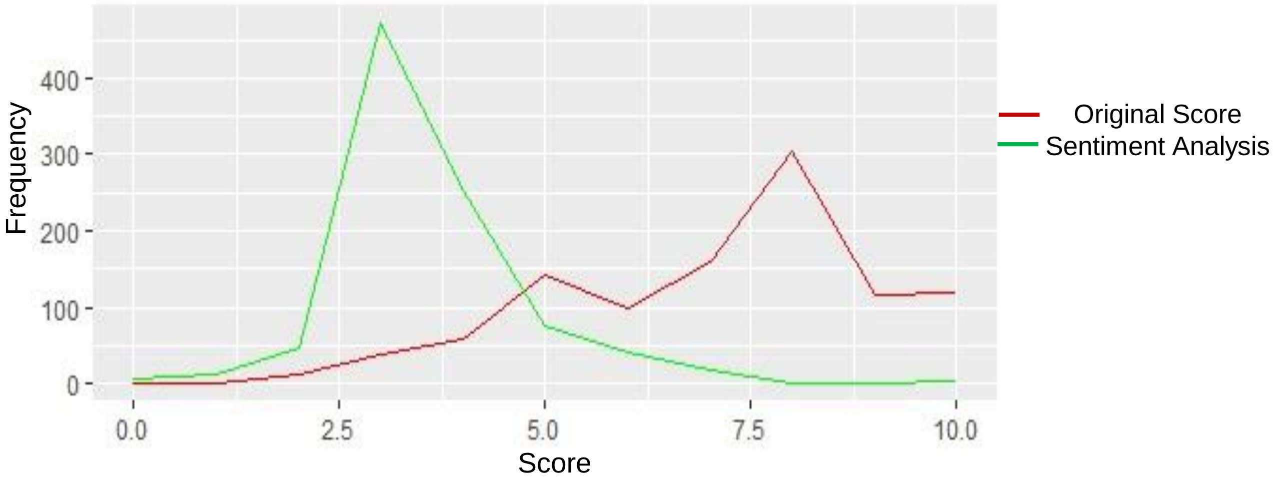
ParkRoyal



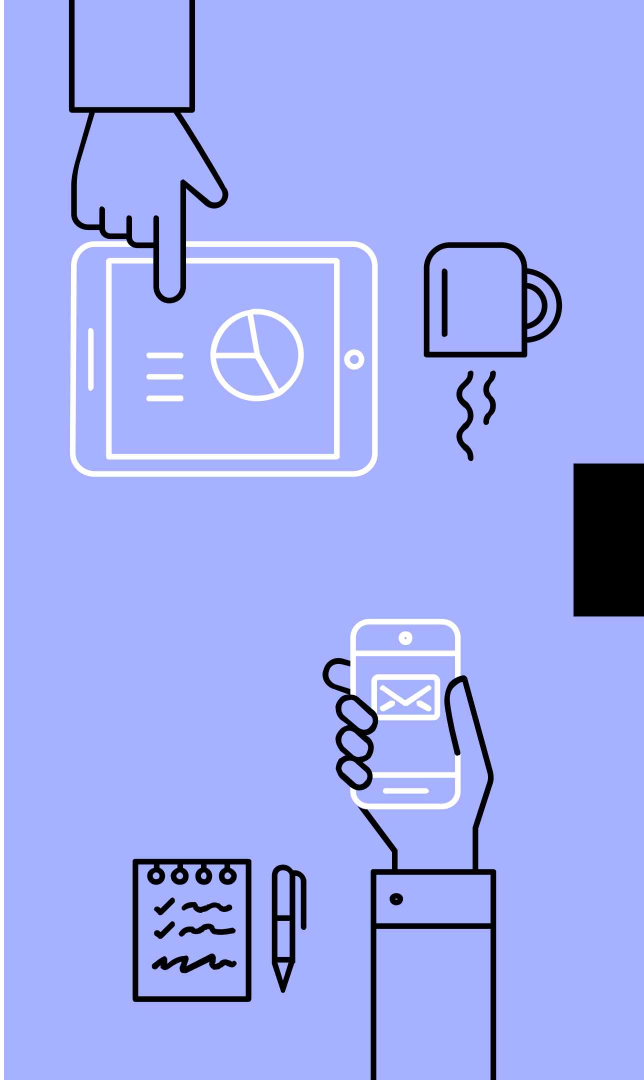
Siloso Beach Resort



Original Score vs Sentiment Analysis



Siloso Beach Resort



Two Factors Limiting Performance

Dictionary

- Level of sentiment differs for different words

Context

- Negators appear ~20% of the time a polarized word appears in a sentence.
- The algorithm cannot understand

Eg. :

Good

Great

Extraordinar
y

"The room could have been cleaner Carpet not vacummed and crumbs found around the video console area The location Stayed in January"

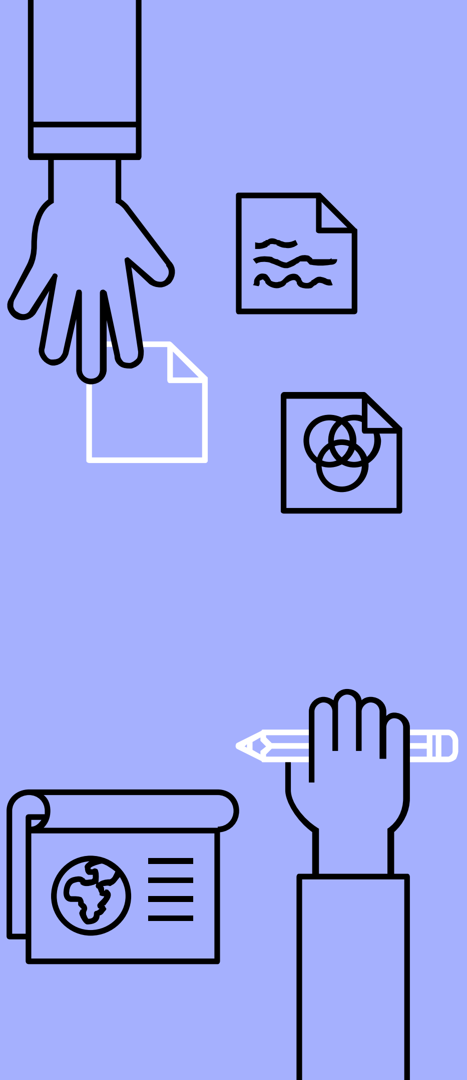
R Package: sentimentr

Update Date:

2018-01-16

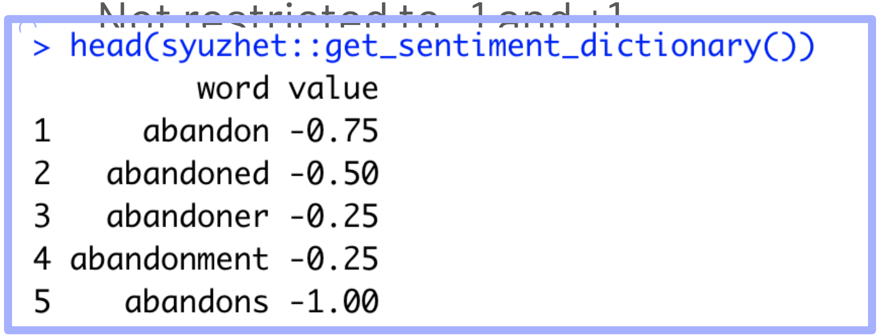
Description:

- Quickly calculate text polarity sentiment at the sentence level and optionally aggregate by rows or grouping variable(s)
- A dictionary lookup approach that tries to incorporate weighting for valence shifters

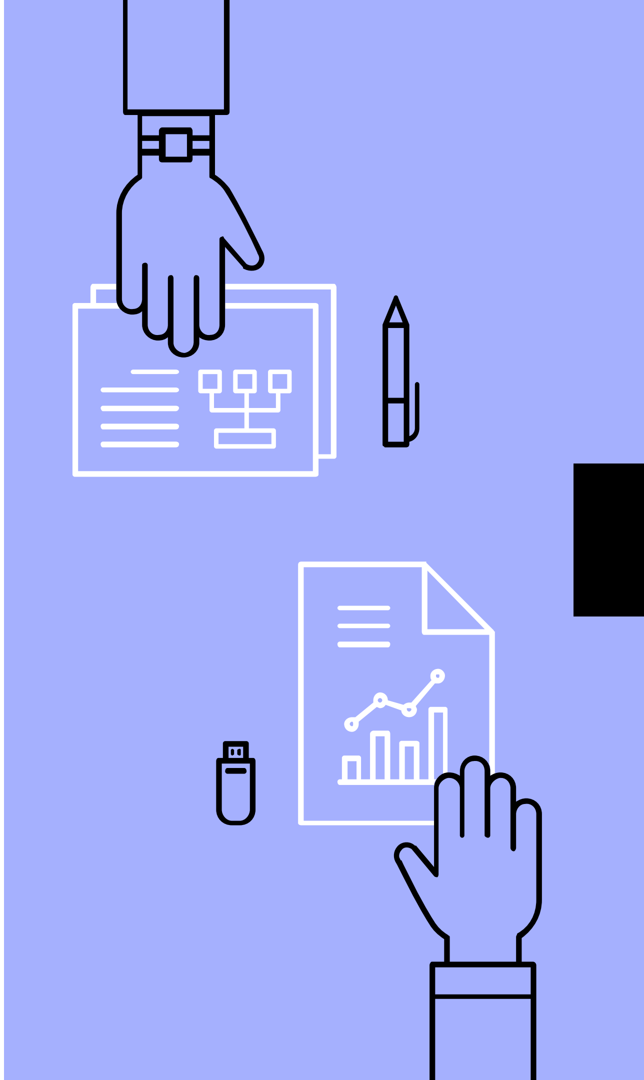


sentimentr: Dictionary

- Uses a polarity table of words and their weights
- Default polarity table is based on Jockers (2017) in syuzhet package.
  - You can create your own polarity table
  - Not restricted to -1 and +1

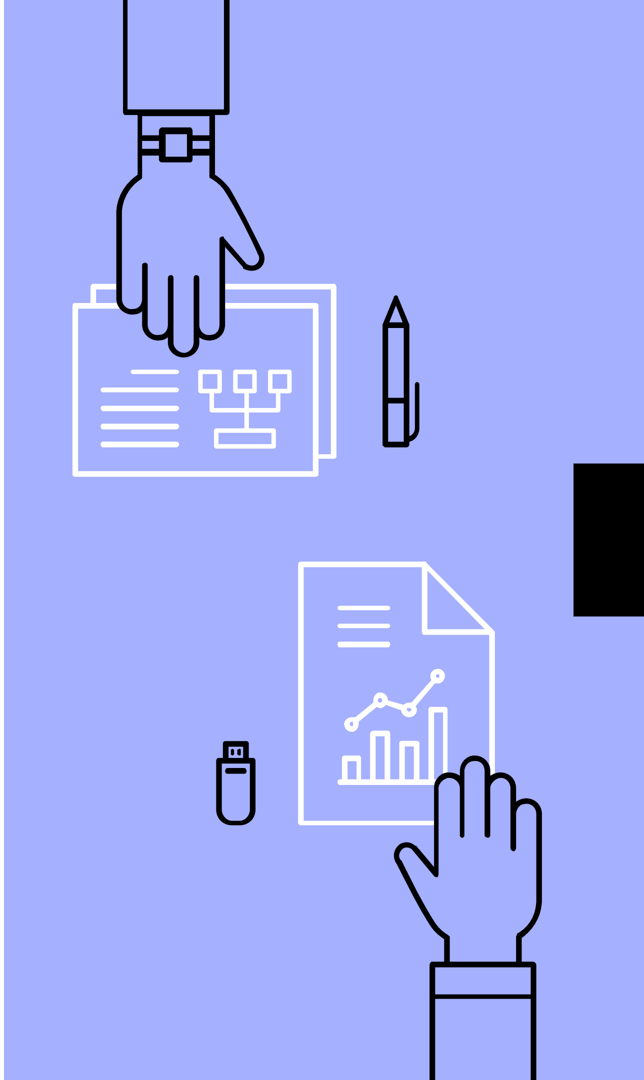


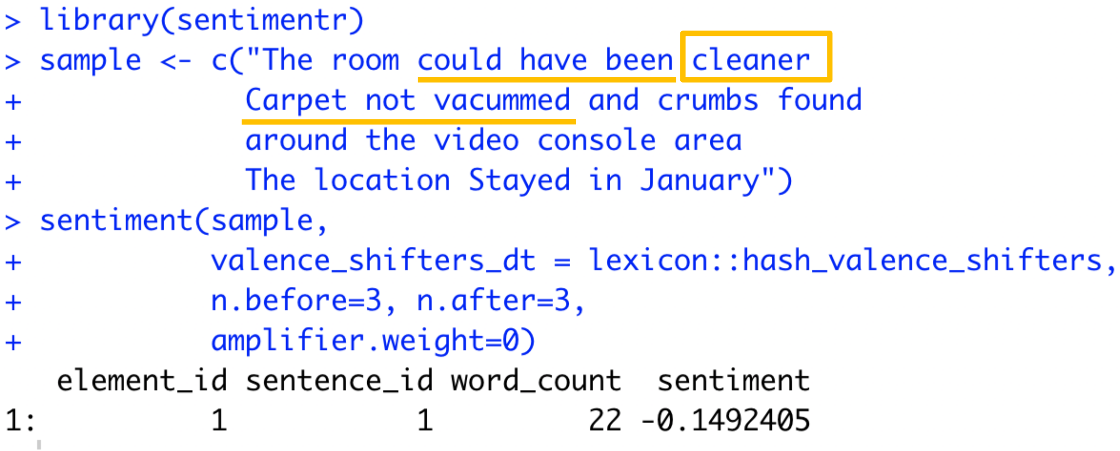
```
> head(syuzhet::get_sentiment_dictionary())
         word value
1     abandon -0.75
2   abandoned -0.50
3   abandoner -0.25
4 abandonment -0.25
5    abandons -1.00
```



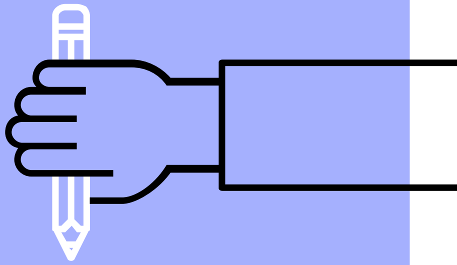
sentimentr: Valence Shifters

- Valence shifters alter or intensify the meaning of polarizing words
  - Includes negators, amplifiers, de-amplifiers, and adversative conjunctions
  - An *amplifier* (intensifier) increases the impact of a polarized word. (e.g., "I **really** like it.”)
  - A *de-amplifier* (downtoner) reduces the impact of a polarized word (e.g., "I **hardly** like it.")

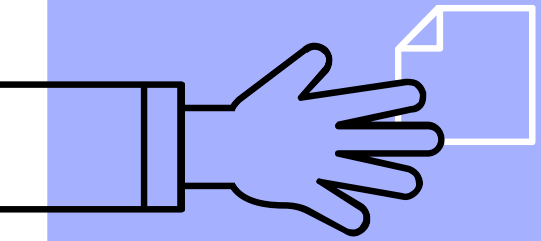




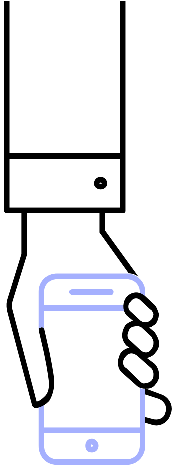
```
> library(sentimentr)
> sample <- c("The room could have been cleaner
+             Carpet not vacummed and crumbs found
+             around the video console area
+             The location Stayed in January")
> sentiment(sample,
+           valence_shifters_dt = lexicon::hash_valence_shifters,
+           n.before=3, n.after=3,
+           amplifier.weight=0)
   element_id sentence_id word_count  sentiment
1:          1           1         22 -0.1492405
```



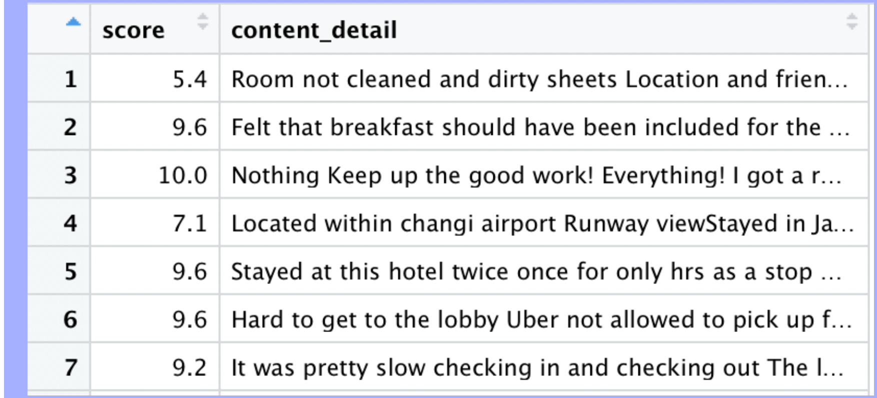
Supervised Sentiment Analysis



The Steps



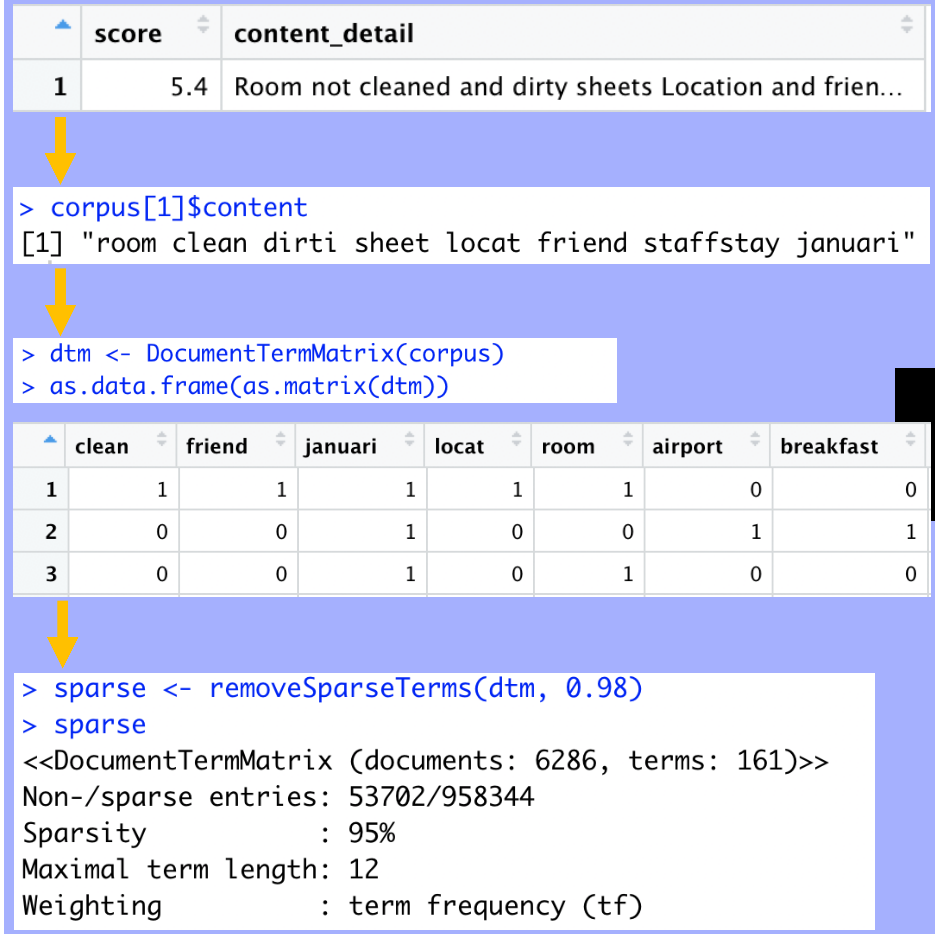
- Define the label
  - Classify scores as positive and negative
  - Use score directly
- Data Preprocessing
- Build a Bag-of-words linear model



	score	content_detail
1	5.4	Room not cleaned and dirty sheets Location and frien...
2	9.6	Felt that breakfast should have been included for the ...
3	10.0	Nothing Keep up the good work! Everything! I got a r...
4	7.1	Located within changi airport Runway viewStayed in Ja...
5	9.6	Stayed at this hotel twice once for only hrs as a stop ...
6	9.6	Hard to get to the lobby Uber not allowed to pick up f...
7	9.2	It was pretty slow checking in and checking out The l...

Data Preprocessing

- R Package: tm
- Inputs: content_detail
- Prepare corpus
  - Lower-casing
  - Removing punctuations, stopwords, whitespace
  - Stemming
- Covert to document term matrix
- Remove unimportant terms



	score	content_detail
1	5.4	Room not cleaned and dirty sheets Location and frien...

```
> corpus[1]$content
[1] "room clean dirti sheet locat friend staffstay januari"
```

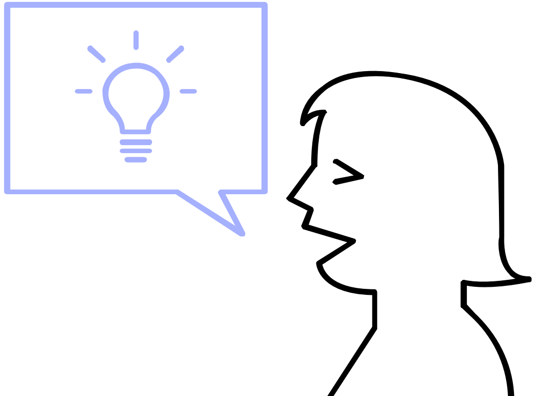
```
> dtm <- DocumentTermMatrix(corpus)
> as.data.frame(as.matrix(dtm))
```

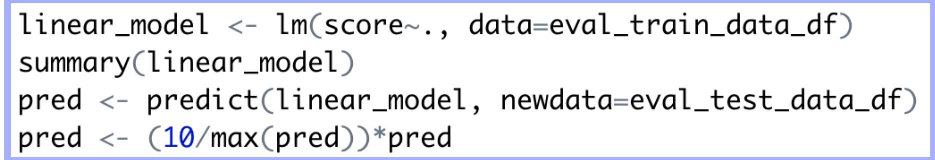
	clean	friend	januari	locat	room	airport	breakfast
1	1	1	1	1	1	0	0
2	0	0	1	0	0	1	1
3	0	0	1	0	1	0	0

```
> sparse <- removeSparseTerms(dtm, 0.98)
> sparse
<<DocumentTermMatrix (documents: 6286, terms: 161)>>
Non-/sparse entries: 53702/958344
Sparsity           : 95%
Maximal term length: 12
Weighting          : term frequency (tf)
```

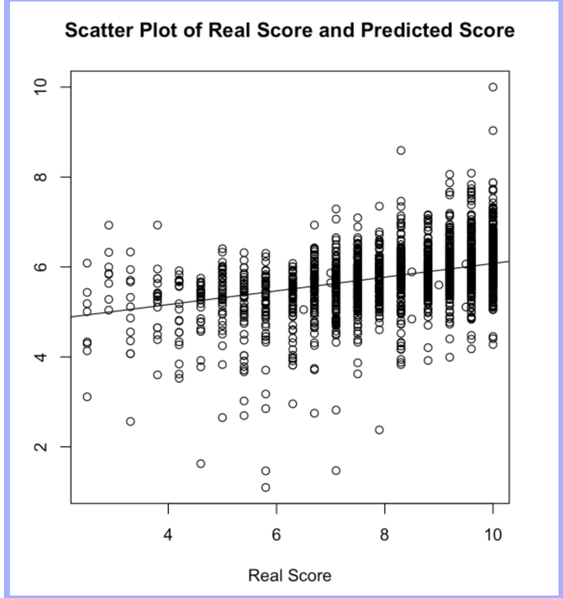
Linear Regression

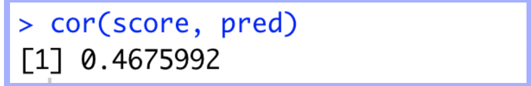
- Split into train and test data
- Build a linear regression model
- Predict test scores and normalization
- Comparison





```
linear_model <- lm(score~., data=eval_train_data_df)
summary(linear_model)
pred <- predict(linear_model, newdata=eval_test_data_df)
pred <- (10/max(pred))*pred
```

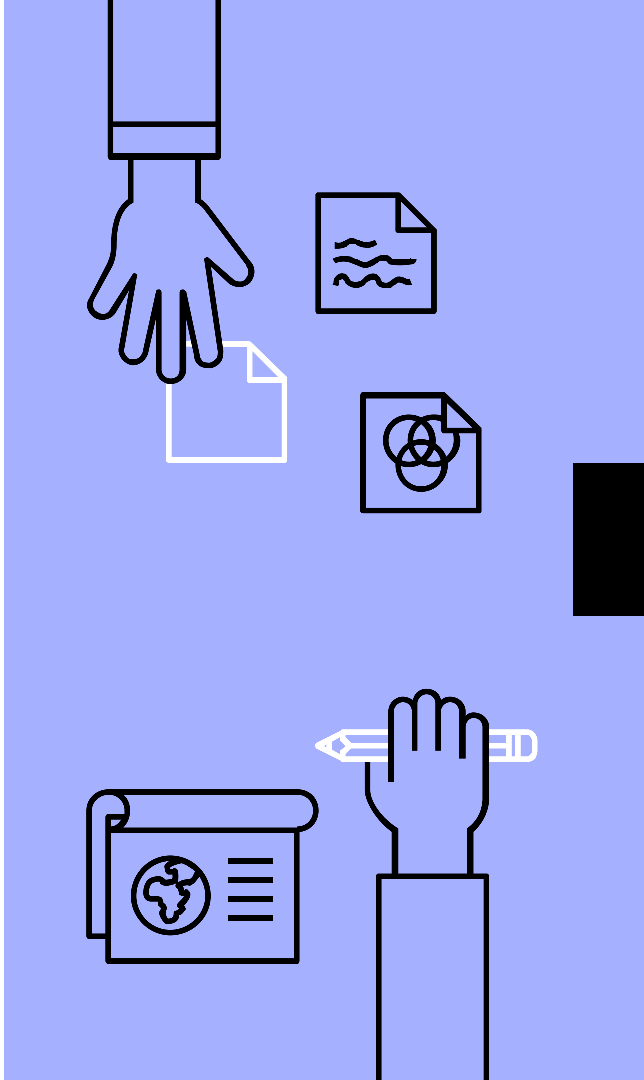


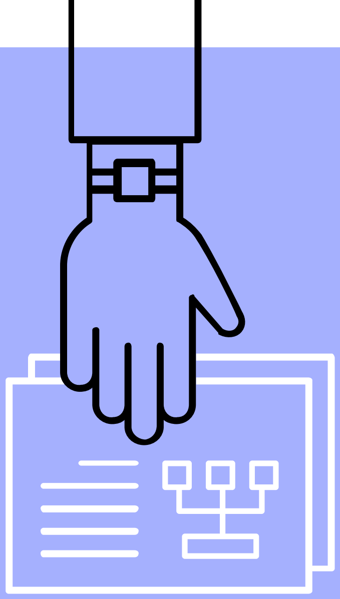


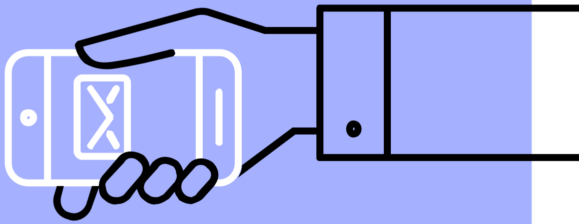
```
> cor(score, pred)
[1] 0.4675992
```

Remarks

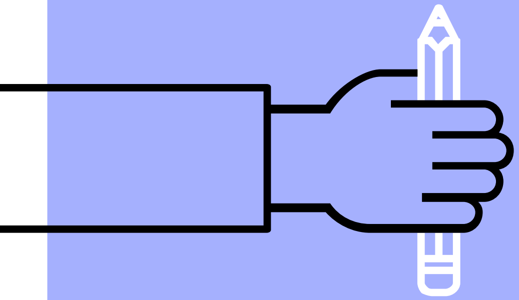
- Limiting factors
  - Lack to training data
  - Score and sentiment are not perfectly correlated
  - Model is simple
  - Linguistics is complicated
- Further improvement
  - Give only positive or negative label to data
  - More complicated algorithms based on larger data set

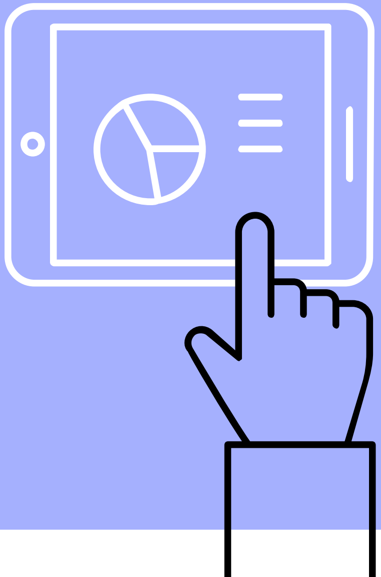






Topic modeling





Topic modeling

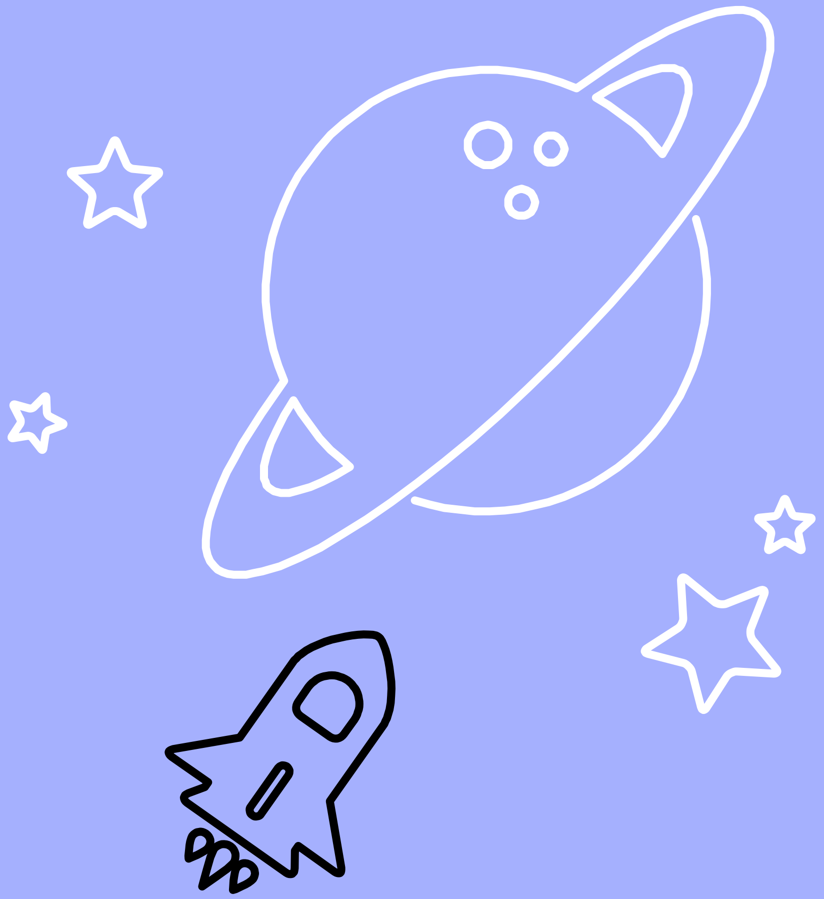
® **Clustering** method

® **Latent Dirichlet allocation (LDA):**

- treats each document as a mixture of topics

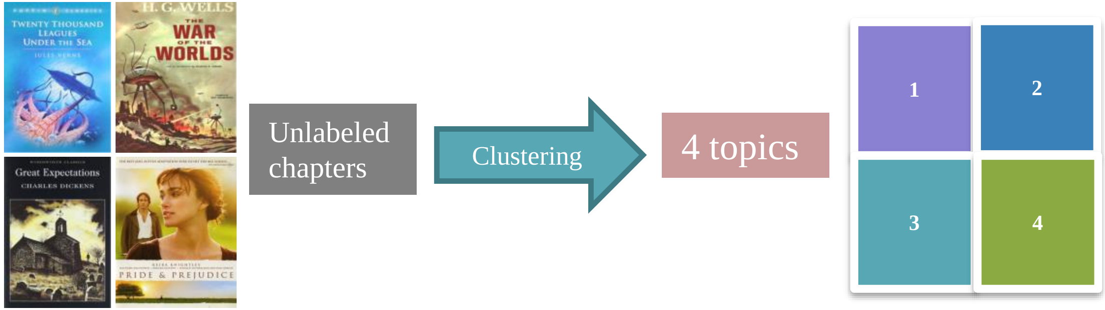
e.g.:in a two-topic model we could say "Document 1 is 90% topic A and 10% topic B, while Document 2 is 30% topic A and 70% topic B.

- treats each topic as a mixture of words



Objective:

- Four novels with chapters unlabeled: we don’t know what words might distinguish them into groups. We’ll thus use topic modeling to discover how chapters cluster into distinct topics, each of them (presumably) representing one of the books
- Four novels: Twenty Thousand Leagues under the Sea / The War of the Worlds / Pride and Prejudice / Great Expectations



Unlabeled chapters

Clustering

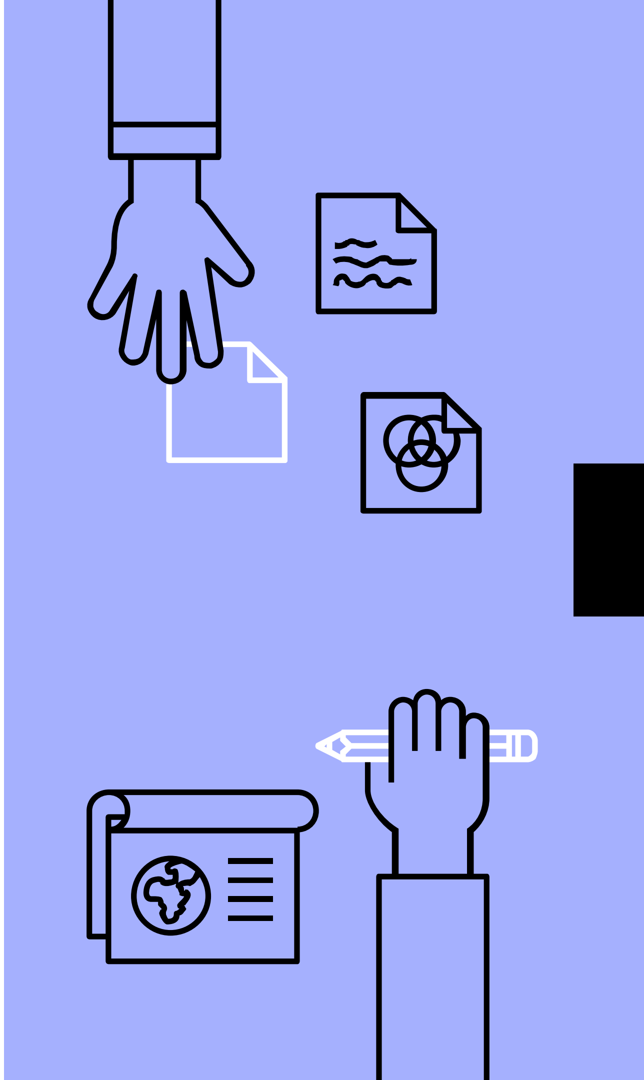
4 topics

1

2

3

4



Packages

gutenbergr: loading books

topicmodels: modeling topics

stringr: deal with string

dplyr: operations on tables or dataframe

wordcloud2: draw word cloud

ggplot2: draw pictures

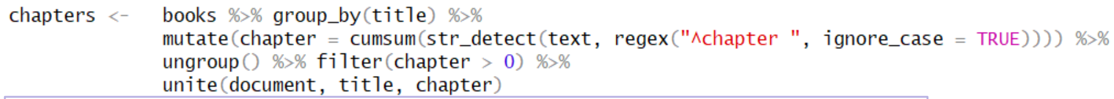
tidytext/tidyr: tidying model objects

STEPS

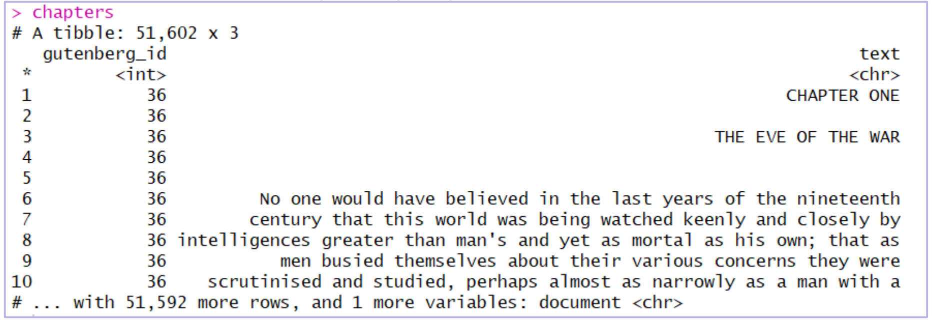
- Load the books -> Split into chapters
- Split into words -> remove stop words -> word counts
- LDA on chapters (clustering topics)
- Per-document classification (check the clustering result)

STEPS

1. Load the books -> split into chapters

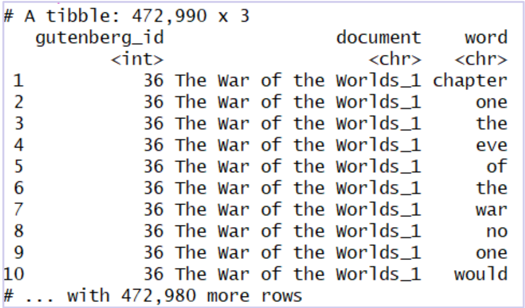


```
chapters <-   books %>% group_by(title) %>%
              mutate(chapter = cumsum(str_detect(text, regex("^chapter ", ignore_case = TRUE)))) %>%
              ungroup() %>% filter(chapter > 0) %>%
              unite(document, title, chapter)
```

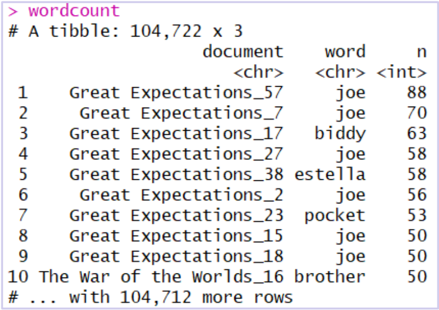


```
> chapters
# A tibble: 51,602 x 3
   gutenberg_id                                                    text
 *        <int>                                                   <chr>
 1           36                                             CHAPTER ONE
 2           36
 3           36                                         THE EVE OF THE WAR
 4           36
 5           36
 6           36        No one would have believed in the last years of the nineteenth
 7           36       century that this world was being watched keenly and closely by
 8           36 intelligences greater than man's and yet as mortal as his own; that as
 9           36          men busied themselves about their various concerns they were
10           36    scrutinised and studied, perhaps almost as narrowly as a man with a
# ... with 51,592 more rows, and 1 more variables: document <chr>
```

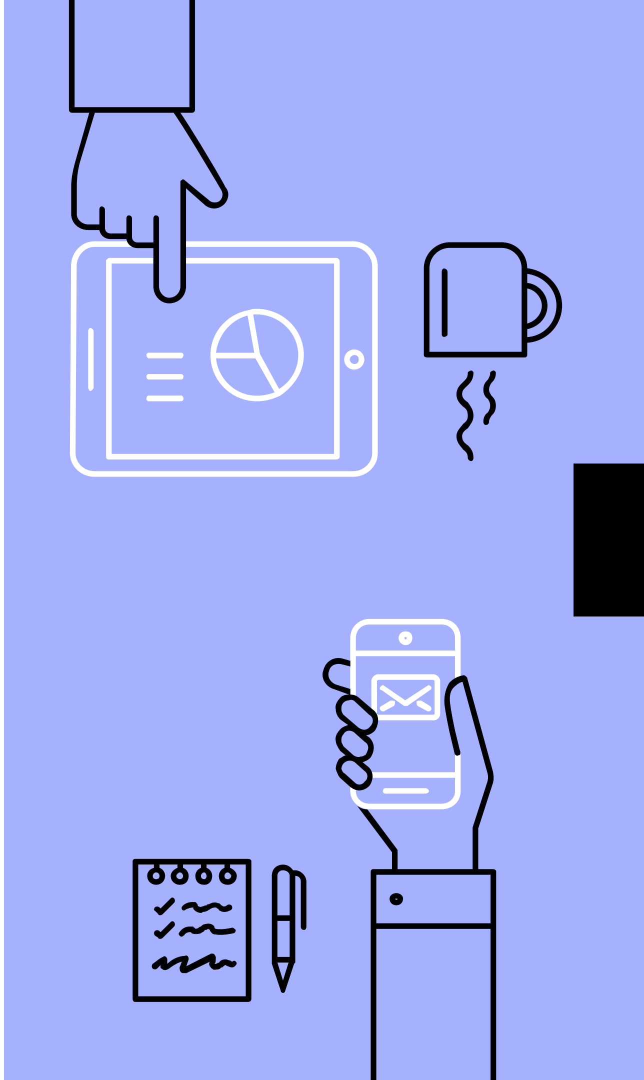
2. Split into words / remove stop words/ word counts



```
# A tibble: 472,990 x 3
  gutenberg_id                document    word
         <int>                   <chr>   <chr>
1           36 The War of the Worlds_1 chapter
2           36 The War of the Worlds_1     one
3           36 The War of the Worlds_1     the
4           36 The War of the Worlds_1     eve
5           36 The War of the Worlds_1      of
6           36 The War of the Worlds_1     the
7           36 The War of the Worlds_1     war
8           36 The War of the Worlds_1      no
9           36 The War of the Worlds_1     one
10          36 The War of the Worlds_1   would
# ... with 472,980 more rows
```

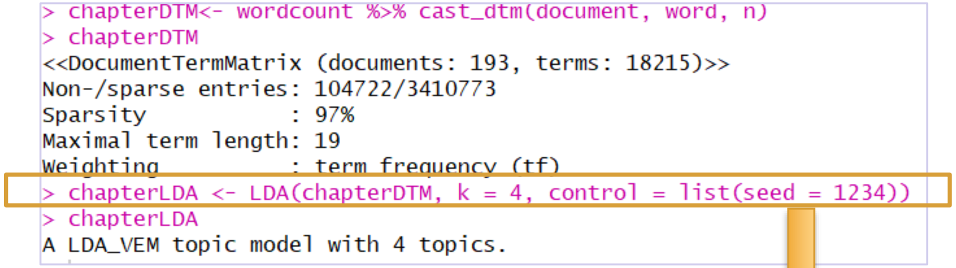


```
> wordcount
# A tibble: 104,722 x 3
                    document    word     n
                       <chr>   <chr> <int>
 1    Great Expectations_57     joe    88
 2     Great Expectations_7     joe    70
 3    Great Expectations_17   biddy    63
 4    Great Expectations_27     joe    58
 5    Great Expectations_38 estella    58
 6     Great Expectations_2     joe    56
 7    Great Expectations_23  pocket    53
 8    Great Expectations_15     joe    50
 9    Great Expectations_18     joe    50
10 The War of the Worlds_16 brother    50
# ... with 104,712 more rows
```



STEPS

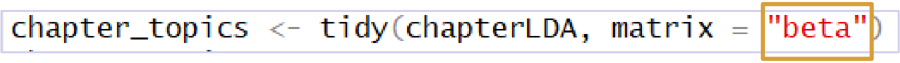
3. LDA on chapters (clustering 4 topics):



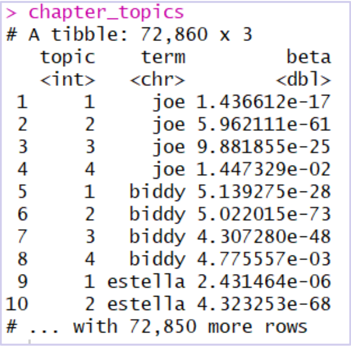
```
> chapterDTM<- wordcount %>% cast_dtm(document, word, n)
> chapterDTM
<<DocumentTermMatrix (documents: 193, terms: 18215)>>
Non-/sparse entries: 104722/3410773
Sparsity           : 97%
Maximal term length: 19
Weighting          : term frequency (tf)
> chapterLDA <- LDA(chapterDTM, k = 4, control = list(seed = 1234))
> chapterLDA
A LDA_VEM topic model with 4 topics.
```

Use wordcounts in each chapter to model 4 topics

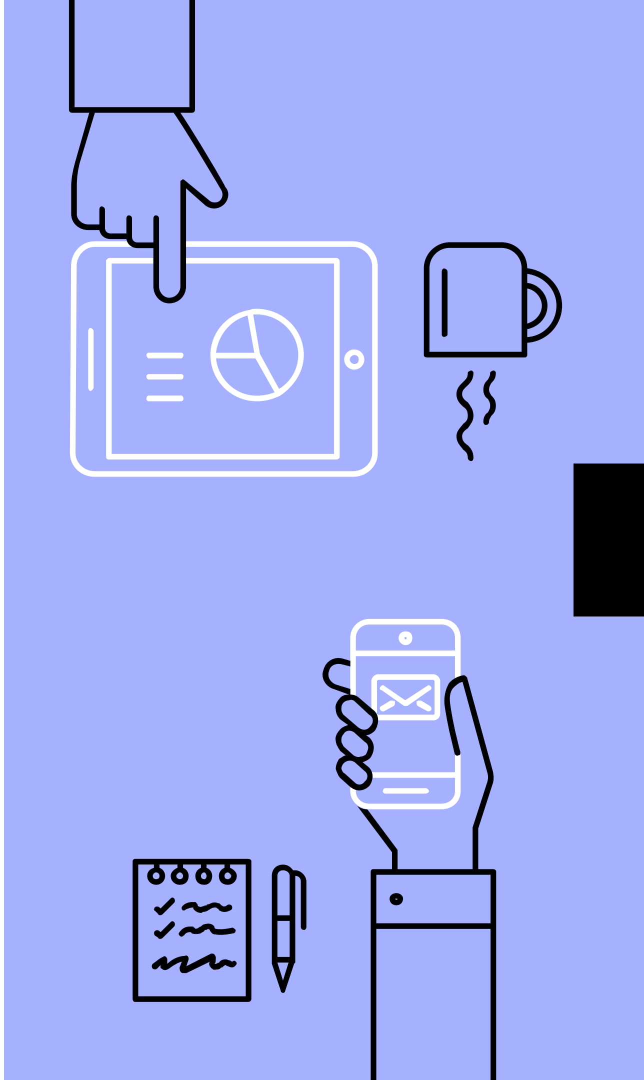
- Find per-topic-per-word probabilities (beta)



```
chapter_topics <- tidy(chapterLDA, matrix = "beta")
```

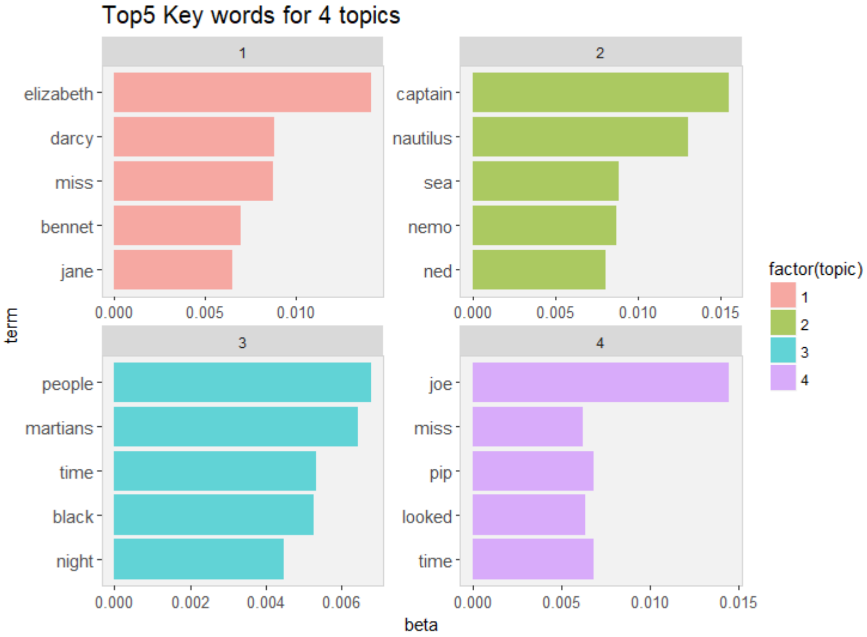


```
> chapter_topics
# A tibble: 72,860 x 3
   topic    term         beta
   <int>   <chr>        <dbl>
 1     1     joe 1.436612e-17
 2     2     joe 5.962111e-61
 3     3     joe 9.881855e-25
 4     4     joe 1.447329e-02
 5     1   biddy 5.139275e-28
 6     2   biddy 5.022015e-73
 7     3   biddy 4.307280e-48
 8     4   biddy 4.775557e-03
 9     1 estella 2.431464e-06
10     2 estella 4.323253e-68
# ... with 72,850 more rows
```

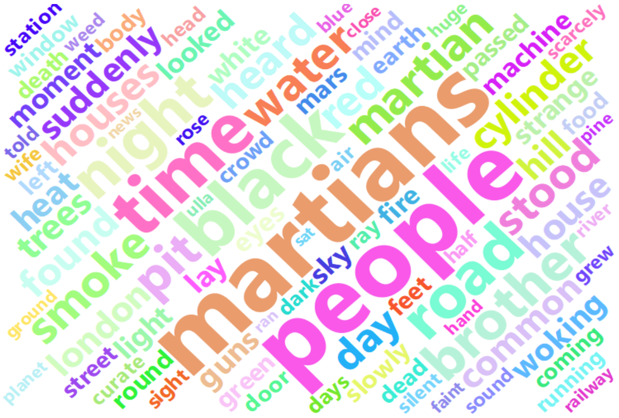


STEPS

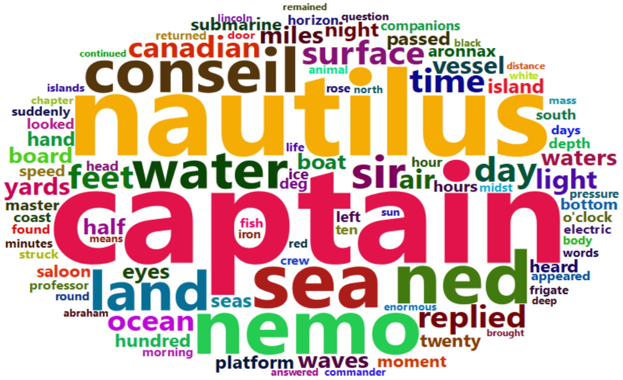
- Find **Top 5 keywords** for each topic:



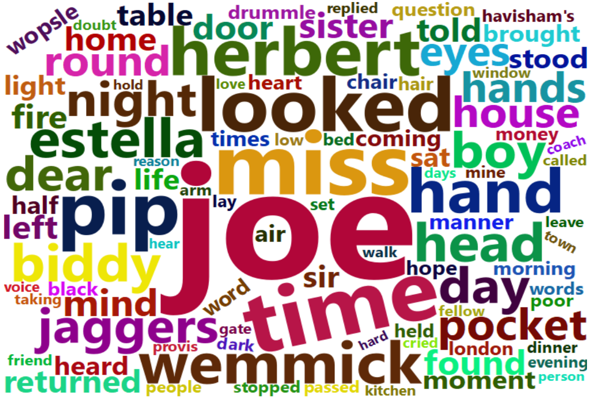
[redacted]



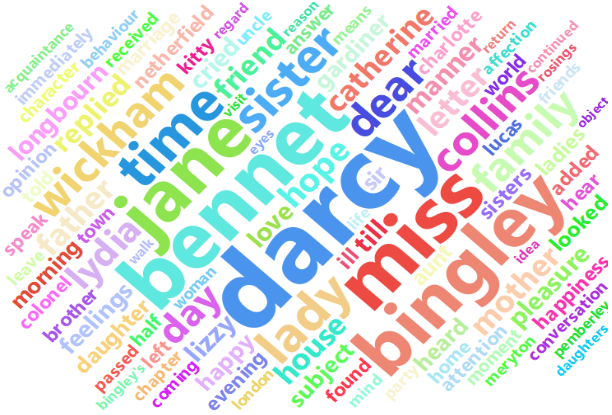
The War of the Worlds



Twenty Thousand Leagues under the Sea



Great Expectations

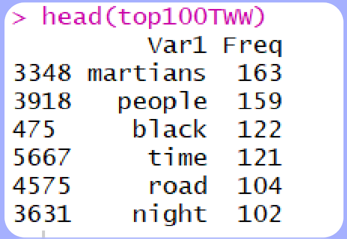


Pride and Prejudice

Wordcloud for top 100 words in four novels

Using pacakage "wordcloud2"

The input is data frame including word and frequency:



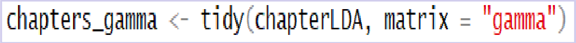
```
> head(top100TWW)
          Var1 Freq
3348 martians  163
3918   people  159
475     black  122
5667     time  121
4575     road  104
3631    night  102
```

>wordcloud2(top100TWW,...)

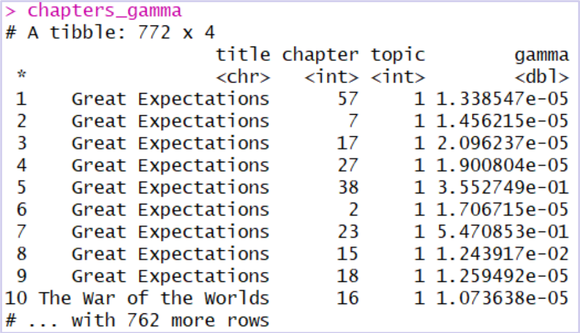
STEPS

4. Per-document classification

4.1 find per-document-per-topic probabilities (gamma)

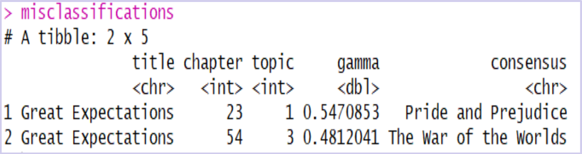


```
chapters_gamma <- tidy(chapterLDA, matrix = "gamma")
```



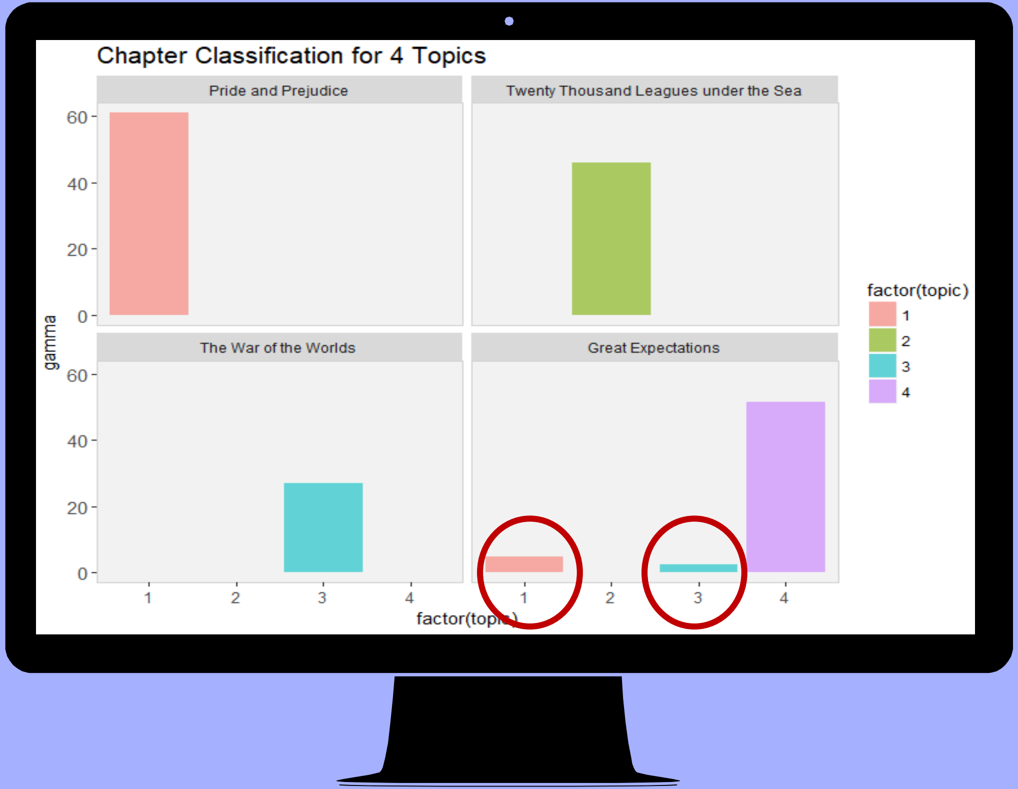
```
> chapters_gamma
# A tibble: 772 x 4
                  title chapter topic        gamma
 *                <chr>   <int> <int>        <dbl>
 1     Great Expectations      57     1 1.338547e-05
 2     Great Expectations       7     1 1.456215e-05
 3     Great Expectations      17     1 2.096237e-05
 4     Great Expectations      27     1 1.900804e-05
 5     Great Expectations      38     1 3.552749e-01
 6     Great Expectations       2     1 1.706715e-05
 7     Great Expectations      23     1 5.470853e-01
 8     Great Expectations      15     1 1.243917e-02
 9     Great Expectations      18     1 1.259492e-05
10 The War of the Worlds      16     1 1.073638e-05
# ... with 762 more rows
```

4.3 find the misclassified chapters



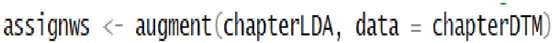
```
> misclassifications
# A tibble: 2 x 5
               title chapter topic     gamma            consensus
               <chr>   <int> <int>     <dbl>                <chr>
1 Great Expectations      23     1 0.5470853   Pride and Prejudice
2 Great Expectations      54     3 0.4812041 The War of the Worlds
```

4.2 find classification on topics for all chapters



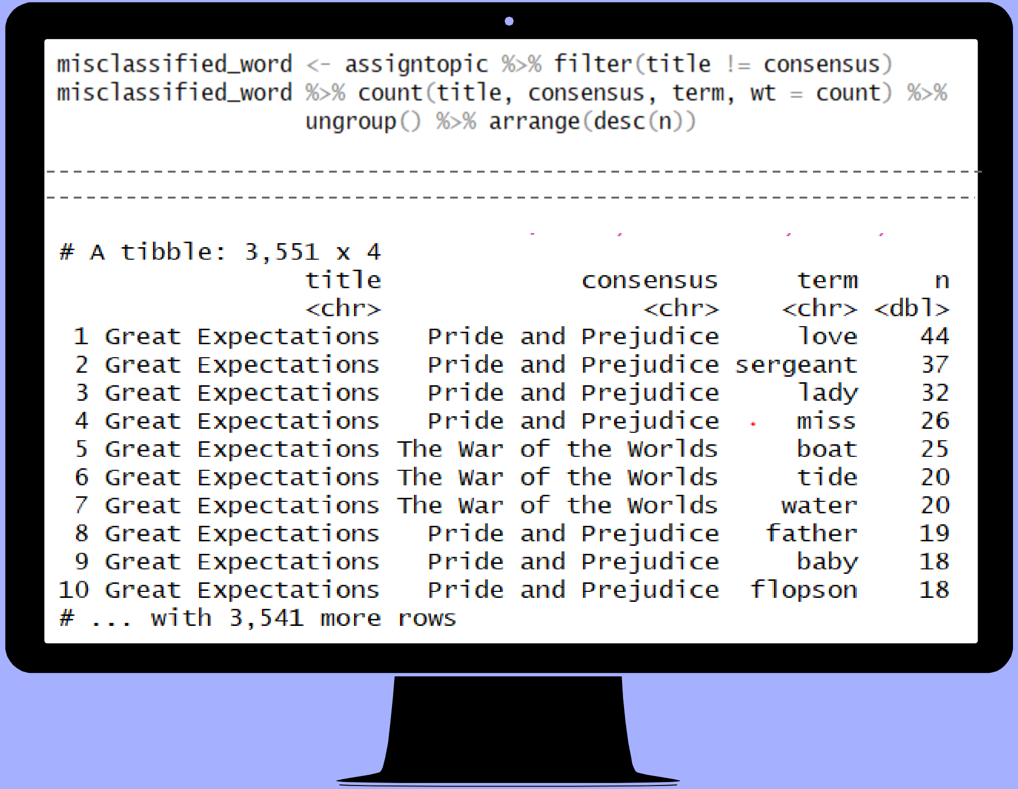
STEPS

4.4 Assgin each word in documents to topics



```
assignws <- augment(chapterLDA, data = chapterDTM)
```

4.5 Find words which are most easily to be misclassified



```
misclassified_word <- assigntopic %>% filter(title != consensus)
misclassified_word %>% count(title, consensus, term, wt = count) %>%
                      ungroup() %>% arrange(desc(n))

------------------------------------------------------------------------------
------------------------------------------------------------------------------

# A tibble: 3,551 x 4
                title               consensus     term     n
                <chr>                   <chr>    <chr> <dbl>
 1 Great Expectations   Pride and Prejudice     love    44
 2 Great Expectations   Pride and Prejudice sergeant    37
 3 Great Expectations   Pride and Prejudice     lady    32
 4 Great Expectations   Pride and Prejudice     miss    26
 5 Great Expectations The War of the Worlds     boat    25
 6 Great Expectations The War of the Worlds     tide    20
 7 Great Expectations The War of the Worlds    water    20
 8 Great Expectations   Pride and Prejudice   father    19
 9 Great Expectations   Pride and Prejudice     baby    18
10 Great Expectations   Pride and Prejudice  flopson    18
# ... with 3,541 more rows
```

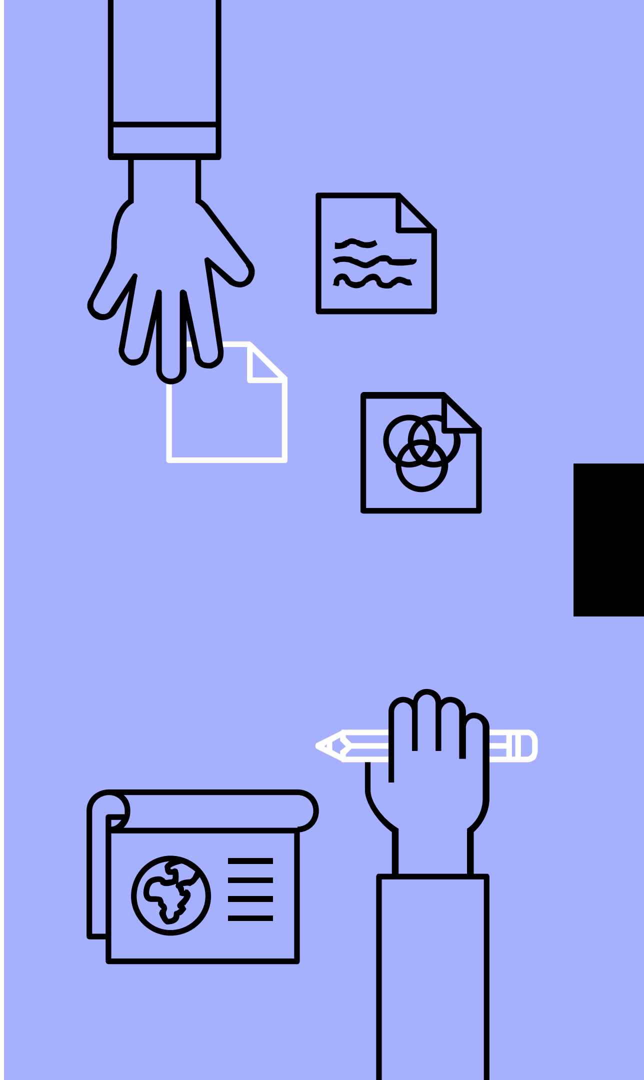
Summary

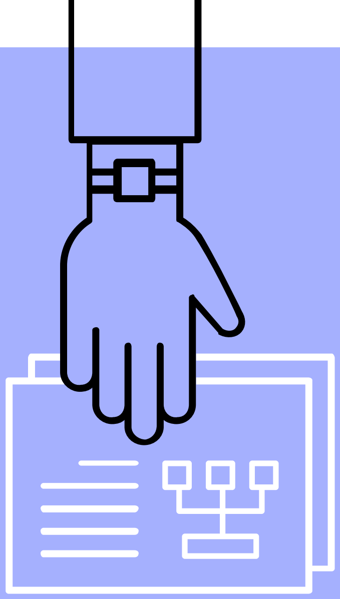
- Sentiment Analysis

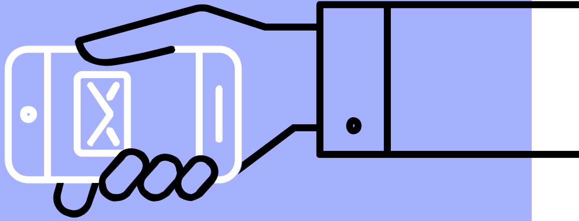
  - unsupervised: dictionary-based

  - supervised (text with label): classification

- Topic modeling







Thank you!