WEEK 3: TEXT MINING I

SECU0050

BENNETT KLEINBERG

30 JAN 2020

Data Science for Crime Scientists

WEEK 3: TEXT MINING 1

TODAY

- text quantification
- numerical representations
- common text metrics


every MINUTE of the DAY
Skype USERS MAKE
110,040 CALLS
UBER PASSENGERS TAKE 694 RIDES
Facebook USERS LIKE 4,166,667 POSTS
TWITTER USERS SEND 347,222 TWEETS
YOUTUBE USERS UPLOAD 300 HOURS OF NEW VIDEO
Buzzfeed USERS VIEW 34,150 VIDEOS
SNAPCHAT USERS SHARE 284,722 SNAPS
Instagram USERS LIKE 1,736,111 PHOTOS
Tinder USERS SWIPE 590,278 TIMES
PINTEREST USERS PIN 9,722 IMAGES
VINE USERS PLAY 1,041,666 VIDEOS
AMAZON RECEIVES 4,310 UNIQUE VISITORS
reddit USERS CAST 18,327 VOTES
Netflix SUBSCRIBERS STREAM 77,160 HOURS OF VIDEO
APPLE USERS DOWNLOAD 51,000 APPS

TEXT IS EVERYWHERE …

- Practically all websites
- Emails
- Messaging
- Government reports
- Laws
- Police reports
- Uni coursework
- Newspapers

… AND EVERYTHING IS TEXT

- videos –> transcripts
- music –> lyrics
- conversations –> transcripts
- speeches –> transcripts

CORE IDEA

Text is a unique documentation of human activity.

We are obsessed with documenting.

TEXT EXAMPLE

How would you analyse the text?

EXAMPLE

All I ever wanted was to love women, and in turn to be loved by them back. Their behavior towards me has only earned my hatred, and rightfully so! I am the true victim in all of this. I am the good guy. Humanity struck at me first by condemning me to experience so much suffering. I didn't ask for this. I didn't want this. I didn't start this war. I wasn't the one who struck first. But I will finish it by striking back. I will punish everyone. And it will be beautiful. Finally, at long last, I can show the world my true worth.

CHALLENGE OF QUANTIFICATION

- text is not a numerical representation
- compare this to trading data, crime statistics, etc.
- text is just that, “a text”
- but: for quantitative analyses, we need numbers

Text –> numerical representation?

FEATURES OF TEXT DATA

- meta dimension
- syntactic dimension
- semantic dimension
- text metrics

META DIMENSION

- no. of words
- no. of sentences

SYNTACTIC DIMENSION

- word frequencies
- verbs, nouns, persons, locations, ..
- structure of a sentence

SEMANTIC DIMENSION

- sentiment
- psycholinguistic features

TEXT METRICS

- readability
- lexical diversity

THE `QUANTEDA` PACKAGE

```
library(quanteda)
```

- quanteda: Quantitative Analysis of Textual Data
  - documentation
  - tutorials
  - examples

LEVELS OF TEXT DATA

- characters `c('h', 'a', 't', 'r', 'e', 'd')`
- words `hatred`
- sentences `I didn’t ask for this.`
- documents: individual text files
- corpora: collection of documents

COUNTING META FEATURES IN R

text level	R function
characters	`nchar()`
words	`quanteda::ntoken()`
sentences	`quanteda::nsentence()`

R EXAMPLES

```r
#sentences
no_of_sentences = nsentence(er)
no_of_sentences
```

```
## text1
##    13
```

```r
#words 1
no_of_words_1 = ntoken(er)
no_of_words_1
```

```
## text1
##   123
```

```r
#words 2
no_of_words_2 = ntype(er)
no_of_words_2
```

```
## text1
##    72
```

TYPE-TOKE RATIO

Note: often used metric for “lexical diversity” is the TTR (type-token ratio).

```
string_a = "I didn’t ask for this. I didn’t want this."
string_b = "But I will finish it by striking back."
```

What are the type-token ratios of each string?

TYPE-TOKEN RATIO

```
ntype(string_a)/ntoken(string_a)
```

```
##     text1
## 0.6363636
```

```
ntype(string_b)/ntoken(string_b)
```

```
## text1
##     1
```

CHARACTERS PER WORD

```
nchar(er)/ntoken(er)
```

```
##    text1
## 4.317073
```

WORDS PER SENTENCE

```
ntoken(er)/nsentence(er)
```

```
##    text1
## 9.461538
```

TEXT REPRESENTATION

AIM

- representing a text by its tokens (terms)
- each text consists of a frequency of its tokens

```
"I think I believe him"
```

TEXT REPRESENTATION BY HAND

1. create a column for each token
2. count the frequency

text_id	I	think	believe	him
text1	2	1	1	1

TERM FREQUENCY

- frequency of each token in each document
- represented in a table (matrix)
- tokens are features of a document
- voilá: **Document Feature Matrix (= DFM)**

```
example_string_tok = tokens("I think I believe him")
```

DFM IN `QUANTEDA`

- from 'tokens' object, create a DFM table

```
dfm(example_string_tok)
```

```
## Document-feature matrix of: 1 document, 4 features (0% sparse).
## 1 x 4 sparse Matrix of class "dfm"
##          features
## docs     i think believe him
##   text1 2     1       1   1
```

SPARSITY

Sparsity = % of zero-cells

- Why is sparsity = 0% here?
- What would you expect if we take additional documents, and why?

DFM WITH MULTIPLE DOCUMENTS

```
multiple_docs_tok = tokens(c("I think I believe him", "This is a cool fu
```

```
dfm(multiple_docs_tok)
```

```
## Document-feature matrix of: 2 documents, 9 features (50% sparse).
## 2 x 9 sparse Matrix of class "dfm"
##        features
## docs    i think believe him this is a cool function
##   text1 2     1       1   1    0  0 0    0        0
##   text2 0     0       0   0    1  1 1    1        1
```

DFM WITH TWO LONE-ACTORS

All I ever wanted was to love women, and in turn to be loved by them back. Their behavior towards me has only earned my hatred, and rightfully so! I am the true victim in all of this. I am the good guy. Humanity struck at me first by condemning me to experience so much suffering. I didn't ask for this. I didn't want this. I didn't start this war. I wasn't the one who struck first. But I will finish it by striking back. I will punish everyone. And it will be beautiful. Finally, at long last, I can show the world my true worth.

DFM WITH TWO TEXTS

The Industrial Revolution and its consequences have been a disaster for the human race. They have greatly increased the life-expectancy of those of us who live in “advanced” countries, but they have destabilized society, have made life unfulfilling, have subjected human beings to indignities, have led to widespread psychological suffering (in the Third World to physical suffering as well) and have inflicted severe damage on the natural world. The continued development of technology will worsen the situation.

USING THE CORPUS METHOD

- Create a “mini corpus” for convenience
- makes using the quanteda pipeline easier

```
mini_corpus = corpus(c(er, ub))
summary(mini_corpus)
```

```
## Corpus consisting of 2 documents:
##
##    Text Types Tokens Sentences
##   text1    72    123        13
##   text2    63     88         3
##
## Source: /Users/bennettkleinberg/GitHub/UCL_SECU0050/lectures/week_3/*
## Created: Thu Jan 30 06:38:03 2020
## Notes:
```

DFM REPRESENTATION

```
corpus_tokenised = tokens(mini_corpus)
corpus_dfm = dfm(corpus_tokenised)
```

document	all	i	ever	wanted	was	to	love	women
text1	2	10	1	1	1	3	1	1
text2	0	0	0	0	0	3	0	0

document	am	the	true	victim	of	this	good	guy
text1	2	4	2	1	1	4	1	1
text2	0	7	0	0	3	0	0	0

Is this ideal?

WHAT ARE THE MOST FREQUENT "TERMS"?

```
topfeatures(corpus_dfm[1])
```

```
##       .      i      ,    the   this     to    and     by     me didn't
##      12     10      4      4      4      3      3      3      3      3
```

```
topfeatures(corpus_dfm[2])
```

```
##       the      have         ,        to         .        of       and
##         7         7         4         3         3         3         2
##        in suffering     world
##         2         2         2
```

ZIPF'S LAW


THE OF AND TO A IN IS I THAT IT FOR YOU WAS WITH ON AS HAVE BUT BE THEY
0:35 / 21:04
The Zipf Mystery
13,315,384 views
340K
3.7K
SHARE
SAVE

WORD HIERARCHIES

- some words add more meaning than others
- stopwords = meaningless (?)
- in any case: too frequent words, don’t tell much about the documents
- ideally: we want to get an *importance score* for each word

WORD IMPORTANCE

document	and	in	turn	be	loved	by
text1	3	2	1	2	1	3
text2	2	2	0	0	0	0

We want to “reward” words that are:

- important locally (for individual documents)
- but not ‘inflated’ globally (for the whole corpus)

METRIC FOR WORD IMPORTANCE: TERM FREQUENCY

- $tf(t, d) = \frac{\#_{t,d}}{nwords_d}$ (proportion)
- $tf(t, d)$ = #_(t, d) (count)

PROPORTIONS

document	and	in	turn	be	loved	by
text1	0.024	0.016	0.008	0.016	0.008	0.024
text2	0.023	0.023	0.000	0.000	0.000	0.000

COUNTS

document	and	in	turn	be	loved	by
text1	3	2	1	2	1	3
text2	2	2	0	0	0	0

DOUBLE-CHECK THE FORMULAE

- $tf(t, d) = \frac{\#_{t,d}}{nwords_d}$
- $tf(t, d)$ = #_(t, d)

```
3/ntoken(mini_corpus[1])
```

```
##      text1
## 0.02439024
```

Term frequency: *reward for words that occur often in a document*

PROBLEM

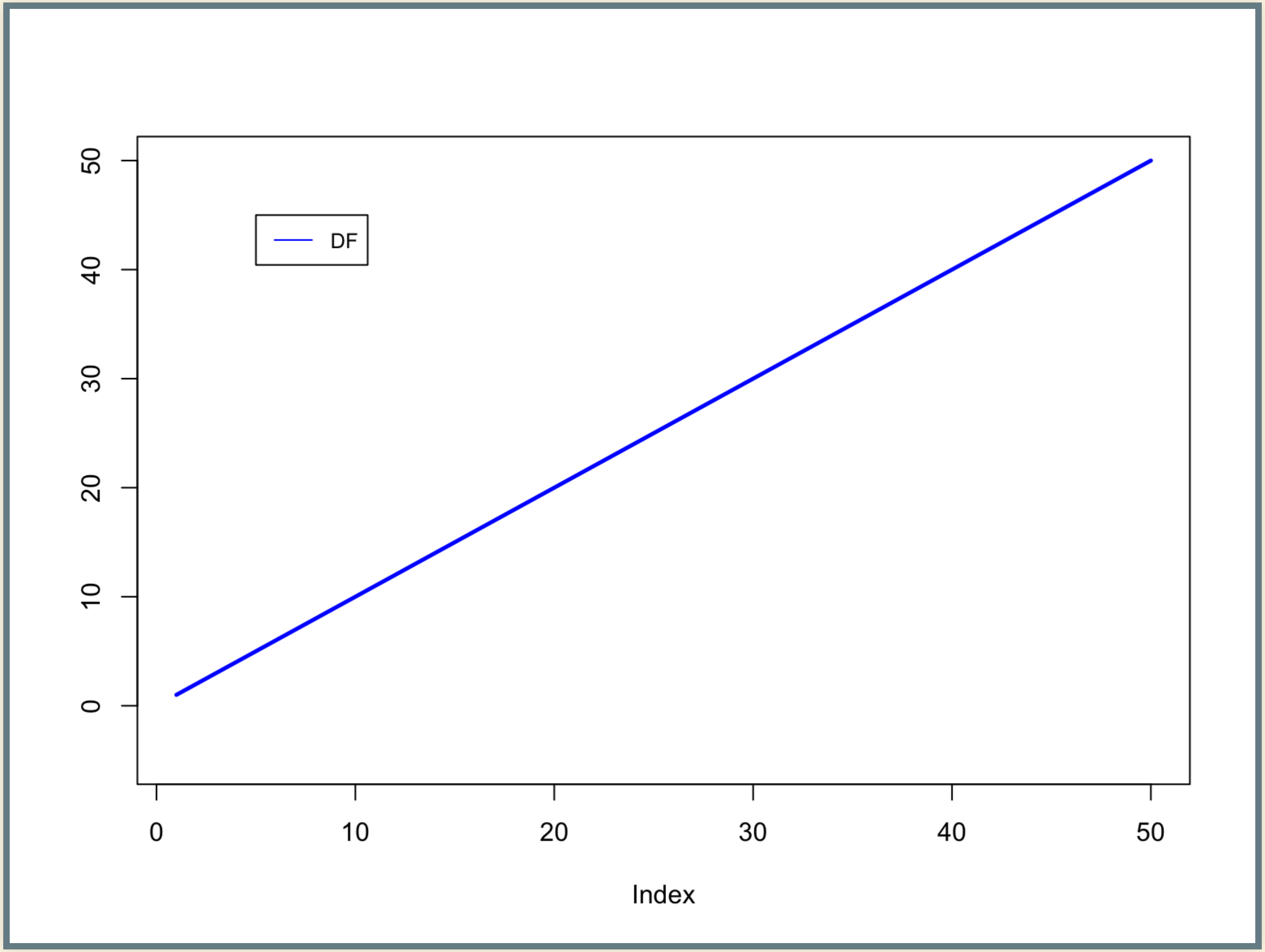
Some words just occur a lot anyway (e.g. “stop words”).

Global occurrence: document frequency

	x
and	2
in	2
turn	1
be	1
loved	1
by	1

$df(t)$ = # of docs with t

DF



CORRECTING FOR TERM INFLATION

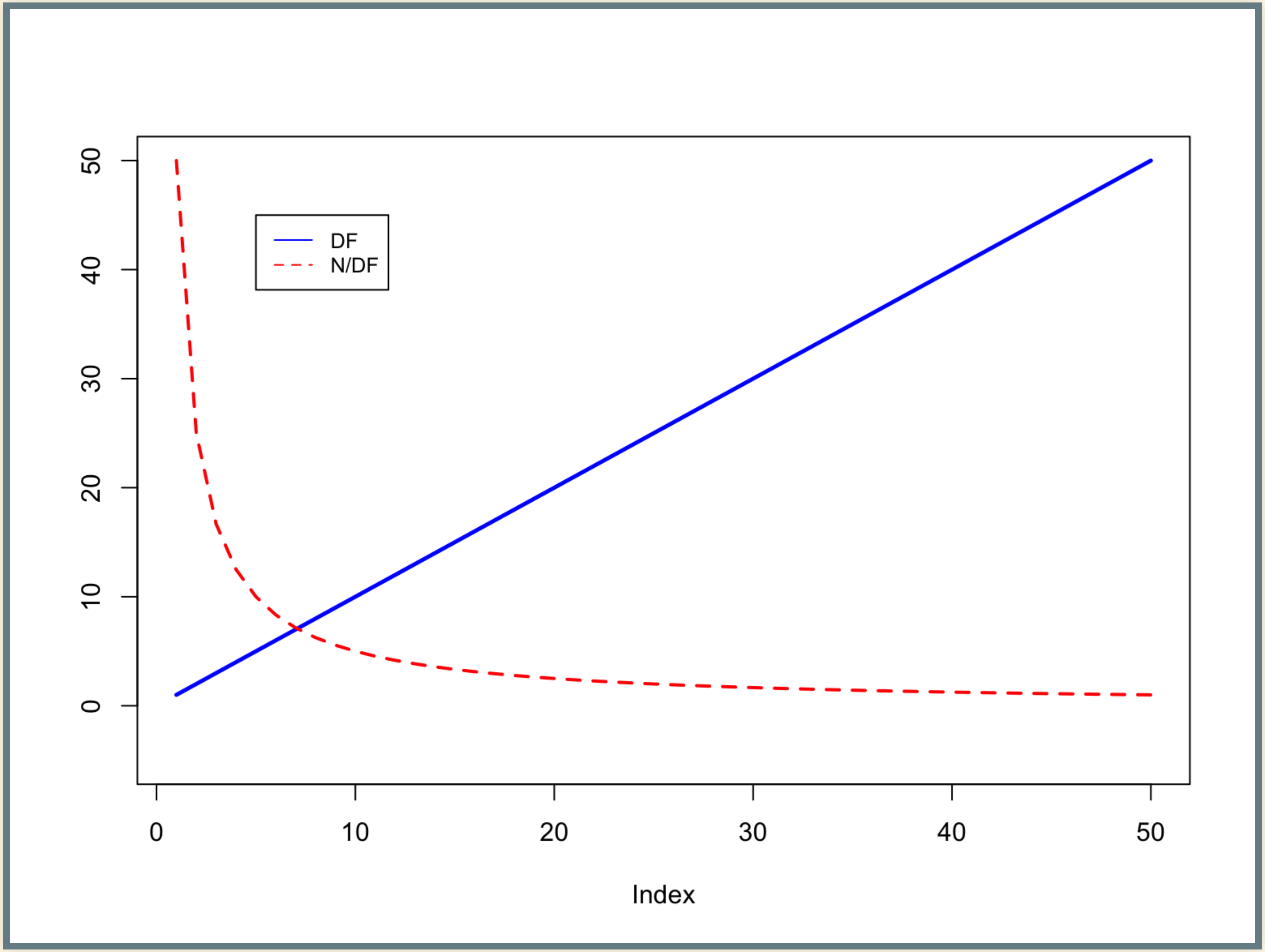
- we want: low values for common words
- and: higher values for *important* words

Trick: *inverse* document-frequency

INVERSE DF (IDF)

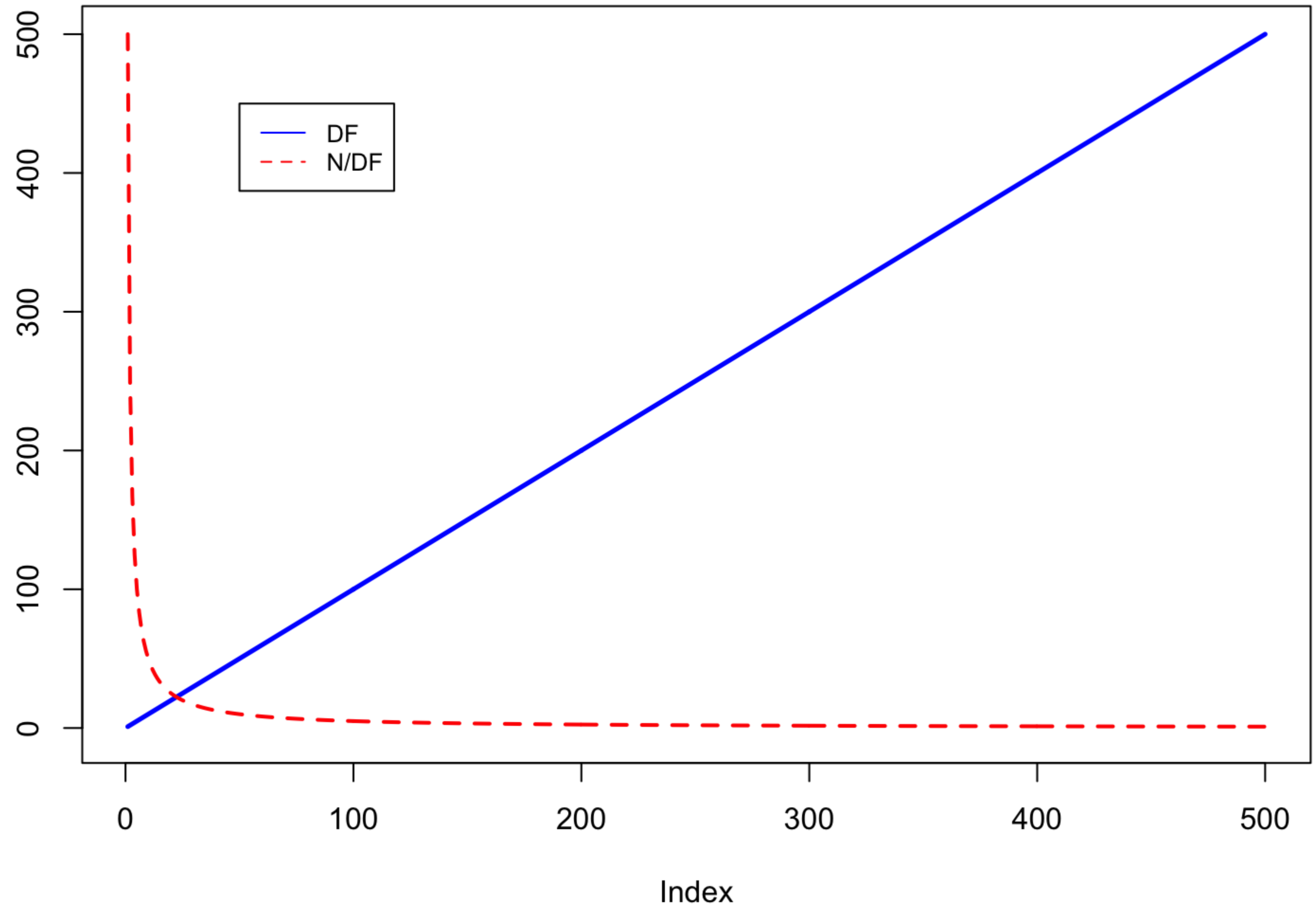
$$idf(t) = \frac{N}{df(t)}$$

IDF



PROBLEM?

- What happens if N becomes really large?



DF
N/DF
0
100
200
300
400
500
0
100
200
300
400
500
Index

LOG IDF

- to avoid extreme values, we use the logarithm
- simple transformation: $idf(t) = log(\frac{N}{df(t)+1})$

ABOUT LOGARITHMS

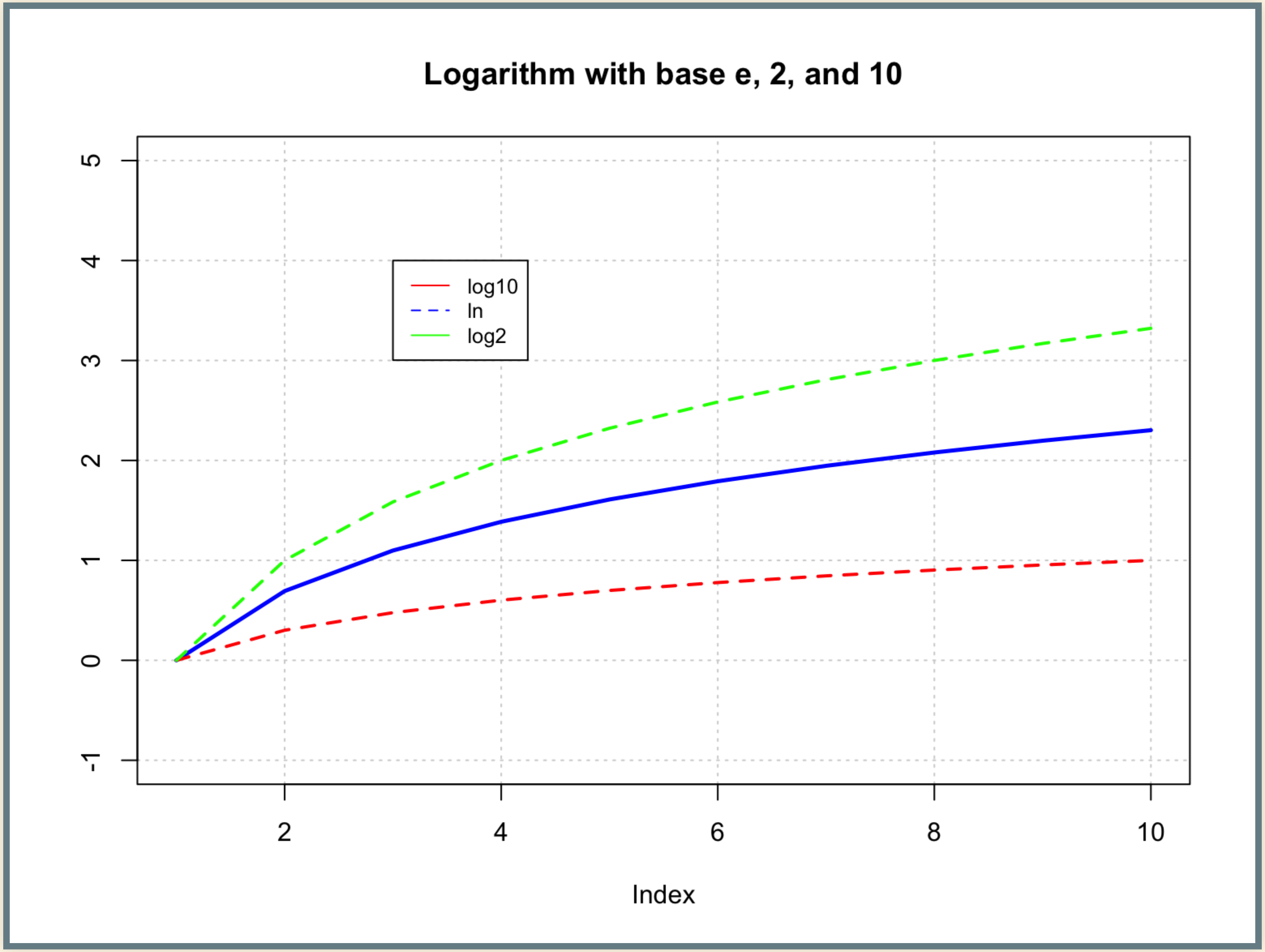
$$log_b(a) = c <=> b^c = a$$

“the power to which {base b} would have to be raised to equal a”

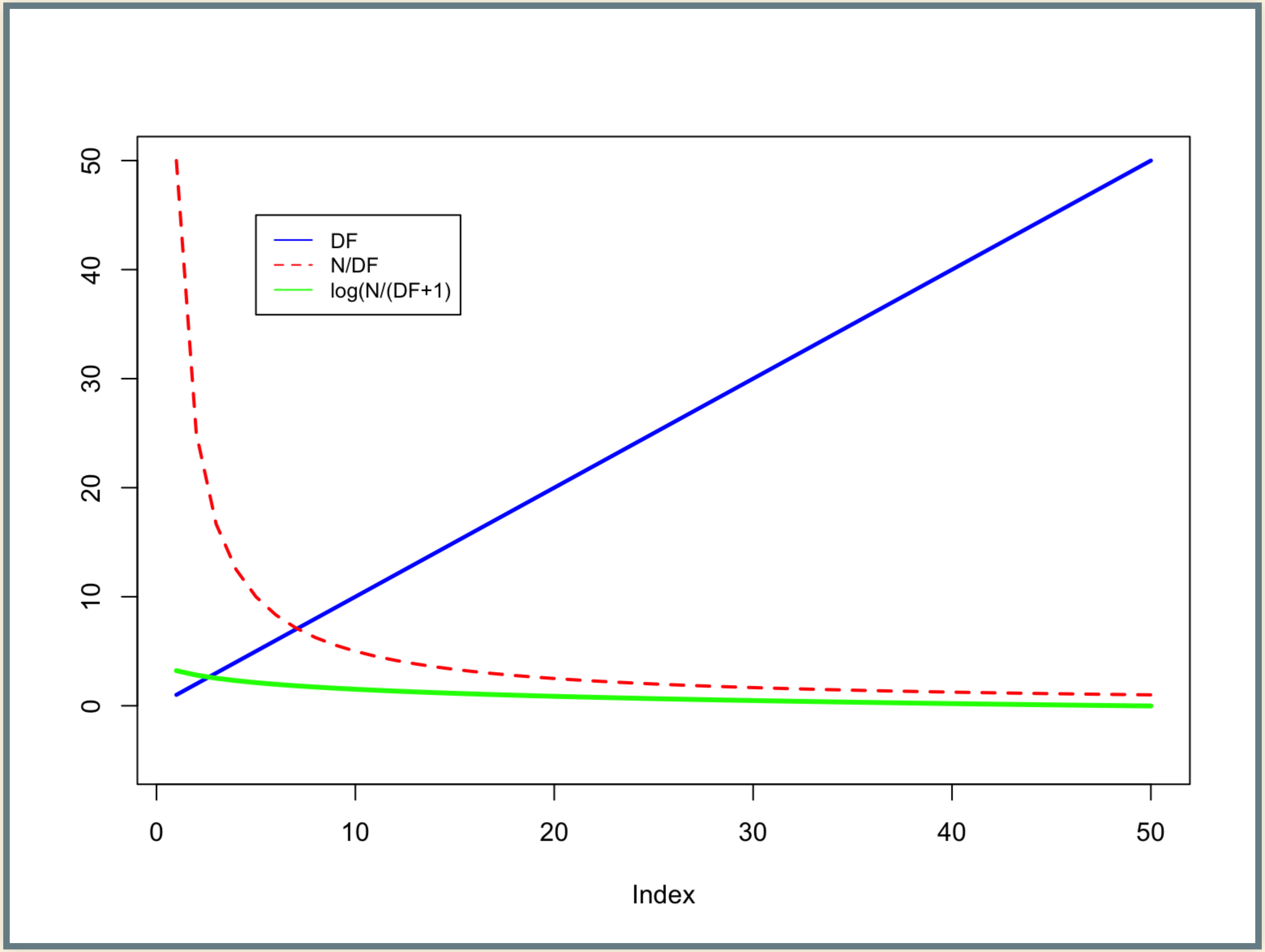
$$log_2(8) =? \text{->} 2^? = 8$$

LOGARITHM BASES

- natural logarithm: base $e = 2.718281...$
- base 10 logarithm: $log_{10}(x)$


Logarithm with base e, 2, and 10
log10
ln
log2
5
4
3
2
1
0
-1
2
4
6
8
10
Index

LOG IDF



COMBINING TERM FREQUENCY AND DOCUMENT FREQUENCY

1. Local importance (TF)

document	and	in	turn
text1	3	2	1
text2	2	2	0

2. Correct for global occurrences (DF)

	x
and	-0.1760913
in	-0.1760913
turn	0.0000000

TF/IDF

```
#text1: "and"
3/-0.1760913

#text2: "and"
2/-0.1760913
```

TF-IDF

TF-IDF

TF-IDF is a measure of originality of a word by comparing the number of times a word appears in a doc with the number of docs the word appears in.

$$\text{TF-IDF} = TF(t, d) \times IDF(t)$$

Term frequency

Number of times term t appears in a doc, d

Inverse document frequency

$$\log \frac{1 + n}{1 + df(d, t)} + 1$$

# of documents

Document frequency of the term t

Img reference

WORKING WITH TEXT AS DATA

RESEARCHER’S DEGREES OF FREEDOM

- stop word removal
- stemming

STOPWORD REMOVAL

- We know many words are “low in meaning”
- so-called stop words

x
i
me
my
myself
we
hers
herself
it
its
itself
they

WITHOUT STOPWORD REMOVAL

document	all	i	ever	wanted	was	to	love	women
text1	2	10	1	1	1	3	1	1
text2	0	0	0	0	0	3	0	0

WITH STOPWORD REMOVAL

document	ever	wanted	love	women	,	turn	loved	back
text1	1	1	1	1	4	1	1	2
text2	0	0	0	0	4	0	0	0

STEMMING

- some words originate from the same “stem”
- e.g. “love”, “loved”, “loving”, “lovely”
- but you might want to reduce all these to the stem

WORD STEMS

```
love_stem = c("love", "loved", "loving", "lovely")
```

document	love	loved	loving	lovely
text1	1	0	0	0
text2	0	1	0	0
text3	0	0	1	0
text4	0	0	0	1

AFTER STEMMING

document	love
text1	1
text2	1
text3	1
text4	1

OUR MINI CORPUS

Incl. stop words and without stemming:

document	all	i	ever	wanted	was	to	love	women
text1	2	10	1	1	1	3	1	1
text2	0	0	0	0	0	3	0	0

... TO

Without stop words and stemmed:

document	ever	want	love	women	,	turn	back	.
text1	1	2	2	1	4	1	2	12
text2	0	0	0	0	4	0	0	3

TEXT METRICS

- lexical diversity (= TTR)
- readability

READABILITY

- ease of understanding for the reader
- readability vs legibility
- here: focus on language

READABILITY ASPECTS

- No. of words
- No. of characters
- No. of *difficult* words
- punctuation
- number of syllables

READABILITY METRICS

- Flesch Reading Ease score
- Coleman–Liau index
- Automated readability index (ARI)

FLESCH READING EASE SCORE

Requires:

- No. of words
- No. of sentences
- No. of syllables

$$FRE = 206.835 - 1.015 * (\frac{nwords}{nsentences}) - 84.6 * (\frac{nsyllables}{nwords})$$

SCORE INTERPRETATION

Score	School level	Notes
100.00–90.00	5th grade	Very easy to read. Easily understood by an average 11-year-old student.
90.0–80.0	6th grade	Easy to read. Conversational English for consumers.
80.0–70.0	7th grade	Fairly easy to read.
70.0–60.0	8th & 9th grade	Plain English. Easily understood by 13- to 15-year-old students.
60.0–50.0	10th to 12th grade	Fairly difficult to read.
50.0–30.0	College	Difficult to read.
30.0–0.0	College graduate	Very difficult to read. Best understood by university graduates.

FRE IN R

```
n_words = ntoken(mini_corpus[1])
n_sentences = nsentence(mini_corpus[1])
n_syllables = nsyllable(mini_corpus[1])
```

CUSTOM FUNCTION

```
fre_score = function(input_text){
  n_words = ntoken(input_text)
  n_sentences = nsentence(input_text)
  n_syllables = nsyllable(input_text)
  fre = 206.835 - 1.015*(n_words/n_sentences) - 84.6*(n_syllables/n_words
  return(unname(fre))
}
```

CALCULATING THE FRE

```
fre_score(mini_corpus[1])
```

```
## [1] 100.2511
```

```
fre_score(mini_corpus[2])
```

```
## [1] 26.12758
```

COLEMAN-LIAU INDEX

$$CLI = 0.0588 * \frac{nchar}{nwords} * 100 - 0.296 * \frac{nsentences}{nwords} * 100 - 1$$

Outcome score ≈ US grade-level.

CLI FUNCTION

```
cli_score = function(input_text){
  n_words = ntoken(input_text)
  n_sentences = nsentence(input_text)
  n_characters = nchar(input_text)
  cli = 0.0588*(n_characters/n_words)*100 - 0.296*(n_sentences/n_words)*1
  return(unname(cli))
}
```

CALCULATING THE FRE

```
cli_score(mini_corpus[1])
```

```
## [1] 6.455935
```

```
cli_score(mini_corpus[2])
```

```
## [1] 17.46864
```

CONSIDERATIONS WITH TEXT DATA

- a lot of assumptions
- text == behaviour?
- produced text == displayed text?
- many decisions in your hand
  - stemming
  - stop words
  - custom dictionary

WHAT'S NEXT?

- Today's tutorial: analysing speeches by US presidents, examining UK rap lyrics
- Homework: quanteda practice, text preprocessing

Next week: Text Mining 2