Data Mining and Machine Learning

Lecture 3

Stopping, F-IDF

Similarity

Peter Jančovič

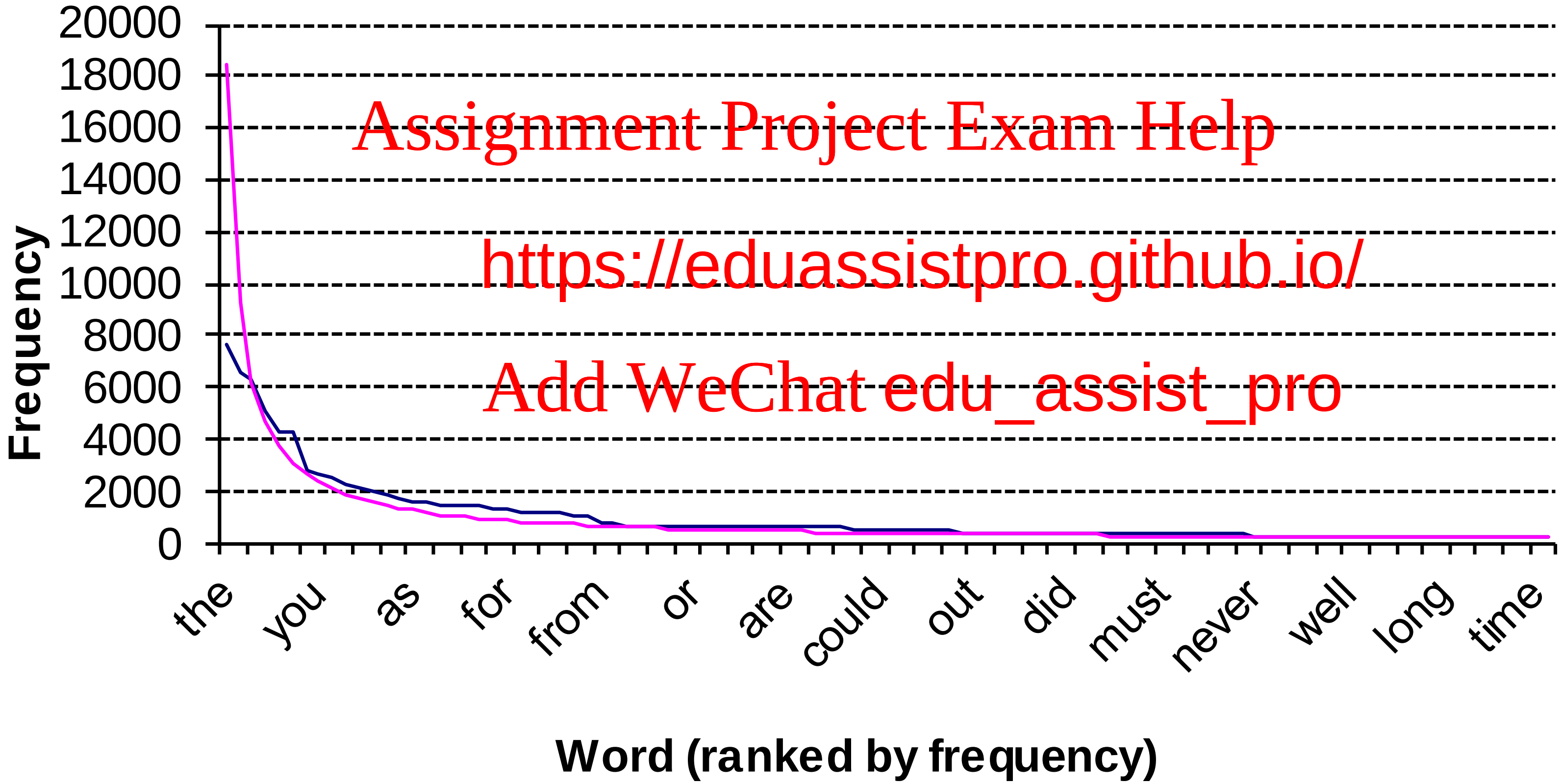
UNIVERSITY OF
BIRMINGHAM

Objectives

- Understand definition and use of **Stop Lists**
- Understand motivation and methods of **Stemming**
- Understand -IDF Similarity between two

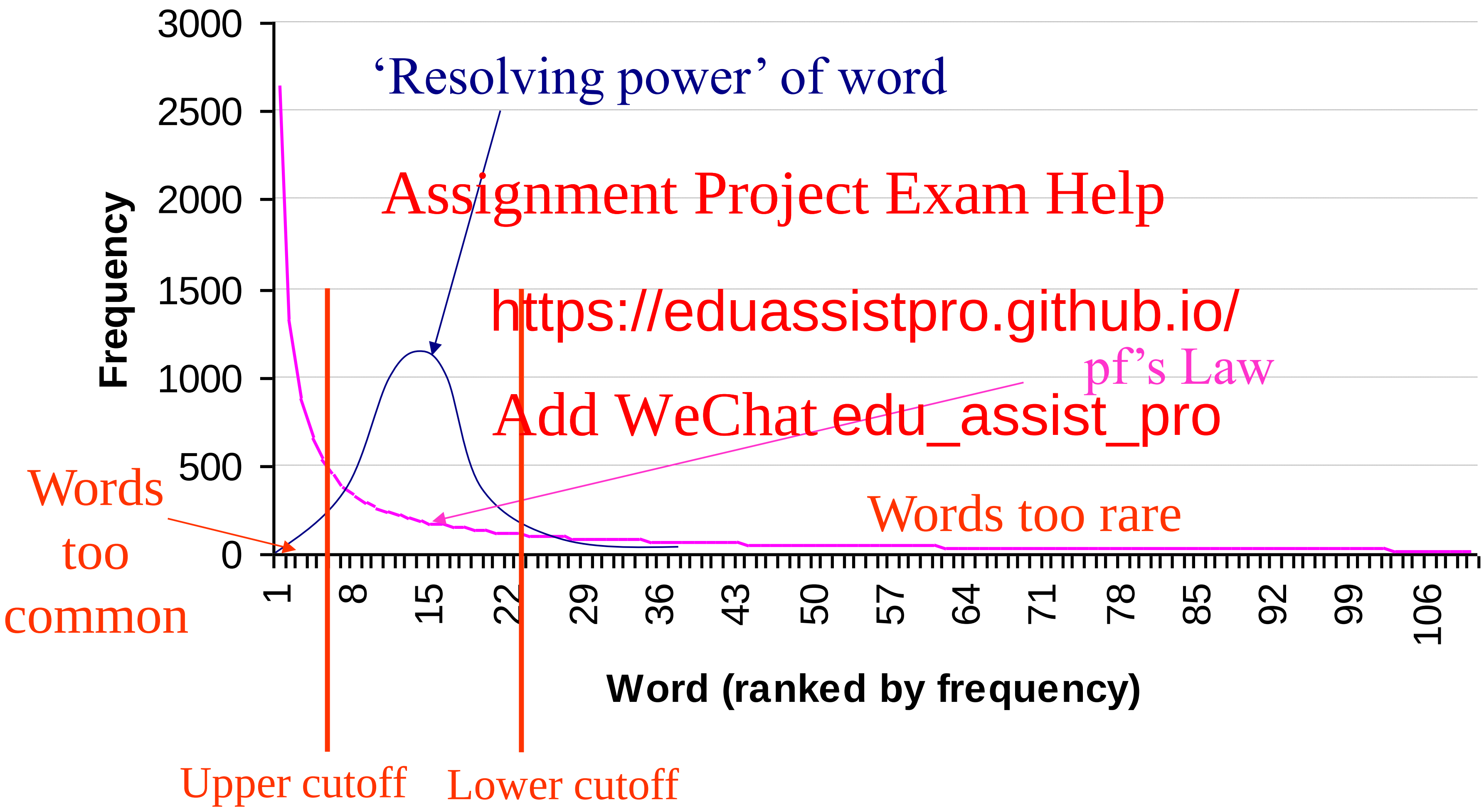
Zipf's Law

Zipf's law —— Actual statistics from "Jane Eyre" ——



UNIVERSITY OF BIRMINGHAM

‘Resolving Power’ of words



Text Pre-Processing

- Stop Word Removal: Simple techniques to remove 'noise words' from texts
  - Remove common 'noise' words which contribute no information to the IR process (e.g. "the")
- Stemming: Re [illegible] s from different 'versions' of the same word
  - Identify different forms of t [illegible] d (e.g. "run" and "ran") identify them with a common stem
- (Later) Exploit semantic relationships between words
  - If two words have the same meaning, treat them as the same word

Stemming (morphology)

- Basic idea: If a query and document contain different forms of the same word, then they are related
- Remove surface markings from words to reveal their basic form:
  - forms → fo
  - formed → form, former →
- “form” is the stem of forms, forming, formed, former

Stemming (morphology)

- Stemming replaces tokens (words) with equivalence classes of tokens (words)
- Equivalence classes are stems
  - Reduces the number of different words in a corpus
  - Increases th [illegible] ach token

Stemming

- Of course, not all words obey simple, regular rules:
  - running → run
  - runs → run
  - women →
  - leaves → le
  - ferries → ferry
  - alumnus → alumni
  - datum → data
  - crisis → crises

[Belew, chapter 2]

Stemming

- Linguists distinguish between different types of morphology:
  - Minor changes, such as plurals, tense
  - Major chan ntivize, which change the word

- Common solution is to ide attern of letters within words and devise rules for dealing with these patterns

Stemming

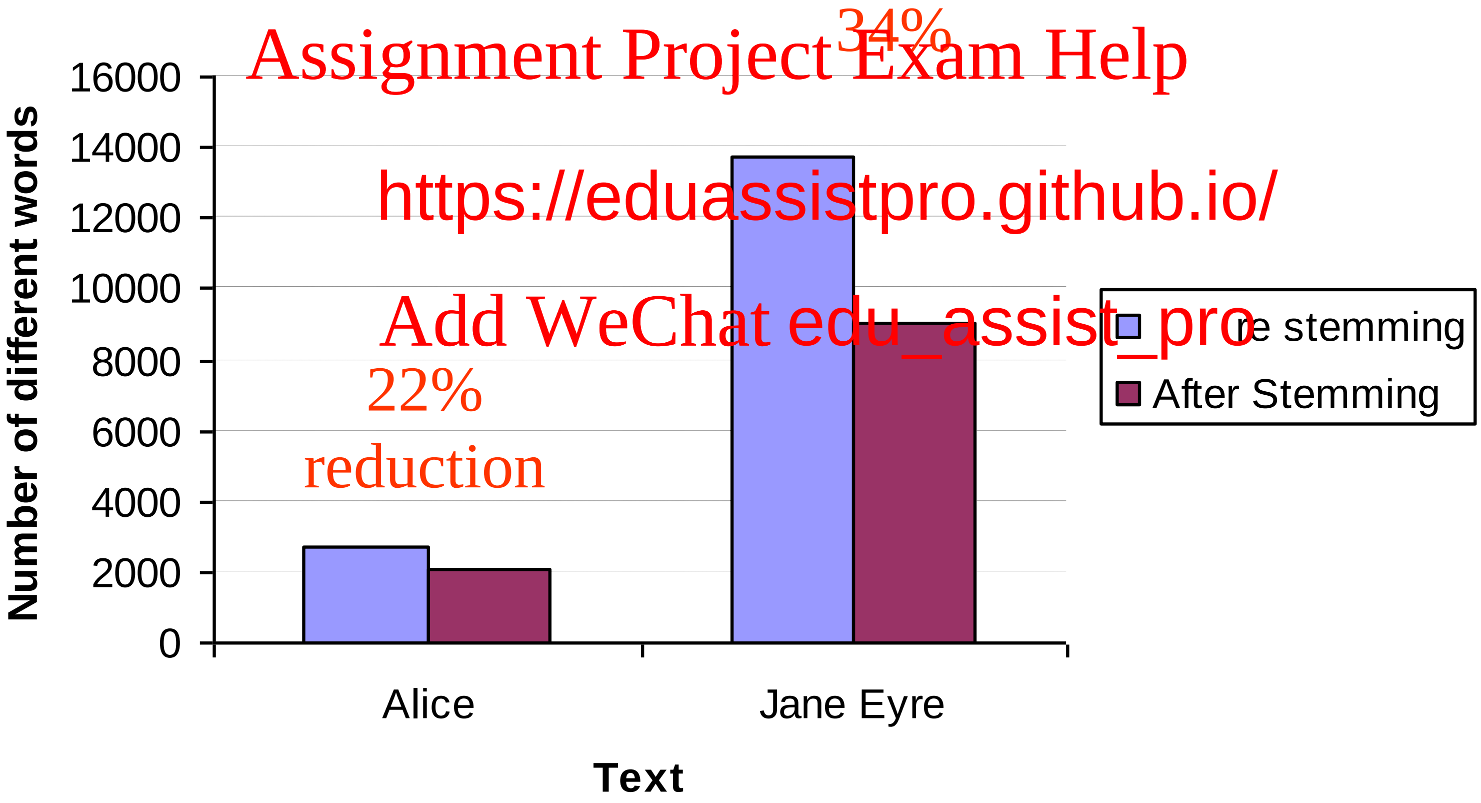
- Example rules [Belew, p 45]
  - (.*)SSES → /1SS
    - Any string ending SSES is stemmed by replacing SSES w
    - E.G: “cl
  - (.[AEIOU].*)ED → /1
    - Any string containing a vowel and ending in ED is stemmed by removing the ED
    - E.G. “classed” → “class”

Stemmers

- A stemmer is a piece of software which implements a stemming algorithm
- The Porter stemmer is a standard stemmer which is available as anvas)
- The Porter st et of about 60 rules
- Use of a stemmer typically reduces vocabulary size by 10% to 50%

Example

- Apply the Porter stemmer to the ‘Jane Eyre’ and ‘Alice in Wonderland’ texts



Example

- Examples of results of Porter stemmer:
  - form → form
  - former → former
  - formed → form
  - forming → form
  - formal → form
  - formality → fo
  - formalism → formal
  - formica → formica
  - formic → formic
  - formant → formant
  - format → format
  - formation → format

Example: First paragraph from ‘Alice in Wonderland’

Before

Alice was beginning to get very tired of sitting by her sister on the bank, and of having nothing to do: once or twic

peeped into the bo

was reading, but it had no pictures or conversations in it, ‘and what is the use of a book,‘thought Alice ‘without pictures or conversation?’

After

alic wa begin to get veri tire of sit by her sister on the bank, and of have noth to do: onc or twice

p into the book her

ad, but it had

p ers in it, ‘and what is the us of a book,’ thought alic ‘without pictur or convers?’

Noise Words – "Stop words"

There was no possibility of taking a walk that day. We had been wandering, indeed, in the leafless shrubbery an hour in the morning; but since dinner (Mrs. Reed, when there was no company, dined early) the cold winter wind had brought with it clouds so sombr , that further out-door exercise wa

- Noise words
  - Vital for the grammatical structure of a text
  - Of little use in the 'bundle of words' approach to identifying what a text is "about"

Stop Lists

- In Information Retrieval, these words are often referred to as Stop Words
- Rather than detecting stop words using rules, stop words are si ystem in a text file: the Stop
- Stop Lists typically consis st common words from some large corpus
- There are lots of candidate stop lists online

Example 1: Short Stop List (50 wds)

the
of
and
to
a
in
that
is
was
he
for

it
with
as
his
on
be
at
by
i
this
had

not
are
but
from
an
they
which
you
were

her
all
she
there
we
him
been
has
when

who
will
more
if

Example 2: 300 Word Stop List

the
of
and
to
a
in
that
is
was
he
for
it
with
as
his

on
be
at
by
i
this
had
not
are
but
from
or
have
an
they
which

one
you
were
her
all
she
there
would
their
we
him
been
has
when
who
will

more
no
if
out
so
said
about
into
than
them
can
only
other

held
keep
sure
probably
free
real
seems
behind
cannot
miss
political
air
question
making
office
brought

whose
special
heard
major
problems
ago
became
federal
moment
study
available
known
result
street
economic
boy

300 most common words from Brown Corpus

The text matters

Alice vs Brown: Most Frequent Words

the	the	as	his	this	an	know	has	thought
and	of	her	on	they	they	them	when	off
to	and	at	be	little	which	like	who	how
a	to	on	at				will	me
she	a	all	by				more	
it	in	with	i	is	her			
of	that	had	this	one	all		if	
said	is	but	Had	down	she			
i	was	for	not	up	there	do		
alice	he	so	are	his	would	have		
in	for	be	but	if	their	when		
you	it	not	from	about	we	could		
was	with	very	or	then	him	or		
that	as	what	have	no	been	there		

stop.c

- C program on course Canvas page
  - Reads in a stop list file (text file, one word per line)
  - Stores stop words in char **stopList
  - Read text fi
  - Compares e ord
  - Prints out words not in stop
- `stop stopListFile textFile > opFile`

Examples

Original first paragraph

Alice was beginning to get very tired of sitting by her sister on the bank, and of having noth

once or twice she had pe

the book her sister was r

it had no pictures or conversations in it, `and what is the use of a book,‘ thought Alice `without pictures or conversation?'

Stop list 50 removed

alice beginning get very tired sitting sister bank having nothing do once twice peeped into book sister reading no pictures conversations what use book thought alice without pictures

wn removed

al tired sitting sister bank twice peeped book sister reading pictures conversations book alice pictures

conversation

Matching

- Given a query ***q*** and a document ***d*** we want to define a number:

$$Sim(\boldsymbol{q},\boldsymbol{d})$$

which define [illegible] n ***q*** and ***d***

- Given the qu ***q*** [illegible] rn the documents $\boldsymbol{d_1}\ \boldsymbol{d_2} \ldots \boldsymbol{d_N}$ such that:
  - $\boldsymbol{d_1}$ is the document for which $Sim(\boldsymbol{q},\boldsymbol{d})$ is biggest
  - $\boldsymbol{d_2}$ has the next biggest value of $Sim(\boldsymbol{q},\boldsymbol{d})$,
  - etc

Similarity

- The similarity between ***q*** and ***d*** will depend on the number of terms which are common to ***q*** and ***d***
- But we also need to know how useful each common term is for di [illegible] different documents.

- For example,
  - It is probably not significant if ***q*** and ***d*** share "*the*"
  - But it probably is significant if they share "*magnesium*"

IDF weighting

- One commonly used measure of the significance of a term for discriminating between documents is the Inverse Document Frequency (IDF)
- For a token ***t***

$$IDF(t) = \log\left(\frac{ND}{ND_t}\right)$$

- *ND* is the total number of documents in the corpus
- ND_t is the number of those documents that include ***t***

Why is IDF weighting useful?

$$IDF(t) = \log\left(\frac{ND}{ND_t}\right)$$

- Case 1: ***t*** occurs equally often in all documents
  - $ND = ND_{t,}$
  - hence *IDF*(
- Case 2: ***t*** occurs in just a f ents
  - $ND > ND_t$
  - hence $\underline{IDF(t) > 0}$
- Note that *IDF*(*t*) ignores how often term ***t*** occurs in a document

Effect of Document Length

- Suppose query ***q*** consists only of term ***t***
- Suppose document $\boldsymbol{d_1}$ also consists only of ***t***
  - Number of shared terms is 1
  - Match is ‘p

- Suppose doc $_{2}$ s, including ***t***
  - Number of shared terms is
  - But in this case co-occurrence of ***t*** appears less significant
- Intuitively the similarity measure *Sim*(***q***,***d***) needs to include <u>normalisation</u> by some <u>function</u> of ***N*** and ***M***

TF-IDF weight

- Let ***t*** be a term and ***d*** a document
- TF-IDF – Term Frequency – Inverse Document Frequency
- The TF-IDF w[illegible] document ***d*** is:

$$w_{td} = f_{td} \cdot IDF(t)$$

where:

f_{td} = term frequency – the number of times ***t*** occurs in ***d***

TF-IDF weight (continued)

$$w_{td} = f_{td} \cdot IDF(t)$$

- For w_{td} to be

  - f_{td} must be l n in ***d***

  - $IDF(t)$ must be large, so ***t*** m ur in relatively few documents

Query weights

- Now suppose ***t*** is a term and ***q*** is a query.
- If ***q*** is a <u>long</u> query, can treat ***q*** as a document:

$$w_{tq} = f_{tq} \cdot IDF(t)$$

where f_{tq} is the (query) term frequency – the number of times the term ***t*** occurs in the query ***q***

- If ***q*** is a <u>short</u> query, define the TF-IDF weight as

TF-IDF Similarity

- Define the similarity between query ***q*** and document ***d*** as:

Sum over all terms in both ***q*** and ***d***

$$Sim\left(q,d\right) = \frac{\boxed{}}{\boxed{\|d\|}\cdot\boxed{\|q\|}}$$

‘Length’ of query ***q***

‘Length’ of document ***d***

Document length

- Suppose ***d*** is a document
- For each term ***t*** in ***d*** we can define the TF-IDF weight $\boldsymbol{w_{td}}$

- The length o d by:

$$Len(d) = \|d\| = \sqrt{\sum_{t\in d} w_{td}^2}$$

Comments on Document Length

- This definition of *Len*(***d***) may not seem very intuitive at first

- It will become more intuitive when we study vector representatio atent Semantic Indexing (L

- For now, just remember th ***$_1$,x$_2$,x$_3$*)** is a vector in 3 dimensional space, then the length of ***x*** is given by:

$$\|x\| \qquad \sqrt{\quad {}^{2} \qquad {}^{2} \qquad {}^{2}}$$

Summary

- Understand definition and use of **Stop Lists**
- Understand motivation and methods of **Stemming**
- Understand -IDF Similarity between two