# TF-IDF

# Contents

# Text Preprocessing

# Text Preprocessing

Common text preprocessing steps:

1. Tokenization: Split text into sentences or words
2. Lowercasing
3. Removing Punctuation and Special Characters
4. Removing Stop Words: Eliminating common words (e.g., “and”, “the”, “is”) that may not add significant meaning.
5. Stemming and Lemmatization

# Text Preprocessing (cont.)

Stemming:

- Just chops off the end of the word.
- e.g. Remove “ement” from the end of the word.
  - e.g. management -> manag
- e.g. Remove “es” when occur “sses”.
  - e.g. bosses -> boss

# Text Preprocessing (cont.)

Stemming Algorithm e.g. Porter:

```
from nltk import PorterStemmer
porter = PorterStemmer
porter.stem("walking") # Return "walk"
```

# Text Preprocessing (cont.)

Lemmatization:

- Use actual rules of language. Return root word.
- Think of a look up table of rules.
- e.g. “better”
  - Stemming: “better” -> “better”
  - Lematization: “better” -> “good”
- Package

```
from nltk.stem import WordNetLemmatizer
```

# TF-IDF

# TF-IDF

$$\text{TFIDF} \simeq \frac{\text{Term Frequency}}{\text{Document Frequency}}$$

$\text{tfidf} = \text{tf}(t, d) \times \text{idf}(t)$

- $\text{tf}(t, d)$ # times t appears in d
- $\text{idf}(t) = \log\left(\frac{N}{N(t)}\right)$
  - $N(t)$ Number of docs term t appears in
  - $N$: total number of docs

# TF-IDF (cont.)

- $tf$ -> Matrix of $N$ by $V$ where
  - $N$: Number of docs
  - $V$: Number of words
- $idf$ -> $V$ size vector. Each element represent the word's frequency.

$$\begin{pmatrix} & \dots & V & \dots \\ \vdots & & & \\ N & & & \\ \vdots & & & \end{pmatrix} \times \begin{pmatrix} \vdots \\ V \\ \vdots \end{pmatrix}$$

Thank you