Natural Language Processing

Text Preprocessing

- Before we can use text data for modeling, we have to make sure it's in the right format.

Preprocessing Pipeline

Phase 1:
1. Tokenize
2. Denoising (remove unnecessary spaces and special characters)
3. Stemming
4. Lemmatization

Phase 2:
5. Numeric representation of words

① **Tokenization**

Sentence : " This is Aneesh's book, isn't it ? "

0 1 2 3 4 5 6

— One way

② **Denoise Text**: Remove special characters (ex. ', ?, #, $, numbers)

③ **Stemming**: Removing and replacing suffixes to get the root form of the word.

Ex. dogs → dog

raining → rain

④ **Lemmatization**: Get base form of word

Ex. feet → foot

Phase 2

⑤ Numeric representation for words.

- One-hot encoding

Ex) Sentence 1: "Time flies like an arrow."

Sentence 2: "Fruit flies like a banana."

vocab: { Time, fruit, flies, like, a, an, arrow, banana } — 8 elements

(Time: 1st, flies: 3rd)

Time : [1, 0, 0, 0, 0, 0, 0, 0]

flies: [0, 0, 1, 0, 0, 0, 0, 0]

like : [0, 0, 0, 1, 0, 0, 0, 0]

an : [0, 0, 0, 0, 0, 0, 1, 0]

arrow : [0, 0, 0, 0, 0, 0, 0, 1]

Sentence 1 = [1, 0, 1, 1, 0, 0, 1, 1]

TF-IDF

① Term-Frequency (TF)

Sent 1. "Fruit flies like a banana" Sent 2. "Time flies like an arrow"

→ Sent 3. "Fruit flies like time flies a fruit"

→ vocab = {Time, Fruit, flies, like, a, an, arrow, banana}

→ [1, 2, 2, 1, 1, 0, 0, 0]

② Inverse document frequency

$$IDF(w) = \log\left(\frac{N}{n_w}\right) = \log\left(\frac{7}{2}\right) = 1.25 \leftarrow \text{flies}$$

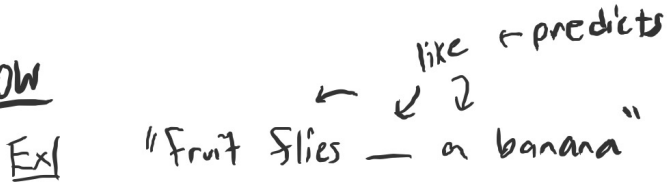
$$\log\left(\frac{7}{1}\right) = 1.94 \leftarrow \text{Fruit}$$

TF-IDF = TF * IDF

Pretrained Word Embeddings

- Word2Vec
  - Used a neural network to train on word representation
  - Better numeric vector representation of words.

CBOW



Skip gram