NLP-1

Narasimha Rohit Katta

January 2024

Question-1:

A:

$$A = \begin{pmatrix} 2 & 7 & 7 \\ 4 & 3 & 1 \end{pmatrix}$$

$$B = \begin{pmatrix} 0 & 2 \\ 1 & 0 \\ 1 & 1 \end{pmatrix}$$

The product AB is:

$$AB = \begin{pmatrix} 14 & 11 \\ 4 & 9 \end{pmatrix}$$

B: Given Matrices:

$$A = \begin{pmatrix} 0 & 6 \\ 9 & 2 \\ 2 & 1 \end{pmatrix}$$

and

$$B = \begin{pmatrix} 1 & 7 \\ 2 & 3 \\ 5 & 1 \end{pmatrix}$$

Solution:

The product $A \times B^\top$ is:

$$\begin{pmatrix} 42 & 18 & 6 \\ 23 & 24 & 47 \\ 9 & 7 & 11 \end{pmatrix}$$

C: INVALID

D:

$$\begin{bmatrix} 3 & 12 & 15 \\ 1 & 4 & 5 \\ 0 & 0 & 0 \end{bmatrix}$$

E: INVALID

Question-2:

a) One or more words (only with lowercase alphabets) separated by spaces
Answer: `^([a-z]+(?:\s+[a-z]+)*)$`

b) Title case sentences, assuming all words are capitalized. Note that the text can contain numbers and punctuation
Answer: `^[A-Z][a-z]*(?:[ \-,.:;!?][A-Z][a-z0-9]*)*$`

c) Strings that contain the word "ice" without matching the words that contain "ice" such as "icecream" or "ice-bucket"
Answer: `\bice\b`

d) The set of all lowercase alphabetic strings ending in a "b"
Answer: `^[a-z]*b$`

e) All strings that start at the beginning of the line with an integer and that end at the end of the line with a word
Answer: `^\d+.+\b\w+$`

1 Question-3:

2 Natural Language Processing Challenges in Telugu

Telugu Language Selection

I have chosen Telugu, a South Indian language, for this analysis.

Challenges

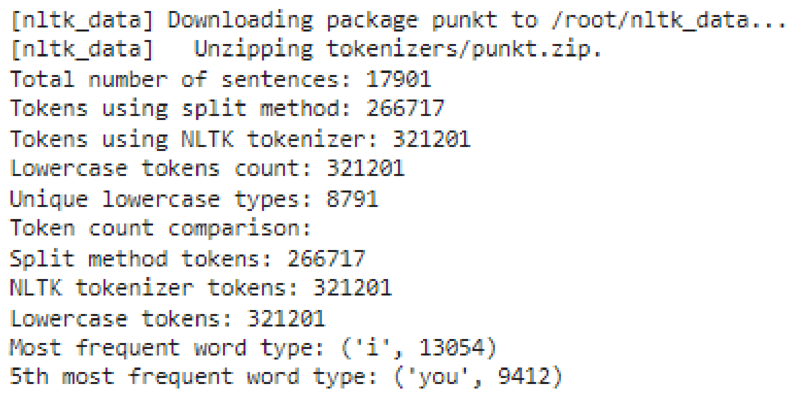
- **Script and Orthographic Complexity:** Telugu uses a complex script. For example, the character క (ka) can change to కా (kaa), కి (ki), etc.
- **Agglutinative Nature:** Telugu forms long words by combining morphemes. For instance, చదివించారు (chadivinchaaru) means ”they made to read.”
- **Context-Dependent Meanings:** Words in Telugu change their meaning based on context. For example, పడుకో (paduko) can mean ”sleep” or ”make someone sleep.”

3 CoNLL-2003 English Dataset Overview

One well-known publicly available English dataset for NLP tasks is the ”Stanford Sentiment Treebank” (SST), specifically designed for sentiment analysis:
Motivation:
The Stanford NLP Group created the Stanford Sentiment Treebank. This dataset was created with the intention of serving as a standard for sentiment analysis research, specifically with regard to comprehending the sentiment of movie review phrases.
Situation:
This dataset contains text that was written as reviews of movies. The Rotten Tomatoes website, which offers a diverse range of viewpoints on different films, was the original source of these reviews.
Collection Process:
The movie reviews dataset is sourced from an already-existing dataset. Preprocessing was done on the reviews to break them up into sentences. Though it’s possible that separate consent wasn’t sought for the development of this dataset, the original authors of the reviews likely gave their approval for their usage on the Rotten Tomatoes website.

- **Annotation Process:**
  Sentiment annotations are made on the SST. Every sentence in the sample has a sentiment classification that ranges from extremely negative to extremely positive. Using Amazon Mechanical Turk, employees annotated sentences by labeling the sentiment of each one. In order to increase accuracy and uniformity, the procedure required making sure that each sentence was classified by different personnel.

4 Question-4:



```
[nltk_data] Downloading package punkt to /root/nltk_data...
[nltk_data]   Unzipping tokenizers/punkt.zip.
Total number of sentences: 17901
Tokens using split method: 266717
Tokens using NLTK tokenizer: 321201
Lowercase tokens count: 321201
Unique lowercase types: 8791
Token count comparison:
Split method tokens: 266717
NLTK tokenizer tokens: 321201
Lowercase tokens: 321201
Most frequent word type: ('i', 13054)
5th most frequent word type: ('you', 9412)
```

Figure 1: Question-4

4.1 (E) Comparing Token Quantities

In this part, the script contrasts the counts of tokens derived through various techniques: splitting by spaces, employing NLTK's "word-tokenize()", and post lowercasing. This comparison underscores the impact of different tokenization and normalization methods on the total number of identified tokens, crucial for textual data analysis.

4.2 (H) Zipf's Law Frequency Plot

This section involves generating a chart to investigate whether the distribution of word frequencies aligns with Zipf's Law. This empirical rule suggests a reciprocal relationship between a word's rank and its frequency. The plot, set on a log-log scale, places ranks on the X-axis against frequencies on the Y-axis. A linear trend in this plot indicates adherence to Zipf's Law, offering insights into the text's lexical characteristics.

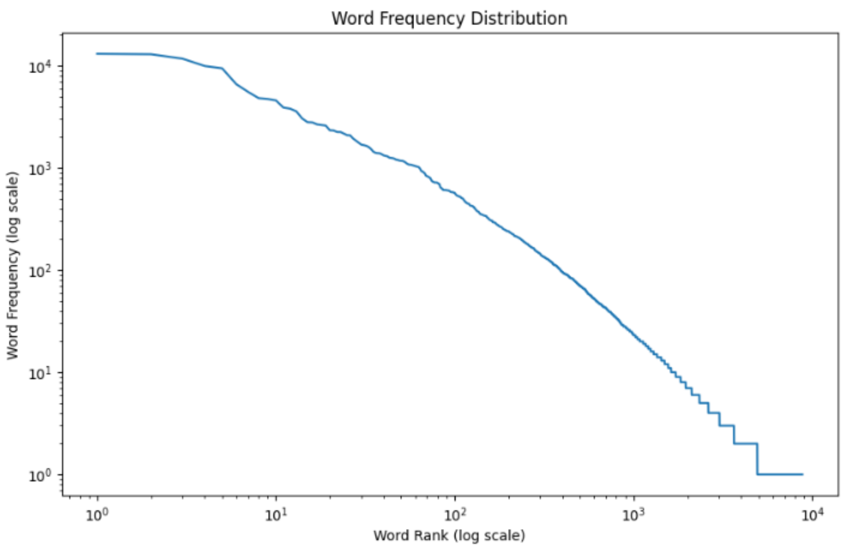


Figure 2: Zipf's Law Frequency Plot