Old:Week – 2: REMOVE NUMBERS WITH INFLECT LIBRARY AND LEMMATIZATION

1. **Convert the numbers into words. This can be done by using the inflect library**

The `inflect` library is a Python package that provides functionality for converting numbers into their textual representation, which can be useful in various applications such as generating reports, creating user-friendly interfaces, and more. It supports conversion of numbers into words for different languages, pluralization of words, and other numeral-related operations.

To use the `inflect` library to convert numbers into words, you first need to install it using pip:

pip install inflect

**code:**

import inflect

# Create an instance of the inflect engine

p = inflect.engine()

# Convert a number into words

number = 12345

words = p.number\_to\_words(number)

print("Number in words:", words)

**output:**

Number in words: twelve thousand, three hundred and forty-five

**Explanation of the code:**

1. We import the `inflect` library.

2. We create an instance of the `inflect` engine using `inflect.engine()`.

3. We specify the number that we want to convert into words (e.g., `12345`).

4. We use the `number\_to\_words` method of the `inflect` engine to convert the number into its textual representation.

5. We print the result, which is the number expressed in words.

For example, if `number` is `12345`, the output will be:

Number in words: twelve thousand three hundred forty-five

This library is particularly useful when you need to generate human-readable text representations of numerical data, such as in financial reports, invoices, or educational applications.

1. **Use the WordNetLemmatizer to get the lemmas of words and also need to provide a context for the lemmatization. So, that add part-of-speech as a parameter.**

In natural language processing (NLP), lemmatization is the process of reducing words to their base or dictionary form, known as lemmas. The WordNetLemmatizer in NLTK is used for lemmatization. One important aspect of lemmatization is that it often requires information about the part-of-speech (POS) of the word to accurately determine its lemma. For example, the lemma of the word "running" can be different based on whether it is used as a verb, noun, adjective, or adverb.

**Code:**

import nltk

from nltk.stem import WordNetLemmatizer

from nltk.corpus import wordnet

# Download WordNet corpus

nltk.download('wordnet')

# Create WordNetLemmatizer instance

lemmatizer = WordNetLemmatizer()

# Function to lemmatize a word with specified part-of-speech

def lemmatize\_word(word, pos):

if pos.startswith('J'):

pos = wordnet.ADJ

elif pos.startswith('V'):

pos = wordnet.VERB

elif pos.startswith('N'):

pos = wordnet.NOUN

elif pos.startswith('R'):

pos = wordnet.ADV

else:

pos = wordnet.NOUN # Default to noun if part-of-speech is not recognized

return lemmatizer.lemmatize(word, pos=pos)

# Function to lemmatize a list of words with their corresponding part-of-speech tags

def lemmatize\_words(words, pos\_tags):

# Zip the words and pos\_tags lists together and lemmatize each word

lemmatized\_words = [lemmatize\_word(word, pos) for word, pos in zip(words, pos\_tags)]

return lemmatized\_words

# Example usage

words = ['running', 'cats', 'better', 'quickly']

pos\_tags = ['VBG', 'NNS', 'RBR', 'RB'] # Verb, plural noun, comparative adjective, adverb

# Lemmatize the list of words with their corresponding part-of-speech tags

lemmatized\_words = lemmatize\_words(words, pos\_tags)

print("Original Words:", words)

print("POS Tags:", pos\_tags)

print("Lemmatized Words:", lemmatized\_words)

**Output:**

Original Words: ['running', 'cats', 'better', 'quickly']

POS Tags: ['VBG', 'NNS', 'RBR', 'RB']

Lemmatized Words: ['run', 'cat', 'well', 'quickly']

**Explanation:**

1. We import necessary modules from NLTK: `WordNetLemmatizer` for lemmatization and `wordnet` for accessing WordNet corpus.

2. We create an instance of `WordNetLemmatizer`.

3. We define a function `lemmatize\_word` that takes a word and its part-of-speech (POS) as parameters.

4. Inside the function, we map the POS tag to WordNet's POS constants (`wordnet.ADJ`, `wordnet.VERB`, `wordnet.NOUN`, `wordnet.ADV`) based on the provided POS tag. If the tag is not recognized, we default to noun.

5. We then call the `lemmatize` method of `WordNetLemmatizer` with the word and its mapped POS to get the lemma.

6. We demonstrate the usage of the function with an example word 'running' and its POS tag 'VBG' (verb, gerund or present participle).

7. Finally, we print the lemma obtained from lemmatization.