BHARATIYA VIDYA BHAVAN'S

SARDAR PATEL INSTITUTE OF TECHNOLOGY

(Empowered Autonomous Institute Affiliated to University of Mumbai)

Department of Computer Engineering

Name	**Raj Kalpesh Mathuria**
UID no.	**2023300139**
Experiment No.	**2**

Aim:	1. Using Library perform comparative study of Porter/Snowball/Lancaster Stemmer and Stemmer vs Lemmatizer 2. Implement Porter Stemmer without library. (Use PennTreebank corpus)
Problem statement :	In Natural Language Processing, words appear in multiple morphological forms, increasing vocabulary size and reducing processing efficiency. The problem is to normalize words by reducing them to their base forms using stemming and lemmatization techniques, and to analyze the differences between standard library-based stemmers and a manually implemented Porter Stemmer.
Theory:	Stemming: Stemming is basically removing the suffix from a word and reducing it to its root word. It also involves producing morphological variants of a root/base word. For example: "Flying" is a word and its suffix is "ing", if we remove "ing" from "Flying" then we will get the base word or root word which is "Fly". We use these suffixes to create a new word from the original stem word. Sometimes spelling may also change in order to make a new word. 1. beauty, duty + -ful → beautiful, dutiful (-y changes to i) 2. heavy, ready + -ness → heaviness, readiness (-y changes to i) The main aim of stemming is to reduce the inflectional forms of each word into a common base word or root word or stem word. Inflection is a process of word formation, in which a word is modified to express different grammatical categories such as tense, case, voice, aspect, person, number, gender, mood, animacy, and definiteness Errors in Stemming: There are mainly two errors in stemming – Overstemming and Understemming. Overstemming occurs when two words

	are stemmed to the same root that are of different stems. Under-stemming occurs when two words are stemmed to the same root that are not of different stems. Porter Stemmer In linguistics (study of language and its structure), a stem is part of a word, that is common to all of its inflected variants. ▪ CONNECT ▪ CONNECTED ▪ CONNECTION ▪ CONNECTING Above words are inflected variants of CONNECT. Hence, CONNECT is a stem. To this stem we can add different suffixes to form different words. The process of reducing such inflected (or sometimes derived) words to their word stem is known as Stemming. For example, CONNECTED, CONNECTION and CONNECTING can be reduced to the stem CONNECT. The Porter Stemming algorithm (or Porter Stemmer) is used to remove the suffixes from an English word and obtain its stem which becomes very useful in the field of Information Retrieval (IR). This process reduces the number of terms kept by an IR system which will be advantageous both in terms of space and time complexity. This algorithm was developed by a British Computer Scientist named Martin F. Porter. You can visit the official home page of the Porter stemming algorithm for further information. Porter Stemmer Algorithm https://vijinimallawaarachchi.com/2017/05/09/porter-stemming-algorithm/ Input: Text Corpus of sufficient length. For example movie reviews, newspaper articles, etc. Output: 1. Using Library perform comparative study of Porter/Snowball/Lancaster Stemmer and Stemmer vs Lemmatizer 2. Implement Porter Stemmer without library.
Algorithm:	1. Import required NLTK and Pandas libraries.

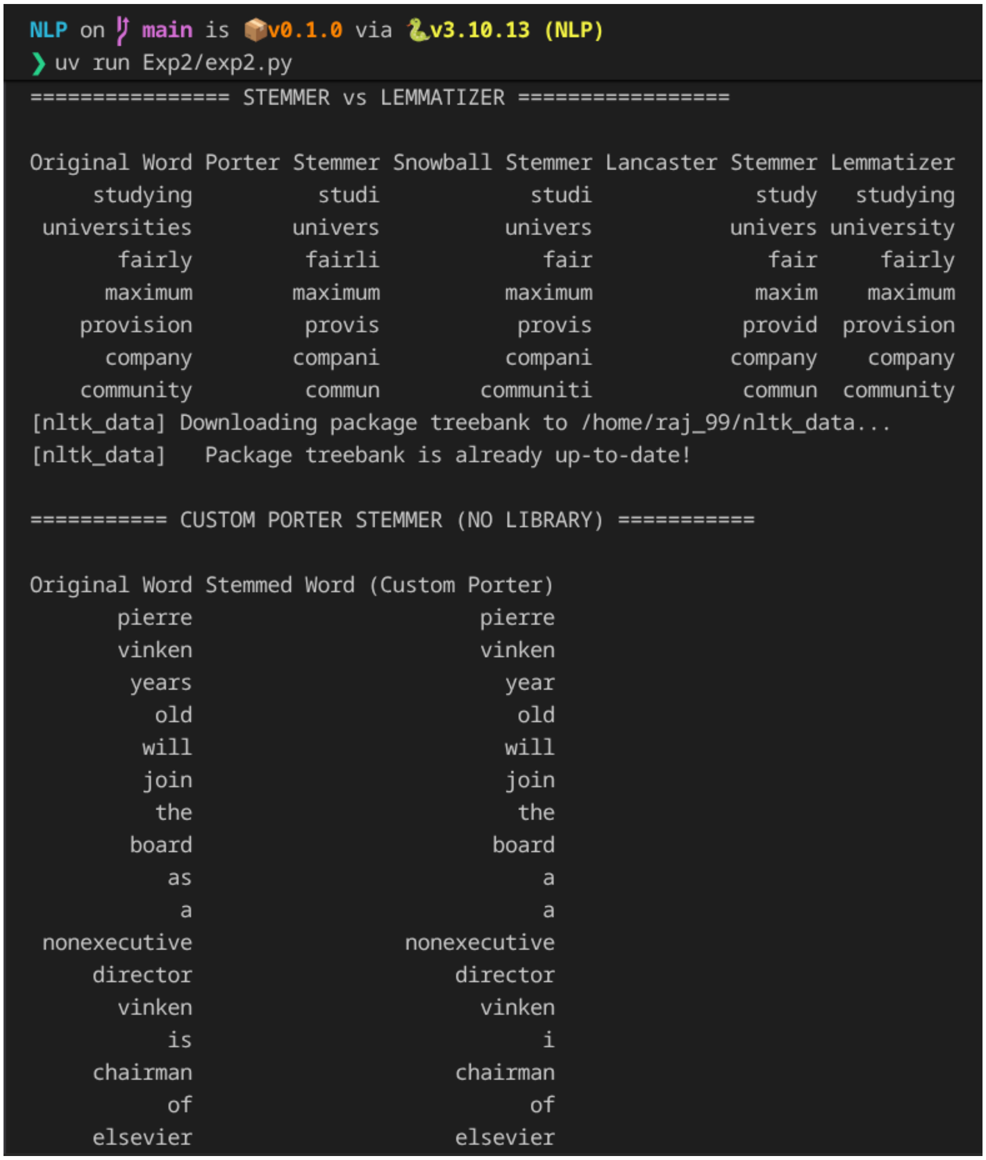
2. Define sample words for comparison.
3. Apply Porter, Snowball, Lancaster stemmers, and WordNet Lemmatizer to the words.
4. Store and display the results in a table.
5. Load the Penn Treebank corpus.
6. Implement a custom Porter Stemmer using suffix-removal rules and vowel–consonant patterns.
7. Apply the custom stemmer to corpus words and display the output.

Program:

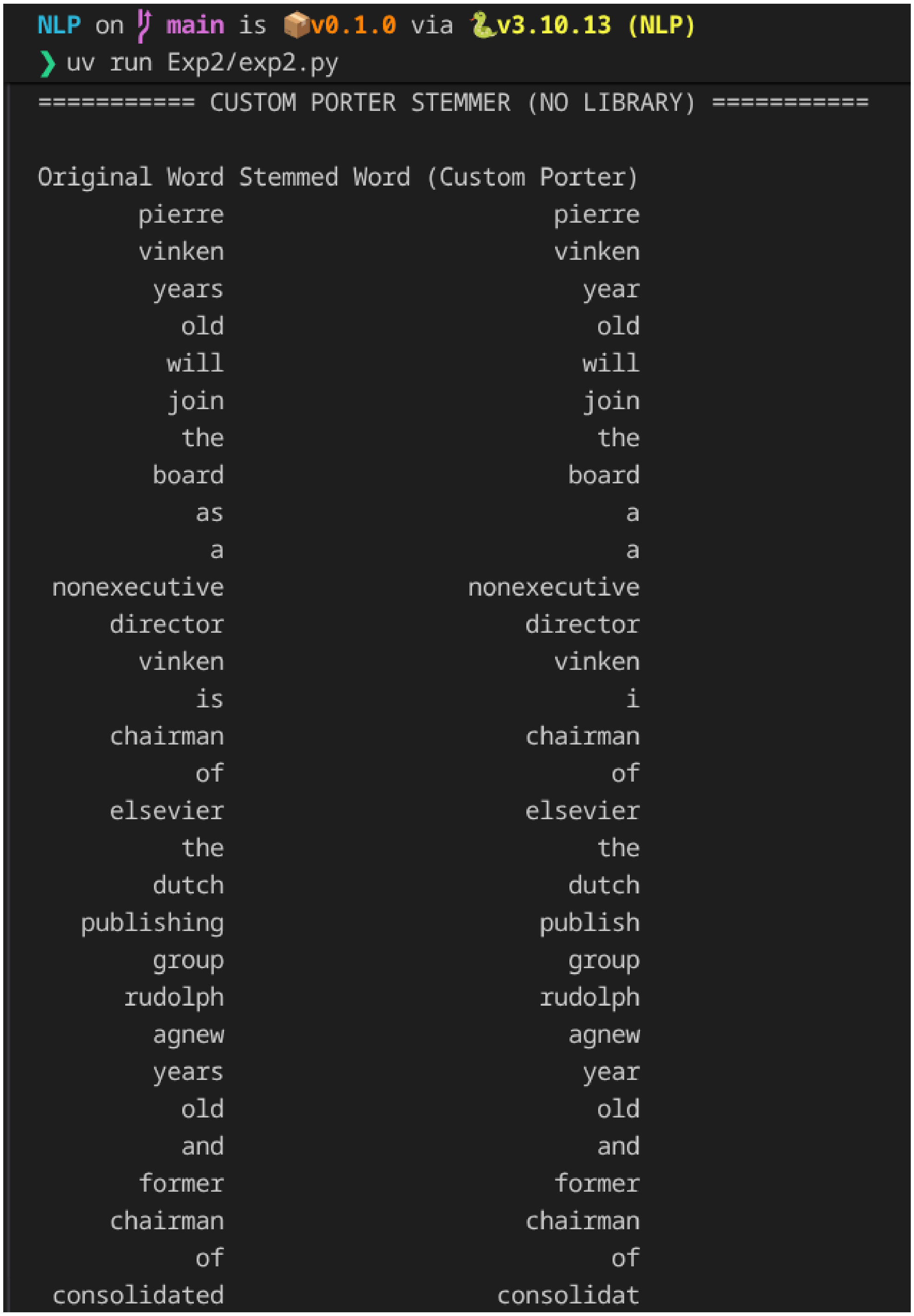
```
import nltk
import pandas as pd

from nltk.corpus import treebank
from nltk.stem import PorterStemmer, SnowballStemmer,
LancasterStemmer, WordNetLemmatizer

nltk.download('treebank')
nltk.download('wordnet')
nltk.download('omw-1.4')

words = [
"studying", "universities", "fairly",
"maximum", "provision", "company", "community"
]

porter = PorterStemmer()
snowball = SnowballStemmer("english")
lancaster = LancasterStemmer()
lemmatizer = WordNetLemmatizer()

comparison = []

for word in words:
comparison.append([
word,
porter.stem(word),
snowball.stem(word),
lancaster.stem(word),
lemmatizer.lemmatize(word)
])

df = pd.DataFrame(
comparison,
columns=[
"Original Word",
"Porter Stemmer",
"Snowball Stemmer",
"Lancaster Stemmer",
"Lemmatizer"
]
)

print("\n================ STEMMER vs LEMMATIZER
=================\n")
print(df.to_string(index=False))


# ---------------- DOWNLOAD DATA ----------------
nltk.download('treebank')

class CustomPorterStemmer:
def __init__(self):
self.vowels = "aeiou"

def is_consonant(self, word, i):
if word[i] in self.vowels:
return False
if word[i] == 'y':
return i == 0 or not self.is_consonant(word, i - 1)
return True

def measure(self, word):
m = 0
i = 0
length = len(word)

while i < length and self.is_consonant(word, i):
i += 1
```

```
while i < length:
while i < length and not self.is_consonant(word, i):
i += 1
if i < length:
m += 1
while i < length and self.is_consonant(word, i):
i += 1

return m

def contains_vowel(self, word):
return any(not self.is_consonant(word, i) for i in range(len(word)))

def step1a(self, word):
if word.endswith("sses"):
return word[:-2]
elif word.endswith("ies"):
return word[:-2]
elif word.endswith("ss"):
return word
elif word.endswith("s"):
return word[:-1]
return word

def step1b(self, word):
if word.endswith("eed"):
if self.measure(word[:-3]) > 0:
return word[:-1]
elif word.endswith("ed"):
stem = word[:-2]
if self.contains_vowel(stem):
return stem
elif word.endswith("ing"):
stem = word[:-3]
if self.contains_vowel(stem):
return stem
return word

def step1c(self, word):
if word.endswith("y") and self.contains_vowel(word[:-1]):
return word[:-1] + "i"
return word

def stem(self, word):
word = word.lower()
word = self.step1a(word)
word = self.step1b(word)
word = self.step1c(word)
return word


# ---------------- LOAD CORPUS ----------------
sentences = treebank.sents()

original_words = [
word.lower()
for sentence in sentences
for word in sentence
if word.isalpha()
]

original_words = original_words[:30]

custom_porter = CustomPorterStemmer()

stemmed_words = [custom_porter.stem(word) for word in original_words]

df = pd.DataFrame({
"Original Word": original_words,
"Stemmed Word (Custom Porter)": stemmed_words
})

print("\n=========== CUSTOM PORTER STEMMER (NO LIBRARY)
===========\n")
print(df.to_string(index=False))
```

Output:



```
NLP on  main is 📦v0.1.0 via 🐍v3.10.13 (NLP)
❯ uv run Exp2/exp2.py
================ STEMMER vs LEMMATIZER =================

Original Word Porter Stemmer Snowball Stemmer Lancaster Stemmer Lemmatizer
     studying          studi            studi             study   studying
 universities        univers          univers           univers university
       fairly         fairli             fair              fair     fairly
      maximum        maximum          maximum             maxim    maximum
    provision         provis           provis            provid  provision
      company        compani          compani           company    company
    community         commun        communiti            commun  community
[nltk_data] Downloading package treebank to /home/raj_99/nltk_data...
[nltk_data]   Package treebank is already up-to-date!

=========== CUSTOM PORTER STEMMER (NO LIBRARY) ===========

Original Word Stemmed Word (Custom Porter)
       pierre                       pierre
       vinken                       vinken
        years                         year
          old                          old
         will                         will
         join                         join
          the                          the
        board                        board
           as                            a
            a                            a
 nonexecutive                 nonexecutive
     director                     director
       vinken                       vinken
           is                            i
     chairman                     chairman
           of                           of
     elsevier                     elsevier
```



```
NLP on  main is 📦v0.1.0 via 🐍v3.10.13 (NLP)
❯ uv run Exp2/exp2.py
=========== CUSTOM PORTER STEMMER (NO LIBRARY) ===========

Original Word Stemmed Word (Custom Porter)
       pierre                       pierre
       vinken                       vinken
        years                         year
          old                          old
         will                         will
         join                         join
          the                          the
        board                        board
           as                            a
            a                            a
 nonexecutive                 nonexecutive
     director                     director
       vinken                       vinken
           is                            i
     chairman                     chairman
           of                           of
     elsevier                     elsevier
          the                          the
        dutch                        dutch
   publishing                      publish
        group                        group
      rudolph                      rudolph
        agnew                        agnew
        years                         year
          old                          old
          and                          and
       former                       former
     chairman                     chairman
           of                           of
 consolidated                   consolidat
```

Questions to be answered:

1. What is a paradigm class? Give an example.

A paradigm class is a group of words that share the same root and differ only by inflectional forms.

Example:
`study, studies, studying, studied` → all belong to the same paradigm class.

2. What are the applications of stemming?

- Information Retrieval (search engines)
- Text classification
- Sentiment analysis
- Document clustering
- Machine learning preprocessing in NLP

3. Is stemming applicable to all languages? What are the exceptions?

No, stemming is not equally effective for all languages.
It works well for morphologically simple languages like English.

Exceptions:

- Highly inflected languages (e.g., Arabic, Turkish, Finnish)
- Languages where meaning changes heavily with suffixes

In such cases, lemmatization or morphological analysis is preferred.

4. Explain over-stemming and under-stemming with example.

- Over-stemming: When different words are reduced to the same incorrect stem.
  Example:

	`university` and `universe` → `univers`
	• Under-stemming: When related words are not reduced to the same stem.
	Example:
	`study` and `studies` → `study`, `studi`

Conclusion:	The experiment demonstrates that stemming reduces words to their base form for efficient text processing. Porter and Snowball stemmers provide balanced results, while Lancaster stemmer is faster but aggressive. Lemmatization gives meaningful base words at higher computational cost. The custom Porter implementation validates the rule-based nature of stemming.