Name	Hatim Sawai
UID No.	2021300108
Experiment No.	2

Experiment 2

Aim	1. Generate word forms from root and suffix information using Add-Delete table. 2. Comparative study of Porter/Snowball/Lancaster Stemmer and Stemmer vs Lemmatizer

1. Installation of NLTK and downloading the required corpus

In [ ]:
```
import nltk
import pandas as pd
import matplotlib.pyplot as plt
from nltk.corpus import wordnet
from prettytable import PrettyTable
from nltk.tokenize import word_tokenize
from pattern.text.en import pluralize, conjugate, comparative
from nltk.stem import WordNetLemmatizer, PorterStemmer, LancasterStemmer, SnowballS
```

In [ ]:
```
nltk.download("punkt")
nltk.download("averaged_perceptron_tagger")
nltk.download("wordnet")
```

```
[nltk_data] Downloading package punkt to
[nltk_data]     C:\Users\hatim\AppData\Roaming\nltk_data...
[nltk_data]   Package punkt is already up-to-date!
[nltk_data] Downloading package averaged_perceptron_tagger to
[nltk_data]     C:\Users\hatim\AppData\Roaming\nltk_data...
[nltk_data]   Package averaged_perceptron_tagger is already up-to-
[nltk_data]       date!
[nltk_data] Downloading package wordnet to
[nltk_data]     C:\Users\hatim\AppData\Roaming\nltk_data...
[nltk_data]   Package wordnet is already up-to-date!
```

Out[ ]: True

2. Generating word forms from root using Add-Delete table

In [ ]:
```
words = ["run", "jump", "play", "sleep", "eat", "walk", "read", "write", "sing", "t
```

In [ ]:
```
# generate diiferent forms of words
def generate_wordforms(words):
    wordforms = []
    for word in words:
        forms = []
        forms.append(word)
        forms.append(pluralize(word))
        forms.append(comparative(word))
        forms.append(conjugate(word, tense="past"))
        forms.append(conjugate(word, tense="part"))
        wordforms.append(forms)
    return wordforms
```

In [ ]:
```
# add and delete table for each word form
def calc_add_del(wordform_arr):
    root = wordform_arr[0]
    table = PrettyTable()
    table.field_names = ["Root", "Del", "Add", "Chars", "Word Form"]
    table.align["Chars"] = "l"
    table.align["Word Form"] = "l"
    for i,form in enumerate(wordform_arr):
        if i == 0:
            continue
        elif form == root:
            wordform_arr[i] = [0, 0, "", form]
            table.add_row([root, 0, 0, "", form])
        else:
            # count equal chars break at first unequal
            count = 0
            for j,char in enumerate(root):
                if char == form[j]:
                    count += 1
                else:
                    break
            # count chars after first unequal
            add = len(form[count:])
            delete = len(root[count:])
            wordform_arr[i] = [delete, add, form[count:], form]
            table.add_row([root, delete, add, form[count:], form])
    print(table)
```

In [ ]:
```
wordforms = generate_wordforms(words)

for i, wordform in enumerate(wordforms):
    calc_add_del(wordforms[i])
```

Root	Del	Add	Chars	Word Form
run	0	1	s	runs
run	0	3	ner	runner
run	2	2	an	ran
run	0	4	ning	running

Root	Del	Add	Chars	Word Form
jump	0	1	s	jumps
jump	0	2	er	jumper
jump	0	2	ed	jumped
jump	0	3	ing	jumping

Root	Del	Add	Chars	Word Form
play	0	1	s	plays
play	0	2	er	player
play	0	2	ed	played
play	0	3	ing	playing

Root	Del	Add	Chars	Word Form
sleep	0	1	s	sleeps
sleep	0	2	er	sleeper
sleep	2	2	pt	slept
sleep	0	3	ing	sleeping

Root	Del	Add	Chars	Word Form
eat	0	1	s	eats
eat	0	2	er	eater
eat	3	3	ate	ate
eat	0	3	ing	eating

Root	Del	Add	Chars	Word Form
walk	0	1	s	walks
walk	0	2	er	walker
walk	0	2	ed	walked
walk	0	3	ing	walking

Root	Del	Add	Chars	Word Form
read	0	1	s	reads
read	0	2	er	reader
read	0	0		read
read	0	3	ing	reading

Root	Del	Add	Chars	Word Form
write	0	1	s	writes
write	0	1	r	writer
write	3	3	ote	wrote
write	1	3	ing	writing

Root	Del	Add	Chars	Word Form
sing	0	1	s	sings
sing	0	2	er	singer
sing	3	3	ung	sung
sing	0	3	ing	singing

Root	Del	Add	Chars	Word Form
think	0	1	s	thinks
think	0	2	er	thinker
think	3	5	ought	thought
think	0	3	ing	thinking

Root	Del	Add	Chars	Word Form
cry	1	3	ies	cries
cry	0	2	er	cryer
cry	1	3	ied	cried
cry	0	3	ing	crying

3. Comparative study of Porter/Snowball/Lancaster Stemmer

In [ ]:
```
# read setences from input.txt
with open("input.txt") as f:
    sentences = f.readlines()

# Tokenize the sentences
tokenized_sentences = [word_tokenize(sentence) for sentence in sentences]
```

In [ ]:
```
# Initialize stemmers and lemmatizer
porter_stemmer = PorterStemmer()
snowball_stemmer = SnowballStemmer("english")
lancaster_stemmer = LancasterStemmer()
lemmatizer = WordNetLemmatizer()
```

In [ ]:
```
# Stem each word in the sentences
stemmed_sentences = []
for sentence in tokenized_sentences:
    porter_stemmed = [porter_stemmer.stem(word) for word in sentence]
    snowball_stemmed = [snowball_stemmer.stem(word) for word in sentence]
    lancaster_stemmed = [lancaster_stemmer.stem(word) for word in sentence]
    stemmed_sentences.append(
        {
            "Porter": porter_stemmed,
            "Snowball": snowball_stemmed,
            "Lancaster": lancaster_stemmed,
        }
    )

# Lemmatize each word in the sentences
lemmatized_sentences = []
for sentence in tokenized_sentences:
    lemmatized = [lemmatizer.lemmatize(word, wordnet.VERB) for word in sentence]
    lemmatized_sentences.append(lemmatized)

# Print the results
comparetable = PrettyTable()
for i, sentence in enumerate(tokenized_sentences):
    print("\nSentence:", sentences[i])
    comparetable.field_names = ["Word", "Porter", "Snowball", "Lancaster", "Lemmati
    for j, word in enumerate(sentence):
        comparetable.add_row([word, stemmed_sentences[i]["Porter"][j], stemmed_sent
    print(comparetable)
    comparetable.clear_rows()
```

Sentence: The enigmatic detective carefully examined the mysterious crime scene.

Word	Porter	Snowball	Lancaster	Lemmatizer
The	the	the	the	The
enigmatic	enigmat	enigmat	enigm	enigmatic
detective	detect	detect	detect	detective
carefully	care	care	car	carefully
examined	examin	examin	examin	examine
the	the	the	the	the
mysterious	mysteri	mysteri	mystery	mysterious
crime	crime	crime	crim	crime
scene	scene	scene	scen	scene
.	.	.	.	.

Sentence: Her effervescent personality lights up the room, bringing joy to everyone around her.

Word	Porter	Snowball	Lancaster	Lemmatizer
Her	her	her	her	Her
effervescent	effervesc	effervesc	effervesc	effervescent
personality	person	person	person	personality
lights	light	light	light	light
up	up	up	up	up
the	the	the	the	the
room	room	room	room	room
,	,	,	,	,
bringing	bring	bring	bring	bring
joy	joy	joy	joy	joy
to	to	to	to	to
everyone	everyon	everyon	everyon	everyone
around	around	around	around	around
her	her	her	her	her
.	.	.	.	.

Sentence: A cacophony of sounds filled the bustling market as people went about thei r daily activities.

Word	Porter	Snowball	Lancaster	Lemmatizer
A	a	a	a	A
cacophony	cacophoni	cacophoni	cacophony	cacophony
of	of	of	of	of
sounds	sound	sound	sound	sound
filled	fill	fill	fil	fill
the	the	the	the	the
bustling	bustl	bustl	bustl	bustle
market	market	market	market	market
as	as	as	as	as
people	peopl	peopl	peopl	people

went	went	went	went	go
about	about	about	about	about
their	their	their	their	their
daily	daili	daili	dai	daily
activities	activ	activ	act	activities
.	.	.	.	.

Sentence: The resilient athlete overcame numerous obstacles to achieve victory in th
e championship.

Word	Porter	Snowball	Lancaster	Lemmatizer
The	the	the	the	The
resilient	resili	resili	resy	resilient
athlete	athlet	athlet	athlet	athlete
overcame	overcam	overcam	overcam	overcome
numerous	numer	numer	num	numerous
obstacles	obstacl	obstacl	obstac	obstacles
to	to	to	to	to
achieve	achiev	achiev	achiev	achieve
victory	victori	victori	vict	victory
in	in	in	in	in
the	the	the	the	the
championship	championship	championship	champ	championship
.	.	.	.	.

Sentence: The intricate details of the antique clock fascinated collectors and histo
rians alike.

Word	Porter	Snowball	Lancaster	Lemmatizer
The	the	the	the	The
intricate	intric	intric	int	intricate
details	detail	detail	detail	detail
of	of	of	of	of
the	the	the	the	the
antique	antiqu	antiqu	ant	antique
clock	clock	clock	clock	clock
fascinated	fascin	fascin	fascin	fascinate
collectors	collector	collector	collect	collectors
and	and	and	and	and
historians	historian	historian	hist	historians
alike	alik	alik	alik	alike
.	.	.	.	.

Sentence: An ethereal mist hung over the tranquil lake, creating a mystical atmosphe
re.

Word	Porter	Snowball	Lancaster	Lemmatizer

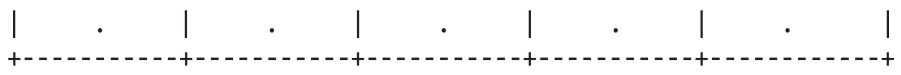
An	an	an	an	An
ethereal	ether	ether	eth	ethereal
mist	mist	mist	mist	mist
hung	hung	hung	ov	over
over	over	over	the	the
the	the	the	tranquil	tranquil
tranquil	tranquil	tranquil	lak	lake
lake	lake	lake	,	,
,	,	,	cre	create
creating	creat	creat	a	a
a	a	a	myst	mystical
mystical	mystic	mystic	atmosph	atmosphere
atmosphere	atmospher	atmospher	.	.
.	.	.		

Sentence: The voracious reader devoured books from various genres, expanding their k
nowledge.

Word	Porter	Snowball	Lancaster	Lemmatizer
The	the	the	the	The
voracious	voraci	voraci	vor	voracious
reader	reader	reader	read	reader
devoured	devour	devour	devo	devour
books	book	book	book	book
from	from	from	from	from
various	variou	various	vary	various
genres	genr	genr	genr	genres
,	,	,	,	,
expanding	expand	expand	expand	expand
their	their	their	their	their
knowledge	knowledg	knowledg	knowledg	knowledge
.	.	.	.	.

Sentence: The resilient tree withstood the force of the storm, standing tall against
adversity.

Word	Porter	Snowball	Lancaster	Lemmatizer
The	the	the	the	The
resilient	resili	resili	resy	resilient
tree	tree	tree	tre	tree
withstood	withstood	withstood	withstood	withstand
the	the	the	the	the
force	forc	forc	forc	force
of	of	of	of	of
the	the	the	the	the
storm	storm	storm	storm	storm
,	,	,	,	,
standing	stand	stand	stand	stand
tall	tall	tall	tal	tall
against	against	against	against	against
adversity	advers	advers	advers	adversity



5. Curiosity Questions

Q1. What is paradigm class? Give example.

Ans: In linguistics, a paradigm class refers to a set of words or forms that share a similar grammatical pattern or inflectional morphology. It involves grouping words based on their shared grammatical features, such as tense, number, gender, or case. Paradigm classes help linguists analyze and understand the systematic relationships between different forms of words within a language.
Example:
Consider the English verb "to be" in the present tense for different pronouns:

I am
You are
We are
He/She/It is

In this case, the paradigm class is formed by the variations of the verb "to be" based on different pronouns, and they share the same grammatical pattern within the present tense.

Q2. What are the different types of morphemes. Give example of each.

Ans: Morphemes are the smallest units of meaning in a language. There are two main types of morphemes: free morphemes and bound morphemes.
a. Free Morpheme:

A free morpheme is a morpheme that can stand alone as a word and carry meaning by itself.
Example:
"Dog" - In this case, "dog" is a free morpheme as it can stand alone and has a distinct meaning.

b. Bound Morpheme:

A bound morpheme is a morpheme that cannot stand alone as a word and needs to be attached to a free morpheme to convey meaning.
Example:
"Un-" (prefix meaning "not") - In the word "unhappy," "un-" is a bound morpheme because it cannot stand alone and needs to be attached to "happy" to convey the meaning of "not happy."

c. Inflectional Morpheme:

An inflectional morpheme is a type of bound morpheme that indicates grammatical information, such as tense, number, or gender.
Example:
"-s" (plural marker) in "cats" - The morpheme "-s" is inflectional as it changes the number of

the noun "cat" from singular to plural.

d. Derivational Morpheme:

A derivational morpheme is a type of bound morpheme that creates a new word or changes the meaning or grammatical category of a word.
Example:
"-er" (suffix forming agent nouns) in "teacher" - The morpheme "-er" is derivational as it changes the base word "teach" into a new word with a different grammatical category and meaning.

6. Conclusion

In this experiment we learned about the different types of morphemes and how to generate word forms from root and suffix information using Add-Delete table. We also learned about the comparative study of Porter/Snowball/Lancaster Stemmer and Stemmer vs Lemmatizer.