Practical no. 4

Name : Shivam Tawari
Roll no : A-58

Aim : Write a python program for Porter Stemmer without using NLTK Library.

Theory:

Porter Stemmer Algorithm:

It is one of the most popular stemming methods proposed in 1980. It is based on the idea that the suffixes in the English language are made up of a combination of smaller and simpler suffixes. This stemmer is known for its speed and simplicity.

The main applications of Porter stemmer include data mining and information retrieval. However, its applications are only limited to English words. Also, the group of stems is mapped on to the same stem and the

Output stem is not necessarily a meaningful word. The algorithms are fairly lengthy in nature and are known to be oldest stemmer.

Pro: It produces the best output as compared to other stemmers and it has less error rate.

Cons: Morphological variants produced are not always real words.

Pros and cons of programming without libraries:

Pros:

① You gain a deep understanding of what and how your code functions.

② You can customise the implementations.

Cons:

① Libraries are quick to setup and hook into the code.

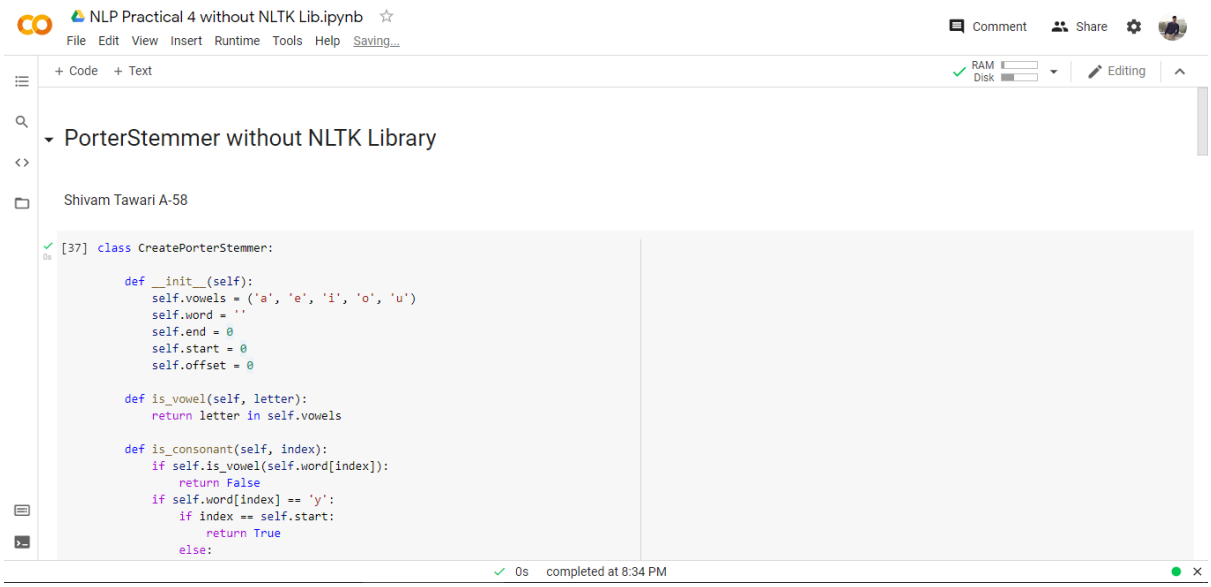
④ Reasonable guarantees that the functions will work as advertised.

Conclusion: Hence, we have successfully implemented the Porter Stemmer in python without NLTK libraries.

Practical – 4

Name: Shivam Tawari

Roll no: A-58



```
            return False
        if self.word[self.end - length + 1: self.end + 1] != s:
            return False
        self.offset = self.end - length
        return True

    def set_to(self, s):
        length = len(s)
        self.word = self.word[:self.offset + 1] + s + self.word[self.offset + length + 1:]
        self.end = self.offset + length

    def replace_morpheme(self, s):
        if self.m() > 0:
            self.set_to(s)

    def remove_plurals(self):
        if self.word[self.end] == 's':
            if self.ends_with("sses"):
                self.end = self.end - 2
            elif self.ends_with("ies"):
                self.set_to("i")
            elif self.word[self.end - 1] != 's':
                self.end = self.end - 1

        if self.ends_with("eed"):
            if self.m() > 0:
                self.end = self.end - 1
```

0s completed at 8:34 PM

```
    def stem_word(self, word):
        if word == '':
            return ''
        self.word = word
        self.end = len(word) - 1
        self.start = 0
        self.remove_plurals()
        return self.word[self.start: self.end + 1]
```

Examples:

caresses -> caress

ponies -> poni

ties -> ti

cats -> cat

meeting -> meet

```
[38] stemmer = CreatePorterStemmer()
     print(stemmer.stem_word('caresses'))
```

```
caress
```

```
[39] print(stemmer.stem_word('ponies'))
```

0s completed at 8:34 PM

```
[38] stemmer = CreatePorterStemmer()
     print(stemmer.stem_word('caresses'))
```

```
caress
```

```
[39] print(stemmer.stem_word('ponies'))
```

```
poni
```

```
[40] print(stemmer.stem_word('ties'))
```

```
ti
```

```
[41] print(stemmer.stem_word('cats'))
```

```
cat
```

```
[42] print(stemmer.stem_word('meetings'))
```

```
meet
```