

PROFESSIONAL TRAINING REPORT
at
Sathyabama Institute of Science and Technology
(Deemed to be University)

Submitted in partial fulfillment of the requirements for the award of
Bachelor of Technology Degree in

Information Technology

By

Sakthivarshini K (38120071)



DEPARTMENT OF INFORMATION TECHNOLOGY
SCHOOL OF COMPUTING
SATHYABAMA INSTITUTE OF SCIENCE AND TECHNOLOGY
JEPPIAAR NAGAR, RAJIV GANDHI SALAI,
CHENNAI – 600119. TAMILNADU.
NOV 2021



**SATHYABAMA INSTITUTE OF SCIENCE AND
TECHNOLOGY**

(Established under Section 3 of UGC Act, 1956)
Jeppiaar Nagar, Rajiv Gandhi Salai, Chennai -
600119

www.sathyabama.ac.in



SCHOOL OF COMPUTING

BONAFIDE CERTIFICATE

This is to certify that this Professional Training Report is the bonafide work of
SAKTHIVARSHINI.K(38120071) who carried out the project entitled “**Tokenizing and
Normalizing text using Machine Learning**” under our supervision from Jun 2021 to Nov
2021.

Internal Guide

Dr. V.Nirmalrani M.E., Ph.D.,

Head of the Department

Dr. R.SUBHASHINI M.E., Ph.D.,

Submitted for Viva voce Examination held on _____

Internal Examiner

External Examiner

DECLARATION

I, Sakthivarshini.K(38120071) hereby declare that the Professional Training Report on **“Tokenizing and Normalizing text using Machine Learning”** done by me under the guidance of **Dr.V.NIRMALRANI M.E., Ph.D.**, at Sathyabama Institute of Science and Technologies submitted in partial fulfillment of the requirements for the award of Bachelor of Technology degree in Information Technology .

DATE:

PLACE: CHENNAI

SIGNATURE OF THE CANDIDATE

ACKNOWLEDGEMENT

I am pleased to acknowledge my sincere thanks to **Board of Management of SATHYABAMA** for their kind encouragement in doing this project and for completing it successfully. I am grateful to them.

I convey my thanks to **Dr. T. Sasikala M.E., Ph.D, Dean, School of Computing, Dr. R. Subhashini, M.E., Ph.D. Head of the Department of Information Technology** for providing me necessary support and details at the right time during the progressive reviews.

I would like to express my sincere and deep sense of gratitude to my Project Guide **Dr. V.Nirmalrani M.E., Ph.D.,** for her valuable guidance, suggestions and constant encouragement paved way for the successful completion of my project work.

I wish to express my thanks to all Teaching and Non-teaching staff members of the **Department of Information Technology** who were helpful in many ways for the completion of the project.

TRAINING CERTIFICATE



ABSTRACT

The stream of characters in a natural language text must be broke into distinct meaningful units (or tokens) before any language processingbeyond the character level can be performed. If languages were perfectly punctuated,this would be a trivial thing to do: a simple program could separate the text into word and punctuation tokens simply by breaking it up at white-space and punctuation marks. And text Normalization is an important step in Natural Language Processing. It plays an important role in various applications such as text-to-speech systems, speech recognition, email voice activators, for visually impaired persons and also for educational purpose. Text normalization means converting the non-standard words into their standard formats (i.e. in the string format or plain text format). The normalization of text is required for words such as dates, currency symbols, abbreviations, acronyms, and numbers etc. to be pronounced or read by the system.. This project presents an overview of work done so far in the field of text normalization and tokenizing using machine learning

TABLE OF THE CONTENT

CHAPTER	TITLE NAME	PGNO
	ABSTRACT	
	LIST OF ABBREVIATONS	
1	INTRODUCTION	1
2	AIM	1
2.1	SCOPE OF THE PROJECT	1
3	OPERATIONAL CONCEPTS	2
3.1	CORPUS	2
3.2	TOKENS	2
3.3	N-GRAMS	2
3.4	TOKENIZATION	3
3.5	NORMALIZATION	4
3.5.1	STEMMING	5
3.5.2	LEMMATIZATION	6
4	SYSTEM REQUIREMENT	6
4.1	HARDWARE SPECIFICATION	7
4.2	SOFTWARE SPECIFICATION	7
5	CODING SECTION AND SCREENSHOTS	8
6	CONCLUSION	16

7	REFERENCE	16
---	-----------	----

LIST OF ABBREVIATIONS

ABBREVIATION

EXPANSION

NLTK

Natural Language ToolKit

NLP

Natural Language Processing

1.INTRODUCTION

Text data is a form of unstructured data. The most prominent examples of text data available on the internet are social media data like tweets, posts, comments, or the Conversation data such as messages, emails, Chats. Also, it can be article data like news articles, blogs, etc.

So Text data is essentially a written form of a natural language such as Hindi, English, Russian, or any other. It consists of characters or words arranged together in a meaningful and ordered manner. This means that text data is driven by grammar rules. and defined structures.

In order to work with text data, it is important to transform the raw text into a form that can be understood and used by machine learning algorithms, this is called text preprocessing.

In this project, we are going to discuss different terms and techniques related to natural language processing.

First Let's discuss the meaning of some important terms like Corpus, Token, and N-grams.

2. AIM

The main aim of this project is that the text data is a type of unstructured data used in NLP Understand how to preprocess the text data before feeding it to the machine learning algorithms.

2.1 SCOPE OF THE PROJECT

Text data is a type of unstructured data used in natural language processing. This project is help to understand how to preprocess the text data before feeding it to the machine learning algorithms.

3 OPERATIONAL CONCEPTS

3.1 CORPUS

Corpus is the collection of text documents. For example, a dataset consists of the news article in a corpus. Similarly, Twitter data containing tweets is a corpus. So Corpus consists of Documents, Documents contain Paragraphs in turn Paragraph consists of Sentences and finally, Sentences comprises of Tokens.

3.2 TOKENS

Tokens are a basic meaningful unit of a sentence or a document. They can consist of words, phrases, subwords like n-grams, or character.

3.3 N-GRAMS

N-grams is a combination of N words or characters together. for example, if we have a sentence "I Love My Phone" we can break this sentence into multiple n-grams as shown below.

Sentence : I love my phone

Unigrams (n =1) : I, Love, my, phone

Bigrams (n=2) : I Love, Love my, my phone

Trigrams (n=3) : I love my, love my phone

Unigrams have only one token like I, love. whereas bigrams will have a combination of tokens in a sequence, for example, I Love, Love MY. Similarly, trigrams will have a combination of three tokens together.

N-grams are very useful in text classification tasks.

Now we have a clear idea about the basic terms. Let's see the few techniques used in text data preprocessing.

3.4 TOKENIZATION

Tokenization is the process of splitting a text object into smaller units known as tokens. Examples of tokens can be words, characters, numbers, symbols, or n-grams.

The most common tokenization process is whitespace/ unigram tokenization. In this process entire text is split into words by splitting them from whitespaces. As shown in the example below the whole sentence is split into unigrams i.e "I", "Went", "To", "New-York" etc.

White space tokenizer / Unigram tokenizer

Sentence : "I went to New-York to play football"

Tokens : "I", "went", "to", "New-York", "to", "play", "football"

New-York is not split further as the tokenization is based on whitespaces only.

Another type of Tokenization is regular expression tokenization. In which a regular expression pattern is used to get the tokens. For example, consider the following string containing multiple delimiters. We can split the sentence by passing a splitting pattern.

Regular expression tokenizer

Sentence : "Football,Cricket;Golf Tennis"

`re.split(r'[;,]', line)`

Tokens : "Football", "Cricket", "Golf", "Tennis"

The tokenization can be performed at the sentence level or at the word level or even at the character level.

3.5 NORMALIZATION

In the field of linguistics and NLP, Morpheme is defined as a base form of the word. A token is basically made up of two components one is morphemes and the other is inflectional form like prefix or suffix.

For example, consider the word **Antinationalist (Anti + national+ ist)** which is made up of Anti and ist as inflectional forms and national as the morpheme.

Normalization is the process of converting a token into its base form. In the normalization process, the inflectional form of a word is removed so that the base form can be obtained. So in our above example, the normal form of antinationalist is national.

Normalization is helpful in reducing the number of unique tokens present in the text, removing the variations in a text. and also cleaning the text by removing redundant information.

Two popular methods used for normalization are stemming and lemmatization. Let's discuss them in detail.

3.5.1 STEMMING

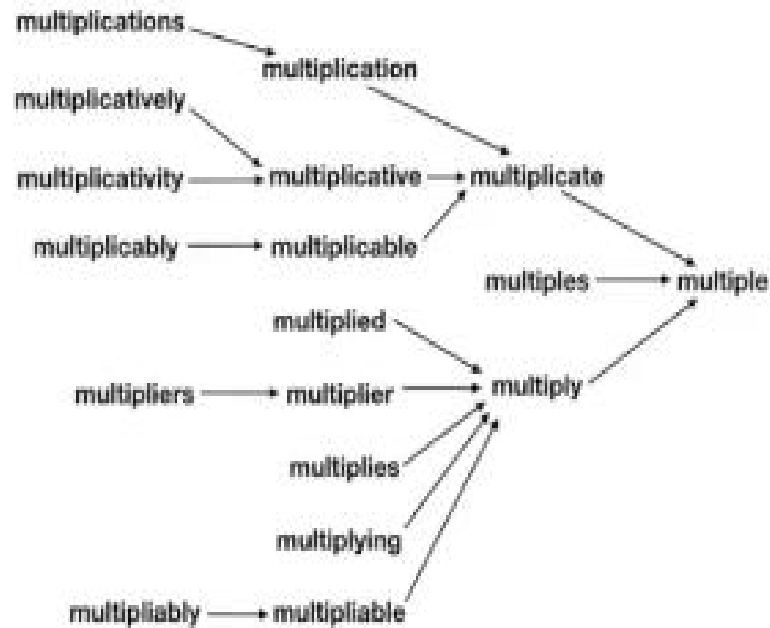
It is an elementary rule-based process for removing inflectional forms from a given token. The output of the process is the stem of a word. For example laughing, laughed, laughs, laugh all will become laugh after the stemming process.

Form	Suffix	Stem
stud ies	-es	studi
stud y ing	-ing	study
niñ as	-as	niñ
niñ ez	-ez	niñ

Stemming is not a good process for normalization. Since sometimes it can produce non-meaningful words which are not present in the dictionary. Consider the sentence "His teams are not winning". After stemming we get "Hi team are not winn". Notice that the keyword winn is not a regular word. Also, "hi" has changed the context of the entire sentence.

3.5.2 LEMMATIZATION

Lemmatization is a systematic process of removing the inflectional form of a token and transform it into a lemma. It makes use of word structure, vocabulary, part of speech tags, and grammar relations.



The output of lemmatization is a root word called a lemma. for example “am”, “are”, “is” will be converted to “be”. Similarly, running runs, ‘ran’ will be replaced by ‘run’.

Also, since it is a systematic process while performing lemmatization one can specify the part of the speech tag for the desired term.

Further, Lemmatization can only be performed if the given word has proper parts of speech tag. for instance if we try to lemmatize the word running as a verb it will be converted to run. But, if we try to lemmatize the same word running as a noun it won’t be transformed.

4 SYSTEM REQUIREMENT

4.1 HARDWARE SPECIFICATION

- RAM-2GB
- Hard disk
- Processor-Intel Pentium 4
- Keyboard

4.2 SOFTWARE SPECIFICATION

- Window OS
- Pycharm community Edition

5.CODING SECTION AND SCREENSHOTS

MANUAL TOKENIZING

Load data

```
# load text
```

```
filename = 'metamorphosis_clean.txt'
```

```
file = open(filename, 'rt')
```

```
text = file.read()
```

```
file.close()
```

Split by whitespace

```
# load text
```

```
filename = 'metamorphosis_clean.txt'
```

```
file = open(filename, 'rt')
```

```
text = file.read()
```

```
file.close()
```

```
# split into words by white space
```

```
words = text.split()
```

```
print(words[:100])
```

Select words

```
# load text
```

```
filename = 'metamorphosis_clean.txt'
```

```
file = open(filename, 'rt')
```

```
text = file.read()
```

```
file.close()
```

```
# split based on words only
```

```
import re
```

```
words = re.split(r'\W+', text)
```

```
print(words[:100])
```

Remove punctuation

```
print(string.punctuation)
```

```
table = str.maketrans("", "", string.punctuation)
```

```
# load text
```

```
filename = 'metamorphosis_clean.txt'
```

```
file = open(filename, 'rt')
```

```
text = file.read()
```

```
file.close()
```

```
# split into words by white space
```

```
words = text.split()

# remove punctuation from each word

import string

table = str.maketrans("", "", string.punctuation)

stripped = [w.translate(table) for w in words]

print(stripped[:100])
```

NORMALIZING CASE

```
filename = 'metamorphosis_clean.txt'

file = open(filename, 'rt')

text = file.read()

file.close()

# split into words by white space

words = text.split()

# convert to lower case

words = [word.lower() for word in words]

print(words[:100])
```

TOKENIZATION AND CLEANING WITH NLTK

INSTALL NLTK

```
sudo pip install -U nltk
```

```
import nltk
```

```
nltk.download()
```

```
python -m nltk.downloader all
```

SPLIT INTO SENTENCES

```
# load data
```

```
filename = 'metamorphosis_clean.txt'
```

```
file = open(filename, 'rt')
```

```
text = file.read()
```

```
file.close()
```

```
# split into sentences
```

```
from nltk import sent_tokenize
```

```
sentences = sent_tokenize(text)
```

```
print(sentences[0])
```

SPLIT INTO WORDS

```
# load data
```

```
filename = 'metamorphosis_clean.txt'
```

```
file = open(filename, 'rt')
```

```
text = file.read()
```

```
file.close()
```

```
# split into words
```

```
from nltk.tokenize import word_tokenize
```

```
tokens = word_tokenize(text)
```

```
print(tokens[:100])
```

FILTER OUT PUNCTUATION

```
filename = 'metamorphosis_clean.txt'
```

```
file = open(filename, 'rt')
```

```
text = file.read()
```

```
file.close()
```

```
from nltk.tokenize import word_tokenize
```

```
tokens = word_tokenize(text)
```

```
# remove all tokens that are not alphabetic
```

```
words = [word for word in tokens if word.isalpha()]
```

```
print(words[:100])
```

FILTER OUT STOP WORDS

```
from nltk.corpus import stopwords
```

```
stop_words = stopwords.words('english')
```

```
print(stop_words)
```

```
# load data
```

```
filename = 'metamorphosis_clean.txt'
```

```
file = open(filename, 'rt')
```

```
text = file.read()
```

```
file.close()
```

```
# split into words
```

```
from nltk.tokenize import word_tokenize
```

```
tokens = word_tokenize(text)
```

```
# convert to lower case

tokens = [w.lower() for w in tokens]

# remove punctuation from each word

import string

table = str.maketrans("", "", string.punctuation)

stripped = [w.translate(table) for w in tokens]

# remove remaining tokens that are not alphabetic

words = [word for word in stripped if word.isalpha()]

from nltk.corpus import stopwords

stop_words = set(stopwords.words('english'))

words = [w for w in words if not w in stop_words]

print(words[:100])
```

STEM WORDS

```
filename = 'metamorphosis_clean.txt'

file = open(filename, 'rt')

text = file.read()

file.close()

# split into words

from nltk.tokenize import word_tokenize

tokens = word_tokenize(text)

# stemming of words

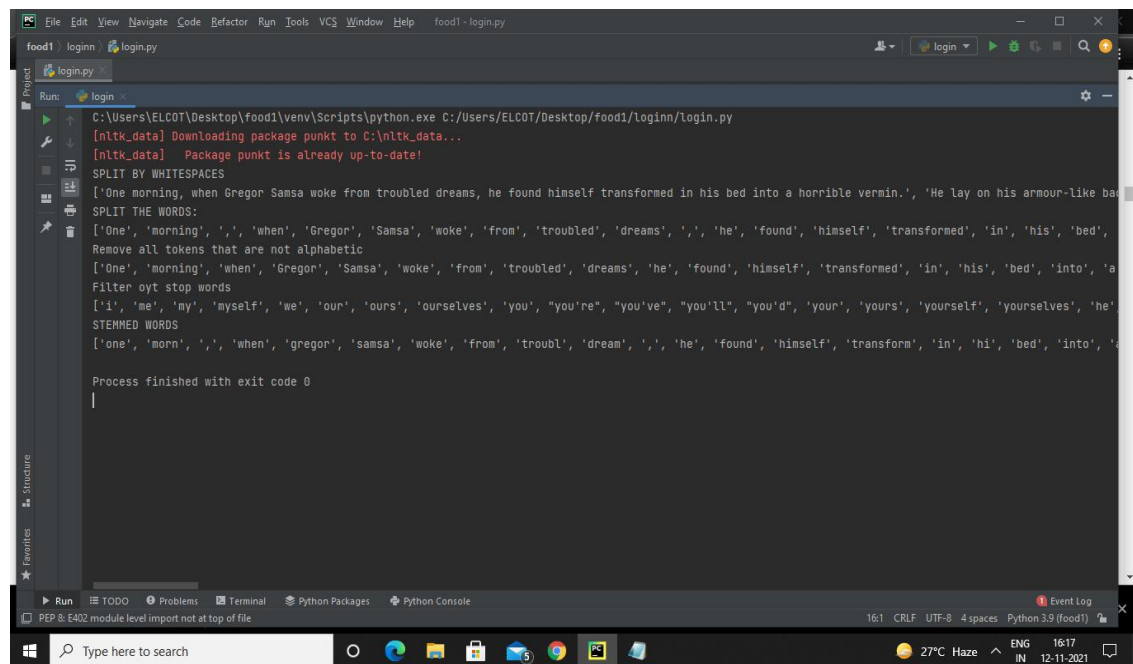
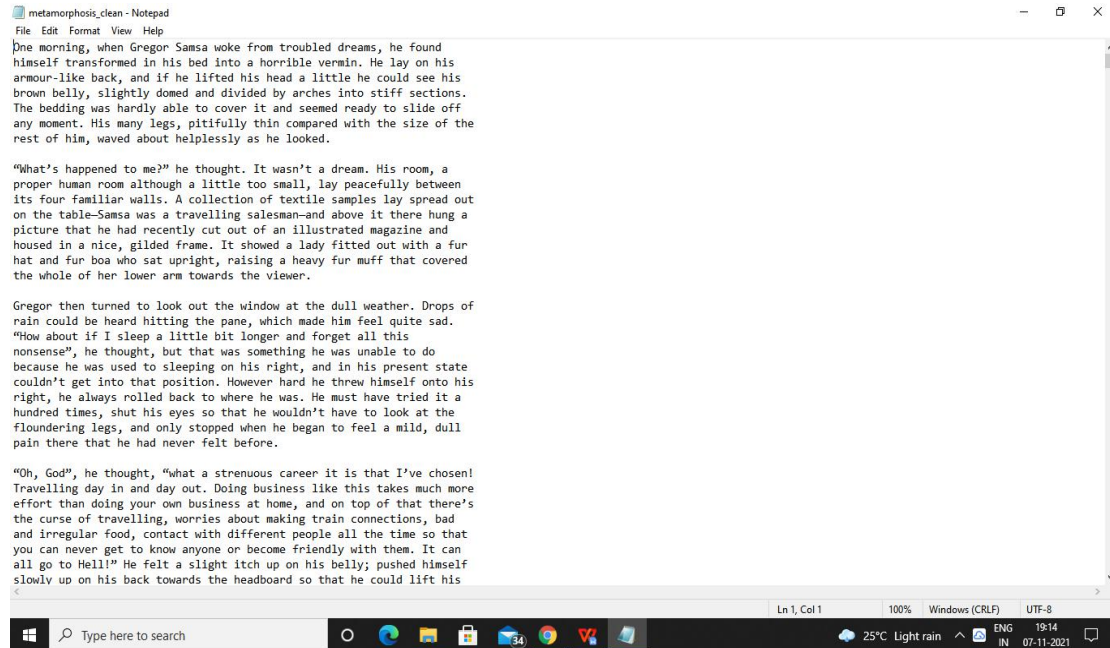
from nltk.stem.porter import PorterStemmer

porter = PorterStemmer()

stemmed = [porter.stem(word) for word in tokens]

print(stemmed[:100])
```

SCREENSHOTS



6 .CONCLUSION

This is all about tokenization and text normalization in this project. To summarize in this project we saw the techniques used for text data preprocessing. Further, we saw the text normalization using stemming and lemmatization and also how these two are different from each other.

7 .REFERENCE

<https://www.javatpoint.com/machine-learning>