Logo

Description automatically generated

INDIVIDUAL ASSIGNMENT

TECHNOLOGY PARK MALAYSIA

CT107-3-3-TXSA

TEXT ANALYTICS AND SENTIMENT ANALYSIS

Marcell Agung Wahyudi (TP058650)

APU3F2211CS(DA)

HAND OUT DATE: 18 February 2023

HAND IN DATE: 28 February 2023

WEIGHTAGE: 25%

INSTRUCTIONS TO CANDIDATES:

1. Submit your assignment at the Moodle System.
2. Students are advised to underpin their answers with the use of references (cited using the APA Style System of Referencing).
3. Late submission will be awarded zero (0) unless Extenuating Circumstances (EC) are upheld.
4. Cases of plagiarism will be penalized.
5. You must obtain 50% overall to pass this module.

Table of Contents

[Question 1: Form Tokenization 2](#_Toc127544049)

[1.1 Sentence Segmentation 2](#_Toc127544050)

[1.2 Tokenization using split, Regular Expression, and NLTK 4](#_Toc127544051)

[1.3 Differences of Tokenization Operations 5](#_Toc127544052)

[1.4 Justification for the most suitable tokenization method 6](#_Toc127544053)

[Question 2: Form Word Stemming 6](#_Toc127544054)

[2.1 Importance of Stemming 6](#_Toc127544055)

[2.2 Difference among RegEx Stemmer and Porter Stemmer 6](#_Toc127544056)

[2.3 Ouput of RegEx or Porter Stemmer 7](#_Toc127544057)

[2.4 Justification of the most suitable stemming method 8](#_Toc127544058)

[Question 3: Filter Stop Words and Punctuation 9](#_Toc127544059)

[3.1 The following code is to filter stop words and punctuation: 9](#_Toc127544060)

[3.2 Stop Words Found in the given text 10](#_Toc127544061)

[3.3 Importance of Stop Words and Punctuations Filtering 10](#_Toc127544062)

[Question 4: Form Parts of Speech (POS) Taggers && Syntactic Analyzers 11](#_Toc127544063)

[4.1 Demonstration of POS tagging (NLTK Pos Tagger, RegEx Tagger) and Report of the output 11](#_Toc127544064)

[4.2 Difference between NLTK Pos Tagger and RegEx Pos Tagger 12](#_Toc127544065)

[4.3 Most suitable POS Tagger for Text Analysis 12](#_Toc127544066)

[4.3 Parse Tree Drawings 13](#_Toc127544067)

# Question 1: Form Tokenization

## Sentence Segmentation

To do sentence segmentation, the NLTK library is utilized, the following code is for importing the NLTK library along with other useful functions:

Text

Description automatically generated

Next step is to import the text file and turn it into a program readable text with the following code:

Text

Description automatically generated

Then, we can segment the sentence using the following code:

Graphical user interface

Description automatically generated with medium confidence

All that’s left Question 1 is to output, and to organize everything, enumerate function is used to easily order the sentences:

Text

Description automatically generated

## Tokenization using split, Regular Expression, and NLTK

To begin tokenization, first is to import all the needed libraries:



Split Tokenization

Then, simply split the text using the following syntax:

Text, letter

Description automatically generated

Regular Expression

Second option is the RegEx Tokenization, using the following code:

Background pattern

Description automatically generated

NLTK Tokenization

For NLTK tokenization, the following syntax is used:

Text, letter

Description automatically generated

## Differences of Tokenization Operations

From the output seen from chapter 1.2, split function and NLTK tokenization has almost similar result, with the slight difference that NLTK splits the last word “fruit?” into [‘fruit’, ‘?’], whereas Split function does not split the “fruit?”, the other tokenization method, the RegEx, splits the words into single characters, this might be useful if the user were to find special characters.

## Justification for the most suitable tokenization method

Comparing all tokenization methods discussed, since Split function runs slightly slower than NLTK, the better choice for general word tokenization is the NLTK technique, but, since text analysis may have many characters to be analyzed, Regular Expression actually the best choice for tokenization as it is the fastest from all three methods and also the most used.

That being said, any methods can be used depending on the situation at hand, split is better if used to solve simpler problems as it is more simpler to use and light in memory space, NLTK is better organized and statistically faster than split function, but in terms of more complex situations, RegEx is the most commonly used method.

# Question 2: Form Word Stemming

## 2.1 Importance of Stemming

Stemming is a cutting-edge technique for shortening words. In the case of computer science, each complex word contains the simplest root (stem) words as well as corresponding suffixes and prefixes as the extensive head and tails of the main word. Stemming refers to the process of cutting the long tops and bottoms to find the root word. Human language is an unresolved issue, with over 6500 languages spoken around the world. Tons of data are generated every day as we speak, text, tweet, and convert speech to text on each social media application, and we need technology like NLP to gain insights from this text data (Agrawal, 2021).

## 2.2 Difference among RegEx Stemmer and Porter Stemmer

The difference between RegEx Stemmer, Porter Stemmer, and Lancaster Stemmer is visible. Each character's suffixes are removed by the RegEx Stemmer. Each word is reduced to its normal form, and certain characters are removed, rendering the words meaningless. The Porter Stemmer, in contrast hand, removes the prevalent ends out of each word, making some words less impactful.

## 2.3 Ouput of RegEx or Porter Stemmer

RegEx

The following code snippet is the syntax to run Regular Expression Stemmer:Text, letter

Description automatically generated

Porter Stemmer

The following code snippet is the syntax to run Porter Stemmer:



## 2.4 Justification of the most suitable stemming method

For the justification of the most suitable stemming method, RegEx is seen to have reduced the words too much to the point that some words completely lost it’s meaning such as “is” becoming only “i”, whereas Porter Stemmer stems the words in a way that some words still keep its meanings, making Porter Stemmer the better choice.

# Question 3: Filter Stop Words and Punctuation

## 3.1 The following code is to filter stop words and punctuation:

Text

Description automatically generatedGraphical user interface, text, application

Description automatically generated

The output of the shown code are as follows:

Scatter chart

Description automatically generated

## 3.2 Stop Words Found in the given text

The following are the stop words found in the given text of Data\_1:

Text, letter

Description automatically generated

## 3.3 Importance of Stop Words and Punctuations Filtering

English, being most common language in the world, primarily for social media platforms, uses considerably useless words for text analysis, such as “i”, “me”, “myself”, etc. which don’t bring value for text analysis. Filtering stop words drastically helps the job of text analysis to only analyze and funnel in major important key words which carry information making text analysis more efficient (Singhal, 2020).

# Question 4: Form Parts of Speech (POS) Taggers & Syntactic Analyzers

## 4.1 Demonstration of POS tagging (NLTK Pos Tagger, RegEx Tagger) and Report of the output

The following code is NLTK Pos Tagger with the output:

Text

Description automatically generated

The next code is for RegEx Tagger and it’s output:

Text

Description automatically generated

## 4.2 Difference between NLTK Pos Tagger and RegEx Pos Tagger

From the output seen from the previous chapter, not much difference was found other than the format the two POS tagger method shown. For NLTK, it is easier for user to read since it is more organized, whereas RegEx POS tagger is more scrambled and more confusing for users to read.

## 4.3 Most suitable POS Tagger for Text Analysis

Since the output for this particular txt file is almost the same, the chosen suitable POS Tagger for Text Analysis is more towards NLTK since it is more organized and easier to read and also is said to have higher accuracy than RegEx POS tagger.

## 4.4 Parse Tree Drawings

A picture containing text

Description automatically generated

# Bibliography

Agrawal, K. (2021, June 11). *Natural Language Processing*. Retrieved from WHAT IS TEXT STEMMING IN NLP?: https://www.linkedin.com/pulse/what-text-stemming-nlp-kavya-kumar

Singhal, G. (2020, October 05). *Importance of Text Pre-processing*. Retrieved from Pluralsight: https://www.pluralsight.com/guides/importance-of-text-pre-processing