Natural Language processing

TP 1 – Prétraitement

L'objectif de ce TP est d'implémenter quelques techniques de prétraitement sur des textes standard.

Tokenisation

1. Importer la libraire python pour faire la tokenisation

```
#Tokenizing sentences
import nltk
nltk.download('punkt_tab')
```

2. Introduire un texte en anglais, avec plusieurs paragraphes. Voici un exemple :

   text = "Backgammon is one of the oldest known board games. Its history can be traced back nearly 5,000 years to archeological discoveries in the Middle East. It is a two-player game where each player has fifteen checkers which move between twenty-four points according to the roll of two dice."

3. Tokenisation des phrases (séparées par un point)

```
#tokenizing by sentence (taking . as separator)
sentences = nltk.sent_tokenize(text)
for sentence in sentences:
    print(sentence)
    print()
```

4. Tokenisation des mots (séparés par un espace)

```
#Tokenizing words
for sentence in sentences:
    #tokenizing by words (taking space as the separator)
    words = nltk.word_tokenize(sentence)
    print(words)
    print()
```

Question :

- **Prenez un texte aléatoire en Français et faites le même process de tokenisation avec un point et un espace.**
- **Si vous prenez autre caractère pour faire la tokenisation, quel est la sortie ?**

Lemmanisation et Stemming

1. Importer les librairies

```
from nltk.stem import PorterStemmer, WordNetLemmatizer
from nltk.corpus import wordnet
nltk.download('wordnet')
nltk.download('omw-1.4')
```

2. Définir une fonction pour imprimer les résultats et initialiser les stemming et la lemmanisation

```
#function to print the results
def compare_stemmer_and_lemmatizer(stemmer, lemmatizer, word, pos):
    """
    Print the results of stemmind and lemmitization using the passed stemmer, lemmatizer, word and pos (part of speech)
    """
    print("Stemmer:", stemmer.stem(word))
    print("Lemmatizer:", lemmatizer.lemmatize(word, pos))
    print()


#Initialize stemmer and lemmatizer
lemmatizer = WordNetLemmatizer()
stemmer = PorterStemmer()
```

3. Essayer quelques mots en anglais
    - **Qu'est qui se passe si l'on fait une erreur de frappe sur un mot ?**
    - **Quel est le but du paramètre pos ? essayer autres valeurs afin de voir les sorties obtenues**

```
#comparing the result on both techniques
compare_stemmer_and_lemmatizer(stemmer, lemmatizer, word = "seen", pos = wordnet.VERB)
compare_stemmer_and_lemmatizer(stemmer, lemmatizer, word = "drove", pos = wordnet.VERB)
compare_stemmer_and_lemmatizer(stemmer, lemmatizer, word = "increible", pos = wordnet.VERB)
compare_stemmer_and_lemmatizer(stemmer, lemmatizer, word = "understood", pos = wordnet.VERB)
```

Stopwords

1. Importer la librairie nécessaire

```
from nltk.corpus import stopwords
nltk.download('stopwords')
```

2. Visualiser les stopwords en français et en anglais

```
print(stopwords.words("french"))
print(stopwords.words("english"))
```

3. Appliquer la fonction stopwords a une phrase en anglais (essayer plusieurs phrases)

```
stop_words = set(stopwords.words("english"))
sentence = "Backgammon is one of the oldest known board games."

words = nltk.word_tokenize(sentence)
without_stop_words = [word for word in words if not word in stop_words]
print(without_stop_words)
```

4. Appliquer la fonction stopwords a une phrase en français

```
stop_words = set(stopwords.words("french"))
sentence = "Dans cet article, je vais passer en revue la majorité des principaux modèles de Machine Learning

words = nltk.word_tokenize(sentence)
without_stop_words = [word for word in words if not word in stop_words]
print(without_stop_words)
```

Question : essayer plusieurs phrases en français et en anglais de votre choix, quel est le résultat ?

N-gram

1. Importer la librairie pour appliquer N-gram

```
import nltk
from nltk.util import ngrams
```

2. Définir une fonction pour exécuter le N-gram et imprimer les résultats

```
# Function to generate n-grams from sentences.
def extract_ngrams(data, num):
    n_grams = ngrams(nltk.word_tokenize(data), num)
    return [ ' '.join(grams) for grams in n_grams]
```

3. Définir un texte d'exemple. Vous pouvez utiliser le texte suivant, ou un autre de votre choix

   data = 'A class is a blueprint for the object. Backgammon is one of the oldest known board games.'

4. Visualizer 1-gram et 2-gram

   Question : Essayer aussi N-gram avec N= {3,4,5,6}

```
print("1-gram: ", extract_ngrams(data, 1))
print("\n2-gram: ", extract_ngrams(data, 2))
```

5. Faire une analyse fréquentiel des mots

```
import nltk
nltk.download('gutenberg')
from nltk.corpus import webtext
from nltk.probability import FreqDist

nltk.download('webtext')
#wt_words = webtext.words('testing.txt')

wt_words =  nltk.corpus.gutenberg.words('austen-emma.txt')


data_analysis = nltk.FreqDist(wt_words)

# Let's take the specific words only if their frequency is greater than 3.
filter_words = dict([(m, n) for m, n in data_analysis.items() if len(m) > 3])

for key in sorted(filter_words):
    print("%s: %s" % (key, filter_words[key]))

data_analysis = nltk.FreqDist(filter_words)
```

Questions :

Comparer le point dernière en faisant l'analyse sans et avec stopwords, quelles diffèrences trouvez-vous ?