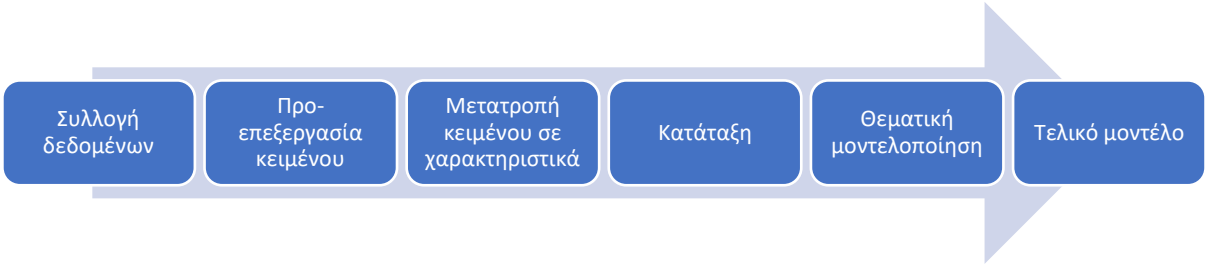
Άσκηση 1

Τεχνικές προ-επεξεργασίας κειμένου

Η διαδικασία που ακολουθείται συνήθως για την λεκτική ανάλυση ενός κειμένου ακολουθεί τα στάδια που φαίνονται στο παρακάτω σχήμα:



Σκοπός της συγκεκριμένης άσκησης είναι εξοικειωθείτε με τις βασικές τεχνικές που χρησιμοποιούνται στο στάδιο της προ-επεξεργασίας του κειμένου ώστε να ακολουθήσουν τα επόμενα στάδια.

Σημείωση: Για να εμφανίζονται οι λίστες σας οριζόντια στο IPyhton console, τρέξτε την εντολή

```
%pprint #turn off/on pretty printing
```

Τα NLTK modules που θα χρειαστούμε περιγράφονται στον παρακάτω πίνακα:

Language processing task	NLTK modules	Functionality
String processing	nltk.tokenize, nltk.stem nltk.PorterStemmer, nltk.LancasterStemmer	tokenizers, sentence tokenizers, stemmers

Συνοπτικά, οι εντολές που θα χρειαστείτε για την συγκεκριμένη άσκηση δίνονται στον παρακάτω πίνακα:

Εντολή	Επεξήγηση
`len(text3)`	Ο αριθμός των λεκτικών μονάδων στο κείμενο text3
`set(text3)`	Το λεξιλόγιο του text3
`len(set(text3))`	Ο αριθμός των λεκτικών μονάδων του λεξιλογίου του text3
`def function_name(parameter)`	Ορισμός μίας συνάρτησης
`freqDist(text3)`	Κατανομή συχνότητας στο text3
`list(bigrams(text3))`	Εμφανίζει τα ζευγάρια των λεκτικών μονάδων του λεξιλογίου του text3
`lower(text3)`	Μετατρέπει τους κεφαλαίους χαρακτήρες του text3 σε μικρά
`split(sentence)`	Χωρίζει το string sentence σε μικρότερα string

Βήμα 1 Απλά στατιστικά

Για τους σκοπούς μας, θα θεωρήσουμε ένα κείμενο ως τίποτα περισσότερο από μια σειρά από λέξεις και σημεία στίξης, δηλαδή λεκτικές μονάδες (tokens).

Α) Δημιουργείστε μία συνάρτηση που να υπολογίζει πόσο πλούσιο είναι το λεξιλόγιο σε ένα από τα βιβλία του nltk. Μπορείτε να πάρετε βοήθεια από το "Counting Vocabulary" παράγραφος 1.4 του βιβλίου [1]

Ερώτηση 1α: Υπολογίστε για τα βιβλία:

i)"Monty Python and the Holy Grail", πόσο πλούσιο είναι το λεξιλόγιο καθώς και πόσες φορές εμφανίζεται η λέξη "LAUNCELOT" και σε τι ποσοστό επί του λεξιλογίου του βιβλίου.

ii)"Chat Corpus". Πόσο πλούσιο είναι το λεξιλόγιο καθώς και πόσες φορές εμφανίζονται οι λέξεις "omg", "OMG" και "lol" και σε τι ποσοστό επί του λεξιλογίου του βιβλίου.

Ερώτηση 1β: Για καθένα από τα παραπάνω βιβλία επιλέξτε τρεις ακόμα λέξεις (αντιπροσωπευτικές ή μη) υπολογίζοντας το ποσοστό χρήσης της καθεμίας επί του λεξιλογίου του κάθε βιβλίου.

Τι συμπεράσματα βγάζετε από τα συνολικά πειράματα των ερωτήσεων 1α και 1β ;

Β) Αν τρέξουμε την εντολή `sent1` θα εμφανίσει την πρώτη πρόταση από το βιβλίο 1 (Moby Dick):

```
['Call', 'me', 'Ishmael', '.']
```

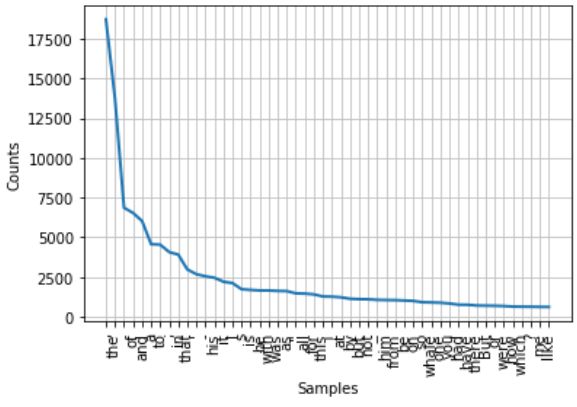
Παρατηρούμε ότι η Python αναπαριστά τις προτάσεις ως λίστες με tokens. Συνεπώς μπορούν να εφαρμοστούν διάφορες εντολές της Python που αφορούν σε λίστες. Μπορείτε να δείτε κάποια παραδείγματα στην ενότητα 2.1. του βιβλίου [1].

Γ) Για να υπολογίσουμε την κατανομή συχνότητας (frequency distribution) κάθε στοιχείου του λεξιλογίου σε ένα κείμενο `text`, χρησιμοποιούμε την εντολή του NLTK `FreqDist(text)` και `freq_dist.most_common(number)`. Τρέξτε τις παρακάτω εντολές.

```
fdist1 = FreqDist(text1) #Βάλε στην μεταβλητή fdist1 την κατανομή συχνότητας στο text1

fdist1 #Δείξε μου την μεταβλητή fdist1

fdist1.most_common(50) #Εμφάνισε τα 50 στοιχεία που εμφανίζονται με τη μεγαλύτερη συχνότητα

fdist1.plot(50)
```

Ερώτηση 2: Από το γράφημα που εμφανίζεται μπορούμε να βγάλουμε συμπέρασμα για το αντικείμενο που διαπραγματεύεται το βιβλίο; Γιατί;

```
In [8]: fdist1.plot(50),
```



Ερώτηση 3: Δείξτε παρόμοιο γράφημα για το βιβλίο “Monty Python and the Holy Grail” και αναφέρετε ποια λέξη (εκτός από προθέσεις κλπ) εμφανίζεται σε μεγαλύτερη συχνότητα από το “LAUNCELOT”. Επίσης, μπορείτε να συμπεράνετε κάποια θεματολογία για το συγκεκριμένο έργο και γιατί;

Βήμα 2 Κανονικοποίηση κειμένου (normalization)

Πριν προχωρήσουμε σε άλλες ενέργειες, πρέπει να κανονικοποιήσουμε το κείμενό μας έτσι ώστε να μην παίζουν ρόλο τα κεφαλαία.

```
sent1 #εμφάνισε την sent1 του βιβλίου 1
tokens1=sent1 #βάλε την sent1 στην μεταβλητή token1
normalized_sent1=[x.lower() for x in tokens1] #για κάθε x που υπάρχει
στο token1 κάνε "μικρά" τα γράμματα
normalized_sent1
```

Ερώτηση 4: Τι παρατηρείτε τρέχοντας τον παραπάνω κώδικα. Μπορείτε να σκεφτείτε κάποιες επιπτώσεις αυτής της κανονικοποίησης στα στατιστικά της προηγούμενης ερώτησης ή κάπου αλλού;

Μία ακόμα τεχνική κανονικοποίησης το λεγόμενο **stemming**, δηλαδή, η εύρεση του κυρίως στελέχους της λέξης (stem) έτσι ώστε να μην παίζουν ρόλο οι χρόνοι και ο αριθμός κλήσης (ενικός-πληθυντικός) των λέξεων. Αυτός επιτυγχάνεται αφαιρώντας με κάποιους κανόνες τις καταλήξεις π.χ. -ed, -ing, -es κλπ.

Βάλτε στη μεταβλητή `tokens1` τις **πρώτες 200 λεκτικές μονάδες** του βιβλίου “Sense and Sensibility” και τρέξτε τις ακόλουθες εντολές:

```
porter = nltk.PorterStemmer()
[porter.stem(t) for t in tokens1]
```

Μία άλλη τεχνική κανονικοποίησης είναι το **lemmatization** που χρησιμοποιεί στο συγκεκριμένο παράδειγμα παρακάτω, το Wordnet σαν βιβλιοθήκη, ώστε να αντιστοιχίσει (mapping) κάποιες λέξεις με κοινό νόημα. Για τις **πρώτες 200 λεκτικές μονάδες** του βιβλίου “ Sense and Sensibility ” (`tokens1` όπως παραπάνω) τρέξτε τις ακόλουθες εντολές:

```
nltk.download('wordnet')
wnl = nltk.WordNetLemmatizer()
```

```
[wnl.lemmatize(t) for t in tokens1]
```

Ερώτηση 5: Κάντε πειράματα με δικές σας προτάσεις-κείμενα. Δοκιμάστε και ελληνικό κείμενο. Εμφανίστε τα αποτελέσματά σας και γράψτε τα σχόλιά σας με κριτική άποψη κάνοντας σύγκριση των δύο παραπάνω τεχνικών καθώς και τη σύγκριση με την απλή κανονικοποίηση, παρουσιάζοντας τα θετικά και τα αρνητικά κάθε μίας.

Βήμα 3 Tokenization

Ο πιο απλός τρόπος για να χωρίσουμε το κείμενο σε λεκτικές μονάδες είναι να χρησιμοποιήσουμε την εντολή `split()`

```
sentence = "Monticello wasn't designated as UNESCO World Heritage Site
until 1987."
sentence.split()
```

ή εναλλακτικά

```
str.split(sentence)
```

```
['Monticello',
 "wasn't",
 'designated',
 'as',
 'UNESCO',
 'World',
 'Heritage',
 'Site',
 'until',
 '1987.']
```

Η εντολή `split()` κάνει ένα πρώτο βήμα tokenization αλλά παρατηρούμε ότι στο τέλος δεν χώρισε το 1987 με την τελεία, κάτι που είναι λάθος.

Υπάρχουν όμως και ειδικά πακέτα και functions για tokenization. Αφού κάνετε `import nltk.tokenize` χρησιμοποιώντας την εντολή `nltk.word_tokenize()` για να χωρίσετε σε λεκτικές μονάδες την παραπάνω πρόταση. Βοήθεια για την συγκεκριμένη εντολή μπορείτε να βρείτε στο βιβλίο [1]

Ερώτηση 6: Κάντε πειράματα με δικές σας προτάσεις-κείμενα καθώς και με τις πρώτες 200 λέξεις του βιβλίου «Sense and Sensibility». Δοκιμάστε και ελληνικό κείμενο. Εμφανίστε τα αποτελέσματά σας και γράψτε τα σχόλιά σας με κριτική άποψη κάνοντας σύγκριση των δύο παραπάνω τεχνικών, παρουσιάζοντας τα θετικά και τα αρνητικά κάθε μίας.

Βήμα 4 Αφαίρεση σημείων στίξης και προθημάτων (stop words)

Για να δείτε τα σημεία στίξης μπορείτε να τρέξετε τις παρακάτω εντολές

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

Για να καθαρίσετε ένα κείμενο από τα σημεία στίξης, μπορείτε να τρέξετε τις παρακάτω εντολές:

```
cleaned_tokens=[]
for token in tokens:
    if token not in string.punctuation:
        cleaned_tokens.append(token )
```

Για να δείτε τα προθήματα της Αγγλικής γλώσσας, μπορείτε να τρέξετε τις παρακάτω εντολές:

```
nltk.download('stopwords')
stopwords = nltk.corpus.stopwords.words('english')
print(stopwords)
```

Για να καθαρίσετε ένα κείμενο από τα προθήματα, μπορείτε να τρέξετε τις παρακάτω εντολές:

```
cleaned_tokens=[]
for token in tokens:
    if token not in stopwords:
        cleaned_tokens.append(token)
```

Ερώτηση 7: Δείξτε και αναφέρετε πόσα είναι τα stopwords για την Αγγλική γλώσσα και πόσα για την Ελληνική;

Δημιουργείστε μία συνάρτηση που όταν την καλείτε θα καθαρίζει ένα κείμενο από τα σημεία στίξης και τα προθήματα.

Ερώτηση 8: Καλέστε την συνάρτηση που φτιάξατε για να καθαρίσετε το κείμενο με τις πρώτες 200 λέξεις του βιβλίου «Sense and Sensibility». Κάντε πειράματα με δικές σας προτάσεις-κείμενα. Δοκιμάστε και ελληνικό κείμενο. Εμφανίστε τα αποτελέσματά σας και γράψτε τα σχόλιά σας με κριτική άποψη για τα συμπεράσματα που βγάζετε από τα πειράματά σας.

Ερώτηση 9: Χρησιμοποιήστε την εντολή από το βήμα 1 ώστε να εμφανίσετε την κατανομή συχνότητας των λέξεων τόσο στο «καθαρό» όσο και στο αρχικό κείμενο με τις πρώτες 200 λέξεις του βιβλίου «Sense and Sensibility». Δώστε τον κριτικό σχολιασμός σας για τα αποτελέσματα.

Αναφορές-Βιβλιογραφία

1] Steven Bird, Ewan Klein, and Edward Loper. 2009. Natural Language Processing With Python. O'REILLY. https://www.nltk.org/book/