30/8/24

Papers

(5) → Talking about ethics → meaningless

(6) ⇒ Ethics doesn't have a legal analogue or any universal acceptance.
'Human rights' → by UN → guarantee & enforced

→ Ethics are different from H.R.
→ H.R. → can be enforced universally.
↳ ∴ easier to practice across borders
→ but also a problem → huge critiques.

(7) Snake oil → 'chuna lagana'
Most of AI claims → snake oil.
+ ⇒ various snake oil → various claims.
eg:- Emotion recognition.
⇒ Snake oil → where fraudulent datasets used to train the model.
eg:- Sentiment analysis

NLTK chapter (3)
Recap:- HTML parsing using 'Beautiful soup'

→ (1) Writing a file
→ Path where file to be saved.
→ In terminal:-
f = open ("path\\ example.txt", 'w')
↳ means to write.

w → write
a → append
r → read

→ f. write ("A fox jumped ..... dog")
→ f. close()

⇒ file saved,

⇒ Now to add something again to it :-

→ f. open ("path \\ example.txt", "a")
appends the data instead of overwriting it.

→ f. write ("what a lovely day")
→ f. close().

⇒ Now to add something in new line.

→ f. open (.....)
→ f. write ("\n a quick fox").
next line sentence element

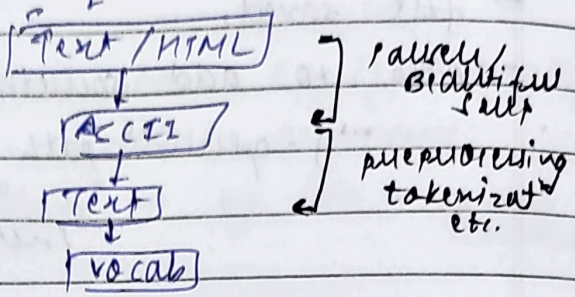
'csv' → comma separated values.
→ f. close()

⇒ Now to read a file.

→ f. open ("path", "r") → read
→ for i in f:
print(i).

⇒ Everything printed.

→ NLP pipeline:-



→ Regular Expressions for deleting word patterns :-

→ To find tense of words,
eg:- ing for present continuous.

→ • import nltk
• import re
• word_list = [w for w in nltk.corpus.cen('', for all english language words
if w.islower()]
↳ only lower-case
• word_list[:50] → print first 50 words
• stuff = [w for w in word_list.
if re.search('ing$', w)]
$ → tell for words ending with 'ing'
• stuff[:50]

→ Now for words starting with 'win'
• stuff [... if re.search('^win', 'w')]
^ → for start

just change this

→ Others operators in chapter 3.4

- **Stemmers:**
  - Process of converting every word in its smallest form, usually removing ing, es, s, etc. from last.
- **Lemmatizer**
  - Similar to stemmers
- **Segmentation:**
  - Breaking down sentence into words.
  - Usually difficult for some languages like chinese.

NLTK Chap. (4) :-

→ '=' → assignment, '==' → comparision.
→ Sequence → list, string, tuple
→ C++ → object oriented prog. (class, struct etc.)
→ python → 'OOP' or 'procedural and declarative'
→ variable scope → local or global.
→ coding discipline → usually not followed.
  ↳ but helps in better understanding and increases readability.

classmate
Date
Page

→ Algorithm design → to ↑ efficiency of a code
- code consumes less memories
- works faster.
- less computer → less harm to environment.

Example:-
- a = [2, 2, 4, 6, 7, 2, 3, 1, 4]

to convert into ascending order.
- answer = []
- for i in range(len(a)): → given size of list.
  small = a[i]
  for x in a:

⇒ Please use CHATGPT :)

→ Time complexity:- (T.C.) → time to run a code for large iterations.
O() → Big O notation → to represent T.C.

eg:-
```
for i in range(n):
    for j in range(n):
        i += j;
```
T.C. = O(n²)

NLTK chapter (5) :-

→ Part of Speech (POS) tagging

→ Given a sentence / text → to find POS tag for every word i.e. grammatical position eg: noun, verb, adjective etc.

→ Used to find position and context of each word in whole text.

→ Try out @ ~~cmd~~ corenlp.run

→ In nltk → ① word_tokenize (text)
② pos_tag (text)

→ Tagged corpora :-
↳ tuple consisting of {text : tag}

→ Taggers :-

→ Default tagging : Tagging everything same (eg: nouns before training.

→ Regular exp. tagger

ing → gerund
ed → past
es → present
→ modal
ould
⋮

→ Not ML yet ;C

→ Look up taggers

→ Fairly dataset find freq. of tags eg:- Noun : 60%
verb : 20%
etc.

→ Now, check for all words in your text and compare with dataset

Dataset	text
Bat → Noun	Bat → "Noun"

→ Larger dataset → more accuracy

→ But this method → not much useful

∴ We use 'Bigrams' :- Bigram Taggers
i.e. Noun is followed by a verb.

→ take two words → tagging them