at context

New Lec

ch-2

Gutenberg corpus
- refers to project Gutenburg
↳ largest collection of books whose copyright has ended
↳ good for text processing

- a subset of it is in nltk

Brown corpus
↳ each doc only 1 category

Reuters corpus
↳ each doc can have multiple categories
↳ so there will be overlaps

↳ CONLL dataset
- de facto std dataset for NLP

↳ Bhāsini project by Indian govmint
- to collect text data in Indian languages

↳ If most of data is English then it will only benefit english speakers

↳ 4 type of corpus

	e.g.
Isolated	gutenberg
Categorized	brown
overlapping	reuters
temporal	inaugural speeches

↳ Generating random text with Bigrams
- you want to predict next word
- find all biagrams of that word
- o/p most repeatating word in biagrams with that word

↳ Wordlist corpora ⟶ Unusual words

↳ Pronouncing dictionary ⟶ for text to sound

↳ WordNet
- dictionary
- provides sense
  ↳ Identical words can have diff senses based on the context
- Its not language but meaning based

ch-3 Processing Raw Text

↳ Accessing text from web and from Disk

import re ⟶ regular expression
↓
technique to find patterns of text

import pprint ⟶ ~~pp~~ pretty print
↓
library for

a = " I wouldn't go to class "

a.split() = ['I', 'wouldn't', 'go', 'to', 'class']

word_tokenize (a) = ['I', 'would', 'not', 'go', 'to', 'class']

from urllib ⟶ allows you to get url data from internet

↳ Text on internet is encoded e.g. on ASCII or UTF-8 } most common

so you'll have to decode them

Most pages of internet are actually html pages
→ merely decoding does not work
→ you need to parse it from html structure

Beautifulsoup uses to Parse html

New lec

Midterm Quiz (1hour or longer)
↳ diff from earlier quizzes

↳ were direct ques

↳ will have actual reasoning ques
- scenario based questions
- won't have coding ques
- but will have conceptual tech ques
(like what algo takes less time)
especially ML part
- more descriptive stuff

eg. que what was reason behind split of cybernetics & AI
- google won't help you much
- you must be literate in concepts
- handwritten & open everything (except open chatbot)

Couple of papers to read
↳ Assigned readings are components of this course
- means will come in exams

(4) The global landscape of AI ethics guidelines
- paper which covers all ethical guidelines that exist

(5) Ethics as an escape from regulation by Ben Wagner
- Ethics is excellent but normative not legal
- more we talk abt ethics, less we talk about regulations
- hence companies only talk abt ethics not regulations

⑥ Governence with Teeth by Marda
↳ means Human rights

- Ethics is something which does not have legal analog or universal acceptance
- Human rights are not normative but globally defined
- There is no legal tool to enforce ethics but human rights have to be enforced on every country.
- Across borders regulations based on human rights can be imposed

⑦ How to recognize AI snake oil by Arvind Narayan
- This is not paper but slides by a MIT prof
- snake oil means fraud
- Most of AI claims are snake oil
- snake oil is of diff types
- this slide talks about how to recognize such AI snake oils, some are very sophisticated like morphed datasets, exaggerated claims etc
- This ~~book~~ prof is writing a book, too.

Let's continue with NLTK ch-3

↳ Prev: how to download url from internet
- problem when HTML page
- hence pass the page through HTML parser (beautifulsoup)

↳ Reading Local files

f = open ('<file-path>', 'w')
↳ means to write

f.write ("<whatever you want to write>")

f.close

- whenever 'w' used, it overwrites
means it will delete prev data
- to append data, use 'a' instead of 'w'
- 'r' used to read data
If you open in 'r' mode & run a for loop

for i in f:
} then i will go sentence by sentence

↳ NLP pipeline
- basic pipeline for all kind of research you'll be doing

HTML ⟶ ASCII ⟶ Text ⟶ Vocab

- nltk_wordpunc Tokenise
↳ this takes care of punctuations

↳ Next part of chapter tells about strings

↳ Then directly jump to "Regular Expressions" topic
- tool to find patterns from text
eg. you want to find all words ending with 'ing'
- every prog lang like python have their own regular expression tool.

- nltk.corpus.words.words

↳ gives every single word in nltk

- nltk.corpus.words.words('en')

gives every single english word in nltk

~~nltk.corpus.words.words~~ ('ing$', w)
- re.search

$ means word ending

will give all words ending with ing

- re.search ('^win', w)

^ means word starting

will give all words starting with win

- re.search ('aa+', w)

will give all words containing 'aa' or multiple aa's

↳ Then chap goes into deeper in regular expressions

↳ Normalizing text
- changing everything in lower case

↳ stemmers → smallest possible word in given word

e.g. Tables $\xrightarrow[\text{word}]{\text{stem}}$ Table

listen $\xrightarrow[\text{word}]{\text{stem}}$ list

↳ Lemmatization
- process of stemming is called lemmatization

↳ Segmentation
- breaking text in sentences

↳ Tokenization or word segmentation
- breaking text in words

↳ list to string : join()
↳ string to list : split()

chapter 4

↳ Assignment of operators : '='
comparison of operators : '=='

↳ list, Tuples, sequence types

↳ No need to know generator expression

↳ Python coding style → coding style → every language has one
e.g. oop in C++

- Procedural vs declarative style
= variable scope
- checking parameter type
  - Python does not allow to declare type of variable
  - it only allows us to check type of variable

↳ Not much imp in context of this course or NLTK
- this chap is more abt learning coding disciplin in python.

- To learn more python, go through python book on moodle

↳ Algorithm design
- how to code program s.t. it consumes less resources or less time
- basically code in an efficient way
e.g. divide and conquer in sorting

↳ Python libraries

chapter 5

↳ starts to talk abt ML in NLP
↳ POS tagging
- classified learning task
- given text, find POS for every word
- POS tag is grammatical role of that word e.g. noun, adverb, adjective, ...
- In POS tag, position & tag of one word affects others
- so words are not indep

↳ Dictionaries in Python
↳ How to make POS tagger?
- one of the easiest ML task
- default tagging: tag Noun to every word
(bcoz majority words are Noun in English)

- then use regular expression tagging: e.g. everything which end with "ould" is modals

- then use lookup tagger: lookup table of majority tags

- find a dataset with POS tags, find out which tag is associated most (majority) of the times with a word

- till now you used only single words
- now increase size of data
- Increasing words of dataset will give upto >90 % accura
- but 90 is still bad
- This 90 to 98-99 , you'll need context
- So you now move to bigrams
- this word followed by that word is noun (or verb or
- lookup table = unigram tagger
- bigram tagger → you'll get 95-96 % accuracy
- This is something, you'll must be able to code