txt is comprised of lines all the text files ,
line == '--------': and if line != '': is used to remove lines , lines comprised of spaces i.e redundant data(here data set unit is lines i text file) .
def read_all_textfile(path): function reads all the lines from all the text files into the array txt .
# this can also be visualized as segmentation part of nlp
parts of nlp - 1) segmentation(DIVIDING ENTIRE TEXT ON THE BASIS OF , FUUL STOP , COMMA , NEW LINE .
2)tokenization - DIVIDING THE ABOVE UNITS INTO WORDS.
3)stemming - HEURISTIC BASED APPROACH TO FIND STEM/BASIC FORM OF WORD EX MOVING , MOVED ,MOVABLE = MOVE .
4) lemmatization - FINDS MOST BASIC FORM OF WORD BY DICTIONARY LOOKUPS, MORPHOLOGICAL ANALYSIS.SLOWER THAN STEMMING
5) part of speech tagging
6)- named entity recognition

//**read_all_textfile (pah)**

tokenization - text data uses vast variety of alphabet space which may have no literal meaning(. , */) , cleaning part .
converting line into constituent words
each word is used as a state in markov model.
cleaned_arr has word using alphabets from space E[a,z] .

// def clean_txt(txt):

n_gram = 2 , we are defining set of 2 consecutive words as our state (TO ADD MORE CONTEXT TO THE MODEL)
markov model is graph data structure where key is the state and each key contains its transition state and its associated prob.

// def make_markov_model(cleaned_arr, n_gram=2

generated text depends upon starting state

//def generate_text(markov_model, limit=100, start='my god'):

limit = 8 , 8 states are produces , total = 16 words , start state = "dear boy"
total no of sentence = 20
each sentence is correct grammatically as the markov model is follows the frequency and statistics of an grammatically correct original data

//generate_text(markov_model, start="dear boy", limit=8)