import re

import string

### input a text dataframe, output a list of vocab. Ex: listVocab(data["text"])

def listVocab(frame):

vocab=Counter()

for i in range(0,len(frame)):

try:

vocab.update(frame.iloc[[i]].item().split(" ").remove(""))

except:

vocab.update(frame.iloc[[i]].item().split(" "))

return vocab

### all functions under take input as strings. Ex: removePunctuations("Oh god fuck me!")

### Removing punctuations. (!"#$%&'()\*+, -./:;<=>?@[\]^\_`{|}~)

### Exceptions takes input as string Ex: removePunctuations(text,".")

def removePunctuations(text,exception=""):

remove=string.punctuation+"“"+"”"+"“"+"–"

if len(exception)!="":

exception = list(exception)

for i in exception:

remove = remove.replace(i, "")

translator = str.maketrans(remove , ' '\* len(remove))

text = text.translate(translator)

return text

def convertLowercase(text: str):

return text.lower()

def normalizeWhitespace(text):

result = str(text)

result = re.sub(r"//t",r"\t", result)

result = re.sub(r"( )\1+",r"\1", result)

result = re.sub(r"(\n)\1+",r"\1", result)

result = re.sub(r"(\r)\1+",r"\1", result)

result = re.sub(r"(\t)\1+",r"\1", result)

return result.strip(" ")

def fixVietnameseSimples(text):

mapper = {'a ̀ ': 'à','a ́ ': 'á','a ̃ ': 'ã','a ̣ ': 'ạ','a ̉ ': 'ả','ă ̀ ': 'ằ','ă ́ ': 'ắ','ă ̃ ': 'ẵ','ă ̣ ': 'ặ','ă ̉ ': 'ẳ','â ̀ ': 'ầ','â ́ ': 'ấ','â ̃ ': 'ẫ','â ̣ ': 'ậ','â ̉ ': 'ẩ','e ̀ ': 'è','e ́ ': 'é','e ̃ ': 'ẽ','e ̣ ': 'ẹ','e ̉ ': 'ẻ','ê ̀ ': 'ề','ê ́ ': 'ế','ê ̃ ': 'ễ','ê ̣ ': 'ệ','ê ̉ ': 'ể', 'i ̀ ': 'ì','i ́ ': 'í','i ̃ ': 'ĩ','i ̣ ': 'ị','i ̉ ': 'ỉ','o ̀ ': 'ò','o ́ ': 'ó','o ̃ ': 'õ','o ̣ ': 'ọ','o ̉ ': 'ỏ','ô ̀ ': 'ồ','ô ́ ': 'ố','ô ̃ ': 'ỗ','ô ̣ ': 'ộ','ô ̉ ': 'ổ','ơ ̀ ': 'ờ','ơ ́ ': 'ớ','ơ ̃ ': 'ỡ','ơ ̣ ': 'ợ','ơ ̉ ': 'ở','u ̀ ': 'ù','u ́ ': 'ú','u ̃ ': 'ũ','u ̣ ': 'ụ','u ̉ ': 'ủ','ư ̀ ': 'ừ','ư ́ ': 'ứ','ư ̃ ': 'ữ','ư ̣ ': 'ự','ư ̉ ': 'ử','y ̀ ': 'ỳ','y ́ ': 'ý','y ̃ ': 'ỹ','y ̣ ': 'ỵ','y ̉ ': 'ỷ','A ̀ ': 'À','A ́ ': 'Á','A ̃ ': 'Ã','A ̣ ': 'Ạ','A ̉ ': 'Ả','Ă ̀ ': 'Ằ','Ă ́ ': 'Ắ','Ă ̃ ': 'Ẵ','Ă ̣ ': 'Ặ','Ă ̉ ': 'Ẳ','Â ̀ ': 'Ầ','Â ́ ': 'Ấ','Â ̃ ': 'Ẫ','Â ̣ ': 'Ậ','Â ̉ ': 'Ẩ','E ̀ ': 'È','E ́ ': 'É','E ̃ ': 'Ẽ','E ̣ ': 'Ẹ','E ̉ ': 'Ẻ','I ̀ ': 'Ì','I ́ ': 'Í','I ̃ ': 'Ĩ','I ̣ ': 'Ị','I ̉ ': 'Ỉ','O ̀ ': 'Ò','O ́ ': 'Ó','O ̃ ': 'Õ','O ̣ ': 'Ọ','O ̉ ': 'Ỏ','Ô ̀ ': 'Ồ','Ô ́ ': 'Ố','Ô ̃ ': 'Ỗ','Ô ̣ ': 'Ộ','Ô ̉ ': 'Ổ','Ơ ̀ ': 'Ờ','Ơ ́ ': 'Ớ','Ơ ̃ ': 'Ỡ','Ơ ̣ ': 'Ợ','Ơ ̉ ': 'Ở','U ̀ ': 'Ù','U ́ ': 'Ú','U ̃ ': 'Ũ','U ̣ ': 'Ụ','U ̉ ': 'Ủ','Ư ̀ ': 'Ừ','Ư ́ ': 'Ứ','Ư ̃ ': 'Ữ','Ư ̣ ': 'Ự','Ư ̉ ': 'Ử','Y ̀ ': 'Ỳ','Y ́ ': 'Ý','Y ̃ ': 'Ỹ','Y ̣ ': 'Ỵ','Y ̉ ': 'Ỷ'}

text=text.translate(mapper)

return text

### Removing HTML entities such as tags and html elements.

### NOTE: only use this if the text actually contains html entity.

### All datasets here uses Beautiful Soup get\_text() to output text,

### which eliminates html tags automatically.

def removeHTMLElements(text):

### Creating regular expression for html tags and elements.

regex\_pattern = re.compile('<[^<]+?>|&([a-z0-9]+|#[0-9]{1,6}|#x[0-9a-f]{1,6});')

return regex\_pattern.sub(r' ',text)

def removeURL(sample):

###Remove URLs from a sample string

return re.sub(r"http\S+", "", sample)

### Removing emoticons from text.

def removeEmoticons(text):

### Creating regularize expression for the emoticons

emoji\_pattern = re.compile("["

u"\U0001F600-\U0001F64F" ### emoticons

u"\U0001F300-\U0001F5FF" ### symbols & pictographs

u"\U0001F680-\U0001F6FF" ### transport & map symbols

u"\U0001F1E0-\U0001F1FF" ### flags (iOS)

u"\U00002500-\U00002BEF" ### chinese char

u"\U00002702-\U000027B0"

u"\U00002702-\U000027B0"

u"\U000024C2-\U0001F251"

u"\U0001f926-\U0001f937"

u"\U00010000-\U0010ffff"

u"\u2640-\u2642"

u"\u2600-\u2B55"

u"\u200d"

u"\u23cf"

u"\u23e9"

u"\u231a"

u"\ufe0f" ### dingbats

u"\u3030"

"]+", flags=re.UNICODE)

return emoji\_pattern.sub(r'', text)

### Apply all text preprocessing.

def preprocessingText(text,html=False):

text = r'{}'.format(text)

text = convertLowercase(text)

if html:

### Remonving HTML tags.

text = removeHTMLElements(text)

### Removing emoticons

text = removeEmoticons(text)

### Removing punctuations.

text = removePunctuations(text)

return text

if \_\_name\_\_ == "\_\_main\_\_":

### TODO: Create Preprocessing method, which includes:

"""### emoticons (Done).

print(removeEmoticons('😀 😃 😄 😁 😆 😅 😂 🤣 🥲 ☺hello️ 😊 world 😇 🙂 🙃 😉 😌 😍 🥰 😘 😗 😙 😚 😋 😛 😝'))"""

"""### punctuations (Done).

print(removePunctuations("!@#$$%%% hello @#$#$()./'''<><.,.,. world"))"""

"""### HTML tags.

print(removeHTMLElements('<p>Hello World</p>'))"""

"""### Testing all three methods.

print(preprocessingText("😀 😃 😄 😁 😆 😅 😂 🤣 🥲 ☺hello️ 😊 world 😇 🙂 🙃 😉 😌 😍 🥰 😘 😗 😙 😚 😋 😛 😝!@#$$%%% hello @#$#$()./''<><.,.,. world<p>Hello World</p>",html=True))"""