import nltk

from nltk.corpus import twitter\_samples

from nltk.tokenize import TweetTokenizer

from nltk.corpus import stopwords

from nltk.stem import PorterStemmer

from nltk import classify

from nltk import NaiveBayesClassifier

nltk.download('twitter\_samples')

nltk.download('stopwords')

positive\_tweets = twitter\_samples.strings('positive\_tweets.json')

negative\_tweets = twitter\_samples.strings('negative\_tweets.json')

tokenizer = TweetTokenizer()

positive\_tweet\_tokens = [tokenizer.tokenize(tweet) for tweet in positive\_tweets]

negative\_tweet\_tokens = [tokenizer.tokenize(tweet) for tweet in negative\_tweets]

stop\_words = set(stopwords.words('english'))

stemmer = PorterStemmer()

def clean\_tokens(tokens):

cleaned\_tokens = []

for token in tokens:

if token.lower() not in stop\_words and token.isalpha():

cleaned\_tokens.append(stemmer.stem(token.lower()))

return cleaned\_tokens

def get\_tweets\_for\_model(cleaned\_tokens\_list):

for tweet\_tokens in cleaned\_tokens\_list:

yield dict([token, True] for token in tweet\_tokens)

positive\_tokens\_for\_model = get\_tweets\_for\_model(positive\_tweet\_tokens)

negative\_tokens\_for\_model = get\_tweets\_for\_model(negative\_tweet\_tokens)

positive\_dataset = [(tweet\_dict, "Positive") for tweet\_dict in positive\_tokens\_for\_model]

negative\_dataset = [(tweet\_dict, "Negative") for tweet\_dict in negative\_tokens\_for\_model]

dataset = positive\_dataset + negative\_dataset

import random

random.shuffle(dataset)

train\_data = dataset[:7000]

test\_data = dataset[7000:]

classifier = NaiveBayesClassifier.train(train\_data)

print("Accuracy:", classify.accuracy(classifier, test\_data))

print(classifier.show\_most\_informative\_features(10))

# Example usage

custom\_tweet = "I love this product!"

custom\_tokens = clean\_tokens(tokenizer.tokenize(custom\_tweet))

print("Sentiment for '{}':".format(custom\_tweet), classifier.classify(dict([token, True] for token in custom\_tokens)))