import pandas as pd

from sklearn.feature\_extraction.text import CountVectorizer

from sklearn.feature\_extraction.text import TfidfTransformer

from sklearn.pipeline import Pipeline

#import re

import numpy as np

import random

cd = pd.read\_csv('chat\_train.csv')

intent = pd.read\_excel('intent\_keys.xlsx',index\_col=0)

from nltk.stem.snowball import SnowballStemmer

stemmer = SnowballStemmer("english", ignore\_stopwords=True)

class StemmedCountVectorizer(CountVectorizer):

def build\_analyzer(self):

analyzer = super(StemmedCountVectorizer, self).build\_analyzer()

return lambda doc: ([stemmer.stem(w) for w in analyzer(doc)])

stemmed\_count\_vect = StemmedCountVectorizer(ngram\_range=(1,3))

pl = Pipeline([

('vect', stemmed\_count\_vect),

('tfidf', TfidfTransformer())])

#Train data

X = cd.qn #question column from dataset

y = cd.intent #intents column from dataset

#One hot Encoding

# Encoding categorical data

from sklearn.preprocessing import LabelEncoder

labelencoder\_y = LabelEncoder()

y = labelencoder\_y.fit\_transform(y)

# one hot encoding

from keras.utils import np\_utils

y = np\_utils.to\_categorical(y)

#model

m = pl.fit(X, y) #training the model

n = pl.transform(X)

l = list(cd.intent.unique())

in\_dim = n.shape[1]

output\_dim = len(l)

ll = sorted(l)

# Model Nnet

import keras

from keras.models import Sequential

from keras.layers import Dense, Dropout

classifier = Sequential()

classifier.add(Dense(units = 1024, kernel\_initializer = 'uniform', activation = 'relu', input\_dim = in\_dim))

classifier.add(Dropout(0.2))

classifier.add(Dense(units = 512, kernel\_initializer = 'uniform', activation = 'relu'))

classifier.add(Dropout(0.2))

classifier.add(Dense(units = 256, kernel\_initializer = 'uniform', activation = 'relu'))

classifier.add(Dense(units = output\_dim, kernel\_initializer = 'uniform', activation = 'softmax'))

classifier.compile(optimizer = 'adam', loss = 'binary\_crossentropy', metrics = ['accuracy'])

classifier.fit(n, y, batch\_size = 5, epochs = 50)

classifier.save('chatbotNnet\_model.h5')

from keras.models import load\_model

classifier = load\_model('chatbotNnet\_model.h5')

#s = pd.Series("WHAT IS YOUR NAME")

#t = pl.transform(s)

#res = classifier.predict(t)

#newres = np.transpose(res)

#newres = newres.tolist()

#reply = ll[np.argmax(newres)]

#print("intent: " + reply)

while True:

H = input('user: ').strip() # taking the raw input from the user

Hlower = H.lower()

Hlower = pd.Series(Hlower)

Hlower = pl.transform(Hlower)

res = classifier.predict(Hlower)

newres = np.transpose(res).tolist()

print((max(newres))[0])

if ((max(newres))[0] > 0.40):

a = ll[np.argmax(newres)]

ans=intent.loc[a].values.tolist()

ans=np.asarray(ans)

ans=ans[ans != 'nan']

print('\nbot: ' + random.choice(ans))

else:

print('\nbot: I am afraid, I dont have answer for this. could you please try another question.')