#CPSC475 Dr.DePalma Fall 2016 asgn6\_test

#Probabilistic Part of Speech Tagger

#Sebastian Vargas

#id: avargas

#

#To run on linux, go to your terminal then go to the directory in which

# this program is stored. Then type "python asgn6\_test.py insert string here"

# in the command line

#This should execute the program

import sys

#import viterbi

import pandas

from pandas import DataFrame

import nltk

import numpy as np

import simplejson

'''

N: number of hidden states

'''

class Decoder(object):

def \_\_init\_\_(self, initialProb, transProb, obsProb):

self.N = initialProb.shape[0]

self.initialProb = initialProb

self.transProb = transProb

self.obsProb = obsProb

print("initialProb: ")

print(initialProb.shape)

print("transProb: ")

print(transProb.shape)

print("obsProb: ")

print(obsProb.shape)

assert self.initialProb.shape == (self.N, 1)

assert self.transProb.shape == (self.N, self.N)

assert self.obsProb.shape[0] == self.N

def Obs(self, obs):

return self.obsProb[:, obs, None]

def Decode(self, obs):

trellis = np.zeros((self.N, len(obs)))

backpt = np.ones((self.N, len(obs)), 'int32') \* -1

# initialization

trellis[:, 0] = np.squeeze(self.initialProb \* self.Obs(obs[0]))

for t in xrange(1, len(obs)):

trellis[:, t] = (trellis[:, t-1, None].dot(self.Obs(obs[t]).T) \* self.transProb).max(0)

backpt[:, t] = (np.tile(trellis[:, t-1, None], [1, self.N]) \* self.transProb).argmax(0)

# termination

tokens = [trellis[:, -1].argmax()]

for i in xrange(len(obs)-1, 0, -1):

tokens.append(backpt[tokens[-1], i])

return tokens[::-1]

argList = str(sys.argv)

sentences = argList[1:]

infileA="matrixA.xlsx"

infileB="matrixB.xlsx"

brown\_train\_file = "brown\_train.txt"

f = open('brown\_train.txt', 'r')

brown\_train = simplejson.load(f)

f.close()

Apandas = pandas.read\_excel(infileA,sheetname="Matrix A")

#print Apandas

rownames = Apandas.index.tolist()

A=np.array(Apandas)

#print A

Bpandas = pandas.read\_excel(infileB,sheetname='Matrix B')

#print Bpandas

B=np.array(Bpandas)

statenames = Bpandas.index.tolist()

#print B

trans=A[0:,:]

print trans

pi=np.expand\_dims(np.array(A[1,:]),1)

print pi

decoder = Decoder(pi,trans, B)

#do the decoding

states = decoder.Decode(np.arange(5))

result = np.array(statenames)[states].tolist()

#sentence = Bpandas.columns.tolist()

#sentence = ["the","and","find","the",";"]

resultTagged = zip(sentences,result)

resultnew = [(tup[0].encode('ascii'),tup[1].encode('ascii')) for tup in resultTagged]

#correct=' Janet/NNP will/MD back/VB the/DT bill/NN'

#correct=[nltk.str2tuple(x) for x in correct.split()]

print (resultnew)

#assert (resultTagged==correct)

#print "PASSED"