22 Sept 2018

Problem 3/4

① Downloaded data files
browncopy_2018.zip
gutenberg.zip
imdb_data.zip
news_data.zip
science_sample.zip

Problem 3

(A) Pre-prep files
- path to a file *fnx*:
import os
os.path.join(dirPath, fnx)
- read entire file into a string:
~~import os~~
with open(fnxPath) as f:
fnx_data = f.read()
- list files in a dir:
import os
files = os.listdir(dirPath)
NOTE: lists files and subdirectories
os.path.isfile(fnxPath)
returns True if path is a file, not dir

- compile list of infrequent tokens:

```
TOO_FEW = 5
fnx_infreq = [ (token, counts[token],) \
    for token in counts \
        if counts[token] <= TOO_FEW ]
```

- compile list of 4-grams in a doc (tokens)

```
four_grams_doc = defaultdict(int)
if len(tokens) >= 4:
    for i in range(len(tokens)-3):
        gram = ( tokens[i],
                 tokens[i+1],
                 tokens[i+2],
                 tokens[i+3] )
        four_grams_doc[gram] += 1
```

- print table

```
    fnx: file name
    countInDoc: 4-grams in doc
    countNew: 4-grams in corpus added from doc
print("%-30s %10d %10d", \
    % (fnx, countInDoc, countNew) )
```

23 Sept 2018

New git repository NEU_CS6120 (master)
in D:\Documents\NLP\NEU_CS6120

includes:

course outline

quiz answers doc

assignment 1 data files

neu_cs6120_a1.py — 3.1 solution (1st version)

- count 4-grams

```
count_in_doc = len(four_grams_doc)
count_new_in_all .= 0
for gram in four_grams_doc:
    if gram not in four_grams_all:
        count_new_in_all += 1
```

Note: the counting can be done incrementally as the 4-grams in a doc are collected, but for each 4-gram must check if it is new in the doc before counting it.

- Change '\n' to ' '

```
import re
fnx_data_unl = re.sub(r'\n', ' ', fnx_data)
```

- Collapse multiple consecutive ' ' to single ' '

```
import re
fnx_data_sb = re.sub(r'( )+', ' ', fnx_data_unl)
```

- Tokenize

```
import nltk
fnx_tokens = nltk.word_tokenize(fnx_data_sb)
```

- Count occurrences of tokens

```
counts = defaultdict(int)
for token in tokens:
    counts[token] += 1
```

- Replace infrequent tokens with 'UNK'

```
TOO_FEW = 5
tokens_prepped = tokens.copy()
for i in range(len(tokens)):
    token = tokens[i]
    if counts[token] <= TOO_FEW:
        tokens_prepped[i] = 'UNK'
    else:
        tokens_prepped[i] = token
```