Text Analytics HW2

Sophie Du

In [1]:
```
import gensim
from gensim.test.utils import common_texts, get_tmpfile
from gensim.models import Word2Vec
import nltk
#nltk.download()
from nltk.tokenize import sent_tokenize
from nltk.tokenize import word_tokenize
import re
import string

path = get_tmpfile("word2vec.model")
```

Unit Tests

In [134]:
```
def run_unit_tests():
    """if all tests pass, then return cleaned and tokenized corpus"""
    sample = 'Newsgroup: sci.med\ndocument_id: 57110\nFrom: bed@intacc.u
ucp (Deb Waddington)\nSubject: INFO NEEDED: Gaucher\'s Disease\n'

    ## test \n removal
    def test_remove_n():
        remove_test = sample.replace('\n', '')
        remove_true = "Newsgroup: sci.meddocument_id: 57110From: bed@int
acc.uucp (Deb Waddington)Subject: INFO NEEDED: Gaucher's Disease"
        assert remove_test == remove_true
        return remove_test

    ## test lower case
    ## pass result from last step
    sample = test_remove_n()
    def test_lower():
        lower_test = sample.lower()
        lower_true = "newsgroup: sci.meddocument_id: 57110from: bed@inta
cc.uucp (deb waddington)subject: info needed: gaucher's disease"
        assert lower_test == lower_true
        return lower_test

    ## pass result from last step
    sample = test_lower()
    ## test number removal
    def test_remove_num():
        remove_num_test = re.sub(r'\d+', '', sample)
        remove_num_true = "newsgroup: sci.meddocument_id: from: bed@inta
cc.uucp (deb waddington)subject: info needed: gaucher's disease"
        assert remove_num_test == remove_num_true
        return remove_num_test

    ## pass result from last step
    sample = test_remove_num()
    ## test sentence tokenization output type
    def test_sent_token_type():
        sent_type_test = type(sent_tokenize(sample))
        assert sent_type_test == 'list'

    ## test sentence tokenization output content
    def test_sent_token_cont():
        sent_cont_test = sent_tokenize(sample)
        sent_cont_true = ["newsgroup: sci.meddocument_id: from: bed@inta
cc.uucp (deb waddington)subject: info needed: gaucher's disease"]
        assert sent_cont_test == sent_cont_true
        return sent_cont_test

    ## pass result from last step
    sample = test_sent_token_cont()
    # test symbol and punctuation removal
    def test_remove_symbol():
        for s in sample:
            remove_symbol_test = re.sub(r'[!”#$%&’()*+,-./:;<=>?@[\]^_`
{|}~]', ' ', s)
            remove_symbol_true = "newsgroup  sci meddocument id  from  b
```

```
ed intacc uucp  deb waddington subject  info needed  gaucher's disease"
            assert remove_symbol_test == remove_symbol_true
            return remove_symbol_test

    ## pass result from last step
    sample = test_remove_symbol()
    ## test word tokenization type
    def test_word_token_type():
        word_type_test = type(word_tokenize(sample))
        assert word_type_test == 'list'

    ## test word tokenization content
    def test_word_token_cont():
        word_cont_test = word_tokenize(sample)
        word_cont_true = ['newsgroup', 'sci', 'meddocument', 'id', 'fro
m',
                          'bed', 'intacc', 'uucp', 'deb', 'waddington',
                          'subject', 'info', 'needed', 'gaucher', "'s",
'disease']
        assert word_cont_test == word_cont_true
        return word_cont_test

    print('All unit tests passed; cleaned and tokenized corpus from samp
le text is shown below')

    return test_word_token_cont()

run_unit_tests()
```

```
All unit tests passed; cleaned and tokenized corpus from sample text is
shown below
```

Out[134]:
```
['newsgroup',
 'sci',
 'meddocument',
 'id',
 'from',
 'bed',
 'intacc',
 'uucp',
 'deb',
 'waddington',
 'subject',
 'info',
 'needed',
 'gaucher',
 "'s",
 'disease']
```

Preprocessing

```
In [7]: ## open text and normalization
with open('Lab1/20-newsgroups/sci.med.txt', encoding = 'utf8', errors =
'ignore') as f:
    ## remove '\n'
    text = str(f.read().replace('\n', ''))
    ## lower cases text
    text = text.lower()
    ## remove numbers
    text = re.sub(r'\d+', '', text)
```

```
In [3]: ## tokenize sentences
sent_nltk = sent_tokenize(text); #sent_nltk
```

```
In [4]: ## preprocess text and generate corpus for model
clean_text = []
corpus = []
for sent in sent_nltk:
    ## remove symbols
    corp = re.sub(r'[!"#$%&'()*+,-./:;<=>?@[\]^_`{|}~]', ' ', sent)
    ## output clean text
    clean_text.append(corp)
    ## tokenize words
    corpus.append(word_tokenize(corp))
```

- Here's the preprocessing result after tokenization and normalizations including converting to lower case, removing alphanumeric characters, numbers, symbols and white spaces, and replacing punctuations with blank.

```
In [5]: clean_text[:2]

Out[5]: ["newsgroup  sci meddocument id  from  bed intacc uucp  deb waddington
subject  info needed  gaucher's diseasei have a  yr old male friend  mi
sdiagnosed as having osteopporosis for two years  who recently found ou
t that his illness is the rare gaucher's disease ",
 "gaucher's disease symptoms include  brittle bones  he lost   inches o
ff his hieght   enlarged liver and spleen  internal bleeding  and fatig
ue  all the time  "]
```

In [27]:
```
corpus[1]
```

Out[27]:
```
['gaucher',
 "'s",
 'disease',
 'symptoms',
 'include',
 'brittle',
 'bones',
 'he',
 'lost',
 'inches',
 'off',
 'his',
 'hieght',
 'enlarged',
 'liver',
 'and',
 'spleen',
 'internal',
 'bleeding',
 'and',
 'fatigue',
 'all',
 'the',
 'time']
```

Compare Embedding Sizes

Keep window and model type the same

Experiment 1: Model using CBOW, with embedding size=100

In [7]:

```python
model1 = Word2Vec(corpus, min_count = 10, size = 100,
                  workers = 4, window = 5, sg = 0)
## cosine similarity
word_list = ['disease', 'treatment', 'drug', 'effective', 'patient']
for i in word_list:
    print('Word: %s' %i)
    print('5 closest neighbors are:')
    print(model1.wv.most_similar(i)[:5])
    print('-'*50)
```

```
Word: disease
5 closest neighbors are:
[('treatment', 0.8861936330795288), ('drug', 0.8447970151901245), ('dis
eases', 0.840467095375061), ('fungal', 0.8193104267120361), ('infectiou
s', 0.8182231783866882)]
--------------------------------------------------
Word: treatment
5 closest neighbors are:
[('drug', 0.922074556350708), ('disease', 0.8861936330795288), ('side',
0.8760125041007996), ('risk', 0.8690671920776367), ('effects', 0.865475
5353927612)]
--------------------------------------------------
Word: drug
5 closest neighbors are:
[('treatment', 0.9220744371414185), ('oral', 0.8860127329826355), ('cos
t', 0.8851606249809265), ('common', 0.8817379474639893), ('brain', 0.87
61096596717834)]
--------------------------------------------------
Word: effective
5 closest neighbors are:
[('common', 0.9430716037750244), ('causing', 0.9008839130401611), ('imp
ortant', 0.9001145958900452), ('commonly', 0.887724757194519), ('antibi
otic', 0.8787003755569458)]
--------------------------------------------------
Word: patient
5 closest neighbors are:
[('condition', 0.8912587761878967), ('given', 0.859172523021698), ('tak
en', 0.8478406667709351), ('difficult', 0.8326812982559204), ('gettin
g', 0.8300775289535522)]
--------------------------------------------------
```

Experiment 2: Model using CBOW, with embedding size = 200

In [8]:
```
model2 = Word2Vec(corpus, min_count = 10, size = 200,
                  workers = 4, window = 5, sg = 0)
## cosine similarity
word_list = ['disease', 'treatment', 'drug', 'effective', 'patient']
for i in word_list:
    print('Word: %s' %i)
    print('5 closest neighbors are:')
    print(model2.wv.most_similar(i)[:5])
    print('-'*50)
```

```
Word: disease
5 closest neighbors are:
[('treatment', 0.9069466590881348), ('drug', 0.8495345115661621), ('com
mon', 0.8433587551116943), ('infection', 0.8334235548973083), ('disease
s', 0.8315562605857849)]
--------------------------------------------------
Word: treatment
5 closest neighbors are:
[('disease', 0.9069467782974243), ('drug', 0.9051125049591064), ('thera
py', 0.8981338739395142), ('common', 0.876920759677887), ('infection',
0.8664539456367493)]
--------------------------------------------------
Word: drug
5 closest neighbors are:
[('common', 0.9112310409545898), ('treatment', 0.9051125049591064), ('b
rain', 0.8861269950866699), ('side', 0.8758144378662109), ('itraconazol
e', 0.8756904602050781)]
--------------------------------------------------
Word: effective
5 closest neighbors are:
[('common', 0.9268741607666016), ('non', 0.9248567223548889), ('importa
nt', 0.9118614196777344), ('causing', 0.9111785888671875), ('likely',
0.902948260307312)]
--------------------------------------------------
Word: patient
5 closest neighbors are:
[('given', 0.8689813613891602), ('normal', 0.8594050407409668), ('antib
iotic', 0.8579296469688416), ('antibiotics', 0.8565875291824341), ('han
d', 0.8512359857559204)]
--------------------------------------------------
```

Experiment 3: Model using CBOW, with embedding size = 300

```
In [25]: model3 = Word2Vec(corpus, min_count = 10, size = 500, #500
                         workers = 4, window = 5, sg = 0)
         ## cosine similarity
         word_list = ['disease', 'treatment', 'drug', 'effective', 'patient']
         for i in word_list:
             print('Word: %s' %i)
             print('5 closest neighbors are:')
             print(model3.wv.most_similar(i)[:5])
             print('-'*50)
```

```
Word: disease
5 closest neighbors are:
[('treatment', 0.9088848233222961), ('drug', 0.8956438302993774), ('dis
eases', 0.8427432179450989), ('common', 0.8276169896125793), ('lyme',
0.8261592388153076)]
--------------------------------------------------
Word: treatment
5 closest neighbors are:
[('drug', 0.93724125623703), ('disease', 0.9088848233222961), ('therap
y', 0.9019935131072998), ('common', 0.8979092836380005), ('risk', 0.893
3203220367432)]
--------------------------------------------------
Word: drug
5 closest neighbors are:
[('treatment', 0.9372413158416748), ('common', 0.9002607464790344), ('b
rain', 0.8975883722305298), ('disease', 0.8956438899040222), ('anti',
0.890290379524231)]
--------------------------------------------------
Word: effective
5 closest neighbors are:
[('common', 0.955552339553833), ('antibiotic', 0.917009174823761), ('si
gnificant', 0.9128857851028442), ('non', 0.9112560153007507), ('positiv
e', 0.9069941639900208)]
--------------------------------------------------
Word: patient
5 closest neighbors are:
[('given', 0.9010419249534607), ('condition', 0.8941221237182617), ('an
tibiotics', 0.8829540014266968), ('made', 0.8776760697364807), ('pms',
0.8770692348480225)]
--------------------------------------------------
```

Experiment 4: Model with skip-gram, with embedding size=100

```
In [10]: model4 = Word2Vec(corpus, min_count = 10, size = 100,
                          workers = 4, window = 5, sg = 1)
         ## cosine similarity
         word_list = ['disease', 'treatment', 'drug', 'effective', 'patient']
         for i in word_list:
             print('Word: %s' %i)
             print('5 closest neighbors are:')
             print(model4.wv.most_similar(i)[:5])
             print('-'*50)
```

```
Word: disease
5 closest neighbors are:
[('alzheimer', 0.737398087978363), ('coronary', 0.6893927454948425),
('lyme', 0.6597265601158142), ('diseases', 0.6523576974868774), ('diagn
osing', 0.643358588218689)]
--------------------------------------------------
Word: treatment
5 closest neighbors are:
[('radiation', 0.7403196096420288), ('invasive', 0.7140617370605469),
('ld', 0.7010745406150818), ('treating', 0.70063316822052), ('establis
h', 0.6939319968223572)]
--------------------------------------------------
Word: drug
5 closest neighbors are:
[('administration', 0.6866377592086792), ('radiation', 0.67578220367431
64), ('edta', 0.6732007265090942), ('particulate', 0.6669666171073914),
('approved', 0.666622519493103)]
--------------------------------------------------
Word: effective
5 closest neighbors are:
[('ad', 0.7535492181777954), ('prostate', 0.7476179003715515), ('safe',
0.732750654220581), ('element', 0.7310301065444946), ('treatable', 0.72
98332452774048)]
--------------------------------------------------
Word: patient
5 closest neighbors are:
[('practitioner', 0.7590723037719727), ('benefits', 0.739516496658325
2), ('lowered', 0.7163631319999695), ('medications', 0.704893708229064
9), ('pharmacist', 0.6945156455039978)]
--------------------------------------------------
```

Experiment 5: Model with skip-gram, with embedding size=200

In [11]:
```
model5 = Word2Vec(corpus, min_count = 10, size = 200,
                  workers = 4, window = 5, sg = 1)
## cosine similarity
word_list = ['disease', 'treatment', 'drug', 'effective', 'patient']
for i in word_list:
    print('Word: %s' %i)
    print('5 closest neighbors are:')
    print(model5.wv.most_similar(i)[:5])
    print('-'*50)
```

```
Word: disease
5 closest neighbors are:
[('diseases', 0.6587753295898438), ('lyme', 0.6549140214920044), ('alzh
eimer', 0.6486173868179321), ('coronary', 0.6450457572937012), ('infect
ious', 0.6255733966827393)]
--------------------------------------------------
Word: treatment
5 closest neighbors are:
[('radiation', 0.7156550288200378), ('ld', 0.6969175338745117), ('nizor
al', 0.6954584717750549), ('dysfunction', 0.6922903060913086), ('itraco
nazole', 0.689016580581665)]
--------------------------------------------------
Word: drug
5 closest neighbors are:
[('administration', 0.7087869048118591), ('approved', 0.704086244106292
7), ('multiple', 0.7028494477272034), ('radiation', 0.692239224910736
1), ('resistant', 0.6900363564491272)]
--------------------------------------------------
Word: effective
5 closest neighbors are:
[('prostate', 0.7670298218727112), ('typically', 0.7625733613967896),
('ad', 0.761953592300415), ('safe', 0.7570924162864685), ('formaldehyd
e', 0.7559657096862793)]
--------------------------------------------------
Word: patient
5 closest neighbors are:
[('medications', 0.6943399906158447), ('lowered', 0.6908197402954102),
('advised', 0.6907704472541809), ('practitioner', 0.6869900822639465),
('weakness', 0.6841328740119934)]
--------------------------------------------------
```

Experiment 6: Model with skip-gram, with embedding size=300

```
In [12]: model6 = Word2Vec(corpus, min_count = 10, size = 300,
                          workers = 4, window = 5, sg = 1)
         ## cosine similarity
         word_list = ['disease', 'treatment', 'drug', 'effective', 'patient']
         for i in word_list:
             print('Word: %s' %i)
             print('5 closest neighbors are:')
             print(model6.wv.most_similar(i)[:5])
             print('-'*50)
```

```
Word: disease
5 closest neighbors are:
[('alzheimer', 0.7332377433776855), ('coronary', 0.6997089385986328),
('lyme', 0.687203049659729), ('diseases', 0.6858249306678772), ('infect
ious', 0.6571913957595825)]
--------------------------------------------------
Word: treatment
5 closest neighbors are:
[('radiation', 0.7353566288948059), ('ld', 0.7236878871917725), ('exist
ent', 0.7073226571083069), ('chemotherapy', 0.7038858532905579), ('medi
cations', 0.6999843716621399)]
--------------------------------------------------
Word: drug
5 closest neighbors are:
[('administration', 0.762872576713562), ('approved', 0.726604580879211
4), ('method', 0.7100827693939209), ('reasons', 0.7014824748039246),
('gang', 0.7012131214141846)]
--------------------------------------------------
Word: effective
5 closest neighbors are:
[('safe', 0.7551177144050598), ('preference', 0.7534196376800537), ('pr
ostate', 0.7498774528503418), ('typically', 0.7496470808982849), ('ad',
0.7446816563606262)]
--------------------------------------------------
Word: patient
5 closest neighbors are:
[('practitioner', 0.7316721081733704), ('medications', 0.72210884094238
28), ('perspective', 0.7095988988876343), ('ordering', 0.70044261217117
31), ('oncologist', 0.691346287727356)]
--------------------------------------------------
```

Compare Window Sizes

Keep embedding size and model type the same

Experiment 7: Model with CBOW, with window=2

In [13]:
```python
model7 = Word2Vec(corpus, min_count = 10, size = 100,
                  workers = 4, window = 2, sg = 0)
## cosine similarity
word_list = ['disease', 'treatment', 'drug', 'effective', 'patient']
for i in word_list:
    print('Word: %s' %i)
    print('5 closest neighbors are:')
    print(model7.wv.most_similar(i)[:5])
    print('-'*50)
```

```
Word: disease
5 closest neighbors are:
[('treatment', 0.9078230261802673), ('diet', 0.8471344709396362), ('hea
rt', 0.8456168174743652), ('infectious', 0.8386905789375305), ('infecti
on', 0.8326027989387512)]
--------------------------------------------------
Word: treatment
5 closest neighbors are:
[('disease', 0.9078230857849121), ('side', 0.9040161371231079), ('ris
k', 0.8889400362968445), ('centers', 0.885377049446106), ('test', 0.881
4008235931396)]
--------------------------------------------------
Word: drug
5 closest neighbors are:
[('test', 0.8758785724639893), ('dietary', 0.8579474687576294), ('treat
ment', 0.8552394509315491), ('natural', 0.8436347246170044), ('method',
0.8435922861099243)]
--------------------------------------------------
Word: effective
5 closest neighbors are:
[('important', 0.9092000722885132), ('useful', 0.906681478023529), ('co
mmon', 0.9044443964958191), ('significant', 0.898126482963562), ('smal
l', 0.8973353505134583)]
--------------------------------------------------
Word: patient
5 closest neighbors are:
[('person', 0.9127682447433472), ('problem', 0.8936103582382202), ('con
dition', 0.8851459622383118), ('child', 0.8827368021011353), ('succes
s', 0.8825277090072632)]
--------------------------------------------------
```

Experiment 8: Model with CBOW, with window=10

In [14]:
```
model8 = Word2Vec(corpus, min_count = 10, size = 100,
                  workers = 4, window = 10, sg = 0)
## cosine similarity
word_list = ['disease', 'treatment', 'drug', 'effective', 'patient']
for i in word_list:
    print('Word: %s' %i)
    print('5 closest neighbors are:')
    print(model8.wv.most_similar(i)[:5])
    print('-'*50)
```

```
Word: disease
5 closest neighbors are:
[('treatment', 0.8798570036888123), ('drug', 0.8785004615783691), ('dis
eases', 0.8487732410430908), ('aids', 0.8359366059303284), ('common',
0.8267649412155151)]
--------------------------------------------------
Word: treatment
5 closest neighbors are:
[('drug', 0.9353200197219849), ('effective', 0.903861403465271), ('immu
ne', 0.8923046588897705), ('disease', 0.8798570036888123), ('fight', 0.
8785746693611145)]
--------------------------------------------------
Word: drug
5 closest neighbors are:
[('treatment', 0.9353200197219849), ('common', 0.908211886882782), ('ef
fective', 0.8985804915428162), ('divide', 0.8930118083953857), ('azt',
0.8846619725227356)]
--------------------------------------------------
Word: effective
5 closest neighbors are:
[('common', 0.957557201385498), ('non', 0.9338162541389465), ('fungal',
0.9305246472358704), ('divide', 0.921190619468689), ('antibiotic', 0.91
67952537536621)]
--------------------------------------------------
Word: patient
5 closest neighbors are:
[('given', 0.886145293712616), ('test', 0.8650234341621399), ('conditio
n', 0.8495171666145325), ('pms', 0.8494040966033936), ('iv', 0.84749376
77383423)]
--------------------------------------------------
```

Experiment 9: Model with skip-gram, with window=2

In [15]:
```python
model9 = Word2Vec(corpus, min_count = 10, size = 100,
                  workers = 4, window = 2, sg = 1)
## cosine similarity
word_list = ['disease', 'treatment', 'drug', 'effective', 'patient']
for i in word_list:
    print('Word: %s' %i)
    print('5 closest neighbors are:')
    print(model9.wv.most_similar(i)[:5])
    print('-'*50)
```

```
Word: disease
5 closest neighbors are:
[('alzheimer', 0.7246717214584351), ('infectious', 0.7190139889717102),
('coronary', 0.7138954401016235), ('virus', 0.6763666868209839), ('acti
ve', 0.6741172075271606)]
--------------------------------------------------
Word: treatment
5 closest neighbors are:
[('method', 0.8418763279914856), ('therapy', 0.8215181827545166), ('tes
ting', 0.7922003865242004), ('lung', 0.7819364666938782), ('instance',
0.7798306345939636)]
--------------------------------------------------
Word: drug
5 closest neighbors are:
[('administration', 0.81230628490448), ('method', 0.7706379294395447),
('therapy', 0.7333471775054932), ('test', 0.7325628995895386), ('iv',
0.7290377616882324)]
--------------------------------------------------
Word: effective
5 closest neighbors are:
[('particularly', 0.8253355622291565), ('safe', 0.8186452388763428),
('dangerous', 0.8179106712341309), ('factor', 0.816403865814209), ('tox
ic', 0.8153945207595825)]
--------------------------------------------------
Word: patient
5 closest neighbors are:
[('practitioner', 0.8112378716468811), ('medications', 0.79962295293807
98), ('remaining', 0.7677609324455261), ('argument', 0.76721668243408
2), ('plan', 0.7619680762290955)]
--------------------------------------------------
```

Experiment 10: Model with skip-gram, with window=10

In [16]:
```
model10 = Word2Vec(corpus, min_count = 10, size = 100,
                   workers = 4, window = 10, sg = 1)
## cosine similarity
word_list = ['disease', 'treatment', 'drug', 'effective', 'patient']
for i in word_list:
    print('Word: %s' %i)
    print('5 closest neighbors are:')
    print(model10.wv.most_similar(i)[:5])
    print('-'*50)
```

```
Word: disease
5 closest neighbors are:
[('alzheimer', 0.7316960096359253), ('diseases', 0.651874303817749),
('race', 0.6493644118309021), ('infectious', 0.6411278247833252), ('cor
onary', 0.6324220895767212)]
--------------------------------------------------
Word: treatment
5 closest neighbors are:
[('radiation', 0.6790452599525452), ('treatments', 0.6412014961242676),
('itraconazole', 0.6372858285903931), ('spirochete', 0.630698204040527
3), ('overuse', 0.6295500993728638)]
--------------------------------------------------
Word: drug
5 closest neighbors are:
[('administration', 0.708935558795929), ('depo', 0.7057257890701294),
('provera', 0.6962777376174927), ('injectable', 0.6919535994529724),
('resistant', 0.6792383193969727)]
--------------------------------------------------
Word: effective
5 closest neighbors are:
[('restricitng', 0.7241989374160767), ('prophylaxis', 0.700043439865112
3), ('agent', 0.6962985396385193), ('challenged', 0.677463710308075),
('prostate', 0.6732975244522095)]
--------------------------------------------------
Word: patient
5 closest neighbors are:
[('lowered', 0.6477791666984558), ('transfusion', 0.6323440074920654),
('persistently', 0.6240548491477966), ('invasive', 0.618701696395874),
('biopsy', 0.6171382665634155)]
--------------------------------------------------
```