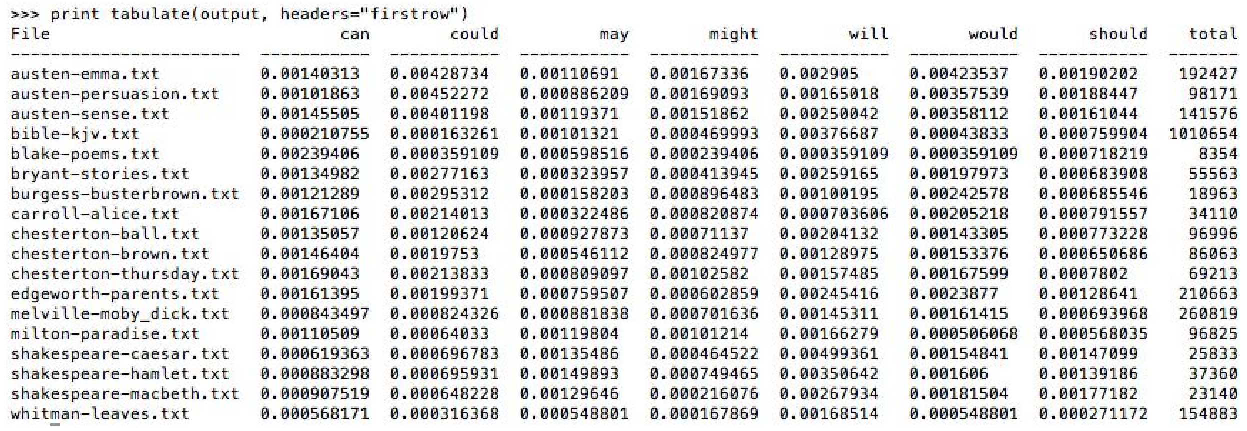
Ryan Ballenger
CSCIE63 Big Data Analytics
Assignment 12

Assignment 12 Solution

Problem 1. Create a table displaying **relative** frequencies with which "modals" (can, could, may, might, will, would and should) are used in 18 texts provided by NLTK in their extract from Gutenberg Corpus. For two modals with the largest span of relative frequencies (most used minus least used), select a text which uses it the most and the text that uses it the least. Compare usage in both texts by examining the concordances of those modals in two texts. Perhaps try to understand how are those words used in different texts.

```
// The current version of NLTK is installed with the pip command. This includes the
// necessary commands and sample texts.
Ryans-MacBook-Pro:~ Ryan$ sudo pip install -U nltk
...
/Library/Python/2.7/site-packages/pip-8.1.1-py2.7.egg/pip/_vendor/requests/packages/urllib3/util/ssl_.py:315: SNIMissingWarning: An HTTPS request has been made, but the SNI (Subject Name Indication) extension to TLS is not available on this platform. This may cause the server to present an incorrect TLS certificate, which can cause validation failures. For more information, see https://urllib3.readthedocs.org/en/latest/security.html#snimissingwarning.
  SNIMissingWarning
/Library/Python/2.7/site-packages/pip-8.1.1-py2.7.egg/pip/_vendor/requests/packages/urllib3/util/ssl_.py:120: InsecurePlatformWarning: A true SSLContext object is not available. This prevents urllib3 from configuring SSL appropriately and may cause certain SSL connections to fail. For more information, see
https://urllib3.readthedocs.org/en/latest/security.html#insecureplatformwarning.
  InsecurePlatformWarning
Requirement already up-to-date: nltk in /Library/Python/2.7/site-packages

// The python shell is started because this console supports the NLTK functionality.
Ryans-MacBook-Pro:~ Ryan$ python
Python 2.7.5 (default, Mar  9 2014, 22:15:05)
[GCC 4.2.1 Compatible Apple LLVM 5.0 (clang-500.0.68)] on darwin
Type "help", "copyright", "credits" or "license" for more information.

// The NLTK library is loaded and the download() command is used to download
// the book collection. In the widget view, the book collection is selected and downloaded.
>>> import nltk
>>> nltk.download()
showing info https://raw.githubusercontent.com/nltk/nltk_data/gh-pages/index.xml
True

// All of the nltk.book data is imported which includes the texts to be analyzed.
>>> from nltk.book import *
*** Introductory Examples for the NLTK Book ***
Loading text1, ..., text9 and sent1, ..., sent9
Type the name of the text or sentence to view it.
Type: 'texts()' or 'sents()' to list the materials.
text1: Moby Dick by Herman Melville 1851
text2: Sense and Sensibility by Jane Austen 1811
text3: The Book of Genesis
text4: Inaugural Address Corpus
text5: Chat Corpus
text6: Monty Python and the Holy Grail
text7: Wall Street Journal
text8: Personals Corpus
text9: The Man Who Was Thursday by G . K . Chesterton 1908

// The tabulate and copy libraries are loaded and the output array is initiated.
// A heading with the modal words is created and added to the output array.
// The modals are counted in each gutenberg file and divided by the total words per
// file to determine the relative frequency. The max and min relative frequencies are
// also saved for each word.
>>> from tabulate import tabulate
>>> import copy
>>> output = []
>>> wordList = ['can', 'could', 'may', 'might', 'will', 'would', 'should']
>>> maximum = dict()
>>> minimum = dict()
>>> # make and add heading to output
... heading = copy.copy(wordList)
>>> heading.insert(0, 'File')
>>> heading.append('total')
>>> output.append(heading)
>>> for fileid in gutenberg.fileids():
...     counts = []
...     # add the file name to the output array
...     counts.append(fileid)
...     for word in wordList:
...             # count the occurrences of each modal in the file
...             count = gutenberg.words(fileid).count(word)
...             # divide the count by the total number of words
...             count = float(count) / float(len(gutenberg.words(fileid)))
...             counts.append(count)
...             # add to max and min arrays if appropriate
...             if word not in maximum:
...                   maximum[word] = count
...                   minimum[word] = count
...             if(maximum[word]<count): maximum[word] = count
...             if(minimum[word]>count): minimum[word] = count
...     counts.append(len(gutenberg.words(fileid)))
...     output.append(counts)
...
```

// The relative frequencies per modal per file are printed along with the total words per file.
// The tabulate function is used to structure the output of the print.



```
>>> print tabulate(output, headers="firstrow")
File                              can        could          may        might         will        would       should    total
-----------------------  -----------  -----------  -----------  -----------  -----------  -----------  -----------  -------
austen-emma.txt          0.00140313   0.00428734   0.00110691   0.00167336   0.002905     0.00423537   0.00190202    192427
austen-persuasion.txt    0.00101863   0.00452272   0.000886209  0.00169093   0.00165018   0.00357539   0.00188447     98171
austen-sense.txt         0.00145505   0.00401198   0.00119371   0.00151862   0.00250042   0.00358112   0.00161044    141576
bible-kjv.txt            0.000210755  0.000163261  0.00101321   0.000469993  0.00376687   0.00043833   0.000759904  1010654
blake-poems.txt          0.00239406   0.000359109  0.000598516  0.000239406  0.000359109  0.000359109  0.000718219     8354
bryant-stories.txt       0.00134982   0.00277163   0.000323957  0.000413945  0.00259165   0.00197973   0.000683908    55563
burgess-busterbrown.txt  0.00121289   0.00295312   0.000158203  0.000896483  0.00100195   0.00242578   0.000685546    18963
carroll-alice.txt        0.00167106   0.00214013   0.000322486  0.000820874  0.000703606  0.00205218   0.000791557    34110
chesterton-ball.txt      0.00135057   0.00120624   0.000927873  0.00071137   0.00204132   0.00143305   0.000773228    96996
chesterton-brown.txt     0.00146404   0.0019753    0.000546112  0.000824977  0.00128975   0.00153376   0.000650686    86063
chesterton-thursday.txt  0.00169043   0.00213833   0.000809097  0.00102582   0.00157485   0.00167599   0.0007802      69213
edgeworth-parents.txt    0.00161395   0.00199371   0.000759507  0.000602859  0.00245416   0.0023877    0.00128641    210663
melville-moby_dick.txt   0.000843497  0.000824326  0.000881838  0.000701636  0.00145311   0.00161415   0.000693968   260819
milton-paradise.txt      0.00110509   0.00064033   0.00119804   0.00101214   0.00166279   0.000506068  0.000568035    96825
shakespeare-caesar.txt   0.000619363  0.000696783  0.00135486   0.000464522  0.00499361   0.00154841   0.00147099     25833
shakespeare-hamlet.txt   0.000883298  0.000695931  0.00149893   0.000749465  0.00350642   0.001606     0.00139186     37360
shakespeare-macbeth.txt  0.000907519  0.000648228  0.00129646   0.000216076  0.00267934   0.00181504   0.00177182     23140
whitman-leaves.txt       0.000568171  0.000316368  0.000548801  0.000167869  0.00168514   0.000548801  0.000271172   154883
```

// The difference between the maximum and minimum relative frequencies per word
// are calculated. The words “could” and “will” have the largest difference.

```
>>> difference = []
>>> for word in wordList:
...     difference.append([word, maximum[word] - minimum[word]]);
...
>>> print tabulate(difference);
------  ----------
can     0.00218331
could   0.00435946
may     0.00134073
might   0.00152306
will    0.0046345
would   0.00387626
should  0.00163085
```

```
------  ----------

// The word “will” occurs most in shakespeare-caesar.txt. The concordance function
// is used to examine the context in which the word is used.
>>> t = nltk.Text(nltk.corpus.gutenberg.words('shakespeare-caesar.txt'))
t.concordance("will")
>>> t.concordance("will")
Displaying 25 of 163 matches:
 way towards the Capitoll , This way will I : Disrobe the Images , If you do f
eathers , pluckt from Caesars wing , Will make him flye an ordinary pitch , Wh
eunt . Manet Brut . & Cass . Cassi . Will you go see the order of the course ?
, That you haue no such Mirrors , as will turne Your hidden worthinesse into y
I as by Reflection ; I your Glasse , Will modestly discouer to your selfe That
eye , and Death i ' th other , And I will looke on both indifferently : For le
s heauy : Coniure with ' em , Brutus will start a Spirit as soone as Caesar .
er moou ' d : What you haue said , I will consider : what you haue to say I wi
ll consider : what you haue to say I will with patience heare , and finde a ti
 Plucke Caska by the Sleeue , And he will ( after his sowre fashion ) tell you
proceeded worthy note to day Bru . I will do so : but looke you Cassius , The
nce , by some Senators Cassi . Caska will tell vs what the matter is Caes Anto
yet , if I could remember it Cassi . Will you suppe with me to Night , Caska ?
. No , I am promis ' d forth Cassi . Will you Dine with me to morrow ? Cask .
er worth the eating Cassi . Good , I will expect you Cask . Doe so : farewell
rut . And so it is : For this time I will leaue you : To morrow , if you pleas
 if you please to speake with me , I will come home to you : or if you will ,
 I will come home to you : or if you will , Come home to me , and I will wait
f you will , Come home to me , and I will wait for you Cassi . I will doe so :
 , and I will wait for you Cassi . I will doe so : till then , thinke of the W
Cassius , He should not humor me . I will this Night , In seuerall Hands , in
, let Caesar seat him sure , For wee will shake him , or worse dayes endure .
here in Italy Cassi . I know where I will weare this Dagger then ; Cassius fro
s Dagger then ; Cassius from Bondage will deliuer Cassius : Therein , yee Gods
omans Hindes . Those that with haste will make a mightie fire , Begin it with

// Similarly, “will” is used least in blake-poems.txt and the concordance function displays the
// context for this file too.
>>> t = nltk.Text(nltk.corpus.gutenberg.words('blake-poems.txt'))
>>> t.concordance("will")
Displaying 3 of 3 matches:
arn ' d the heat to bear , The cloud will vanish , we shall hear His voice , S
lver hair , And be like him , and he will then love me . THE BLOSSOM Merry , m
alone nor or itself : fear not and I will call , The weak worm from its lowly
```

Usage Comparison: The major difference is that the shakespeare text uses “will” frequently and to communicate several different parts of phrases. The poems on the other hand only use the word “will” three times and for the same purpose of referring to future actions. Shakespeare often uses the term to describe a decision. For example “I will consider” meaning I accept what you said but need to think about it. The term “will” is also used as a supportive verb in phrases such as “I will expect you” meaning you are supposed to meet me somewhere. There is also reference to someone's’ will, describing death-related instructions, and the term is used in the phrasing of questions such as “And will not palter ?” Overall, the term is very popular and broadly used in shakespeare’s writing style whereas it is seldomly used and for a single purpose in the blake poems text.

```
// The word ‘could’ has the second largest relative frequency range. It is used the
// most in austen-persuasion.txt and the concordance function displays the context.
>>> t = nltk.Text(nltk.corpus.gutenberg.words('austen-persuasion.txt'))
>>> t.concordance("could")

Displaying 25 of 451 matches:
every other leaf were powerless , he could read his own history with an interes
as still a very fine man . Few women could think more of their personal appeara
ersonal appearance than he did , nor could the valet of any new made lord be mo
l ; but it was only in Anne that she could fancy the mother to revive again . A
mild dark eyes from his own ), there could be nothing in them , now that she wa
ood looks of everybody else ; for he could plainly see how old all the rest of
self - possession and decision which could never have given the idea of her bei
heir , and whose strong family pride could see only in him a proper match for S
aronet from A to Z whom her feelings could have so willingly acknowledged as an
ing black ribbons for his wife , she could not admit him to be worth thinking o
 were hereafter to be his own . This could not be pardoned . Such were Elizabet
 alarm , set seriously to think what could be done , and had finally proposed t
l part of his estate that Sir Walter could dispose of ; but had every acre been
em , as anybody of sense and honesty could well be . She was a benevolent , cha
the most comprehensive retrenchments could secure , and saw no dignity in anyth
 Russell ' s had no success at all : could not be put up with , were not to be
id not appear to him that Sir Walter could materially alter his style of living
ondon ; but Mr Shepherd felt that he could not be trusted in London , and had b
beyond their own circle . Sir Walter could not have borne the degradation of be
it a friendship quite out of place , could hint of caution and reserve . Lady R
 . They will be all wanting a home . Could not be a better time , Sir Walter ,
d observed , Sir Walter ' s concerns could not be kept a secret ,)-- accidental
 man who knew it only by description could feel ; and given Mr Shepherd , in hi
an extraordinary taste , certainly , could they have been supposed in the secre
 than Admiral Croft bid fair to be , could hardly offer . So far went his under
```

```
// The term could is used the least in bible-kjv.txt and the concordance function
// displays the context of those occurrences below.
>>> t = nltk.Text(nltk.corpus.gutenberg.words('bible-kjv.txt'))
>>> t.concordance("could")
Displaying 25 of 166 matches:
r substance was great , so that they could not dwell together . 13 : 7 And ther
, and his eyes were dim , so that he could not see , he called Esau his eldest
the land wherein they were strangers could not bear them because of their cattl
 his brethren , they hated him , and could not speak peaceably unto him . 37 :
 his dream ; but there was none that could interpret them unto Pharaoh . 41 : 9
And when they had eaten them up , it could not be known that they had eaten the
 magicians ; but there was none that could declare it to me . 41 : 25 And Josep
ording to the tenor of these words : could we certainly know that he would say
me on my father . 45 : 1 Then Joseph could not refrain himself before all them
y father yet live ? And his brethren could not answer him ; for they were troub
Israel were dim for age , so that he could not see . And he brought them near u
im three months . 2 : 3 And when she could not longer hide him , she took for h
 the river stank , and the Egyptians could not drink of the water of the river
 river for water to drink ; for they could not drink of the water of the river
ments to bring forth lice , but they could not : so there were lice upon man ,
pon beast . 9 : 11 And the magicians could not stand before Moses because of th
 they were thrust out of Egypt , and could not tarry , neither had they prepare
3 And when they came to Marah , they could not drink of the waters of Marah , f
y the dead body of a man , that they could not keep the passover on that day :
 12 Therefore the children of Israel could not stand before their enemies , but
of Jerusalem , the children of Judah could not drive them out ; but the Jebusit
17 : 12 Yet the children of Manasseh could not drive out the inhabitants of tho
he inhabitants of the mountain ; but could not drive out the inhabitants of the
r enemies round about , so that they could not any longer stand before their en
t closed upon the blade , so that he could not draw the dagger out of his belly
```

Usage comparison: The term “could” is most often used for one purpose in the bible-kjv.txt as a setup for a negative declaration. For example, “Israel **could not** stand before their enemies”. The majority of its uses occur in this sentence structure where “not” follows “could”. In austen-persuasion.txt, the term “could” is used in more contexts that are similar to its current usage in the English language. For example it describes a person’s capabilities in “he could read his own history”. It is used to precede “not” but that does not consist of the majority of its use as in the previous text. There is no negative such as “not” in the following sentence and the term “could” is being used to describe relationship complexities: “she could not admit him”. The author of the austen text regularly uses “could” in a variety of sentence structures while the bible-kjv.txt author seldom uses it and mostly for a “not” sentence structure.

Problem 2. In the Inaugural corpus identify 10 most frequently used words longer than 7 characters. Which one of those has the largest number of synonyms? List all synonyms for those 10 words. Which one of those 10 words has the largest number of hyponyms? List all hyponyms of those 10 most frequently used "long" words.

```
// FreqDist counts the occurrences of words in the inaugural corpus and the top 200
// are extracted to find the most common words with length of 7 or more.
>>> mostCommon = FreqDist(inaugural.words()).most_common(200)
>>> results  = set()
>>> for x in range(0, 200):
...     if(len(mostCommon[x][0]) > 7):
...             results.add(mostCommon[x][0].lower())
...     if(len(results)==10): break
...

// The most common words are iterated and the words that are longer than 7 are
// saved in the results array. The loop ends when 10 words that meet the criteria are found.

>>> results
set([u'interests', u'constitution', u'congress', u'government', u'national', u'citizens', u'political',
u'principles', u'american', u'progress'])


// The wordnet library is loaded to find synonyms. For the synonym sets for each word,
// the synonyms are retrieved with lemma_names() and added to a set. The word, the
// synonym set, and the size of the set are printed.
>>> from nltk.corpus import wordnet
>>> for word in results:
...     theSet = set()
...     for one in wordnet.synsets(word):
...             for syno in one.lemma_names():
...                     theSet.add(syno.lower())
...     print word
...     print theSet
...     print len(theSet)
...     print "\n"
...
interests
set([u'sake', u'stake', u'interest_group', u'matter_to', u'interestingness', u'occupy', u'involvement',
u'pursuit', u'interest', u'pastime', u'worry', u'concern'])
12
```

```
constitution
set([u'make-up', u'makeup', u'constitution', u'us_constitution', u'organisation', u'composition',
u'constitution_of_the_united_states', u'fundamental_law', u'physical_composition', u'formation',
u'u.s._constitution', u'organization', u'united_states_constitution', u'establishment',
u'organic_law', u'old_ironsides'])
16


congress
set([u'congress', u'intercourse', u'sexual_intercourse', u'sex_act', u'coition', u'sexual_relation',
u'carnal_knowledge', u'sexual_congress', u'relation', u'u.s._congress', u'copulation', u'coitus',
u'united_states_congress', u'us_congress'])
14


government
set([u'government', u'administration', u'governance', u'political_science', u'government_activity',
u'governing', u'authorities', u'politics', u'regime'])
9


national
set([u'interior', u'home', u'national', u'internal', u'subject'])
5


citizens
set([u'citizen'])
1


political
set([u'political'])
1


principles
set([u'precept', u'rationale', u'principle', u'rule'])
4
```

```
american
set([u'american', u'american_english', u'american_language'])
3


progress
set([u'advance', u'forward_motion', u'move_on', u'come_along', u'progression', u'shape_up',
u'get_on', u'march_on', u'advancement', u'procession', u'build_up', u'build', u'onward_motion',
u'work_up', u'come_on', u'progress', u'get_along', u'pass_on', u'go_on'])
19
```

// “progress” has the most synonyms with a total of 19.

// The hyponyms for the 10 words are found. The synonym sets are determined for
// each word and then the hyponyms for each synset are retrieved and added to a set.

```
>>> for word in results:
...     theSet = set()
...     for one in wordnet.synsets(word):
...             for hyp in one.hyponyms():
...                     theSet.add(hyp.name().lower())
...     print word
...     print theSet
...     print len(theSet)
...     print "\n"
...
interests
set([u'newsworthiness.n.01', u'undivided_interest.n.01', u'topicality.n.01', u'reversion.n.01',
u'concern.n.01', u'insurable_interest.n.01', u'behalf.n.02', u'intrigue.v.01',
u'security_interest.n.01', u'fee.n.02', u'fascinate.v.02', u'right.n.08', u'grubstake.n.01',
u'simple_interest.n.01', u'avocation.n.01', u'special_interest.n.01', u'color.n.02',
u'compound_interest.n.01', u'equity.n.02', u'shrillness.n.01', u'terminable_interest.n.01',
u'charisma.n.01', u'absorb.v.09', u'enthusiasm.n.03', u'controlling_interest.n.01',
u'vested_interest.n.01', u'vested_interest.n.02'])
27


constitution
set([u'collectivization.n.01', u'karyotype.n.01', u'colonization.n.01', u'structure.n.02',
u'texture.n.05', u'phenotype.n.01', u'unionization.n.01', u'genotype.n.02', u'communization.n.02',
u'federation.n.03'])
10
```

```
congress
set([u'hank_panky.n.01', u'unlawful_carnal_knowledge.n.01', u'defloration.n.02', u'fuck.n.01',
u'penetration.n.06', u'continental_congress.n.01'])
6


government
set([u'puppet_government.n.01', u'downing_street.n.02', u'totalitarian_state.n.01',
u'realpolitik.n.01', u'authoritarian_state.n.01', u'bureaucracy.n.02', u'local_government.n.01',
u'state_government.n.01', u'trust_busting.n.01', u'geopolitics.n.01', u'papacy.n.01',
u'legislation.n.02', u'state.n.03', u'ancien_regime.n.01', u'court.n.03', u'misgovernment.n.01',
u'empire.n.02', u'military_government.n.01', u'palace.n.02', u'government-in-exile.n.01',
u'federal_government.n.01'])
21


national
set([u'patriot.n.01', u'compatriot.n.01', u'citizen.n.01'])
3


citizens
set([u'thane.n.02', u'repatriate.n.01', u'voter.n.01', u'freeman.n.01', u'active_citizen.n.01',
u'private_citizen.n.01', u'civilian.n.01'])
7


political
set([])
0


principles
set([u'mass-energy_equivalence.n.01', u'scruple.n.03', u'feng_shui.n.01',
u'principle_of_equivalence.n.01', u'principle_of_superposition.n.02', u'pillar.n.01', u'ethic.n.01',
u'principle_of_superposition.n.01', u'moral_principle.n.02',
u'principle_of_liquid_displacement.n.01', u'hellenism.n.01', u'legal_principle.n.01',
u'higher_law.n.01', u"occam's_razor.n.01", u'mass-action_principle.n.01',
u"naegele's_rule.n.01", u'logic.n.03', u'pleasure_principle.n.01', u"gresham's_law.n.01",
u'dialectics.n.01', u'localization_of_function.n.01', u'caveat_emptor.n.01',
u'accounting_principle.n.01', u"le_chatelier's_principle.n.01", u'dictate.n.02', u'yang.n.01',
u'fundamentals.n.01', u'insurrectionism.n.01', u'tao.n.02', u'chivalry.n.02',
u'hypothetical_imperative.n.01', u'gestalt_law_of_organization.n.01', u'conservation.n.03',
u'yin.n.01', u'reality_principle.n.01'])
35


american
set([u'arizonan.n.01', u'tory.n.01', u'texan.n.01', u'franco-american.n.01', u'hawaiian.n.02',
u'kansan.n.01', u'bostonian.n.01', u'nebraskan.n.01', u'south_american.n.01',
u'north_carolinian.n.01', u'creole.n.01', u'alabaman.n.01', u'tennessean.n.01',
u'asian_american.n.01', u'spanish_american.n.01', u'delawarean.n.01', u'new_yorker.n.01',
u'georgian.n.01', u'south_carolinian.n.01', u'new_mexican.n.01', u'coloradan.n.01',
u'kentuckian.n.01', u'west_indian.n.01', u'missourian.n.01', u'west_virginian.n.01',
u'african_american_vernacular_english.n.01', u'yankee.n.03', u'virginian.n.01',
u'wisconsinite.n.01', u'idahoan.n.01', u'german_american.n.01', u'indianan.n.01', u'yankee.n.01',
u'pennsylvanian.n.02', u'puerto_rican.n.01', u'alaskan.n.01', u'vermonter.n.01',
u'new_englander.n.01', u'ohioan.n.01', u'new_jerseyan.n.01', u'californian.n.01',
u'north_dakotan.n.01', u'washingtonian.n.01', u'bay_stater.n.01', u'mississippian.n.02',
u'african-american.n.01', u'montanan.n.01', u'marylander.n.01', u'illinoisan.n.01',
u'louisianan.n.01', u'connecticuter.n.01', u'new_hampshirite.n.01', u'michigander.n.01',
u'floridian.n.01', u'minnesotan.n.01', u'iowan.n.01', u'mesoamerican.n.01',
u'north_american.n.01', u'nisei.n.01', u'washingtonian.n.02', u'carolinian.n.01',
u'southerner.n.01', u'creole.n.02', u'mainer.n.01', u'south_dakotan.n.01', u'wyomingite.n.01',
u'arkansan.n.01', u'oregonian.n.01', u'rhode_islander.n.01', u'oklahoman.n.01', u'nevadan.n.01',
u'latin_american.n.01', u'anglo-american.n.01', u'utahan.n.01', u'appalachian.n.01'])
75


progress
set([u'pass.v.07', u'work_flow.n.01', u'penetrate.v.05', u'march.n.03', u'press_on.v.01',
u'career.n.02', u'ratchet.v.01', u'leapfrog.n.01', u'edge.v.01', u'leapfrog.v.02', u'forwarding.n.02',
u'plain_sailing.n.01', u'headway.n.02', u'stride.n.03', u'close_in.v.01', u'encroach.v.01',
u'forge.v.04', u'elapse.v.01', u'creep_up.v.01', u'push.n.05', u'string.v.03', u'climb.v.05'])
22
```

// American has the most hyponyms with a total of 75.