IST 664-Homework 1

Submitted by Sandya Madhavan

Contents

Introduction:

This assignment deals with the analysis of State of the Union Addresses dataset which is a collection of annual speeches delivered by the presidents of the United States, from George Washington to Barack Obama, to a joint session of the United States Congress for the span of 1790-2016. The goal is to find the most common words and bigrams in the content that is spread across two files and reason the similarity or difference regarding the same.

Dataset description:

Three txt files were provided as part of the assignment. They are ‘State_union_part1.txt’, ‘State_union_part2.txt’, ‘State_union_policy.txt’. The files have been named in the format ‘state_union_***.txt’ , where ‘***’ varies depending on the type of file it is. ‘State_union_part1.txt’ contains the annual US presidential speeches delivered between the years 1790 and 1860 whereas ‘State_union_part2.txt’ contains the presidential address delivered between the years 1946 and 2016.The third file ‘state_union_policy’.txt’ contains the policies regarding the usage of content regarding the US presidential addresses. These files are a part of project Gutenberg. Gutenberg Corpus is gratis, libre, and completely without cost to readers. Content from Gutenberg corpus can be accessed via their respective URLs for further processing. Here, the afore mentioned files were copied from the Gutenberg website and provided as txt files. The text files can be read in using a Plain text corpus Reader for further processing and analysis

Tools

Spyder IDE - For Python coding

MS-Excel- For offline analysis

Data loading process

The Text files “state_union_part1.txt” and “state_union_part2.txt” were loaded using nltk package and PlaintextCorpusReader in Spyder IDE

Analysis 1: Analysis of State of the Union Addresses dataset: Part1

The following tasks are analyzed with “state_union_part1.txt” file.

a) List of top 50 words by frequency (normalized by the length of the document)

The following steps were performed via python coding to list the top 50 words based on word frequency

Step 1: Load the state_union_part1.txt using nltk package and PlaintextCorpusReader
Step 2: Perform tokenization to separate words from the text file
Step 3: Convert tokenized words to lower case
Step 4: Calculate the frequency distribution of the tokenized words

There are several punctuation marks and stop words as a part of the result obtained from step 4. So those needs to be removed.

Step 5: Filter or remove the non-alphabetical characters from the tokenized words
Step 6: Perform stop words filtering on the filtered words from step 5 using pre-defined stop words from English language. Add user defined stop words to the stop words list and perform a stop word filtering to get a final list of words without stop words.
Step 7: Calculate frequency distribution of the stop words by dividing the frequency by the total no.of words in the text file.
Step 8: Find the top 50 words by frequency based using most_common function

The resultant of this process is the top 50 words by frequency.

b) List of top 50 bigrams by frequencies

The following steps were performed via python coding to list the top 50 bigrams by frequencies

Step 1: import collocation finder package from nltk to calculate bigram measure
Step 2: Apply collocation finder to the initial list of tokenized words from step 3 of previous task (task a)
Step 3: print the bigrams with their raw frequencies

There are several punctuation marks and stop words as a part of the result obtained from step 3. So those needs to be removed.

Step 4: Use the finder function with the filter created to remove non-alphabetical characters in order to remove non-alphabetical characters from tokenized words
Step 5: Use the finder function with the stop words filter
Step 6:print the top 50 bigrams with their raw frequencies

c) List of top 50 bigrams by their Mutual Information scores (using min frequency 5)

The following steps were performed via python coding to list the top 50 bigrams by Mutual information score

Step 1: Use a finder to identify the bigrams with min frequency as 5
Step 2:print the top 50 bigrams with their pmi scores

Note: Stop word filtering and non-alphabetical filtering were not applied after a Manual verification was carried out on the generated bigrams.

Output of Analysis 1:

The following table is the output of the three tasks of analysis 1

Text 1-top 50 words by frequency	**Text 1-top 50 bigrams by frequencies**	**Text 1-top 50 bigrams by their PMI**
('states', 0.0008684605715713495)	(('united', 'states'), 0.003265589965928175)	(('bona', 'fide'), 16.767819779008715)
('government', 0.0007075165023443655)	(('great', 'britain'), 0.000491093112329484)	(('posse', 'comitatus'), 16.767819779008715)

('united', 0.0005940589010675213)	(('last', 'session'), 0.0004337391722034129)	(('punta', 'arenas'), 16.767819779008715)
('may', 0.000497811160658513)	(('public', 'debt'), 0.00032082360258021037)	(('ballot', 'box'), 16.50478537317492)
('congress', 0.00047805169077321994)	(('fiscal', 'year'), 0.0002580927305673201)	(('del', 'norte'), 16.50478537317492)
('upon', 0.0004637101400500233)	(('union', 'address'), 0.0002580927305673201)	(('millard', 'fillmore'), 16.50478537317492)
('public', 0.0004382140498754516)	(('public', 'lands'), 0.00023300038176216395)	(('guadalupe', 'hidalgo'), 15.919822872453764)
('country', 0.0003706494109128365)	(('two', 'countries'), 0.00021866189673064619)	(('porto', 'rico'), 15.919822872453764)
('great', 0.00034196630946644333)	(('present', 'year'), 0.00018998492666761062)	(('franklin', 'pierce'), 15.767819779008715)
('made', 0.00033814189594025756)	(('fellow', 'citizens'), 0.0001738541310071531)	(('la', 'plata'), 15.630316255258778)
('state', 0.0003330426779053432)	(('general', 'government'), 0.00016668488849139423)	(('vera', 'cruz'), 15.504785373174922)
('last', 0.00029033672686293554)	(('british', 'government'), 0.0001648925778624545)	(('entangling', 'alliances'), 15.43439604528352)
('war', 0.0002657967400699103)	(('two', 'governments'), 0.00015951564597563533)	(('gun', 'boats'), 15.112467950396159)
('present', 0.0002587853152719031)	(('federal', 'government'), 0.00015234640345987644)	(('costa', 'rica'), 15.089747873896076)
('time', 0.00025751051076317445)	(('annual', 'message'), 0.00014517716094411754)	(('nucleus', 'around'), 15.089747873896076)
('people', 0.0002504990859651672)	(('public', 'service'), 0.00014338485031517782)	(('santa', 'anna'), 15.002285032645737)
('year', 0.0002501803848379851)	(('year', 'ending'), 0.00013980022905729837)	(('santa', 'fe'), 15.002285032645737)
('power', 0.00023711363862351708)	(('last', 'annual'), 0.00013442329717047922)	(('van', 'buren'), 15.002285032645737)
('citizens', 0.000230420914952692)	(('public', 'money'), 0.00012366943339684087)	(('project', 'gutenberg'), 15.002285032645736)
('subject', 0.00022659650142650624)	(('indian', 'tribes'), 0.0001182925015100217)	(('sublime', 'porte'), 14.96046485695111)
('shall', 0.00022117858226440974)	(('mexican', 'government'), 0.00011650019088108198)	(('martin', 'van'), 14.832360031203425)
('without', 0.00021129884732176322)	(('treasury', 'notes'), 0.00011650019088108198)	(('ad', 'valorem'), 14.76781977900871)
('union', 0.00020492482477812028)	(('commercial', 'intercourse'), 0.00011291556962320254)	(('quincy', 'adams'), 14.63031625525878)
('act', 0.00019982560674320594)	(('several', 'states'), 0.0001021617058495642)	(('water', 'witch'), 14.630316255258778)
('treaty', 0.0001988695033616595)	(('new', 'mexico'), 0.00010036939522062447)	(('statute', 'book'), 14.566185917839064)
('one', 0.0001975946988529309)	(('favorable', 'consideration'), 9.857708459168476e-05)	(('buenos', 'ayres'), 14.50478537317492)

('part', 0.0001969572965985666)	(('naval', 'force'), 9.857708459168476e-05)	(('de', 'facto'), 14.356393533282247)
('mexico', 0.0001928141819451987)	(('central', 'america'), 9.140784207592586e-05)	(('franking', 'privilege'), 14.334860371732606)
('general', 0.0001915393774364701)	(('present', 'session'), 9.140784207592586e-05)	(('rocky', 'mountains'), 14.282392951838471)
('every', 0.0001880336650374665)	(('french', 'government'), 8.961553144698614e-05)	(('andrew', 'jackson'), 14.199930791646498)
('treasury', 0.0001880336650374665)	(('new', 'york'), 8.782322081804641e-05)	(('retired', 'list'), 14.144889428088536)
('necessary', 0.0001832531481297343)	(('friendly', 'relations'), 8.603091018910669e-05)	(('circulating', 'medium'), 14.045353754537622)
('constitution', 0.00017751652784045566)	(('existing', 'laws'), 8.065397830228753e-05)	(('john', 'quincy'), 14.002285032645739)
('new', 0.00017464821769581636)	(('good', 'faith'), 8.065397830228753e-05)	(('precious', 'metals'), 13.94339134359217)
('duty', 0.00016859289627935557)	(('american', 'citizens'), 7.706935704440808e-05)	(('thomas', 'jefferson'), 13.914822191395396)
('foreign', 0.0001654058850075341)	(('foreign', 'nations'), 7.706935704440808e-05)	(('lake', 'erie'), 13.860929183400197)
('two', 0.0001625375748628948)	(('taken', 'place'), 7.706935704440808e-05)	(('almighty', 'god'), 13.832360031203425)
('commerce', 0.0001612627703541662)	(('last', 'year'), 7.527704641546836e-05)	(('john', 'tyler'), 13.832360031203425)
('nations', 0.0001599879658454376)	(('past', 'year'), 7.527704641546836e-05)	(('san', 'francisco'), 13.80434565503383)
('peace', 0.00015966926471825546)	(('september', 'last'), 7.527704641546836e-05)	(('san', 'jacinto'), 13.804345655033828)
('system', 0.00015743835682798042)	(('american', 'people'), 7.348473578652863e-05)	(('san', 'juan'), 13.804345655033828)
('laws', 0.00015680095457361613)	(('military', 'force'), 7.348473578652863e-05)	(('per', 'cent'), 13.732195869277994)
('duties', 0.00015552615006488756)	(('ensuing', 'year'), 7.169242515758891e-05)	(('rio', 'grande'), 13.697430451117315)
('within', 0.00015265783992024823)	(('minister', 'plenipotentiary'), 7.169242515758891e-05)	(('inferior', 'quality'), 13.623773409392006)
('law', 0.00015202043766588395)	(('slave', 'trade'), 6.990011452864918e-05)	(('grateful', 'acknowledgments'), 13.597894777566399)
('us', 0.0001475586218853339)	(('spanish', 'government'), 6.990011452864918e-05)	(('hudsons', 'bay'), 13.535159022218439)
('interests', 0.00014373420835914812)	(('charge', "d'affaires"), 6.631549327076975e-05)	(('cumberland', 'road'), 13.373540839896668)
('interest', 0.0001415033004688731)	(('present', 'fiscal'), 6.631549327076975e-05)	(('st.', 'marys'), 13.36182741933288)
('amount', 0.00014118459934169095)	(('two', 'nations'), 6.631549327076975e-05)	(('st.', 'croix'), 13.361827419332878)
('also', 0.00013959109370578022)	(('great', 'extent'), 6.27308720128903e-05)	(('st.', 'petersburg'), 13.361827419332878)

Analysis 2: Analysis of State of the Union Addresses dataset: Part2

Repeat Analysis 1 with “state_union_part2.txt” file.

a) List of top 50 words by frequency (normalized by the length of the document)

The following steps were performed via python coding to list the top 50 words based on word frequency

Step 1: Load the state_union_part2.txt using nltk package and PlaintextCorpusReader
Step 2: Perform tokenization to separate words from the text file
Step 3: Convert tokenized words to lower case
Step 4: Calculate the frequency distribution of the tokenized words

There are several punctuation marks and stop words as a part of the result obtained from step 4. So those needs to be removed.

Step 5: Filter or remove the non-alphabetical characters from the tokenized words
Step 6: Perform stop words filtering on the filtered words from step 5 using pre-defined stop words from English language. Add user defined stop words to the stop words list and perform a stop word filtering to get a final list of words without stop words.
Step 7: Calculate frequency distribution of the stop words by dividing the frequency by the total no. of words in the text file.
Step 8: Find the top 50 words by frequency based using most_common function

The resultant of this process is the top 50 words by frequency.

b) List of top 50 bigrams by frequencies

The following steps were performed via python coding to list the top 50 bigrams by frequencies

Step 1: import collocation finder package from nltk to calculate bigram measure
Step 2: Apply collocation finder to the initial list of tokenized words from step 3 of previous task (task a)
Step 3: print the bigrams with their raw frequencies

There are several punctuation marks and stop words as a part of the result obtained from step 3. So those needs to be removed.

Step 4: Use the finder function with the filter created to remove non-alphabetical characters in order to remove non-alphabetical characters from tokenized words
Step 5: Use the finder function with the stop words filter
Step 6:print the top 50 bigrams with their raw frequencies

c) List of top 50 bigrams by their Mutual Information scores (using min frequency 5)

The following steps were performed via python coding to list the top 50 bigrams by Mutual information score

Step 1: Use a finder to identify the bigrams with min frequency as 5

Step 2:print the top 50 bigrams with their pmi scores

Note: No further filtering was applied after a manual verification on the generated bigrams.

Output of Analysis 2:

Text 2-top 50 words by frequency	**Text 2-top 50 bigrams by frequencies**	**Text 2-top 50 bigrams by their PMI**
(‘people', 0.0005744320522405138)	((‘united', 'states'), 0.0009541097969728698)	((‘el', 'salvador'), 16.30034362211258)
(‘world', 0.0005683291884716903)	((‘american', 'people'), 0.0004935762802521989)	((‘bin', 'laden'), 16.077951200776134)
(‘new', 0.0005496391681796683)	((‘last', 'year'), 0.0004646638621621119)	((‘saudi', 'arabia'), 16.07795120077613)
(‘america', 0.00048479624063591843)	((‘fiscal', 'year'), 0.00038412212605401253)	((‘gerald', 'r.'), 15.662913701497288)
(‘year', 0.0004825076667226096)	((‘federal', 'government'), 0.00037999178061257153)	((‘sam', 'rayburn'), 15.62227171699994)
(‘congress', 0.00046915765222830814)	((‘social', 'security'), 0.00037379626245041006)	((‘jimmy', 'carter'), 15.563378027946374)
(‘us', 0.00046381764643058755)	((‘health', 'care'), 0.00036760074428824854)	((‘northern', 'ireland'), 15.300343622112582)
(‘government', 0.0004237676029476832)	((‘let', 'us'), 0.00035934005340536654)	((‘r.', 'ford'), 15.203482082859992)
(‘years', 0.0004237676029476832)	((‘years', 'ago'), 0.0003345579807567206)	((‘lyndon', 'b.'), 15.184866404692645)
(‘american', 0.00036235753627389655)	((‘union', 'address'), 0.0002849938354594287)	((‘floor', 'appears'), 15.148340528667532)
(‘nation', 0.0003284103565598157)	((‘united', 'nations'), 0.00027879831729726715)	((‘iron', 'curtain'), 15.077951200776134)
(‘one', 0.0003066689043833819)	((‘billion', 'dollars'), 0.0002684724536936647)	((‘grass', 'roots'), 15.037309216278786)
(‘every', 0.0002975146087301466)	((‘million', 'dollars'), 0.00026227693553150316)	((‘200th', 'anniversary'), 14.97841552722522)
(‘make', 0.0002967517507590437)	((‘soviet', 'union'), 0.00025814659009006216)	((‘william', 'j.'), 14.97841552722522)
(‘work', 0.0002875974551058084)	((‘free', 'world'), 0.00022303865383781373)	((‘thomas', 'jefferson'), 14.921831998858849)
(‘federal', 0.0002837831652502937)	((‘ca', "n't"), 0.00021064761751349073)	((‘red', 'tape'), 14.885306122833738)
(‘time', 0.0002826388782936393)	((‘every', 'american'), 0.00020445209935132926)	((‘jill', 'biden'), 14.814916794942338)
(‘states', 0.0002711960087270952)	((‘economic', 'growth'), 0.00019412623574772676)	((‘b.', 'johnson'), 14.797843281583399)
(‘americans', 0.0002624231420594114)	((‘middle', 'east'), 0.00018793071758556527)	((‘barack', 'obama'), 14.797843281583395)
(‘help', 0.0002616602840883084)	((‘make', 'sure'), 0.00018173519942340377)	((‘teen', 'pregnancy'), 14.715381121391424)
(‘security', 0.000261278855102757)	((‘free', 'nations'), 0.0001734745085405218)	((‘abraham', 'lincoln'), 14.627918280141085)

('war', 0.0002570831362616908)	(('first', 'time'), 0.0001672789903783603)	(('p.', "o'neill"), 14.57787759764149)
('economic', 0.0002559388493050364)	(('four', 'years'), 0.0001672789903783603)	(('j.', 'clinton'), 14.563378027946376)
('peace', 0.00025479456234838197)	(('armed', 'forces'), 0.0001548879540540373)	(('empowerment', 'zones'), 14.492988700054976)
('united', 0.00024831026959400696)	(('world', 'war'), 0.0001528227813333168)	(('ronald', 'reagan'), 14.425874504196438)
('nations', 0.0002460216956806982)	(('21st', 'century'), 0.0001507576086125963)	(('synthetic', 'fuels'), 14.411374934501325)
('also', 0.00024373312176738935)	(('work', 'together'), 0.0001486924358918758)	(('small-business', 'owner'), 14.399879295663494)
('program', 0.00024335169278183789)	(('foreign', 'policy'), 0.0001445620904504348)	(('harry', 's.'), 14.222341110111305)
('country', 0.00024030026089742613)	(('mr.', 'speaker'), 0.0001445620904504348)	(('dwight', 'd.'), 14.162840098362643)
('national', 0.00023229025220084524)	(('new', 'jobs'), 0.0001445620904504348)	(('intercontinental', 'ballistic'), 14.139351745440276)
('economy', 0.00022428024350426437)	(('two', 'years'), 0.00013836657228827334)	(('w.', 'bush'), 14.13041862067027)
('great', 0.00022237309857650703)	(('vice', 'president'), 0.00013836657228827334)	(('small-business', 'owners'), 14.077951200776134)
('last', 0.00021817737973544085)	(('national', 'security'), 0.00012804070868467084)	(('thomas', 'p.'), 14.047362880942709)
('many', 0.00021474451886547763)	(('human', 'rights'), 0.00012184519052250935)	(('river', 'basins'), 14.027325127706165)
('free', 0.0002128373739377203)	(('health', 'insurance'), 0.00011978001780178885)	(('status', 'quo'), 14.027325127706163)
('first', 0.00021093022900996292)	(('fellow', 'americans'), 0.00011564967236034786)	(('prime', 'minister'), 13.978415527225216)
('let', 0.00020940451306775704)	(('fellow', 'citizens'), 0.00011564967236034786)	(('nationwide', 'radio'), 13.937773542727872)
('state', 0.00019834307248676442)	(('past', 'year'), 0.00011564967236034786)	(('project', 'gutenberg'), 13.797843281583399)
('tax', 0.00019605449857345558)	(('civil', 'rights'), 0.00011151932691890686)	(('f.', 'kennedy'), 13.797843281583397)
('know', 0.0001933844956745953)	(('young', 'people'), 0.00011151932691890686)	(('al', 'qaeda'), 13.75602310588877)
('million', 0.0001933844956745953)	(('private', 'sector'), 0.00010738898147746586)	(('al', 'qaida'), 13.756023105888769)
('freedom', 0.00019185877973238943)	(('god', 'bless'), 0.00010532380875674537)	(('richard', 'nixon'), 13.73894959252983)
('budget', 0.00019109592176128648)	(('local', 'governments'), 0.00010532380875674537)	(('george', 'h.w'), 13.715381121391424)
('health', 0.00018651877393466884)	(('nuclear', 'weapons'), 0.00010532380875674537)	(('george', 'w.'), 13.715381121391424)
("n't", 0.00018270448407915415)	(('interest', 'rates'), 0.00010119346331530438)	(('d.', 'eisenhower'), 13.703408479725347)
('future', 0.00018117876813694827)	(('next', 'year'), 0.00010119346331530438)	(('line-item', 'veto'), 13.675852757204785)

('system', 0.00017660162031033063)	(('balanced', 'budget'), 9.912829059458388e-05)	(('saddam', 'hussein'), 13.66647152091048)
('programs', 0.00017622019132477914)	(('high', 'school'), 9.706311787386338e-05)	(('supreme', 'court'), 13.66130444863563)
('tonight', 0.00017583876233922768)	(('minimum', 'wage'), 9.706311787386338e-05)	(('carbon', 'pollution'), 13.605198203641)
('union', 0.00017545733335367622)	(('war', 'ii'), 9.706311787386338e-05)	(('baby', 'boom'), 13.599903903971487)

Comparison on Analysis 1 and Analysis 2:

Comparison of results from text 1 and text 2 based on the use of the language

1) **Comparing the results of top 50 words from analysis 1 and 2:**

Out of the top 50 words that were generated from text 1 and text 2, There were 21 words in common. On an average there is a 42 percent similarity in the usage of words on comparing the top 50 common words from text file 1 and 2

The common words from the top 50 words are: *states, government, united, congress, country, great, state, last, war, time, people, year, union, one, every, new, nations, peace, system, us, also*

2) **Comparing the results of top 50 bigrams with frequency from analysis 1 and analysis 2:**

There were 8 bigrams in common out of the top 50 bigrams generated with frequency for text 1 and text 2. On an average there is a 16 percent similarity in the usage of bigrams on comparing the top 50 bigrams with frequency from text file 1 and 2. They are:
united,states
fiscal,year
union,address
fellow,citizens
federal,government
last,year
past,year
american,people

3) **Comparing the results of top 50 bigrams with pmi from analysis 1 and analysis 2:**

There were 2 bigrams in common out of the top 50 bigrams generated with pmi for text 1 and text 2. On an average there is a 4 percent similarity in the usage of bigrams on comparing the top 50 bigrams with pmi from text file 1 and 2. They are:

project, gutenberg
thomas, Jefferson

Interpretations

a) After an analysis over the common words/bigrams, it can be noticed that they are related to the government and people. These are common words that would be spoken in any presidential address.
b) The language is concentrated on the *country, government, people, law, unity* in the first text file where as it is *people, world, security, economy, health, future* in the second text file. The most commonly used words from the second file proves that US presidential addresses concentrated on the security, economy and health aspects after the second world war.

Problems with the word or bigram lists found and possible solutions

After analyzing the words and bigrams generated by the above analysis carried out, the following were discovered as problems with the generated list

a) When a word had an apostrophe,the bigram was not properly split
For example: (('ca', "n't"), 0.00021064761751349073)
(***source-top 50 bigrams by frequencies analysis done for text file 2)

b) Words and bigrams were repeated in singular and plural forms.
For Example: ('states', 2725) and ('state', 1045), ('interests', 451) and ('interest', 444)
(***source-top 50 words by frequency analysis done for text file 1)

c) Auxiliary verbs in word list
For Example: ('shall', 694)
(***source-top 50 words by frequency analysis done for text file 1)

To get a better list of bigrams, the following can be incorporated as part of analysis
1.Stemming and lemmatization: to obtain the root words. This will reduce problems a) and b)
2.Add more user defined stop words to reduce problem c)

Comparison of top 50 bigrams by frequency and top 50 bigrams scored by Mutual Information

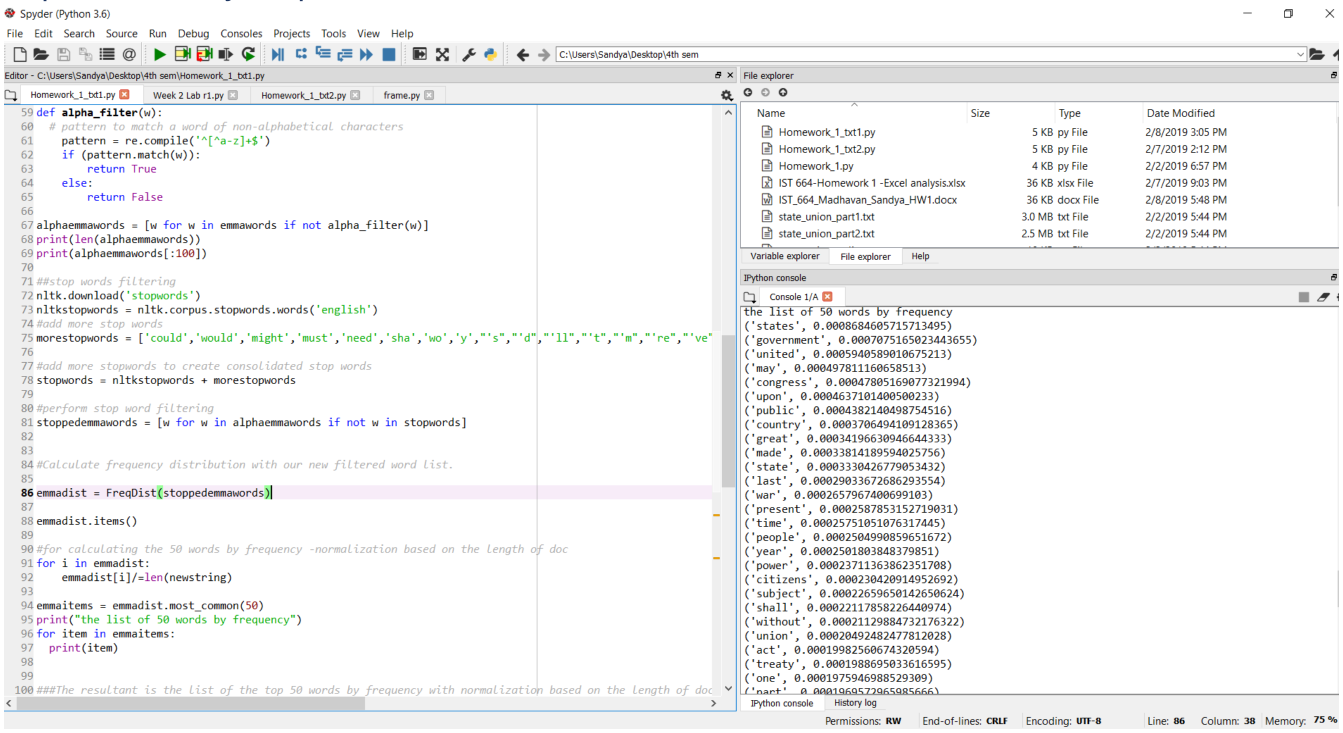
After manually analyzing the top 50 bigrams by frequency and top 50 bigrams scored by Mutual information in both analysis 1 and analysis 2, no commonality was found.

This can be attributed to the frequency that was set (i.e. 5) during bigram generated with mutual information scores.

Python Processing Screen shots

Analysis 1

Top 50 words by frequencies



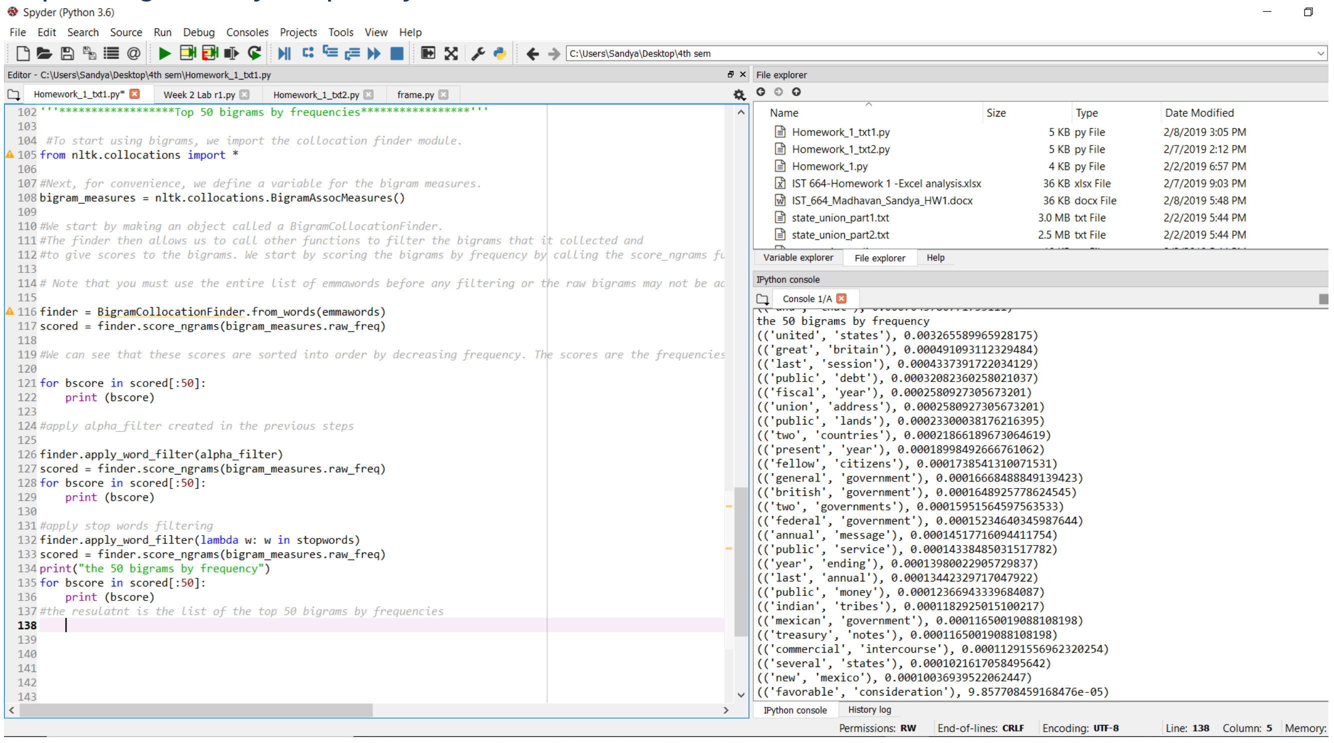
```python
def alpha_filter(w):
  # pattern to match a word of non-alphabetical characters
    pattern = re.compile('^[^a-z]+$')
    if (pattern.match(w)):
        return True
    else:
        return False

alphaemmawords = [w for w in emmawords if not alpha_filter(w)]
print(len(alphaemmawords))
print(alphaemmawords[:100])

##stop words filtering
nltk.download('stopwords')
nltkstopwords = nltk.corpus.stopwords.words('english')
#add more stop words
morestopwords = ['could','would','might','must','need','sha','wo','y',"'s","'d","'ll","'t","'m","'re","'ve"

#add more stopwords to create consolidated stop words
stopwords = nltkstopwords + morestopwords

#perform stop word filtering
stoppedemmawords = [w for w in alphaemmawords if not w in stopwords]


#Calculate frequency distribution with our new filtered word list.

emmadist = FreqDist(stoppedemmawords)

emmadist.items()

#for calculating the 50 words by frequency -normalization based on the length of doc
for i in emmadist:
    emmadist[i]/=len(newstring)

emmaitems = emmadist.most_common(50)
print("the list of 50 words by frequency")
for item in emmaitems:
  print(item)


###The resultant is the list of the top 50 words by frequency with normalization based on the length of doc
```

```
Name                                   Size  Type       Date Modified
Homework_1_txt1.py                     5 KB  py File    2/8/2019 3:05 PM
Homework_1_txt2.py                     5 KB  py File    2/7/2019 2:12 PM
Homework_1.py                          4 KB  py File    2/2/2019 6:57 PM
IST 664-Homework 1 -Excel analysis.xlsx 36 KB xlsx File 2/7/2019 9:03 PM
IST_664_Madhavan_Sandya_HW1.docx       36 KB docx File  2/8/2019 5:48 PM
state_union_part1.txt                  3.0 MB txt File  2/2/2019 5:44 PM
state_union_part2.txt                  2.5 MB txt File  2/2/2019 5:44 PM
```

```
the list of 50 words by frequency
('states', 0.0008684605715713495)
('government', 0.0007075165023443655)
('united', 0.0005940589010675213)
('may', 0.000497811160658513)
('congress', 0.0004780516907732194)
('upon', 0.0004637101400500233)
('public', 0.0004382140498754516)
('country', 0.0003706494109128365)
('great', 0.0003419663094664433)
('made', 0.0003381418959402575)
('state', 0.0003330426779053432)
('last', 0.0002903367268629355)
('war', 0.0002657967400699103)
('present', 0.0002587853152719031)
('time', 0.0002575105107631744)
('people', 0.0002504990859651672)
('year', 0.0002501803848379851)
('power', 0.0002371136386235170)
('citizens', 0.0002304209149526692)
('subject', 0.0002265965014265062)
('shall', 0.0002211785822644097)
('without', 0.0002112988473217632)
('union', 0.0002049248247781202)
('act', 0.0001998256067432059)
('treaty', 0.0001988695033616595)
('one', 0.0001975946988529309)
('part', 0.0001969572965985666)
```

Permissions: RW End-of-lines: CRLF Encoding: UTF-8 Line: 86 Column: 38 Memory: 75 %

Top 50 bigrams by frequency

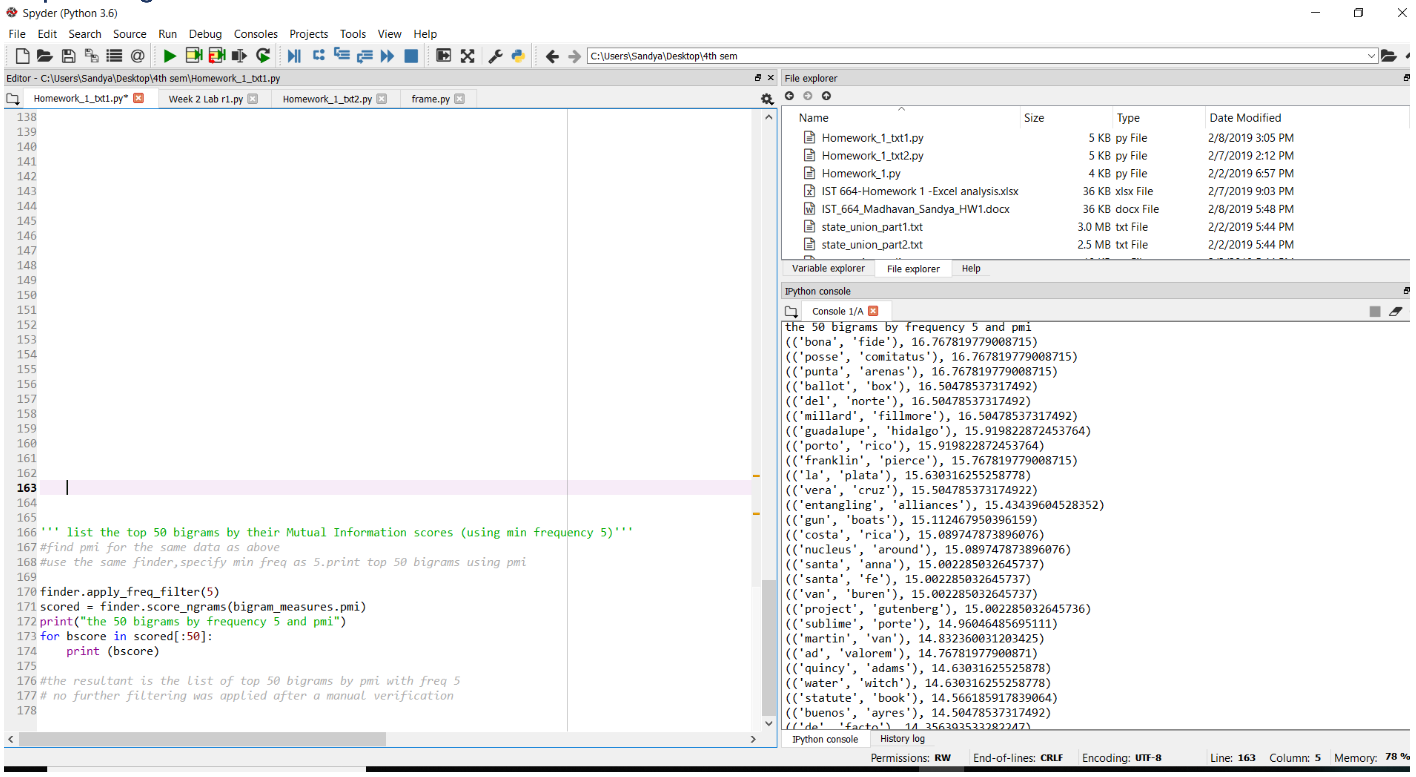


```python
'''*****************Top 50 bigrams by frequencies*****************'''

 #To start using bigrams, we import the collocation finder module.
from nltk.collocations import *

#Next, for convenience, we define a variable for the bigram measures.
bigram_measures = nltk.collocations.BigramAssocMeasures()

#We start by making an object called a BigramCollocationFinder.
#The finder then allows us to call other functions to filter the bigrams that it collected and
#to give scores to the bigrams. We start by scoring the bigrams by frequency by calling the score_ngrams fu

# Note that you must use the entire list of emmawords before any filtering or the raw bigrams may not be ac

finder = BigramCollocationFinder.from_words(emmawords)
scored = finder.score_ngrams(bigram_measures.raw_freq)

#We can see that these scores are sorted into order by decreasing frequency. The scores are the frequencies

for bscore in scored[:50]:
    print (bscore)

#apply alpha_filter created in the previous steps

finder.apply_word_filter(alpha_filter)
scored = finder.score_ngrams(bigram_measures.raw_freq)
for bscore in scored[:50]:
    print (bscore)

#apply stop words filtering
finder.apply_word_filter(lambda w: w in stopwords)
scored = finder.score_ngrams(bigram_measures.raw_freq)
print("the 50 bigrams by frequency")
for bscore in scored[:50]:
    print (bscore)
#the resulatnt is the list of the top 50 bigrams by frequencies
```

```
the 50 bigrams by frequency
(('united', 'states'), 0.003265589965928175)
(('great', 'britain'), 0.000491093112329484)
(('last', 'session'), 0.0004337391722034129)
(('public', 'debt'), 0.0003208236025802103)
(('fiscal', 'year'), 0.0002580927305673201)
(('union', 'address'), 0.0002580927305673201)
(('public', 'lands'), 0.0002330003817621639)
(('two', 'countries'), 0.0002186618967306461)
(('present', 'year'), 0.0001899849266761062)
(('fellow', 'citizens'), 0.0001738541310071531)
(('general', 'government'), 0.0001666848884913942)
(('british', 'government'), 0.000164892577862454)
(('two', 'governments'), 0.0001595156459756353)
(('federal', 'government'), 0.0001523464034598764)
(('annual', 'message'), 0.0001451771609441175)
(('public', 'service'), 0.0001433848503151778)
(('year', 'ending'), 0.0001398002290572983)
(('last', 'annual'), 0.0001344232971704792)
(('public', 'money'), 0.0001236694333968408)
(('indian', 'tribes'), 0.000118292501510021)
(('mexican', 'government'), 0.0001165001908810819)
(('treasury', 'notes'), 0.0001165001908810819)
(('commercial', 'intercourse'), 0.0001129155696232025)
(('several', 'states'), 0.0001021617058495642)
(('new', 'mexico'), 0.0001003693952206244)
(('favorable', 'consideration'), 9.857708459168476e-05)
```

Permissions: RW End-of-lines: CRLF Encoding: UTF-8 Line: 138 Column: 5 Memory:

Top 50 bigrams with PMI


Spyder (Python 3.6)
File Edit Search Source Run Debug Consoles Projects Tools View Help
C:\Users\Sandya\Desktop\4th sem
Editor - C:\Users\Sandya\Desktop\4th sem\Homework_1_txt1.py
Homework_1_txt1.py*
Week 2 Lab r1.py
Homework_1_txt2.py
frame.py
File explorer
Variable explorer
Help
IPython console
Console 1/A
History log

Name	Size	Type	Date Modified
Homework_1_txt1.py	5 KB	py File	2/8/2019 3:05 PM
Homework_1_txt2.py	5 KB	py File	2/7/2019 2:12 PM
Homework_1.py	4 KB	py File	2/2/2019 6:57 PM
IST 664-Homework 1 -Excel analysis.xlsx	36 KB	xlsx File	2/7/2019 9:03 PM
IST_664_Madhavan_Sandya_HW1.docx	36 KB	docx File	2/8/2019 5:48 PM
state_union_part1.txt	3.0 MB	txt File	2/2/2019 5:44 PM
state_union_part2.txt	2.5 MB	txt File	2/2/2019 5:44 PM

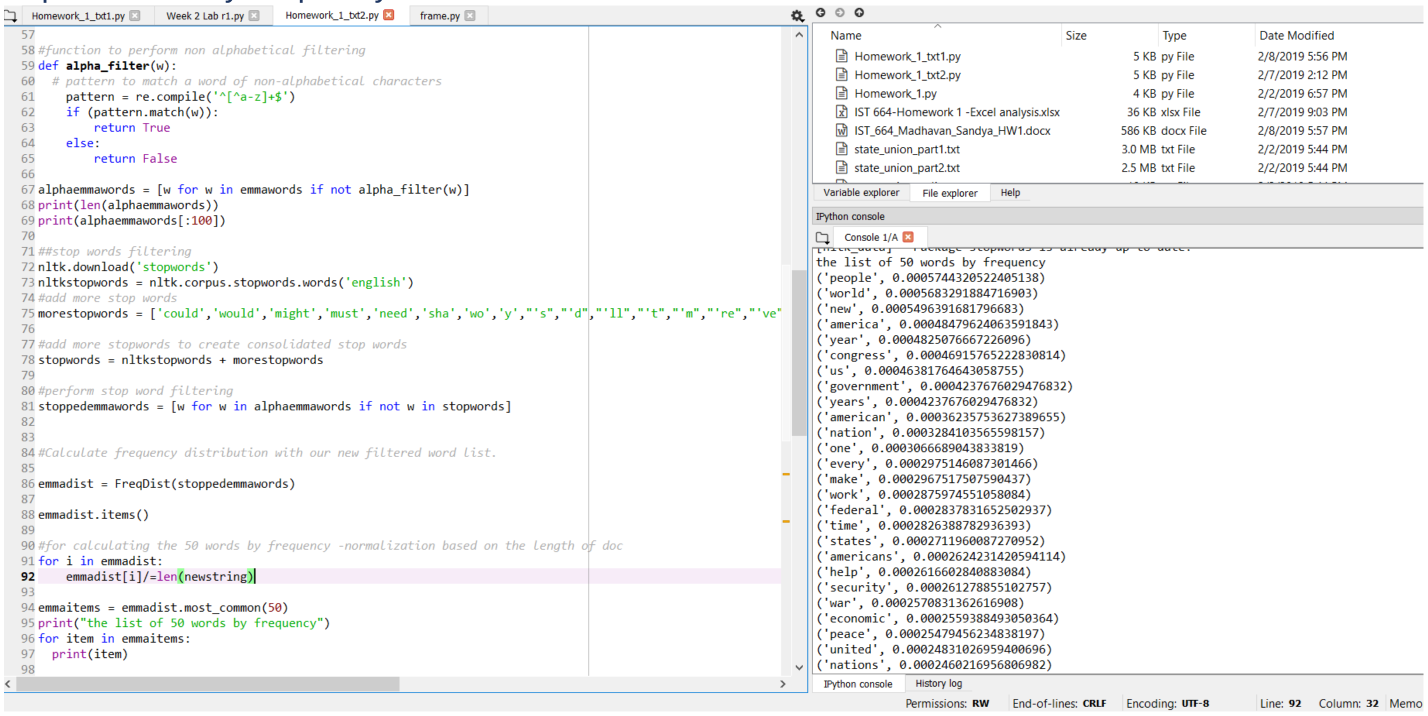
```python
''' list the top 50 bigrams by their Mutual Information scores (using min frequency 5)'''
#find pmi for the same data as above
#use the same finder,specify min freq as 5.print top 50 bigrams using pmi

finder.apply_freq_filter(5)
scored = finder.score_ngrams(bigram_measures.pmi)
print("the 50 bigrams by frequency 5 and pmi")
for bscore in scored[:50]:
    print (bscore)

#the resultant is the list of top 50 bigrams by pmi with freq 5
# no further filtering was applied after a manual verification
```

```
the 50 bigrams by frequency 5 and pmi
(('bona', 'fide'), 16.767819779008715)
(('posse', 'comitatus'), 16.767819779008715)
(('punta', 'arenas'), 16.767819779008715)
(('ballot', 'box'), 16.50478537317492)
(('del', 'norte'), 16.50478537317492)
(('millard', 'fillmore'), 16.50478537317492)
(('guadalupe', 'hidalgo'), 15.919822872453764)
(('porto', 'rico'), 15.919822872453764)
(('franklin', 'pierce'), 15.767819779008715)
(('la', 'plata'), 15.630316255258778)
(('vera', 'cruz'), 15.504785373174922)
(('entangling', 'alliances'), 15.43439604528352)
(('gun', 'boats'), 15.112467950396159)
(('costa', 'rica'), 15.089747873896076)
(('nucleus', 'around'), 15.089747873896076)
(('santa', 'anna'), 15.002285032645737)
(('santa', 'fe'), 15.002285032645737)
(('van', 'buren'), 15.002285032645737)
(('project', 'gutenberg'), 15.002285032645736)
(('sublime', 'porte'), 14.96046485695111)
(('martin', 'van'), 14.832360031203425)
(('ad', 'valorem'), 14.76781977900871)
(('quincy', 'adams'), 14.63031625525878)
(('water', 'witch'), 14.630316255258778)
(('statute', 'book'), 14.566185917839064)
(('buenos', 'ayres'), 14.50478537317492)
(('de', 'facto'), 14.356393533282247)
```

Permissions: RW End-of-lines: CRLF Encoding: UTF-8 Line: 163 Column: 5 Memory: 78 %

Analysis 2

Top 50 words by frequency



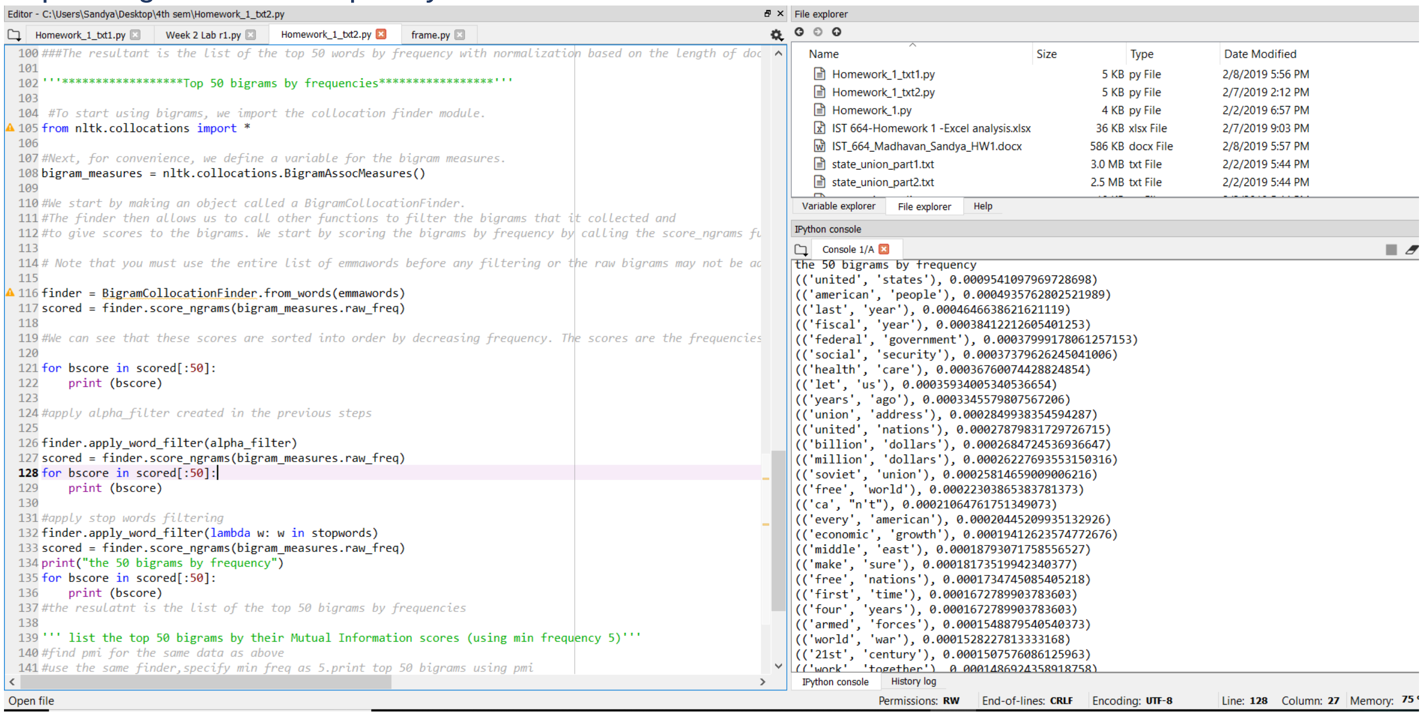
Name	Size	Type	Date Modified
Homework_1_txt1.py	5 KB	py File	2/8/2019 5:56 PM
Homework_1_txt2.py	5 KB	py File	2/7/2019 2:12 PM
Homework_1.py	4 KB	py File	2/2/2019 6:57 PM
IST 664-Homework 1 -Excel analysis.xlsx	36 KB	xlsx File	2/7/2019 9:03 PM
IST_664_Madhavan_Sandya_HW1.docx	586 KB	docx File	2/8/2019 5:57 PM
state_union_part1.txt	3.0 MB	txt File	2/2/2019 5:44 PM
state_union_part2.txt	2.5 MB	txt File	2/2/2019 5:44 PM

```python

#function to perform non alphabetical filtering
def alpha_filter(w):
  # pattern to match a word of non-alphabetical characters
    pattern = re.compile('^[^a-z]+$')
    if (pattern.match(w)):
        return True
    else:
        return False

alphaemmawords = [w for w in emmawords if not alpha_filter(w)]
print(len(alphaemmawords))
print(alphaemmawords[:100])

##stop words filtering
nltk.download('stopwords')
nltkstopwords = nltk.corpus.stopwords.words('english')
#add more stop words
morestopwords = ['could','would','might','must','need','sha','wo','y',"'s","'d","'ll","'t","'m","'re","'ve"

#add more stopwords to create consolidated stop words
stopwords = nltkstopwords + morestopwords

#perform stop word filtering
stoppedemmawords = [w for w in alphaemmawords if not w in stopwords]


#Calculate frequency distribution with our new filtered word list.

emmadist = FreqDist(stoppedemmawords)

emmadist.items()

#for calculating the 50 words by frequency -normalization based on the length of doc
for i in emmadist:
    emmadist[i]/=len(newstring)

emmaitems = emmadist.most_common(50)
print("the list of 50 words by frequency")
for item in emmaitems:
  print(item)

```

```
[nltk_data]   Package stopwords is already up to date!
the list of 50 words by frequency
('people', 0.0005744320522405138)
('world', 0.0005683291884716903)
('new', 0.0005496391681796683)
('america', 0.00048479624063591843)
('year', 0.0004825076667226096)
('congress', 0.00046915765222830814)
('us', 0.00046381764643058755)
('government', 0.00042376760294768 32)
('years', 0.00042376760294768 32)
('american', 0.00036235753627389655)
('nation', 0.0003284103565598157)
('one', 0.0003066689043833819)
('every', 0.0002975146087301466)
('make', 0.0002967517507590437)
('work', 0.0002875974551058084)
('federal', 0.0002837831652502937)
('time', 0.0002826388782936393)
('states', 0.0002711960087270952)
('americans', 0.0002624231420594114)
('help', 0.0002616602840883084)
('security', 0.0002612788551027 57)
('war', 0.0002570831362616908)
('economic', 0.0002559388493050364)
('peace', 0.0002547945623483 8197)
('united', 0.0002483102695940 0696)
('nations', 0.0002460216956806982)
```

Permissions: RW End-of-lines: CRLF Encoding: UTF-8 Line: 92 Column: 32 Memo

Top 50 bigrams with frequency



Editor - C:\Users\Sandya\Desktop\4th sem\Homework_1_txt2.py

Homework_1_txt1.py | Week 2 Lab r1.py | Homework_1_txt2.py | frame.py

```python
###The resultant is the list of the top 50 words by frequency with normalization based on the length of doc

'''*****************Top 50 bigrams by frequencies*****************'''

 #To start using bigrams, we import the collocation finder module.
from nltk.collocations import *

#Next, for convenience, we define a variable for the bigram measures.
bigram_measures = nltk.collocations.BigramAssocMeasures()

#We start by making an object called a BigramCollocationFinder.
#The finder then allows us to call other functions to filter the bigrams that it collected and
#to give scores to the bigrams. We start by scoring the bigrams by frequency by calling the score_ngrams fu

# Note that you must use the entire list of emmawords before any filtering or the raw bigrams may not be ac

finder = BigramCollocationFinder.from_words(emmawords)
scored = finder.score_ngrams(bigram_measures.raw_freq)

#We can see that these scores are sorted into order by decreasing frequency. The scores are the frequencies

for bscore in scored[:50]:
    print (bscore)

#apply alpha_filter created in the previous steps

finder.apply_word_filter(alpha_filter)
scored = finder.score_ngrams(bigram_measures.raw_freq)
for bscore in scored[:50]:
    print (bscore)

#apply stop words filtering
finder.apply_word_filter(lambda w: w in stopwords)
scored = finder.score_ngrams(bigram_measures.raw_freq)
print("the 50 bigrams by frequency")
for bscore in scored[:50]:
    print (bscore)
#the resulatnt is the list of the top 50 bigrams by frequencies

''' list the top 50 bigrams by their Mutual Information scores (using min frequency 5)'''
#find pmi for the same data as above
#use the same finder,specify min freq as 5.print top 50 bigrams using pmi
```

File explorer

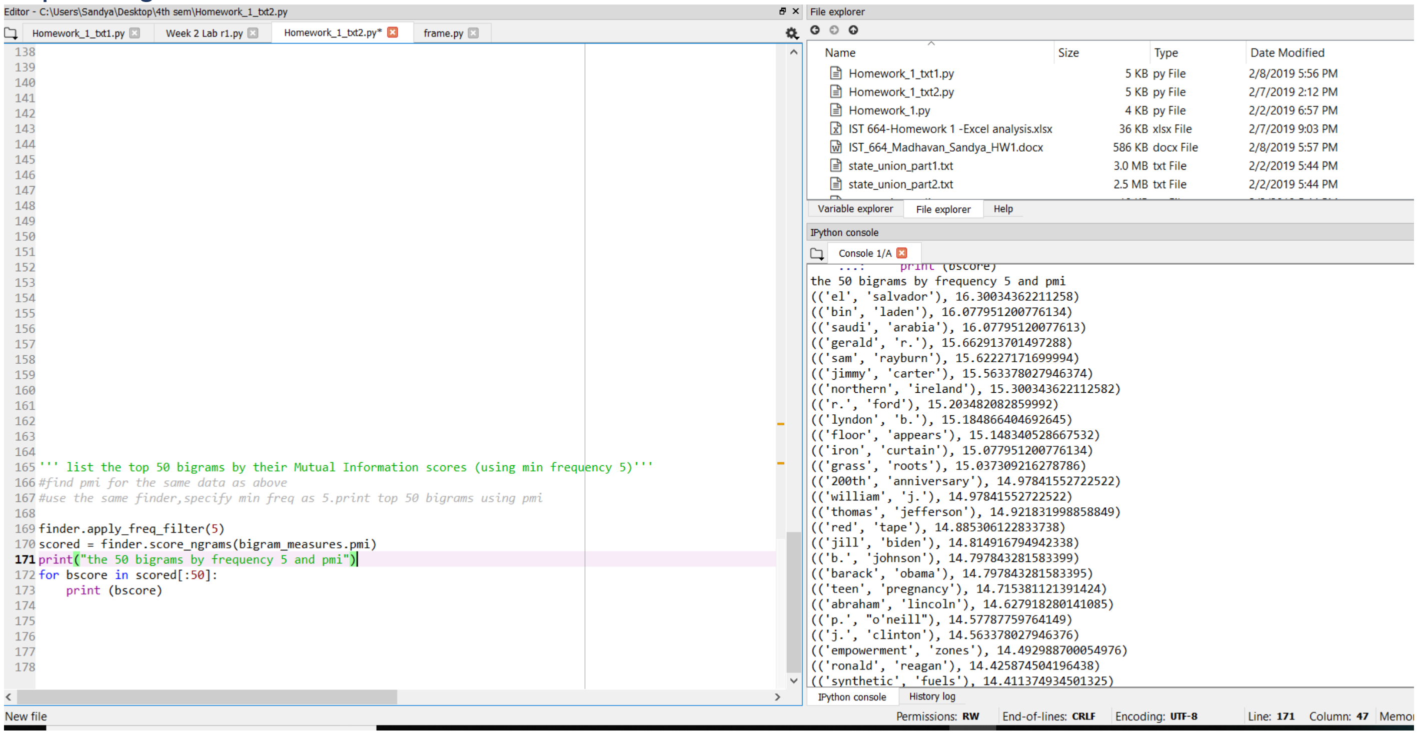
Name	Size	Type	Date Modified
Homework_1_txt1.py	5 KB	py File	2/8/2019 5:56 PM
Homework_1_txt2.py	5 KB	py File	2/7/2019 2:12 PM
Homework_1.py	4 KB	py File	2/2/2019 6:57 PM
IST 664-Homework 1 -Excel analysis.xlsx	36 KB	xlsx File	2/7/2019 9:03 PM
IST_664_Madhavan_Sandya_HW1.docx	586 KB	docx File	2/8/2019 5:57 PM
state_union_part1.txt	3.0 MB	txt File	2/2/2019 5:44 PM
state_union_part2.txt	2.5 MB	txt File	2/2/2019 5:44 PM

IPython console — Console 1/A

```
the 50 bigrams by frequency
(('united', 'states'), 0.0009541097969728698)
(('american', 'people'), 0.0004935762802521989)
(('last', 'year'), 0.0004646638621621119)
(('fiscal', 'year'), 0.00038412212605401253)
(('federal', 'government'), 0.00037999178061257153)
(('social', 'security'), 0.00037379626245041006)
(('health', 'care'), 0.00036760074428824854)
(('let', 'us'), 0.00035934005340536654)
(('years', 'ago'), 0.0003345579807567206)
(('union', 'address'), 0.0002849938354594287)
(('united', 'nations'), 0.00027879831729726715)
(('billion', 'dollars'), 0.0002684724536936647)
(('million', 'dollars'), 0.00026227693553150316)
(('soviet', 'union'), 0.00025814659009006216)
(('free', 'world'), 0.00022303865383781373)
(('ca', "n't"), 0.00021064761751349073)
(('every', 'american'), 0.00020445209935132926)
(('economic', 'growth'), 0.00019412623574772676)
(('middle', 'east'), 0.00018793071758556527)
(('make', 'sure'), 0.00018173519942340377)
(('free', 'nations'), 0.0001734745085405218)
(('first', 'time'), 0.0001672789903783603)
(('four', 'years'), 0.0001672789903783603)
(('armed', 'forces'), 0.0001548879540540373)
(('world', 'war'), 0.0001528227813333168)
(('21st', 'century'), 0.0001507576086125963)
(('work', 'together'), 0.0001486924358918758)
```

Permissions: RW End-of-lines: CRLF Encoding: UTF-8 Line: 128 Column: 27 Memory: 75

Top 50 bigrams with PMI



Editor - C:\Users\Sandya\Desktop\4th sem\Homework_1_txt2.py

Homework_1_txt1.py | Week 2 Lab r1.py | Homework_1_txt2.py* | frame.py

```python
''' list the top 50 bigrams by their Mutual Information scores (using min frequency 5)'''
#find pmi for the same data as above
#use the same finder,specify min freq as 5.print top 50 bigrams using pmi

finder.apply_freq_filter(5)
scored = finder.score_ngrams(bigram_measures.pmi)
print("the 50 bigrams by frequency 5 and pmi")
for bscore in scored[:50]:
    print (bscore)
```

File explorer

Name	Size	Type	Date Modified
Homework_1_txt1.py	5 KB	py File	2/8/2019 5:56 PM
Homework_1_txt2.py	5 KB	py File	2/7/2019 2:12 PM
Homework_1.py	4 KB	py File	2/2/2019 6:57 PM
IST 664-Homework 1 -Excel analysis.xlsx	36 KB	xlsx File	2/7/2019 9:03 PM
IST_664_Madhavan_Sandya_HW1.docx	586 KB	docx File	2/8/2019 5:57 PM
state_union_part1.txt	3.0 MB	txt File	2/2/2019 5:44 PM
state_union_part2.txt	2.5 MB	txt File	2/2/2019 5:44 PM

IPython console — Console 1/A

```
    ...:     print (bscore)
the 50 bigrams by frequency 5 and pmi
(('el', 'salvador'), 16.30034362211258)
(('bin', 'laden'), 16.077951200776134)
(('saudi', 'arabia'), 16.07795120077613)
(('gerald', 'r.'), 15.662913701497288)
(('sam', 'rayburn'), 15.62227171699994)
(('jimmy', 'carter'), 15.563378027946374)
(('northern', 'ireland'), 15.300343622112582)
(('r.', 'ford'), 15.203482082859992)
(('lyndon', 'b.'), 15.184866404692645)
(('floor', 'appears'), 15.148340528667532)
(('iron', 'curtain'), 15.077951200776134)
(('grass', 'roots'), 15.037309216278786)
(('200th', 'anniversary'), 14.97841552722522)
(('william', 'j.'), 14.97841552722522)
(('thomas', 'jefferson'), 14.921831998858849)
(('red', 'tape'), 14.885306122833738)
(('jill', 'biden'), 14.814916794942338)
(('b.', 'johnson'), 14.797843281583399)
(('barack', 'obama'), 14.797843281583395)
(('teen', 'pregnancy'), 14.715381121391424)
(('abraham', 'lincoln'), 14.627918280141085)
(('p.', "o'neill"), 14.57787759764149)
(('j.', 'clinton'), 14.563378027946376)
(('empowerment', 'zones'), 14.492988700054976)
(('ronald', 'reagan'), 14.425874504196438)
(('synthetic', 'fuels'), 14.411374934501325)
```

Permissions: RW End-of-lines: CRLF Encoding: UTF-8 Line: 171 Column: 47 Memo