CSE 250B: Homework 2

Xiyun Liu A53099348

October 27, 2016

0.1 Description of the idea for prototype selection

1. Remove stop words.
2. Sort the vocabulary by their occurrences(probability) among all the documents.
3. Compensating for burstiness. Using $log(1+f)$ instead of the number of occurrences f of a word to calculate P_{jw}.

A subset vocabulary of size M is the top M frequent words after removing stop words.

0.2 Concise and unambiguous pseudocode

Algorithm 1 Prototype selection

Input: *trainData*,*trainLabel*,*vocabulary*, number M
Output: *subsetVocabulary* of size M, pjw, pi;

procedure CreateSebset(trainData, trainLabel, vocabulary, M)
    $classWordCount = np.ones((20, numVoc))$
    for $i \in 0...trainData.size()$ **do**
        $docId, wordId, wordCount \leftarrow trainData[i]$
        $docLabel \leftarrow trainLabel[docId - 1]$
        $classWordCount[docLabel - 1][wordId - 1] += wordCount$
    $numTotalWordCount \leftarrow$ *Sum each column of eachClassWordCount*
    $numTotalWordCount.sortInDescending$
    $topFrequentWords \leftarrow$ *words mapped from numTotalWordCount*
    $subsetVocabulary \leftarrow topFrequentWords[0 : M]$
    $idxNeedRemove \leftarrow subsetVocabularyIndex + stopWordsIdx$
    $filteredClassWordCount \leftarrow classWordCount.delete(idxNeedRemove, axis = 1)$
    $filteredClassWordCountLog \leftarrow log(1 + filteredClassWordCount)$
    $numWordInClass \leftarrow$ *sum each row of filteredClassWordCountLog*
    $pjw \leftarrow$ *each raw of* $filteredClassWordCountLog/numWordInClass$
    $pi \leftarrow [trainLabel.count(label) * 1.0/numDocforlabelinrange(1, 21)]$
return $subsetVocabulary, pjw, pi$

0.3 Experimental results

Compare the performance between random selection and my method.
The error rate:

M	random Selection	Prototype Selection
5000	54.370%	25.076%
10000	43.177%	22.038%
20000	32.931%	20.266%

Calculate the error bars and confidence interval for correctness percentage at M = 5000,10000,20000 using Random Selection.

M	5000	10000	20000
Number of Samples	10	10	10
Mean	54.373%	43.177%	32.931%
Standard Deviation	1.297%	1.105%	0.754%
Error Bars	53.076% to 55.670%	42.072% to 44.282%	32.177% to 33.685%
Confidence Interval	51.831% to 56.915%	41.011% to 45.343%	31.453% to 34.408%

The error bars are basically calculated by mean and standard deviation. Using mean μ and standard deviation σ, we can state that, based on the experiments, the correctness percentage at M = 1000 is 88.5526% $\pm$ 0.3917%.

$$Error\ Bars = \mu \pm \sigma$$

For the confidence interval at the 95% confidence level:

$$Lower\ limit = \mu - Z_{.95} * \sigma$$
$$Upper\ limit = \mu + Z_{.95} * \sigma$$

where Z is the 0.95 critical value of the standard normal distribution which can be found in the table of the standard normal distribution. $Z_{.95} = 1.96$

0.4 Inspection of models

The way I selected representatives for each class is that first generate the subset with M = 1000, then calculate the probability of words appears in each class *probWordInClass*. For each word, find the class j with maximum *probWordInClass*. Then I put this word into a representative of class j.
After the representatives are generated for each class, I also filter those words that have no specific meaning, such as 'make','we', etc. The select make sense to me. They indicates the meaning of the class.

class	random Selection	Prototype Selection
1	alt.atheism	'thing', 'little', 'course', 'example', 'wrote', 'seem', 'makes', 'evidence', 'perhaps', 'agree', 'claim', 'religion', 'argument', 'sense', 'exist', 'position', 'therefore', 'statement'
2	comp.graphics	'etc', 'software', 'looking', 'ftp', 'image', 'graphics', 'full', 'stuff', 'color', 'anybody', 'format', 'various', 'reference', 'quality', 'images', 'site', 'useful'
3	comp.os.ms-windows.misc	'windows', 'files', 'dos', 'ms', 'driver', 'mouse', 'printer', 'latest', 'microsoft', 'em', 'fonts', 'bytes'
4	comp.sys.ibm.pc.hardware	'drive', 'hard', 'card', 'scsi', 'mb', 'pc', 'disk', 'local', 'speed', 'memory', 'machine', 'ibm', 'fast', 'uses', 'os', 'board', 'fine', 'advance', 'mode', 'bus'
5	comp.sys.mac.hardware	'problem', 'mac', 'apple', 'video', 'hardware', 'sound', 'built', 'monitor', 'cd', 'mhz', 'ram', 'plus', 'machines', 'se', 'extra', 'cable', 'modem', 'internal', 'option', 'cpu'
6	comp.windows.x	'get', 'work', 'file', 'using', 'thanks', 'program', 'help', 'read', 'available', 'name', 'set', 'try', 'run', 'line', 'list', 'support', 'email', 'window', 'call', 'send'
7	misc.forsale	'new', 'please', 'interested', 'price', 'original', 'sell', 'offer', 'sale', 'cover', 'included','asking', 'condition', 'excellent', 'shipping', 'trade', 'picture'
8	rec.autos	'car', 'buy', 'anyway', 'deal', 'model', 'cars', 'road','performance', 'engine','sounds', 'bought', 'btw', 'driving', 'oil'
9	rec.motorcycles	'com', 'dod', 'bike', 'turn', 'front', 'ed', 'disclaimer', 'street', 'chris', 'advice', 'ride', 'hey', 'cb', 'fit'
10	rec.sport.baseball	'good', 'year', 'last', 'runs', 'hit', 'gets', 'lost', 'early','baseball', 'average', 'base', 'field','fan'
11	rec.sport.hockey	'best', 'great', 'second', 'game', 'team', 'next', 'bad', 'win', 'play', 'st', 'mark', 'games', 'april', 'season', 'mike','hockey', 'washington', 'players', 'points', 'vs','league'

12	sci.crypt	'system','need', 'mail', 'information', 'government', 'cs', 'number', 'without', 'bit', 'key', 'data', 'law', 'part', 'probably', 'every', 'computer', 'public', 'internet', 'general', 'systems'
13	sci.electronics	'power','current', 'box', 'low', 'usually', 'copy', 'company', 'radio', 'ground', 'tv', 'higher', 'range', 'supply', 'wire', 'cheap', 'hot'
14	sci.med	'edu', 'problems', 'research', 'getting', 'science', 'cause', 'major', 'per', 'common', 'week', 'health', 'test', 'effect', 'friend','water', 'study',
15	sci.space	'space', 'high', 'nasa', 'gov', 'large', 'small', 'cost', 'earth', 'level', 'black','light','design', 'air', 'bitnet', 'market', 'development', 'sci', 'project', 'launch', 'organization', 'commercial', 'institute', 'engineering', 'station
16	soc.religion.christian	'god', 'say', 'really', 'believe', 'find', 'true', 'jesus', 'life', 'though', 'john', 'mean','man', 'wrong', 'person'
17	talk.politics.guns	'news', 'keep', 'gun', 'control', 'national', 'bill', 'issue', 'police', 'self', 'laws', 'fire', 'rate', 'crime', 'usa', 'weapons', 'due', 'act', 'defense', 'worth', 'action'
18	talk.politics.mideast	'people', 'like', 'know', 'well', 'even', 'us', 'way', 'first', 'many', 'said', 'back', 'still', 'take', 'years', 'might', 'another', 'world'
19	talk.politics.misc	'writes', 'article', 'think', 'make', 'want', 'something', 'going', 'sure', 'made', 'look', 'case', 'actually', 'free', 'yes', 'mr', 'done', 'less', 'american', 'already', 'money','kind', 'president', 'opinions', 'states', 'show', 'important', 'working', 'care', 'men', 'consider'
20	talk.religion.misc	'jim', 'robert', 'knowledge', 'reply', 'brian', 'deleted', 'frank', 'objective', 'values', 'context', 'specifically', 'kent', 'stephen', 'lee', 'sandvik', 'tony', 'creation'

0.5 Critical evaluation

By comparing the error rate with random selection, my method improved the performance heavily. Therefore, I think my method is a clear benefit.
There is still further scope for improvement. First, in my method, I didn't implement inverse document frequency to weight different words. In the next step, I would like to try implement it and do several experiment to compare the performance.
Moreover, I have the intuition that those words whose standard deviation of probability of occurrence in each class should be considered to be added into subset, since they can help separate the class better. However, I have tried implementing it but the performance is not very good. In the next step, I would like to dig into this idea and hope it would work.

1 Appendix

M = 5000: [0.5530979347, 0.559360426382, 0.520319786809, 0.549500333111, 0.518854097268, 0.543904063957, 0.552831445703, 0.550566289141, 0.547501665556, 0.541372418388]

M = 10000: [0.44836775483011326, 0.4407728181212525, 0.4361092604930047, 0.44183877415056627, 0.43584277148567624, 0.4251832111925383, 0.40826115922718187, 0.4325116588940706, 0.427981345769487 0.42091938707528315]

M = 20000: 0.318321119254 [0.341505662891, 0.337375083278, 0.328181212525, 0.32871419054, 0.338840772818, 0.330446369087, 0.322318454364, 0.319920053298, 0.327514990007]