COMP3608 Artificial Intelligence (Adv): Email Classification

Dongjie Zhang: 310241316
Tianyu Pu: 310182212

May 28, 2012

1 Introduction

1.1 Aims

To classify emails into one of two classes: spam and non-spam by implementing the Naive Bayes machine learning algorithm using relative frequencies of individual words appearing in the texts. The performance of the classifier is evaluated through ten-fold cross validation and then compared with Weka's implementation of Naive Bayes and several other classification algorithms[1].

1.2 Why is this study important?

The amount of digital information is growing at an increasing rate and it becomes increasingly difficult to search for information. Therefore, it is important to accurately classify the huge number of documents. To classify texts by creating rules specific to that field or text type requires expert knowledge would be costly and impractical. In comparison, the method explored here is completely general and can be used to classify texts from any area, not just emails into spam or non-spam (although fine-tuning of parameters will be necessary to maximise accuracy).

[1] http://cs.waikato.ac.nz/ml/weka/

2 Data Preprocessing

2.1 Procedure

In order to be able to read the files, the Python `gzip` module was used to decompress the text format. The *LingSpam* emails were split into two corpora: one containing the text of the subject and the other containing the text of the body. Individual words were extracted from each corpus, stopwords and punctuation removed and document frequency of the words calculated. In each corpus the top 200 words by document frequency were selected as features and then each feature was assigned a weighting using the *term frequency-inverse document frequency* formula explained in [2]. These tf-idf scores are normalised to between 0 and 1 using the cosine normalisation also described in [3] and saved as two CSV files, one for each corpus.

2.2 Preprocessing results

The numer of words before and after stopword removal [4], are shown in the table below:

Corpus	Words before stopword removal	Words after stopword removal
Subject	1830	938
Body	153851	19929

The top 100 words for each corpus, based on their DF (document frequency, or the number of documents in the collection that the word appeared in) score, are shown below.

[2]Sebastiani: http://doi.acm.org/10.1145/505282.505283

[3]Sebastiani: http://doi.acm.org/10.1145/505282.505283

[4]Stopword list courtesy of http://jmlr.csail.mit.edu/papers/volume5/lewis04a/a11-smart-stop-list/english.stop

Corpus: Subject		Corpus: Body	
Word	Document Frequency	Word	Document Frequency
sum	30	information	205
summary	26	language	192
language	21	mail	183
free	20	university	179
disc	19	time	178
english	19	list	171
query	18	address	165
linguistics	15	english	159
sex	13	http	156
words	12	linguistics	156
opposites	12	people	146
book	10	send	146
method	9	free	144
mail	9	make	140
comparative	9	email	133
correction	8	number	128
ob	8	work	128
program	7	www	122
qs	7	languages	119
million	7	find	118
syntax	7	fax	116
email	7	order	108
announcement	7	call	103
armey	6	form	101
call	6	research	100
slip	6	linguistic	99
st	6	state	99
internet	6	world	98
japanese	6	years	98
part	6	subject	98
german	6	contact	97
dick	6	de	96
speaker	6	money	94
workshop	6	word	91
money	6	message	91

Corpus: Subject		Corpus: Body	
Word	Document Frequency	Word	Document Frequency
business	6	ll	89
lang	6	receive	88
chinese	5	phone	88
resources	5	check	88
word	5	good	87
list	5	interested	86
languages	5	day	86
native	5	year	86
grammar	5	working	85
research	5	case	85
spanish	5	include	85
needed	5	ve	84
nglish	5	based	84
linguist	5	note	83
software	5	home	83
hey	5	part	83
time	5	made	83
omparative	4	mailing	81
address	4	including	81
great	4	type	80
fwd	4	web	79
intuitions	4	give	79
information	4	place	79
banning	4	program	79
american	4	date	78
phonetics	4	line	78
request	4	special	78
web	4	days	77
secrets	4	internet	76
conference	4	back	76
systems	4	service	75
read	4	american	75
programs	4	full	74
summer	4	system	74
www	4	business	74

Corpus: Subject		Corpus: Body	
Word	Document Frequency	Word	Document Frequency
obs	4	ac	73
pig	4	today	73
synthetic	3	interest	72
teaching	3	remove	72
dutch	3	questions	72
fall	3	john	71
school	3	found	70
linguists	3	related	70
video	3	site	69
french	3	linguist	69
change	3	usa	69
credit	3	read	68
adjectives	3	point	68
addresses	3	text	68
names	3	ago	67
live	3	week	67
counting	3	book	67
mac	3	cost	66
youthese	3	dear	66
policy	3	making	66
decimal	3	simply	65
dialect	3	question	65
books	3	offer	63
profit	3	received	63
millions	3	general	63
care	3	data	62
misc	3	ca	62
reference	3	important	62
lists	3	summary	61
released	3	long	61

The DF scores are lower for the subject corpus by a factor of about 15-20, which reflects the proportionately smaller size of the subject corpus: there are fewer words, so the document frequencies are much lower. The lists are somewhat similar, with 26 terms in common.

The counts for the subject corpus seem quite low: most words have a document frequency of fewer than 10.

3 Results and Discussion

3.1 Findings

Corpus: Subject		Corpus: Body	
Classifier	Accuracy[%]	Classifier	Accuracy[%]
ZeroR	67	ZeroR	67
OneR	70	OneR	82
1-NN	79	1-NN	79
3-NN	74	3-NN	88
NB	68	NB	92
DT	67	DT	92
MLP	78	MLP	96
SVM	81	SVM	96
MyNB	70	MyNB	93

3.2 Discussion

To evaluate the accuracy of the classifiers, 10-fold stratified cross-validation is used and then compared to the *Weka* implementation of Naive Bayes and several other classification algorithms also with 10-fold stratified cross-validation. The results are tabulated above. The stratified cross-validation is performed on the feature values, which are the normalised weighted tf-idf scores of each top 200 word by document frequency. To simplify computation, these feature values are not recalculated for each fold.

As a result, the feature values of the training data will be influenced by the validation data. This difference would not be significant, as the size of the validation data is 9 times smaller, hence its contribution to any given word's document frequency would be 9 times smaller as well. The effect of this is the classifier might perform slightly better than the case where the feature scores are calculated from only the training data. The above comparisons are still valid, as the *Weka* data input also consists of the same feature scores, therefore having a similar accuracy bias.

3.2.1 Comparisons

The accuracy of Naive Bayes and Weka's Naive Bayes are very similar for both the body and subject corpora. This is expected, as both implementations are performing training and validation on the same transformed data. There is a small difference of about 1% accuracy between Weka's and our implementation of Naive Bayes. This variation is because Weka performed its own stratified cross-validation by generating its own randomised stratified subsets.

The prediction accuracy on the subject and body corpora vary significantly. The body corpus has much more text than the subject corpus, which effectively means a larger sample size. A larger sample size results in a smaller variance and a smaller mean squared error of the estimate. This is reflected in the empirical result when Naive Bayes is run on body corpus resulting in an accuracy of 94% compared to 70% on the subject corpus. In addition, all other classification algorithms performed as well or better on the body corpus. As the amount of text increases we note the simpler algorithms such as 1-Rule and k-NN are not as accurate as statistical algorithms such as Naive Bayes and decision tree learning.

For the subject corpus, despite having lower accuracies across all the classifiers (not counting Zero Rule), the statistical algorithms, Naive Bayes and DT scored particularly low; not much better than the Zero Rule baseline, because of the effectively reduced sample size and hence increased error variance. On the other hand, the geometric classifiers such as Support Vector Machine and 1-Nearest Neigbour algorithms performed much better, achieving accuracies of around 80%, suggesting the classes are highly linearly separable in the 200-dimension feature space.

4 Extension

Stemming and Categorical Proportional Difference (CPD) are implemented as the extension.

4.1 Stemming

Stemming is the process of reducing words to their base form. The simple solution, removing suffixes using a suffix list will yeild results with a low accuracy. For instance, if *wand* and *wander* occur in the document, the suffix *er* will get erronously stemmed, as it is not a suffix, but part of the stem. Conversely in the case of *probe* and *probate*, despite having the same stem, the words have quite different meanings and should not be stemmed. In light of this, the implementation of stemming used is the Porter Stemming Algorithm[5]. General lexicographical rules are devised to progressively remove suffixes from words.
There are few cases the algorith does not cover such as *indices* and *index*, but these cases are sufficiently rare in real vocabularies to not warrant the specific rules. The results from stratified 10-fold cross validation are shown below and are compared with non-stemmed Naive Bayes from the previous section.

Corpus	Accuracy[%]	Accuracy after stemming[%]
Subject	70	80
Body	93	94

There is a significant increase in accuracy for the subject corpora and a marginal increase in the body corpus (and a corresponding accuracy increase in Weka's implementation of Naive Bayes). Stemming removes correlation between the factors by combining them. For example, in this report, the occurence of the word *stem* is highly correlated with the occurence of the word *stemming*, if one word appears in a document, it is likely the other word would appear also. Such a factor would act as a confounding factor and violate the independence of factors assumption inherent in Naive Bayes.

[5]Porter: http://tartarus.org/martin/PorterStemmer/def.txt

4.2 CPD

CPD is a feature selection model, an alternative to simply selecting the top 200 words by document frequency, is a number between -1 and 1 measuring the degree to which a word contributes to differentiating a particular class from another class. Formally, it is the difference in occurences of the word in spam and nonspam documents divided by the total number of occurences of the word. A CPD near -1 indicates the word occurs equally across both classes.[6] A CPD of 1 indiciates the word occurs only in documents of one class. The 200 words with the highest CPD (instead of document frequency) are chosen. That is, the 200 words that contribute most towards differentiating spam from nonspam are chosen. The results from stratified 10-fold cross validation using CPD feature selection and CPD feature selection with stemming are shown below.

Corpus	Accuracy[%]	With CPD[%]	With CPD and stemming[%]
Subject	70	77.5	79
Body	93	94	94

As with stemming, there is a significant increase in accuracy for the subject corpus(70% vs. 77.5%). Choosing more differentiating features would have the biggest impact when the sample has fewer words. The accuracy increase is almost insignificant for the body corpus (93% vs. 94%), where CPD assigns high value to words that occurs rarely, but only in one class and a lower value to words that occur much more often, which may be slightly less differentiating and could be a better choice.
When the two extension strategies are combined, the accuracies improve only marginally, if at all. This is because performing stemming ”combines” words, reducing (or keeping the same, but never increasing) the degree to which the stem word contributes to the differetiating of a class. Therefore any increase in accuracy gained by stemming (which reduces correlation between factors) is at least partially mitigated by poorer factor choice, as stemming alters the differentiating amount of a word.

[6] Simeon: http://129.96.12.107/confpapers/CRPITV87Simeon.pdf

5 Conclusions and Next Steps

The Naive Bayes algorithm was able to classify emails into the two classes spam or nonspam very well with an accuracy of 93% using only the text of the body. Similar results for other classification algorithms, especially 1-Nearest Neighbour and Support Vector Machine suggest the two classes are linearly separate in the 200-dimensional feature space. The algorith performs much worse on the subject corpus because there is much less text from which to build the model and to to make a prediction.
Further investigation may be made into the effect of varying the number of selected features and varying the number of emails used for training and validation on the prediction accuracy. We did not synthesise the estimates for body and subject to produce an overall estimate for the whole email. Combining estimates from the body and subject and assigning weights to each (the body should have a higher weight as it is much more accurate) would form a more accurate classifier if the estimates are uncorrelated.

6 Reflection

Over the course of this assignment, I was able to solve(predict with a high degree of accuracy) a deceptively simple real-world problem through the application of machine learning algorithms. I was impressed how the algorithms I implemented (Naive Bayes, stemming and CPD) can easily be adapted to make predictions for any text classification problem without any domain-specific knowledge. Even more generally, Naive Bayes can be used on any sort of regression problem involving any number of explanatory variables; numeric, categorical or some combination thereof (though some variable selection and transformation must be performed in some cases to ensure factors are roughly orthogonal).