Classification of Corporate and Public Text

Kevin Nguyen

December 16, 2011

1 Introduction

In this project we try to tackle the problem of classifying a body of text as a corporate message (text usually sent to a private, select audience, usually sent in a company setting and often related to company information) and public messages (text that is more open and can be broadcasted to a larger audience). Corporate messages usually contain sensitive, private data that would put an organization or individual at risk if leaked. This includes messages about financial trades, company deals, and business meetings. Public messages, however, are less secretive and more casual by nature. This problem is very similar to Data Loss Prevention, a security issue that involves systems that identify, monitor and protect confidential data from leakage. While information about these systems is also confidential, the general industry techniques involve regular expressions, keywords, and hashing. Regular expressions are used to match data such as social security numbers, telephone numbers, and addresses. Keyword matching is used to identify a few words that are marked as private. And hashing works by hashing the substrings of private documents and classifies a new document as private if it contains a substring with a matching hash.

Our problem is similar to DLP, but given our data set it would be erroneous to consider it as DLP. Instead of looking for confidential information, we look to see whether it would be in a corporate message. Still, this work can help shed light on DLP, perhaps by improving the keywords used in DLP techniques. In or comparison of text classifiiers, we used Naive Bayes, Logistic Regression, and Support Vector Machine classifiers and found that SVMs showed consistently better results. However, noticing that the corporate and public messages were centered around certain topics, we used LDA to improve our logistic regression model, which has probablistic inclinations. While we found that with LDA logistic regression results improved, they were still slightly below SVMs.

1.1 Data Set

Data was hard to gather for this project, which explains why this work cannot be extrapolated to DLP or confidential text classification. Finding corporate emails, usually private, was indeed a tall order. The Enron data set was used, but since it also had personal and non-enterprise messages and also was not labeled as corporate private (company related messages that cannot be leaked) or corporate public (company related messages that can be leaked), we can only label and consider it as corporate text. We found a clean data set and attempted to run the classifiers on all of the enron corpus. However, seeing that this would be infeasible, we randomly selected a tenth of the emails, giving us 22,145 emails. From the public sphere, we gathered 6,293 twitter messages, 5,523 myspace forum discussions, and 8,012 slashdot comments on news-posts, which are labeled as public text.

1.2 Data Preprocessing

Before releasing the data to the classifiers, the data was pre-processed for better classification. Stop words, words which are filtered out because they are so common that they are not informative, were removed. A C implementation of the Porter Stemming Algorithm written and maintained by Martin

Porter was used to stem the words to their root form was then applied. At first, we attempted to run the classifiers on the complete vocabulary of 260,000 words. For computation reasons, however, the top 5000 words were selected and used to transform the documents into a vector representation where each position in a document's vector was a binary value that represented the presence or absence of a word.

2 Classifiers

We decided to train Naive Bayes, Logistic Regression and Support Vector Machine models, all known for text classification. Our results were consistent with observations of other works. We did not have any off-the-shelf industry DLP software for comparison.

2.1 Naive Bayes

Naive Bayes is a very strong candidate for text classification, and has shown good results in spam filtering. Naive Bayes is a probablistic classifier within the class of generative models and tries to model $p(x|y)$. It assumes that words are independent given the document's class (which is not true in reality), so the joint distribution of a document can be written as $p(x|y) = p(x_1|y)p(x_2|y, x_1) \dots p(x_n|y, x_1, \dots, x_{n-1}) = \prod p(x_i|y)$ And so to make a prediction, we simply calculate $p(y = 1|x) = \frac{p(x|y=1)p(y=1)}{p(x)}$.

Naive Bayes is fast and highly scalable, and works well with a small training set. In our Binomial Naive Bayes with Laplace smoothing implementation and with our fairly large data set, Naive Bayes performed well but poorly in comparison to the other classifiers. This is not surprising since generative models generally have lower effectiveness than discriminative techniques. Also, this can be explained by dependence of words in the data set. For example, the words "general" and "electric" alone may not be good indicators of a corporate email, but next to each other they constitute the name of a company.

2.2 Logistic Regression

In contrast to Naive Bayes, which models the input patterns, Logistic Regression is a discriminative model that is used to models the decision boundary with $p(y|x)$ directly. This is done using the logistic sigmoid function

$\frac{1}{1+exp(-\theta^t x)}$

which is always a value between 0 and 1. In our tests, logistic regression performs better than Naive Bayes, as expected. This comes at the expense of computation time, however.

2.3 Support Vector Machines

The best classifier is SVM. SVm is a non-probablistic linear classifier that constructs a hyperplane in high or infinite dimensional space. Using LIBSVM [1], we were able to get very good results.

3 Results

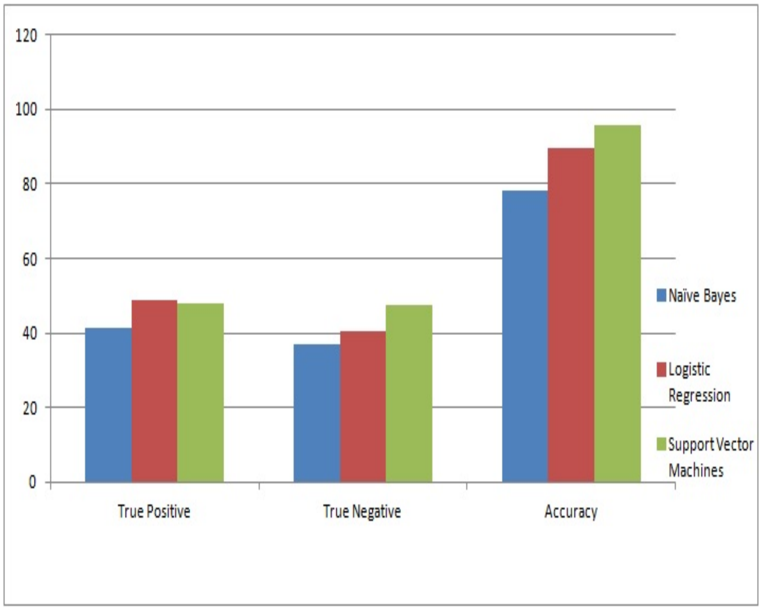
5-fold cross validation was used on each classifier and recorded their accuracy, false positive, true positive, false negative and false positive percentages.

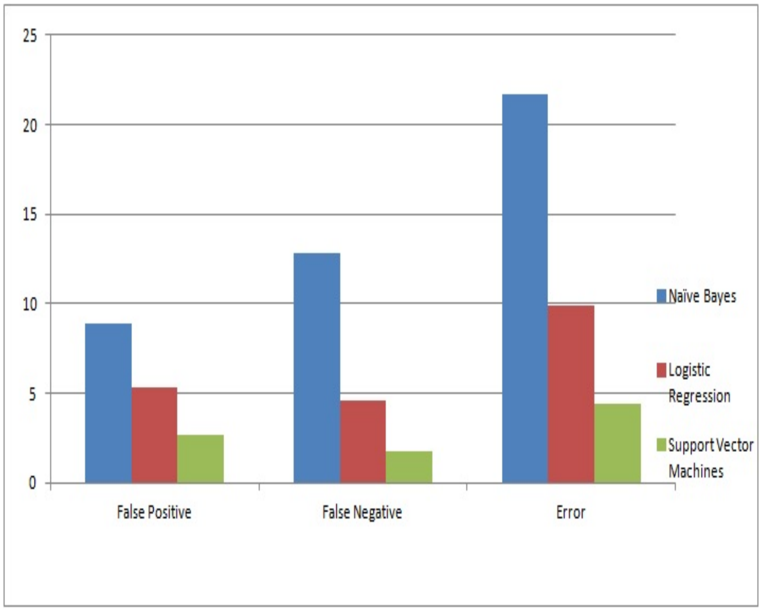
Relative to the other two, Naive Bayes does poorly. If most of the corporate documents were confidential, with a false negative rate of 12.8% (misclassifying a corporate document) the Naive Bayes implementation would not be a good industry classifier. Also, it could be the case that the naive probablistic assumption does not hold in the context of corporate documents as they might contain business "lingo" which could contain multiple words.

Logistic Regression and SVMs do considerably better, especially in terms of their respective false negative rates. These results further substantiate the numerous observations that text catgorization are

	fp	tp	fn	tn	accuracy
Naive Bayes	8.9%	41.2%	12.8%	37.1%	78.3%
Logistic Regression	5.3%	48.9%	4.6%	40.6%	89.5%
Support Vector Machine	2.66%	48%	1.74%	47.6%	95.6%

linearly separable. Also, SVMs might perform well for text categorization because document vectors are sparse (documents contain only a few entries which are not zero), and Kivinen et al. [2] claim that "additive" algorithms, which have a similar inductive bias like SVMs, are well suited for problems with dense concepts and sparce instances.





	Topic 1	Topic 2	Topic 3	Topic 4	Topic 5
Enron Data	enron compani trade busi manag	time imag week free click	pleas messag subject attach agreement	power energi price market california	origin subject mail schedul messag
Myspace, Twittter Slashdot	movi film love look watch	game school plai kid video	people govern time monei system	people obama countri time american	love time christma thank twitter

Table 1: Top 5 words in each topic for Enron and public data

4 Logistic Regression with LDA

After examining some documents in both texts, we noticed that there was little overlap between the topics of corporate messages and public posts. Financial trading, energy, and meetings were often at the core of many corporate emails, while public messages ranged from games to movies to education. Thus, Latent Dirichlet Allocation, a generative model that identifies a collection of topics, is a natural candidate for improving our model. However, since SVMs are non-probabilistic classifiers, they do not lend themselves easily to this technique. Logistic regression, then, was chosen as the baseline in improving classification.

4.1 Latent Dirichlet Allocation

LDA assumes that a word in a document arises from a set of latent topics. Each document, then, is a mix of topics, in contrast to the classical document mixture models which associate each document with a single unknown topic. Each topic is a probability distribution over a finite vocabulary, with topic z receiving weight $\theta_z^{(d)}$ and word w having probability $\phi_w^{(z)}$in topic z. We used GibbsLDA++, a C/C++ implementation of LDA using Gibbs Sampling [3], and we ran 1000 iterations and with $n = 5$ topics over the two corpuses. From the results in Table 1, we can see that the topics of corporate and public text vary drastically.

4.2 Incorporating with Logistic Regression

Normally, Logistic Regression uses the likelihood

$L(\theta) = p(y|X;\theta)=\prod_{i=1}^{m} p(y^{(i)}|x^{(i)};\theta)$ Using LDA with n topics, we ran LDA over the union of both training sets of text and trained n logistic regression models where each model represents a topic and the likelihood becomes

$\prod_{i=1}^{m} p(y^{(i)}|x^{(i)})p(z|x)= \prod_{i=1}^{m} p(y^{(i)}|x^{(i)})\theta_z^{(d)}$ We are essentially weighing our likelihood with $\theta_z^{(d)}$ from LDA. When predicting a document d, we find $\theta_z^{(d)}$ for each topic and add weighted probabilities:

$\sum_{i=1}^{n} p(y|x,z)p(z|x)$. If the probability was above 0.5, we labeled it as corporate.

Table 2: 5-fold Cross Valdiation on Logistic Regression with LDA

n	fp	tp	fn	tn	accuracy
2	4.19%	46%	2.9%	46%	92.89%
5	3.8%	46.6%	2.28%	47.2%	93.86%
10	3.9%	46.7%	2.2%	47.2%	93.83%
15	3.7%	46.6%	2.3%	47.3%	93.9%
20	3.9%	46.5%	2.4%	47.2%	93.6%

4.3 Testing

We ran Logistic Regression with LDA with $\alpha = 50/n$, $\beta = 0.1$, and for 500 iterations. We varied n and used 5-fold cross validation.

Our tests showed that LDA did slightly improve logistic regression, consistently achieving higher testing accuracy and lower testing error for all values of n. The lowest testing accuracy observed was 3% higher than without LDA. However, none of the tests was able to beat the SVM. Although the Enron corpus includes corporate email, it is riddled with personal emails, such as holiday greetics and political discussions. Similarly, there are some messages in the public data that are business related. There is then, some overlap of topics between the two sets.

Also, although the application of LDA on logistic regression does improve accuracy, it isn't clear what values of n are best, as they all perform relatively the same.

5 Conclusion and Future Work

The results from Naive Bayes, Logistic Regression and SVMs were unsurprising. SVM tops the list, which gives further evidence to why they might be the best classifier for text classification. Using LDA to learn topics, we were able to observe a small increase in accuracy for Logistic Regression. However, it was not enough to surprass SVMs. Future work could be made possible with a better data set. For example, some of the topics learned by LDA were very specific to the Enron data set and would not be good for other corporate emails: the word "enron" was a top word in many topics, and in the topics that involved travel, "houston" showed up as a top word, which is also the name of the city in which Enron was located. Also, LDA topics could help identify confidential documents because it could be the case that personal, non-confidential documents in the enron set matched closely to topics in the public corpus and would be correctly marked as non-confidential, but the framing of our problem erred by classifying it as public while it was corporate. With better labels, topics can be used to improve the DLP problem or confidential information (corporate and business) classification.

I'd like to thank the CS229 staff for their help and guidance.

References

[1] Chih-Chung Chang and Chih-Jen Lin. LIBSVM: a library for support vector machines. http://www.csie.ntu.edu.tw/~cjlin/libsvm.

[2] J. kivinen, M. Warmuth, and P. Auer. The Perceptron algorithm vs. winnow: Linear vs. logarithmic mistake bounds when few input variables are relevant. In Conference on computation Learning Theory, 1995.

[3] X.-H. Phan. http://gibbslda.sourceforge.net/.