University of California Davis

Department of Statistics

Machine Learning (STA-208)

Ying-Chen Chou
Chia-Hui Shen
Jiahui Tan
Pei-Ying Ling

Contents

1 Introduction

With the progress of the times, email has been one of the most efficient communication media nowadays. However, with the ubiquitous use of email, useless messages and advertisements spread widely which caused email misuse. To improve this issue, lots of people did the researche for some approaches to construct filters by using machine learning. Most researche mainly focus on adjusting classical methods to make filters more efficient.

To have better understanding, we referred from lots of papers about detection of spam email. Some papers tried to compare different methods and found the best one. However, the results in different papers sometimes are different because most of them did not compare the same methods at the same time. Another reason may come from different data set they used. Moreover, the data set they used typically are experimental and smaller. Therefore, we want to make a overall comparison for the methods they tried and apply those methods on larger data set we combined. We would compare four methods: SVMs, Decision Trees, Naive Bayesian, KNN, Multi Layer Perceptron (MLP), and Logistics by calculating accuracy rate and cost time. In addition, we would use two kinds of data prepossessing (uni-gram and bi-gram tfidf) to increase complexity in our project.

Moreover, our data set contains over than 150,000 email from 1999 to 2007. We supposed that the keyword per year would change because of the innovated technology or social cognition. We would like to discuss keywords in spam email by year to explore the characteristics in each year.

2 Description of Data

2.1 Data Preprocessing

There are only a few public sources for email data to which almost everyone trying to do similar analysis will turn to. For our project, we downloaded datasets from the following three locations:

1. Enron-Spam datasets

2. SpamAssassin data

3. TREC email corpus

We combine all emails and extracted the information including date, from, to, subject, content, number of cc, and number of bcc. In the process of data preprocessing, we faced some challenges to get the information of year, weekday, and hour at which the email was sent.

We tried to use the string methods in python but end up finding regular expression is more powerful to extract the weekday and month. The other challenge of cleaning up the emails come from trying to remove the html and css elements, such that when we tokenize we wouldn't end up with tag elements as the most frequent words. Although we cannot remove all html and css, we did manage to get rid of most of it by the removal of contents between brackets, parathesis, and curly braces as well as words that begin with a period.

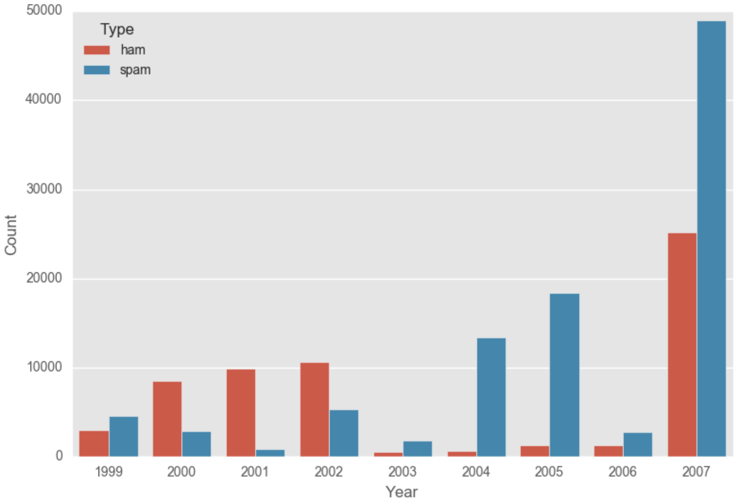


Figure 1: Amount of Email From 1999 to 2007

Year	Ham	Spam	Ham %	Spam %	Total
1999	2978	4611	0.392410	0.607590	7589
2000	8512	2851	0.749098	0.250902	11363
2001	9872	848	0.920896	0.079104	10720
2002	10663	5280	0.668820	0.331180	15943
2003	545	1773	0.235116	0.764884	2318
2004	627	13420	0.044636	0.955364	14047
2005	1309	18418	0.066356	0.933644	19727
2006	1226	2730	0.309909	0.690091	3956
2007	25219	48999	0.339796	0.660204	74218

Table 1: Proportion of Ham Email and Spam Email From 1999 to 2007

After finishing the data cleaning up, we delete the emails with year not between 1999 to 2007 to prevent the situation that the date of the emails is after when the data was collected and that the date of email is so early that the email are still not common used. There are 159981 email with 60951 of them are ham and the other 98930 are spam. From Table 1 and

Figure 1, we can see that there is a disproportional number of emails between each year as well as between spam and ham groups. The imbalance dataset may be worrisome for our classifiers. However, there is not much we can do about this situation. Maybe if we can see that there is not too much variation between ham and spam emails across years or there is no time obvious time effect on the emails, then combining the different years wouldn't be a problem. To see whether there are differences in email across the email, we will examine the top words by year. Figure 2 shown the amount of the email sent each hour and each day. There is a peak for sending spam email at around 12:00 to 15:00. However, ham emails were usually sent between 8:00 to 20:00. Also, according to the right plot of Figure 2, ham emails tend to be sent during weekdays and has lower proportion in weekend but spam email seems to be balance each day.

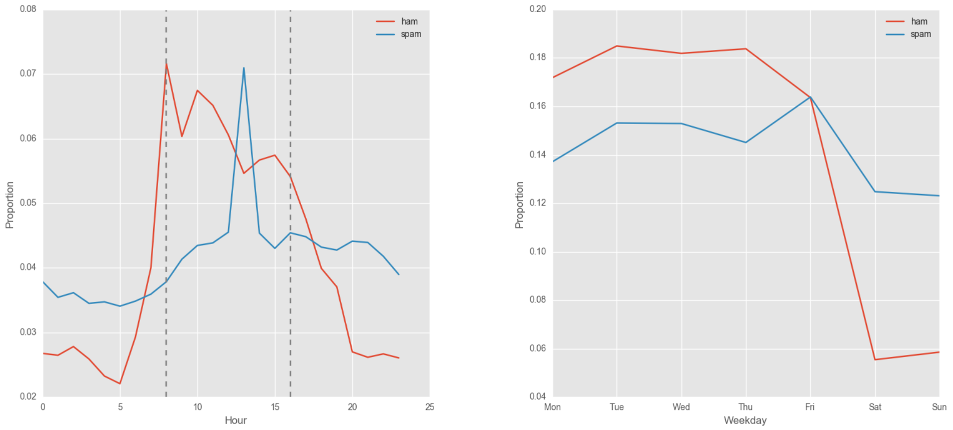


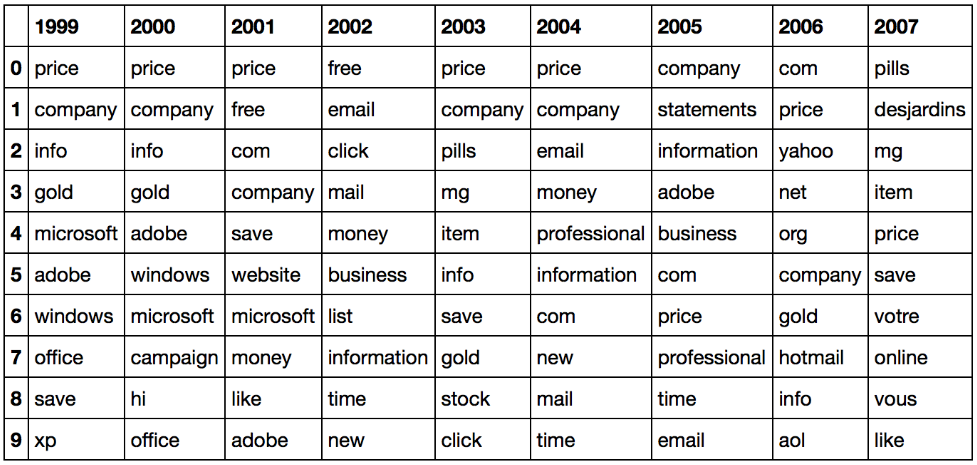
Figure 2: Amount of Email Hourly and On Different Weekday

2.2 Top 10 Words Per Year

One of the main purposes for this project is to explore whether the keywords for spam and ham email changed by year. In this section, we counted the appear frequency of each word as a vector by year respectively. Sort the frequency and find out the top 10 frequent words per year. We would like to explore that whether the frequent words changed by year. Moreover, in the next section, we will use word cloud to visualize the results we found out.

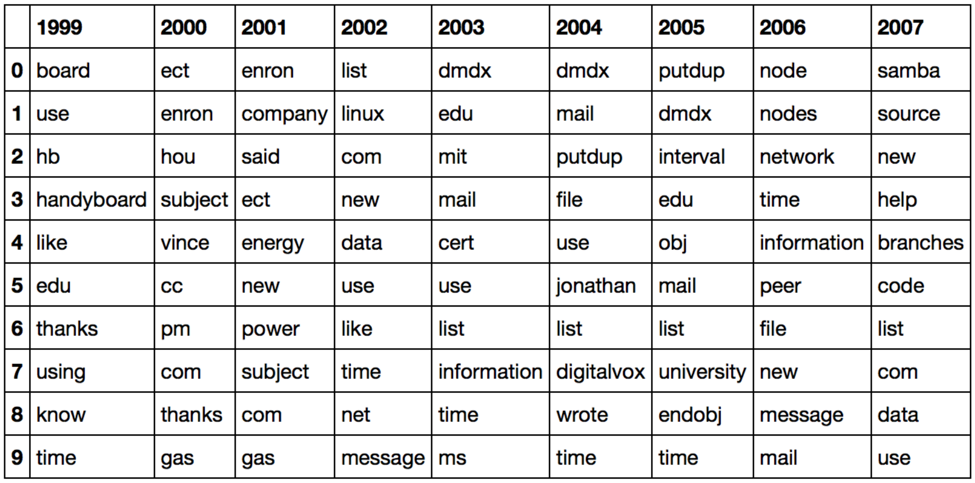
According to the Figure.3, although keywords did not change year by year, we still have found out that there might have a difference in 2002. Before 2001, some keywords appeared repeatedly, such as ”microsoft”, ”adobe”, ”windows”, and ”free”. It seems before 2001, in our data set, the spam emails mostly are related to computer and Microsoft topics. From 2002, ”stock”, ”business”, ”money”, and ”com” become top frequent keywords. We categorize those as economy and Internet topics.

For the ham email (Figure. 4), there is not specific topic for each year. We cannot conclude any specific topic or gap for year. However, overall keywords mainly focus on acadmic, such as ”edu”, ”university”, ”data”. We inferred that ham email data may mostly come from academic organizations.



	1999	2000	2001	2002	2003	2004	2005	2006	2007
0	price	price	price	free	price	price	company	com	pills
1	company	company	free	email	company	company	statements	price	desjardins
2	info	info	com	click	pills	email	information	yahoo	mg
3	gold	gold	company	mail	mg	money	adobe	net	item
4	microsoft	adobe	save	money	item	professional	business	org	price
5	adobe	windows	website	business	info	information	com	company	save
6	windows	microsoft	microsoft	list	save	com	price	gold	votre
7	office	campaign	money	information	gold	new	professional	hotmail	online
8	save	hi	like	time	stock	mail	time	info	vous
9	xp	office	adobe	new	click	time	email	aol	like

Figure 3: Top 10 Frequent Words Of Spam Email



	1999	2000	2001	2002	2003	2004	2005	2006	2007
0	board	ect	enron	list	dmdx	dmdx	putdup	node	samba
1	use	enron	company	linux	edu	mail	dmdx	nodes	source
2	hb	hou	said	com	mit	putdup	interval	network	new
3	handyboard	subject	ect	new	mail	file	edu	time	help
4	like	vince	energy	data	cert	use	obj	information	branches
5	edu	cc	new	use	use	jonathan	mail	peer	code
6	thanks	pm	power	like	list	list	list	file	list
7	using	com	subject	time	information	digitalvox	university	new	com
8	know	thanks	com	net	time	wrote	endobj	message	data
9	time	gas	gas	message	ms	time	time	mail	use

Figure 4: Top 10 Frequent Words Of Ham Email

2.3 Word Cloud

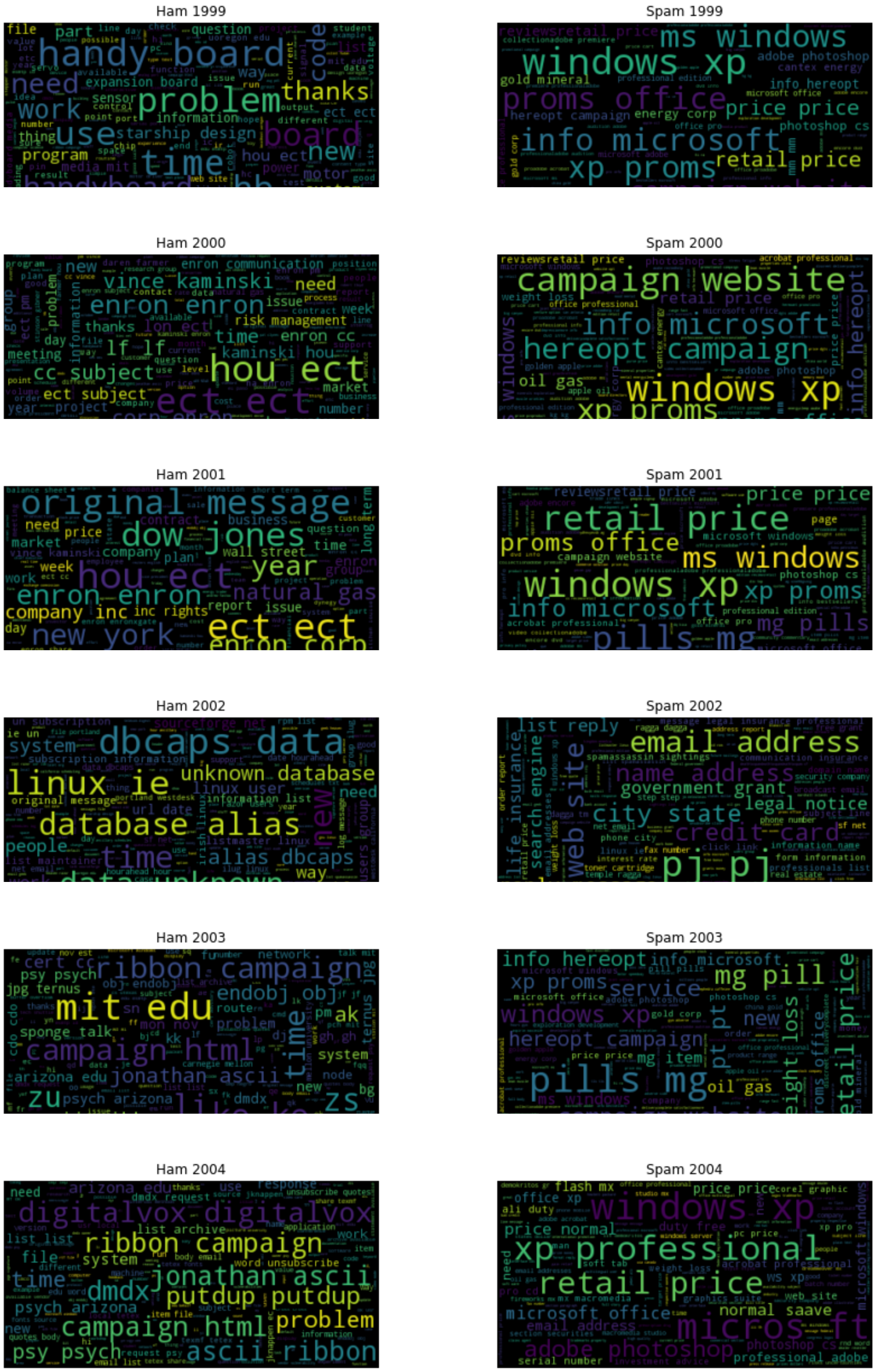


Figure 5: Word Cloud For Spam And Ham Email From 1999 to 2004

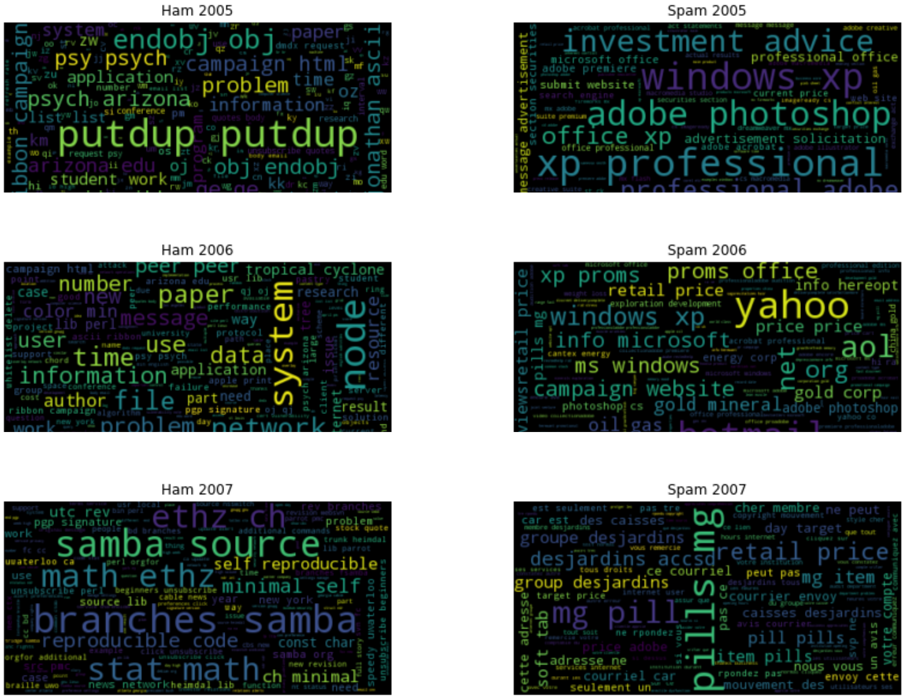


Figure 6: Word Cloud For Spam And Ham Email From 2005 to 2007

From the wordclouds, we see that Microsoft and windows XP seem to be a consistent theme in spam. However, we do see a shift less on windows and more towards sales and products during the later half. For ham, based on the words that show up, we may think that the email data originate from an education or technical source due to terms like edu and systems.

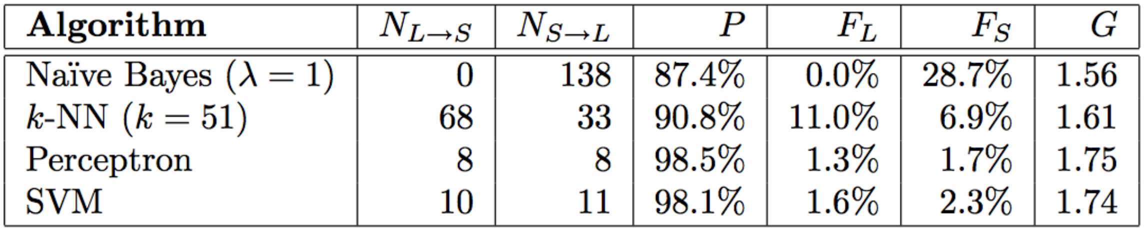
3 Previous Studies

3.1 Purpose

It is important to study what people did and create our study on top of it. In this section, we focus on summarize previous studies of spam/ham e-mail filtering and their machine learning methods. At the end of this section, we point out the new methods and new data we use in this project to show our understanding.

3.2 What We Learn

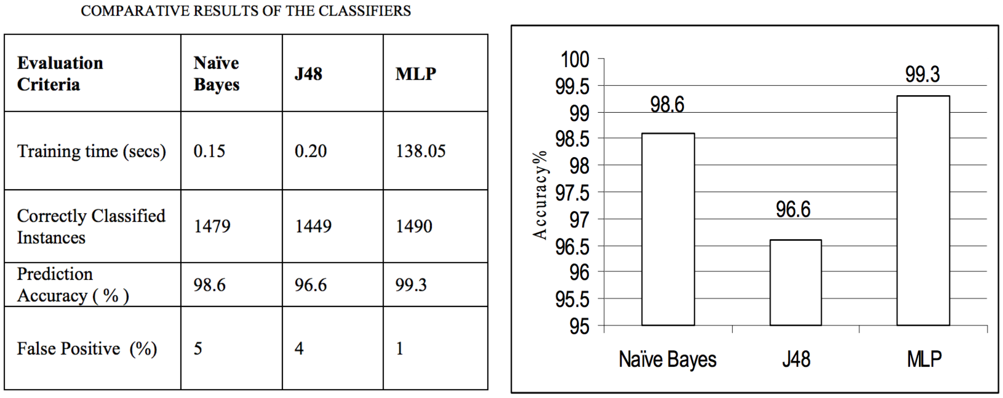
People already came up with the idea of spam/ham e-mail filtering before 2004. In the paper *Machine Learning Techniques in Spam Filtering* written by Konstantin in 2004, the experiment used four main methods : Naive Bayes, K-NN, Perceptron, and SVM. And compare accuracy rate of each methods. In this basic practice, they found the Perceptron method has the highest accuracy rate, 98.5% with a corpus of 1099 messages.



Algorithm	$N_{L\to S}$	$N_{S\to L}$	P	F_L	F_S	G
Naïve Bayes ($\lambda = 1$)	0	138	87.4%	0.0%	28.7%	1.56
k-NN ($k = 51$)	68	33	90.8%	11.0%	6.9%	1.61
Perceptron	8	8	98.5%	1.3%	1.7%	1.75
SVM	10	11	98.1%	1.6%	2.3%	1.74

Figure 7: Summary of *Machine Learning Techniques in Spam Filtering*

In 2010, *Email Spam Filtering using Supervised Machine Learning Techniques* written by V.Christina, they used Naive Bayes, J-48(Decision Tree) and Multilayer Perceptron. And they found out MLP performed the best with 99.3% accuracy rate when experimenting with a corpus of 1500 messages.



COMPARATIVE RESULTS OF THE CLASSIFIERS

Evaluation Criteria	Naïve Bayes	J48	MLP
Training time (secs)	0.15	0.20	138.05
Correctly Classified Instances	1479	1449	1490
Prediction Accuracy (%)	98.6	96.6	99.3
False Positive (%)	5	4	1

Figure 8: Summary of *Email Spam Filtering using Supervised Machine Learning Techniques*

In 2016, *Spam Mail Detection using Classification* written by Parhat and Gambhir used Naive Bayes, SVM and J-48(Decision Tree). And they found out Naive Bayes performed the best with 76% accuracy rate in their experiment.

And *Email Spam Detection* written by Ge and Lauren, used the corpus from TREC 2007 with 1000 messages. They tried logistic regression, Naive Bayesian, Decision Tree and K-NN. The finally found KNN with highest 99% accuracy rate.

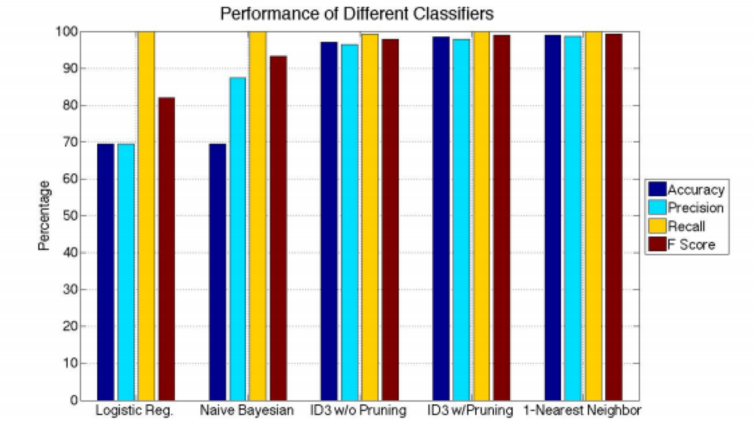


Figure 9: Performance of Different Classifiers

The summary of methods in each previous studies by year is shown in Table 2.

Year	NB	KNN	SVM	Decision Tree	MLP	Logistic	Best Model
2004	V	V	V		V Perceptron		MLP(98.5%)
2010	V			V C4.5	V		MLP(99.3%)
2016	V	V	V	V J48	V	V	KNN(99%)

Table 2: Methods Summary

3.3 Our Works

1. **Use multiple data source**: In each paper, they mainly use a single year of corpus data. In our project, we tried to source different e-mail and integrate them. The format of each data source is different thus hard to clean. And we successfully got to manage a huge data set.

2. **Try 6 methods at the same time**: Previous studies compare accuracy rate with different methods, but they didn't compare them all at a time. So we studied the methods from 2004 to 2016, and apply all of possible methods with adequate tuning parameters to compare them

3. **Apply Uni-Gram and Bi-Gram**: Each paper marked that data processing step is important to a good result. Here, we introduced bag of words of Uni-Gram and Bi-Gram methods in the feature engineering part. And we can see different result of accuracy rate in the following section.

4 Method

4.1 Feature Engineering

First all of, we decoded and encoded the email content to ascii. Then we generalized the notion of word to every basic unit of characters based on whitespace and made all of characters as a list. This process is called tokenization. After creating the list of texts, we remove the stop word to reduce the noise from contents. In addition to create the list of unit characters, we also tried bag of n-grams models for unit-gram and bi-gram to count the frequency of words.

After having previous step, we used tf-idf to give weight of words in each email. Total number of 1-gram bag of words in training data is 1053414. We would use this tf-idf model to fit the testing data.

Because the total number of email data we have is 111916. The number of word features is much larger than the data we have. In this case, the number of features(p) is larger than the number of data set(n). Therefore, we need to do the data reduction.

The method we used for data reduction is called Latent Semantic Analysis (LSA) which is similar with Singular Value Decomposition (SVD). LSA can help us reduce the number of features without losing too much information. LSA also used the consine of the angle between two vectors to form the feature matrix. We reduced our features to only 100 dimensions and used reduced features matrix to fit the models.

In the Figure. 10, we displayed the scatter plot of first two components. We can find out that the spam and ham email have been lightly separated.

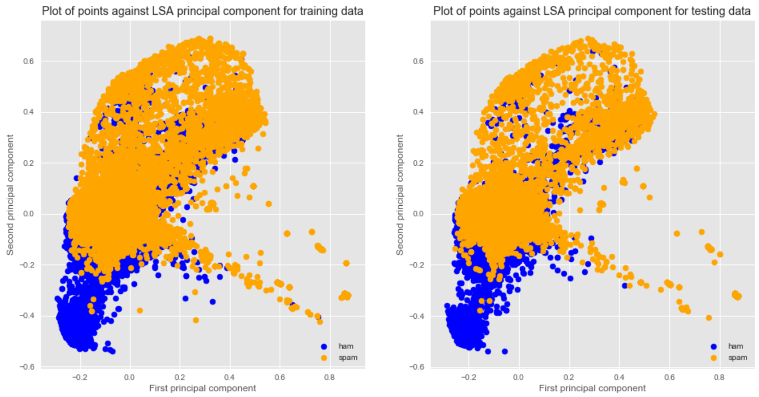


Figure 10: Latent Semantic Analysis Components Plots

4.2 Fit Models

4.2.1 Navie Bayes

Naive Bayes has a very strong assumption that components of the vector x are independent in each class. However, in this data set, we may easily think of the situation that words are related to each other. So in this case, we will not expect Naive Bayes to perform better than other methods. And we found out that the accuracy rate of Naive Bayes has the lowest accuracy rate: 0.7304 for uni-gram and 0.7281 for bi-gram.

Unit-gram		Actual	
		Ham	Spam
Predicted	Ham	15123	9877
	Spam	3054	19919
Bi-gram		Actual	
		Ham	Spam
Predicted	Ham	14905	9624
	Spam	3416	20020

Table 3: Confusion Matrix with Naive Bayes

Unit-gram		Actual	
		Ham	Spam
Predicted	Ham	16837	1167
	Spam	1484	28477
Bi-gram		Actual	
		Ham	Spam
Predicted	Ham	16787	1353
	Spam	1534	28291

Table 4: Confusion Matrix For Decision Tree

4.2.2 Decision Tree

For decision tree, we use cross validation to tune the parameter max_depth which represent the maximum depth of the tree. The candidate of the parameter is the 10, 20, 50, and the default one which is that the nodes are expanded until all leaves are pure or until all leaves contain less than min_samples_split samples. When tuning the model fitting with unit-gram tfidf after data reduction, the average running time for each is around 80 seconds while when max_depth is 10, the running time would be the shortest. The best parameter is max_depth=20 with its corresponding accuracy rate 0.9447. The confusion matrix is shown in Table 4. When the data original is ham email, we will mis-classificated it to be spam email 8.1% of the time. While when the data is a spam email, we will mis-classificated it to be ham email 3.9367% of time. This implies that for a ham email, we will have higher probability to have wrong prediction.

If right now we change to use bi-gram tfidf to fit the models with the same candidate of parameters, the time for fitting a model will be much longer than fitting unit-gram. It's around 1300 seconds to fit a model. The best parameter here is also max_depth = 20. The accuracy rate here is 0.9398, which is smaller than that of one-gram model with max_depth = 20. The confusion matrix is shown in Table 4. The mis-classification rate when the data is ham will be 8.3729% and that for a spam email is 4.5641%. The ham email will also have higher probability to predict wrong.

4.2.3 SVMs

We would like to use two SVM methods: Linear and RBF to do the classification. Because of the size of data, it is not easy to tune cross-validation. Therefore, we decide to use the 3-folder cross validation as our standard then tune the penalty parameter of the error term.The number of penalty parameter we chose for tuning is : 0.001,0.01,0.05,0.1,0.2,0.5,0.7, and 1. After tuning the penalty parameter, the result from test data indicated taht accuracy rate in Linear SVM performs better than RBF SMV. For the uni-gram and bi-gram, Linear SVM has similar accuracy rate. However, RBF SVM has higher accuracy in uni-gram. About the confusion matrix, the prediction of spam/ham email has similar error rate in each method. The weakness of SVM is that tuning the parameters and fitting the model is time consuming. In practice, costing too much time to predict results may increase the cost on industry especially when the data set is large. Therefore, we thought in the SVMs, time will be particularly noteworthy.

Unit-gram		Actual	
		Ham	Spam
Predicted	Ham	16667	944
	Spam	1654	28700
Bi-gram		Actual	
		Ham	Spam
Predicted	Ham	16730	998
	Spam	1591	28646

Table 5: Confusion Matrix with Linear SVM

Uni-gram		Actual	
		Ham	Spam
Predicted	Ham	15303	701
	Spam	3018	28943
Bi-gram		Actual	
		Ham	Spam
Predicted	Ham	14628	658
	Spam	3693	28986

Table 6: Confusion Matrix with RBF SVM

4.2.4 MLP

For the multilayer perceptron model, we tune the hidden layer size. The following tables shows the runtime and accuracy results with respect to each layer for both the unit-gram and bi-gram method. The other model parameters are kept at default with the exception of letting the activation function be logistic.

Based on these results, we get that the best hidden-layer for uni-gram is (50,2) with accuracy of 0.96770561868 and the best hidden-layer for bi-gram is also (50,2) with accuracy of 0.967142708225. In this case, uni-gram does slightly better than bi-gram, but there are also other parameter variations in which bi-gram show improvement over uni-gram. However, whatever improvement there are it appears to be minor. One thing to note is that for bi-gram, there is a speed up in runtime compared to uni-gram. What we observed here is that an MLP with 2 hidden can model our data pretty well. The general consensus with how many hidden layers one should use in any kind of neural net is between one or two. Also increasing the number of hidden units does not guarantee improve performance as we can see in this case with 10 and 50 hidden units.

(a) Uni-Gram

Hidden Layer Size	Runtime	Accuracy
(5,2)	62.582267046	0.956822683207
(10,2)	97.1860368252	0.963827791098
(20,2)	112.540017128	0.947753570312
(50,2)	147.681282043	0.96770561868
(5,3)	73.3039970398	0.958219535078
(10,3)	100.10469985	0.962847909934
(20,3)	114.212602139	0.967497133326
(50,3)	146.349582911	0.949212967789

(b) Bi-Gram

Hidden Layer Size	Runtime	Accuracy
(5,2)	67.9229331017	0.95888668821
(10,2)	95.0826570988	0.963702699885
(20,2)	89.4319069386	0.947461690816
(50,2)	127.49882412	0.967142708225
(5,3)	66.2931339741	0.960346085687
(10,3)	78.5074460506	0.96128426978
(20,3)	114.212602139	0.967017617012
(50,3)	129.54850316	0.948879391223

Table 7: Tuning Results

Uni-gram		Actual	
		Ham	Spam
Predicted	Ham	17371	595
	Spam	954	29045
Bi-gram		Actual	
		Ham	Spam
Predicted	Ham	17359	616
	Spam	960	29030

Table 8: Confusion Matrix For MLP With One-gram And Bi-gram TFIDF

4.2.5 KNN

When fitting KNN models, we tune the parameters with three candidate of K including 3, 5, and 7. The running time is much longer comparing to decision tree. For unit-gram tfidf, the running time for predicting testing data is 457, 441, 494 for each K respectively. After tuning the parameter, the best K for uni-gram tfidf is K=3. The accuracy rate is 0.9657. When using bi-gram tfidf to fit the models, the best K is also 3 with the accuracy rate 0.9678. Using bi-gram in KNN model will improve our prediciton. Table 9 showed the confusion matrix of KNN model with both uni-gram and bi-gram tfidf. Comparing the mis-classification rate for each class, when the observations are ham email, we will have high probability to have wrong prediction.

Unit-gram		Actual	
		Ham	Spam
Predicted	Ham	17331	651
	Spam	990	28993
Bi-gram		Actual	
		Ham	Spam
Predicted	Ham	17405	626
	Spam	916	29018

Table 9: Confusion Matrix For KNN

Unit-gram		Actual	
		Ham	Spam
Predicted	Ham	16343	1375
	Spam	1834	28413
Bi-gram		Actual	
		Ham	Spam
Predicted	Ham	16705	1336
	Spam	1616	28308

Table 10: Confusion Matrix For Logistic

4.2.6 Logistic

Here we first train the logistic parameters. And we found in 1-gram we got the maximum accuracy rate with C = 7 (or 8) and 2-gram we accuracy rate converge when C goes larger so we set C= 199. By running approximately 1000 secs, we got accuracy rate of 0.9331 for uni-gram and 0.9384 for bi-gram. This is a method with decent running time (cost) and good accuracy rate.

5 Conclusion and exploration

From our exploratory data analysis with the word clouds and top-10 words by year, we notice that there is a transition year for spam by the focus away from Windows XP and Microsoft toward more business/monetary associated terms. From the word clouds, we can also see that after 2002, we have a greater variety of terms appearing. No obvious pattern can be identified in the ham emails, besides the consistent occurrence of the term edu which leads us to suspect that our email corpus may come from an educational setting.

For the model comparisons, applying bi-gram does not improve our accuracy and more time consuming. Naive Bayesian has the lowest accuracy rate but latest time spending. The reason of lowest accuracy rate may come from the assumption of feature independence. SVMs spend longest time to fit the models which are not practical if data set is large. The accuracy rate at KNN and MLP are both relatively high but MLP is more efficient than KNN. Those result correspond to our our previous studies Table 2. Among of all, if we needs to balance the time and accuracy, we can find out that logistic model provides us over than 93 percent accuracy rate but only cost less than 1 minute.

Model	Logistic	Naive Bayes	KNN	Decision Tree	MLP	Linear SVM	rbf SVM
Accuracy	0.93309	0.7304	0.9658	0.9447	0.9677	0.9458	0.92246
Time (sec)	17	0.64	457	81.1206	147.6813	6504	4024

Table 11: Summary of Models Using Uni-Gram

Model	Logistic	Naive Bayes	KNN	Decision Tree	MLP	Linear SVM	rbf SVM
Accuracy	0.9384	0.7281	0.9678	0.9398	0.9671	0.9460	0.9093
Time (sec)	30	0.55	1107	1220.9498	127.4988	16802	144056

Table 12: Summary of Models Using Bi-Gram

6 Reference